مقرر لغات برمجة متقدمة

www.facebook.com/groups/Applied.Mathematics.Tishreen.University/

منشورات جامعة تشرين
كلية العلوم

مقرر لغات برمجة متقدمة

لطلاب السنة الثالثة رياضيات تطبيقية

الدكتور محمود شاهين عثمان
أستاذ في قسم الرياضيات

مديرية الكتب والمطبوعات الجامعية

1437-1436هـ
2015-2014م

www.facebook.com/groups/Applied.Mathematics.Tishreen.University/

فهرس المحتويات

16	**الفصل الأول** **أساسيات بلغة C++**
16	1-1- مقدّمة حول Turb C++
16	1-2 - تفحص برنامج c++ الأول
20	1-3- مراجعة إيعازات البرنامج
21	1-4- الاهتمام بالتفاصيل
21	1-5- العمل ضمن بيئة التطوير TC++
22	1-6- ابجدية لغة C++
22	1-7- البيانات الأساسية في اللغة C++
25	1-8 الثوابت
26	1-9- المتغيرات
27	1-10-العمليات الحسابية
28	1-11- الدوال الرياضية
30	1-12- أمر الطباعة cout
31	1-13- أمر القراءة cin
36	1-14- تمارين محلولة 1
40	تمارين 1

42	**الفصل الثاني** **البنى الشرطية واستخدام حلقات التكرار في اللغة++ C**
42	2-1-البنى الشرطية
42	2-1-1 بنية if
43	2-1-2- بنية if....else
44	2-1-3-بنية switch
48	2-1-4-المعاملات المنطقية
48	2-1-5-المعامل الشرطي ?
49	2-1-6-معامل الفاصلة
50	2-2-استخدام حلقات التكرار
50	2-2-1- حلقة for
51	2-2-2- حلقة while
52	2-2-3- حلقة do ....while
53	2-2-4 حلقات التكرار المتداخلة
55	2-2-5 التحكم بالحلقات
58	2-3 تمارين محلولة 2
70	تمارين 2

72	**الفصل الثالث** **الإجراءات والدوال الرياضية وتوليد الأعداد العشوائية**
72	3-1-الإجراءات
72	3-1-1 تعريف الإجراء
73	3-1-2 الإعلان عن الإجراءات
74	3-1-3 استدعاء الإجراء
74	3-1-4-تمرير البيانات للإجراء
75	3-1-5 إجراء يستقبل قيمة من البرنامج ويعطي قيمة إلى البرنامج
80	3 -1- 6-الإجراءات السطرية
81	3-1-7-التحميل الزائد للإجراء
81	3-1-8-قوالب الإجراءات
83	3-1-9- الإعادة الذاتية للاجراء
85	3-2 الدوال الرياضية
86	3-3-توليد الأرقام العشوائية
86	3-3-1 وظيفة randomize()
87	3-3-2 وظيفة random(n)
90	3-4-تمارين محلولة 3
97	تمارين 3

99	**الفصل الرابع** **استخدام المصفوفات في اللغة c++**
99	4-1- تعريف المصفوفة
99	4-2-المصفوفة الأحادية
100	4-3-تهيئة المصفوفة عند الإعلان عنها
103	4-4-الإعلان عن السلاسل المحرفية
104	4-4-1-بعض الوظائف الهامة التي يمكن تطبيقها على السلاسل المحرفية
111	4-5-المصفوفات متعددة الأبعاد
113	4-5-1- مصفوفة السلاسل
128	4-6 تمرير المصفوفة إلى الإجراءات
133	4-7 تمارين محلولة 4
148	تمارين 4
152	**الفصل الخامس** **السجلات وفهم المؤشرات**
152	5-1- السجلات
152	5-1-1-تعريف السجل
152	5-1-2-الإعلان عن السجل
153	5-1-3-استخدام السجل

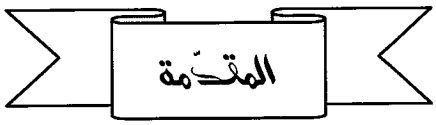
5-1-4- مصفوفة السجلات	155
5-1-5- وضع المدخلات ضمن السجل	158
5-2- الاتحاد	161
5-3- فهم المؤشرات	162
5-3-1-تمهيد	162
5-3-2-تعريف المؤشر	162
5 -3-3- معامل العنوان(&) ومعامل المحتوى(*)	162
5-3-4- المرجع	164
5 -3-5- الإعلان عن المؤشر	166
5-3-6- تهيئة المؤشر	167
5-3-7-استخدام المؤشر	168
5-3-8 -تعابير المؤشرات	171
5-3-9-المساواة الإسنادية للمؤشرات	172
5-3-10- العمليات الحسابية على المؤشرات	172
5-3-11- مقارنة المؤشرات	174
5-3-12- استدعاء الإجراءات بالمؤشرات	175
5-3-13- العودة بأكثر من قيمة من الإجراء الفرعي	180
5-3-14-العلاقة بين المؤشرات والمصفوفات	182

5-3-15-مصفوفة المؤشرات	185
5-3-16- المؤشرات والسجلات	186
5-4- الحجز والغاء الحجز للذاكرة	189
5-5- تمارين محلولة 5	191
تمارين 5	200
الفصل السادس **الأصناف والأغراض**	203
6-1-الصنف(الفئة)	203
6-2- العلاقة بين السجلات والأصناف	208
6-3- الإجراءات الصديقة	209
6-4-الأصناف الصديقة	213
6-5- الإجراءات السطرية والأصناف	214
6-6- تعريف الإجراءات السطرية ضمن الصنف	215
6-7- إجراءات البناء والهدم	216
6-7-1 إجراء البناء	216
6-7-2 إجراء الهدم	218
6-8- أعضاء الصنف الساكنة	220
6-8-1-البيانات الساكنة	220

6-8-2-إجراءات الأعضاء الساكنة	222
6-9- التحميل الزائد للإجراءات البانية	223
6-10- المؤشر This	224
6-11 تمارين محلولة 6	226
تمارين 6	235
الفصل السابع **الوراثة**	237
7-1- تمهيد	237
7-2- التصريح عن الصنف المشتق	238
7-3- الوراثة المحمية	241
7-4- الوراثة والأعضاء المحمية	242
7-5- الصنف المشتق من صنف أخر مشتق	244
7-6- الوراثة المتعددة	246
7-8-إجراءات البناء والهد م والوراثة	247
7-9-تمارين محلولة 7	251
تمارين 7	253

258	**الثامن** **مجاري الإدخال والإخراج في اللغة C++**
258	8-1-تمهيد
258	8-2- المجاري
258	8-2-1-أعلام التنسيق
260	8-3-المناورات
263	8-4-إجراءات مجاري الإدخال والإخراج
264	8-5-الصنف Istream
265	8-6-الصنف Ostream
269	8-7 تمارين محلولة 8
274	تمارين 8
275	**الفصل التاسع** **معالجة الملفات**
275	9-1-تمهيد

9-2- الملفات ومجاري الإدخال والإخراج	276
9-3- كيف تفتح ملفا" نصيا" وتغلقه	278
9-4- القراءة من الملفت النصية	278
9-5- الكتابة إلى ملف سجل بسجل	281
9-6- الملفات الثنائية	286
9-7- القراءة من الملفات الثنائية	287
9-8- ملفات الوصول العشوائي	291
9-8-1 إنشاء ملف وصول عشوائي	291
9-9- تمارين محلولة 9	197
تمارين 9	305
المكتبات في اللغة C	306
المراجع	313
المصطلحات العلمية	314



بسم الله الرحمن الرحيم

المقدّمة

تُعدّ لغة C من أشهر لغات البرمجة في العالم اليوم،إذ إن معظم البرامج التجارية المشهورة بما فيها أنظمة التشغيل مكتوبة بلغة C .

ظهرت لغة C في منتصف السبعينات على يد دينيس ريتشي Dennis Richie في معامل بل ,إذ تأثرت بلغة تدعى B التي تأثرت بدورها بلغة تدعى BCPL ،وقد قام دينيس بتنظيم لغة C تحت نظام التشغيل UNIX وفي عام 1978 قام دينيس ريتشي وبرايان كرينجان بتأليف كتابهما الشهيرThe C Programming Language الذي يُعدّ الكتاب الأم والمرجع الأساسي لهذه اللغة.

وبعد ظهور لغة C بفترة قصيرة انتشرت هذه اللغة انتشارا واسعا وأصبحت اللغة المفضلة لدى معظم المبرمجين.

بعدها تم تطوير لغة C++ من قبل Bjarne Stroustuf في بداية الثمانينات من القرن العشرين وكان من أهم هذه التطويرات على لغة C البرمجة غرضيه التوجيه والتي تعطي برمجيات سريعة واقتصادية وسهلة الفهم وقابلة للتعديل والتصحيح بسهولة.

تغطي مواضيع هذا الكتاب مقرر لغات برمجة متقدّمة لطلاب السنة الثالثة فرع الرياضيات التطبيقية وحرصنا أن يكون _ في الوقت ذاته _مرجعاً واضحاً ونافعا للمهندسين والمهتمين بالرياضيات النظرية و التطبيقية جميعهم.

تم إعداد هذا الكتاب وفق خطة مؤلفة من تسعة فصول يمكن إدراجها وفق الآتي:

يتطرق الفصل الأول إلى أساسيات بلغة c++ مثل تعريف الثوابت والمتغيرات وكيفية الإعلان عنها، كما تناول العمليات الحسابية ، وأمري الطباعة والقراءة.

الفصل الثاني يتناول البنى الشرطية واستخدام حلقات التكرار في اللغة ++ C.

الفصل الثالث: يعالج الإجراءات وطريقة الإعلان عنها ، كما تطرق هذا الفصل أيضا إلى توليد الأرقام العشوائية.

الفصل الرابع: يخصص لاستخدام المصفوفات في اللغة++ C وطريقة الإعلان عنها وكيفية الإدخال والإخراج لهذه المصفوفات ، وقد تم استخدامها لحل جملة معادلات خطية.

الفصل الخامس: يقدم السجلات وطرقة الإعلان عنها ، وتمت دراسة فهم المؤشرات وكيفية التعامل معها .

الفصل السادس: يتناول الأصناف وكيفية التعامل معها كما تمت دراسة إجراءات البناء والهدم .

الفصل السابع : يقدم الوراثة والتي هي أهم مفاهيم البرمجة غرضية التوجيه.

الفصل الثامن: يدرس مجاري الإدخال والإخراج وكيفية تنسيق المخرجات.

الفصل التاسع : وهو الأخير يتضمن معالجة الملفات وكيفية الكتابة والقراءة من الملفات النصية والملفات الثنائية.

آملا" أن أكون قد وفقت في تقديم المفيدوالنافع لطلابنا الأعزاء وخدمة لهذا الوطن الغالي والله ولي التوفيق.

المؤلف

www.facebook.com/groups/Applied.Mathematics.Tishreen.University/

الفصل الأول

أساسيات بلغة C++

1-1- مقدمة حول Turb C++

Turb C++ هي مجموعة أدوات تساعدك على جعل اللغة c++ أكثر سهولة فيما يخص المبتدئين بهذه اللغة وتشمل هذه المجموعة:

1- عناصر اللغة c++ جميعها والتي تتألف من أوامر التشغيل والعمليات والوظائف ،التي تحتاجها من أجل إنشاء برامجك الخاصة بوساطة هذه اللغة.

2- بيئة تطوير البرامج للمجموعة TC++ وهي مجموعة الوسائل العملية التي تساعدك على تصميم وتنظيم وتنقيح وإكمال مشروع برمجة معين.

ومن أجل إنشاء برامج ناجحة بوساطة اللغة TC++ يجب أن نلم باللغة C++ وببيئة التطوير ويقدم لنا TC++ الأدوات للقيام بالأمور الأتية:

- طباعة وتحرير نص البرنامج كملف نصوص على قرص التسجيل.
- تركيم البرنامج أي ترجمته إلى ترميز يستطيع حاسوبك تنفيذه.
- وصل الترميز المركم مع وسائل برامج الحاسوب الأخرى الضرورية لإنشاء ملف مستقل وكامل.

إن اللغة c++ هي لغة تراكمية (compiled language) يصف هذا التعبير طريقة ترجمةTc++ برامجك إلى إيعازات يستطيع حاسوبك تنفيذها فعليا.

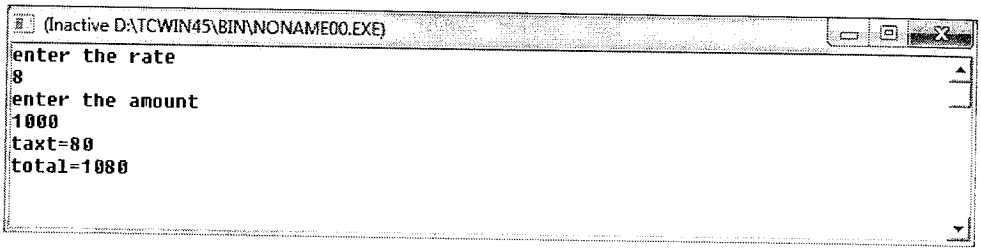
وقد يكون الجانب الأكثر أهميةفي عملية التركيم متمثلا" في النتيجة النهائية عندما تقوم بتركيم ووصل برنامج ما ينتج TC++ ملفا يحمل الامتداد exe. وتستطيع تنفيذ هذا البرنامج من خلال النظام Dos .

ترتبط _غالبا_ اللغة c++ بمشاريع البرمجة الطويلة والمعقدة وعلى سبيل المثال فإن المبرمجين المحترفين يعملون كثيرا مع اللغة c++ لتطوير برامج مثل برامج معالج الكلمات والبرامج الإحصائية.

ولكن TC++ مهيأة بصورة جيدة لتطوير برامج من أي حجم بما في ذلك الأدوات الصغيرة والبسيطة التي تفي باحتياجات مباشرة ومحدودة ولنعرض الآن البرنامج الأول:

```
//program 1-1
#include<iostream.h>
main()
{
   float rate amount,taxt;// the input values
cout << "enter the rate\n";
cin>>rate;
cout<< "enter the amount \n";
cin>>amount;
taxt=amount*rate/100;
cout<< "taxt="<<taxt<<endl;
cout<<"total="<<amount+taxt;}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة لإخراج النتائج كمايأتي:



إن هذا البرنامج على الرغم من بساطته يمثل النمط العام لطريقة العمل التي سوف نراها تتكرر في جميع فصول هذا الكتاب.

إذ إن البرنامج مصمم من أجل أداء أربعة أنواع أساسية من المهام.

1- الدخل(input) يستقبل البرنامج قيم المعطيات من لوحة مفاتيح التشغيل.

2- تخزين المعطيات(data storage) يمثل البرنامج المعطيات في ذاكرة الحاسوب بصورة مناسبة وملائمة في الآن عينه.

3- العمليات(operations) ينفذ البرنامج الحسابات على معطيات الدخل لتحقيق نتائج معينة

4- الخرج(output) يعرض البرنامج المعلومات الجديدة المحتسبة على الشاشة.

ويمكننا الآن القيام بشرح عام لمظهر هذا البرنامج:

يتألف البرنامج من العناصر الآتية:

أ- كتابة الملحوظات داخل نص البرنامج لتوثيق وشرح هدف عمل البرنامج.

ب- استعمال إيعازات التوجيه السابقة لجهاز المعالجة والتي هي إيعازات خاصة إلى Tc++. تتعلق بالعمليات التي تستوجب الأداء على نص البرنامج قبل تركيمه والتي تسمى مكاتب الربط الديناميكية .

ت- انشاء البرنامج حول الإجراء (main وهو جزء مطلوب في أي برنامج .

ويمكننا شرح هذه العناصر بصورة موجزة.

- **توثيق برنامج اللغة C++ باستعمال الملحوظات Comments :**

إن الكتابة والاحتفاظ بنوع التوثيق لأي برنامج ننشئه يمثل أهم المهام للمبرمج ،أي يجب أن يرافق أي برنامج ملحوظات تشرح عمل الترميز وكيفية قيامه بهذا العمل .

وأفضل مكان لوضع مثل هذه الملحوظات هو داخل البرنامج، لذلك توفر لنا اللغة C++ تدوينا خاصا" يستعمل لتحديد الأسطر المعينة للنص وهي ملحوظات وليست أسطر إيعازات فعلية ،ويستعمل هذا التدوين الخطين المائلين // كمايأتي:

يبدأ أي سطر من الملحوظات بالحرفين // مثل:

```
// this is a comment
```

وسوف يتجاهل المترجم كل مايرد بعد // حتى نهاية السطر

هناك نوع آخر من التعليقات يتيح لنا كتابة تعليقات تمتد إلى عدة أسطر multi-line comments ، نستطيع كتابة التعليق على الصورة:

```
/* Program 1-1
 This program will display a message on the screen
*/
```

نبدأ بالرمز */ ثم كتابة التعليق وننهيه بالرمز /* . نجد أن نهاية السطر لا تعنى انتهاء التعليق ،لذا يمكننا كتابة ما نشاء من أسطر التعليقات قبل الانتهاء بالرمز /*.

-إيعاز التوجيه #include

يوجد في بداية البرنامج بيان يبدأ بالمحرف # تسمى مثل هذه البيانات ملفات ترويسة (header files) أو مكاتب الربط الديناميكية وهذه البيانات تظهر عند الجزء العلوي في معظم برامج اللغة++ c وهي جزء من مجموعة اللغة++ c .

يقوم أمر التشغيل #include بقراءة الملف iostream.h والذي يتضمن الوظائف كافة التي تتعامل مع الإدخال من لوحة المفاتيح والإخراج على وحدة الخرج الرئيسية للشاشة .

وهناك كثير من ملفات الترويسة الأخرى، وعلى سبيل المثال إذا كنا نستعمل في برنامجنا دالات رياضية كـ ( )sin و( )cos نحتاج إلى شمل ملف ترويسة يدعى math.h، وإذا كنا نتعامل مع سلاسل الأحرف سنحتاج للملف string.h ، وعموماً هناك عدد كبير من ملفات الترويسات التي يجب تضمينها وفق طبيعة البرنامج ، تُعدّ ملفات الترويسات جزءا" مهما" من برامج لغة C++ .

- **كتابة الإجراء ()main**

يتم تنظيم جميع برامج اللغة c++ ضمن وحدات بنيوية تدعى إجراءات ونستطيع تجزئة مشروع برمجة كبير إلى عدة إجراءات منفردة يقوم كلا منها بتنفيذ جزء محدد ومحصور من المهمة الإجمالية المطلوبة . ويكون كل إجراء وحدة مستقلة بذاتها ولكن الإجراءات تستطيع الاتصال مع بعضها بعض بإرسال بنود المعلومات واستلامها عند استعمالها ويتم استعمال الإجراء أو استدعائه باستعمال اسمه وسوف نشرح استعمال الإجراءات بالتفصيل في الفصول القادمة.

كما أنه ينبغي على أي برنامج بلغة c++ أن يحتوي على إجراء واحد على الأقل يدعى ()main والدور النموذجي لهذا الإجراء هو تحديد عمل البرنامج بالاتصال مع

الإجراءات المتوافرة الأخرى ، ولتصميم أي برنامج يجب وضع عمل البرنامج بكامله داخل الإجراء ()main باستثناء إيعازات التوجيه السابقة لجهاز المعالجة ، وذلك بإدراجه ضمن قوسين كبيرين { و }.

3-17- تراجمة ايعازات البرنامج

يوضح لنا البرنامج السابق استعمال ثلاثة أنواع أساسية من الأدوات المتوفرة في اللغة C++ وهي الكلمات الأساسية والعمليات والوظائف المكتبية ولنتطرق على هذه النقاط بصورة موجزة.

- **الكلمات الأساسية في اللغة++ C**

يتألف قلب اللغة C++ من مجموعة من الكلمات الأساسية يتجاوز عددها 36 كلمة وهي تمثل العناصر الأساسية للغة وسوف نستعمل بعضا من هذه الكلمات الأساسية لإجراء مهام أساسية في برنامج اللغة C++ وأهم هذه الكلمات:

auto	break	Case	char	const	continue
default	do	Double	else	enum	extern
float	for	Goto	if	int	long
register	return	Short	signed	sizeof	static
struct	swietch	Typedef	union	unsigned	void
volatile	while				

ولقد لحظنا في البرنامج السابق تعليمة لتحدد المتغيرين reat و amount لتمثيل قيمة الدخل للبرنامج والكلمة الأساسية في هذه التعليمة float .

```
float  reat,amount;
```

والكلمة الأساسية float تحدد بأن المتغيرين يمثلان قيما بنقطة عائمة أي أعداد قد تحتوي على أرقاما" عشرية قبل الفاصلة وبعدها.

• **العمليات في اللغة ++C**

العمليات هي الأعمال التي تؤديها على قيم المعطيات ضمن البرنامج، ولعل أكثر هذه العمليات استعمالا هي العمليات الحسابية مثل الجمع والطرح والضرب والتقسيم والتي تمثل ب+ و - و * و / على الترتيب.

وظائف مكتبة برامج اللغة++ c

يشمل++ c على مجموعة كبيرة من أدوات البرمجة تدعى وظائف مكتبية وهي جزء أساسي من اللغة++ c ولقد لحظنا في البرنامج السابق استدعائين لوظيفتين هما cin و cout وهما اثنين من أكثر وظائف الدخل والخرج استعمالا.

ترسل الوظيفة cout المعلومات المنسقة التشكيل على شاشة العرض وتقرأ الوظيفة cin بنودا من المعلومات من لوحة مفاتيح التشغيل .

والآن لنشرح بعض التفاصيل الهامة لتنسيق تشكيل برنامج اللغة++ c .

[illegible]

(محارف الأرقام وحالة الأحرف وإزاحة الفقرات)

في أثناء طباعة البرنامج يجب الأخذ بالحسبان:

- تنتهي كل تعليمة في البرنامج بفاصلة منقوطة (;) ويستثنى من ذلك إيعازات التوجيه السابقة لجهاز التحكم.
- التعليمات التي تحدد عمل الإجراء توضع ضمن زوجين من الأهلة { } ونستطيع رؤية ذلك في الإجراء ()main.
- يجب الانتباه بأن اللغة ++c تفرق بين الأحرف الكبيرة والصغيرة .

[illegible] TC++

تحتوي بيئة التطوير ++TC على أربعة مناطق مختلفة للشاشة والتي تخدم أغراضا مختلفة خلال تطوير البرنامج وهذه المناطق هي:

1- نافذة EDIT حيث تدخل وتحرر نص البرنامج.

2- نافذة الرسائل MESSAGE إذ يوضح ++TC الرسائل خلال عملية التركيم.

3- نافذة المراقبة نستطيع استعمالها خلال إزالة العلل من البرنامج.

4- شاشة المستعمل: إذ يظهر خرج شاشة البرنامج.

ولا تظهر هذه المناطق الأربعة جميعها في الوقت نفسه على الشاشة ولكل منها دور مستقل في عملية تطوير البرنامج.

تمر برامج C++ بست مراحل أساسية قبل أن يتم تنفيذها وهي:

الكتابة edit , ماقبل الترجمة preprocess، الترجمة compile، الوصل link ، الشحن load ، والتنفيذ execute .

تهدف المرحلة الأولى إلى كتابة نص البرنامج ويمكن أن يتم ذلك بوساطة محرر النصوص الذي يستخدمه المبرمج لكتابة برنامج بلغة C++ والقيام بالتصحيحات في حال الضرورة, بعدها يتم تخزين البرنامج على وحدات التخزين المعروفة.

يقوم المبرمج بعدها بطلب أمر الترجمة compile للبرنامج ، فيقوم المترجم بترجمة برنامج C++ إلى لغة الآلة والتي تسمى أحيانا (object code) .

2-6 أبجدية لغة C++

تتألف أبجدية لغة c++ من:

- جميع الأرقام 0,1,2,3,4,.....9
- جميع الأحرف الصغيرة a,b,c,.....x,y,z
- جميع الأحرف الكبيرة A,B,C,.....X,Y,Z
- بعض الرموز الخاصة مثل *,+,-,%,&,#
- بعض الكلمات في اللغة الإنكليزية .

2-7 البيانات الأساسية في اللغة C++

توجد هناك سبعة أنواع بيانات أساسية في C++ ، واحد منها يمثل الأحرف وثلاثة تمثل أرقاماً كاملة (أعداد صحيحة) وثلاثة تمثل أرقاماً حقيقية. الجدول الآتي يلخّص هذه الأنواع.

اسم النوع	يستعمل لتخزين	أمثلة عن القيم المخزنة
char	أحرف	"a"
short	أرقام صحيحة قصيرة	222
int	أرقام صحيحة عادية الحجم	153,406
long	أرقام صحيحة طويلة	123,456,789
float	أرقام حقيقية قصيرة	3.7
double	أرقام حقيقية مزدوجة	7.533,039.395
long double	أرقام حقيقية ضخمة	9.176,321.236,01202.6

1/ الأحرف char :

يتم تخزين الأحرف في متغيرات من النوع char العبارة:

char ch;

تُنشأ مساحة من الذاكرة لحرف وتسميه ch. لتخزين حرف ما في هذا المتغير نكتب

ch='z'

ودائماً تكون الأحرف الثابتة كـ 'a' و 'b' محصورة بعلامة اقتباس فردية.

يمكن استعمال المتغيرات من النوع char لتخزين أرقام كاملة بدلاً من أحرف ، فمثلاً يمكننا كتابة:

ch=2;

لكن نطاق القيم الرقمية التي يمكن تخزينها في النوع char يتراوح بين 128- إلى 127، لذا فإن هذه الطريقة تعمل مع الأرقام الصغيرة فقط.

2/ الأعداد الصحيحة:

تمثل الأعداد الصحيحة أرقاماً كاملة أي قيم يمكن تعدادها ، كعدد أشخاص ، أو أيام ،أو عدد صفحات مثلاً ، ولا يمكن أن تكون الأعداد الصحيحة أرقاماً ذات نقطة عشرية ولكنها يمكن أن تكون سالبة.

هناك ثلاثة أنواع من الأعداد الصحيحة في short:C++ قصير، int عدد صحيح، long طويل وهي تستحوز على مساحات مختلفة في الذاكرة. الجدول الآتي يبين هذه الأنواع والمساحة التي تأخذها في الذاكرة ونطاق الأرقام التي يمكن أن تأخذها:

اسم النوع	الحجم	المدى
char	1byte	128- إلى 127
short	2byte	32768- إلى 32767
int	مثل short في أنظمة 16bit ومثل long في أنظمة 32bit	
long	4byte	2147483648- إلى 2147483647

3/ الأرقام العائمة(الحقيقية) (Float):

يتم استعمال الأرقام العائمة لتمثيل قيم يمكن قياسها كالأطوال أو الأوزان. ويتم تمثيل الأرقام العائمة _عادة _ برقم كامل على اليسار مع نقطة عشرية وكسر على اليمين.

هناك ثلاثة أنواع من الأرقام العائمة في أنظمة التشغيل الشائعة الاستعمال. وأشهر نوع لأرقام عائمة هو النوع double والذي يتم استعماله في معظم دالات C++ الرياضية. يتطلب النوع float ذاكرة أقل من النوع double . الجدول الآتي يوضح هذه الأنواع والحجم الذي تأخذه في الذاكرة.

اسم النوع	الحجم
float	4byte
double	8byte

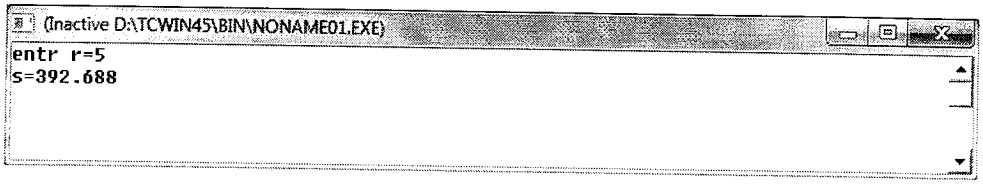
long double	10byte

الثابت هو كمية لاتتغير في أثناء تنفيذ البرنامج ويجب أن يشار إلى الثابت بعد ملفات الترويسة بالتعليمة #define .

ضع برنامجا لحساب حجم الكرة:

```
//program 1-2
#include<iostream.h>
#define pi 3.1415
        main()
        {
        float r,s;
        cout<<"entr r=";
        cin>>r;
        s=4/3*pi*r*r*r;
        cout<<"s="<<s;
        }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:

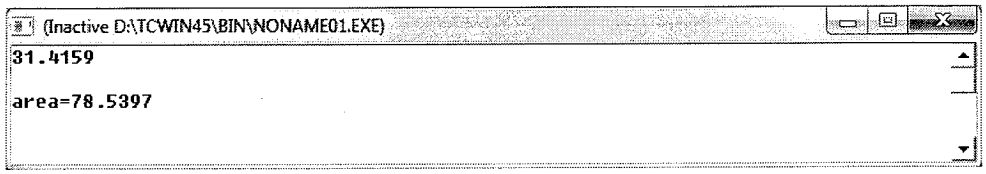


البرنامج االآتي يعرف ثابت "\n" newline أي الانتقال إلى سطر جديد:

```
//program 1-3
#include <iostream.h>
#define PI 3.14159
#define newline "\n"
```

```
int main ()
{
double r=5.0; // radius
double circle,area;
circle = 2 * PI * r;
area  =PI*r*r;
cout << circle<<endl;
cout << newline;
cout<<"area="<<area;
return 0;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كماياتي:



1-9-المتغيرات

المتغير هو كمية تتغير في أثناء تنفيذ البرنامج وفي أثناء استخدام المتغيرات يجب الإعلان عنها، وهذا يعني ان تخبر المترجم عن نوع المتغير وعن اسمه .

1-9-1-أسماء المتغيرات:

عند اختيار اسم المتغير يجب الأخذ بالحسبان النقاط الآتية:

1- يجب أن لا يكون الاسم الذي تختاره للمتغير من الكلمات المحجوزة في اللغة++ C.

2- الحرف الأول للاسم يجب أن يكون حرفا هجائيا أو حرف (_) .

3- لغة c++ تميز بين الأحرف الكبيرة والصغيرة.

4- عدم وجود فراغات ضمنها أو إشارات خاصة.

5- من الأفضل أن نختار اسما ذو معنى.

Int number;

Char t;

Float salary;

Double rate;

وإذا كنت تريد الإعلان عن عدة متغيرات من نفس النوع يمكن أن تضعها جميعا في عبارة واحدة

. int i,j,k;

والمكان المناسب لوضع الإعلان عن المتغيرات في بداية الإجراء الرئيسي ()main .

1-10-العمليات الحسابية:

تتيح لنا لغة C++ إجراء العمليات الحسابية على المتغيرات والثوابت وهناك خمس معاملات حسابية موضحة بالجدول الآتي:

المعامل	استخدامه
+	الجمع مثل : a=b+2
–	الطرح مثل : c=d–4
*	الضرب مثل: s=a*b
/	القسمة مثل: d=s/i
%	المتبقي مثل: t=13%5

T=13%5

يعطي العدد 3 .

1-10-1-عبارات حسابية مختصرة:

توجد في لغة C++ بعض المعاملات التي تختصر العمليات الحسابية مثل:

i=i+1 يمكن التعويض عنها بـ ++i وكذلك i=i–1 ب/ –– i والعبارة s=s+5 بـ s+=5

والطريقة نفسها تنطبق على الطرح والضرب والقسمة مثل:

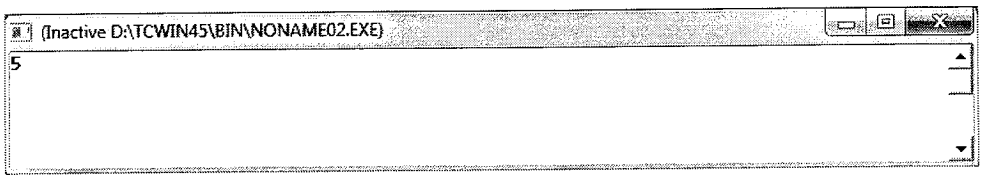
max =max*5 بـ max *=5

a = a / b; ب a /= b;

نوضح هذه العمليات بالبرنامج الآتي:

```
//program 1-4
#include <iostream.h>
int main ()
{
int a, b=3;
a = b;
a+=2; // equivalent to a=a+2
cout << a;
return 0;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:



11-1- الدوال الرياضية:

تسمح لنا اللغة c++ استخدام مجموعة من الدوال الرياضية الشهيرة ومن أجل تحقيق ذلك يجب إدراج ملف الترويسة #include<math.h> في بداية البرنامج وأهم هذه الدوال هي:

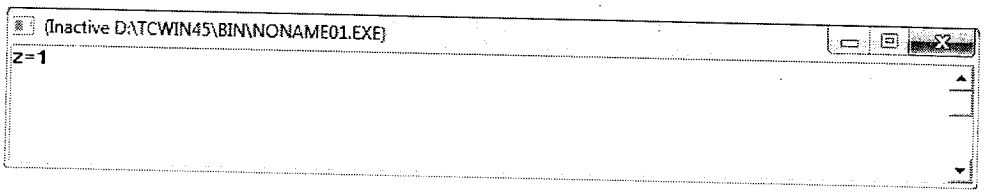
الدالة الرياضية	الدالة في اللغة c++
Sin x	Sin(x)
Cos x	Cos(x)
Tan x	Tan(x)
Arctan x	Atan(x)
Arccos x	Acos(x)
Arcsin x	Asin(x)

e^x	Exp(x)
Ln x	اللوغاريتم الطبيعي Log(**x**)
$Log_{10}(x)$	اللوغاريتم العشري Log10(**x**)
$\sqrt{x}$	Sqrt(**x**)
|x|	عدد صحيح x; abs(x) عدد ذو نقطة عائمة x; fabs(x)
x^y	Pow(x,y)
تقريب x لأصغر عدد صحيح أكبر من x	ceil(x)
تقريب x لأكبر عدد صحيح أصغر من x	floor(x)
يعطي ماتبقى من القسمة للأعداد العائمة	Fmod(x,y)

نوضح استخدام fmod بالبرنامج الآتي:

```
  //program 1-5
#include<iostream.h>
#include<math.h>
main()
{
  float x,y,z;
  x=3;y=2;z=fmod(x,y);
  cout<<"z="<<z;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:



12-أمر الطباعة cout:

يستخدم هذا الأمر لطباعة قيم المتغيرات على الشاشة والشكل العام لهذا الأمر:

cout <<اسم المتغير;

cout<<s;

كما يستخدم لطباعة العبارات الكلامية:

cout<<"enter the number";

الاسم cout والذي يلفظ كـ C out يمثل كائن في C++ مقترن مع الشاشة والعامل >> والذي يسمى بعامل الوضع Put to operator يجبر على إرسال الأشياء التي على يساره إلى الشاشة.

البرنامج الآتي يوضح استخدام cout:

```
//program 1-6
#include <iostream.h>
main ( )
{
      cout  << 15 << " is an integer.\n";
      cout << 'b' << "is a character.\n";
}
```

الخرج للبرنامج:

```
15  is an integer.
b  is a character
```

من خرج البرنامج يتضح لنا الآتي:

1-يتم حصر النص المطلوب ظهوره على الشاشة بين علامتي اقتباس

." is an integer"

2- تتم كتابة الثوابت الرقمية بدون علامتي اقتباس 15>>.

3- يتم حصر حرف واحد مطلوب ظهوره على الشاشة بعلامة اقتباس فردية 'b'>>.

2-1-6-الأحرف غير المطبوعة:

يوضح الجدول الآتي أهم هذه الأحرف:

الحرف	تأثيره
\b	حرف للخلف
\n	سطر جديد
\t	حرف الجدولة tab

2-1-7- أمر القراءة cin:

الشكل العام لهذا الإجراء :

cin>> اسم المتغير;

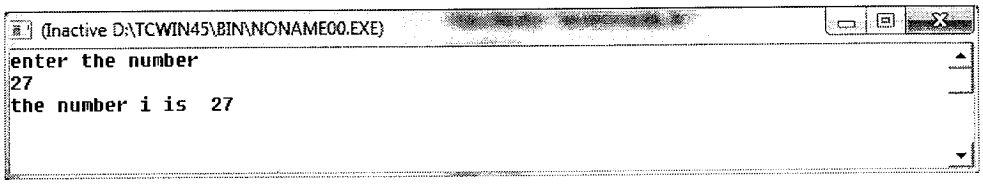
فمثلا: **cin>> i;**

هذه العبارة تخزن الرقم الذي يكتبه المستخدم من لوحة المفاتيح في متغير يدعى i . يمثل الكائن cin والذي يلفظ كـ C in لوحة المفاتيح ، ويأخذ عامل الحصول get from (>>) الأشياء الموضوعة على يساره ويضعها في المتغير الموجود على يمينه، عند تنفيذ هذه العبارة ينتظر البرنامج أن يكتب المستخدم رقماً من النوع الذي تم الإعلان عنه ويضغط على مفتاح Enter ، يتم تعيين القيمة التي أدخلها المستخدم إلى المتغير i .

المثال الآتي يوضح استخدام cin:

```
  //program 1-7
#include<iostream.h>
main()
{
int i;
cout<<"enter the number\n;"
cin>>i;
cout<<"the number i is  "<<i<<endl;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كما يلي:

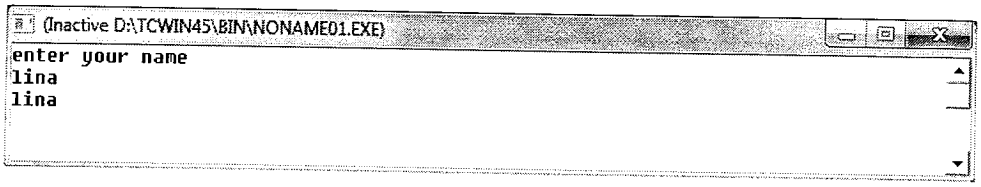


ضع برنامج لإدخال أسم شخص ما ثم طباعة هذا الاسم:

البرنامج:

```
  //program 1-8
#include<iostream.h>
 main()
{
char name[10];
cout<<"enter your name\n";
cin>>name;
cout<<name;
}
```

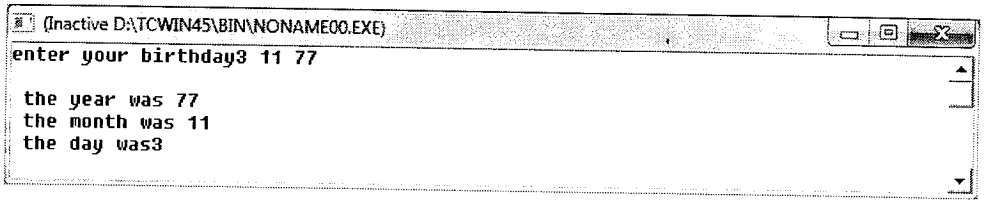
وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:



ملحوظة1: لإدخال عدة بيانات دفعة واحدة نوضحها بالبرنامج الآتي:

```
//program 1-9
#include<iostream.h>
main()
{
int day,month,year;
cout<<"enter your birthday";
cin>>day>>month>>year;
cout<<"\n the year was "<<year;
cout<<"\n the month was "<<month;
cout<<"\n the day was"<<day;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



ملحوظة2: العمليات الآتية مقبولة بلغة C++

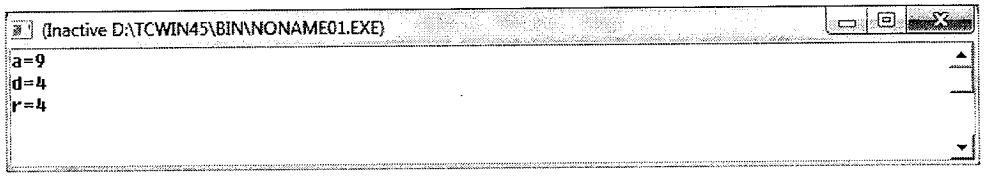
أ) A=2+(c=5)

ب) X=y=z=5;

نوضح ذلك بالبرنامج التالي:

```
//program 1-10
  #include<iostream.h>
              main()
            {
  int a,c,d,r;
       a=5+(c=4);
       d=r=c;
       cout<<"a="<<a<<endl;
             cout<<"d="<<d<<endl;
  cout<<"r="<<r;}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كما يأتي:



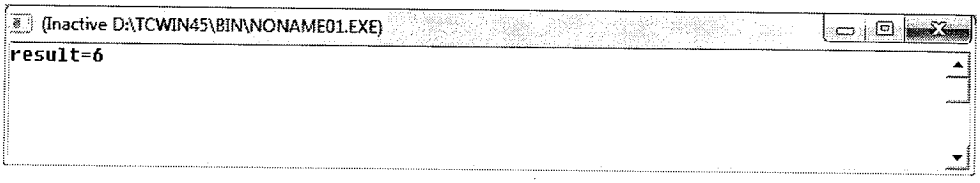
ملحوظة 3: يمكننا إسناد قيمة ابتدائية للمتغير، وذلك بأن نضع قيمة المتغير بين قوسين وذلك على النحو الآتي:.int a(0); وهذا يعني أن a=0 .

نوضح ذلك بالبرنامج الآتي:

```
//program 1-11
  #include <iostream.h>
  int main ()
  {
  int a=5; // initial value = 5
```

```
int b(2); // initial value = 2
int result; // initial value
a = a + 3;
result = a - b;
cout << "result="<<result;
return 0; }
```

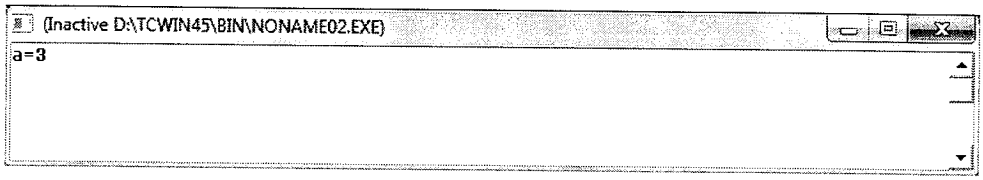
وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كما يلي:



ملحوظة 4: يمكننا أخذ القسم الصحيح من العدد الحقيقي بالتعليمة int نوضحها بالمثال الآتي:

```
//program 1-12
#include<iostream.h>
        main()
        { int a;
        float b=3.14 ;
        a=int(b);
        cout<<"a="<<a;
        }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:



1-اجب بكلمة صح أو خطأ أمام العبارات الآتية وفي حالة الخطأ اشرح لماذا.

أ) عند تنفيذ البرنامج تقوم التعليقات بطباعة النص المذكور بعد إشارة // على الشاشة.

ب) تسبب \n عند استخدامها مع cout انتقال المؤشر إلى بداية السطر التالي على الشاشة.

ت) يجب التصريح عن كل المتغيرات قبل استخدامها.

ث) يجب تحديد نمط المتغيرات عند التصريح عنها.

ج) تُعدّ لغة C++ المتغيرين sort و SORT على أنها المتغير نفسه.

ح) يجب أن يحتوي برنامج C++ الذي يقوم بطباعة ثلاثة أسطر على الخرج على ثلاث تعليمات خرج باستخدام cout.

خ) A+b اسم لمتغير.

الجواب:

أ) خطأ لاتسبب التعليقات بأي فعل عند تنفيذ البرنامج ولكن يتم استخدامها من أجل توثيق عمل البرنامج وزيادة القدرة على قراءته.

ب) صح

ت) صح

ث) صح

ج) خطأ لأن لغة C++ تفرق بين الأحرف الصغيرة والكبيرة.

ح) خطأ. يمكننا الطباعة على عدة أسطر باستخدام تعليمة cout واحدة وعدة سلاسل هروب n/.

خ) خطأ. لأن اسم المتغير لايتضمن إشارات خاصة.

2-اكتب تعليمة واحدة بلغة C++ للقيام بما يأتي:

أ) التصريح عن المتغيرات a و b و c على انها من النوع float.

ب) الطلب من المستخدم طباعة الرسالة :enter the number .

ت) الطلب من المستخدم طباعة الرسالة.Damas is capital of syria على أن ينتهي السطر الأول ب is.

ث) اطبع الرسالة this is a C++ program على أن يفصل بين الكلمات مسافة الجدولة (tab).

ج) الطلب من المستخدم إدخال ثلاثة عناصر.

الجواب:

أ) float a,b,c;

ب) cout<<"enter the number";

ت) cout<<"Damas is\n capital of syria"

ج) cin>>a>>b>>c;.

3-اكتب بلغة C++ العبارة الرياضيةالآتية: $y=x^3-2\sin(x)+tg(x)$

الحل :

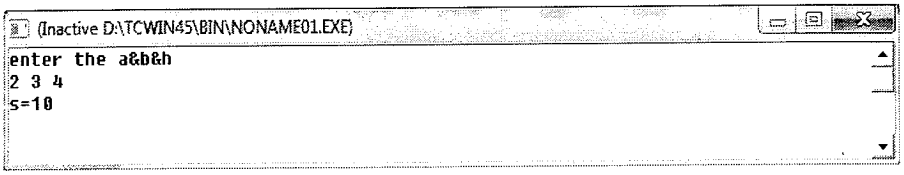
Y=pow(x,3)-2*sin(x)+tan(x)

4-ضع برنامجا لحساب مساحة شبه المنحرف

البرنامج:

```
#include<iostream.h>
main()
{
float a,b,h,s;
 cout<<"enter the a&b&h\n";
 cin>>a>>b>>h;
 s=(a+b)*h/2;
 cout<<"s="<<s;
 }
```

وبعد تنفيذ البرنامج تظهر لنا شاشة الإخراج كما يأتي:



5- تتبع عمل البرنامج الآتي واكتب الناتج:

```
#include <iostream.h>
int main ()
{
int a, b;
a = 10;
b = 4;
a = b;
b = 7;
cout << "a:";
cout << a;
cout << " b:";
cout << b;
return 0; }
```

شرح البرنامج: في البداية تم الإعلان عن عددين a و b من النوع الصحيح ثم أسندنا إلى المتغير a العدد 10 ثم أسندنا إلى المتغير b العدد 4 ثم أسندنا للمتغير a قيمة العدد b والمساوية إلى 4 أي أن قيمة a أصبحت مساوية إلى 4 ثم أسندنا إلى المتغير b القيمة 7 ثم طبعنا :a وطبعنا بعدها قيمة a والمساوية إلى 4 ثم طبعنا

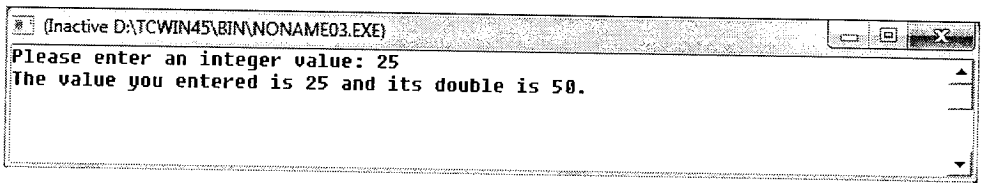
:b وبعدها طبعنا قيمة b والمساوية إلى 7 فيكون ناتج البرنامج على النحو الآتي:

a:4 b:7

تتبع عمل البرنامج الآتي واكتب الناتج:

```
#include <iostream.h>
int main ()
{
int i;
cout << "Please enter an integer value: ";
cin >> i;
cout << "The value you entered is " << i;
cout << " and its double is " << i*2 << ".\n";}
```

شرح البرنامج: أعلنا في البداية عن المتغير i من النوع الصحيح ثم طبعنا العبارة Please enter an integer value: ثم طلب منّا إدخال قيمة المتغير i فإذا أعطينا الحاسب العدد 25 فتكون قيمة i مساوية إلى 25 ثم طبعنا العبارة The value you entered is وبعدها تم طباعة قيمة i والمساوية إلى 25 بعد ذلك تمت طباعة العبارة and its double is بعد ذلك تمت طباعة ناتج ضرب العدد 2 بقيمة i والمساوي إلى العدد 50 فيكون ناتج البرنامج كما يأتي:



تمارين 1

1-ضع برنامج لحساب مساحة ومحيط مربع طول ضلعه a .

2-ماالذي يتم طباعته باستخدام التعليمات الآتية وذلك بفرض ان x=5 و y=7

أ) Cout<<x;

ب) Cout<<x+x;

ت) Cout<<"x=";

ث) Cout<<"x="<<x;

ج) Cout<<x+y<<"="<<y+x;

ح) Cout<<x<<"+"<<y<<"="<<x+y;

3-إذا كانت الدالة $y=ax^2+bx+c$ فأي من العبارات الأتية المكتوبة بلغة c++ تعبر عن الدالة السابقة.

أ) Y=a*x+b*x+c;

ب) $Y=ax^2+bx+c$;

ت) Y=a*x*+b*x+c;

ث) Y=a*x*(x+b)+c;

4-ماذا تطبع التعليمة:

Cout<<"*\n**\n***\n****\n*****\n";

5- اكتب برنامجا لإدخال ثلاثة أعداد صحيحة ثم يطبع مجموعهم وجدائهم والمتوسط الحسابي لهم .

6-اكتب خرج البرنامج الآتي:

```
#include <iostream.h>
main ()
{
```

```
int a, b; // a:?, b:?
a = 10; // a:10, b:?
b = 4; // a:10, b:4
a = b; // a:4, b:4
b = 7; // a:4, b:7
cout << "a:";
cout << a;
cout << " b:";
cout << b;
        }
```

7-تتبع عمل البرنامج الآتي واكتب ناتج البرنامج:

```
#include <iostream.h>

#define PI 3.14159
#define NEWLINE '\n'
int main ()
{
double r=5.0; // radius
double circu;
circu = 2 * PI * r;
cout << "circu="<<circu;
cout << NEWLINE;
 return 0;}
```

الفصل الثاني

البنى الشرطية واستخدام حلقات التكرار في اللغة++ C

2-1 البنى الشرطية

البنية الشرطية تعني أن تقوم بتنفيذ جزء معين من البرنامج إذا تحقق شرط ما فإذا لم يتحقق هذا الشرط فإنك تتخطى هذا الجزء.

2-1-1 بنية if:

الشكل العام لهذه البنية:

If(الشرط)

أمر واحد....

فإذا تحقق الشرط بين القوسين فإن البرنامج ينفذ الأمر الذي يلي if والجدول الآتي يعرض المعاملات التي يمكن استخدامها :

المعامل	معناه
>	أكبر من مثل: a>b
=<	أكبر من أو يساوي مثل :a>=6
<	أصغر مثل: a<2
<=	أصغر من أو يساوي مثل: a<=b
==	يساوي مثل: c=='d'
=!	لايساوي مثل: a!=5

فعلى سبيل المثال إذا كنت ترغب في اختبار القيمة الموجودة في المتغير num تساوي 6 فإنك تكتب:

If(num==6)

ويمكن أن يستخدم أمر if في شكل آخر:

if(الشرط)

{

.....

مجموعة الأوامر....

.....

}

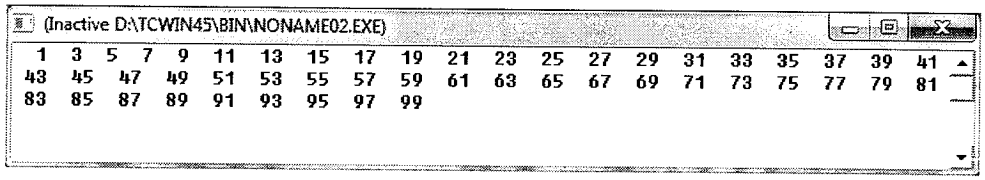
فإذا كان الشرط صحيحا" يقوم البرنامج بتنفيذ جميع العبارات الموجودة بين القوسين{ }

: ضع برنامجا لطباعة الأعداد الطبيعية الفردية والمحصورة بين 1 و100

البرنامج:

```
//program 2-1
#include<iostream.h>
main()
{
int i;
i=1;
mm:if(i<=100)
{ cout<<"  "<<i;
i=i+2;
goto mm;}}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



2-1-2-بنية if ....else :

تستخدم هذه البنية عندما ترغب في اختبار شرط معين بحيث إذا كان صحيحا فإنك تنفذ مجموعة من الأوامر وإن كان خاطئا فإنك تنفذ مجموعة أخرى :

if(الشرط)

أمر واحد بدون أقواس أو عدة أوامر بين قوسين{}

else

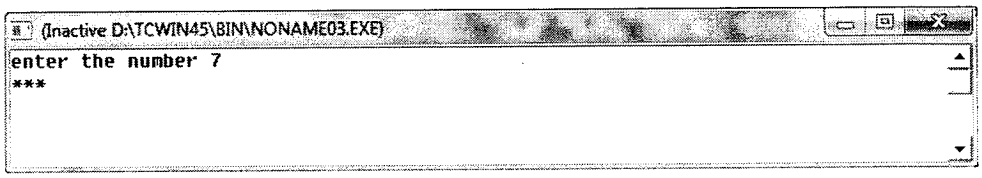
أمر واحد بدون أقواس أو عدة أوامر بين قوسين{}

[redacted] ضع برنامج لطباعة *** إذا كان العدد المدخل إلى الحاسب أكبر من خمسة وإلا يطبع :+++

البرنامج:

```
//program 2-2
#include<iostream.h>
main()
{
int n1;
cout<<"enter the number ";
cin>>n1;
if(n1>5)cout<<"***\n"<<endl;
else
cout<<"+++\n"<<endl;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:



2-1-3-بنية switch

لها الصيغة االآتية:

Switch(المتغير الذي تريد اختياره)

{

case 's':

أمر واحد أو مجموعة من الأوامر

Break;

case 12:

أمر واحد أو مجموعة من الأوامر

Break;

أمر واحد أو مجموعة من الأوامر

Default:

أمر واحد أو مجموعة من الأوامر

}

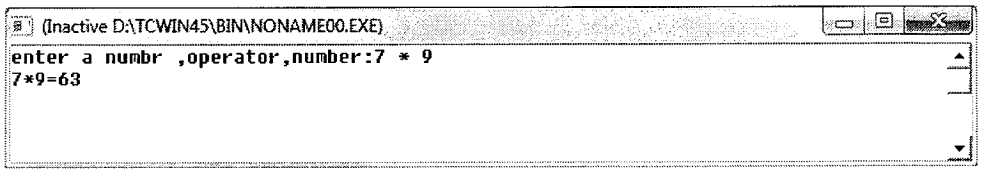
ضع برنامجا يقوم بعمليات الجمع والطرح والضرب

البرنامج:

```
//program 2-3
#include<iostream.h>
 main()
{
int n1,n2;
char op;
cout<<"enter a numbr ,operator,number:";
cin>>n1>>op>>n2;
switch(op)
{
case '+':
cout<<n1<<"+"<<n2<<"="<<n1+n2;
break;
case '-':
```

```
cout<<n1<<"-"<<n2<<"="<<n1-n2;
break;
case '*':
cout<<n1<<"*"<<n2<<"="<<n1*n2;
break;
default:
cout<<" I don't know the operator"<<op;
  }
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:



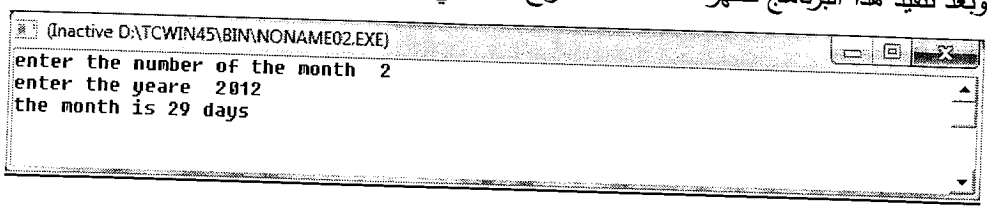
مثال ضع برنامجا لمعرفة عدد الأيام للأشهر الميلادية:

البرنامج:

```
//program 2-4
#include<iostream.h>
{
int i,y;
cout<<"enter the number of the month  ";
cin>>i;
switch(i)
{
case 1:
case 3:
case 5:
case 7:
```

```
case8:
case 10:
case 12:
cout<<"the month is 31 days\n";
break;
case 4:
case 6:
case 9:
case11:
cout<<"the month is 30 days\n";
break;
case 2:
cout<<"enter the yeare  ";
cin>>y;
if(y%4==0)
cout<<"the month is 29 days";
else
cout<<"the month is 28 days";
break;
default:
cout<<"the number not of the month";
}
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كما يأتي:



2-1-4-المعاملات المنطقية:

المعاملات المنطقيةَ تتيح لك إجراء أكثر من اختبار في شرط واحد فمثلا يمكنك أن تختبر قيمة المتغير num تساوي 5 أو إذا كانت قيمة المتغير cont تساوي 3 في شرط واحد

```
If(num==5 || count==3)
```

المعامل المنطقي || يعني أو والجدول الآتي يوضح المعاملات المنطقية ومعناها:

المعامل	معناه
&&	و and مثل:(if(a==12&&b==9
||	أو or مثل:(if(a>3 || b<7
!	ليسnot يستخدم لتحديد إن كان أحد المتغيرات يساوي صفرا" أو لا.مثل:(if(!result سيكون صحيحا إذا كانت قيمة result تساوي صفرا" وسيكون خاطئا فيما عدا ذلك.

2-1-5-المعامل الشرطي ? :

إضافة الى البنى الشرطية السابقة هنالك معامل شرطي آخر في لغة++C هذا المعامل هو معامل الاستفهام ? وله الصيغة الآتية:

```
result=(الشرط)? Var1:var2;
```

إذا كان الشرط الوارد في العبارة صحيحا فإنه ينقل القيمة الموجودة في المتغير var1 إلى المتغير result وإذا كان الشرط خاطئا فإنه ينقل القيمة الموجودة في المتغير الثاني var2 إلى المتغير result

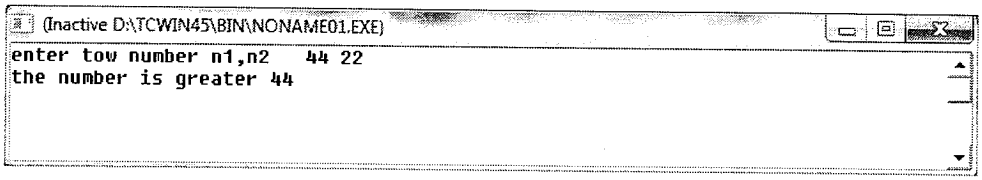
مثال ضع برنامجا لطباعة أكبر العددين الذي تم إدخالهما:

البرنامج:

```
//program 2-5
#include<iostream.h>
  main()
{
```

```
int n1,n2,max;
cout<<"enter tow number n1,n2   ";
cin>>n1>>n2;
max=(n1>n2)?n1:n2;
cout<<"the number is greater "<<max;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كما يأتي:

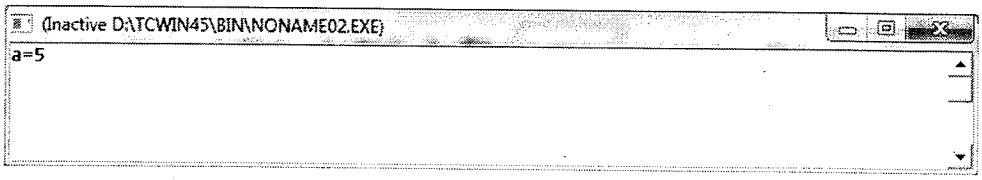


2-1-6 -معامل الفاصلة ، :

يستخدم هذا المعامل للفصل بين تعليمتين أو أكثر نوضح ذلك في البرنامج الآتي:

```
//program 2-6
#include<iostream.h>
   main()
      {
            int a,b;
            a=(b=2,b+3);
        cout<<"a="<<a;}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كما يأتي:



2-2 استخدام حلقات التكرار

التكرار هو جوهر العديد من تطبيقات الحاسوب ،إذ إن حلقة التكرار هي بنية تحكم تقوم بتنظيم العمليات المتكررة في البرنامج .

يوفر Tc++ ثلاث بنيات مستقلة لحلقات التكرار وكل منها يملك تصميما مختلفا للتحكم بخطوة التكرار

2-2-1- حلقة for:

تستطيع هذه الحلقة تحديد مدى من خطوات التكرار بشكل محدد مسبقا

الشكل العام لهذه الحلقة:

```
For(مقدار تزايد المتغير; الشرط;القيمة الابتدائية للمتغير)
                {
            مجموعة من التعليمات
                }
```

إذا كانت تعليمة واحدة لاداعي لوجود القوسين {}

مثل:

```
for(i=1; i<100;i++)
                {
            مجموعة من الأوامر
                }
```

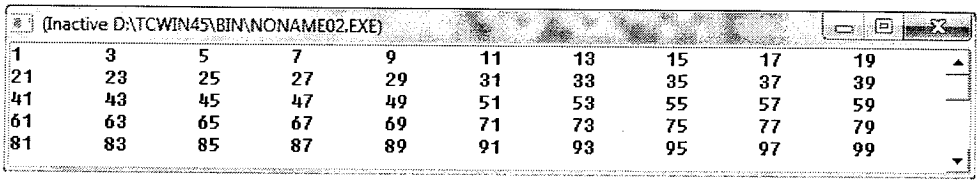
مثال: ضع برنامجا لطباعة الأعداد الطبيعية الفردية والمحصورة بين 1 و100

البرنامج:

```
//program 2-7
#include<iostream.h>
  main()
      {
int i;
for(i=1;i<=100;i+=2)
cout<<i<<"\t";
```

}

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:



2-2-2- حلقة while :

تعبر عن شرط يتحكم بحالة مواصلة التكرار ويوجد الشرط عند أعلى حلقة التحكم وتتوقف خطوات التكرار عندما يصبح الشرط خطأ.

الصيغة العامة لهذه الحلقة:

```
While(الشرط)
{
.....
مجموعة أوامر
.....
}
```

وإذا كنت ترغب في تكرار عبارة واحدة فلن تحتاج إلى استخدام القوسين وعندما يبدأ التكرار سيختبر الشرط الموجود بين القوسين فإذا وجده صحيحا فإنه ينفذ العبارة أو العبارات التي تلي كلمة while ثم يعود ويختبر الشرط مرة أخرى ،وهكذا حتى يصبح الشرط خاطئا فيتوقف.

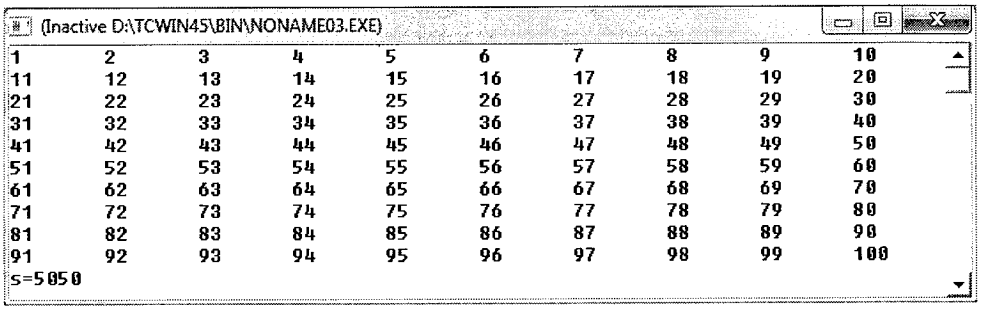
[redacted]ضع برنامجا لطباعة الأعداد الطبيعية من 1 وحتى 100 وإيجاد مجموعها.

البرنامج:

```
//program 2-8
#include<iostream.h>
main()
{
int i,s;
```

```
 i=1;
 s=0;
while(i<=100)
{
s=s+i;
 cout<<i<<"\t";
 i=i+1;
}
cout<<"s="<<s;
}
```

وبعد تنقيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



2-2-3-حلقة do .....while :

تحتوي هذه الحلقة على تعبير شرطي وهذا الشرط يعبر عنه في أسفل حلقة التكرار والصيغة العامة لهذه الحلقة:

```
do
{
.....;
مجموعة من الأوامر....
.....
}while(الشرط);
```

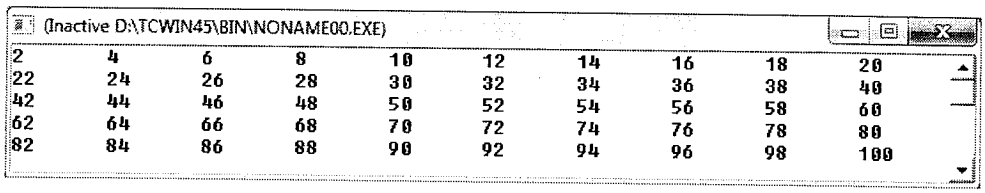
تقوم الحلقة بتنفيذ العبارات الموجودة بين القوسين مرة واحدة ثم يختبر الشرط الموجود بعد حلقة while فإذا وجد الشرط صحيح يعود إلى تنفيذ العبارات مرة أخرى وإذا وجده خاطئا فإنه يتوقف.

ضع برنامجا لطباعة الأعداد الطبيعية الزوجية والمحصورة بين 1 و 100 .

البرنامج:

```
//program 2-9
#include<iostream.h>
        main()
{ int i;
i=2;
do
{ cout<<i<<"\t";
i=i+2;
}
while(i<=100);
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:



2-2-4 -**حلقات التكرار المتداخلة**:(نوضحها بالمثال االآتي)

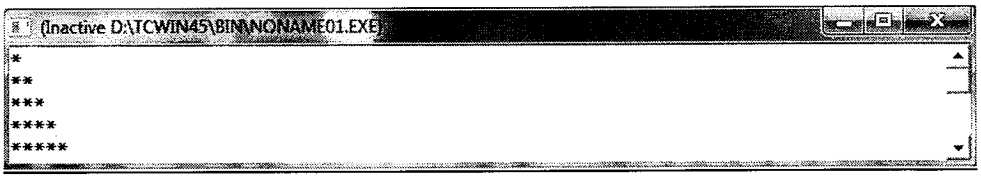
مثال: ضع برنامج لرسم الشكل الآتي:

```
*
**
****
*****
******
```

البرنامج:

```
//program 2-10
#include<iostream.h>
main()
{
int i,j;
for (i=1;i<=5;i++)
{
for(j=1;j<=i;j++)
cout<<"*" ;
cout<<"\n";
}}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



ضع برنامجا يمثل ساعة رقمية

البرنامج:

```
//program 2-11
#include<iostream.h>
#include<conio.h>
  main()
       {
  int a1,a2,a3,h,m,s;
  long i;
```

```
 cout<<"enter the hour & min & sec\n";
       cin>>a1>>a2>>a3;
for(h=a1;h<=23;h++)
{
  for(m=a2;m<=59;m++)
            {
  for(s=a3;s<=59;s++){
            for(i=1;i<=500000;i++)
                gotoxy(30,10);
                cout<<h<<":"<<m<<":"<<s;}
   a3=0;
   }
                a2=0;
   }
                a1=0;
               }
```

5-2-2 -التحكم بالحلقات:

تعمل الحلقات عادة بصورة جيدة إلا أننا في بعض الأوقات نحتاج ألى التحكم بعمل الحلقات ، العبارتين break و continue توفران هذه المرونة المطلوبة.

العبارة break :

تتيح لنا العبارة break الخروج من الحلقة في أي وقت.

المثال الآتي يبين لنا كيفية عمل العبارة break :

مثال ضع برنامجا لمعرفة العدد المعطى فيما إذا كان عددا أوليا أم لا.

البرنامج:

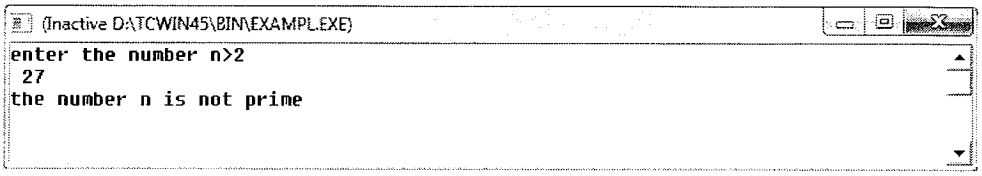
```
//program 2-12
#include<iostream.h>
main( )
```

```
{
int  i ,n,prime;
 prime=1;
cout<<"enter the number n>2\n ";
cin>>n;
for (i=2;i<n;i++)
{
if (n%i== 0)
 {
prime=0;
break;
}
}
 if(prime==0)cout<<"the number n is not prime";
 else
          cout<<"the number n is  prime";
}
```

لقد أعطينا في هذا البرنامج قيمة المتغير prime العدد 1 وفي حال ماتبقى من قسمة العدد n على 2 أو 3 أو.....حتى n-1 يكون مساويا للصفر يعطى المتغير prime القيمة 0 .

ثم تمت المناقشة إذا كانت قيمة المتغير prime مساوية للواحد كان العدد n عددا أوليا وإلا فالعدد n عدد غير أولي نلحظ في حال باقي قسمة n على i يكون مساويا للصفر يأخذ المتغير prime القيمة 0 ويخرج من الحلقة.

وبعد تنفيذ البرنامج السابق يكون الناتج:



العبارة continue:

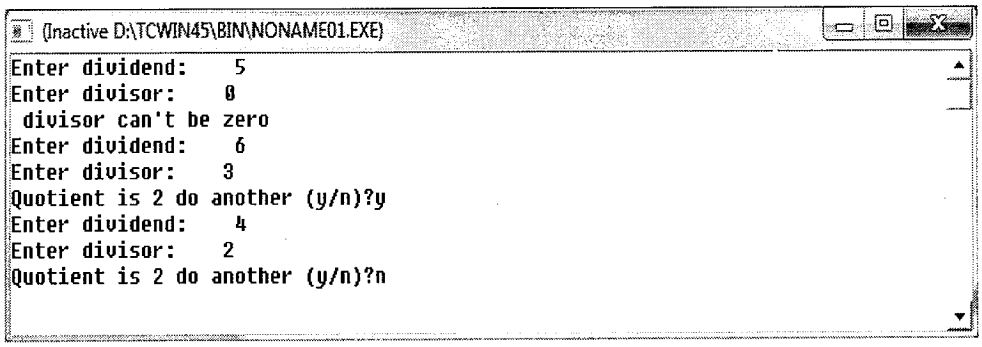
تعيد العبارة continue التنفيذ إلى أعلى الحلقة.

نوضح ذلك من خلال البرنامج الآتي:

```
//program 2-13
#include<iostream.h>
main( )
{
int dividend , divisor;
char ch;
do
//Continued
{
cout << "Enter dividend:    ";
cin>>dividend;
cout<< "Enter divisor:    ";
cin>>divisor;
if( divisor == 0)
{
```

```
cout<<" divisor can't be zero\n" ;
continue;
}
 cout <<"Quotient is "<< dividend/divisor;
cout<<" do another (y/n)?";
cin>>ch;
}
while (ch != 'n');
}
```

ويعد تنفيذ البرنامج يكون ناتج الخرج:



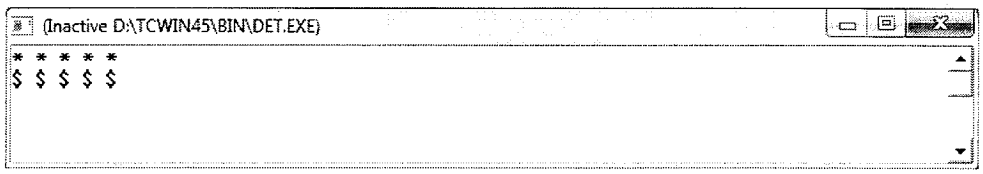
1-السؤال الأول في الدورة الأولى/2012-2013

بفرض أن x = 9 و y = 11 والمطلوب ما هي مخرجات الجزء الآتي من البرنامج:

```
if ( x < 10)
if ( y > 10)
        cout << "* * * * *" << endl;
        else
        cout << "# # # # #" << endl;
        cout << "$ $ $ $ $" << endl;
```

الجواب:

الناتج هو:



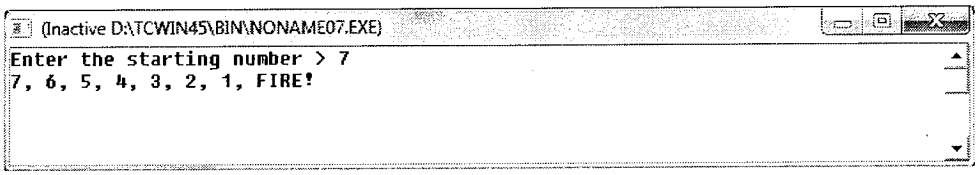
2-ضع برنامج لإدخال عدد صحيح ثم استخدم حلقة while لطباعة العدد الصحيح وجميع الأعداد الصحيحة الي هي أقل منه حتى العدد 1.

البرنامج:

```
#include <iostream.h>
int main ()
{
int n;
cout << "Enter the starting number > ";
cin >> n;
while (n>0) {
cout << n << ", ";
--n;
}
```

```
return 0;}
```

وبعد تنفيذ البرنامج يكون الناتج:



3- ضع برنامج لتحويل أي عدد صحيح من النظام العشري إلى النظام الثنائي علما أن خوارزمية التحويل يعطى بالشكل الآتي:

المدخلات: العدد الصحيح بالنظام العشري a غذ إن a>1.

المخرجات: أرقام العدد بالنظام الثنائي b_k.

خطوة 1 : نجعل i=0 .

خطوة 2: نحسب b_i =a mod 2 ماتبقى من قسمة a على 2.

خطوة 3: إذا كانت b_i =0 محققة نضع قيمة a الجديدة تساوي ناتج قسمة a على 2 وإلا فتكون قيمة a الجديدة مساوية إلى ناتج قسمة (a-1) على 2 .

خطوة 4: إذا كانت a>1 محققة ننفذ الخطوة 5 وإلا ننفذ الخطوة 6.

خطوة 5: نعطي i تزايد بمقدار 1 أي i=i+1 ثم نذهب إلى الخطوة 2.

خطوة 6: نطبع أرقام العدد بالنظام الثنائي b_k إذ إن k=i,i-1,i-2, ...0

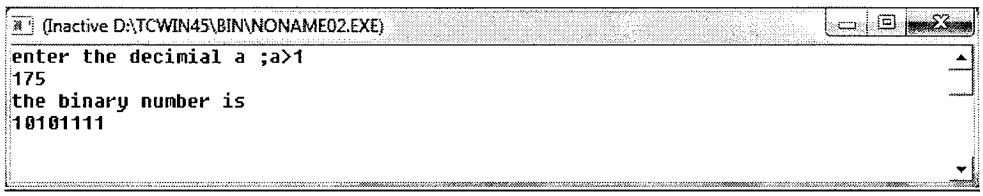
خطوة 7: النهاية.

البرنامج:

```
#include<iostream.h>
main()
{
int a,i,k;
float b[20];
  cout<<"enter the decimial a ;a>1\n";
```

```
 cin>>a;
 i=0;
mm: b[i]=a%2;

 if(b[i]==0)a=(a/2);
            else a=(a-1)/2;
 if(a>1){i=i+1; goto mm;}
 cout<<"the  binary number is\n" ;
 cout<<"1";
for(k=i;k>=0;k--)
 cout<<b[k];
}
```

وبعد تنفيذ البرنامج يكون الناتج:



4- لتكن الدالة $y=f(x)$ معطاة بالجدول الآتي:

x	x_0	x_1	x_2	...	x_n
$f(x)$	y_0	y_1	y_2	...	y_n

المطلوب: ضع برنامجا لإيجاد ما يأتي:

1- قيم مضاريب لاغرانج $L_0(x), L_1(x),...,L_n(x)$ وذلك عند $x=x_p$.

2-قيمة كثيرة حدود لاغرانج الموافقة للدالة عند $x=x_p$.

تعطى كثيرة حدود لاغرانج الموافقة للدالة $y=f(x)$ بالشكل الآتي:

$$p_n(x)=\sum_{j=0}^{n} y_j L_j(x)=L_0(x)y_0+L_1(x)y_1+\cdots+L_n(x)y_n$$

$$L_j(x)=\frac{(x-x_0)(x-x_1)\ldots(x-x_{j-1})(x-x_{j+1})\ldots(x-x_n)}{(x_j-x_0)(x_j-x_1)\ldots(x_j-x_{j-1})(x_j-x_{j+1})\ldots(x_j-x_n)}$$
$$=\prod_{\substack{i=0\\ i\neq j}}^{n}\frac{(x-x_i)}{(x_j-x_i)}$$

ويمكننا كتابة الخوارزمية على الشكل الآتي:

المدخلات: n (عدد النقاط -1)، $x=x_p$ ،

نقاط الاستيفاء x_i، وقيم الدالة y_i من أجل i=0,1,2,....n .

المخرجات: L_i إذ إن i=0,1,...n قيم مضاريب لاغرانج عند x_p ،

P قيمة كثيرة الحدود الموافقة للدالة $y=f(x)$ عند x_p

خطوة1:حساب وطباعة مضاريب لاغرانج :

for i=0,1,2,....n

$$\{\ L_i=\prod_{\substack{j=0\\ j\neq i}}^{n}\frac{x_p-x_j}{x_i-x_j}$$

Write L_i }

خطوة2: حساب قيمة كثيرة حدود لاغرانج الموافقة في النقطة x_p :

$$p_n=\sum_{i=0}^{n} L_i(x_p)\,y_i$$

خطوة3: طباعة P_n.

خطوة4: النهاية .

البرنامج:

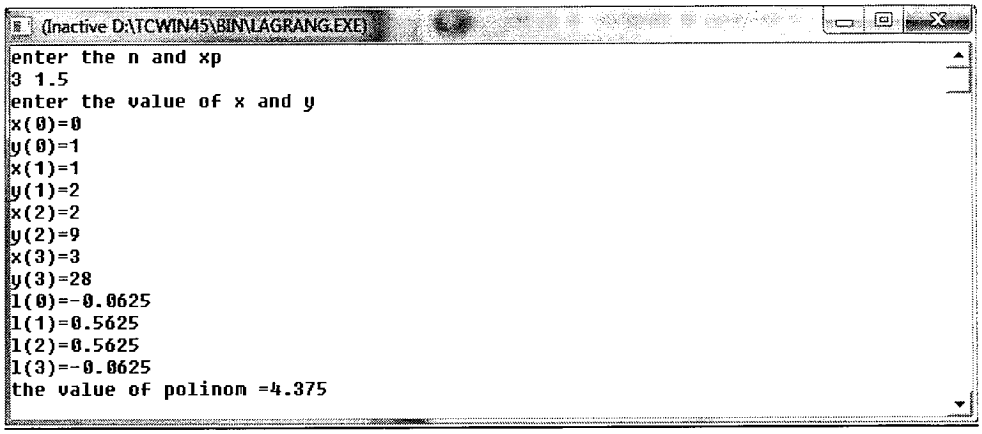
```
#include<iostream.h>
#include <iomanip.h>
main()
{
int i,j,n;
float l[10],x[10],y[10];
 float a,b,p,xp;
  cout<<"enter the n and xp\n";
        cin>>n>>xp;
        cout<<"enter the value of x and y\n";
        for(i=0;i<=n;i++)
        { cout<<"x("<<i<<")=";
        cin>>x[i];
         cout<<"y("<<i<<")=";
         cin>>y[i];}
         for(i=0;i<=n;i++)
         {
         a=1.;b=1.;
         for(j=0;j<=n;j++){
         if(i==j)goto mm;
         a=a*(xp-x[j]);b=b*(x[i]-x[j]);
              mm:continue;}
              l[i]=a/b;
              cout<<"l("<<i<<")="<<l[i]<<endl;
              }
              p=0.;
```

```
for(i=0;i<=n;i++)
p=p+l[i]*y[i];
cout<<"the value of polinom ="<<p;
}
```

الشرح:

أدخلنا في البداية قيم كل من n (عدد النقاط المدخلة −1) و xp قيمة الدالة في النقطة المطلوبة. ثم أدخلنا قيم كل من x وy الموجودتين في الجدول، ثم بدأنا بحساب قيم كثيرات حدود لاغرانج، آخذين a ممثلة للقيم الناتجة من البسط، و b القيمة الناتجة من المقام، ثم قمنا بطباعة قيم كثيرات حدود لاغرانج، وبعدها طبعنا قيمة الدالة في النقطة المطلوبة .

وبعد تنفيذ البرنامج تظهر لنا شاشة الإخراج كما يأتي:



5- لتكن الدالة y=f(x) معطاة بجدول من القيم المطلوب:

ضع برنامجا لإيجاد قيمة كثيرة حدود نيوتن−غريغوري التقدمية من الدرجة n=3 عند $x=x_p$ الموافقة للدالة المعطاة قيمها بالشكل $\{(x_i, y_i)\}_{i=0}^{n}$.

تعطى كثيرة حدود نيوتن غريغوري من خلال العلاقة الآتية:

$$P_n(x)=\binom{S}{0}\Delta^0 y_0+\binom{S}{1}\Delta y_0+\binom{S}{2}\Delta^2 y_0+\cdots+\binom{S}{n}\Delta^n y_0$$

إذ إن: $s=(x-x_0)/h$

ويمكننا وضع الخوارزمية الموافقة على النحو الآتي:

المدخلات: n=3، و y_i قيم الدالة $y=f(x)$ ، عند x_i.

x_0 نقطة البداية في الجدول، h مقدار الخطوة .

$x=x_p$ النقطة المطلوب حساب قيمة كثيرة الحدود عندها .

المخرجات: قيمة كثيرة حدود نيوتن–غريغوري التقريبية للدالة f عند $x=x_p$.

خطوة 1: نحسب $S=(x_p-x_0)/h$.

خطوة 2: حساب الفروق من المرتبة الأولى

For i=0,1,2,…n-1

$d1_i=y_{i+1}-y_i$

خطوة 3: حساب الفروق من المرتبة الثانية

For i=0,1,2,…n-2

$d2_i=d1_{i+1}-d1_i$

خطوة 4: حساب الفروق من المرتبة الثالثة

For i=0,1,2,…n-3

$d3_i=d2_{i+1}-d2_i$

خطوة 5: نحسب قيمة كثيرة حدود نيوتن–غريغوري التقريبية للدالة f عند $x=x_p$ من العلاقة:

$$P_3 = f_0 + S*d1_0 + \frac{S(S-1)}{2}d2_0 + \frac{S(S-1)(S-2)}{6}d3_0$$

خطوة 6: طباعة قيمة P_3.

خطوة 8: النهاية.

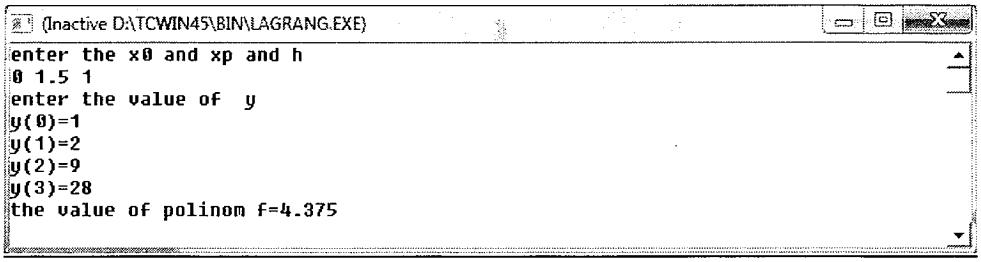
البرنامج:

```
#include<iostream.h>
main()
{
int i;
 float d1[10],d2[10],d3[10],y[10],h,s,f,xp,x0;
       cout<<"enter the x0 and xp and h\n";
        cin>>x0>>xp>>h;
        cout<<"enter the value of  y\n";
        for(i=0;i<=3;i++){
              cout<<"y("<<i<<")=";
         cin>>y[i]; }
         s=(xp-x0)/h;
         for(i=0;i<3;i++)
         d1[i]=y[i+1]-y[i];
               for(i=0;i<2;i++)
         d2[i]=d1[i+1]-d1[i];
              for(i=0;i<1;i++)
         d3[i]=d2[i+1]-d2[i];
f=y[0]+s*d1[0]+(1./2)*s*(s-1)*d2[0]+(1./6)*s*(s-1)*(s-2)*d3[0];
cout<<"the value of polinom f="<<f;
              }
```

الشرح:

بدأنا بإدخال قيم كل من $x_0=0$ و $xp=1.5$ و $h=1$ ثم أدخلنا قيم الدالة $y=f(x)$ الموجودة في الجدول، بعدها تم استنتاج جداول الفروق التقدمية من المرتبة الأولى، والثانية والثالثة، ثم أوجدنا قيمة $p_n(x)$ في النقطة $x=1.5$.

وبعد تنفيذ البرنامج تظهر لنا شاشة الإخراج كما يلي:



6- باستخدام تعليمة switch ضع برنامجا لحساب مساحة الأشكال الآتية:

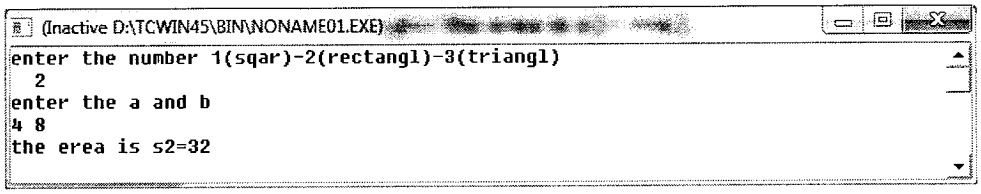
المربع-المستطيل-المثلث.

البرنامج:

```
#include<iostream.h>
main()
{
 int i;
 float a,b,h,s1,s2,s3;
cout<<"enter the number 1(sqar)-2(rectangl)-3(triangl)\n  ";
cin>>i;
switch(i)
{
case 1:
cout<<"enter the a=";
cin>>a;
s1=a*a;
cout<<"the erea is s1="<<s1;
break;
case 2:
 cout<<"enter the a and b\n";
cin>>a>>b;
s2=a*b;
cout<<"the erea is s2="<<s2;
break;
case 3:
cout<<"enter the a and h\n";
```

```
cin>>a>>h;
s3=(a/2.)*h;
cout<<"the erea is s3="<<s3;
break;
default:
cout<<"the number not valid";
}
}
```

وبعد تنفيذ البرنامج تظهر لنا شاشة الإخراج كما يأتي:

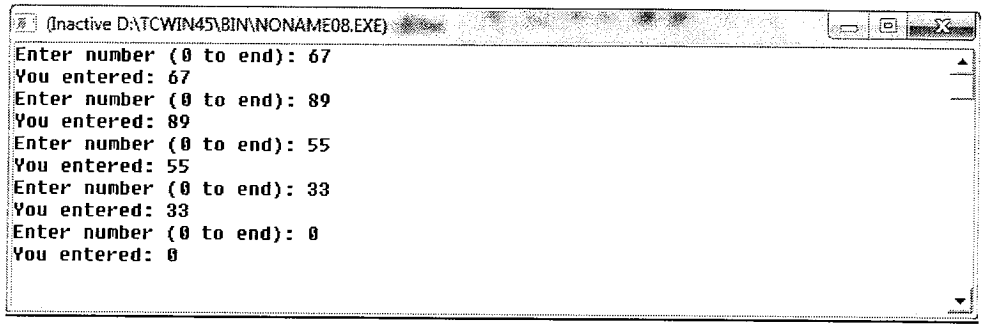


7- ضع برنامجا يستخدم حلقة do ...while لإدخال عدد ثم يطبع العدد الذي أدخلته وتستمر عملية الإدخال والطباعة حتى يتم إدخال العدد 0 .

البرنامج:

```
#include <iostream.h>
int main ()
{
 long n;
do {
cout << "Enter number (0 to end): ";
cin >> n;
cout << "You entered: " << n << "\n";
} while (n != 0);}
```

وبعد تنفيذ البرنامج تظهر لنا شاشة الإخراج كما يأتي:

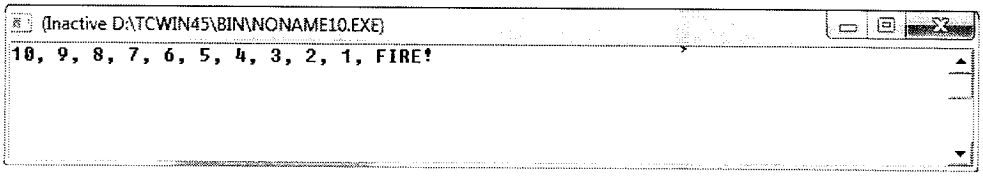


8- تتبع عمل البرنامج الآتي ثم اكتب الناتج:

```
#include<iostream.h>
int main ()
{
int n=10;
loop:
cout << n << ", ";
n--;
if (n>0) goto loop;
cout << "FIRE!\n";}
```

شرح البرنامج: في البداية تم الإعلان عن العدد الصحيح n وأسندنا إليه القيمة 10 ثم طبعنا قيمة العدد n والتي تساوي 10 ثم أنقصنا قيمة n بمقدار 1 إي اصبحت القيمة الجديدة ل n مساوية إلى 9 وبع ذلك تم التحقق من الشرط وهو هل قيمة n أكبر من الصفر فإذا كان الشرط محققا" يتم الذهاب إلى الكلمة lopp وذلك نتيجة استخدام التعليمة ;goto loop فيتم طباعة قيمة n ومن ثم يتم إنقاص قيمة n مقدار 1 لتكون القيمة الجديدة ل n مساوية إلى 8 وهكذا تستمر العملية مادام قيمة العدد n أكبر من الصفر

ويكون ناتج خرج البرنامج:



تمارين 2

1-ضع برنامجا لإيجاد الجذور الحقيقية والعقدية لمعادلة من الدرجة الثانية .

2-ضع برنامجا لطباعة الأعداد الطبيعية التي تقبل القسمة على 3 و 5 في آن واحد وإيجاد مجموعها والتي هي محصورة بين الواحد والمئة.

3-ضع برنامجا لمعرفة العدد المدخل إلى الحاسب فيما إذا كان عدد فردي أم زوجي .

4-ضع برنامجا لطباعة الأعداد الزوجية من 100 وحتى 1 وبترتيب تنازلي.

5-ضع برنامجا لمعرفة العدد المدخل إلى الحاسب فيما إذا كان عدد موجب أم سالب .

6-حدد وصحح الأخطاء الواردة في كل مما يأتي:

```
a)int x=10;
while (x<=5){
y *=x;
++x;
b)cin<<y;
c)if(x==7)cout<<"latakia"<<enfl;
else;
cout<<"damas"<<endl;
d)switch(x){
case 1:cout<<"the number is 1"<<endl;
case 2:cout<<"the number is 2"<<endl;
break;
default:
cout << "the nuber is not 1 or 2"<<endl;
break;
}
```

7-اكتب برنامجا لحساب قيمة s=n! .

8 -ماهو عدد مرات تنفيذ الحلقات الآتية:

a. for(i=1;i<15;i=i+3)

b. **for**(**int** i=0,**int** j=9;i<=16&&j>6;i++,j-=2)

c. int t=4;

while(t<=17)

{....

t+=3;}

d. int t=4;

do

{....

t+=3;} **while**(t>17);

e. **for**(i=9;i>5;i++)

9- ضع برنامجا يطلب إدخال درجة طالب في مادة ويخرج التقدير حسب الجدول الآتي :

"grade" التقدير	"degree" الدرجه
A+	95 فما فوق
A	90 أقل من 95
B+	85 أقل من90
B	80 أقل من 85
C+	75 أقل من 80
C	70 أقل من 75
D+	65 أقل من70
D	60 أقل من65
F	أقل من 69

الفصل الثالث

الإجراءات والتوابع الرياضية وتوليد الأعداد العشوائية

1-1-3 تعريف الإجراء:

الإجراء هو مجموعة من التعليمات بوضعها جزء ا"مستقلا" عن البرنامج وله الصيغة العامة الآتية:

Data type name(المتغيرات التي تستقبلها)

{

.....

مجموعة من الأوامر

.....

}

وكما ترى أنه يتكون من جزأين : رأس الإجراء وهو جزء مستقل يعطي تعريفا عن الإجراء والجزء الثاني وهو جسم الاجراء وهو مجموعة من الأوامر التي يتكون منها الإجراء محصورة بين {} رأس الاجراء يتكون من ثلاثة أجزاء:

الجزء الأول data type:

يحدد القيمة التي يعود بها الإجراء ،فالإجراءات في لغة++C نوعان ،نوع ينفذ الأوامر الموجودة ولا يعود بقيمة إلى البرنامج ،ونوع ينفذ الأوامر الموجودة ثم يعود بقيمة إلى البرنامج ، فإذا كان الإجراء االذي نكتبه يعود بقيمة إلى البرنامج يجب أن تحدد نوع هذه القيمة في رأس البرنامج فإذا كان الإجراء يعود بقيمة صحيحة يجب أن يبدأ رأس الإجراء بـ int وهكذا.....

أما إذا كان الإجراء لايعود بقيمة إلى البرنامج فيجب أن تستخدم كلمة void .

الجزء الثاني name:

هو اسم الإجراء ويمكنك اختيار أي اسم ترغب به بشكل مشابه لأسماء المتغيرات .

الجزء الثالث:

المتغيرات التي يستقبلها الإجراء فعلى سبيل المثال إذا كنت تكتب إجراء لتقوم بفتح أحد الملفات على القرص سيحتاج هذا الإجراء إلى اسم الملف الذي ستفتحه. عندها يمكن أن تمرر له اسم الملف بوضعه متغيرا" بين القوسين إما إذا كنت لاتريد أن تمرر أية متغيرات إلى الإجراء ضع كلمة void بين القوسين بعد اسم الإجراء مثل: void start(void) .

هنا اسم الإجراء start وهو لايعود بقيمة إلى البرنامج، لذلك وضعنا قبله كلمة void كما أنه لايحتاج إلى متغيرات لذلك وضعنا بعده كلمة void .

char start(int j)

هنا اسم الإجراء start وهو يعود بحرف هجائي، لذلك وضعنا قبله كلمة char كما أنه يتطلب تمرير عدد صحيح، لذلك وضعنا بعده كلمة int بين القوسين.

2-1-4 الإعلان عن الإجراءات:

قبل أن تستخدم الإجراء تعلن عنه في بداية البرنامج وقبل الإجراء الرئسي main()

ويتكون الإعلان عن الاجراء من المعلومات الواردة نفسها في رأس البرنامج:

فمثلا إذا كان لديك الإجراءالآتي:

```
void newline(viod)
{
cout<< "----------------------\n";
}
```

فإنك تعلن عنه في بداية البرنامج كمايأتي:

```
void newline(void);
```

نلحظ وجود الفاصلة المنقوطة في نهاية الإعلان.

3-1-3- استدعاء الاجراء:

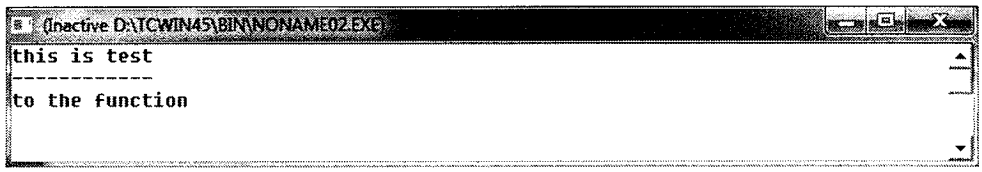
بعد أن تنتهي من كتابة الإجراء والإعلان عنه يمكنك في أي وقت استدعائه وعندما تستدعي الإجراء سيقوم البرنامج بالانتقال إلى هذا الإجراء وينفذ العبارات الواردة فيه.

وعندما ينتهي من تنفيذ العبارات الواردة في الإجراء يعود إلى العبارة التي تلي العبارة التي استدعت الإجراء.

ضع برنامج لاستخدام إجراء يتضمن رسم خط بحيث يمكننا استخدام هذا الإجراء في البرنامج الرئيسي

```
//program 3-1
#include<iostream.h>
        void newline(void);
main()
{
cout<<"this is test\n";
newline();
cout<<"to the function\n";
}
void newline(void)
{
cout<<"------------\n";
 }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:



تسمى هذه الإجراءات إجراءات بسيطة .

3-1-4- تمرير البيانات للإجراء

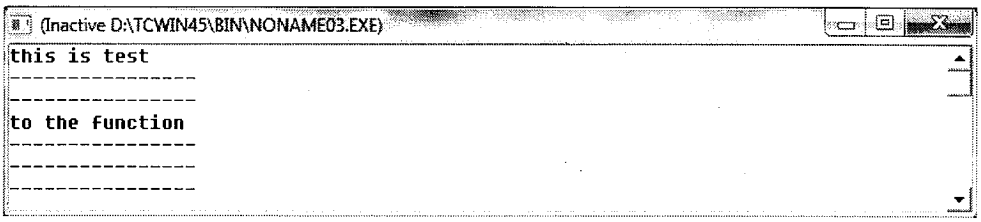
نوضحها بالمثال الآتي:

ضع إجراء يستقبل قيما" من البرنامج الرئيسي لرسم عدة خطوط دفعة واحدة.

البرنامج:

```
//program 3-2
#include<iostream.h>
        void newline(int i);
main()
{
cout<<"this is test\n";
newline(2);
cout<<"to the function\n";
newline(3);
}
void newline( int n)
{ int k;
for(k=1;k<=n;k++)
cout<<"----------------\n";
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:



في هذا المثال يتم طباعة خطين تحت عبارة this is test وثلاث خطوط تحت عبارة to the function .

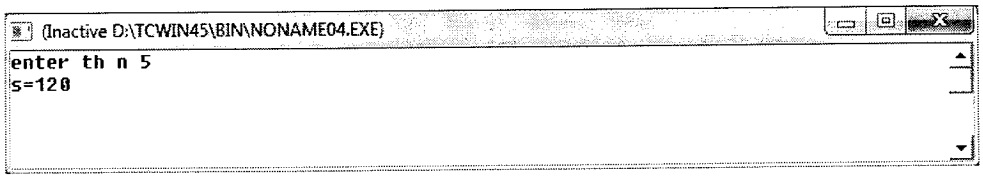
5-1-3 إجراء يستقبل قيمة من البرنامج الرئيسي ويعيد قيمة إليه

ضع إجراء لحساب !n ثم استفد من ذلك في حساب عاملي أي عدد .

البرنامج:

```
//program 3-3
#include<iostream.h>
int fact(int i);
main()
{
int n,s;
cout<<"enter th n ";
cin>>n;
s=fact(n);
cout<<"s="<<s;
}
int fact(int n1)
{ int k,s1;
s1=1;
for(k=1;k<=n1;k++)
s1=s1*k;
return s1;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:

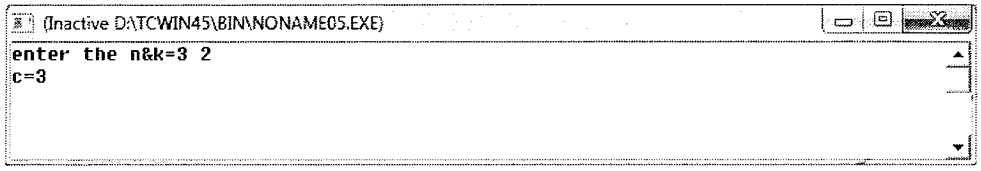


ضع برنامجا لحساب العلاقة $C_k^n = \frac{n!}{(k!)(n-k)!}$.

البرنامج :

```
//program 3-4
#include<iostream.h>
  int fact( int n1);
main()
{  int n,k,p;
       int s,s1,s2;
   float c;
 cout<<"enter the n&k=";
 cin >>n>>k;
      s=fact(n);
  s1=fact(k);
  p=n-k;
  s2=fact(p);
  c=s/(s1*s2);
     cout<<"c="<<c;
 }
int fact (n2)
{int i,s3;
s3=1;
for(i=1;i<=n2;i++)
s3=s3*i;
return s3;
          }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



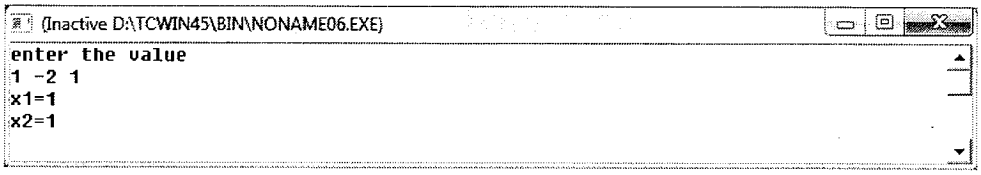
ضع برنامجا لحساب الجذور الحقيقية لمعادلة من الدرجة الثانية واضعا حساب المميز في إجراء خاص.

البرنامج:

```
//program 3-5
#include<iostream.h>
  #include<math.h>
  float delta(float a1,float b1,float c1);
main()
{ float a,b,c,d,x1,x2;
cout<<"enter the value\n";
 cin>>a>>b>>c;
d=delta(a,b,c);
if(d<0)cout<<"nosultion";
      else
      {x1=(-b+sqrt(d))/(2*a);
      x2=(-b-sqrt(d))/(2*a);
                cout<<"x1="<<x1<<endl;
                cout<<"x2="<<x2<<endl;
                }
}
float delta(float a2,float b2,float c2)
{
```

```
float d2;
d2= b2*b2-4*a2*c2;
   return d2;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:



ملحوظة :يمكننا كتابة الإجراء قبل الإجراء الرئيسي ()main نوضحه بالمثال الآتي:

```
//program 3-6
#include <iostream.h>
int addition (int a, int b)
{
int r;
r=a+b;
return (r);
}
int main ()
{
int z;
z = addition (5,3);
cout << "The result is " << z; }
```

3-1- 6-الإجراءات السطرية:

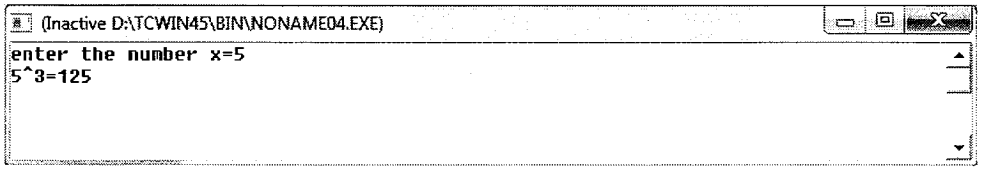
يتوافر بلغة c++ إجراءات تسمى بالإجراءات السطرية inline function وهي تساعد على التقليل من الضياع الناتج عن عملية اللاستدعاء –وبصفة خاصة– في حالة الإجراءات الصغيرة (الممكن كتابتها في سطر واحد) ؛لذلك نستخدم تعليمة inline .

اكتب إجراء سطريا" لحساب مكعب أي عدد ثم استفد من ذلك في حساب مكعب أي عدد معطى إلى الحاسب.

البرنامج:

```
//program 3-7
#include<iostream.h>
#include<math.h>
inline float cub(float a)
{
return pow(a,3); }
main()
{
  float x;
  cout<<"enter the number x=";
  cin>>x;
  cout<<x<<"x^3"<<cub(x);
  }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الاخراج كمايأتي:

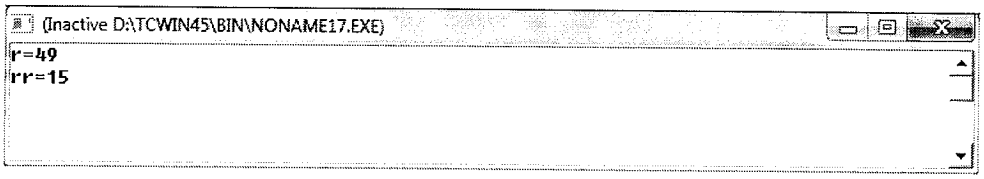


3-1-7-التحميل الزائد للإجراء:

المقصود من ذلك استخدام الإجراء أكثر من مرة في مهام مختلفة ، وهذا يعني أن نستخدم اسم الإجراء لتنفيذ إجراء أخر مختلف ، وهذا مايسمى التحميل الزائد للإجراء. نوضحه بالمثال الآتي:

```
//program 3-8
#include<iostream.h>
int sqar(int x)
{
return x*x;}
double sqar(double y)
        {
        return y+y;}
        int main()
        {
        int r=sqar(7);
        double rr=sqar(7.5);
        cout<<"r="<<r<<endl;
        cout<<"rr="<<rr<<endl;
        return 0;
        }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



3-1-8- قوالب الإجراءات:

تتم كتابة تعريف وحيد لقالب الإجراء ويتم بعدها توليد إجراءات منفصلة لمعالجة نوع من أنواع المعطيات المستخدمة ،عند استدعاء الإجراء ويعطى الشكل العام للقالب على النحو الآتي:

templat <class T>

T name(T var_1,T var_2,.....,T var_n)

{

جسم الاجراء

}

أي متغير في جسم الاجراء نعلن عنه بوضع حرف T قبله .

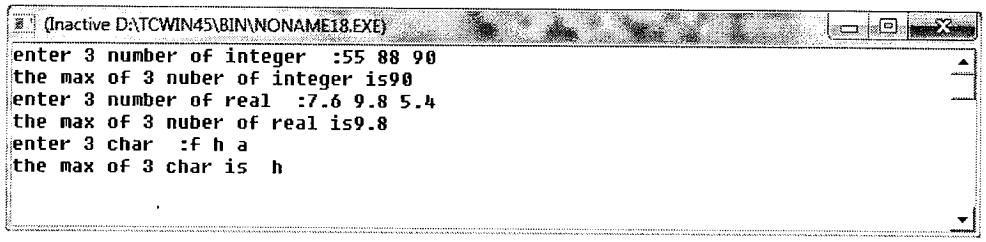
هنا تأخذ T أنواعا" متغيرة للوسطاء فعند إرسال قيمة من نوع int تستبدل كل T بالنوع الصحيح ،وعند إرسال قيمة char تستبدل كل T بالنوع المحرفي.

ضع برنامجا" يستخدم قالب لحساب أكبر القيم لثلاثة معطيات من النوع int ومن النوع double ومن النوع char .

```
//program 3-9
#include<iostream.h>
template <class T>
T maximum(T value1,T value2,T value3)
{
T max=value1;
if (value2>max)max=value2;
if(value3>max)max=value3;
return max;
}
int main()
{
int n1,n2,n3;
```

```
cout<<"enter 3 number of integer  :";
cin>>n1>>n2>>n3;
cout<<"the max of 3 nuber of integer is"<<maximum( n1,n2,n3)<<endl;
        double a1,a2,a3;
cout<<"enter 3 number of real  :";
cin>>a1>>a2>>a3;
cout<<"the max of 3 nuber of real is"<<maximum( a1,a2,a3)<<endl;
char ch1,ch2,ch3;
cout<<"enter 3 char  :";
cin>>ch1>>ch2>>ch3;
cout<<"the max of 3 char is  "<<maximum( ch1,ch2,ch3);
return 0;}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



3-1-9- الاعادة الذاتية للإجراء:

إجراء يعيد نفسه ، يمكن للإجراء أن يستدعي نفسه إذا كانت التعليمات المستخدمة في جسم الإجراء هي تعليمة استدعاء له.

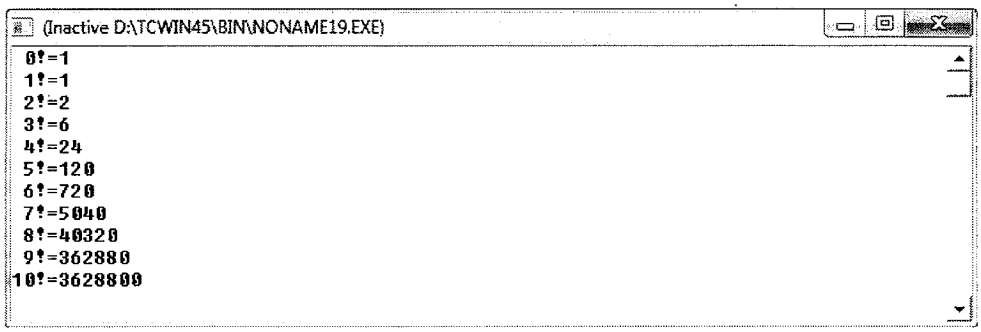
ضع برنامجا" لحساب عاملي عدد صحيح $n!$ والذي يساوي جداء الأعداد الصحيحة من الواحد وحتى العدد المرغوب في إيجادعامله n:

البرنامج:

```
//program 3-10
#include<iostream.h>
#include<iomanip.h>
long fact(long n)
{
if(n<=1) return 1;
else
 return n*fact(n-1);}
 void main()
 {
 int i;
 for(i=0;i<=10;i++)
 cout<<setw(2)<<i<<"!="<<fact(i)<<endl;
 }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كمايأتي:

ملحوظة: تستخدم setw(n) في الطباعة وذلك لإزاحة المؤشر بمقدار n .



تعريف التوابع الرياضية

توفر لنا لغة c++ وظيفة نستطيع من خلالها حساب قيمة أي تابع رياضي من أجل أية قيمة نختارها للمتغير، وذلك باستخدام #define وتكتب هذه الوظيفة على الشكل الآتي في رأس البرنامج:

معادلة التابع #define name(var)

مثل : #define y(x) x*x-2*x+1

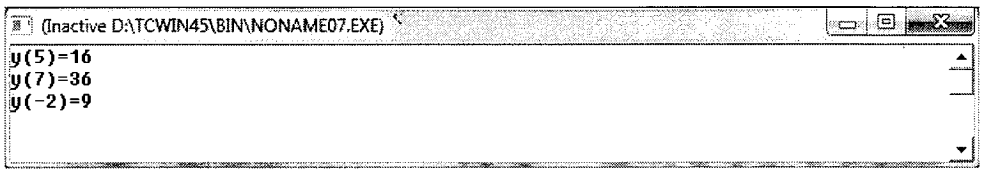
مثال:

احسب قيمة التابع $y(x)=x^2-2x+1$ في النقاط الآتية 5 و 7 و 2 - .

البرنامج:

```
//program 3-11
#include<iostream.h>
#define y(x)  x*x-2*x+1
main()
{  cout<<"y(5)="<<y(5)<<endl;;
  cout<<"y(7)="<<y(7)<<endl;;
  cout<<"y(-2)="<<y(-2);
    }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



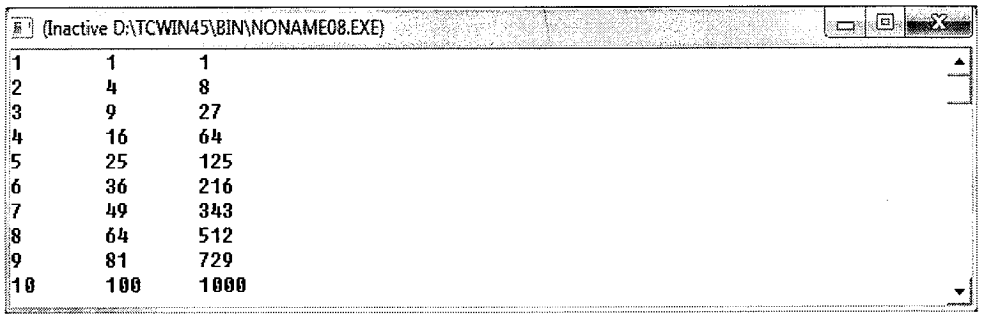
مثال:

ضع برنامجا لطباعة العدد ومربعه ومكعبه، وذلك للأعدادالطبيعية من واحد وحتى عشرة .

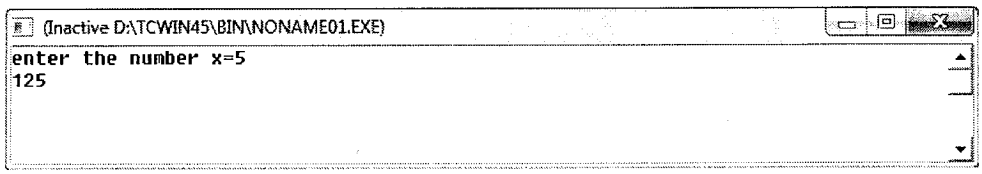
البرنامج:

```
//program 3-12
#include<iostream.h>
 #define q(x)  x*x
 #define cub(x) x*x*x
main()
{  int i;
 for(i=1;i<=10;i++)
        cout<<i<<"\t"<<q(i)<<"\t"<<cub(i)<<endl;
    }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كمايأتي:



وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



3-3- توليد الأرقام العشوائية

3-3-1- وظيفة randomize() :

إن استعمال هذه الوظيفة يؤدي إلى تتابع غير متوقع من الأعداد التي سيتم توليده باستخدام وظيفة التوليد االآتية.

4-3-2 وظيفة random(n) :

تعطي هذه الدالة أي عددا" طبيعيا" عشوائيا" بين العدد 0 والعدد n-1 .

ضع برنامجا يمثل لعبة مكونة على الشكل الآتي:

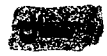
يقوم الحاسب بتخزين عدد طبيعي عشوائي محصور بين 1 و100 وعليك اكتشاف هذا العدد ويقوم الحاسب بإرشادك إذا كان العدد الذي أدخلته كبيرا أو صغيرا وفي حالة حصولك على العدد يقوم الحاسب بطباعة عدد المحاولات.

البرنامج :

```
//program 3-13
#include<iostream.h>
      #include<stdlib.h>
main()
{
int x,z,s;
 randomize();
   s=1;
 x=random(100);
md: cout<<"enter the number(1- 100)";
 cin>>z;
 if(z>x) {cout<<"the number is big";s=s+1;goto md;}
       if(z<x) {cout<<"the number is smol";s=s+1;goto md;}
    else
 { cout<<"the number is "<<z<<endl;
 cout<<"ypur tray is="<<s;}
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:

```
(Inactive D:\TCWIN45\BIN\NONAME09.EXE)
enter the number(1- 100)50
the number is bigenter the number(1- 100)25
the number is smolenter the number(1- 100)35
the number is bigenter the number(1- 100)30
the number is smolenter the number(1- 100)33
the number is bigenter the number(1- 100)32
the number is 32
ypur tray is=6
```



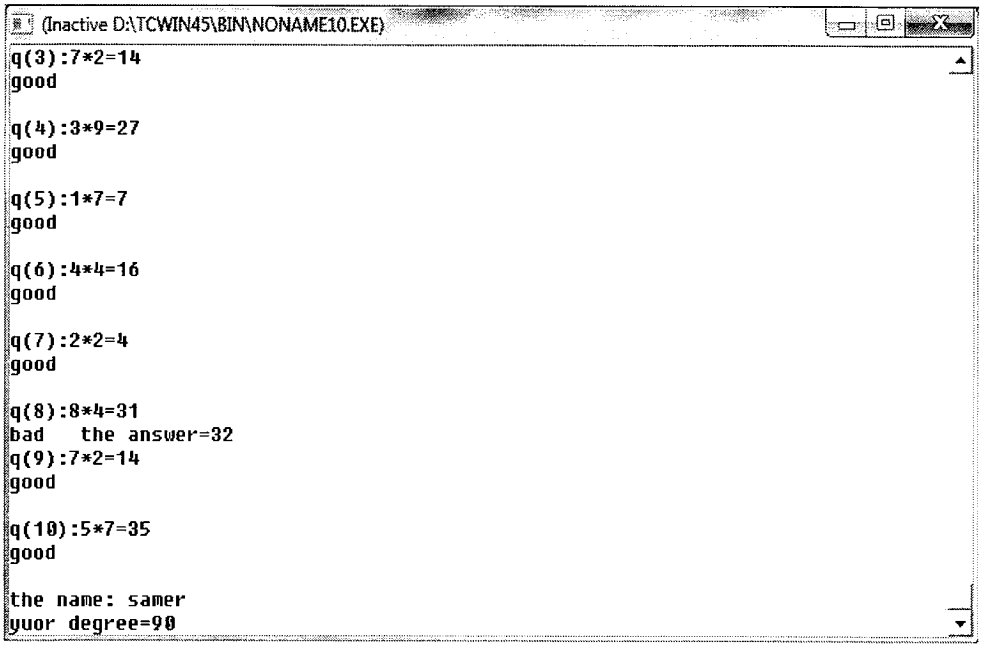
ضع برنامجا يمثل امتحان طلاب في المرحلة الابتدائية في جدول الضرب بصورة يكون الامتحان فيها مكونا" من عشرة أسئلة معطاة من الحاسب بشكل عشوائي وفي النهاية يسجل الحاسب العلامة التي أخذها الطالب علما أن علامة أي سؤال من عشر درجات:

البرنامج:

```
//program 3-14
#include<iostream.h>
#include<stdlib.h>
main()
{
  int i,s,x,y,z;
  char name[10];
  randomize();
  cout<<"enter yuor name:  ";
  cin>>name;
  s=0;
  for(i=1;i<=10;i++)
  {
   x=1+random(9);
   y=1+random(9);
        cout<<"q("<<i<<"):"<< x<<"*"<<y<<"=";
```

```
    cin>>z;
    if(z==x*y) {cout<<"good\n"; s=s+10;}
else
    { cout<<"bad   ";
    cout<<"the answer="<<x*y;}
    cout<<"\n";
    }
    cout<<"the name: "<<name<<endl;;
    cout<<"yuor degree="<<s;

}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



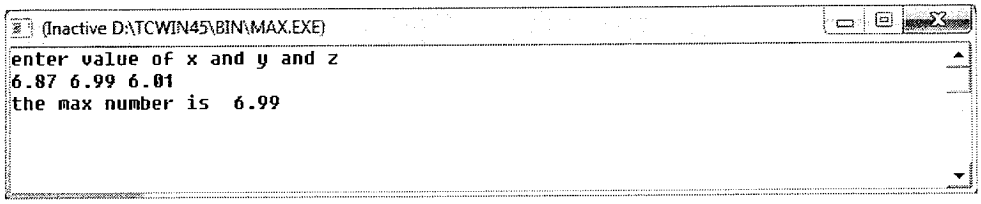
3-4-تمارين محلولة 3

1- ضع برنامجا يتضمن إجراء لإيجاد أكبر قيمة بين ثلاثة أعداد حقيقية ثم استفد من ذلك في إيجاد أكبر قيمة لثلاثة أعداد حقيقية معطاة للحاسب .

البرنامج:

```
#include<iostream.h>
float  maximum(float a,float b,float c );
main()
{
float x,y,z,max1;
cout<<"enter value of x and y and z\n";
cin>>x>>y>>z;
max1=maximum(x,y,z);
cout<<"the max number is  ";
cout<<max1;
}
float maximum(float a1,float b1,float c1)
{
float max;
max=a1;
if(b1>max)max=b1;
if(c1>max)max=c1;
return max;
}
```

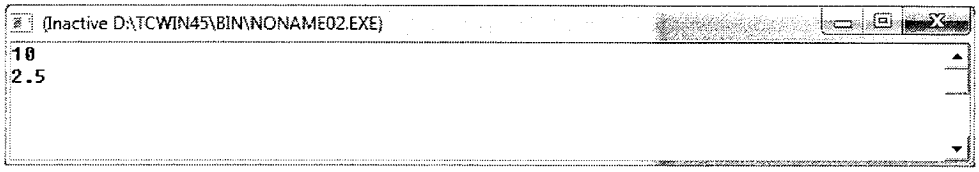
وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



2- ضع برنامجا يتضمن إجراء يقوم يضرب عددين ثم استخدم الإجراء نفسه لإيجاد حاصل قسمة هذين العددين.

```
#include <iostream.h>
int operate (int a, int b)
{
return (a*b);
}
float operate (float a, float b)
{
return (a/b);
}
int main ()
{
int x=5,y=2;
float n=5.0,m=2.0;
cout << operate (x,y);
cout << "\n";
cout << operate (n,m);
cout << "\n";}
```

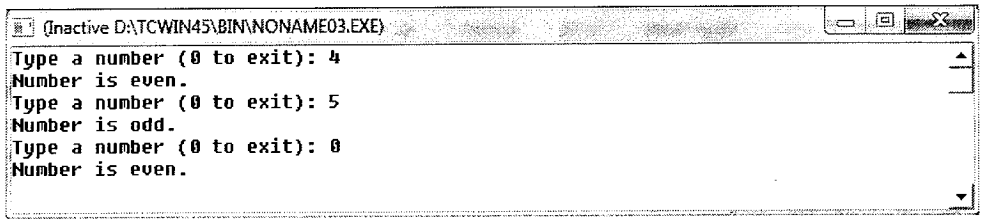
وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأي:



3- ضع برنامجا يتضمن إجرائين الأول لمعرفة إذا كان العدد المدخل عددا" فرديا" والثاني لمعرفة إذا كان العدد المدخل عددا" زوجيا" واستمر في عملية الإدخال حتى يتم إدخال العدد صفر.

```
#include <iostream.h>
void odd (int a);
void even (int a);
int main ()
{
int i;
do {
cout << "Type a number (0 to exit): ";
cin >> i;
odd (i);
} while (i!=0);
return 0;
}
void odd (int a)
{
if ((a%2)!=0) cout << "Number is odd.\n";
else even (a);
}
void even (int a)
{
if ((a%2)==0) cout << "Number is even.\n";
else odd (a);}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كمايأي:



4- لتكن لدينا المعادلة $f(x)=x^3-x-1=0$، والمطلوب

ضع برنامجا لإيجاد الجذر التقريبي لهذه المعادلة، والذي يقع في المجال [1,2] متخذا $\varepsilon = 0.05$ ، وفق طريقة تنصيف المجال، علما بأن طريقة تنصيف المجال تعطى وفق الخوارزمية الآتية.

المدخلات: a,b طرفي المجال، ε مقدار التفاوت Tolerance.

المخرجات: x_0 الجذر التقريبي.

خطوة(1): نحسب نقطة منتصف المجال [a,b] ولتكن نقطة المنتصف x_0 .

خطوة(2): نتحقق فيما إذا كان $f(x_0)=0$ ، فإذا كان محققاً يكون الجذر المضبوط هو x_0 ، ثم نذهب إلى الخطوة 5.

خطوة(3): نتحقق إذا كان $f(a).f(x_0)<0$، فإذا كان محققا نأخذ $b=x_0$ ويصبح المجال الجديد $[a,b]=[a,x_0]$، وإلا نأخذ $a=x_0$ ويصبح المجال الجديد $[a,b]=[x_0,b]$.

خطوة(4): نتحقق إذا كان $|a-b|<\varepsilon$ ، فإذا كان محققاً يكون x_0 هو الجذر التقريبي، وإلا نعود إلى الخطوة الأولى.

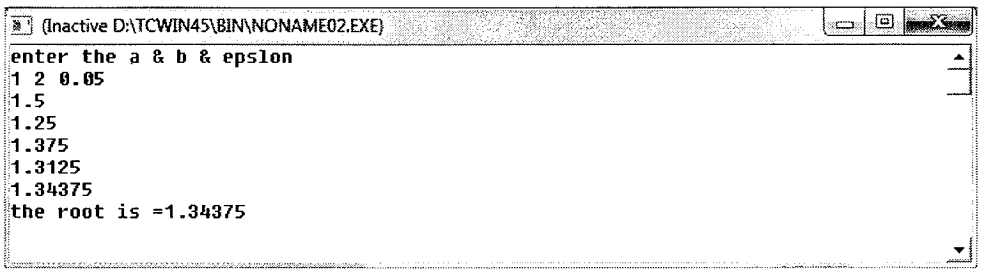
خطوة (5) : نطبع x_0

خطوة 6: النهاية

البرنامج :

```
#include<iostream.h>
#include<math.h>
#define f(x) pow(x,3)-x-1
main()
 {
 float a,b,x0,a1,b1,r,epslon;
 cout<<"enter the a & b & epslon\n";
 cin>>a>>b>>epslon;
 mm:x0=(a+b)/2;
 cout<<x0<<endl;
 if(f(x0)==0)goto gg;
       a1=f(a);b1=f(x0);
       r=a1*b1;
 if(r<0)b=x0;
        else
        a=x0;
        if(fabs(b-a)>epslon)goto mm;
        gg:cout<<"the root is ="<<x0;
        }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأي:



5- لتكن الدالة $y = x^2 e^x$ المطلوب ضع برنامجا لإيجاد مايأتي:

a. قيمة الدالة y في النقاط x=0,0.1,0.2,.....0.8,0.9,1 .

b. حساب قيمة التكامل العددي وفق طريقة سمبسون:

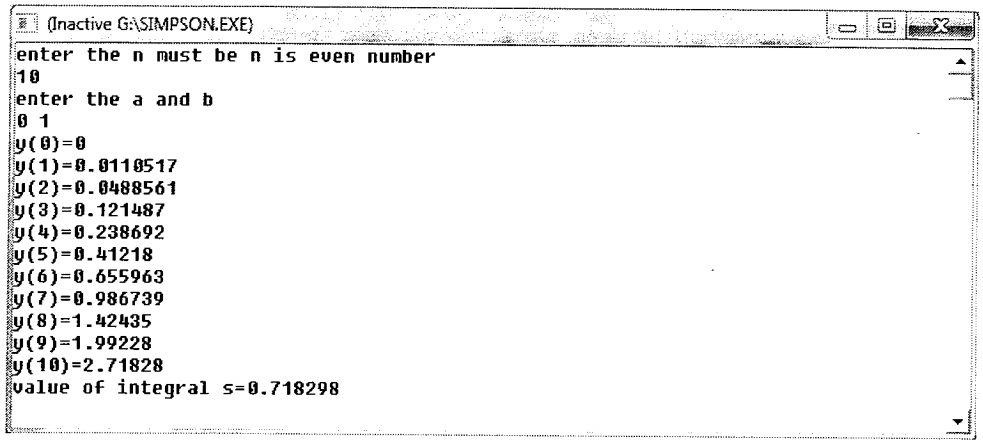
$$I = \int_0^1 x^2 e^x dx$$

علما أن عدد التجزئة n=10 .

البرنامج:

```
include<iostream.h>
#include<math.h>
#define f(x) x*x*exp(x)
main()
{ int n,i;
float y[20],a,b,h,s,s1,s2,m,x;
cout<<"enter the n must be n is even number\n";
 cin>>n;
            cout<<"enter the a and b\n";
                  cin>>a>>b;
                  h=(b-a)/n;
 for(i=0;i<=n;i++){
 x=i*h;
 y[i]=f(x);
 cout<<"y("<<i<<")="<<y[i]<<endl;}
 m=n/2;
 s1=0;
      for(i=1;i<m;i++)
            s1=s1+2*y[2*i];
      s2=0;
      for(i=1;i<=m;i++)
            s2=s2+4*y[2*i-1];
      s=(h/3.)*(y[0]+y[n]+s1+s2);
      cout<<"value of integral s="<<s;
      }
```

وبعد تنفيذ البرنامج تظهر نتائج البرنامج وفق الآتي:

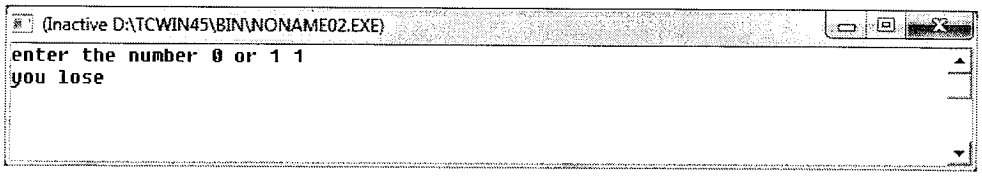


6- ضع برنامجا يمثل لعبة تعتمد على رمي قطعة نقود وذلك بأن نعتمد العدد 0 من أجل الطرة و العدد 1 من أجل النقش ويسأل الحاسب ماهي نتيجة الرمية فإذا كان الجواب صحيح فيطبع لك الحاسب you win وإلا يطبع عبارة you lose .

البرنامج:

```
#include<iostream.h>
#include<stdlib.h>
main()
{
 int x,y,i;
 randomize();
       x=random(2);
       cout<<"enter the number 0 or 1 ";
       cin>>y;
if(y==x)cout<< "you win";
else
cout<<"you lose";
  }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



تمارين 3

1- اكتب إجراء لحساب !n ثم استفد من هذا الإجراء لحساب القيمة التقديرية للثابت الرياضي e باستخدام الصيغة :

$$e=1+1/1!+1/2!+1/3!+.....$$

2- باستخدام طريقة نيوتن رافسون لإيجاد أحد الجذور التقريبية للمعادلة:

$$x^3+2x-10=0$$

والذي يقع ضمن المجال [1,2] متخذاً التفاوت $\varepsilon = 0.05$.

علما" أن إيجاد الجذر وفق طريقة نيوتن يعطى بالخوارزمية الآتية:

المدخلات: a,b طرفي المجال، ε مقدار الخطأ .

المخرجات: الجذر التقريبي x_n .

خطوة 1: نتحقق من الشرطين السابقين فإذا كان الشرط الأول محقق نختار $x_0=a$ وإذا كان الشرط الثاني محقق نختار $x_0=b$.

خطوة 2: نضع n=0 .

خطوة 3: نوجد x_{n+1} من العلاقة: $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$.

خطوة 4: نتحقق من العلاقة: $| x_{n+1} - x_n | < \varepsilon$ فإذا كانت محققة ننفذ الخطوة 6 .

خطوة 5: نعطي n تزايد بمقدار 1 أي n=n+1 ونذهب إلى الخطوة 3 .

خطوة 6: نطبع x_{n+1} .

خطوة 7: النهاية .

3-لتكن الدالة y=f(x) معطاة بجدول من القيم يعطى التكامل المحدد $S = \int_a^b f(x)dx$ وفق طريقة سيمبسون بالعلاقةالآتية:

$$I \approx \frac{h}{3}[f_0 + 4f_1 + 2f_2 + 4f_3 + 2f_4 + ... + 2f_{n-2} + 4f_{n-1} + f_n]$$

حيث n عدد زوجي:

يمكننا تمثيل هذه الطريقة بالخوارزمية الآتية:

خوارزمية: إيجاد القيمة التقريبية للتكامل المحدد بطريقة سيمبسون:

المدخلات: n عددا زوجيا لنقاط التجزئة لمجال التكامل.

y_i قيم الدالة عند x_i إذ إن i=0,1,2,....n

المخرجات: S قيمة التكامل.

خطوة 1: نحسب مقدار الخطوة h=(b-a)/n.

خطوة 2: نحسب m=n /2 .

خطوة 3: نضع $S_1=0$ ثم نحسب مجموع قيم الدالة :

For i=1,2,.....m-1

$S_1=S_1+2*y_{2i}$

خطوة 4: نضع $S_2=0$ ثم نحسب مجموع قيم الدالة :

For i=1,2,.....m

$S_2=S_2+4*y_{2i-1}$

خطوة 5: نحسب قيمة التكامل التقريبية:

$S=(h/3)*(y_0+y_n+S_1+S_2)$

خطوة 6: نطبع قيمة S

خطوة 7: النهاية

المطلوب ضع برنامجا لإيجاد قيمة التكامل وفق الخوارزمية السابقة.

4-ضع برنامجا يمثل رمي حجرتي نرد فإذا كان مجموع الرقمين الناتجين من خلال الرمية أكبر أو يساوي 9 يعطيك الحاسب عبارة you have good shanc وإذا كان حاصل مجموع الرقمين نتيجة الرمية الواحدة أصغر من 9 يطبع لك الحاسب العبارة

. you have bad shanc

5-اكتب إجراء سطريا" لإيجاد الجذر التربيعي لأي عدد ثم استفد من ذلك في إيجاد الجذر التربيعي لإي عدد معطى إلى الحاسب.

الفصل الرابع

استخدام المصفوفات في اللغة c++

4-1 تعريف المصفوفة:

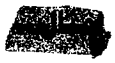
المصفوفة هي مجموعة من البيانات التي تتشابه في النوع ولها اسم مشترك وفي أثناء التعامل مع المصفوفات يجب الإعلان عنها .

4-2- المصفوفة الأحادية:

هي جدول مؤلف من n عنصرا ويتم الإعلان عن هذه المصفوفة على النحو الآتي:

Type name[n];

إذ إن **type** هو نوع المصفوفة و name اسم المصفوفة و n عدد عناصر هذه المصفوفة فإذا كنت تتعامل مع درجات الطلاب ولنفترض أن درجات الطلاب أعداد صحيحة يتم الإعلان عنها على النحو الآتي: **int** degre [30] .



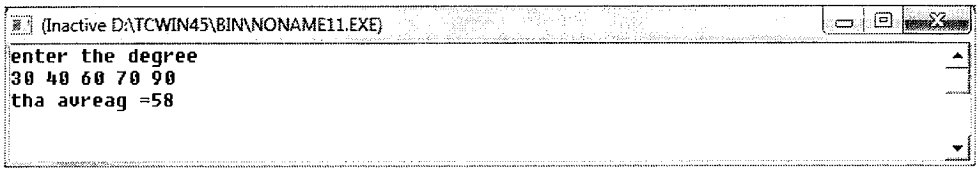
ضع برنامجا لإيجاد المتوسط الحسابي لخمسة أعداد .

البرنامج:

```
//program 4-1
#include<iostream.h>
main()
{int i;
float degre[5];
float avr,s;
      cout<<"enter the degree\n";
s=0;
  for(i=0;i<5;i++)
```

```
{
    cin>>degre[i];
  s=s+degre[i];
  }
avr=s/5;
      cout<<"tha avreag ="<<avr;
      }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



ملحوظة: في أثناء التعامل مع المصفوفات يجب عدم تجاوز حدود المصفوفة.

4-3- تهيئة المصفوفة عند الإعلان عنها:

من الممكن أن تقوم بتهيئة المصفوفة عند الإعلان عنها، فإذا كنت تعرف محتوياتها مسبقا يمكننا تهيئتها على النحو الآتي:

```
Int marks[5]={10,9,12,8,7};
```

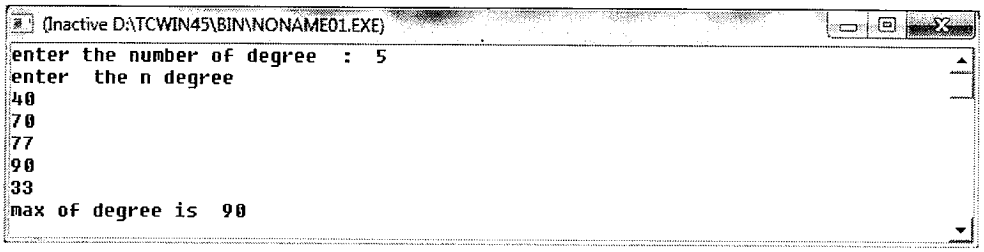
هنا تعلن عن مصفوفة مكونة من خمسة عناصر عددية **int** وفي الوقت نفسه سيقوم المترجم بوضع الأرقام المذكورة في عناصر المصفوفة فيضع الرقم **10** في العنصر الأول (العنصر رقم صفر) والرقم **9** في العنصر الثاني وهكذا .

ضع برنامجا لطباعة أكبر عدد افي جدول مؤلف من n عنصرا .

البرنامج:

```
//program 4-2
#include<iostream.h>
 main()
 {
 int a[10],max,n,i;
 cout<<"enter the number of degree  :  ";
 cin>>n;
 cout<<"enter  the n degree \n";
 for(i=0;i<n;i++)
 cin>>a[i];
 max=a[0];
 for(i=1;i<n;i++)
{ if(max<a[i])
 max=a[i];}
 cout<<"max of degree is  "<<max;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:

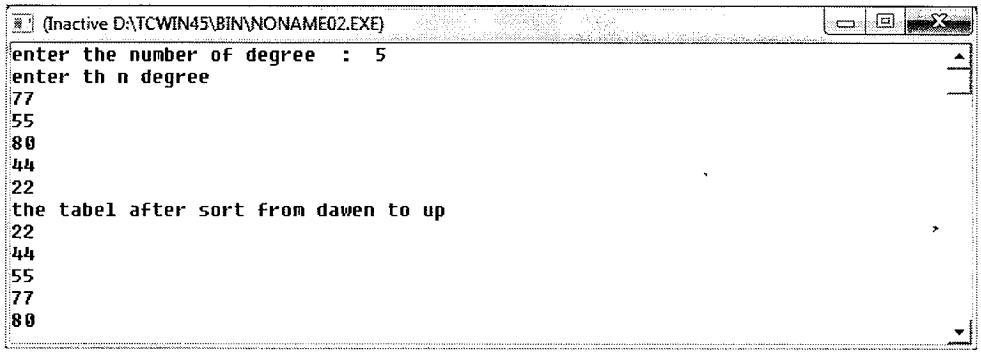


ضع برنامجا لترتيب جدول مؤلف من n عنصرا ترتيبا تصاعديا.

البرنامج:

```
//program 4-3
#include<iostream.h>
        main()
 {
 int a[10],n,i,j,m;
 cout<<"enter the number of degree  :  ";
 cin>>n;
 cout<<"enter th n degree \n";
 for(i=0;i<n;i++)
 cin>>a[i];
  for(i=0;i<n-1;i++)
{
for(j=i+1;j<n;j++)
if(a[i]>a[j])
{m=a[i];a[i]=a[j];a[j]=m;}
}
 cout<<"the tabel after sort from dawen to up\n";
 for(i=0;i<n;i++)
 cout<<a[i]<<"\n";
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



4-4- الإعلان عن السلاسل الحرفية

عندما ترغب في استخدام مصفوفة حروف فأنك تعلن عنها كما تعلن عن بقية المصفوفات الأخرى ولكن تستخدم نوع البيانات **char** فمثلا نكتب:

```
char name[80];
```

هنا تعلن عن المصفوفة name التي تتسع ل **79** حرفا إلحظ الرقم الذي تذكره بين القوسين يجب أن يحدد عدد الحروف التي تريد وضعها في المصفوفة زائدا واحد لأن لغة **C++** تفترض أن المصفوفة تنتهي بحرف الصفر.

وبالتالي يمكن تهيئة مصفوفة الحروف عند الإعلان عنها كما يأتي:

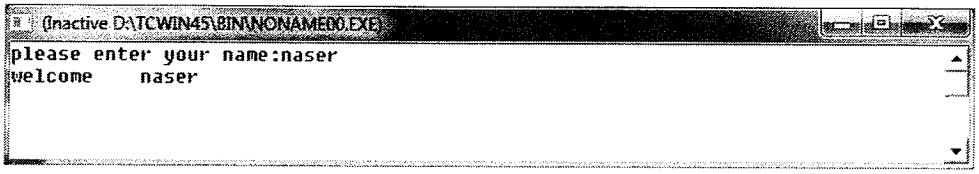
```
Char name[ ]="mahmoud osman";
```

ضع برنامجا لإدخال اسم الطالب وطباعته.

البرنامج:

```
//program 4-4
#include<iostream.h>
  main()
{
char name[80];
cout<<"please enter your name:";
cin>>name;
cout<<"welcome    "<<name;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



4-4-1- بعض الوظائف الهامة التي يمكن تطبيقها على السلاسل المحرفية

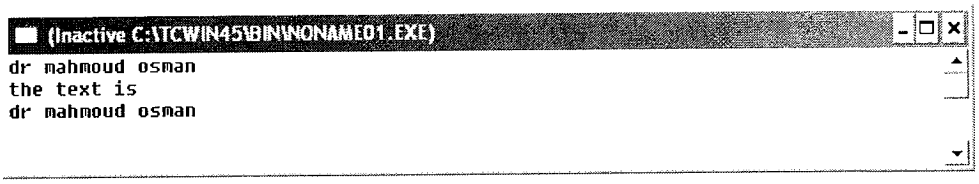
- الوظيفتان : getchar() و putchar()

الوظيفة getchar() : تستخدم لإدخال محرف من لوحة المفاتيح .

الوظيفة putcha(): تقوم بطباعة الحرف المدخل وهذا مانوضحه بالمثال الآتي:

```
//program 4-5
   #include<iostream.h>
   #include<stdio.h>
   void main()
   {
     char word[50];
   int i,ch,n;
   n=0;
     do{
     ch=getchar();
     word[n]=ch ;
     n++;}while (ch!='\n');
     cout<<"the text is\n";
     for(i=0;i<n;i++)
     putchar(word[i]);}
```

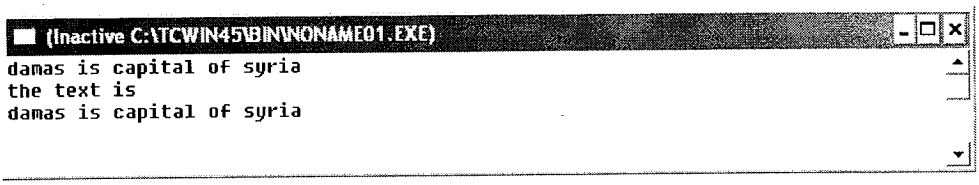
وتظهر شاشة الإخراج كما يأتي:



الوظيفتان: **cin.get()** و **cou.put** يعملان عمل الوظيفتين السابقتين نفسه وهذا ما يوضحه البرنامج الآتي:

```
//program 4-6
   #include<iostream.h>
   #include<stdio.h>
   void main()
   {
     char word[50];
   int i,ch,n;
   n=0;
     do{
     ch=cin.get();
     word[n]=ch ;
     n++;}while (ch!='\n');
     cout<<"the text is\n";
     for(i=0;i<n;i++)
     cout.put(word[i]);
     }
```

وتظهر شاشة الإخراج كما يأتي:



- االوظيفة: **cin.getline**(str,n,ch); تسمح هذه الوظيفة بإدخال نص إذ:

str تمثل الصف المحرفي الذي يخزن في النص.

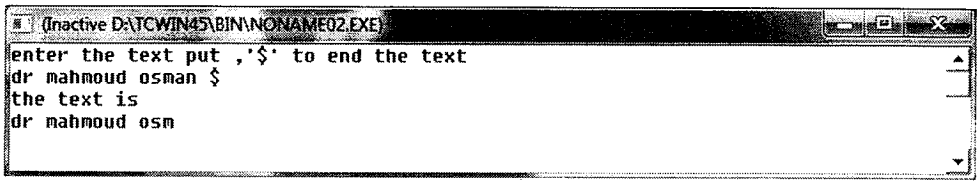
n عدد عناصر الصف المحرفي.

ch المحرف الناهي الذي يحدد نهاية النص الذي يخزن في **str** .

البرنامج:

```
//program 4-7
   #include<iostream.h>
   #include<string.h>
   void main()
   {
     char word[50];
   cout<<"enter the text put ,'$' to end the text\n";
   cin.getline(word,15,'$');
   cout<<"the text is\n";
   cout<<word<<endl;
    }
```

وتظهر شاشة الإخراج كما يأتي:



ملحوظة: في اثناء استخدام وظائف السلاسل الحرفية يجب إدخال الملف الرأسي:

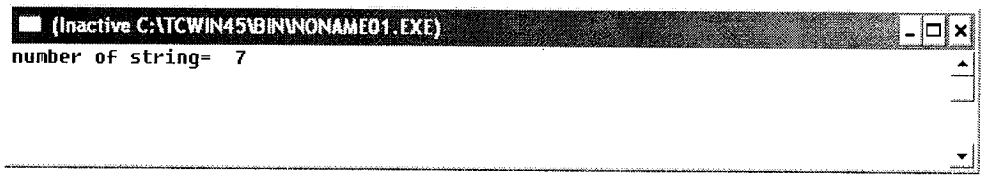
#include<string.h>

الوظيفة :**strln**(s) يعطي طول السلسلة الحرفية s

المثال يوضح استخدام هذه الوظيفة:

```
//program 4-8
   #include<iostream.h>
   #include<string.h>
   void main()
   { int i;
     char name[]="shahine";
   cout<<"number of string= "<<strlen(name)<<endl;
     }
```

وتظهر النتائج كما يأتي:



- الوظيفة: **strcmp(s1,s2)**

الوظيفة **strcmp**(s1,s2): تستخدم لمقارنة السلسلة s1 مع السلسلة s2 حرفا حرفا فإذا كانتا متطابقتين يعطينا العدد 0 وإذا كانت s1>s2 يعطينا عددا" موجبا" وإذا كانت s1< s2 يعطينا عددا" سالبا".

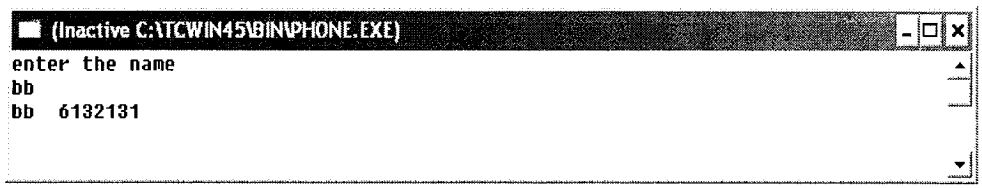
ضع برنامجا يمثل دليل الهاتف بحيث عندما تطلب الاسم يعطيك الحاسب رقم الهاتف:

البرنامج:

```
//program 4-9
   #include<iostream.h>
   #include<string.h>
   main()
   {
```

```
 long a[3]={417196,6132131,311198};
 char b[3][10]={"aa","bb","cc"};
int i;
char name[10];
cout<<"enter the name\n";
cin>>name;
for(i=0;i<3;i++)
{ if(strcmp(name,b[i])= =0){cout<<name<<"  "<<a[i];goto mm;}}
cout<<"not found";
mm:
}
```

وتظهر نتائج الإخراج كما يأتي:



تمرين:

ضع برنامجا يمثل دليل الأسعار في مخزن، وعندما تعطي اسم السلعة يعطيك الحاسب سعر هذه السلعة.

- **الوظيفة: strcat(s1,s2)**

تقوم هذه الوظيفة بإضافة السلسلة s2 في نهاية السلسلة s1 ويجب أن نعلم أن طول s1 يكون كافيا ليضم s1 مع s2:

المثال الآتي يوضح استخدام هذه الوظيفة:

```
//program 4-10
   #include<iostream.h>
   #include<string.h>
   main()
   {
   char s1[]="mahmoud ";
   char s2[]="osman";
   strcat(s1,s2);
   cout<<s1<<endl;
   }
```

وتظهر النتائج كمايأتي: mahmoud osman .

- الوظيفة:**strcpy**(s1,s2)

تقوم هذه الوظيفة بنسخ السلسلة s2 مكان السلسلة s1 .

البرنامج الآتي يوضح الوظائف السابقة:

```
//program 4-11
#include<iostream.h>
 #include<string.h>
main()
{
char string1[80];
char string2[80];
int result,choice;
cout<<"enter the firset string    :";
cin>>string1;
cout<<"enter the second string     :";
```

```
cin>>string2;
cout<<"enter the function you want\n" ;
cout<<"1-strcpy\n";
cout<<"2-strcat\n";
cout<<"3-strcmp\n";
cout<<"enter your choice: ";
cin>>choice;
switch(choice)
{
case 1:
cout<<"copy second string to first\n";
strcpy(string1,string2);
cout<<string1;
break;
case 2:
cout<<"nmerging the two strings \n";
strcat(string1,string2);
cout<<string1;
break;
case 3:
cout<<"ncomparing the two strings\n";
result=strcmp(string1,string2);
if( result==0)
cout<<"the two stringes are identical";
else
cout<<"the two stringes are different";
break;
```

```
default:
cout<<"pleas enter a number 1-3";
}
}
```

4-5 المصفوفات متعددة الأبعاد

اذا أردت إدخال عشرة مواد ل 30 طالبا يمكنك إنشاء مصفوفة ذات بعدين ويتم الإعلان عنها كما يأتي:

```
Int student[30][10];
```

هنا تعلن عن المصفوفةstudent التي تتكون من30 عنصر أساسي وفي أي عنصر أساسي يوجد 10 عناصر فرعية في هذه الحالة ستتسع المصفوفة لتخزين 10 درجات لأي طالب من الطلبة الثلاثين فمثلا لإدخال درجات الطالب الأول نكتب:

```
Student[0][0]=87;
Student[0][1]=90;
Student[0][2]=42;
```

هنا نذكر العنصر الأساسي هو الطالب رقم صفر ثم نذكر العنصر الفرعي أي المادة رقم 1 أو رقم 2 وعلى هذا إذا أردنا إدخال العنصر رقم 22 فأننا نكتب :

```
Student[21][0]=98;
Student[21][1]=66;
```

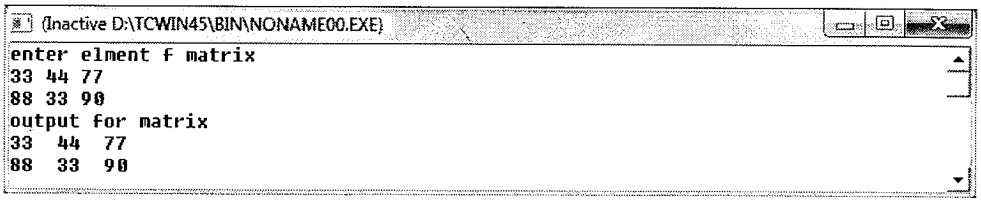
[redacted]

ضع برنامجا لإدخال علامة طلابين وكل طالب له خمس مقررات.

البرنامج:

```
//program 4-12
 #include<iostream.h>
            main()
      {
      int i,j;
      int a[3][5];
            cout<<"enter elment f matrix\n";
      for(i=0;i<2;i++)
      {
      for(j=0;j<3;j++)
      cin>>a[i][j];}
  cout<<"output for matrix\n";
      for(i=0;i<2;i++){
      for(j=0;j<3;j++)
      cout<<a[i][j]<<"  ";
      cout<<"\n";
      }

      }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



4-5-1- مصفوفة السلاسل :

يتم التصريح عنها بالشكل الآتي:

char name[n1][n2];

فالتصريح:

char name[4][20];

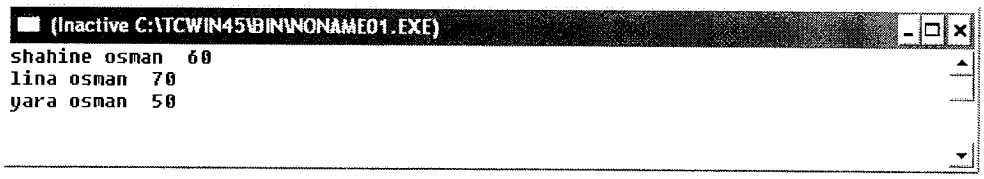
يعني التصريح عن مصفوفة سلاسل اسمها name ومؤلفة من 4 أسطر وكل سطر يتضمن سلسلة لايزيد عدد محارفها عن 19 محرف.

ضع برنامجا لإدخال اسماء ثلاثة طلاب مع درجاتهم الامتحانية.

البرنامج:

```
//program 4-13
#include<iostream.h>
#include<string.h>
void main()
{ int i;
  char name[][20]={"shahine osman","lina osman","yara osman"};
int degre[]={60,70,50};
 for(i=0;i<3;i++)
 cout<<name[i]<<"  "<<degre[i]<<endl;
 }
```

وتظهر شاشة الإخراج كمايأتي:

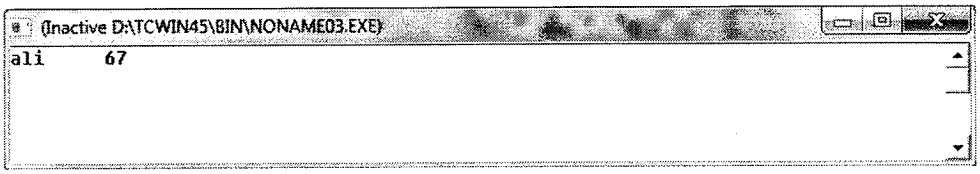


تقدم إلى امتحان مقرر البرمجة ثلاثة طلاب والمطلوب وضع برنامج لطباعة أسماء الطلاب الناجحين علما أن علامة النجاح هي 50 وما فوق.

البرنامج:

```
//program 4-14
#include<iostream.h>
 main()
{ char a[3][10]={"ahmad","ali","nader"};
  int b[3]={40,67,50};
  int i;
  for(i=0;i<3;i++)
  if(b[i]>=60)cout<<a[i]<<"\t"<<b[i]<<endl;
  }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كمايأتي:



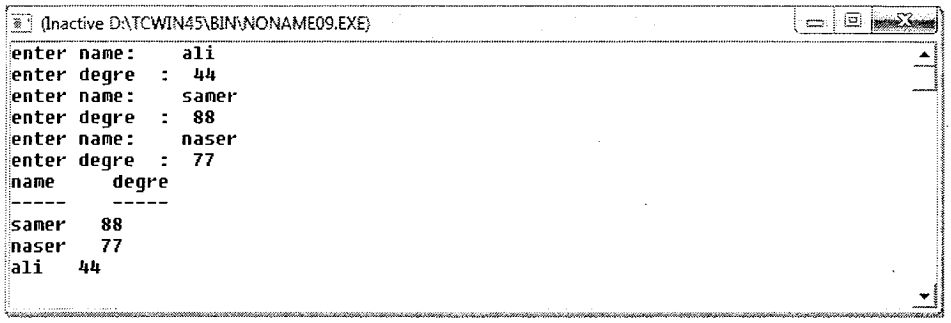
ضع برنامجا لترتيب علامات ثلاثة طلاب ترتيبا تنازليا.

البرنامج:

```
//program 4-15
#include<iostream.h>
 #include<string.h>
main()
```

```
{
char name[10][40];
int degre[40];
        int i,j,z;
 char z1[10];
for(i=0;i<3;i++)
{
cout<<"enter name:    ";
cin>>name[i];
cout<<"enter degre  :  ";
cin>>degre[i];
}
  for(i=0;i<2;i++)
  {
  for(j=i+1;j<3;j++)
  if(degre[i]<degre[j])
  { z=degre[i];degre[i]=degre[j];degre[j]=z;
        strcpy(z1,name[i]);strcpy(name[i],name[j]);strcpy(name[j],z1); }
                }
cout<<"name"<<"\t"<<" degre\n";
cout<<"-----\t"<<" -----\n";
for(i=0;i<3;i++)
cout<<name[i] <<"   "<<degre[i]<<endl;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



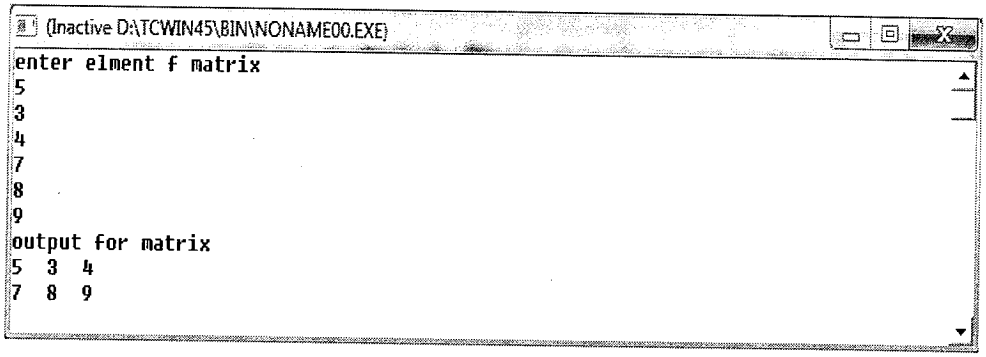
ضع برنامجا لإدخال مصفوفة حسب الأعمدة وطباعتها حسب الأسطر.

البرنامج:

```
//program 4-16
#include<iostream.h>
main()
{
int i,j,m,n;
int a[10][10];
cout<<"enter the m , n ";
cin>>m>>n;
cout<<"enter the elment of matrix\n";
for(j=0;j<n;j++){
for(i=0;i<m;i++)
cin>>a[i][j];
}
for(i=0;i<m;i++)
{
  for(j=0;j<n;j++)
  cout<<a[i][j]<<"  ";
```

```
cout<<"\n";
}
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



لدينا مصفوفة من المرتبة MxN والمطلوب وضع برنامج لإيجاد مايأتي:

1- مجموع عناصر هذه المصفوفة.

2- أكبر العناصر في هذه المصفوفة.

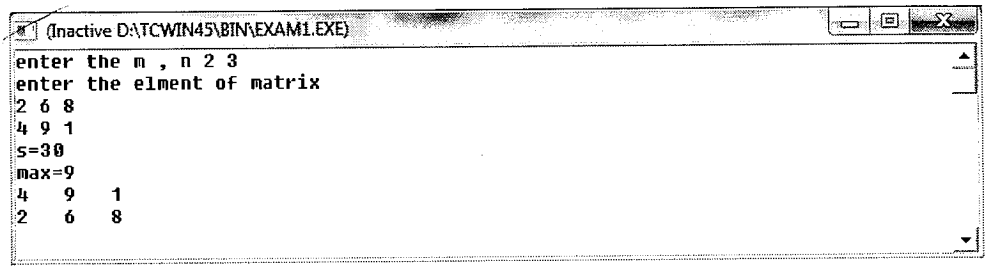
3-استبدال السطر الأول بالسطر الأخير ثم طباعة المصفوفة بعد الاستبدال.

البرنامج:

```
//program 4-17
   #include<iostream.h>
   main()
   {
   int i,j,m,n,s,max,t;
   int a[10][10];
```

```
cout<<"enter the m , n ";
cin>>m>>n;
cout<<"enter the elment of matrix\n";
s=0;max=-12;
for(i=0;i<m;i++)
for(j=0;j<n;j++){
cin>>a[i][j];
s=s+a[i][j];
if(max<a[i][j])max=a[i][j];}
cout<<"s="<<s<<endl;;
cout<<"max="<<max<<endl;
for(j=0;j<n;j++){
t=a[0][j];a[0][j]=a[m-1][j];a[m-1][j]=t;}
for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
cout<<a[i][j]<<"   ";
cout<<"\n";
}}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



تمرين:

لتكن A مصفوفة من المرتبة N والمطلوب ضع برنامجا لإيجاد مايأتي:

1- مجموع السطر الأول والسطر الأخير.

2- المتوسط الحسابي للقطر الرئيسي .

يتنافس على عضوية مجلس الشعب 5 أشخاص في أربع مناطق والمطلوب

ضع برنامجا لما يلي:

1- طباعة عدد المنتخبين في جميع المناطق.

2- طباعة عدد المنتخبين في كل منطقة.

3- طباعة عدد الأصوات التي أخذها كل منتخب.

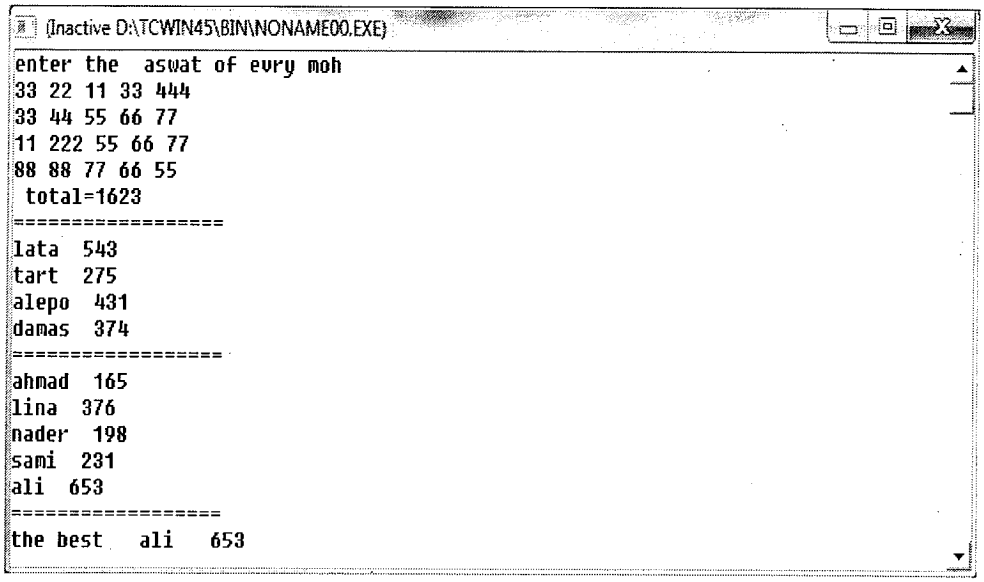
4- طباعة اسم الفائز .

البرنامج:

```
//program 4-18
#include<iostream.h>
        main()
  {
  char name[5][10]={"ahmad","lina","nader","sami","ali"};
  char moh[4][10]={"lata","tart","alepo","damas"};
   int i,j,s,s1,s2,max,t,v[10],a[10][10];
        cout<<"enter the  aswat of evry moh\n";
   s=0;
   for(i=0;i<4;i++){
        for(j=0;j<5;j++){
```

```
    cin>>a[i][j];
    s=s+a[i][j];
} }
    cout<<" total="<<s<<"\n";
     cout<<"==================\n";
    for(i=0;i<4;i++)
    {s1=0;
for(j=0;j<5;j++)
s1=s1+a[i][j];
    cout<<moh[i]<<"  "<<s1<<endl;}
    cout<<"==================\n";
                          for(j=0;j<5;j++)
{s2=0;
for(i=0;i<4;i++)
s2=s2+a[i][j];
v[j]=s2;
    cout<<name[j]<<"  "<<s2<<endl;}
          cout<<"==================\n";
 max=-3000;
 for(i=0;i<5;i++) {
 if(max<v[i]) { max=v[i];t=i;} }
     cout<<"the best"<<"   "<<name[t]<<"   "<<max;
 }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



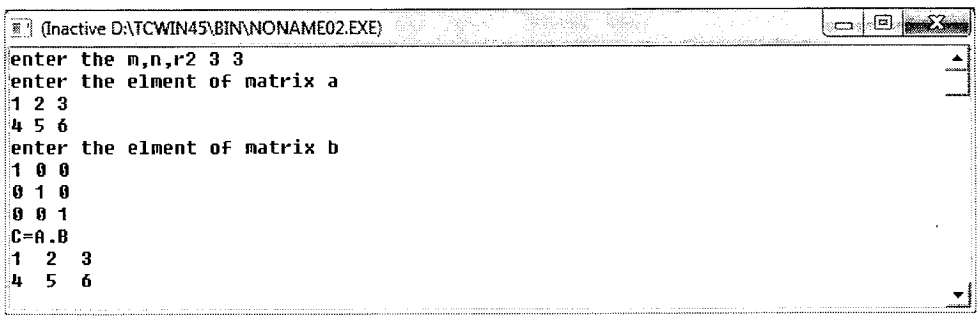
ضع برنامجا لضرب مصفوفتين.

البرنامج:

```
//program 4-19
#include<iostream.h>
    main()
{
int a[10][10],b[10][10],c[10][10];
int i,j,k,m,n,r;
cout<<"enter the m,n,r";
cin>>m>>n>>r;
  cout<<"enter the elment of matrix a\n";
for(i=0;i<m;i++)
{for(j=0;j<n;j++)
cin>>a[i][j];
}
```

```
  cout<<"enter the elment of matrix b\n";
for(i=0;i<n;i++)
{for(j=0;j<r;j++)
cin>>b[i][j];
}
for(i=0;i<m;i++)
{
for( j=0;j<r;j++)
{ c[i][j]=0;
for(k=0;k<n;k++)
c[i][j]=c[i][j]+a[i][k]*b[k][j];
}}
cout<<"C=A.B\n";
for(i=0;i<m;i++)
{
  for( j=0;j<r;j++)
            cout<<c[i][j]<<"  ";
            cout<<"\n";
     }
            }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



ضع برنامجا لحل جملة معادلات خطية AX=B وذلك بتحليل المصفوفة A إلى جداء مصفوفتين A=L.U حيث L مصفوفة مثلثية سفلية و U مصفوفة مثلثية علوية.

يعطى تحليل المصفوفة A وفق العلاقات الآتية:

$$\begin{bmatrix} a_{11}a_{12}.....a_{1n} \\ a_{21}a_{22}.....a_{2n} \\ . \\ . \\ a_{n1}a_{n2}.....a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11}0.....0 \\ l_{21}l_{22}0.....0 \\ . \\ . \\ l_{n1}l_{n2}.....l_{nn} \end{bmatrix} \begin{bmatrix} u_{11}u_{12}.....u_{1n} \\ 0u_{22}.....u_{2n} \\ 00 \\ 0 \\ 00000.....u_{nn} \end{bmatrix}$$

وفي حالة $u_{11}=u_{22}=.....=u_{nn}=1$ تعطى عناصر المصفوفة L ,و المصفوفة U بالدساتير الآتية:

$$l_{ij} = a_{ij} - \sum_{s=1}^{j-1} l_{is}u_{sj}; j \le i, i = 1,2,.....,n$$

$$u_{ij} = \frac{a_{ij} - \sum_{s=1}^{i-1} l_{is}u_{sj}}{l_{ii}}; j > i, j = 2,3,...,n$$

نلحظ أن $l_{11} = a_{11}, l_{21} = a_{21},.....l_{n1} = a_{n1}$ أي إن العمود الأول في المصفوفة L يساوي العمود الأول من المصفوفة A .

وكذلك : $u_{11} = \frac{a_{11}}{l_{11}}, u_{12} = \frac{a_{12}}{l_{11}},.....u_{1n} = \frac{a_{1n}}{l_{nn}}$ أي أن عناصر السطر الأول من المصفوفة U تساوي عناصر السطر الأول من المصفوفة A، مقسومة على l_{11} أي إن $u_{1j} = \frac{a_{1j}}{l_{11}}; j = 1,2,.....n$.

ويمكننا تمثيل المعطيات السابقة وفق الخوارزمية الآتية:

المدخلات: n عدد المعادلات في الجملة.

a_{ij} عناصر مصفوفة المعاملات حيث $i=1,2,....n$ & $j=1,2,....n$.

bi عمود الثوابت إذ إن $i=1,2,.....n$.

المخرجات : x_i حل الجملة حيث $i=1,2,....n$.

خطوة 1: نعطي عناصر المصفوفة L و U القيمة صفر.

for i=1,2,....n

for j=1,2,.....n

$L_{ij}=u_{ij}=0$

خطوة 2: نشكل عناصر العمود الأول في L:

For i=1,2,.....n

$L_{i1}=a_{i1}$

خطوة 3: نشكل عناصر السطر الأول في U عدا العنصر الأول:

For j=1,2,3,.....n

$U_{1j}=a_{1j}/L_{11}$

خطوة 4: نشكل عناصر القطر الرئيسي في U :

For i=1,2,.....n

$U_{ii}=1$

خطوة 5: نشكل بقية العناصر في L و U من العلاقتين:

$$l_{ij} = a_{ij} - \sum_{s=1}^{j-1} l_{is} u_{sj}; j \le i, i = 1,2,.....,n$$

$$u_{ij} = \frac{a_{ij} - \sum_{s=1}^{i-1} l_{is} u_{sj}}{l_{ii}}; j > i, j = 2,3,...,n$$

خطوة 6: نحل جملة المعادلات Ly=B

خطوة 7: نحل جملة المعادلات Ux=y

خطوة 8: نطبع الحل x_i حيث i=1,2,.....n

خطوة 9: النهاية.

البرنامج:

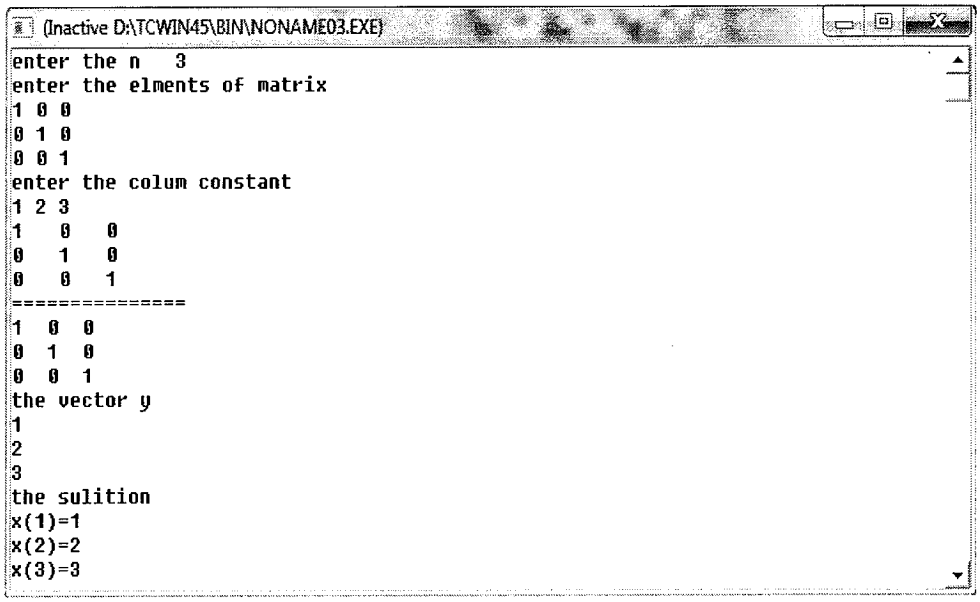
```
//program 4-20
#include<iostream.h>
        main()
{
  int n,i,j,s,k;
  float a[10][10],l[10][10],u[10][10],w,b[10],s1,s2,y[10],x[10];
        cout<<"enter the n   ";
  cin>>n;
        cout<<"enter the elments of matrix\n";
        for(i=1;i<=n;i++ )
        for(j=1;j<=n;j++)
        cin>>a[i][j];
        cout<<"enter the colum constant\n";
        for(i=1;i<=n;i++)
        cin>>b[i];
```

```
    for(i=1;i<=n;i++){
    for(j=1;j<=n;j++){
    u[i][j]=0.0;l[i][j]=0.0;}}
    for(i=1;i<=n;i++)
    u[i][i]=1.;
    for(i=1;i<=n;i++)
    l[i][1]=a[i][1];
    for(j=1;j<=n;j++)
    u[1][j]=a[1][j]/l[1][1];
    for(i=2;i<=n;i++){
    for(j=2;j<=n;j++){
    if(j<=i){
w=0.0;
    for(s=1;s<=j-1;s++)
    w=w+l[i][s]*u[s][j];
    l[i][j]=a[i][j]-w;
}
    else
    {w=0.0;
     for(s=1;s<=i-1;s++)
w=w+l[i][s]*u[s][j];
    u[i][j]=(a[i][j]-w)/l[i][i];}
    }}
    for(i=1;i<=n;i++)
    {
    for(j=1;j<=n;j++)
    cout<<l[i][j]<<"   ";
```

```
    cout<<"\n";
    }
    cout<<"==============\n";
        for(i=1;i<=n;i++)
    {
    for(j=1;j<=n;j++)
    cout<<u[i][j]<<"  ";
    cout<<"\n"; }
    y[1]=b[1]/l[1][1];
for(i=2;i<=n;i++)
    {  s1=0.0;
for(k=1;k<=i-1;k++)
    s1=s1+l[i][k]*y[k];
y[i]=(b[i]-s1 )/l[i][i];
    }
    cout<<"the vector y\n";
    for(i=1;i<=n;i++)
    cout<<y[i]<<endl;
    x[n]=y[n];
for(i=n-1;i>=1;i--)
    { s2=0.0;
  for(k=i+1;k<=n;k++)
        s2=s2+u[i][k]*x[k];
  x[i]=y[i]-s2;
  }
        cout<<"the sulition\n";
  for(i=1;i<=n;i++)
```

```
        cout<<"x("<<i<<")="<<x[i]<<endl;
    }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



6-4 تمرير المصفوفة إلى الإجراءات:

من بين الاستخدامات المفيدة للمصفوفات سواء المصفوفات العادية أم مصفوفات الحروف هو أن نتمكن من تمريرها إلى بعض الإجراءات التي نكتبها . وعندما ترغب في تمرير إحدى المصفوفات إلى إجراء يجب أن تعلن عن الإجراء بحيث تخبر المترجم أنه سيتلقى مصفوفة وليس متغيرا عاديا.

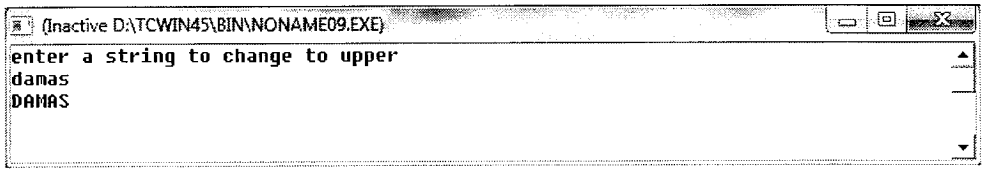
```
void fn(int num[ ]);
```

هنا نخبر المترجم أن الإجراء **fn** سيتلقى مصفوفة من نوع **int** وبعد ذلك عندما ترغب في تمرير المصفوفة إلى الإجراء ، لذلك نكتفي بذكر اسم المصفوفة فقط مثل: **fn(num);** .

يتم توضيح ذلك بالمثال الآتي:

```
//program 4-21
#include<iostream.h>
   #include<string.h>
        #include<ctype.h>
void chang(char string[ ]);
main()
{
char string[80];
cout<<"enter a string to change to upper\n" ;
cin>>string;
chang(string);
}
void chang(char string[ ])
{
 int i;
 char c;
for(i=0;i<strlen(string);i++)
{c= toupper(string[i]);
cout<<c ;

} }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



نلحظ أنه تم استدعاء الإجراء **toupper** من مكتبة التشغيل وهذا الإجراء يقوم بتحويل الحرف الذي تمرره له- إلى حرف كبير فمثلا:

Ch=toupper('a');

وأثناء طباعة **ch** سوف يطبع الحرف **A** .

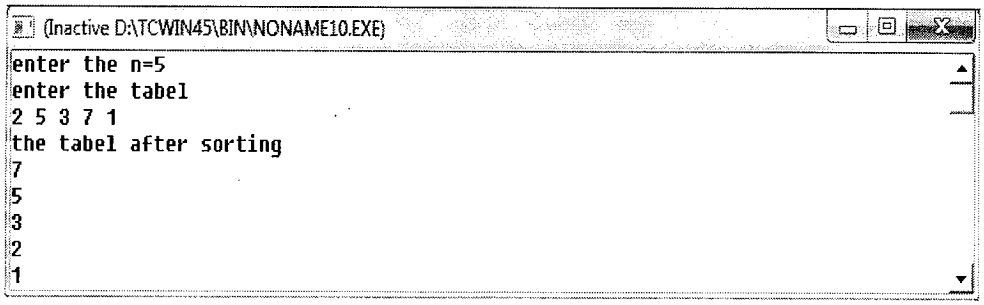
وكمثال على تمرير المصفوفات إلى الإجراء: ضع برنامجا يتضمن إجراء يقوم بترتيب عناصر جدول مؤلف من n عنصراً ترتيبا تنازليا ثم استفد من ذلك في ترتيب عناصر أي جدول تم إدخاله إلى الحاسب والمؤلف من n عنصر ترتيبا تنازليا.

البرنامج:

```
//program 4-22
#include<iostream.h>
void sort(int a[ ],int n);
main()
{
int a[50];
int i,n;
cout<<"enter the n=";
cin>>n;
cout<<"enter the tabel\n";
for(i=0;i<n;i++)
        cin>>a[i];
sort(a,n);
cout<<"the tabel after sorting\n";
for(i=0;i<n;i++)
cout<<a[i]<<endl;
}
```

```
void sort(int a[ ],int n)
{ int i,j,z;
for(i=0;i<n-1;i++)
for(j=i+1;j<n;j++)
if(a[i]<a[j])
{ z=a[i];
a[i]=a[j];
a[j]=z;
}
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



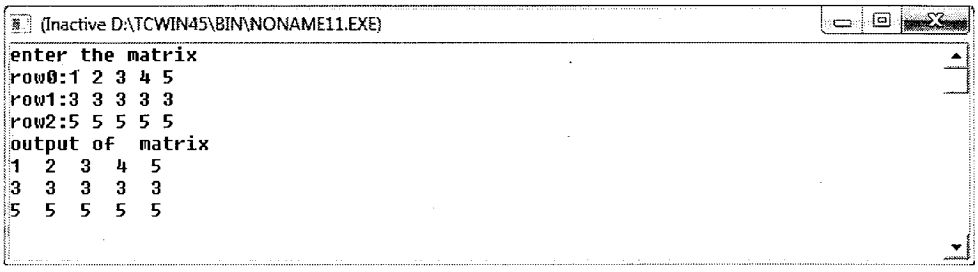
ضع إجرائين لإدخال وإخراج مصفوفة ثنائية البعد مؤلفة من ثلاثة أسطر وخمسة أعمدة بحيث يتم استدعاء هذين الإجرائين من البرنامج الرئيسي.

البرنامج:

```
//program 4-23
#include<iostream.h>
        void read (int a[][5]);
    void print (int a[][5]);
```

```
main()
{
int a[3][5];
 read(a);
 print(a);
 }
 void read(int a[][5])
{  int i,j;
 cout<<"enter the matrix\n";
  for(i=0;i<3;i++)
        {cout<<"row"<<i<<":";
        for(j=0;j<5;j++)
        cin>>a[i][j];}}
              void print(int a[][5])
{  int i,j;
 cout<<"output of  matrix\n";
  for(i=0;i<3;i++)
              {
        for(j=0;j<5;j++)
        cout<<a[i][j]<<"  ";
            cout<<"\n";}}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



1- ضع برنامجا لإيجاد قيمة محددة مصفوفة A من المرتبة nXn .

الحل: نقوم باستخدام التحويلات البسيطة بتحويل المصفوفة A إلى مصفوفة مثلثية ثم نقوم بضرب عناصر القطر الرئيسي، ويمكننا وضع الخوارزمية الآتية.

المدخلات: n رتبة المصفوفة .

a_{ij} عناصر المصفوفة حيث i=1,2,.....n , j=1,2,.....n

المخرجات: det قيمة المحددة.

خطوة 1: بفرض $a_{11} \neq 0$ نضرب السطر الأول ب $m_i = \frac{a_{i1}}{a_{11}}; i = 2,3,.....n$

ونطرحه من بقية الأسطر في المحددة فنحصل على المحددة:

$$\Delta = \begin{vmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22} & a_{23} & \cdots & a_{2n} \\ 0 & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & a_{n2} & a_{n3} & \cdots & a_{nn} \end{vmatrix}$$

خطوة 2: بفرض $a_{22} \neq 0$ نضرب السطر الثاني ب $m_i = \frac{a_{i1}}{a_{22}}; i = 3,.....n$

ونطرحه مما تبقى من الأسطر في المحددة فنحصل على المحددة:

$$\Delta = \begin{vmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22} & a_{23} & \cdots & a_{2n} \\ 0 & 0 & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & a_{n3} & \cdots & a_{nn} \end{vmatrix}$$

خطوة 3: نكرر العمل بصورة مشابهة للخطوات السابقة حتى نحصل على المحددة:

$$\Delta = \begin{vmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22} & a_{23} & \cdots & a_{2n} \\ 0 & 0 & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn} \end{vmatrix}$$

خطوة 4: نضع قيمة ابتدائية للمحددة det=1 .

خطوة 5: نوجد قيمة المحددة:

for i=1,2,3,….n

det=det.a_{ii}

خطوة 6: نطبع قيمة المحددة det.

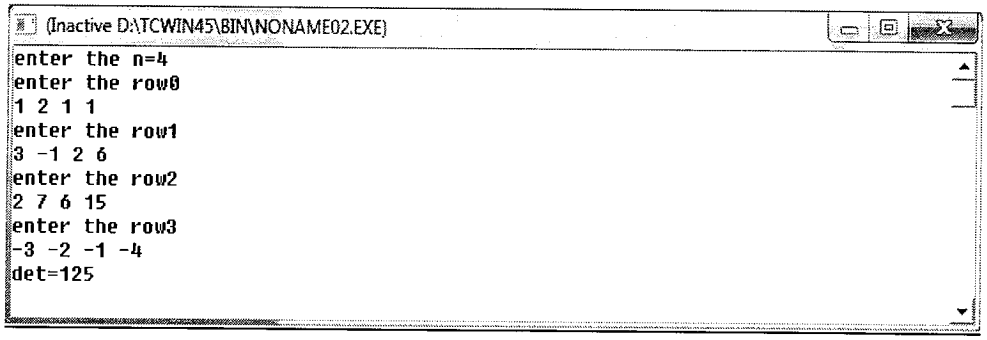
خطوة 7: النهاية.

البرنامج:

```
#include<iostream.h>
main()
{ float a[10][10],m[10],det;
int i,j,k,n;
cout<<"enter the n=";
cin>>n;
for(i=0;i<n;i++){
cout<<"enter the row"<<i<<endl;
 for(j=0;j<n;j++)
 cin>>a[i][j];}
 for(k=0;k<n-1;k++)
 {for(i=k+1;i<n;i++)
       {if(a[k][k]==0)a[k][k]=0.00000001;
       m[i]=a[i][k]/a[k][k];
```

```
for(j=k;j<n;j++)
a[i][j]=a[i][j]-m[i]*a[k][j];
}}
det=1;
for(i=0;i<n;i++)
det=det*a[i][i];
cout<<"det="<<det;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



2- لدينا مصفوفة من المرتبة MxN والمطلوب ضع برنامجا لحساب مايأتي:

- **أكبر العنا صر في السطر الأول**
- **المتوسط الحسابي لكل عمود من أعمدة المصفوفة**

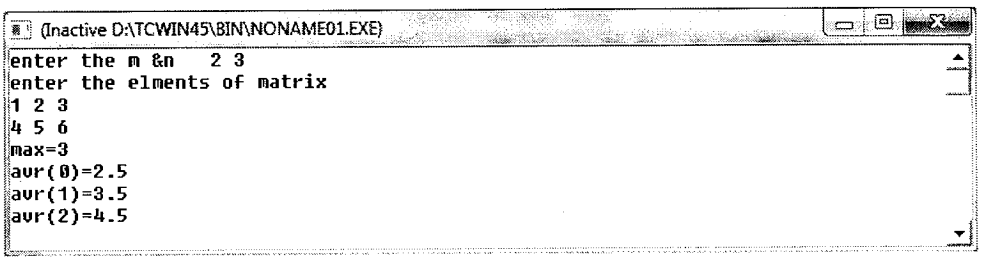
(السؤال الثالث في الدورة الاضافية/2012-2013)

البرنامج

```
#include<iostream.h>
main()
{
  int n,i,j,s,m;
  int a[10][10],max;
```

```
    float avr,s1;
    cout<<"enter the m &n   ";
cin>>m>>n;
     cout<<"enter the elments of matrix\n";
     for(i=0;i<m;i++ )
     for(j=0;j<n;j++)
     cin>>a[i][j];
      max=-300;
      for(j=0;j<n;j++)
      if(max<a[0][j])max=a[0][j];
            cout<<"max="<<max<<endl;;
            for(j=0;j<n;j++)
            {
            s1=0;
            for(i=0;i<m;i++)
            s1=s1+a[i][j];
            avr=s1/m;
            cout<<"avr("<<j<<")="<<avr<<endl;}
                  }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



3-(السؤال الرابع وارد في الدورة الاضافية/2012-2013)

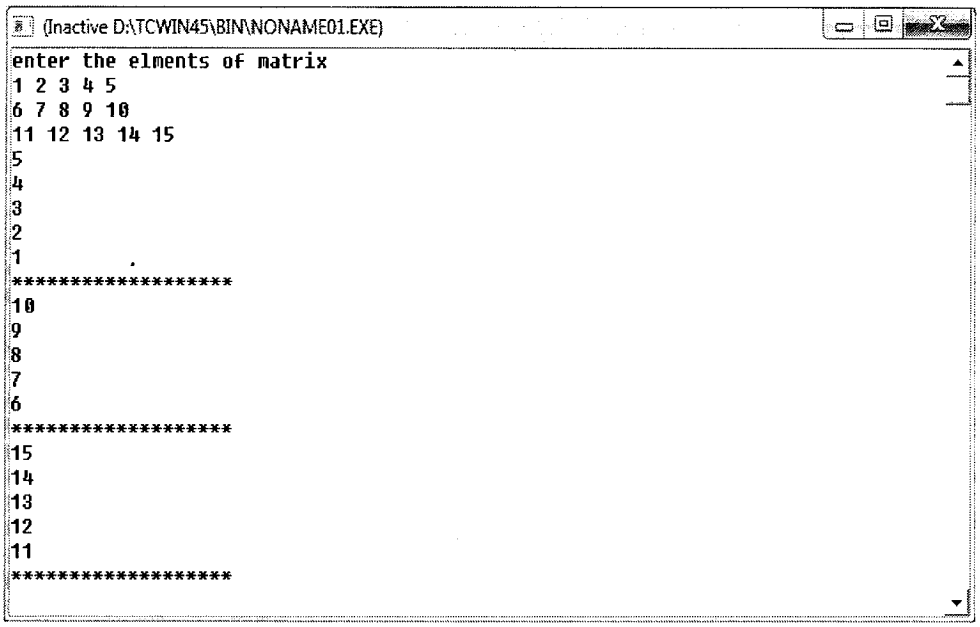
ضع إجراء لترتيب عناصر متجهة مؤلفة من خمسة عناصر ترتيبا تنازليا ثم استفد من ذلك في ترتيب العناصر لكل سطر من أسطر المصفوفة التي هي من المرتبة 3x5 .

البرنامج:

```
#include<iostream.h>
void sort(int a[5]);
main()
{
int i,j, a[3][5],v1[5],v2[5],v3[5];
cout<<"enter the elments of matrix\n" ;
 for(i=0;i<3;i++)
  for(j=0;j<5;j++)
 cin>>a[i][j];
 for(j=0;j<5;j++)
 v1[j]=a[0][j];
sort(v1);
        for(j=0;j<5;j++)
 v2[j]=a[1][j];
sort(v2);
for(j=0;j<5;j++)
 v3[j]=a[2][j];
sort(v3);
}
void sort(int a[5])
{
 int i,j,z;
 for(i=0;i<4;i++)
 for(j=i+1;j<5;j++)
```

```
if(a[i]<a[j]){
z=a[i];a[i]=a[j];a[j]=z;}
for(i=0;i<5;i++)
cout<<a[i]<<endl;
cout<<"********************"<<endl;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



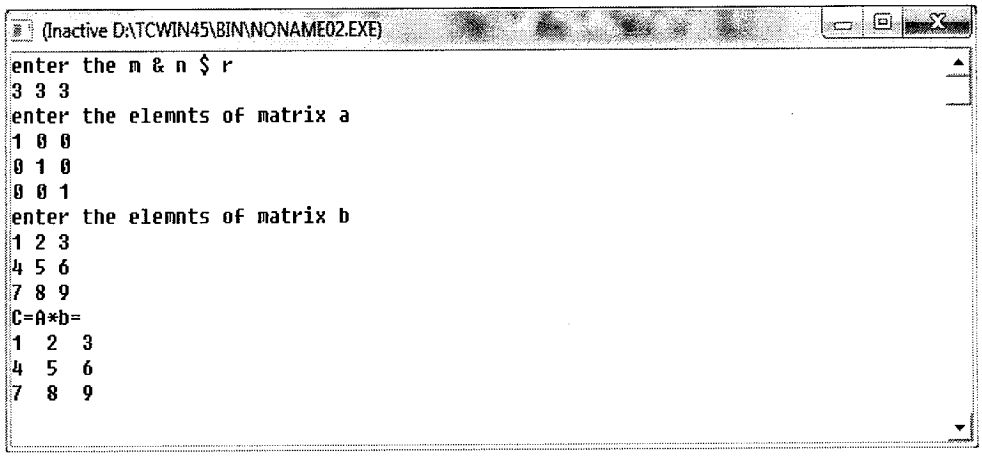
4- ضع برنامجا يتضمن إجراء لإيجاد الجداء الداخلي لمتجهتين تحتوي كل منهما على n عنصرا ثم استفد من ذلك في حساب جداء مصفوفتين C=A*B حيث A من المرتبة mXn وB من المرتبة nXr.

البرنامج:

```
#include<iostream.h>
 int pro(int a[5],int b[5]);
 main(){
 int i,j, a[10][10],b[10][10],c[10][10],m,n,r,v1[10],s,k,v2[10];
 cout<<"enter the m & n $ r\n";
 cin>>m>>n>>r;
  cout<<"enter the elemnts of matrix a\n" ;
  for(i=0;i<m;i++)
   for(j=0;j<n;j++)
  cin>>a[i][j];
                cout<<"enter the elemnts of matrix b\n" ;
  for(i=0;i<n;i++)
   for(j=0;j<r;j++)
  cin>>b[i][j];
  for(i=0;i<m;i++)
  {for(j=0;j<r;j++){
  c[i][j]=0;
  for(k=0;k<n;k++)
  {v1[k]=a[i][k];
  v2[k]=b[k][j];}
 s= pro(v1,v2);
 cout<<"C=A*b=\n";
  c[i][j]=c[i][j]+s;  }}
 for(i=0;i<m;i++)
  {for(j=0;j<r;j++)
  cout<<c[i][j]<<"  ";
  cout<<"\n";}}
```

```
int pro(int a[5],int b[5]) {
int i,s1;
  s1=0;
for(i=0;i<5;i++)
s1=s1+a[i]*b[i];
return s1; }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



5- (السؤال الثالث وارد في الدورة الثالثة /2010-2011)

لدينا مصفوفة من المرتبة MxN والمطلوب ضع برنامجا لحساب مايأتي:

1- مجموع عناصر هذه المصفوفة.

2- المتوسط الحسابي لكل عمود من أعمدة المصفوفة.

3-ترتيب العمود الأول ترتيبا تنازليا ثم طباعة المصفوفة الجديدة.

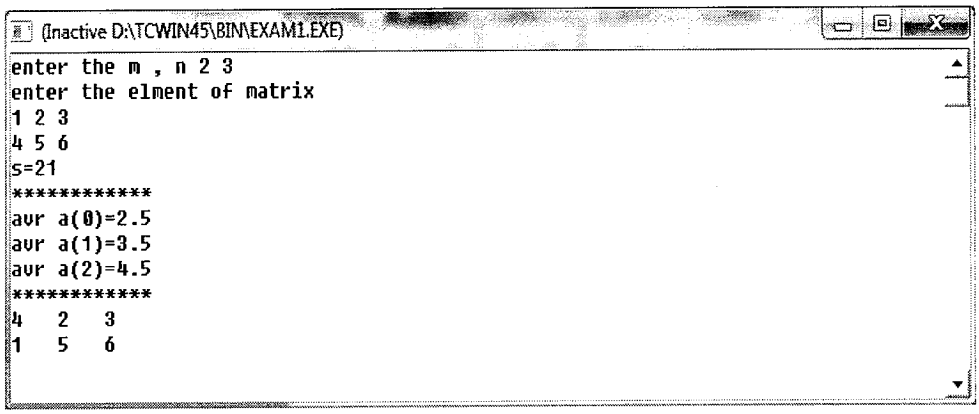
البرنامج:

```
#include<iostream.h>
main()
{
```

```
int i,j,m,n,s,t,k;
float avr,s1;
int a[10][10];
cout<<"enter the m , n ";
cin>>m>>n;
cout<<"enter the elment of matrix\n";
  s=0;
for(i=0;i<m;i++)
for(j=0;j<n;j++){
cin>>a[i][j];
s=s+a[i][j];
}
  cout<<"s="<<s<<endl;;
        cout<<"************\n";
  for(j=0;j<n;j++) {
  s1=0;
  for(i=0;i<m;i++)
  s1=s1+a[i][j];
  avr=s1/m;
  cout<<"avr a("<<j<<")="<<avr<<endl;
  }
 cout<<"************\n";
for(i=0;i<m-1;i++)
for(j=i+1;j<m;j++)
if(a[i][0]<a[j][0]){
t=a[i][0];a[i][0]=a[j][0];a[j][0]=t;}
for(i=0;i<m;i++)
```

```
{
  for(j=0;j<n;j++)
 cout<<a[i][j]<<"   ";
  cout<<"\n";
  }
  }
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



الدورة الاضافية 2014

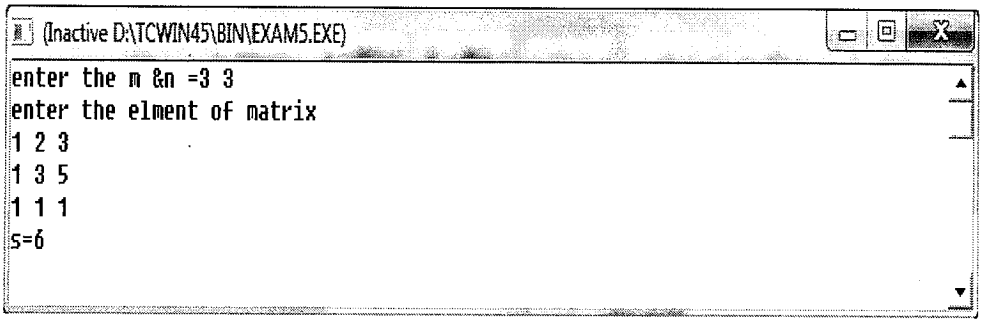
6- ضع برنامجا يتضمن إجراء يمثل الجداء الداخلي لمتجهتين تحوي كلٌ منها n عنصرا" ثم استفد من هذا الإجراء في إيجاد الجداء الداخلي للسطرين الأول والأخير في مصفوفة A من المرتبة mXn

البرنامج:

```
#include<iostream.h>
int pro(int a[],int b[],int n);
main()
{ int i,j,s,m,n,a[10][10];
cout<<"enter the m &n =";
cin>>m>>n;
```

```
cout<<"enter the elment of matrix\n";
for(i=0;i<m;i++)
for(j=0;j<n;j++)
cin>>a[i][j];
s=pro(a[0],a[m-1],n);
cout<<"s="<<s;
}
int pro(int a[],int b[],int n)
{
int s1,i;
s1=0;
for(i=0;i<n;i++)
s1=s1+a[i]*b[i];
return s1;}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:

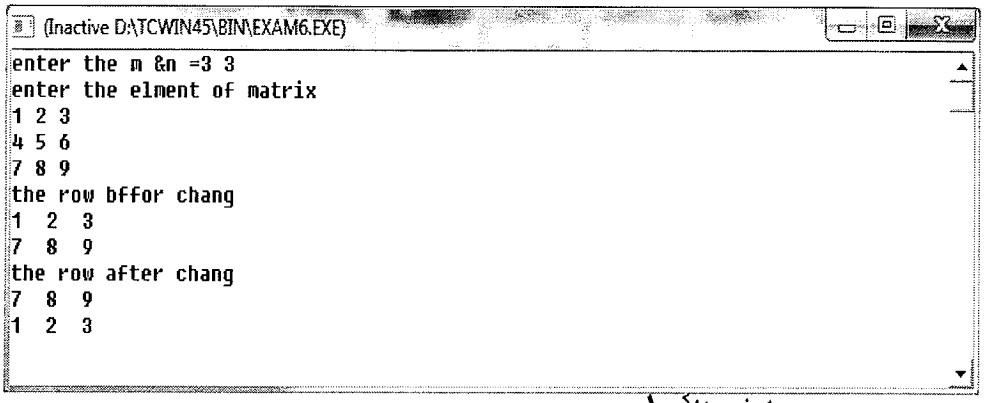


7) ضع برنامج يتضمن إجراء يمثل استبدال عناصر متجهتين تحوي كلا منهما n عنصرا ثم استفد من هذا الإجراء في استبدال لسطر الأول بدلا من الأخير في مصفوفة A من المرتبة mXn مع طباعة السطرين قبل الاستبدال وبعده.

البرنامج:

```
#include<iostream.h>
void chang(int a[],int b[],int n);
main()
{ int i,j,s,m,n,a[10][10];
cout<<"enter the m &n =";
cin>>m>>n;
cout<<"enter the elment of matrix\n";
for(i=0;i<m;i++)
for(j=0;j<n;j++)
cin>>a[i][j];
cout<<"the row bffor chang\n";
for(j=0;j<n;j++)
cout<<a[0][j]<<"  ";
cout<<"\n";
        for(j=0;j<n;j++)
cout<<a[m-1][j]<<"  ";
cout<<"\n";
chang( a[0],a[m-1], n);
}
void chang(int a[],int b[],int n)
{
int z,j;
for(j=0;j<n;j++) {
z=a[j];a[j]=b[j];b[j]=z;}
cout<<"the row after chang\n";
for(j=0;j<n;j++)
```

```
cout<<a[j]<<"  ";
cout<<"\n";
        for(j=0;j<n;j++)
cout<<b[j]<<"  ";
cout<<"\n"; }
```



دورة الفصل الأول 2014

9- لتكن A مصفوفة مربعة من المرتبة n. نقول إن A مصفوفة تعامدية(original) إذا تحقق الشرط التالي: $A.A^T=E_n$ إذ إن E_n تمثل المصفوفة الواحدية المطلوب:

ضع برنامجا لإيجاد مايأتي:

1- طباعة عناصر المصفوفة A^T منقول المصفوفة A.

2- طباعة عناصر المصفوفة $C= A.A^T$.

3- التأكد من أن المصفوفة A تعامدية أم لا. تعامدية أم لا.

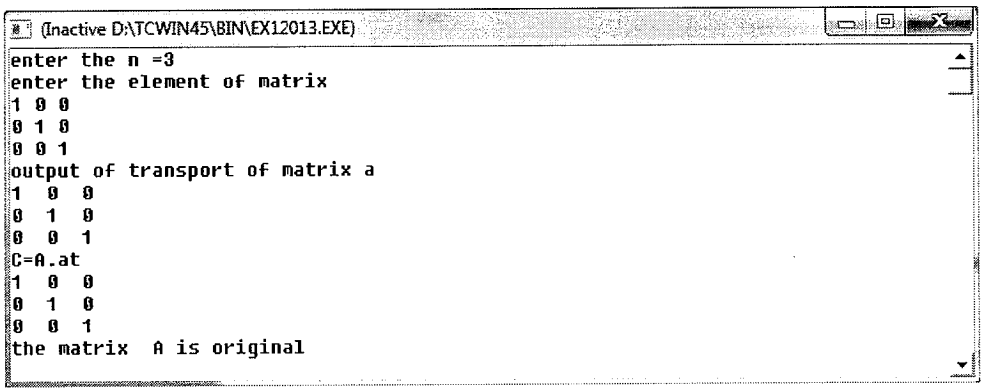
البرنامج:

```
#include<iostream.h>
#include<string.h>
#include<stdlib.h>
main()
```

```
{
int a[10][10],at[10][10],c[10][10],i,j,n,k;
char w;
cout<<"enter the n =";
cin>>n;
cout<<"enter the element of matrix\n";
for(i=0;i<n;i++)
for(j=0;j<n;j++)
cin>>a[i][j];
for(i=0;i<n;i++)
for(j=0;j<n;j++)
at[i][j]=a[j][i];
cout<<"output of transport of matrix a\n";
  for(i=0;i<n;i++)
  {
        for(j=0;j<n;j++)
        cout<<at[i][j]<<"  ";
        cout<<"\n";}
        for(i=0;i<n;i++)
{
for( j=0;j<n;j++)
{ c[i][j]=0;
for(k=0;k<n;k++)
c[i][j]=c[i][j]+a[i][k]*at[k][j];
}}
cout<<"C=A.at\n";
for(i=0;i<n;i++)
```

```
{
  for( j=0;j<n;j++)
            cout<<c[i][j]<<"  ";
            cout<<"\n";
            }
            for(i=0;i<n;i++)
            if(c[i][i]!=1)w='n';
            for(i=0;i<n;i++)
            for(j=0;j<n;j++)
            if(i!=j&&c[i][j]!=0)w='n';
      if(w=='n')cout<<"the matrix A is not original";
                        else
                  cout<<"the matrix  A is original";
                        }
```

وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:



تمارين 4

1-تتبع البرنامج الآتي وأكتب الناتج:

```
#include <iostream.h>
int a [] = {1, 2, 6, 7, 12};
int n,s=0;
int main ()
{
for ( n=0 ; n<5 ; n++ )
{
s += a[n];
}
cout << "s="<<s;}
```

2-اكتب خرج البرنامج التالي:

```
#include <iostream.h>
int main ()
{
char question[] = "Please, enter your first name: ";
char greeting[] = "Hello, ";
char yourname [80];
cout << question;
cin >> yourname;
cout << greeting << yourname << "!";
return 0;}
```

دورة الفصلية 2/2/+ الإضافية 2013

3-: ترغب جامعة تشرين في إجراء مناقصة لشراء مجموعة من الحواسيب والمطلوب وضع برنامج لمعرفة اسم الشركة الفائزة.

4- ترغب جامعة تشرين في قبول خمسة طلاب في التعليم الموازي تختارهم بالمفاضلة بين درجاتهم في الثانوية العامة بحيث تأخذ اثنين لفرع الطب واثنين لفرع الصيدلة وواحد للمعلوماتية فتقدم لهذه المفاضلة عشرة طلاب والمطلوب:

وضع برنامج لطباعة أسماء الطلبة المقبولين في الأفرع المطلوبة تبعا للفرع الواحد .

5- لتكنA مصفوفة مربعة من المرتبة nXn والمطلوب:

و ضع برنامج لطباعة عناصر المصفوفة $C=A^3$.

6-لتكن A مصفوفة مربعة من المرتبة nXn المطلوب:

ضع برنامجا لطباعة عناصر القطرين الرئيسي والثانوي وإيجاد مجموعهما.

7-ضع إجراء لإيجاد أكبر العناصر في جدول مؤلف من خمسة عناصر ثم استفد من ذلك في إيجاد أكبر العناصر في كل سطر من أسطر المصفوفة A من المرتبة (3X5).

8-لتكن A مصفوفة مربعة من المرتبة nXn والمطلوب:

وضع برنامج لإيجاد نظيمات المصفوفة A علما أن النظيمات تعطى بالعلاقات الآتية.

$$\| A \|_1 = \max(\sum_{i=1}^{n} | a_{ij} |); (1 \le j \le n)$$

$$\| A \|_\infty = \max(\sum_{j=1}^{n} | a_{ij} |); (1 \le i \le n)$$

$$\| A \|_E = (\sum_{i=1}^{n} \sum_{j=1}^{n} (a_{ij})^2)^{\frac{1}{2}}$$

يمكننا صياغة االعلاقات السابقة بالخوارزمية الآتية:

$$\| A \|_1 = \max(\sum_{i=1}^{n} | a_{ij} |); (1 \le j \le n)$$

$$\| A \|_\infty = \max(\sum_{j=1}^{n} | a_{ij} |); (1 \le i \le n)$$

$$\| A \|_E = (\sum_{i=1}^{n} \sum_{j=1}^{n} (a_{ij})^2)^{\frac{1}{2}}$$

9- لتكن جملة المعادلات الخطية:

$$\begin{aligned} &a_{11}x_1 + a_{12}x_2 + \ldots\ldots\ldots\ldots + a_{1n}x_n = b_1 \\ &a_{21}x_1 + a_{22}x_2 + \ldots\ldots\ldots\ldots + a_{2n}x_n = b_2 \\ &\vdots \qquad\quad \vdots \qquad\qquad\qquad\quad \vdots \qquad \vdots \\ &a_{n1}x_1 + a_{n2}x_2 + \ldots\ldots\ldots\ldots + a_{nn}x_n = b_n \end{aligned}$$

المطلوب وضع برنامج لحل جملة المعادلات السابقة وفق طريقة غاوص علما أن طريقة غاوص تعطى بالخوارزمية الآتية:

المدخلات: n عدد المعادلات .

a_{ij} عناصر مصفوفة المعاملات إذ إن i=1,2,.....n , j=1,2,.....n

b_i عناصر عمود الثوابت إذ إن i=1,2,.....n .

المخرجات: x_i حل جملة المعادلات.

خطوة 1: بفرض $a_{11} \neq 0$ وليكن $m_i = \frac{a_{i1}}{a_{11}}; i = 2,3,\ldots\ldots\ldots n$ ، نضرب المعادلة الأولى ب m_i ، ونطرحها من المعادلات الباقية (أي نضرب المعادلة الأولى ب m_2 ، ونطرحها من المعادلة الثانية، وكذلك المعادلة الأولى ب m_3 ونطرحها من المعادلة الثالثة وهكذا...) فنحصل على جملة جديدة تكون على النحو الآتي :

$$\begin{aligned} &a_{11}x_1+a_{12}x_2+\ldots\ldots\ldots\ldots+a_{1n}x_n=b_1 \\ &\qquad a_{22}x_2+a_{23}x_3+\ldots\ldots+a_{2n}x_n=b_2 \\ &\ldots\ldots\ldots\ldots\ldots \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ &\qquad a_{n2}x_2+a_{n3}x_3+\ldots\ldots+a_{nn}x_n=b_n \end{aligned}$$

إذ إن a_{ij} و b_i الجديدة تختلف عن a_{ij} و b_i في الجملة الأساسية.

خطوة 2: بفرض $a_{22} \neq 0$ نضرب المعادلة الثانية في الجملة التي حصلنا عليها من الخطوة السابقة ب $m_i = \frac{a_{i1}}{a_{22}}; i = 3,.....n$ ونطرحها من المعادلات الباقية فنحصل على جملة جديدة تكون على النحو الآتي:

$$a_{11}x_1+a_{12}x_2+.....+a_{1n}x_n=b_1$$
$$a_{23}x_3+.....+a_{2n}x_n=b_2$$
.....
.....
$$a_{n3}x_3+.....+a_{nn}x_n=b_n$$

إذ إن a_{ij} و b_i الجديدة تختلف عن a_{ij} و b_i في الجملة التي حصلنا في الخطوة 1 .

وهكذا نستمر بالعمل لنحصل على الجملة الأخيرة:

$$\mathbf{a_{11}x_1+a_{12}x_2+.....+a_{1n}x_n=b_1}$$
$$\mathbf{a_{22}x_2+a_{23}x_3+.....+a_{2n}x_n=b_2}$$
$$\mathbf{a_{33}x_3+.....+a_{3n}x_n=b_2}$$
.....
$$\mathbf{a_{nn}x_n=b_n}$$

خطوة 3: نحسب حل الجملة من العلاقات التالية:

$$x_n = \frac{b_b}{a_{nn}}$$
$$\vdots$$
$$x_j = \frac{b_j - (a_{j,j+1}x_{j+1} + ... + a_{jn}x_n)}{a_{ij}}$$
$$\vdots$$
$$x_1 = \frac{b_1 - (a_{12}x_2 + ... + a_{1n}x_n)}{a_{11}}$$

خطوة 4: نطبع الحلول x_i إذ إن $i=1,2,.....n$.

خطوة 5: النهاية

الفصل الخامس

السجلات وفهم المؤشرات

5-1- السجلات:

ذكرنا فيما سبق أن الشرط الأساسي لعمل المصفوفة يتمثل في أن تكون البيانات المخزنة فيها كلها من نوع واحد . ولكن هل يعني هذا أنه لايمكن تجميع عدة بيانات من أنواع مختلفة في مجموعة واحدة؟ .

يمكن ذلك بوساطة استخدام السجلات وسوف نشرح ماهو السجل.

5-1-1- تعريف السجل:

السجل معناه أن تضع مجموعة من البيانات التي تختلف في النوع معا بحيث يمكن التعامل معها بوضعها وحدة واحدة ،أو يمكنك التعامل مع العناصر المفردة داخله.ولكي تتعرف على فائدة السجل تخيل إنك تكتب برنامجا لتسجيل بيانات الموظفين في الشركة التي تعمل فيها. في هذه الحالة أنت تحتاج إلى اسم الموظف وعنوانه (سلسلة حروف من نوع char) وسن الموظف(من نوع int) وراتبه(من نوع float) وربما معلومات أخرى .لذلك تقوم بإنشاء سجل يضم هذه العناصر سويا بوضعها وحدة واحدة. وبعد أن تصنع هذا السجل ستتمكن من التعامل مع هذه البيانات المختلفة بوضعها وحدة واحدة أو تتعامل مع كل عنصرها كل على حدى.

5-1-2- الإعلان عن السجل:

عندما ترغب في الإعلان عن سجل جديد أتبع الصيغة الآتية:

```
struct اسم السجل
{
.....;
مجموعة البيانات التي يتكون منها السجل.....
```

.....;

};

هنا يبدأ السجل بالكلمة المفتاحية struct بعد ذلك نختار اسما للسجل ثم نذكر أنواع البيانات التي ستضعها في السجل بين القوسين{ } ثم تختم السجل بالفاصلة المنقوطة.

```
Struct  employee
{
char name[40];
char address[40];
int age;
float salary;
};
```

وعندما ترغب في استخدام السجل يجب أن تعلن عن متغير من النوع نفسه مثل:

```
Struct employee emp1;
```

وبناء على الإعلان فإن المتغير emp1 يتكون من أربعة عناصر وهي الاسم والعنوان والعمر والراتب.

5-1-3- استخدام السجل:

عندما ترغب في التعامل مع أي عنصر داخل السجل تذكر اسم السجل ثم النقطة ثم اسم العنصر داخل االسجل.

```
emp1.age=35;
```

وإذا أردت أن تنقل القيمة الموجودة في العنصر إلى متغير أخر نكتب:

```
a=emp1.salary;
```

وإذا رغبت في قراءة العنصر من لوحة المفاتيح نكتب:

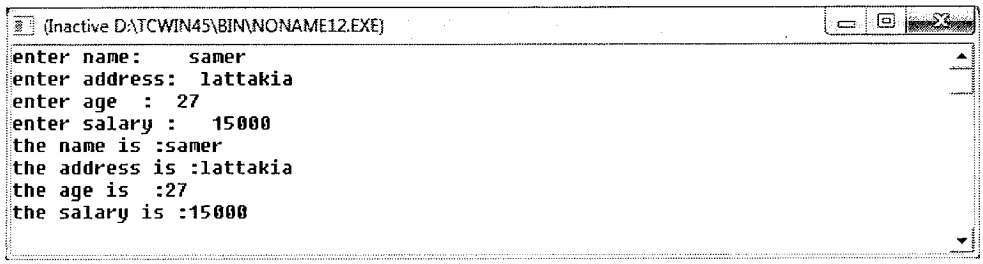
```
gets(emp1.name);
```

البرنامج الآتي يوضح استخدام السجل:(يشمل اسم الشخص وعنوانه وعمره وراتبه)

```
//program 5-1
#include<iostream.h>
main()
{ struct  employee
{
char name[40];
char address[40];
int age;
float salary;
};
struct employee emp1;
cout<<"enter name:    ";
cin>>emp1.name;
cout<<"enter address: ";
cin>>emp1.address;
cout<<"enter age  :  ";
cin>>emp1.age;
cout<<"enter salary :   ";
cin>>emp1.salary;
cout<<"the name is :"<<emp1.name<<endl;
cout<<"the address is :"<<emp1.address<<endl;
cout<<"the age is  :"<<emp1.age<<endl;
cout<<"the salary is :"<<emp1.salary<<endl;
```

}

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



5-1-4-مصفوفة السجلات:

في المثال السابق تم عرض بيانات لشخص واحد أما إذا كان لدينا 100 موظف فنستخدم مصفوفة من السجلات على النحو الآتي:

```
{ struct  employee
        {
    char name[40];
    char address[40];
        int age;
      float salary;
        };

struct employee emp [100];
```

وهنا إذا رغبت في أن تدخل بيانات للموظف الخامس فإنك تكتب:

```
emp[4].age=35;
```

ويمكن أن تحدد رقم السجل داخل المصفوفة باستخدام أحد المتغيرات كما يأتي:

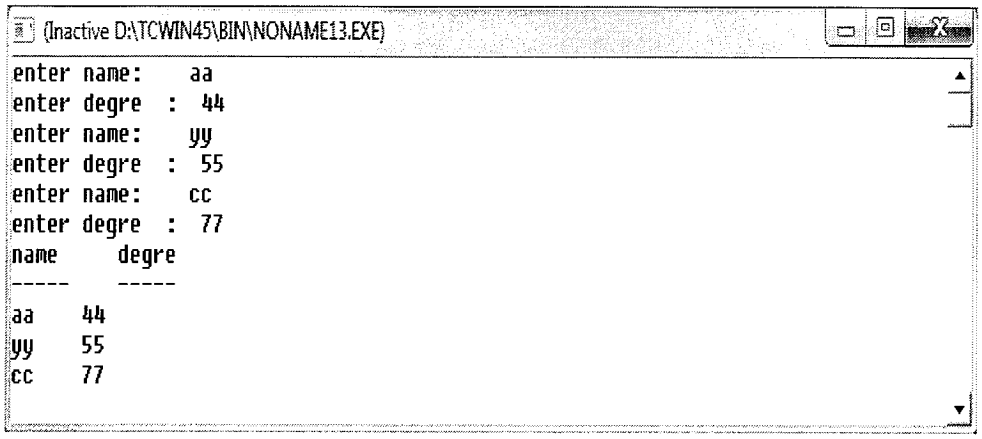
```
emp[count].salary;
```

باستخدام السجلات ضع برنامجا لإدخال أسماء ثلاثة طلاب ودرجاتهم الامتحانية في مقرر ما ثم طباعة الدرجات .

البرنامج:

```
//program 5-2
#include<iostream.h>
 #include<string.h>
main()
{
 struct  student
{
char name[40];
int degre;
};
struct student st[3];
int i;
for(i=0;i<3;i++)
{
cout<<"enter name:    ";
cin>>st[i].name;
cout<<"enter degre  :  ";
cin>>st[i].degre;
}
cout<< "name"<<"\t"<<" degre"<<"\n";
cout<<"-----"<<"\t"<<" -----\n";
for(i=0;i<3;i++)
cout<<st[i].name<<"    "<< st[i].degre<<endl;
}
```

ويعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



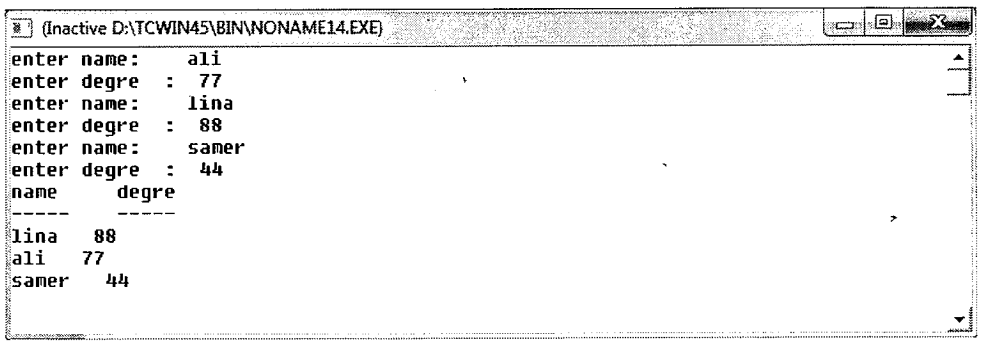
باستخدام السجلات ضع برنامجا يقوم بترتيب أسماء الطلاب حسب درجاتهم ترتيبا تنازليا.

البرنامج:

```
//program 5-3
#include<iostream.h>
 #include<string.h>
main()
{
 struct  student
{
char name[40];
int degre;
};
struct student st[3];
int i,j,z;
 char z1[40];
for(i=0;i<3;i++)
```

```
{
cout<<"enter name:    ";
cin>>st[i].name;
cout<<"enter degre  :  ";
cin>>st[i].degre;
}
  for(i=0;i<2;i++)
  {
  for(j=i+1;j<3;j++)
  if(st[i].degre<st[j].degre)
  { z=st[i].degre;st[i].degre=st[j].degre;st[j].degre=z;
strcpy(z1,st[i].name);strcpy(st[i].name,st[j].name);strcpy(st[j].name,z1
); }
      }
cout<<"name"<<"\t"<<" degre\n";
cout<<"-----\t"<<" -----\n";
for(i=0;i<3;i++)
cout<<st[i].name <<"   "<<st[i].degre<<endl;
}
```

ويعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:

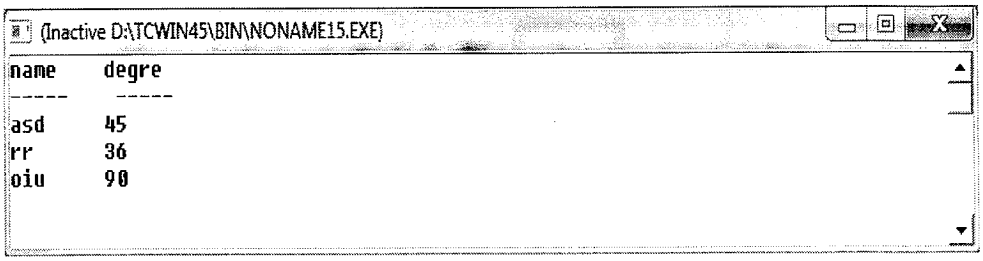


5-1-5-وضع المدخلات ضمن السجل

يمكننا وضع المعطيات ضمن السجل نوضحها بالمثال الآتي:

```
//program 5-4
#include<iostream.h>
#include<string.h>
main()
{
 struct  student
{
char name[40];
int degre;
}st[6]={"asd",45,"rr",36,"oiu",90};
int i;
cout<<"name\t"<<"degre\n";
cout<<"-----\t"<<" -----\n";
for(i=0;i<3;i++)
cout<<st[i].name <<"\t"<< st[i].degre<<endl;;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:

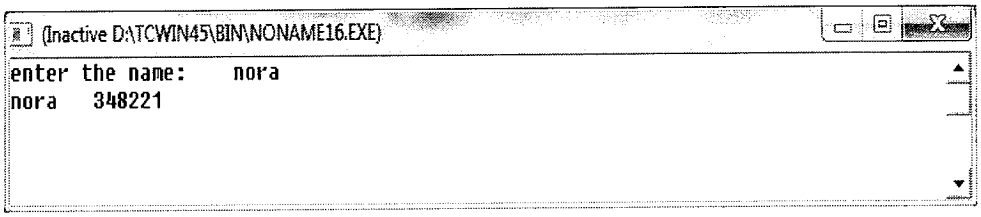


باستخدام السجلات ضع برنامجا يمثل دليل الهاتف.

البرنامج:

```
//program 5-5
#include<iostream.h>
#include<string.h>
main()
{
 struct  student
{
char name[40];
long phone;
}st[6]={"ali",417196,"sami",318653,"nora",348221};
int i;
 char string[40];
cout<<"enter the name:    ";
 cin>>string;
for(i=0;i<3;i++)
 if(strcmp(st[i].name,string)==0)
{cout<<st[i].name<<"   "<<st[i].phone;goto mm;}
cout<<"not found";
mm:cout<<endl;}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج كما يأتي:



5-2-الاتحاد

الاتحاد union يشبه السجل structure إلى درجة كبيرة ؛ إذ إنهما يستخدمان من أجل وضع عدة أنواع من البيانات المختلفة في مجموعة واحدة بصورة تتمكن فيها من التعامل مع المجموعة بوضعها كيانا" واحدا" أو تتعامل مع كل عنصر فيها بمفرده .ويتم الإعلان عن الاتحاد بالطريقة نفسها التي تعلن بها عن السجل ولكن تستخدم كلمة union بدلا من كلمة struct .

فمثلا يمكن أن نكتب:

```
union sample
{ long num;
  float f1;
};
```

ويمكنك الإعلان عن متغيرات من الاتحاد والتعامل مع عناصره المختلفة بالطريقة نفسها التي شرحناهاللسجل.فمثلا يمكن أن تعلن عن متغيرات في الاتحاد كما يأتي:

```
union sample a;
```

وإذا رغبت في التعامل مع أحد عناصر الاتحاد تتبع الصيغة الآتية:

```
a.num=123;
```

والفرق بين السجل والاتحاد في السجل يقوم المترجم بحجز ذاكرة تتسع لجميع العناصر الموجودة فيه بينما في الاتحاد يقوم المترجم بحجز ذاكرة تكفي لتخزين أكبر عنصر فيه فقط ،وسوف يستخدم المساحة نفسها للتعامل مع بقية العناصر الموجودة فيه.

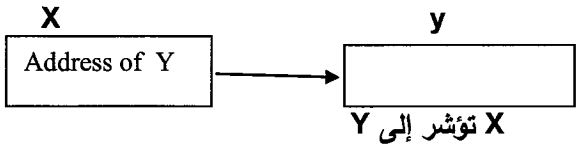
5-3- تعلم المؤشرات:

5-3-1- تمهيد:

للمتغير ثلاثة مفاهيم أساسية وهي اسم المتغير ونوع القيم التي يأخذها وعنوان المتغير في الذاكرة علما أن هذا العنوان يوضع من إدارة الذاكرة.

5-3-2- تعريف المؤشر :

المؤشر pointer هو متغير يستخدم ليحمل عنوان متغيرٍ آخر في الذاكرة نوضحه بالمخطط الآتي:



نلحظ أن المتغير X عبارة عن مؤشر يؤشر إلى المتغير Y .

ويمكن أن تستخدم المؤشر ليحمل عنوان متغير عادي أو يحمل عنوان سجل أو يحمل عنوان مصفوفة أو يحمل عنوان إجراء أو غير ذلك ؛ أي أن المؤشر يشير إلى مكان في الذاكرة ومن هنا جاءت تسميته. وتكمن قوة المؤشرات في أنه يمكنك أن تستخدم المؤشر نفسه كأن تمرره إلى إجراء آخر؛ أي أن تزيد من قيمته لتجعله يشير إلى عنوان آخر أو تستخدمه للوصول إلى البيانات التي يشير إليها.

5-3-3- معامل العنوان (&) ومعامل المحتوى(*)

مع المؤشرات يتم استخدام أداتان مهمتان وهما :

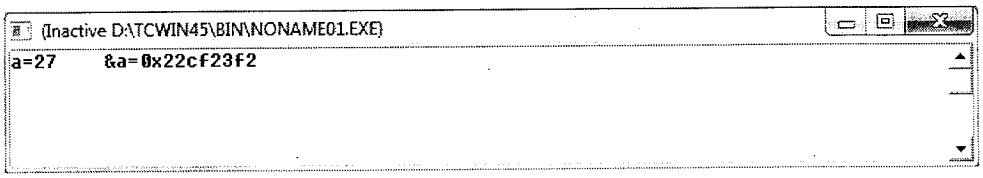
1- الأداة & : تستخدم لإعطاء عنوان متغير في الذاكرة ويمكننا طباعة عنوان المتغير بالأمر الآتي: cout<<&المتغير اسم .

ضع برنامجا" لتخزين عدد ثم طباعة هذا العدد وعنوانه في الذاكرة.

البرنامج:

```
//program 5-6
#include<iostream.h>
  main()
       {
int a;
a=27;
cout<<"a="<<a<<"\t"<<"&a="<<&a;
}
```

وبعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج الشكل الآتي:



نلحظ أن العنوان يبدأ ox وهذا يدل بأن العنوان مكتوب بالنظام الست عشري.

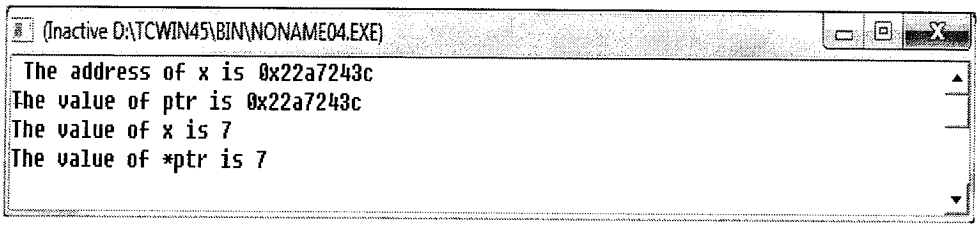
2- الأداة *: تستخدم لإعطاء قيمة المتغير للمؤشر (المشار إليه)

والبرنامج الآتي يوضح العملية كما يأتي:

```
//program 5-7
#include<iostream.h>
main ( )
{
  int x ;
```

```
int  *ptr;
      x = 7;
      ptr = &x;
      cout <<" The address of x is " << &x <<endl
                        << "The value of ptr is " << ptr<< endl;
cout << "The value of x is " << x<< endl
      cout << "The value of *ptr is " << *ptr<< endl;
}
```

ويعد تنفيذ هذا البرنامج تظهر لنا شاشة الإخراج بالشكل الآتي:



5-3-4-المرجع:

المرجع هو اسم لمتغير يعلن عنه باستخدام العامل المرجعي & والذي يلحق بنوع المرجع فعلى سبيل المثال إذا كان لدينا متغير صحيح a=25 وتم الإعلان عن مرجع له وليكن b فيتم ذلك بالشكل الآتي:

```
int a=25;
int &b=a;
```

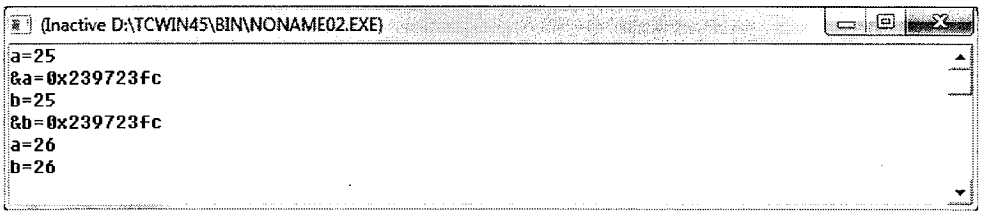
عندما يعرف b بأنه مرجعا ل a فهذا يعني أن قيمة b هي ذاتها قيمة a وأي تغير في قيمة a سيصيب قيمة bوالعكس صحيح ، اضف إلى ذلك فإن a و b لهما العنوان نفسه في الذاكرة وسوف يتم توضيح الفكرة بالمثال الآتي:

ضع برنامجا لتخزين عدد ما في متغير ثم ضع مرجعا لهذا المتغير ثم اطبع العدد ، والمرجع وكذلك عنوان كل منهما في الذاكرة ثم قم بزيادة المتغير وبعد ذلك اطبع العدد ومرجعه من جديد.

البرنامج:

```
//program 5-8
#include<iostream.h>
main()
 {
 int a=25;
 int &b=a;
cout<<"a="<<a<<endl;
cout<<"&a="<<&a<<endl;
 cout<<"b="<<a<<endl;
cout<<"&b="<<&b<<endl;
a++;
 cout<<"a="<<a<<endl;
 cout<<"b="<<b<<endl;
}
```

وتكون نتائج البرنامج كما يأتي:



5-3-5-الإعلان عن المؤشر:

عندما ترغب في الإعلان عن المؤشر يجب أن تحدد شيئين:

الأول : هو نوعية البيانات التي سيشير اليها .

والثاني هو اسم المؤشر على النحو الآتي:

```
data type *pointer name ;
```

وعلى سبيل المثال:

```
Int *ptr;
```

في هذا الإعلان نخبر المترجم أننا نريد أن نستخدم المؤشر ptr وأن هذا المؤشر سيشير إلى عدد صحيح . هنا سيعرف المؤشر أننا نريد أن نستخدم مؤشر وليس متغير عادي بسبب الحرف * .

وكمثال آخر: float *degree وكذلك الإعلان char *string يخبر المترجم أننا نريد استخدام المؤشر string ليشير إلى سلسلة من الحروف وهكذا.

وإذا كنت ستستخدم المؤشر ليشير إلى سجل يجب أن تعلن عن السجل أولا كالآتي:

```
struct student
{
char name[40];
int mark ;
}
```

ثم تعلن عن المؤشر بعد ذلك مثل:

```
Struct student *st;
```

هنا نخبر المترجم أننا نريد أن نستخدم المؤشر st ليشيرإلى سجل من نوع student الذي أعلنا عنه سابقا.

يفضل إعطاء قيمة ابتدائية للمؤشر، وعادة ما يتم إعطاء المؤشرات القيمة الابتدائيةNULL والتي تعني أن المؤشر يؤشر على" لا شيء , والقيمة NULL يمكن إسنادها إلى أي نوع من المؤشرات. وبالطبع يمكن إسناد القيمة الابتدائية NULL إلى المؤشر في أثناء التصريح عنه كما هو الحال مع في ماتبقى من أنواع المتغيرات كما في المثال الآتي:

```
int *ptr=NULL;
```

كما نستخدم القيمة NULL لأغراض أخرى في البرنامج فمثلاً لاختبار فيما إذا كان المؤشر prt يؤشر على قيمة فعلية أو أنه يؤشر على لاشيء نستخدم العبارة الشرطية الآتية:

```
if (ptr!=NULL) ...
```

ملحوظة:

عند إسناد قيمة أو متغير إلى مؤشر يجب أن تكون هذه القيمة أو المتغير من نوع بيانات المؤشرنفسه، وغير ذلك يؤدي إلى خطأ قواعدي في البرنامج.

5-3-6- تهيئة المؤشر:

- إن تهيئة المؤشر هي أن تجعله يشير إلى عنوان في الذاكرة. ويتم ذلك باستخدام معامل العنوان & هذا المعامل يستخدم للحصول على عنوان أي متغير في ذاكرة الجهاز.

وكمثال على تهيئة المتغير وجعله يشير إلى عنوان في الذاكرة أنظر العبارات الآتية:

```
int num;
int *ptr;
num=5;
ptr=&num;
```

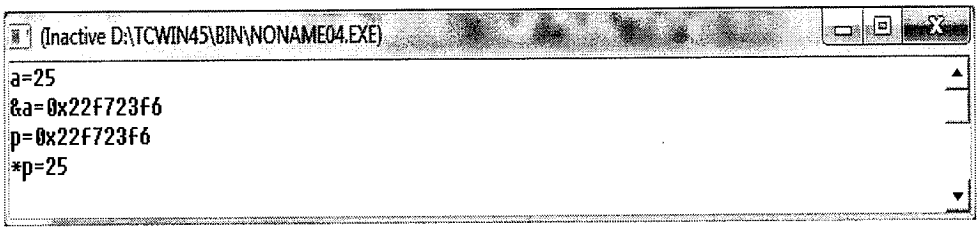
في هذه العبارات نعلن عن المتغير num ثم نعلن عن المتغيرptr الذي سنستخدمه بوصفه مؤشرا" ليشير إلى عنوان المتغير num في الذاكرة بعد ذلك نهيئ المتغيرnum ونضع فيه العدد5 أما في العبارة الأخيرة فإننا نستخدم المعامل & لنحصل على عنوان المتغير num ونضع هذا العنوان في المؤشر ptr بهذا الشكل يصبح المؤشر ptr حاملا لعنوان المتغيرnum ؛أي أنه يشير إلى هذا المتغير،و بعد أن يشير المؤشر إلى المتغير يمكن الوصول إلى القيمة المخزنة في العنوان باستخدام معامل المحتوى (*) .

ضع برنامجا لتخزين عدد ثم ضع مؤشر ليحمل عنوان ذلك العدد ثم اطبع العدد وعامله وكذلك المؤشر والقيمة المخزنة في المؤشر.

البرنامج:

```
//program 5-9
#include<iostream.h>
main()
 {
 int a=25;
 int *p=&a;
 cout<<"a="<<a<<endl;
  cout<<"&a="<<&a<<endl;
       cout<<"p="<<p<<endl;
     cout<<"*p="<<*p;   }
```

وتكون نتائج البرنامج كما يأتي:



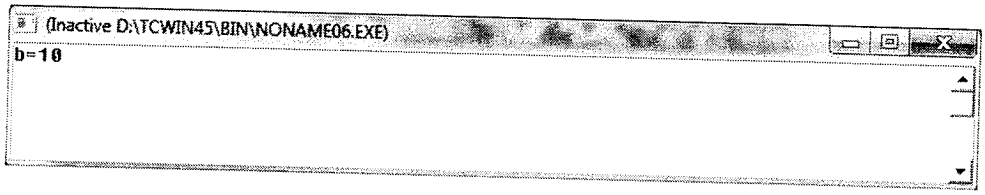
5-3-7-استخدام المؤشرات :

عندما ترغب في استخدام المؤشرات هنالك خطوتان يجب أن تتبعهما: الأولى أن تعلن عن هذا المؤشر، فتخبر المترجم عن اسمه وعن نوعية البيانات التي يشير إليها والخطوة الثانية هي أن تهيئ هذا المؤشر بان تجعله يشير إلى أحد المتغيرات فعلا.

المثال الآتي يوضح ذلك:

```
//program 5-10
#include<iostream.h>
main()
{
int a=10;
int b;
int *p;
p=&a;
b=*p;
cout<<"b="<<b;
}
```

وسوف يكون ناتج البرنامج كما يأتي:



البرنامج الآتي يوضح كيفية نسخ المؤشرات:

```
//program 5-11
#include <iostream.h>
int main()
{
int i =3;
int *p,*q;
p=&i;
```

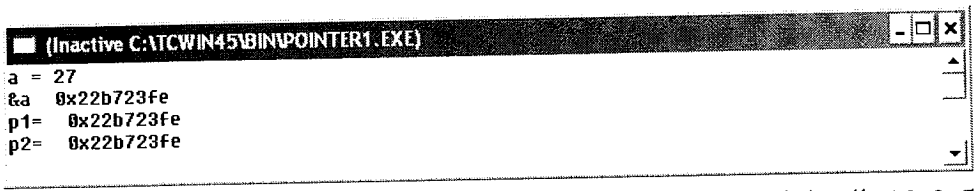
5-3-9- المساواة الاسنادية للمؤشرات:

يمكننا إسناد مؤشر إلى مؤشر آخر علما أن المؤشرين يجب أن يشيرا إلى معطيات من النوع نفسه.

وهذا مانوضحه في البرنامج الآتي:

```
//program 5-13
   #include<iostream.h>
    main()
    {
     int a=27;
     int *p1,*p2;
     p1=&a;
     p2=p1;
    cout<<"a = "<<a<<endl;
     cout<<"&a  "<<&a<<endl;
     cout<<"p1=  "<<p1<<endl;
     cout<<"p2=  "<<p2<<endl;
    }
```

وتظهر شاشة الخرج كمايأتي:



5-3-10- العمليات الحسابية على المؤشرات:

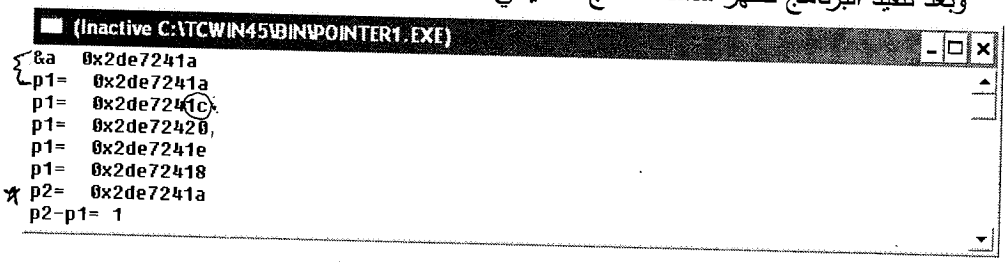
توجد مجموعة محددة من العمليات الحسابية يمكن تطبيقها على المؤشرات وهي:

++ أو -- أو =+ أو =-

ويمكننا طرح مؤشر من مؤشر آخر وهذا مانوضحه بالمثال الآتي:

```
//program 5-14
   #include<iostream.h>
    main()
    {
     int a=27;
     int *p1,*p2;
     p1=&a;
     cout<<"&a  "<<&a<<endl;
     cout<<"p1=  "<<p1<<endl;
     p1++;
     cout<<"p1=  "<<p1<<endl;
     p1+=2;
    cout<<"p1=  "<<p1<<endl;
    p1--;
    cout<<"p1=  "<<p1<<endl;
    p1-=3;
    cout<<"p1=  "<<p1<<endl;
    p2=&a;
    cout<<"p2=  "<<p2<<endl;
    cout<<"p2-p1= "<<p2-p1<<endl;
   }
```

وبعد تنفيذ البرنامج تظهر شاشة النتائج كما يأتي:



نلحظ من خرج البرنامج: أن زيادة المؤشر ب 1 ستزيد حجم المؤشر بمقدار حجم النوع الذي يشير إليه المؤشر.

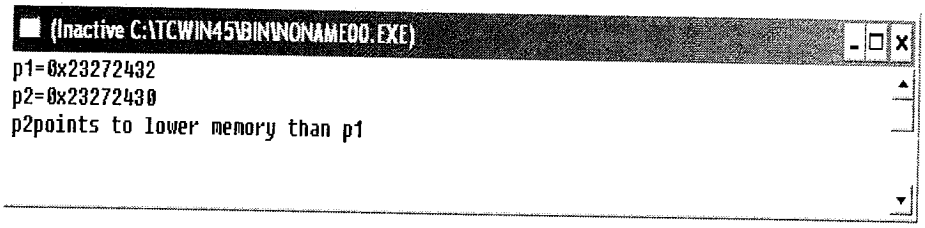
وزيادة المؤشر ب2 ستزيد قيمة هذا المؤشر بمقدار حجم النوع الذي يشير إليه المؤشر مضروبا ب2 وهكذا .وكذلك الأمر فيما يتعلق بالانقاص ، طرح مؤشر من مؤشر سيتم طرح قيم هذه المؤشرات من بعضها البعض ثم يتم تقسيم الناتج على حجم النوع ليعطينا عدد البايتات.

5-3-11- مقارنة المؤشرات:

تتم عملية المقارنة بين المؤشرات بموجب عملية المقارنة بين العناوين المخزنة في الذاكرة وهذا مايوضحه المثال الآتي.

```
//program 5-15
   #include<iostream.h>
    main()
    {
    int a=27;
    int b=35;
    int *p1=&a;
    int *p2=&b;
   cout<<"p1="<<p1<<endl;
     cout<<"p2="<<p2<<endl;
     if (p1<p2)
   cout<<"p1points to lower memory than p2"<<endl;
         else
   cout<<"p2points to lower memory than p1"<<endl;
   }
```

وبعد تنفيذ البرنامج تظهر شاشة النتائج كما يأتي:



12-3-5- استدعاء الإجراءات بالمؤشر:

هنالك ثلاث طرق لتمرير الوسائط للإجراء:

1-التمرير بالقيمة .

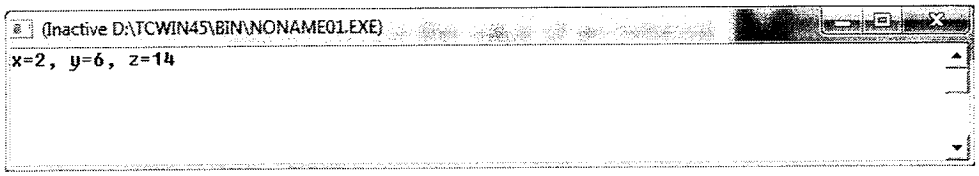
2-التمرير بالمرجع.

3-التمرير بالمرجع مع مؤشر.

تمرير الوسطاء بالقيمة تم ذكرها سابقا في الفصل الثالث، أما تمرير الوسائط للإجراء بالمرجع نوضحها بالمثال الآتي.

```
//program 5-16
  #include <iostream.h>
   void duplicate (int& a, int& b, int& c)
  {
  a*=2;
  b*=2;
  }
  int main ()
  {
  int x=1, y=3, z=7;
  duplicate (x, y, z);
  cout << "x=" << x << ", y=" << y << ", z=" << z; }
```

وبعد تنفيذ البرنامج تظهر شاشة النتائج كما يأتي:

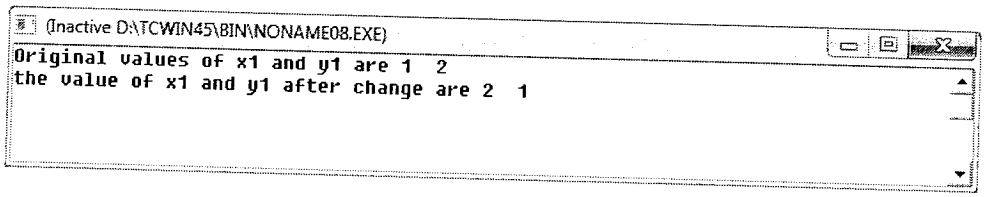


ضع إجراء للتبديل بين عددين وذلك بتمرير الوسطاء للإجراء بالمرجع.

البرنامج:

```
//program 5-17
   #include <iostream.h>
   void chang (int &x , int &y);
   main ( )
   {
   int x1= 1;
   int y1= 2;
   cout <<"Original values of x1 and y1 are " << x1<<"  "<<y1<<endl;
    chang (x1, y1);
    cout <<"the value of x1 and y1 after change are " <<x1<<"
   "<<y1<<endl;
   return 0;
   }
   void chang (int& a, int & b)
   {
   int z =a;
   a=b;
   b=z;}
```

ويعد تنفيذ البرنامج تظهر شاشة النتائج كما يأتي:

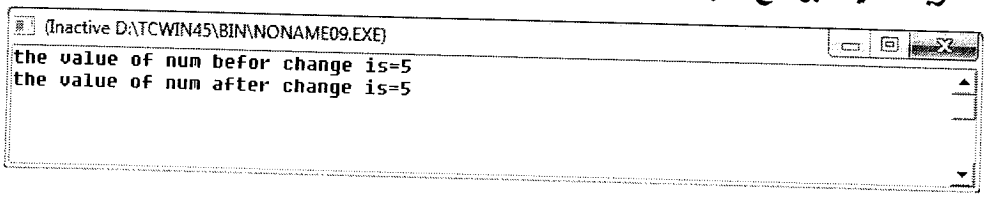


ومن المعلوم أن الإجراء بدون استخدام المؤشرات يتلقى قيمة المتغير وليس المتغير نفسه كما يتضح من البرنامج الآتي:

البرنامج

```
//program 5-18
#include<iostream.h>
 void change(int y);
main()
{
int num=5;
cout<<"the value of num befor change is="<<num<<endl;
change(num);
cout<<"the value of num after change is="<<num<<endl;
}
void change(int num)
{
num=7;
}
```

ويعد تنفيذ البرنامج تظهر شاشة النتائج كما يأتي:



نلحظ في الإجراء ()main أننا نعلن عن المتغير num ونضع فيه العدد 5 ثم نستدعي الإجراء change ونمرر له هذا المتغير. في هذه الحالة يقوم المترجم بتمرير القيمة الموجودة في المتغير num وهي العدد 5 إلى اجراء change ولم يتمكن الإجراء change من رؤية المتغير num الموجود في ()main على الرغم من أن هذا المتغير موجود في main و change إلا أن كل واحد يختلف عن الأخر. فكل واحد منهما متغير محلي خاص بالإجراء الذي يوجد فيه.

أما إذا اردت أن تتيح للإجراء الفرعي أن يعدل المتغير الذي تمرره له في هذه الحالة لا تمرر له المتغير نفسه ولكن مرر له عنوانه في الذاكرة. وسوف يتمكن الإجراء من باستخدام المؤشرات أن يعدل من قيمة المتغير الأصلي ولكي نقوم بذلك يجب أن نقوم بخطوتين:

الخطوة الأولى: أن تعدل الإعلان عن الإجراء في بداية البرنامج ورأس البرنامج نفسه بحيث تخبر المترجم أنه ستلقى مؤشرا وليس متغيرا عاديا مثل:

```
Void change(int *ptr);
```

الخطوة الثانية: هي أن تمرر للإجراء عنوان في الذاكرة مثل:

```
change(ptr);
```

على أساس أن ptr هو مؤشر لأحد المتغيرات.

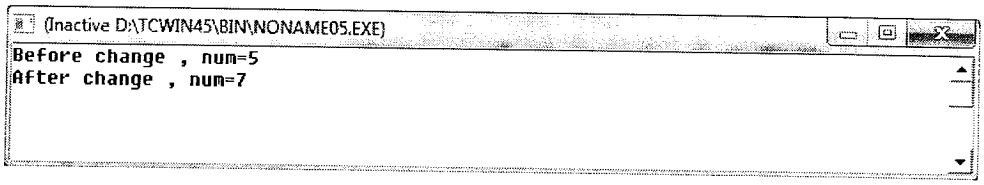
بهذا الشكل سيكون لدى الإجراء الفرعي عنوان المتغير ويتمكن بالتالي من تعديل قيمته.

يتم توضيح ذلك بالمثال الآتي:

```
//program 5-19
#include<iostream.h>
void change(int *ptr);
main()
{
int num=5;
int *ptr;
ptr=&num;
```

```
cout<<"Before change , num="<<num<<endl;
change(ptr);
cout<<"After change , num="<<num;
}
void change(int *ptr)
{
*ptr=7;
}
```

وسوف تكون النتيجة كالآتي:



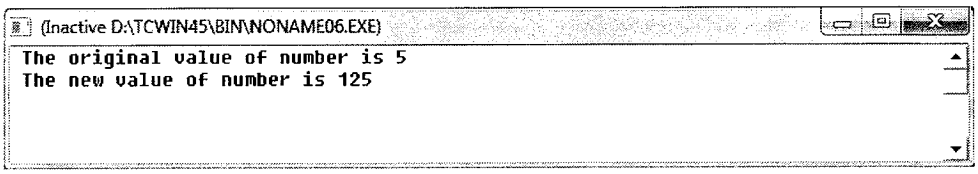
ضع إجراء لإيجاد مكعب عدد وذلك بتمرير الوسطاء للإجراء بالمرجع مع المؤشر.

البرنامج:

```
//program 5-20
#include<iostream.h>
void cube (int *n1);
main( )
{
int number = 5;
cout<< " The original value of number is " << number<<endl;
cube(&number);
cout<< " The new value of number is " << number <<endl;
return 0;
```

```
}
void cube (int *n)
{
*n = *n * *n * *n;
}
```

وسوف تكون النتيجة كالآتي:



5-3-13- العودة بأكثر من قيمة من الإجراء الفرعي:

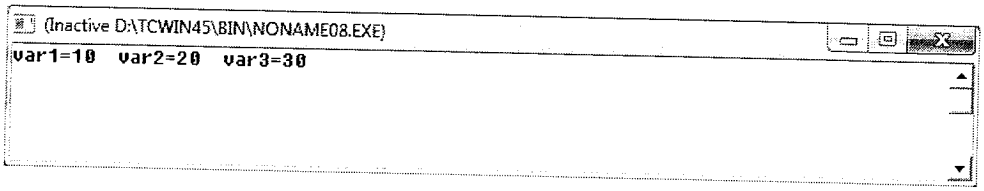
لقد ذكرنا سابقا أن استخدامreturn تعود بقيمة واحدة فقط وحتى نعود بأكثر من قيمة من الإجراء الفرعي هو أن تعلن عن متغيرات محلية بعدد المعلومات التي يعود بها الإجراء، ثم تمرر مؤشرات لهذه المتغيرات ليضع المعلومات فيها.

المثال الآتي يوضح ذلك:

```
//program 5-21
#include<iostream.h>
        void fn(int *var1,int *var2,int *var3);
main()
{
int var1,var2,var3;
fn(&var1,&var2,&var3);
cout<<"var1="<<var1<<"  "<<"var2="<<var2<<"  "<<"var3="<<var3<<endl;
}
void fn(int *p1,int *p2,int *p3)
{
```

```
*p1=10;
*p2=20;
*p3=30;
}
```

وتكون نتائج البرنامج كما يأتي:



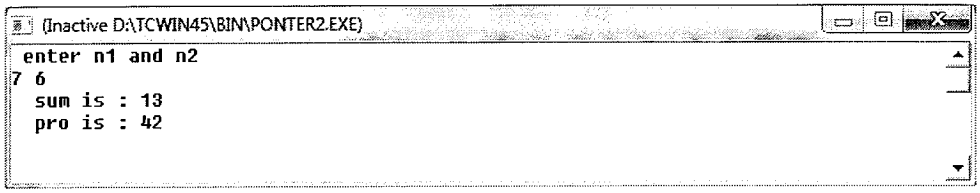
ضع برنامج إجراء يتضمن مؤشرين الأول لجمع عددين والثاني لحاصل ضربهما ثم استدعي الإجراء من البرنامج الرئيسي لإيجاد النتيجة.

البرنامج:

```
//program 5-22
#include<iostream.h>
        int fun(int n1,int n2,int *sum,int *pro)
{
*sum=n1+n2;
*pro=n1*n2;
}
int main()
{
int no1,no2,sum,pro;
cout<<" enter n1 and n2"<< endl;
cin>>no1>>no2;
fun(no1,no2,&sum,&pro);
```

```
cout<<" sum is : "<<sum<<endl;
cout<<" pro is : "<<pro<<endl;
 }
```

وتكون نتائج البرنامج كما يأتي:



14-3-5-العلاقة بين المؤشرات والمصفوفات:

توجد علاقة وثيقة بين المصفوفات والمؤشرات ؛إذ يمكن استخدام أحدها للحلول مكان الآخر ويمكن استخدام المؤشرات للقيام بأي عملية تستدعي استخدام صف مع تحديد أدلة عناصره المعنية. بفرض أنه تم التصريح عن صف ب int a[4] فإن اسم الصف a يشير إلى أول عنصر في الصف وبالتالي فإنه يمكن إسناد اسم الصف إلى مؤشر p ليشير إلى بداية الصف، وذلك كما يأتي

p=a

إن التعليمة السابقة تكافئ ;p=&a[0] ويمكننا الوصول إلى العنصر a[3] **بالأمر** ;**cout<<*(p+3)** وهي القيمة التي يشير إليها المؤشر p بعد زيادته بمقدار 3 .

ملحوظة: إن التعليمة *p+3 تعني إضافة العدد 3 على العنصر a[0] .

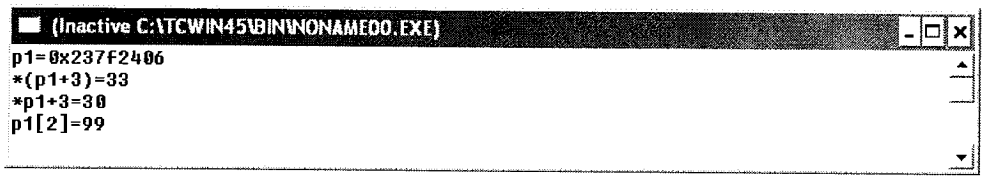
ايضا يمكننا أن نعامل اسم الصف على أنه مؤشر ويمكن استخدامه في العمليات الحسابية فمثلا التعبير عن *(a+3) يعطي قيمة العنصر a[3] كذلك يمكن إضافة أدلة على المؤشرات؛إذ إن التعبير عن p[1] تدل على العنصر a[1] أما فيما يتعلق بالمصفوفة الثنائية فيمكننا الحصول على عنوان أول عنصر فيها بالأمر ;**cout<<*a**

ويمكننا الحصول على عنوان العنصر a[i][j] بالأمر cout<<*(a)+(m*i)+j; إذ إن m عدد أعمدة المصفوفة a , ونحصل على عناصر المصفوفة الثنائية بالأمر cout<<*((a)+(m*i)+j); .

نوضح ما سبق بالبرنامجين الآتين:

```
//program 5-23
  #include<iostream.h>
   main()
   {
   int a[]={27,7,99,33,25};
    int *p1=a;
  cout<<"p1="<<p1<<endl;
    cout<<"*(p1+3)="<<*(p1+3)<<endl;
  cout<<"*p1+3="<<*p1+3<<endl;
  cout<<"p1[2]="<<p1[2]<<endl;
  }
```

وبعد التنفيذ تظهر شاشة النتائج كما يأتي:



دورة الوصلة الثانية 2014

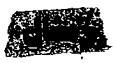
لدينا مصفوفة من المرتبة 3x2 المطلوب باستخدام المؤشرات ضع برنامج لطباعة عنوان أول عنصر في المصفوفة a ثم طباعة العدد المخزن في العنصر a[1][2] ثم طباعة عناصر الصف الأول في المصفوفة a .

البرنامج:

```
//program 5-24
   #include<iostream.h>
   const n=2;
   const m=3;
    main()
    {
    int a[n][m];
    int i,j;
    cout<<"enter the element of matrix\n";
    for(i=0;i<n;i++)
    for(j=0;j<m;j++)
    cin>>a[i][j];
   cout<<*a<<endl;
    cout<<"a[1][2]="<<*(*(a)+(m*1)+2)<<endl;
   cout<<"row1=";
   for(j=0;j<m;j++)
   cout<<*(*(a)+(m*0)+j)<<"    ";
   }
```

وتظهر شاشة النتائج كمايأتي:

```
(Inactive C:\TCWIN45\BIN\NONAME00.EXE)
enter the element of matrix
1 2 3
4 5 6
0x22772502
a[1][2]=6
row1=1    2    3
```

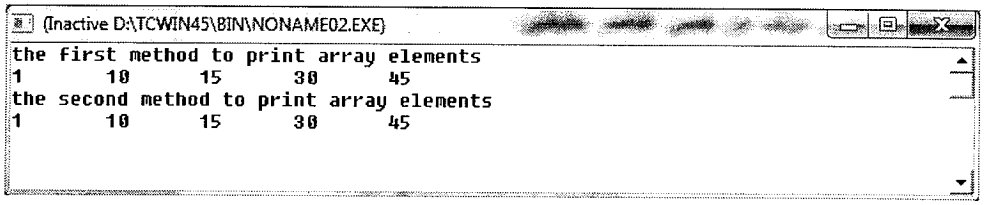


ضع برنامجا لطباعة عناصر جدول مؤلف من خمسة عناصر بطريقتين.

البرنامج:

```
//program 5-25
#include<iostream.h>
int main()
{
int a[5]={1,10,15,30,45};
int *p;
p=a; //p=&a[0];
cout<<"the first method to print array elements \n";
for(int i=0;i<5;i++)
cout<<*(p+i)<<"\t";
cout<<"\n";
cout<<"the second method to print array elements\n";
for(p=a;p<&a[5];p++)
cout<<*p<<"\t";
cout<<"\n";
return 0;}
```

وبعد التنفيذ تظهر شاشة النتائج كما يأتي:



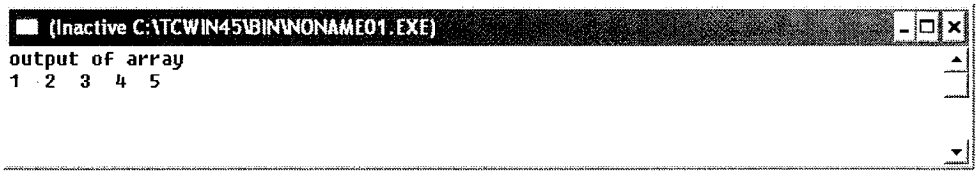
5-3-15-مصفوفة المؤشرات:

يمكننا تنظيم المؤشرات كصفوف كأي نوع من معطيات آخرى، وللتصريح عن صف من عشرة عناصر كل عنصر هو مؤشر لعدد صحيح نكتب ;int *a[10] ولإسناد عنوان متحول صحيح

b إلى العنصر الثالث نكتب a[3]=&b; ولطباعة قيمة b نكتب cout<<*a[3]; البرنامج الآتي يوضح ذلك:

```
//program 5-26
   #include<iostream.h>
   void main()
   {
     int a[]={1,2,3,4,5};
     int *x[5];
     int I;
     for(i=0;i<5;i++)
     x[i]=&a[i];
     cout<<"output of array\n";
     for(i=0;i<5;i++)
     cout<<*x[i]<<"  ";
     }
```

وتظهر شاشة النتائج كما يأتي:



5-3-16-المؤشرات والسجلات:

لقد ذكرنا سابقا انه يمكن استخدام المؤشرات مع السجلات ولعمل ذلك نحتاج إلى ثلاث خطوات:

الأولى: أن تعلن عن السجل نفسه وعلى سبيل المثال:

```
strcut book
{
char title[40];
```

```
char author[40];
int pages;
int date;}
```

والخطوة الثانية أن تعلن عن المؤشر الذي يشير إلى هذاالسجل:

```
Struct book *bk_ptr;
```

والخطوة الثالثة هو تجعل المؤشر يشير إلى أحد السجلات مثل:

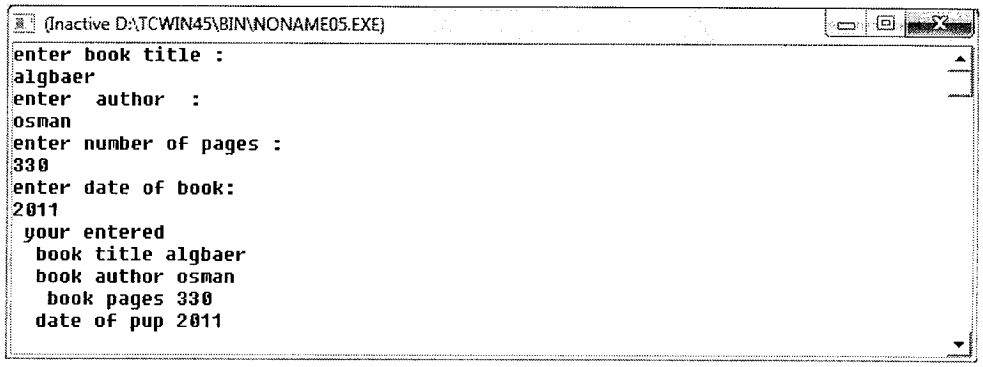
```
struct book bk;
bk_ptr=&bk;
```

المثال الآتي يوضح ذلك:

```
//program 5-27
#include<iostream.h>
struct book
{
char title[40];
char author[40];
int pages;
int date;
};
void print(struct book *bk);
main()
{
struct book bk;
cout<<"enter book title : ";
cin>>bk.title;
cout<<"enter  author  : ";
cin>>bk.author;
cout<<"enter number of pages : ";
```

```
cin>>bk.pages;
cout<<"enter date of book: ";
cin>>bk.date;
print (&bk);
}
void print(struct book *bk)
{
cout<<" your entered "<<endl;
cout<<"  book title "<<bk->title<<endl;
cout<<"  book title "<<bk->author<<endl;
cout<<"   book author  "<<bk->pages<<endl;
cout<<"  date  of  pup "<<bk->date<<endl;
}
```

وبعد تنفيذ البرنامج تظهر النتائج كمايأتي:



في بداية البرنامج نعلن عن السجل book بعد ذلك نعلن عن الإجراء print الذي يستخدم لطباعة محتويات هذاالسجل:

```
void print(strcut book *bk);
```

هذا الإعلان معناه أن الإجراء print يتوقع أن نمرر له مؤشرا لسجل من نوع book لذي أعلنا عنه سابقا.

وعندما يبدأ البرنامج نعلن عن متغير محلي book باسم bk ثم نطلب المعلومات الخاصة بإحدى الكتب ونضعها فيه وبعد أن ننتهي من استقبال بيانات الكتاب من المستخدم نمرر عنوان المتغير bk إلى الإجراء print إلحظ كيف نفعل ذلك: print(&bk); .

4-5 الحجز والغاء الحجز للذاكرة

تسمح لغة C++ باستخدام المؤشر new لحجز الذاكرة الديناميكية والمؤشر delete لتنظيف الذاكرة ويمكن استخدام المؤشر new لحجز ذاكرة موقع واحد حسب الصيغة الآتية:

```
data type *p-var = new data type (value);
```

وعلى سبيل المثال لحجز ذاكرة لموقع واحد من نوع int ووضع فيه العدد 7 فإننا نضع التعليمة الآتية:

```
int *x =new int(7);
```

كما يمكن حجز ذاكرة لعدة مواقع (n موقع) كما في حالة المصفوفات لذلك نستخدم الصيغة الآتية:

```
data type *p-var = new data type [n];
```

وعلى سبيل المثال:

لحجز ذاكرة لعشرة مواقع لبيانات من النوع الحقيقي نضع التعليمة:

```
float *y =new float[10];
```

كما يستخدم المؤشر delete لحذف موقع واحد من الذاكرة وذلك وفق الصيغة الآتية:

```
Delete  name-var;
```

وعلى سبيل المثال:

```
int *x =new int(8);
delete x;
```

أما في حالة حذف عدة مواقع نستخدم الصيغة الآتية:

```
delete[ ] p-var;
```

وعلى سبيل المثال:

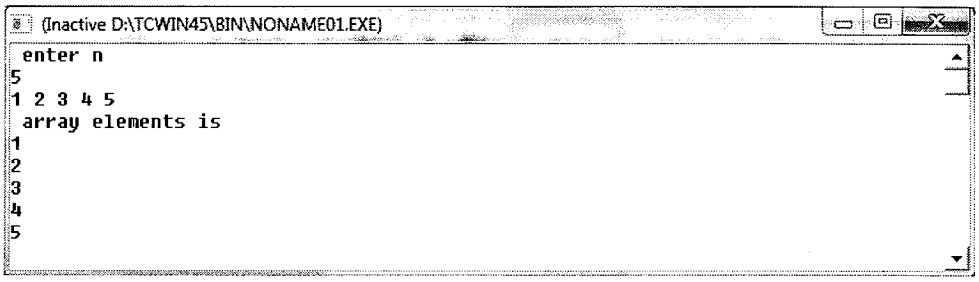
```
float *y =new float[10];
delete [ ] y;
```

اكتب برنامجا لطباعة جدول مؤلف من n عنصرا ثم طباعة هذا الجدول باستخدام المؤشر new.

البرنامج:

```
//program 5-28
#include<iostream.h>
int main()
{
int n,i;
cout<<" enter n\n";
cin>>n;
int *x = new int [n];
for(i=0;i<n;i++)
cin>>*(x+i);
cout<<" array elements is \n";
for(i=0;i<n;i++)
cout<<*(x+i)<<"\n";
return 0;}
```

وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:



5-5- تمارين محلولة 5:

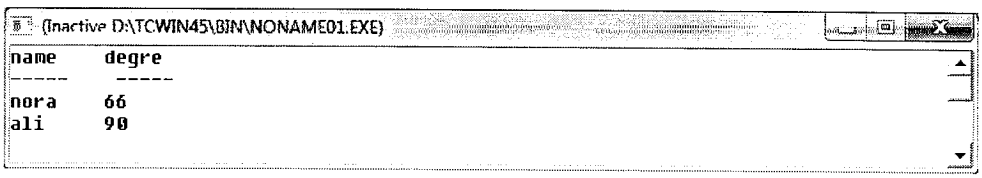
1- تقدم إلى امتحان مقرر لغات برمجة ثلاثة طلاب المطلوب:

باستخدام السجلات ضع برنامجا لطباعة أسماء الطلاب الناجحين علما أن علامة النجاح 60 وما فوق.

البرنامج:

```
#include<iostream.h>
#include<string.h>
main()
{
 struct  student
{
char name[40];
int degre;
}st[6]={"ahmad",45,"nora",66,"ali",90};
int i;
cout<<"name\t"<<"degre\n";
cout<<"-----\t"<<" -----\n";
for(i=0;i<3;i++){
if(st[i].degre>=60)
cout<<st[i].name <<"\t"<< st[i].degre<<endl;;
      }}
```

وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:



دورة الفصل 2014 + الفصل الثاني 2014

2-تقدم إلى امتحان مقرر لغات البرمجة المتقدمة ثلاثة طلاب باستخدام السجلات ضع برنامجا يتضمن تعليمة switch لإيجاد مايأتي:

1-طباعة أسماء الطلاب مع درجاتهم الامتحانية .

2-طباعة أسماء الطلاب مع دجاتهم الامتحانية بعد ترتيب الدرجات ترتيبا تنازليا .

3-طباعة إسم الطالب الي نال أعلى درجة امتحانية .

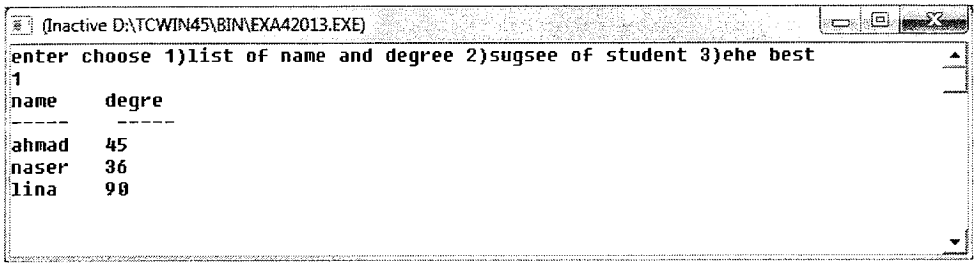
واضعا المدخلات ضمن السجل .

البرنامج:

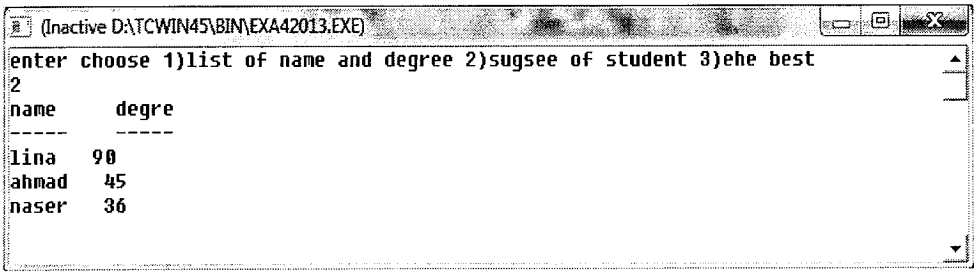
```
#include<iostream.h>
#include<string.h>
main()
{
 struct  student
{
char name[40];
int degre;
}st[6]={"ahmad",45,"naser",36,"lina",90};
int i,k,z,j,max,f;
char z1[40];
cout<<"enter choose 1)list of name and degree 2)sugsee of student
3)ehe best\n";
cin>>k;
switch(k)
{
case 1:
cout<<"name\t"<<"degre\n";
cout<<"-----\t"<<" -----\n";
for(i=0;i<3;i++)
```

```
cout<<st[i].name <<"\t"<< st[i].degre<<endl;;
 break;
 case 2:
 for(i=0;i<2;i++)
  {
  for(j=i+1;j<3;j++)
  if(st[i].degre<st[j].degre)
  { z=st[i].degre;st[i].degre=st[j].degre;st[j].degre=z;
strcpy(z1,st[i].name);strcpy(st[i].name,st[j].name);strcpy(st[j].name,z1
); }
      }
cout<<"name"<<"\t"<<" degre\n";
cout<<"-----\t"<<" -----\n";
for(i=0;i<3;i++)
cout<<st[i].name <<"   "<<st[i].degre<<endl;
break;
case 3:
max=-100;
for(i=0;i<3;i++)
if(max<st[i].degre){max=st[i].degre;k=i;}
cout<<"the best student=======> "<<st[k].name<<"\t"<<max;
 break;
default:
cout<<"the number not valid";
}}
```

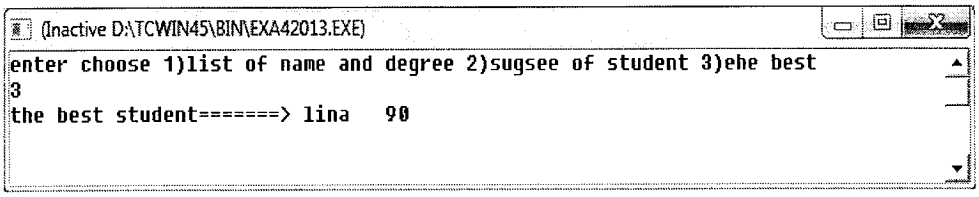
وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:



وإذا تم اختيار 2 فتكون النتائج كما يأتي:



وإذا تم اختيار 3 فتكون النتائج كما يأتي:



3- اكتشف الخطأ في البرنامج الآتي وقم بتصحيحه:

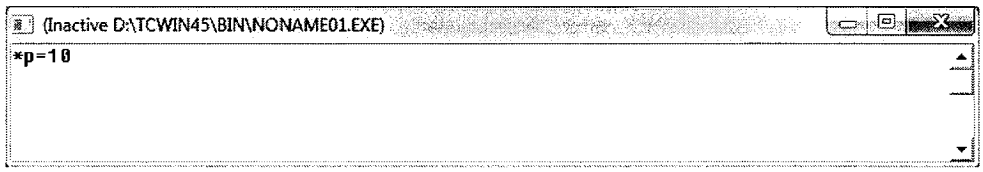
```
#include<iostream.h>
main()
{
    Int  x,*p;
   x=10;
*p=x;
cout<<*p<<endll;
```

```
}
```

الخطأ:لم نجعل المؤشر p يشير إلى المتغير x والتصحيح هو p=&x ليصبح البرنامج بالشكل الآتي:

```
#include<iostream.h>
main()
{
        int  x,*p;
        p=&x;
       x=10;
*p=x;
cout<<"*p="<<p<<endl;
}
```

وتكون نتائج البرنامج بعد التنفيذ:



4- اشرح عمل البرنامج الآتي :

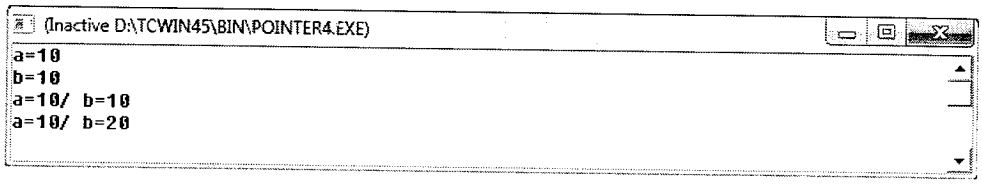
```
#include<iostream.h>
int main (){
int a = 5, b = 15;
int* p1;
int* p2;
p1 = &a;
p2 = &b;
*p1 = 10;
cout<<"a="<<a<<endl;
```

```
*p2 = *p1;
  cout<<"b="<<b<<endl;
p1 =p2;
cout << "a=" <<a<<"/ b="<< b<<endl;;
*p1 = 20;
cout << "a=" <<a<<"/ b="<< b;
return 0; }
```

الشرح:

في البداية تم الإعلان عن عددين صحيحين وهما a وخزنا فيه العدد 5 والثاني b وخزنا فيه العدد 15 ثم تم الإعلان عن مؤشرين من النوعint الأول p1 والثاني p2 ثم جعلنا المؤشر الأول يشير إلى المتغير الأول والمؤشر الثاني يشير إلى المتغير الثاني ثم خزنا في المتغير الذي يشير إليه المؤشر الأول العدد 10 ثم طبعنا قيمة العدد المخزن في a فيكون الناتج a=10، ثم خزنا في المتغير b الذي يشير إليه المؤشر الثاني القيمة نفسها المخزنة في a وهي القيمة 10 ،ثم طبعنا القيمة المخزنة في b فيكون الناتج b=10 ثم جعلنا عنوان المؤشر الأول يأخذ عنوان المؤشر الثاني نفسه ، وذلك باستخدام التعليمة p1=p2(أي أن المؤشر الأول يشير إلى المتغير b) ثم طبعنا قيمة المتغيرين a و b فيكون الناتج a=10 /b=10 ثم خزنا القيمة 20 في المتغير الذي يشير إليه المؤشر الأول وهو b ثم طبعنا قيمة المتغيرين a و b فيكون الناتج a=10/b=20 .

وتكون نتائج البرنامج بعد التنفيذ كما يأتي:

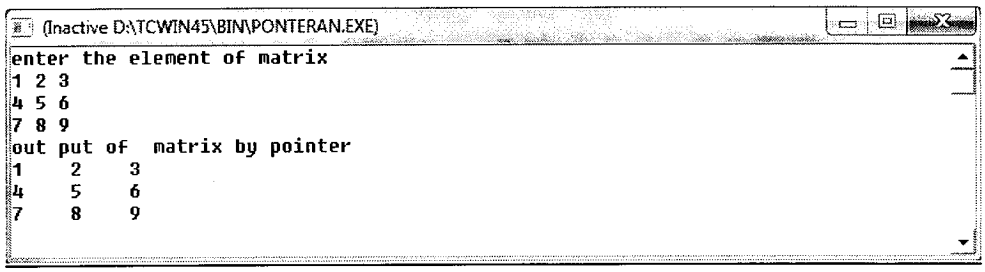


5- ضع برنامجا لإدخال مصفوفة من المرتبة nXm ثم استخدم المؤشرات لطباعة المصفوفة التي تم إدخالها.

البرنامج:

```
#include<iostream.h>
const n=3;
const m=3;
 main()
 {
 int a[n][m];
 int i,j;
  cout<<"enter the element of matrix\n";
 for(i=0;i<n;i++)
 for(j=0;j<m;j++)
 cin>>a[i][j];
 cout<<"out put of  matrix by pointer\n";
 for(i=0;i<n;i++){
 for(j=0;j<m;j++)
cout<<*(*(a)+(m*i)+j)<<"    ";
cout<<"\n";}
}
```

وتكون نتائج البرنامج بعد التنفيذ كما يأتي:



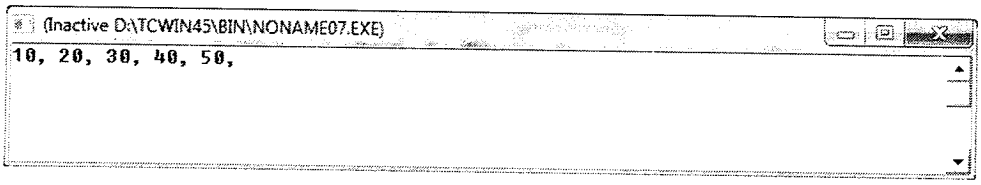
6-اشرح عمل البرنامج التالي واستنتج المخرجات:

```
#include <iostream.h>
 int main ()
{
int a[5];
int * p;
p = a; *p = 10;
p++; *p = 20;
p = &a[2]; *p = 30;
p = a + 3; *p = 40;
p = a; *(p+4) = 50;
for (int i=0; i<5; i++)
cout << a[i] << ", ";
return 0;
}
```

الشرح:

في البداية أعلنا عن مصفوفة a ثم أعلنا عن مؤشر p وجعلنا المؤشر يشير إلى أول عنصر في المصفوفة بالتعليمة p=a ثم خزنا في هذا العنصر العدد 10 باستخدام التعليمة *p=10 ثم نقلنا المؤشر ليشير إلى العنصر الثاني في المصفوفة a باستخدام التعليمة p++ ثم خزنا في هذا العنصر العدد 20 ثم جعلنا المؤشر يشير إلى العنصر الثالث باستخدام التعليمة p=&a[2] وخزنا في هذا العنصر العدد 30 ثم نقلنا المؤشر ليشير إلى العنصر الرابع باستخدام التعليمة p=a+3 وخزنا في هذا العنصر العدد 40 ثم جعلنا المؤشر يشير إلى العنصر الأخير باستخدام التعليمة p=a وخزنا في هذا العنصر العدد 50 ثم طبعنا جميع عناصر المصفوفة a ليكون الناتج 10,20,30,40,50 .

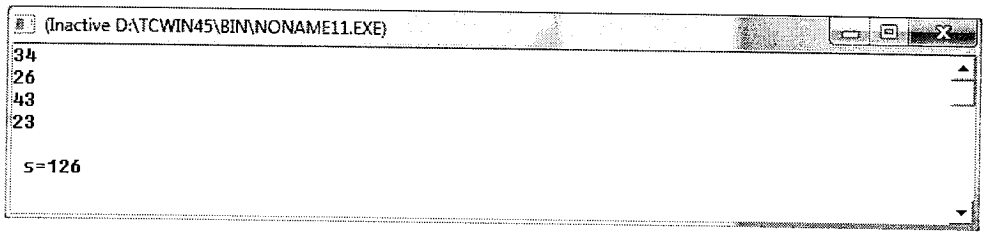
وتكون نتائج البرنامج بعد التنفيذ كما يأتي:



7)-باستخدام المؤشرات ضع برمانجا لطباعة عناصر مصفوفة مؤلفة من سطرين وعمودين ثم إيجاد مجموع عناصر هذه المصفوفة علما أن عناصر المصفوفة معطاة ضمن البرنامج:

```
#include<iostream.h>
main()
{ int i, first_array[2][2]={{34,26},{43,23} };
int *ptr=& first_array[0][0],a; int s=0;
for(i=0;i<4;i++){
a=*ptr++;cout<<a<<endl;
 s=s+a;
 }
cout<<"\n s="<<s;}
```

ويعد تنفيذ البرنامج تظهر نتائج البرنامج كما يأتي:



تمارين 5

1-تتبع البرنامج الآتي مع الشرح ، و حاول أن تستنتج شكل المخرجات

```
#include <iostream.h>
int main ()
{
int value1 = 15, value2 = 25;
int* p1;
int* p2;
p1 = &value1;
p2 = &value2;
*p1 = 33;
*p2 = *p1;
p1 = p2;
*p1 = 70;
cout << "value1==" <<value1<<"\t value2=="<< value2<<endl;
return 0;
}
```

2- تتبع البرنامج الآتي مع الشرح ، و حاول أن تستنتج شكل المخرجات.

```
#include <iostream.h>
int main ()
{
```

```
int firstvalue, secondvalue;
int * mypointer;
mypointer = &firstvalue;
*mypointer = 10;
mypointer = &secondvalue;
*mypointer = 20;
cout << "firstvalue is " << firstvalue << endl;
cout << "secondvalue is " << secondvalue << endl;
return 0;}
```

3-ضع برنامجا لإدخال جدول مؤلف من n عنصرا واستخدم المؤشرات لطباعة عناصر هذا الجدول.

4-تتبع عمل البرنامج الآتي واستنتج المخرجات:

```
#include<iostream.h>
 main()
{
int x= 300;
int &ref=x;
cout<<"x= "<<x<<endl;
cout<<" p= "<<ref<<endl;
ref= 600;
cout<<" x= "<<x<<endl;
cout<<" p= "<<ref<<endl;
```

}

5- ضع برنامجا لإدخال جدول مؤلف من n عنصرا ثم استخدم المؤشرات لإيجاد مجموع عناصر هذا الجدول.

6- ضع برنامجا لإدخال جدول مؤلف من n عنصرا ثم استخدم المؤشرات لإيجاد أكبرالعناصر في هذا الجدول.

7-ضع برنامجا لإدخال جدول مؤلف من n عنصرا ثم استخدم المؤشرات لإيجاد المتوسط الحسابي لعناصر هذا الجدول.

8-أكتب عبارة C++ صحيحة لكل من الآتي (افترض أنه تم الإعلان عن عددين صحيحين a و b وتم تمهيد قيمة المتغير a عند 200000) :

- الإعلان عن مؤشر ptr ليشير إلى متغير من النوع int.
- تعيين عنوان المتغير a إلى المؤشر ptr
- طباعة القيمة التي يشير إليها المؤشر ptr.
- تعيين القيمة التي يشير إليها المؤشر ptr إلى المتغير b.
- طباعة قيمة المتغير b.
- طباعة عنوان المتغير b.
- طباعة العنوان المخزن في المؤشر ptr. (هل تتساوى هذه القيمة مع عنوان المتغير a؟)

الفصل السادس

الأصناف

6-1-الصنف(الفئة):

هو مجموعة من البيانات يتم تمثيلها على شكل معطيات أعضاء (Data member) عادة تكون أسماء لمتغيرات الصنف وتنطوي تحت اسم أعضاء خاصة (private) كما يمكنها أن تقوم بعدة عمليات يتم تمثيلها على شكل أعضاء إجراءات (member Functions) وتنطوي تحت اسم أعضاء عامة(public) .

والصنف مشابه من حيث الصيغة للسجلات فيبدأ بالكلمة الأساسية class يتبعه اسم الصنف ثم نضع جسم الصنف محصور ضمن الآهلتين { } وننهي الصنف بالفاصلة المنقوطة ويكون على الشكل الآتي:

```
class class_name{
private:
عادة اسماء متغيرات الصنف
public:
اجراءات مختلفة
};
```

استخدم مفهوم الصنف لحساب مساحة المستطيل.

البرنامج:

```
//program 6-1
#include <iostream.h>
 class rectangle
{
```

```
 private:
int x,y;
public:
void val(int,int);
int area (void) ;
};
void rectangle::val(int a, int b) {
x=a;
y=b;
}
int rectangle::area(void)
{return(x*y);}
int main() {
rectangle x,z;
x.val (3,4);
z.val(5,8);
cout<<"area:"<<x.area()<<endl;;
 cout<<"area:"<<z.area();
}
```

شرح البرنامج:

نلحظ أن البرنامج يحتوي صنف يحمل الاسم rectangle والأعضاء العامة هي الإعلان عن إجرائين :

الأول ;(void val(int,int لإدخال عددين.

والثاني ; (int area (void لحساب مساحة المستطيل.

أما الأعضاء الخاصة هي المتغيرين int x,y;

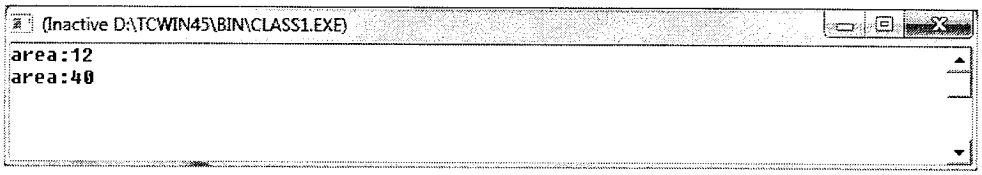
وفي أثناء كتابة الإجراء يجب ذكر اسم الصنف يليه :: ثم اسم الإجراء

int rectangle::area(void)

وكما في السجلات يجب وضع متغير الصنف في البرنامج الرئيسي وذلك بكتابة اسم الصنف يليه متغير الصنف أومتغيرات الصنف rectangle x,z; .

وفي أثناء استخدام هذه المتغيرات نضع اسم المتغير يليه نقطة . ثم اسم الاجراء: x.val (3,4);و z.val(5,8); .

وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:



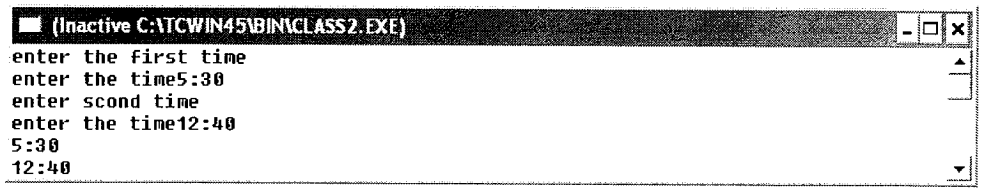
ضع برنامجا يتضمن صنف يحمل الاسم time ويتضمن إجرائين الأول ()enter يستخدم لإدخال الوقت والإجراء الثاني display لإظهار الوقت المدخل.

البرنامج:

```
//program 6-2
#include <iostream.h>
 #include <iomanip.h>
class Time {
        public:
       void enter();
       void display();
        private:
       int hour;     // 0 - 23 (24-hour clock format)
       int minute;   // 0 - 59
                     };
void Time::enter()
{   char d;
```

```
        cout<<"enter the time";
        cin>>hour>>d>>minute;
  }
void Time::display()
{
        cout << hour << ":"<<minute <<endl;
}
int main()
{
        Time t1,t2;
                cout<<"enter the first time"<<endl;;
                t1.enter();
                cout<<"enter scond time"<<endl;
                t2.enter();
                t1.display();
                t2.display();
                }
```

وتظهر نتائج البرنامج السابق كما يأتي:

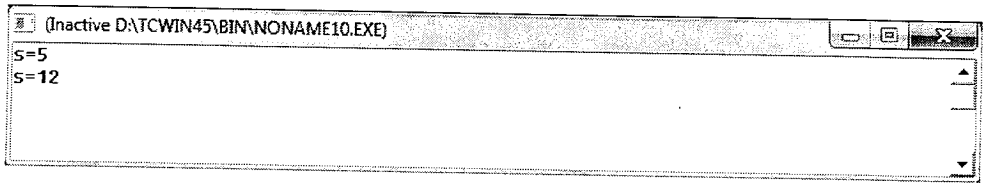


باستخدام الأصناف ضع برنامجا لجمع عددين ثم إظهار ناتج الجمع .

البرنامج:

```
//program 6-3
#include<iostream.h>
class sumnumber{
private:
int a,b,s;
public:
 void sum(int a,int b);
 void display();
};
 void sumnumber ::sum(int a,int b)
{
s=a+b;
      }
       void sumnumber ::display()
{
      cout<<"s="<<s<<endl;}
int main()
{  sumnumber x,z;
       x.sum(2,3);
       z.sum(5,7);
       x.display();
        z.display();
       }
```

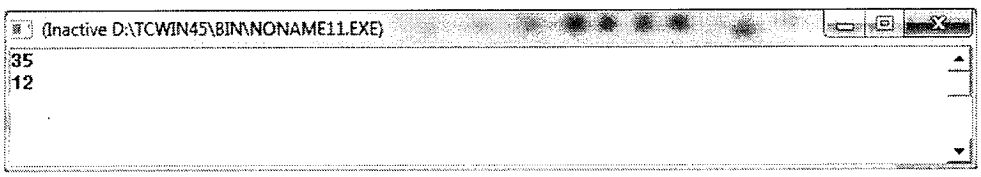
وتظهر شاشة الإخراج:



ملحوظة :يمكننا استخدام المتغيرات (الأعضاء الخاصة)كمتغيرات عامة والبرنامج يوضح ذلك.

```
//program 6-4
#include<iostream.h>
  class mayclass{
 public:
 int i,j,k;
 };
int main()
 {  mayclass a,b;
        a.i=7;
        a.j=5;
        a.k=a.i*a.j;
        b.k=a.i+a.j;
        cout<<a.k<<endl;
        cout<<b.k<<endl;

        }
```

وتظهر شاشة النتائج كما يأتي:



6-2- العلاقة بين السجلات والأصناف:

لقد قلنا فيما سبق أن الصنف مشابه للسجل ولكن الفارق الوحيد هو أن أعضاء الصنف تكون أعضاء خاصة بشكل افتراضي . أي إذا لم يتم استعمال الكلمات الأساسية public أو private تكون أعضاء الصنف خاصة.

3-6- الإجراءات الصديقة:

تعريف الإجراء الصديق: هو عبارة عن إجراء تفاصيله خارج الصنف مع امتيازه بحق الوصول لجميع أعضاء الصنف.

ولتعريف الإجراء الصديق نستخدم الكلمة المفتاحية friend كما يأتي:

Friend type funname(class name parameters)

أو

type Friend funname(class name parameters)

ملحوظات هامة:

1-مع أن تعريف الإجراء الصديق يتم داخل الصنف ، إلا أنه لايُعدّ عضو إجراء من أعضاء الصنف ، لذلك من الخطأ استدعاؤه باستخدام اسم الكائن مع أداة الوصول (.).

2-لا تؤثر مستويات الحماية في التصريح عن الإجراء الصديق ؛ لذلك يمكن التصريح عن الإجراء الصديق في أي موضع داخل الصنف.

3- الإجراء الصديق له امتيازات عضو الإجراء في الصنف ، أي أنه له المقدرة على الوصول إلى جميع خصائص الصنف .

4-يتم استدعاء الإجراء الصديق عن طريق اسمه مع الوسائط.

المثال الآتي يوضح ذلك:

```
//program 6-5
#include <iostream.h>
class circle
{
int r;
float ar,c;
public:
```

```
circle (int r1){r=r1;}
void area ()
{
ar=3.14*r*r;
}
void cir()
{
c=2*3.14*r;
}
void display()
{
cout<<"area = "<<ar<<endl;
cout<<"cir =  "<<c<<endl;
}
friend  compare (circle,circle);
};
compare (circle c1,circle c2 )
{
if (c1.r>c2.r)
cout<<"r1 is greator";
else
cout<<"r2 is gretaor";
return c1.r>c2.r;
}
int main()
{
int r1,r2;
```

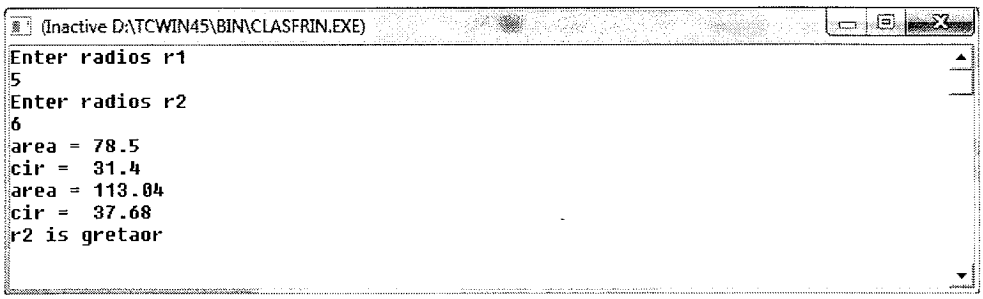
```
cout<<"Enter radios r1"<<"\n";
cin>>r1;
cout<<"Enter radios r2"<<"\n";
cin>>r2;
circle cc1(r1),cc2(r2);
cc1.area();
cc1.cir();
cc1.display();
cc2.area();
cc2.cir();
cc2.display();
compare (cc1,cc2);
  return 0;
}
```

الشرح:

في هذا البرنامج تم الإعلان عن صنف اسمه **circle** ويتضمن الإجراءات الآتية:

الأول اسمه **circle** يتضمن التعليمة **r1=r** والإجراء الثاني يحمل الاسم **area** لحساب مساحة الدائرة والثالث اسمه **cir** لحساب محيط الدائرة والرابع اسمه **display** لإظهار قيمة مساحة الدائرة ومحيطها وأخيرا تم الإعلان عن إجراء صديق اسمه **compare** وتم كتابة محتويات هذا الإجراء خارج الصنف ومن السهل معرفة مايتضمنه البرنامج الرئيسي.

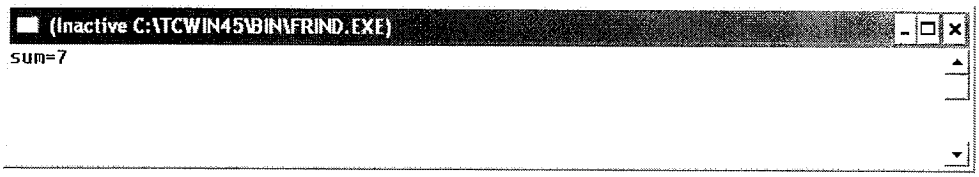
وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:



تمرين :اشرح عمل البرنامج الآتي:

```
//program 6-6
  #include<iostream.h>
  #include<string.h>
  class mayclass{
    int a,b;
  public:
   friend int sum(mayclass x);
   void st(int i,int j);
   };
   void mayclass::st(int i,int j)
  { a=i;
   b=j;
  }
   int sum(mayclass x)
   { return x.a+x.b;}
   int main()
   {  mayclass n;
    n.st(3,4);
    cout<<"sum="<<sum(n);
   return 0;}
```

وتظهر شاشة النتائج كما يأتي:



6-7- الأصناف الصديقة:

تعريف الصنف الصديق: هو عبارة عن صنف يستطيع كل أعضاؤه الإجرائية أن تستخدم خصائص الصنف الآخر وللتصريح عن الصنف الصديق نستخدم الكلمة المفتاحية friend .

وعلى سبيل المثال لجعل الصنف tow صديق للصنف one نكتب العبارة الآتية:

friend class tow داخل الصنف one .

ملحوظات:

1-يمكن التصريح عن الصنف الصديق بأي موضع من الصنف الآخر.

2- الصداقة في الأصناف ليست تبديليه أي إذا كان الصنف A صديق للصنف B فهذا لايعني أن الصنف B صديق للصنف A .

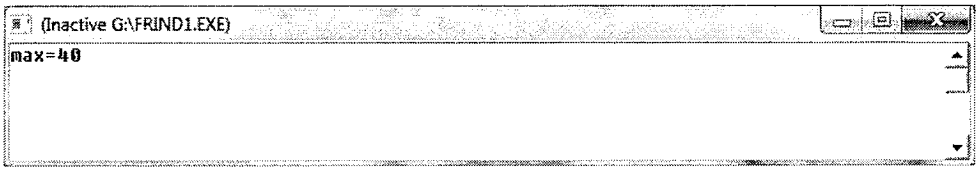
3- الصداقة في الأصناف ليست متعدية بمعنى أنه إذا كان الصنف A صديقا للصنف B والصنف B صديقا للصنف C فهذا لايعني أن الصنف A صديقا للصنف C .

البرنامج الآتي يوضح مثل هذا الاستخدام.

```
//program 6-7
   #include<iostream.h>
   class TwoValues  {
   int a;
   int b;
   public:
   TwoValues(int i, int j)     {a = i, b= j;}
   friend class Max;
   };
   class Max {
   public:
   int max(TwoValues x);
   };
```

```
int Max::max (TwoValues x)
{
return x.a> x.b? x.a: x.b;
}
int main( )
{
TwoValues ob(40, 20);
Max m;
cout<<"max="<< m.max(ob);
return 0;
}
```

وتظهر النتائج كما يأتي:



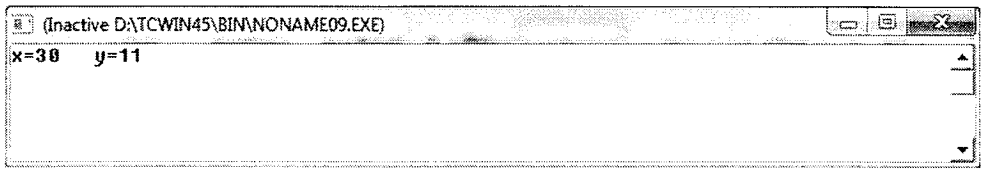
6-5- الإجراءات السطرية والأصناف

يمكن للإجراءات السطرية أن تكون أعضاء في الصنف وهذا مايوضحه المثال الآتي:

```
//program 6-8
   #include<iostream.h>
   class myclass
   { private:
   int x,y;
   public :
   void inl(int i,int j);
   void display();
```

```
};
  inline void myclass::inl(int i,int j)
  { x=i;
y=j;
 }
 inline void myclass :: display()
{
 cout<<"x="<<x<<"   "<<"y="<<y<<endl;
}
 int main()
 {
 myclass t;
 t.inl(30,11);
 t.display();
 return 0;}
```

وتظهر شاشة الاخراج كما يأتي:



6-6- تعريف الإجراءات السطرية ضمن الصنف:

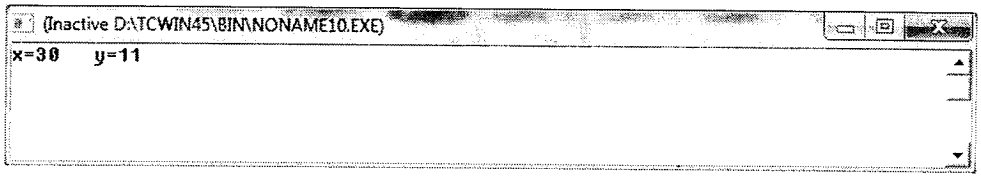
عند تعريف إجراء ما داخل الصنف فإنه تلقائيا يجعله تابعا سطريا وليس من الضروري سبقه بكلمة inline .

على سبيل المثال يمكن كتابة المثال السابق بالشكل الآتي:

```
//program 6-9
   #include<iostream.h>
   class myclass
```

```
{ private:
int x,y;
public :
void inl(int i,int j);
void display();
};
void myclass::inl(int i,int j)
  { x=i;
y=j;
 }
void myclass :: display()
{
 cout<<"x="<<x<<"   "<<"y="<<y<<endl;
}
 int main()
 {
 myclass t;
 t.inl(30,11);
 t.display();
 return 0;}
x.show();
 return 0;}
```

وتظهر شاشة النتائج كما يأتي:



6-7- إجراءات البناء والهدم

6-7-1 إجراء البناء

هو عبارة عن إجراء يأخذ اسم الصنف نفسه ويتم استدعاؤه مباشرة عند استدعاء كائن من الصنف ويستخدم لإعطاء متغيرات الصنف قيما ابتدائية.

ملحوظات هامة:

1-لايمكن تهيئة بيانات العضو في أثناء التعريف

2-لايوجد return لإجراءات البناء

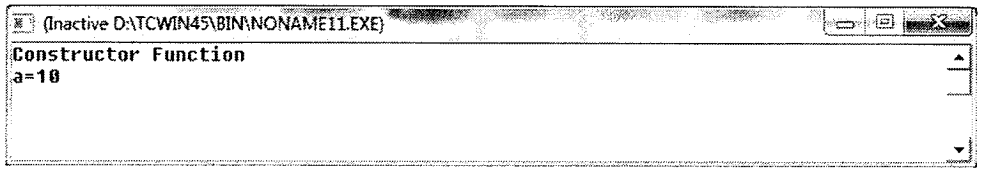
3-يمكن أن يحوي الصنف أكثر من إجراء بناء

البرنامج الآتي يوضح هذا الإجراء:

```
//program 6-10
   #include <iostream.h>
   class myclass
   {
   int a;
   public:
   myclass();
   void show();
   };
   myclass::myclass()
   {
   cout<<"Constructor Function \n";
   a=10;
   }
   void myclass::show()
   {
   cout<<"a="<<a<<"\n";
```

```
}
int main()
{
myclass ob;
ob.show();
return 0;
}
```

وتظهر شاشة تنفيذ البرنامج النتائج كما يأتي:



والآن نقدم شرح بسيط لأهم ماورد في البرنامج:

تم الإعلان عن إجراء ()mayclass وهو إجراء بناء لأن اسمه من اسم الصنف وقد وضعنا في نهاية هذا الإجراء فاصلة منقوطة لأننا لانريد كتابة محتويات هذا الإجراء داخل الصنف، ثم كتبنا الكود ()myclass::myclass بعد إقفال الصنف ولكتابة محتويات إجراء خارج الصنف نتبع الطريقة الآتية:

نكتب في البداية اسم الصنفmayclass يليه مؤشر دقة الوصول :: ثم اسم الإجراء ()mayclass إلحظ أننا كتبنا هذا الإجراء بدون datatype لأنه إجراء بناء عكس إجراء ()show ؛إذ بدأ بكلمة void وتم كتابته على النحو اللآتي :()void myclass::show .

2-7-6 إجراء الهدم :

وهو عبارة عن إجراء عضو في الصنف ويتشكل اسمه من المحرف ~ مضافا إليه اسم الصنف ويستخدم لإنهاء الأعمال الداخلية للكائنات حتى يستطيع النظام استعادة الذاكرة لحفظ كائنات جديدة.

ملاحظات على إجراء الهدم:

1-ليس له وسائط ولا يرجع قيمة.

2-يمكن للصنف أن يحتوي على إجراء هدم واحد فقط .

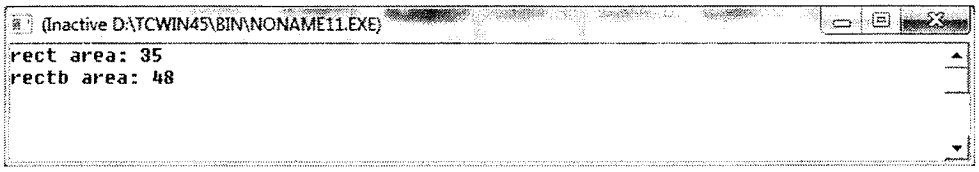
3-لايمكن تحميل إجراء الهدم بشكل زائد

البرنامج الآتي يوضح هذا الإجراء:

```
//program 6-11
   #include <iostream.h>
   class rectangle {
   int *width, *height;
   public:
   rectangle (int,int);
   ~rectangle ();
   int area () {return (*width * *height);}
   };
   rectangle::rectangle (int a, int b) {
   width = new int;
   height = new int;
   *width = a;
   *height = b;
   }
   rectangle::~rectangle () {
   delete width;
   delete height;
   }
   int main () {
   rectangle rect (5,7), rectb (8,6);
```

```
cout << "rect area: " << rect.area() << endl;
cout << "rectb area: " << rectb.area() << endl;
return 0; }
```

وتظهر شاشة تنفيذ البرنامج النتائج كما يأتي:



6-8- أعضاء الصنف الساكنة:

تنقسم هذه الأعضاء إلى قسمين بيانات ساكنة وأعضاء إجراءات ساكنة.

6-8-1-البيانات الساكنة:

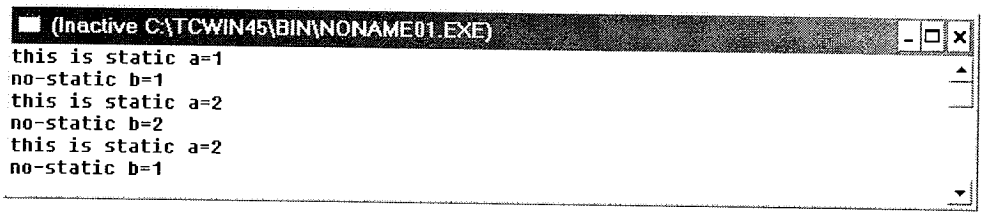
هي عبارة عن متغيرات مشتركة بين كل كائنات الصنف ؛إي أن كل كائنات الصنف لها النسخة نفسها من المتغيرات. وعند التصريح عن البيانات الساكنة نستخدم الكلمة المفتاحية static .

المثال الآتي يوضح ذلك:

```
//program 6-12
   #include<iostream.h>
   class sh
   {static  int a;
   int b;
     public :
   void set(int i,int j){a=i;b=j;}
   void show();
   };
```

```
int sh::a;
void sh::show()
{ cout<<"this is static a="<<a<<endl;
 cout<<"no-static b="<<b<<endl;
 }
 int main()
 {
 sh x,y;
 x.set(1,1);
 x.show();
 y.set(2,2);
 y.show();
 x.show();
 return 0;}
```

وتظهر شاشة تنفيذ البرنامج النتائج كمايأتي:



6-8-2-أعضاء إجراءات ساكنة:

هي إجراءات مشتركة بين كل إجراءات الصنف وعند التصريح عن الإجراءات الساكنة تسبق بالكلمة المفتاحية static

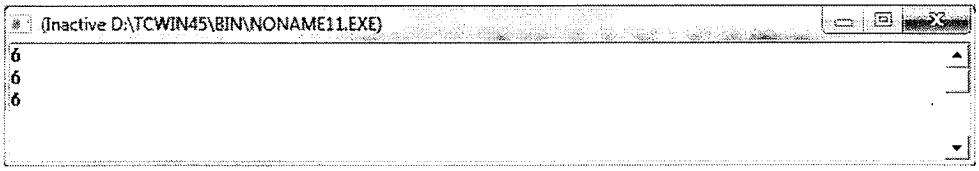
البرنامج الآتي يوضح :

```
//program 6-13
 #include <iostream.h>
 class property
```

```
{
private:
int static n;
public:
property ()
{++n;}
static void display()
{
cout<<n<<endl;
}};
int property::n=2;
int main()
{
property s1,s2,s3;
property::display();
s1.display();
s3.display();
return 0;
}
```

هنا تم تعريف الإجراء display() على أنه إجراء ساكن ومن ثم تم تهيئة المتغير n بالعدد 2 ، بعدها تم اشتقاق الكائنات من الصنف property بالكود property s1,s2,s3 ، ومن ثم تمت طباعة المخرجات بطرق مختلفة .

وتظهر شاشة تنفيذ البرنامج النتائج كما يأتي:



6-9- التحميل الزائد للإجراءات البانية:

يمكن للإجراءات البانية أن تحمل بشكل زائد وهنالك ثلاثة أسباب:

1-زيادة المرونة .

2-إمكانية إنشاء أغراض مهيأة بقيم ابتدائية .

3-لتعريف الإجراءات البانية الناسخة .

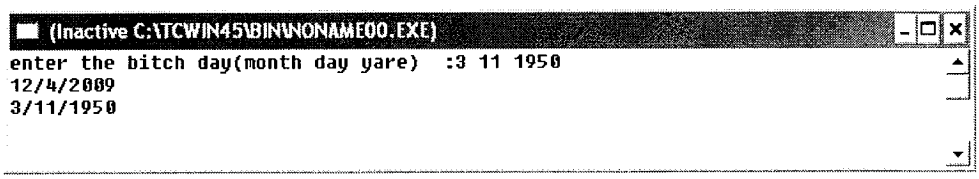
6-9-1 -التحميل الزائد لزيادة المرونة:

البرنامج الآتي يوضح ذلك:

```
//program 6-14
   #include<iostream.h>
   #include<stdio.h>
   class date
   {
    int day, month,year;
    public :
    date(char *d);
    date(int m,int d,int y);
    void show();
    };
    date::date(char *ch)
    {
    cout<<"enter the bitch day(month day yare)  :";
   scanf("%d%d%d",&month,&day,&year);
   }
   date::date(int m,int d,int y)
   { day=d;
```

```
month=m;
year=y;
}
 void date::show()
 {
cout<<month<<"/"<<day;
cout<<"/"<<year<<"\n";
}
int main()
{
date ob1(12,4,2009);
date ob2("/");
ob1.show();
 ob2.show();
return 0;
}
```

وتظهر شاشة تنفيذ البرنامج النتائج كما يأتي:



10-6- المؤشر This:

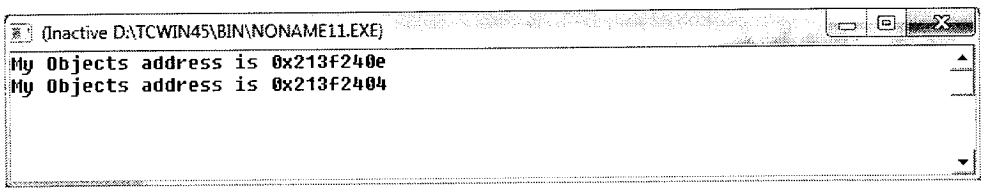
يمتلك كل كائن في صنف مؤشراً خاصاً يسمى this يشير إليه، وباستخدام هذا المؤشر يستطيع أي إجراء عضو في الصنف معرفة عنوان الكائن الذي استدعاه .

المثال الآتي يوضح هذا :

```
//program 6-15
 #include<iostream.h>
class where
 {
  private:
        char chararray[10];
  public:
        void reveal( ) {
 cout <<"My Objects address is "<<this<<endl;  }
};
main( )
{
  where w1,w2;
  w1.reveal( );
  w2.reveal( );
}
```

ينشئ هذا البرنامج كائنات من النوع where، ويطلب من كل منها عرض عنوانه باستعمال الإجراء reveal( )، والذي يعرض قيمة المؤشر this.

وبعد تنفيذ البرنامج تكون النتائج كما يأتي:



نلحظ إن عنوان الكائن w2 يبتعد 10 Bytes عن عنوان w1، وذلك لأن البيانات في كل كائن تتألف من مصفوفة من 10 Bytes.

6-11: تمارين محلولة 6

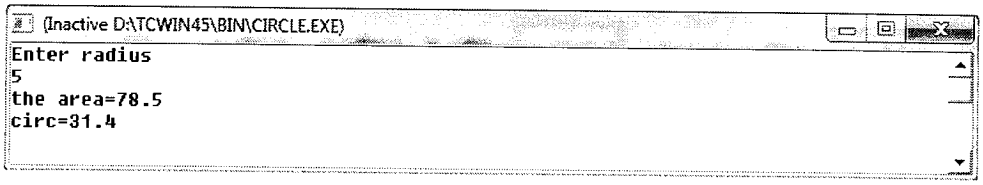
1- باستخدام الأصناف ضع برنامجا لحساب مساحة ومحيط الدائرة :

البرنامج:

```
//program 6-16
#include <iostream.h>
class circle
{
private :
int r;
float area1,c;
public :
void get_r()
{
cout<<"Enter radius \n";
cin>>r;
}
void area()
{
area1=r*r*3.14;
}
void cir()
{
c=2*r*3.14;
}
void display ()
{
cout<<"area="<<area1<<endl;
```

```
cout<<"circ="<<c<<endl;
}
};
main()
{
circle c;
c.get_r();
c.area();
c.cir();
c.display();
}
```

وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:



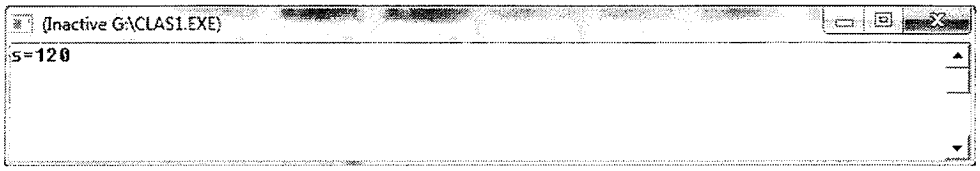
2- باستخدام الأصناف ضع برنامجا لحساب n! .

البرنامج:

```
//program 6-17
   class myclass {
   int i,s;
   public:
   int fact (int n){s=1;
   for(i=1;i<=n;i++)
   s=s*i;
   }
```

```
void show ( ) {cout <<"s="<<s; }
};
int main( )
{
myclass ob;
ob.fact(5);
ob.show( );
return 0;
}
```

وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:



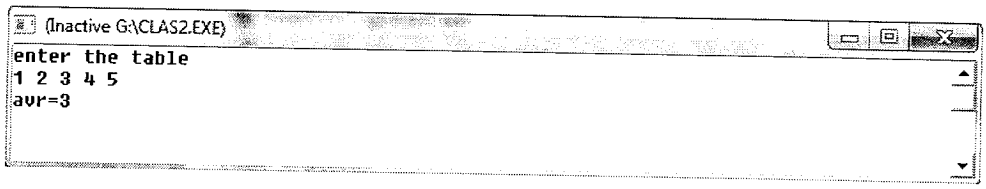
3- باستخدام الأصناف ضع برنامجا لحساب المتوسط الحسابي لجدول مؤلف من خمسة عناصر .

البرنامج:

```
//program 6-18
   #include <iostream.h>
   class myclass {
   int i,a[10];
   float avr,s;
   public:
   void av (int n){cout<<"enter the table\n";s=0;
                          for(i=0;i<n;i++) {
                          cin>>a[i];s=s+a[i];}
                          avr=s/n;
```

```
}
void show ( ) {cout <<"avr="<<avr; }
};
int main( )
{
myclass ob;
ob.av(5);
ob.show( );
return 0;
}
```

ويعد تنفيذ البرنامج تظهر النتائج كما يأتي:



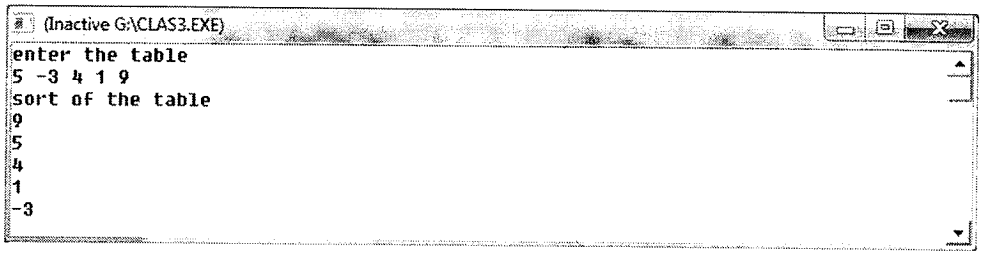
4- باستخدام الأصناف ضع برنامجا لترتيب جدول مؤلف من خمسة عناصر ترتيبا تنازليا.

البرنامج:

```
//program 6-19
  #include <iostream.h>
  class myclass {
  int i,j,z,a[10];
  public:
  void sort (int n){cout<<"enter the table\n";
  for(i=0;i<n;i++)
  cin>>a[i];
  cout<<"sort of the table\n";
```

```
for(i=0;i<n-1;i++)
for(j=i+1;j<n;j++)
if(a[i]<a[j]){z=a[i];a[i]=a[j];a[j]=z;}
}
void show (n ) {for(i=0;i<n;i++)
cout<<a[i]<<endl; }
};
int main( )
{
myclass ob;
ob.sort(5);
ob.show( 5);
return 0;
}
```

وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:



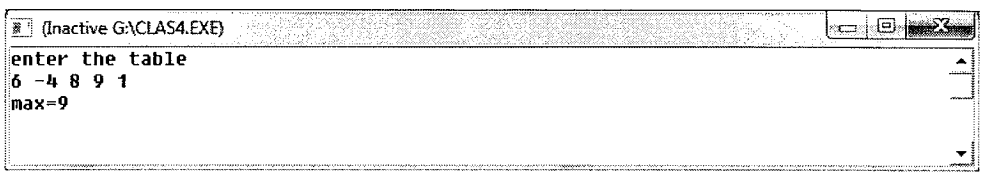
5- باستخدام الأصناف ضع برنامجا لإيجاد أكبر الأعداد في جدول مؤلف من خمسة عناصر.

البرنامج:

```
//program 6-20
   #include <iostream.h>
   class myclass {
```

```
int i,max,a[10];
public:
void sort (int n){cout<<"enter the table\n";
for(i=0;i<n;i++)
cin>>a[i];
max=a[0];
for(i=1;i<n;i++)
if(max<a[i])max=a[i];
}
void show ( ) {
cout<<"max="<<max; }
};
int main( )
{
myclass ob;
ob.sort(5);
ob.show( );
return 0;
}
```

وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:

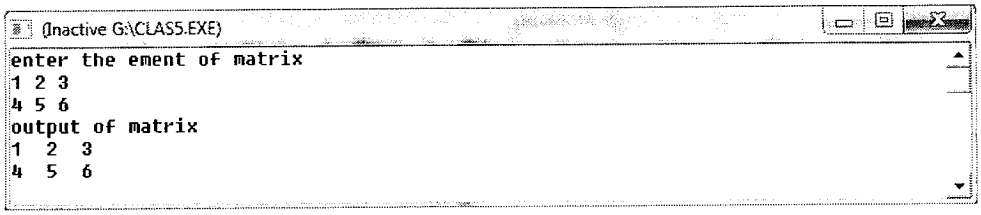


6- باستخدام الأصناف ضع برنامجا لإدخال وإخراج مصفوفة من المرتبة 3X2 .

البرنامج:

```
//program 6-21
 #include <iostream.h>
class myclass {
int i,j,a[10][10];
public:
void read (int m ,int n){cout<<"enter the ement of matrix\n";
for(i=0;i<m;i++)
for(j=0;j<n;j++)
cin>>a[i][j];
}
void print (int m ,int n ) { cout<<"output of matrix\n";
for(i=0;i<m;i++){
for(j=0;j<n;j++)
cout<<a[i][j]<<"  ";
cout<<"\n";
}
}
};
int main( )
{
myclass ob;
ob.read(2,3);
ob.print( 2,3);
return 0;
}
```

وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:



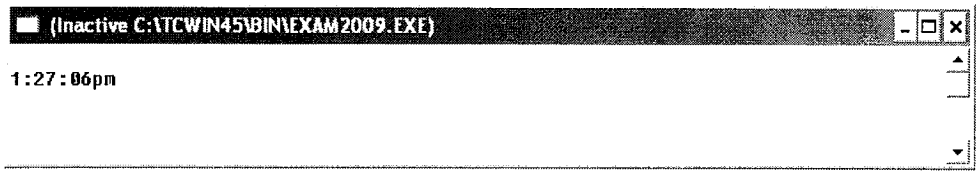
7- ضع برنامجا يستخدم class time مؤلف من المعطيات الآتية:second,minute,hour ولإجراءات التالية constructer للإسناد القيم الافتراضية و settime لوضع قيم زمن جديدة مع التحقيق من صلاحيتها و printime لطباعة الوقت وفق النظام am , pm .

البرنامج:

```
//program 6-22
   #include<iostream.h>
     class time
     {
     public :
   time();
     void settime(int  ,int ,int  ,char );
     void primtmili();
   void  printtime();
   private :
     int hour;
     int minute;
     int second;
     char ch;
     };
     time::time(){hour=minute=second=0;ch=':';}
     void time::settime(int h,int m,int s,char c)
     {
```

```
  hour=(h>=0 && h<24)?h:0;
  minute=(m>=0 && m<60)?m:0;
  second=(s>=0 && s<60)?s:0;
  ch=c;}
  void time::printtime()
 { cout<<((hour ==0 || hour ==12)?12:hour %12)
  <<":"<<(minute <10 ? "0":"")<<minute
  <<":"<<(second <10  ? "0":"")<<second
  <<(hour <12? "am" :"pm");
 }
main()
{ time t;
t.settime(13,27,6,':');
cout<<endl;
t.printtime();
return 0;
 }
```

وتظهر شاشة النتائج كما يأتي:



تمارين 6

1- تتبع عمل البرنامج الآتي واستنتج الناتج:

```
#include <iostream.h>
class square;
class rectangle {
private:
int width, height;
public:
int area ()
{return (width * height);}
void convert (square a);
};
class square {
private:
int side;
public:
void set_side (int a)
{side=a;}
friend class rectangle;
};
void rectangle::convert (square a) {
width = a.side;
height = a.side;
}
int main () {
square sqr;
rectangle rect;
```

```
sqr.set_side(4);
rect.convert(sqr);
cout << rect.area();
return 0;
}
```

2-تتبع عمل البرنامج الآتي واكتب ناتج البرنامج:

```
#include<iostream.h>
#include<string.h>
class twovalues{
  int a,b;
public:
  twovalues(int i,int j){a=i;b=j;}
  friend  class Min;
};
class Min
{public :
 int min(twovalues x);
 };
 int Min:: min(twovalues x)
{
return x.a<x.b?x.a:x.b;
 }
 int main()
 {  twovalues ob(10,20);
  Min m;
  cout<<m.min(ob);
 return 0;}
```

3- اشرح عمل البرنامج الآتي واكتب ناتج البرنامج:

```
#include<iostream.h>
class shared {
static int a;
int b;
public:
void set(int i,int j) { a=i; b=j;}
void show( );
};
int shared :: a;
void shared :: show( )
{
cout <<" This is static a:  "<< a;
cout<<"\nThis is non_static b:  " <<  b;
cout << "\n";
}
int main( )
{
shared x, y;
x.set(1, 1);
x.show( );
y.set(2, 2);
y.show( );
x.show( );
return 0;
}
```

الفصل السابع

الوراثة

7-1- تمهيد :

تعتمد فكرة الوراثة على إمكانية إنشاء أصناف جديدة من أصناف موجودة مسبقا .تأخذ الأصناف الجديدة صفات وتصرفات الأصناف البرمجية القديمة وتضيف إليها إمكانيات جديدة تحتاجها.
تسمى الأصناف الموجودة مسبقا بالأصناف الأساسية base classes وتسمى الأصناف المشتقة عنها بالأصناف المشتقة derived classes . في حالة الوراثة الوحيدة ، يجري اشتقاق صنف من صنف أساس وحيد.

تدعم لغة c++ الوراثة المتعددة التي يقوم فيها الصنف المشتق بالوراثة من أكثر من صنف أساس.وتعتبر الوراثة الوحيدة مباشرة وغير معقدة ،أما الوراثة المتعددة فيمكن أن تكون معقدة ويمكن أن تؤدي إلى أخطاء.

عادة يضيف الصنف المشتق بيانات وأعضاء إجراءات خاصة به وعليه يكون الصنف المشتق أكبر من الصنف الأساس.

نلحظ أن كل كائن تابع للصنف المشتق هو بالضرورة تابع للصنف المشتق منه ولكن العكس غير صحيح فكائنات الأصناف المشتقة تحمل صفات أكثر من كائنات الصنف الأساس ، فمثلا صنف المستطيل هو صنف مشتق من صنف الأشكال الرباعية وعليه يمكن القول إن أي مستطيل هو شكل رباعي ولا يمكننا القول إن أي شكل رباعي هو مستطيل.

ملاحظة: عندما يقوم صنف بوراثة صنف آخر فإن أعضاء الصنف الأساس تصبح أعضاء في الصنف المشتق .

7-2- التصريح عن الصنف المشتق:

يتم التصريح عن الصنف المشتق على الشكل الآتي:

```
class derived-class-name : access base-class-name
```

{

جسم الصنف المشتق

};

تحدد access و تسمى محدد وصول إمكانية الوصول إلى أعضاء الصنف الأساس وهى يمكن أن تكون إما public أو private أو protected وإذا لم يتم تحديدها فسيفترض المصرف أن محدد الوصول هو private .

عندما يستخدم محدد الوصول public تسمى الوراثة وراثة عامة، وعندما يستخدم المحدد private تسمى الوراثة وراثةخاصة وعندما يستخدم محدد الوصول protected تسمى الوراثة وراثة محمية.

فإذا كانت الوراثة وراثة عامة ففيها تتم وراثة الأعضاء العامة والمحمية في الصنف الأساس كأعضاء عامة ومحمية في الصنف المشتق ولكن في كل الأحوال الأعضاء الخاصة في الصنف الأساس تبقى خاصة بالصنف الأساس ولا يمكن الوصول إليها من أعضاء الصنف المشتق.

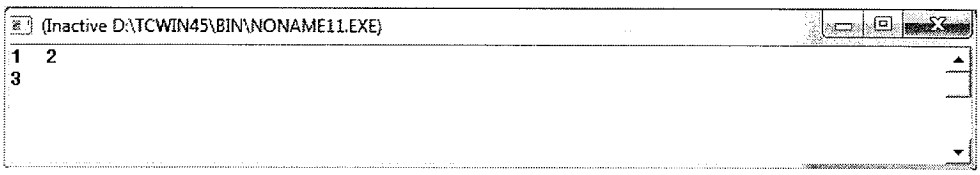
في البرنامج الآتي يتضح لنا أن الكائنات التابعة للصنف المشتق يمكنها الوصول إلى الأعضاء العامة في الصنف الأساس إذا كانت الوراثة عامة.

البرنامج الآتي يوضح ذلك.

```
//program 7-1
  #include <iostream.h>
  class base {
  int i , j;
  public:
  void set( int a , int b) { i= a; j= b; }
  void show( ) { cout<<i << " " << j << "\n"; }
  };
  class derived : public base {
  int k;
  public:
```

```
derived (int x)  { k=x;  }
void showk( ) { cout << k << "\n" ;   }
};
int main( )
{
derived ob(3);
ob.set(1 ,2);
ob.show( );
ob.showk( );
return 0;
}
```

وتظهر شاشة النتائج كما يأتي:



شرح البرنامج

في هذا البرنامج: تم الإعلان في البداية عن صنف أساس class base ويتضمن إجرائين: الأول void set(int a ,int b) يستقبل هذا الأجراء عددين يعطيان لكل من a و b .

والإجراء الثاني void show( ) .

ثم أعلنا عن صنف مشتق class derived : public base وهذا الصنف يرث الصنف الأساس وراثة عامة ،ثم أعلنا عن كائن للصنف المشتق derived ob(3) وفي هذه الحالة يأخذ المتغير x القيمة 3 ومن ثم تعطى القيمة 3 للمتغير k وبما أن الوراثة عامة فإن الكائن ob يستطيع الوصول إلى أعضاء الإجراءات العامة set( ) و show( ) في الصنف base ، بعد ذلك استخدمنا التعليمة ob.set(1,2) والتي تعطي القيمتين1 و2 للمتغيرين a و b الموجودين في الإجراء void set وفي هذا الإجراء تصبح قيمة i=1 و j=2 .

ومن ثم استخدمنا التعليمة ob.show() فيتم طباعة قيمة المتغيرين i و j أي يكون ناتج الطباعة 2 1 ، ثم استخدمنا التعليمة ob.showk() والتي تقوم بطباعة قيمة المتغير k والمساوية للعدد 3.

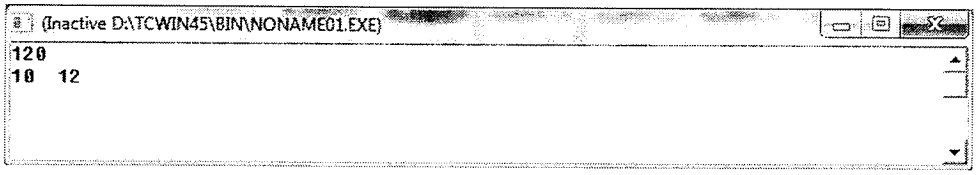
7-3- الوراثة المحمية

إذا كان محدد الوصول محمى (protected) تسمى الوراثة محمية وعندها كل الأعضاء العامة والمحمية في الصنف الأساس تصبح أعضاء محمية في الصنف المشتق، أي يمكن الوصول إليها من الكائنات في الصنف المشتق، البرنامج الآتي يوضح ذلك:

```
//program 7-2
  #include <iostream.h>
  class base {
  protected:
  int i ,j ;
  public:
  void setij( int a , int b) { i= a; j= b;  }
  void showij( ) { cout<<i <<  "  " << j << "\n"; }
  };
  class derived : protected base {
  int k;
  public:
  void setk( ) { setij( 10, 12) ; k = i*j;  }
  void showall( ) { cout << k<< " "<<endl ; showij( ) ; }
  };
  int main ( )
  {
  derived ob ;
```

```
ob.setk( ) ;
 ob.showall( );
return 0 ;
}
```

وتظهر شاشة النتائج كما يأتي:



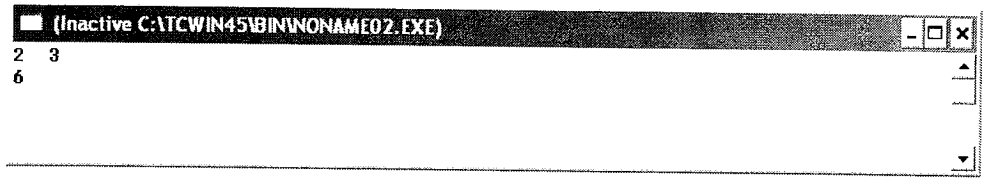
7-4- الوراثة والأعضاء المحمية:

عندما يتم الإعلان عن عضو في صنف ما على أنه محمى protected لا يمكن الوصول إلى هذا العضو من الأعضاء خارج الصنف تماماً كالعضو الخاص private ولكن هنالك استثناء هام ، ففي الوراثة العامة في حين أن العضو الخاص لا يمكن الوصول إليه حتى من الأعضاء في الصنف المشتق، يمكن الوصول إلى العضو المحمى في الصنف الأساس من قبل الأعضاء في الصنف المشتق. وعليه باستخدام محدد الوصول protected يمكنك تعريف أعضاء خاصة بالصنف يمكن الوصول إليها من الكائنات في الأصناف المشتقة وإليك البرنامج الذي يوضح ذلك:

```
//program 7-3
#include<iostream.h>
class base
 { protected :
int i,j;
 public :
void set(int a,int b){i=a;j=b;}
void show(){cout<<i<<" "<<j<<endl;}
```

```
};
class derived :public base
  {
  int k;
public :
void setk(){k=i*j;}
void showk()
{ cout<<k<<endl;}
};
int main()
{
  derived ob;
  ob.set(2,3);
  ob.show();
  ob.setk();
  ob.showk();
  return 0;
}
```

وتظهر النتائج كما يأتي:



في هذا المثال تمت وراثة الصنف derived من الصنف base وراثة عامة و تم الإعلان عن البيانات i و j على أنها محمية عضو الإجراء ( )setk في الصنف derived ولذلك يمكن لعضو الإجراء الوصول إلى هذه البيانات .

ملاحظة: لو كان المتغيران i,j محدد الوصول اليهما من private فإن الصنف المشتق لايتمكن من الوصول إليهما.

7-5- الصنف المشتق من صنف آخر مشتق:

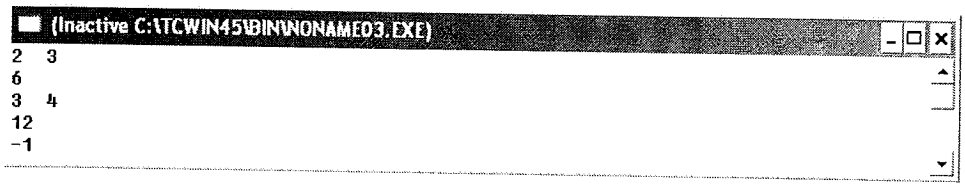
لدى استخدام الصنف المشتق بوصفه صنفا" أساسا" من صنف مشتق آخر فإن إي أعضاء محمية في الصنف الأساس الأول التي تمت وراثتها من قبل الصنف المشتق الأول وراثة عامة(محدد الوصول public) يمكن أن تتم وراثتها بوصفها أعضاء محمية ضمن الصنف المشتق الثاني.

المثال االآتي يوضح ذلك:

```
 //program 7-4
#include<iostream.h>
class base
  { protected :
 int i,j;
  public :
 void set(int a,int b){i=a;j=b;}
 void show(){cout<<i<<"  "<<j<<endl;}
};
class derived1 :public base
  {
  int k;
public :
void setk(){k=i*j;}
void showk()
{ cout<<k<<endl;}
};
class derived2 :public derived1
```

```
{ int m;
public :
void setm(){ m=i-j;}
void showm(){cout<<m<<endl;}
};
int main()
{
  derived1 ob1;
  derived2 ob2;
  ob1.set(2,3);
  ob1.show();
  ob1.setk();
  ob1.showk();
ob2.set(3,4);
  ob2.show();
ob2.setk();
ob2.setm();
  ob2.showk();
  ob2.showm();
  return 0;
  }
```

وتظهر النتائج كما يأتي:



7-6-الوراثة المتعددة:

تسمح لغة C++ لصنف ما أن يرث أكثر من صنف ، وهذا النوع من التوارث يتحقق بطريقتين:

1-الصنف الوارث يمكن أن يُستخدم بوصفه صنفا" موروثا" من صنف وارث آخر .

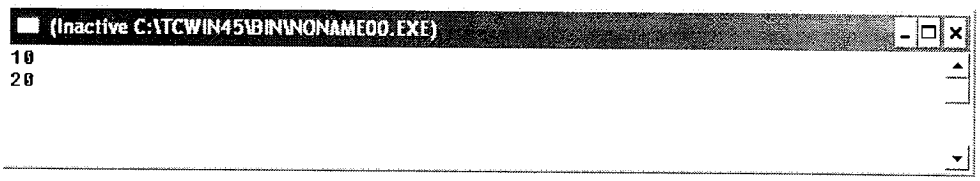
2-الصنف الوارث يمكن أن يرث أكثر من صنف موروث.

ومن الممكن أن يرث صنفا" مشتقا" صنفا" أساسا" أو أكثر وهذا مانوضحه بالمثال الآتي:

```
 //program 7-5
#include <iostream.h>
class base1
  { protected :
int x;
  public :
 void showx(){cout<<x<<endl;}
};
 class base2
  { protected :
int y;
  public :
 void showy(){cout<<y<<endl;}
};
class derived :public base1,public base2
  {
public :
void set(int l,int j){x=l;y=j;}
};
int main()
{
```

```
derived ob;
ob.set(10,20);
ob.showx();
ob.showy();
  return 0;
}
```

وتظهر شاشة الاخراج كما يأتي:



7-7-إجراءات البناء والهدم والوراثة:

من المهم أن نعرف ترتيب تنفيذ إجراءات البناء والهدم عند إنشاء كائن تابع للصنف المشتق .

ملحوظات:

1-إجراء بناء الصنف الموروث تنفذ قبل إجراء الصنف الوارث.

2- إجراء هدم الصنف الوارث تنفذ قبل إجراء هدم الصنف الموروث.

3- لا يمكن توريث إجراءات البناء والهدم من الصنف الأساس إلى الصنف المشتق.

4- يتم إرسال المعاملات لإجراء بناء الصنف الموروث من خلال إجراء بناء الصنف الوارث باستخدام مهيء العضو. (:)

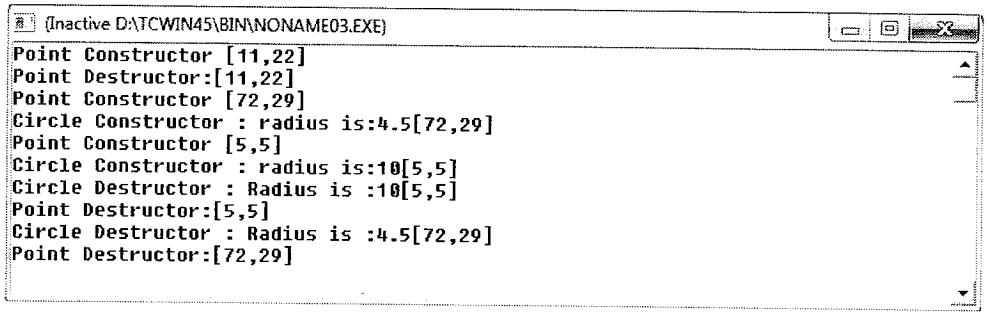
مثال : يوضح هذا البرنامج كيفية استدعاء إجراءات البناء والهدم للصنف الأساس والصنف المشتق.

```
//program 7-6
#include <iostream.h>
class point
```

```
{
public:
point ( int =0,int =0);
~point();
protected:
int x,y;
};
point::point(int a,int b)
{
x=a;
y=b;
cout<<"Point Constructor ["<<x<<','<<y<<']'<<endl;
}
point::~point()
{
cout<<"Point Destructor:["<<x<<','<<y<<']'<<endl;

}
class circle:public point
{
public:
circle(double r=0,int x=0,int y=0);
~circle();
private:
double radius;
};
circle::circle(double r,int a,int b):point(a,b)
```

```
{
radius=r;
cout<<"Circle Constructor : radius is:"<<radius<<'['
<<x<<','<<y<<']'<<endl;
}
circle::~circle()
{
cout<<"Circle Destructor : Radius is :"<<radius
<<'['<<x<<','<<y<<']'<<endl;
}
int main()
{
 {
  point p(11,22);
 }
circle circle1(4.5,72,29);
circle circle2(10,5,5);
return 0;
}
```

وتظهر شاشة النتائج كما يأتي:



شرح البرنامج:

كما يظهر في هذا المثال لدينا إجراء البناء Point والصنف circle الذي يرث من الصنف Point وراثة عامة Public كما يظهر في الشكل الآتي:

class circle:public point

الأهم في هذا الكود هو أولوية استدعاء الإجراء ، في الإجراء الرئيسي ()main لدينا الكود الآتي:

```
{{
  point p(11,22);
 }
circle circle1(4.5,72,29);
circle circle2(10,5,5);
return 0;
}
```

أول شيء لدينا الكائن P في مجال معين(بين أقواس)

```
{
  point p(11,22);
 }
```

في هذه الحالة يقوم باستدعاء إجراء البناء point ومن ثم إجراء الهدم مباشرة قبل الخروج من هذا المجال بحيث تكون المخرجات بالشكل الآتي.:

Point Constructor [11,22]

Point Destructor:[11,22]

بعد ذلك لدينا اشتقاق للكائن circle1 ، وكما ذكرنا سابقاً في الملحوظات (ملحوظة رقم 1)، حيث يتم في البداية تنفيذإجراء البناء point أولاً ومن ثم إجراءالبناء لـ circle ، وكذا الحال مع الكائنcircle2 بحيث تكون المخرجات بالشكل الآتي.:

Point Constructor [72,29]

Circle Constructor : radius is:4.5[72,29]

Point Constructor [5,5]

Circle Constructor : radius is:10[5,5]

نلاحظ هنا يتم أولاً تنفيذ إجراءات البناء كلها قبل تنفيذ أي إجراء هدم قبل الخروج من المجال المحدد.
بعد ذلك يتم استدعاء إجراء هدم الصنف الوارث قبل الصنف الموروث بترتيب معاكس .

1- ضع الكلمة المناسبة في الفراغات الآتية:

أ) يمكن الوصول إلى الأعضاء -------التابعة للصنف الأساس من خلال التعريفات الواردة في الصنف الأساس والأصناف المشتقة عنه فقط.

ب)-حسب مفهوم الوراثة الوحيدة يمكن لصنف أن يوجد ضمن علاقة ------مثله مثل بقية الأصناف المشتقة عنه.

ت) -يمكن الوصول إلى الأعضاء -------لصنف أساس من إي مكان في البرنامج يتضمن تعاملا مع غرض من أغراضه أو أغراض تابعة للأصناف المشتقه.

ث)- توفر لغة c++ --------السماح لصنف مشتق بالوراثة من عدة أصناف أساسية حتى ولو كانت هذه الأصناف غير مرتبطة ببعضها البعض.

ج)- عند اشتقاق صنف من صنف أساس باستخدام الوراثة العامة ،تصبح الأعضاء العامة المرتبطة بالصنف الأساس أعضاء ------ضمن الصنف المشتق وتصبح الأعضاء المحمية المرتبطة بالصنف الأساس أعضاء ----ضمن الصنف المشتق.

الجواب:

أ) المحمية protected

ب) الهرمية

ت) Public

ث) الوراثة المتعددة

ج) Public و protected

2- وضح فيما إذا كانت الحقائق التالية صحيحة أم لا

أ)-من الممكن معاملة أغراض الصنف الأساس وأغراض الصنف المشتق بالطريقة نفسها والأسباب نفسها.

ب)-لاتجري وراثة الإجراءات البناءة المرتبطة بالصنف الأساس من قبل الأصناف المشتقه

ت)- تشجع الوراثة على إعادة استخدام البرمجيات عالية المستوى

الأجوبة: أ صح - ب صح - ت صح

تمارين 7

1-تتبع عمل البرنامج الآتي واستنتج ناتج البرنامج:

```
#include<iostream.h>
class parent
{  private:
  float a;
  public :
 void pget()
 { cout<<"enter the float number\n";
cin>>a;
}
void  pdisplay()
{ cout<<"a="<<a<<endl;
}};
class child : public parent
{
private:
 int b;
 public:
void cget(){
pget();
cout<<"enter integer value\n";
cin>>b ;}
void cdisplay()
{
 pdisplay();
```

```
 cout<<"b="<<b<<endl;}
 };
 void main()
 {
child ch;
ch.cget();
ch.cdisplay();
}
```

2-تتبع عمل البرنامج الآتي واستنتج ناتج البرنامج:

```
#include <iostream.h>
 class B
{
int i;
public:
void set_i(int);
int get_i();
};
class D:public B
{
int j;
public:
void set_j(int);
int multi();
};
void B::set_i(int n)
{i=n;}
int B::get_i()
```

```
{
return i;
}
void D::set_j(int m)
{
j=m;}
int D::multi()
{
return get_i()*j;
}
int main()
{
int a,b;
cout<<"Enter A,B \n";
cin>>a>>b;
D ob;
ob.set_i(a);
ob.set_j(b);
cout<<"Multi ="<<ob.multi()<<endl;
        return 0;
}
```

3-تتبع عمل البرنامج الآتي واستنتج ناتج البرنامج:

```
#include <iostream.h>
class base
{ public :
base(){cout<<"constructing base\n";}
~base(){cout<<"destructing base\n";}
```

```
};
class derived :public base
{ public:
 derived(){cout<<"constructing derived\n";}
  ~derived(){ cout<<"destructing derived\n";}
  };
  int main()
  {
  derived ob;
  return 0;}
```

4-تتبع عمل البرنامج الآتي واستنتج ناتج البرنامج:

```
#include <iostream.h>
class B1
{
int a;
public:
B1(int x)
{
a=x;
}
int get_a()
{return a;
}
};
class D1:public B1
{
int b;
public:
D1(int x,int y):B1(y)
```

```
{
b=x;
}
int get_b()
{
return b;
}
};
class D2:public D1
{
int c;
public:
D2(int x,int y,int z):D1(y,z)
c=x;
}
void show()
{
cout<<get_a()<<"\n"<<get_b()<<"\n"<<c<<endl;
}};
int main()
{
int a,b,c;
cout<<"Entera,b,c \n";
cin>>a>>b>>c;
D2 ob(a,b,c);
ob.show();
        return 0;
}
```

الفصل الثامن

مجاري الإدخال والإخراج في اللغة C++

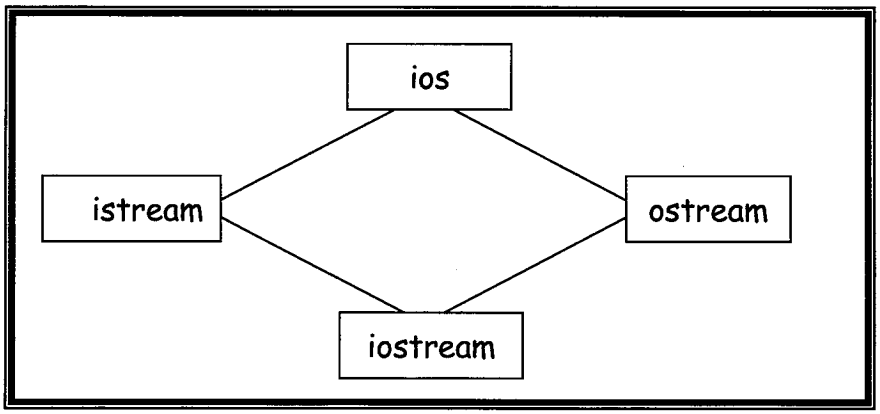
8-1-تمهيد:

يتوافر في لغة c++ عدد كبير من إمكانيات الإدخال والإخراج ؛ إذ تم عرض بعض هذه الأمكانيات فيما سبق ،وسوف نعرض في هذا الفصل معلومات إضافية حول أمكانية (الإدخال/الإخراج) المتاحة ضمن لغة C++ .

8-2- المجاري:

تتم عملية (الإدخال/الإخراج) في لغة c++ من خلال مجاري من المعطيات وهي سلسلة من البايتات(bytes) في أثناء عملية الدخل تسير هذه السلسلة من وحدات الإدخال إلى الذاكرة الرئيسية،أما في أثناء عملية الإخراج فإنها تأخذ مسارا عكسيا من الذاكرة الرئيسية إلى وحدات الإخراج.

ويتم تمثيل مجاري(الإدخال/الإخراج) بكائن تابع لصنف معين ، فمثلاً رأينا في جميع الأمثلة السابقة كائنات المجاري cin, cout والتي استعملناها لعمليات الدخل والخرج كانت تابعة للصنف iostream ويمكننا توضيح هذه الأصناف في المخطط الآتي:



كما نرى من المخطط الصنف ios هو الصنف الأساس لهرمية مجاري الدخل والخرج وهى تحتوى على العديد من الثوابت وأعضاء الإجراءات المشتركة بين مختلف الأنواع من أصناف الدخل والخرج. الصنفين istream و ostream مشتقات من الصنف ios وهما متخصصتان بأعمال الدخل والخرج .يحتوي الصنف istream على أعضاء إجراءات لـ ( )get , ( )getline وعامل الدخل(>>) بينمايحتوي الصنف ostream على ( )put و ( )write وعامل الخرج (<<).

يحتوى الصنف ios على غالبية الميزات التي تحتاج إليها ومن أهم هذه الميزات أعلام التنسيق.

8-2-1-أعلام التنسيق Format state flags

هي مجموعة من الأعضاء في الصنف ios تعمل لتحديد خيارات في عمل وتنسيق الدخل والخرج.

هناك عدة طرق لضبط أعلام التنسيق ، وبما أن الأعلام هي أعضاء في الصنف ios يجب عادة وضع اسم الصنف ios وعامل دقة المدى قبلها . يمكن ضبط الأعلام عنها باستعمال الأعضاء الدالية ( )setf و( )unsetf التابعة للفئة ios:-

نبين في الجدول الآتي بعضا لأعلام تنسيق الفئة ios:-

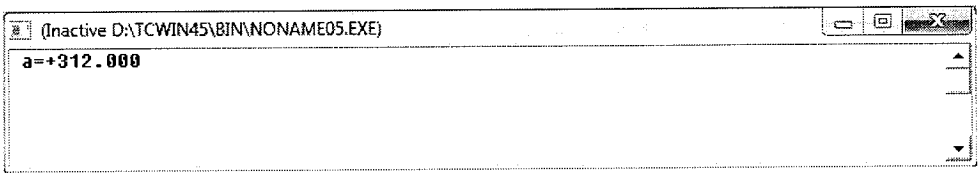
العلم	معناه
skipws	تجاهل المسافات البيضاء الموجودة في الدخل
left	محاذاة الخرج إلى اليسار
right	محاذاة الخرج إلى اليمين
dec	تحويل إلى عشري
showbase	استعمال مؤشر الأساس في الخرج

showpoint	إظهار النقطة العشرية في الخرج
uppercase	استعمال الأحرف الكبيرة في الخرج
showpos	عرض (+) قبل الأعداد الصحيحة الموجبة

البرنامج الآتي يوضح كيفية استعمال علمي التنسيق showpos و showpoint:

```
//program 8-1
#include <iostream.h>
int main( )
{ float a;
cout.setf(ios:: showpoint);
cout.setf(ios:: showpos);
 a=312;
cout<<" a="<<a;
return 0 ;
}
```

وتظهر شاشة النتائج كما يأتي:



8-3-المناورات Manipulators

المناورات هي تعليمات تنسيق تدرج في مجاري الإخراج مباشرة ، رأينا منها حتى الآن المناور endl . والذي يرسل سطراً جديداً إلى مجاري الإخراج.

يوجد نوعان من المناورات ، نوع يأخذ وسيط والآخر لا يأخذ أي وسيط، الجدول الآتي يوضح بعض المناورات التي لا تأخذ أية وسائط:

بعض المناورات التي لا تأخذ أية وسائط

المناور	هدفه
ws	تنشيط ميزة تخطى المسافات البيضاء الموجودة في الداخل
dec	التحويل إلى عشري
oct	التحويل إلى ثماني
hex	التحويل إلى ست عشري
endl	إدراج سطر جديد
ends	إدراج حرف خامد لإنهاء سلسلة خرج

تدرج هذه المناورات في المجاري مباشرة ، فمثلاً لخرج المتغير var في التنسيق الست عشرى نكتب:

```
cout<<hex<<var;
```

إن الحالة التي تضبطها المناورات ليس لها وسائط تبقى نشطة إلى أن يتم تدمير المجرى وعليه يمكننا خرج عدة أرقام في التنسيق الست عشرى من خلال إدراج مناور hex واحد فقط.

الجدول الآتي يلخص بعض المناورات التي تأخذ وسائط ونحتاج إلى إدراج ملف الترويسة iomanip.h لكي نستعمل هذه المناورات:

المناور	الوسيطة	هدفه
setw()	عرض الحقل (int)	ضبط عرض الحقل المطلوب عرضه
setfill()	حرف الحشو (int)	ضبط حرف الحشو في الخرج(الحرف الافتراضي هو المسافة

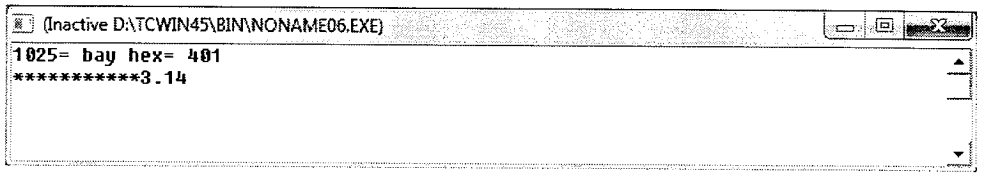
setprecision()	الدقة (int)	ضبط الدقة(كمية الأرقام المعروضة)
set iosflags()	أعلام تنسيق (long)	ضبط الأعلام المحددة
Resetiosflags()	أعلام تنسيق (long)	مسح الأعلام المحددة

إن المناورات التي تأخذ وسيطات تؤثر فقط في البند التالي في المجاري فمثلاً إذا استعملنا المناور ( )setw لضبط عرض الحقل الذي يتم إظهار رقم ما فيه سنحتاج إلى استعماله مجدداً مع الرقم التالي.

المثال الآتي يستعمل بعض هذه المناورات :

```
//program 8-2
#include <iostream.h>
#include <iomanip.h>
int main( )
{ int a;
 float b;
 a=1025;
 b=3.14;
cout <<a<<"= bay hex= "<< hex << a << endl;
cout << setfill('*') << setw(15) << b;
return 0;
}
```

وتظهر شاشة النتائج كما يأتي:



8-4-إجراءات مجاري الإدخال والإخراج

يحتوى الصنف ios على عدد من الإجراءات التي يمكن استخدامها لضبط أعلام التنسيق وتنفيذ مهام أخرى . الجدول الآتي يبين معظم هذه الإجراءات .

الإجراء	هدفه
ch=fill( );	إعادة حرف الحشو(الفراغ هو الافتراضي)
fill(ch);	ضبط حرف الحشو
p=precision( );	الحصول على الدقة
precision(p);	ضبط الدقة
w=width( );	الحصول على عرض الحقل التالي
setf(flags);	ضبط أعلام التنسيق المحددة
unsetf (flags);	إلغاء ضبط أعلام التنسيق المحددة
setf(flags,field);	مسح الحقل أولاً ثم ضبط الأعلام

يتم استدعاء هذه الإجراء بوساطة كائنات المجاري باستعمال عامل النقطة ،فمثلاً لضبط عرض الحقل عند 5 يمكننا كتابة :

```
cout.Width(5);
```

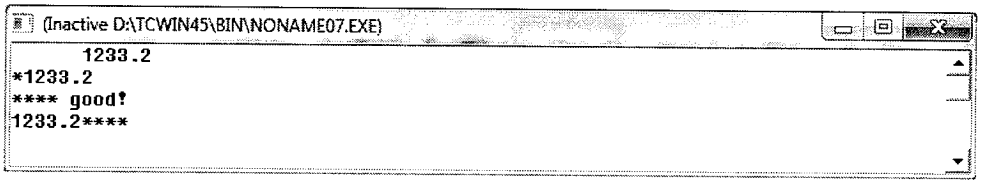
أيضاً تضبط العبارة الآتية حرف الحشو عند * :: cout.fill('*');

البرنامج الآتي يستخدم الإجراءات ( )width و( )precision و( )fill .

```
//program 8-3
#include <iostream.h>
#include <iomanip.h>
int main( )
```

```
{ float a;
a=1233.25;
cout.precision (5) ;
cout.width(12);
cout<< a <<"\n" ;
cout<<setfill('*');
cout.width(7);
cout<< a <<"\n" ;
cout.width(10);
cout<< " good!" <<"\n" ;
cout.width(10);
cout.setf(ios::left);
cout<< a  ;
return 0;
}
```

وتظهر شاشة النتائج كما يأتي:



8-5-الصنف istream

ينفذ الصنف istream المشتق من الصنف ios نشاطات خاصة بالدخل ونشاطات إضافية. الجدول الآتي يوضح بعض إجراءات الصنف istream .

الصنف	هدفه
>>	إدخال منسق لكل الأنواع الأساسية والمحملة بشكل زائد
get(ch)	إدخال حرف واحد
get(str)	إدخال أحرف إلى مصفوفة وصولاً إلى '\o'
get(str,max)	إدخال حتى max أحرف إلى المصفوفة
peek(ch)	قراءة حرف واحد وتركه في الدفق
putpack(ch)	إعادة إدراج الحرف الأخير المقروء في دفق الدخل
count=gcount	إعادة عدد الأحرف التي قرأها استدعاء الدالة ()get و ()getline

لقد رأينا حتى الآن بعضاً من هذه الإجراءات كـ ( )get مثلاً . معظمها يعمل على الكائن cin الحقيقي يمثل لوحة المفاتيح.

8-6-الصنف ostream

يعالج ostream نشاطات الخرج، يبين الجدول الآتي الإجراءات الذي يستعمله هذه الصنف:-

الإجراء	هدفه
<<	إخراج منسق لكل الأنواع الأساسية والمحملة بشكل زائد
put(ch)	إخراج الحرف ch في الدفق
flush( )	مسح محتويات الدارئ (Buffer) وإدراج سطر جديد
write(str,size)	إخراج size أحرف من المصفوفة str

لقد استعملنا حتى الآن كائني المجاري cin و cout . يرتبط هذان الكائنان عادة بلوحة المفاتيح والشاشة على التوالي . هناك كائنان آخران هما cerr و clog .

يتم –غالباً– ما يتم استعمال الكائن cerr لرسائل الخطأ. الخرج المرسل إلى cerr يتم عرضه فوراً ولا يمكن تغيير وجهته لذا ترى رسالة الخرج من cerr في حال تعطل البرنامج كلياً.

يمكن أيضاً للمستخدم إنشاء مناورات تقوم بتنسيق خاص بالمستخدم .

الصورة العامة لإنشاء مناور خرج هي:-

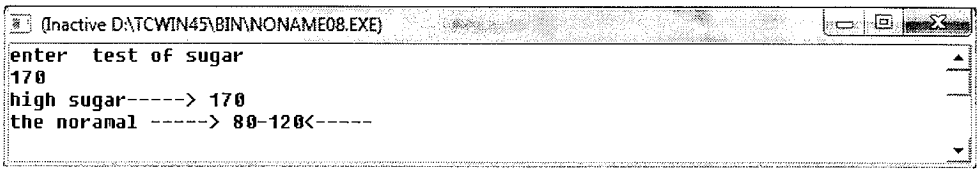
```
ostream & mani-name( ostream & stream)
{
//your code here
return stream;
}
```

المثال الآتي يقوم بإنشاء مناورين ( )la و( )ra يقومان بإخراج (→) و (←).

```
//program 8-4
#include <iostream.h>
#include <iomanip.h>
ostream &ra(ostream &stream)
{
stream << "-----> " ;
return stream;
}
ostream &la(ostream &stream)
{
stream << "<----- " ;
return stream;
}
int main( )
{ int a;
```

```
 cout<<"enter  test of sugar\n";
cin>>a;
if(a>120)  cout << "high sugar" <<ra<< a<<"\n";
if(a<80) cout << "low sugar" <<ra<< a<<"\n";
cout <<"the noramal "<<ra<<"80-120"<< la;
return 0;
}
```

وتظهر شاشة النتائج كما يأتي:



الصورة العامة لإنشاء مناور دخل هي:

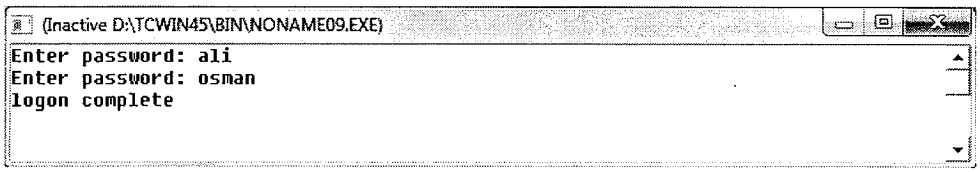
```
istream & mani-name(istream & stream)
{
//your code here
return stream;
}
```

المثال اللآتي يقوم بإنشاء مناور دخل ( )getpass والثاني يقوم بإخراج صوت جرس باستعمال تتابع الهروب 'a\' ويطلب من المستخدم إدخال password .

```
 //program 8-5
#include <iostream.h>
#include <string.h>
istream &getpass (istream &stream(
```

```
}
cout << '\a' ; // sound bell
cout << "Enter password;" :
return stream;
{
int main( )
{
char pw[80;[
do cin>> getpass >>pw;
        while (strcmp (pw, "osman"));
cout <<"logon complete\n";
return 0;
}
```

وتظهر شاشة النتائج كما يأتي:



7-8 تمارين محلولة:

1- ضع برنامج لطباعة الأعداد الآتية :

a=123.45

b=1.2

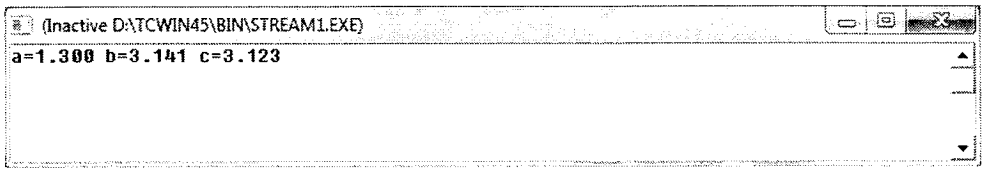
c=1.2345

بوجود دقة ثلاثة أرقام بعد الفاصلة.

البرنامج:

```
#include<iostream.h>
#include <iomanip.h>
main()
{
 float a,b,c;
 a=123.45;
b=1.2;
c=1.2345;

 a=1.3;b=3.1415;c=3.123456;
        cout.setf(ios::fixed);
cout.precision(3);
         cout<<"a="<<a<<"\t"<<"b="<<b<<"\t"<<"c="<<c;
         }
```

وتظهر شاشة تنفيذ البرنامج النتائج كما يأتي:

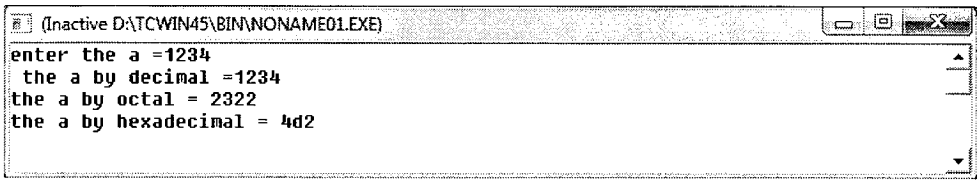


2-ضع برنامجا" لإدخال عدد صحيح من النظام العشري ثم أطبع هذا العدد بالنظام العشري وبالنظام الثماني والنظام الست عشري.

البرنامج:

```
#include<iostream.h>
main()
{
int a;
   cout<<"enter the a =";
cin>>a;
cout<<" the a by decimal ="<<a<<endl;;
cout<<"the a by octal = "<<oct<<a<<endl;
  cout<<"the a by hexadecimal = "<<hex<<a<<endl;;
}
```

وتظهر شاشة النتائج كما يأتي:

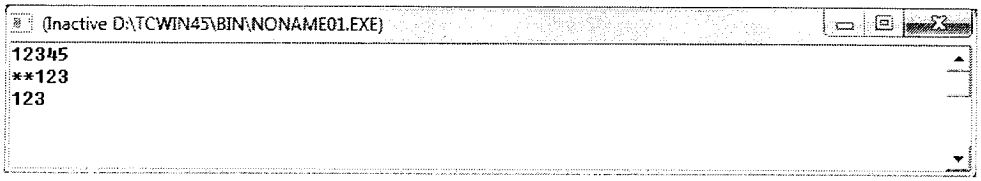


3-بين خرج كلا من التعليمات الآتية:

أ-

```
cout<<"12345"<<endl;
cout.width(5);
cout.fill('*');
cout<<123<<endl<<123;
```

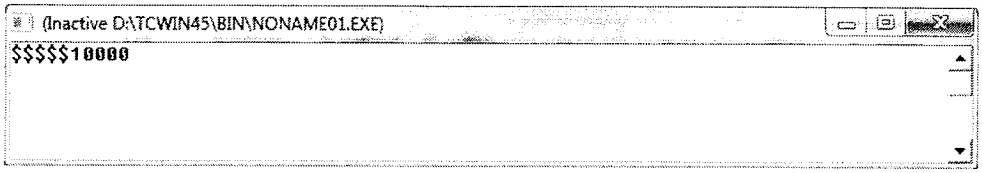
الخرج يكون في شاشة التنفيذ الآتية:



ب-

```
cout<<setw(10)<<setfill('$')<<10000;
```

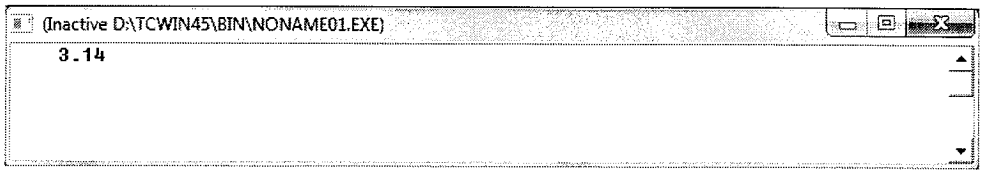
الخرج :



ج-

```
cout<<setw(8)<<setprecision(3)<<3.14256;
```

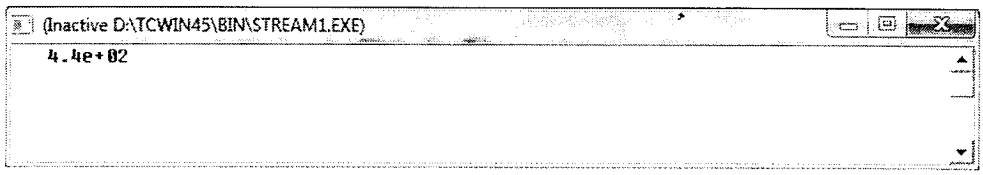
الخرج:



د-

```
cout<<setw(10)<<setprecision(2) <<444.1234;
```

الخرج:



4-تتبع عمل البرنامج الآتي وأكتب ناتج البرنامج:

```
#include<iostream.h>
#include<iomanip.h>
main()
{
        int x=10000;
        cout<<"x="<<x<<endl;
        cout<<setw(10)<<x<<endl;
        cout<<setw(10)<<setfill('*')<<x<<endl;
        cout.fill('$');
        cout<<setw(10)<<dec<<x<<endl;
        cout<<setw(10)<<setfill('#')<<hex<<x;
        }
```

الشرح:

في البداية أسندنا إلى المتغير x العدد 10000 ثم طبعنا المتغير x وقيمته فيظر على الشاشة:

x=10000

ثم ينتقل المؤشر إلى سطر جديد ثم طبعنا قيمة المتغير x في حقل عرضه عشر خانات فيظهر على الشاشة:

10000

ثم انتقلنا إلى سطر جديد لتطبع قيمة المتغير x في عشر خانات وتملأ الخانات الفارغة بالإشارة *؛إذ يظهر على الشاشة:

*****10000

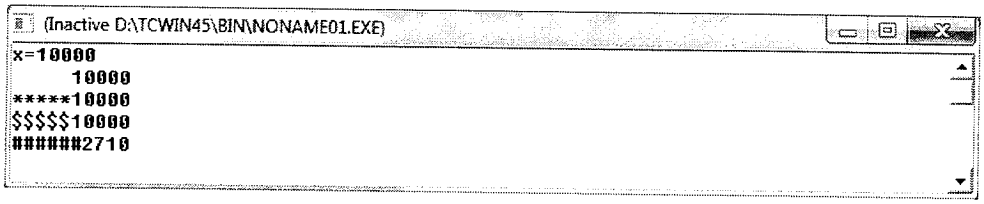
ثم انتقلنا إلى سطر جديد واستخدمنا في البرنامج التعليمة ;('$')cout.fill لكي يتم تعبئة الخانات الفارغة بالإشارة $ وبالتالي يتم طباعة قيمة المتغير x بالنظام العشري في حقل عرضه 10 خانات، وتملأ الخانات الفارغة بإشارة $ إذ يظهر على الشاشة:

$$$$$10000

ثم ننتقل إلى سطر جديد ونطبع قيمة المتغير x بالنظام الست عشري وبحقل عرضه عشر خانات والخانات الفارغة تملأ بالإشارة # إذ يظهر على الشاشة:

######2710

وإذا تم تنفيذ البرنامج تكون النتائج:



تمارين 8

1- قم بكتابة برنامج ينفذ الآتي:

- طباعة العدد الصحيح 70000 مع محاذاته على اليسار على أن يكون عرض الحقل 12.
- قراءة سلسلة وتخزينها في مصفوفة أحرف str.
- طباعة 700 بعلامة وبدون علامة.
- طباعة العدد 1000 بالنظام الست عشري .

2-اكتب برنامجا لطباعة القيمة 123.34567 بعد تقريبها إلى أقرب آحاد وعشرات ومئات وآلاف ومئات الآلاف.

3-اكتب برنامج لتحويل درجات الحرارة الفهرنهايت ذات القيم الصحيحة من 0 وحتى 212 درجة إلى قيم ذات فاصلة عائمة لدرجات الحرارة العادية وبدقة ثلاث خانات بعد الفاصلة.

4-اكتب برنامجا لطباعة القيمة الصحيحة 12345 والقيمة ذات الفاصلة العائمة 1.2345 ضمن حقول ذات أطوال مختلفة .

5-اكتب برنامجا لإدخال سلسلة حرفية من لوحة المفاتيح ولتحديد طول هذه السلسلة أطبع السلسلة ضمن حقل طوله مساوي لضعف طول السلسلة.

الفصل التاسع

معالجة الملفات

9-1-تمهيد:

تستخدم الملفات لتخزين كميات كبيرة من المعطيات بصورة دائمة ويقوم الحاسب بتخزين الملفات بدوره ضمن وحدات التخزين الثانوية مثل الأقراص المغناطيسية،الأقراص الضوئية والأشرطة الممغنطة.وتحمل هذه الملفات امتدادات مختلفة منها(.BIN , .dat., .TXT) . يُعدّ التعامل مع الملفات مثل فتحها أو الكتابة إليها من العمليات الأساسية التي تحتاجها معظم البرامج وعلى الرغم من أن لغة++c لاتوجد بها كلمات أساسية للتعامل مع الملفات ولكن الإجراءات التي تتعامل مع الملفات تقع ضمن مكتبة <iostream.h> و <fstream.h> .

9-2- الملفات ومجاري الإدخال والإخراج:

عند فتح ملف يتم إنشاء كائن يقترن معه مجاري الإدخال والإخراج. لقد رأينا من قبل أربعة كائنات منشأة أوتوماتيكيا، وهي cout، cin ، cerr و clog.

يستخدم الكائن cin لإدخال بيانات من لوحة المفاتيح، والكائن cout يستخدم لإخراج بيانات إلى الشاشة، والكائنان cerr و clog يستخدمان لإخراج رسائل الأخطاء إلى الشاشة.

عند التعامل مع الملفات، يجب تضمين ملفي الترويسة iostream.h و fstream.h إذ يحتوي الملف fstream.h على أصناف مجاري الإدخال والإخراج :{ifstream (يستخدم في إدخال بيانات إلى الملفات) و ofstream (يستخدم لإخراج بيات من الملفات)، و fstream (لإدخال وإخراج بيانات من الملفات)}.

لفتح ملف، نحتاج لإنشاء كائن يتبع لإحد هذه الأصناف.

9-3-كيف تفتح ملف نصي وتغلقه:

عندما ترغب في التعامل مع أحد الملفات سواء لتقرأ منه أم لتكتب إليه يجب أن تفتح هذا الملف أولا. وفتح الملف يعني أن تخبر نظام التشغيل عن اسم الملف الذي ترغب بالتعامل معه وعن الطريقة التي تستخدمه فيها .ولكي تفتح أحد الملفات أتبع الخطوات الآتية:

- نعرف كائن من نوع الملف باستخدام الكود ; ofstream fin
- نفتح الملف للكتابة باستخدام الكود

```
fin.open("file path",iostream family);
```

- نكتب إلى داخل الملف باستخدام الكود "fin<<" data

الخطوة الأولى: fin تعني اسم كائن من نوع ملف للتعامل معه داخل البرنامج وممكن أن يكون أي اسم نرغب فيه

الخطوة الثانية:file path هو اسم مسار الملف المراد خزنه فيه داخل الجهاز ويوضع بين علامتي تنصيص .

(iostream family) هي رموز تستخدم للدلالة على عمليات معينة .

والجدول الآتي يبين أهم هذه الر موز

جدول برموز iostream family	
الرمز	وظيفته
ios::app	يلحق الإدخال الجديد بنهاية الملف
ios::ate	يقوم بالقراءة أو الكتابة من نهاية الملف
ios::trunce	في حال وجود ملف فيقوم بحذف محتوياته
ios::in	فتح الملف للقراءة
ios::out	فتح الملف للكتابة
ios::binary	فتح الملفات على هيئة ثنائية ولست نصية

الخطوة الثالثة: ندخل البيانات إلى داخل الملف.

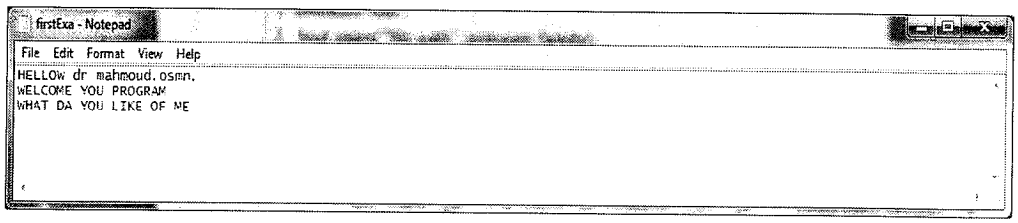
أما من أجل إغلاق الملف نستخدم الكود:;()fin.close

ضع برنامجا لتخزين ملف نصي على القرص D:

البرنامج:

```
//program 9-1
#include <fstream.h>
int main()
 { ofstream test;
 test.open("D:\\firstExa.txt");
 test << "HELLOW dr mahmoud.osmn.\n"
<< "WELCOME YOU PROGRAM\n"
<< "WHAT DA YOU LIKE OF ME\n";
 test.close();
}
```

وإذا رغبنا في مشاهدة محتويات الملف نفتح القرص D ثم نفتح الملف firstExa.txt وذلك بالنقر على اسم الملف مرتين، فتظهر المحتويات الآتية:



ملاحظة:

في حال تم فتح الملف المخزن على القرص D وتم إضافة بيانات جديدة عليه فإن المعلومات القديمة سوف يتم حذفها وحتى لايتم حذف محتويات الملف نستخدم الكود:

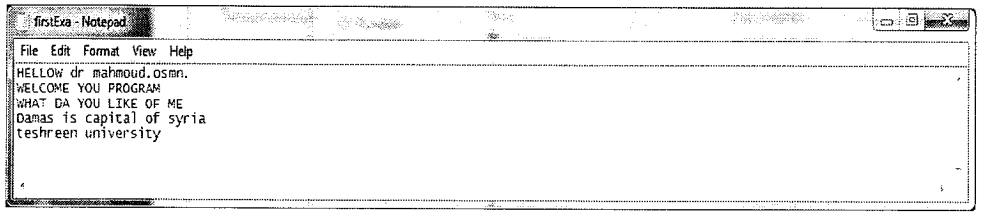
test.open("D:\\firstExa.txt",ios::app);

وفي هذه الحالة سوف يضيف المعلومات الجديدة في نهاية الملف

وعلى سبيل المثال لو كتبنا البرنامج السابق مع إضافة بعض العبارات النصية:

```
//program 9-2
#include <fstream.h>
int main()
 { ofstream test;
 test.open("D:\\firstExa.txt",ios::app);
 test << "Damas is capital of syria\n"
<< "teshreen university\n"
test.close();
}
```

وإذا ذهبت إلى القرص D وفتحت الملف firstExa.txt فيكون الملف الجديد على النحو الآتي:



ونلحظ إتمام إضافة المعلومات الجديدة إلى نهاية الملف .

9-4-القراءة من الملفات النصية:

للقراءة من الملفات النصية نتبع الخطوات الآتية:

1- نعرف كائن من نوع ملف باستخدام الكود ; ifstream fout

2- نفتح الملف للقراءة باستخدام الكود

fout.open("file path",iostream family);

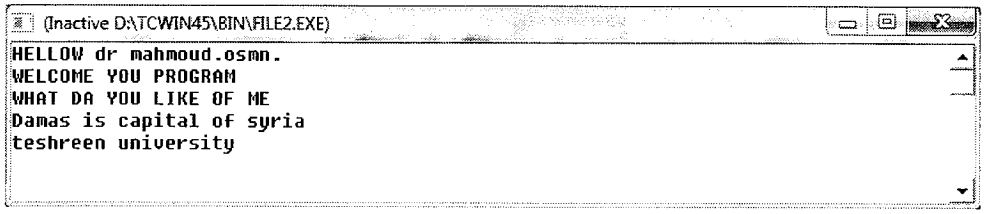
3- نقرأ من داخل الملف ونقوم بتخزين سطر واحد من البيانات من الملف إلى داخل المصفوفة .

مثال: ضع برنامجا" لقراءة محتويات الملف السابق D:\\firstExa.txt

البرنامج:

```
//program 9-3
#include <fstream.h>
int main()
 {ifstream test;
 char array[80];
 test.open("D:\\firstExa.txt",ios::out);
 while(!test.eof())
 {test.getline(array,80);
cout<<array<<endl};
 test.close();
}
```

وبعد تنفيذ البرنامج تظهر شاشة النتائج كما يأتي :



نلاحظ أننا استخدمنا حلقة while للتخزين في المصفوفة array وتم استخدام التعليمة eof() وهي اختصار للعبارة end of file وتستمر حلقة while بالعمل حتى تصل القراءة إلى نهاية الملف وقد استخدمنا من أجل ذلك الكود: while(!test.eof()) .

باستخدام أمر التفرع switch ضع برنامجا يتضمن الاختيارات الآتية:

1- إنشاء ملف

2- الإضافة إلى الملف اللذي فتحته

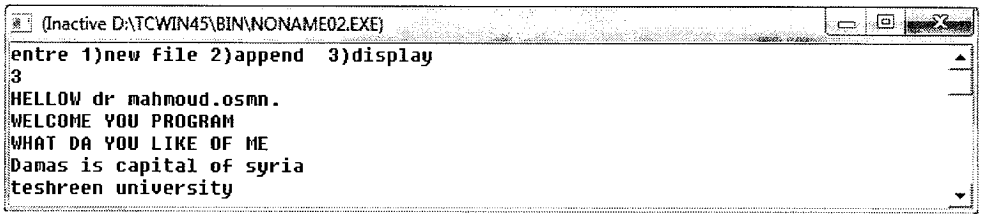
3- إظهار محتويات الملف السابق

البرنامج:

```
//program 9-4
#include<iostream.h>
#include <fstream.h>
int main()
 {  ofstream test1; ifstream test3; ofstream test2;
       int k;
cout<<"entre 1)new file 2)append  3)display\n";
cin>>k;
switch(k)
{
 case 1:
test1.open("D:\\firstExa.txt");
 test1 << "HELLOW dr mahmoud.osmn.\n"
<< "WELCOME YOU PROGRAM\n"
<< "WHAT DA YOU LIKE OF ME\n";
 test1.close();
break;
 case 2:
 test2.open("D:\\firstExa.txt",ios::app);
 test2 << "Damas is capital of syria\n"
<< "teshreen university\n";
test2.close();
break;
```

```
        case 3:
char array[80];
 test3.open("D:\\firstExa.txt",ios::out);
 while(!test3.eof())
 {test3.getline(array,80);
cout<<array<<endl;}
 test3.close();
break;
default :
cout<<"the number is not need\n";
}
}
```

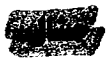
وبعد تنفيذ البرنامج تظهر لنا شاشة النتائج كما يأتي:



- في البرنامج السابق فتحنا أولا الملف وكتبنا فيه العبارات الموجودة في case 1 ومن جديد أخذنا رقم 2 وذلك لفتح الملف من جديد وأضفنا إليه البيانات الموجودة في case 2 ثم نفذنا البرنامج من جديد وأخذنا رقم 3 فظهرت النتائج كما هو أعلاه.

9-5-الكتابة إلى ملف سجل بسجل:

عند التعامل مع السجلات، يجب تضمين ملفي الترويسة iostream.h و fstream.h .



باستخدام السجلات ضع برنامجا لإنشاء ملف يحمل الاسم exam.txt والمخزن على القرص D والذي يتضمن أسماء ثلاثة طلاب مع درجاتهم الامتحانية.

البرنامج:

```
//program 9-5
#include<iostream.h>
#include <fstream.h>
main()
{
struct student
{
char name[20];
 int degre;
  };
 struct student st[3];
int i;
for(i=0;i<3;i++){
cout<<"\n enter the name: ";
 gets(st[i].name );
cout<<"enter degre: ";
 cin>>st[i].degre;}
ofstream fout;
 fout.open("D:\\exam.txt");
for(i=0;i<3;i++){
 fout << st[i].name;
fout << "\tthe degree: ";
fout << st[i].degre<<"\n";}
```

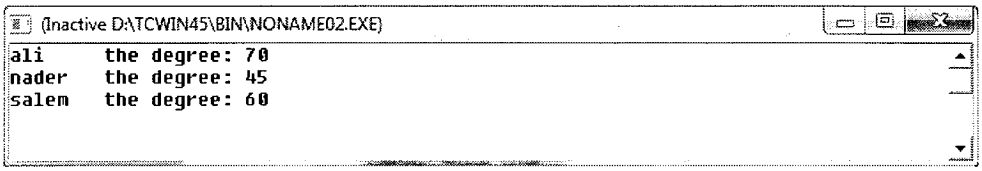
```
fout.close(); }
```

ضع برنامجا لإخراج محتويات الملف السابق:

البرنامج:

```
//program 9-6
#include <fstream.h>
int main()
 {ifstream fout;
 char array[300];
 fout.open("D:\\exam.txt",ios::out);
 while(!fout.eof())
 {fout.getline(array,300);
cout<<array<<endl;}
 fout.close();}
```

وبعد تنفيذ البرنامج تظهر لنا شاشة النتائج كما يأتي:



ضع برنامجا لإنشاء ملف يتضمن رقم الحساب واسم الزبون والرصيد حيث تنتهي عملية الإدخال بالضغط على ctrl مع z .

البرنامج:

```
//program 9-7
#include<iostream.h>
#include<fstream.h>
#include<stdlib.h>
```

```
main( )
{
  ofstream  outclientfile("D:\\clients.dat",ios::out);
  if (!outclientfile){
     cerr<<"File could not be opened"<<endl;
     exit (1);
}
   cout<<"Enter the account, name, and balance."
  <<endl
         <<"(Enter EOF to end input)"<<endl
         <<"? ";
  int account;
  char name[10];
//Continued
  float balance;
while(cin>>account>>name>>balance){
outclientfile<<account<<" "<<name<<" "<<balance
<<endl;
cout<<"? ";
}
 return 0;
}
```

البرنامج السابق ينشئ ملفاً تسلسليا؛ إذ يمكن استخدامه في نظام حسابات مثلاً ليساعد في إدارة حسابات العملاء.

يتضمن البرنامج رقم حساب العميل account واسم العميل name ورصيد العميل balance. وهذه البيانات تمثل سجل ذلك العميل.

يستخدم رقم حساب العميل بوصفه حقلا" مفتاحيا"، وعليه يكون الملف مرتباً بترتيب أرقام حسابات العملاء.

تم فتح الملف للكتابة فيه، لذلك ينشئ البرنامج كائن خرج تابع للصنف ofstream يدعى fptr، وتم تمرير وسيطين لبناء ذلك الكائن وهما أسم الملف Clients.dat، طريقة فتح الملف (File open mode) ios::out .

يقوم البرنامج، باستقبال البيانات المدخلة وحفظها في الملف، إلى أن يتم إدخال رمز نهاية الملف (<ctrl> Z).

ونلحظ أننا قمنا بتضمين ملف الترويسة stdlib.h الذي يحتوي على تعريف الدالة exit، والتي تنهي البرنامج في حالة عدم فتح الملف بصورة صحيحة.

ضع برنامجا لإظهار محتويات الملف السابق:

البرنامج:

```
//program 9-8
#include<iostream.h>
#include<fstream.h>
#include<iomanip.h>
#include<stdlib.h>
void outputline(int, char *, float);
main( )
{
      ifstream inClientFile("D:\\clients.dat",ios::in);
 if (!inClientFile) {
     cerr << "File could not be opened" <<endl;
     exit(1);
  }
int account;
char name[10];
```

```
//Continued
float balance;
cout  <<setiosflags(ios::left) <<setw(10) <<"Account"
      <<setw(13) <<"Name" <<"Balance"<<endl;
while(inClientFile >> account >>name >>balance)
 outputline(account, name, balance);
 return 0;
}
void outputline(int account, char *name, float balance)
{
      cout << setiosflags(ios::left) << setw(10)<< account
<< setw(13) << name<< setw(7)
            << setprecision(2)
<< setiosflags(ios::showpoint | ios::right)
<< balance << endl;
}
```

ويعد تنفيذ البرنامج تظهر لنا النتائج الأتية:

```
(Inactive D:\TCWIN45\BIN\NONAME01.EXE)
Account    Name          Balance
100        lina          2.3e+02
200        sami          4.6e+02
300        ali              -50.
400        fadi               0.0
500        mahmoud          -55.
600        salem              0.0
```

9-6 الملفات الثنائية:

الملفات الثنائية تستخدم لتخزين المصفوفات والسجلات وللكتابة إلى داخل أي ملف ثنائي نتبع الخطوات الآتية:

1-نعرف كائن من نوع ملف باستخدام الكود : ; ofstream fout ويمكننا أن نختار اسم كائن آخر .

2-نفتح الملف للكتابة باستخدام الكود:

```
fout.open("file path",iostream family|ios::binary);
```

3-نكتب إلى داخل الملف باستخدام الكود:

```
fout.write((char*)& data ,sizeof(data)) ;
```

ضع برنامجا لإنشاء ملف ثنائي يتضمن جدول مؤلف من عشرة عناصر ويحمل الاسم mat.bin ومخزن على القرص D .

البرنامج:

```
//program 9-9
#include<iostream.h>
#include <fstream.h>
main()
{ int a[80],i;
for(i=0;i<10;i++)
cin>> a[i];
ofstream fout;
 fout.open("D:\\mata.bin",ios::binary);
 fout.write((char *) & a , sizeof(a));
fout.close();
}
```

ملحوظة: الملف لايمكن فتحه بالطرق العادية لأنه يحمل الامتداد *.bin

9-7- القراءة من الملفات الثنائية :

للقراءة من الملفات الثنائية نتبع الخطوات الآتية:

1-نعرف كائن من نوع ملف باستخدام الكود:; ifstream fin ويمكن أن يكون الاسم fin اختياريا .

2-نفتح الملف للقراءة باستخدام الكود:

```
fin.open("file path",iostream family);
```

3-نقرأ من داخل الملف باستخدام الكود:

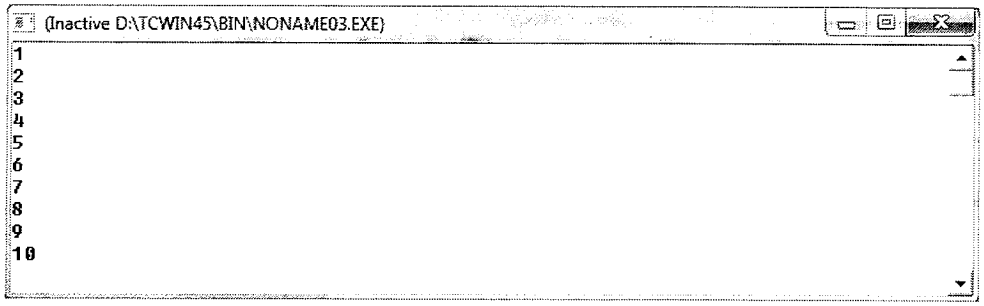
```
fin.read((char*)& array,sizeof(data));
```

مثال: ضع برنامج لإظهار محتويات الملف السابق:

البرنامج:

```
//program 9-10
#include<iostream.h>
#include <fstream.h>
main()
{ int a[10],i;
ifstream fout;
 fout.open("D:\\mata.bin",ios::binary);
 fout.read((char *) & a , sizeof(a));
for(i=0;i<10;i++)
cout<< a[i]<<endl;
fout.close();
}
```

وبعد تنفيذ البرنامج تظهر شاشة النتائج كما يأتي:



باستخدام السجلات ضع برنامجا لإنشاء ملف ثنائي يحمل الاسم exam1.bin والمخزن على القرص D والذي يتضمن أسماء ثلاثة طلاب مع درجاتهم الامتحانية.

البرنامج:

```
//program 9-11
#include<iostream.h>
#include<stdio.h>
#include <fstream.h>
main()
{
struct student
{
char name[20];
 int degre;
  };
 struct student st[3];
int i;
for(i=0;i<3;i++){
cout<<"\n enter the name: ";
 gets(st[i].name );
cout<<"enter degre: ";
 cin>>st[i].degre;}
```

```
ofstream fout;
 fout.open("D:\\exam1.bin",ios::binary);
for(i=0;i<3;i++)
 fout.write((char*)& st[i] ,sizeof( st[i]));
                fout.close(); }
```

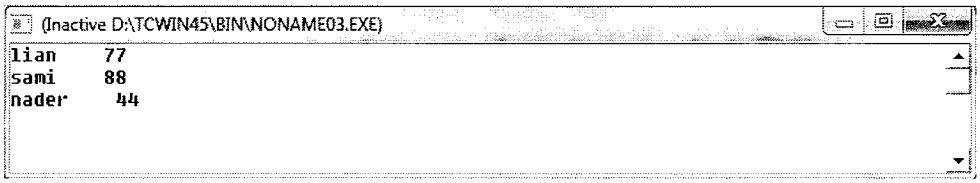
ضع برنامجالإظهار محتويات الملف السابق:

البرنامج:

```
//program 9-12
#include<iostream.h>
#include<stdio.h>
#include <fstream.h>
main()
{
struct student
{
char name[20];
 int degre;
  };
 struct student st[3];
int i;
ifstream fout;
 fout.open("D:\\exam1.bin",ios::binary);
 fout.read((char *) & st , sizeof(st));
fout.close();
for(i=0;i<3;i++){
 cout << st[i].name;
 cout<<"    "<<st[i].degre ;
```

```
cout<<"\n";}
        fout.close(); }
```

ويعد تنفيذ البرنامج تظهر لنا النتائج الآتية:



ملحوظة: إذا رغبنا في إظهار سجلين فقط نغير الكود:

```
fout.read((char *) & st , sizeof(st));
```

بالكود الآتي:

```
fout.read((char *) & st ,2*sizeof(st[0]));
```

الذي أخذناه هو أخذنا حجم سجل واحد وهو السجل الأول st[0] وضربناه ب 2 لكي نحصل على حجم سجلين من داخل الملف فقط وفي هذه الحالة يظهر على الشاشة محتويات سجلين فقط .

9-8-ملفات الوصول العشوائي:

الملفات التي استعرضناها حتى الآن ما هي إلا ملفات تسلسلية أي أنها تبدأ من بداية الملف وتستمر حتى نهايته. ولا يمكننا الوصول مباشرة إلى سجل معين من المعطيات. كما هو الحال فيما يتعلق بعمليات الحجز على رحلات الخطوط الجوية أوفيما يتعلق بأنظمة البنوك ويمكن تحقيق ذلك بوساطة ملفات الوصول العشوائي ؛إذ نستطيع الوصول إلى أي سجل ضمنها بصورة مباشرة وسريعة بدون أية عملية بحث ضمن السجلات الأخرى.

9-8-1 إنشاء ملف وصول عشوائي:

يملك كل كائن ملفا"، مؤشران مقترنان به يسميان مؤشر الحصول get pointer ومؤشر الوضع put pointer، ويسميان أيضاً مؤشر الحصول الحالي ومؤشر الوضع الحالي.

يتيح الإجراءان seekg و seekp ضبط مؤشري الحصول والوضع على التوالي.
يتم استخدم الإجراء ()seekg لتحريك المؤشر داخل الملف باستخدام الكود الآتي:

```
fin.seekg(offset, iostream family );
```

fin : هو اسم الملف داخل البرنامج

Offset: هو بدء القراءة إي من إي بايت تبدأ القراءة والكتابة .

(iostream family) : إذا لم تضع أي رمز معنا سبدأ (offset) من بداية الملف وإذا وضعنا (ios::end) معناه سيبدأ ا(offset) من نهاية الملف.

فمثلاً العبارة :-

```
seekp( -10,ios:: end);
```

ستضع مؤشر الوضع: 10 بايتات قبل نهاية الملف.

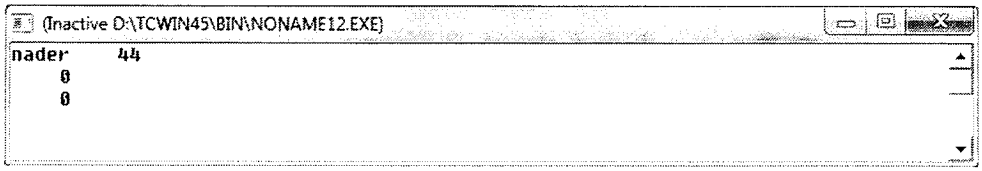
ضع برنامجا لإظهار محتويات السجل الثالث فقط والموجود في الملف exam1.bin .

البرنامج:

```
//program 9-13
#include <fstream.h>
struct student
{
char name[20];
 int degre;}  st[3];
main()
{
int i;
ifstream fout;
 fout.open("D:\\exam1.bin",ios::binary);
       fout.seekg(2*sizeof(st[0]));
 fout.read((char *) & st , sizeof(st[0]));
```

```
fout.close();
for(i=0;i<3;i++){
 cout << st[i].name;
 cout<<"    "<<st[i].degre ;
 cout<<"\n";}
                fout.close(); }
```

وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:



[redacted]ضع برنامجا" لفتح الملف ("D:\\clients.dat") الذي تم إنشاؤه سابقا ثم أظهر على الشاشة سجلات العملاء التي رصيدها صفر و التي رصيدها سالب (مديونة) وسجلات العملاء التي لها رصيد في البنك :

البرنامج:

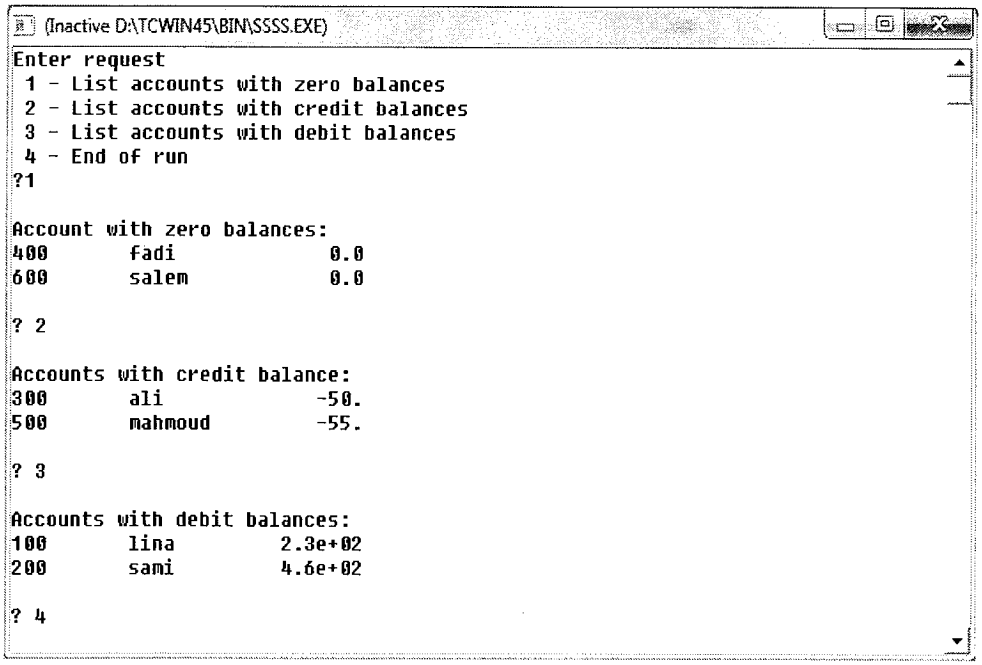
```
//program 9-14
 #include<iostream.h>
 #include<fstream.h>
 #include<iomanip.h>
 #include<stdlib.h>
 void outputline(int, char *, float);
 main( )
 {
 //Continued
```

```
  ifstream inClientFile("D:\\clients.dat", ios::in);
  if (!inClientFile){
     cerr<< "File could not be opened" <<endl;
     exit(1);
  }
cout <<"Enter request "<<endl
        <<" 1 - List accounts with zero balances" << endl
        <<" 2 - List accounts with credit balances"<<endl
        <<" 3 - List accounts with debit balances"<<endl
        <<" 4 - End of run" <<endl<<"?";
int request;
//Continued
cin>> request;
while(request !=4) {
int account;
        char name[10];
float balance;
inClientFile >>account >>name>>balance;
switch (request) {
    case 1:
       cout <<endl <<"Account with zero balances:"
              <<endl;
      while(!inClientFile.eof()) {
        if (balance==0)
           outputline(account, name, balance);
       inClientFile >>account >>name >>balance;
}
```

```
 break;
 case 2:
        cout<<endl<<"Accounts with credit balance:"
          <<endl;
  while(!inClientFile.eof()) {

  if (balance <0)
              outputline(account, name, balance);
//Continued
         inClientFile>>account >>name >>balance;
}
break;
case 3:
       cout<<endl<<"Accounts with debit balances:"
                    <<endl;
        while(!inClientFile.eof()) {
              if (balance > 0)
               outputline(account, name, balance);
inClientFile >>account>>name>>balance;
}
break;
}
  inClientFile.clear( );       //reset eof for next input
  inClientFile.seekg(0); //position to beginning of file
  cout<<endl <<"? ";
  cin>>request;
} }
```

```
void outputline(int account, char *name, float balance)
{
        cout << setiosflags(ios::left) << setw(10)<< account
                << setw(13) << name<< setw(7)
                << setprecision(2)
<< setiosflags(ios::showpoint | ios::right)
<< balance << endl;
}
```

وبعد تنفيذ البرنامج تظهر النتائج كما يأتي:



9-8-تمارين محلولة:

1-ضع برنامجا لتخزين ملف نصي على القرص D تحت اسم elph.txt إذ تنهي عملية إدخال النص بالضغط عل enter .

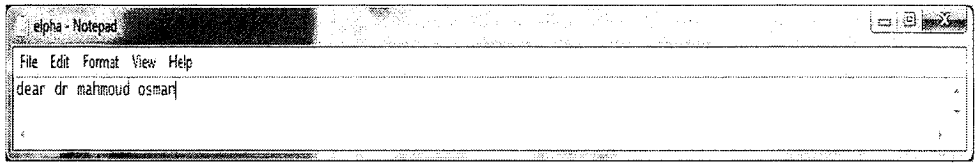
البرنامج:

```
#include <fstream.h>
 #include<iostream.h>
#include<stdio.h>
int main()
 { ofstream test1;
        char word[50];
int i,ch,n;
n=0;
 test1.open("D:\\elpha.txt");
        do{
  ch=getchar();
  word[n]=ch ;
  test1<<word[n];
  n++;}while (ch!='\n');
 test1.close();
}
```

وأثناء تنفيذ الملف كتبنا العبارات الآتية:

Dear dr mahmoud osman

ثم ضغطنا على مفتاح enter فإذا ذهبنا إلى القرص D وفتحنا الملف elph.txt فتكون شاشة الإخراج على النحو الآتي:

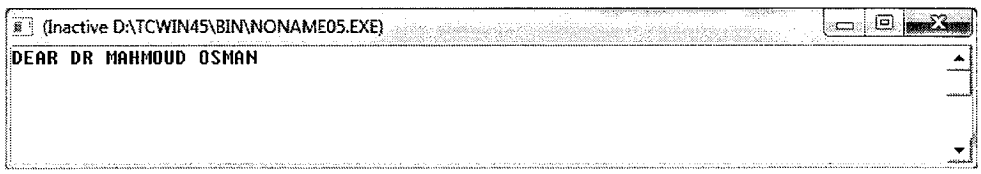


2-ضع برنامجا لإظهار محتويات الملف السابق

البرنامج:

```
 #include <fstream.h>
int main()
 {ifstream test;
 char array[80];
 test.open("D:\\elpha.txt",ios::out);
 while(!test.eof())
 {test.getline(array,80);
cout<<array<<endl;}
 test.close();
}
```

ويعد تنفيذ البرنامج تظهر النتائج على النحو الآتي:



3. ضع برنامجا يتضمن إنشاء ملف يحتوي على أسماء ستة موظفين مع كلمة السر لكل منهم:

```
#include<iostream.h>
#include<stdio.h>
#include <fstream.h>
struct password_User
{char username[20];
```

```
 char password[20]; }userProtection[6];
main()
{int i;
for(i=0;i<6;i++){
cout<<"\nenter user name: ";
 gets(userProtection[i]. username );
cout<<"enter password: ";
 gets( userProtection[i]. password);}
ofstream fout;
 fout.open("D:\\up.txt");
for(i=0;i<6;i++){
fout << "User ID :";
fout << userProtection [i]. username;
fout << "\tuser Password: ";
fout << userProtection [i]. password;
fout <<"\n";}
 fout.close();}
```

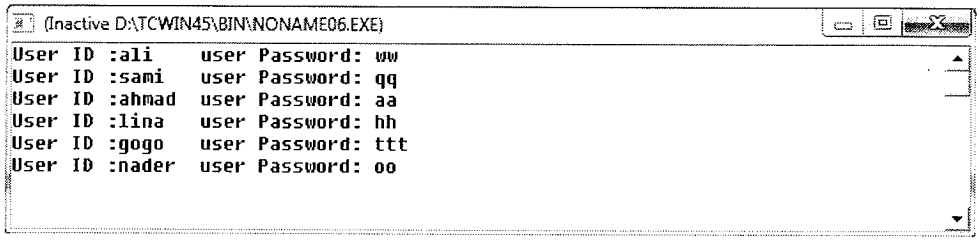
4 . ضع برنامجا لإظهار محتويات الملف السابق:

البرنامج:

```
#include <fstream.h>
int main()
 {ifstream test;
 char array[80];
 test.open("D:\\up.txt",ios::out);
 while(!test.eof())
 {test.getline(array,80);
cout<<array<<endl;}
```

```
 test.close();
}
```

وبعد تنفيذ البرنامج تكون النتائج كما يأتي:



2- ضع برنامجا لإنشاء ملف ثنائي يتضمن أسماء ستة موظفين مع كلمة السر لكل منهم.

البرنامج:

```
#include<iostream.h>
#include<stdio.h>
#include <fstream.h>
struct password_User
{char username[20];
 char password[20]; }userProtection[6];
main()
{int i;
for(i=0;i<6;i++){
cout<<"\nenter user name: ";
 gets(userProtection[i]. username );
cout<<"enter password: ";
 gets( userProtection[i]. password);}
ofstream fout;
 fout.open("D:\\up.bin",ios::binary);
for(i=0;i<6;i++)
```

```
 fout.write((char*)& userProtection[i] ,sizeof(
userProtection[i]));}
```

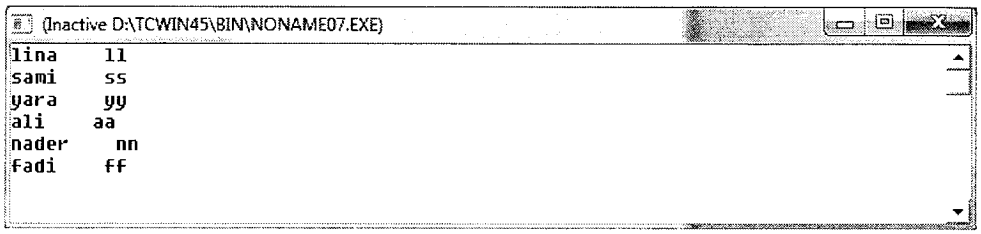
3-ضع برنامجا لإظهار محتويات الملف السابق

```
#include <fstream.h>
struct password_User
{char username[20];
 char password[20]; }userProtection[6];
main()
{
int i;

ifstream fout;
 fout.open("D:\\up.bin",ios::binary);
 fout.read((char *) & userProtection , sizeof(userProtection));

fout.close();

for(i=0;i<6;i++){
 cout << userProtection[i].username;
 cout<<"    "<<userProtection[i].password ;
 cout<<"\n";}

            fout.close(); }
```

ويعد تنفيذ البرنامج تكون النتائج كما يأتي:

```
#include <fstream.h>
struct password_User
{char username[20];
```

```
 char password[20]; }userProtection[6];
main()
{
int i;

ifstream fout;
 fout.open("D:\\up.bin",ios::binary);
 fout.read((char *) & userProtection ,2*
sizeof(userProtection[0]));

fout.close();

for(i=0;i<6;i++){
 cout << userProtection[i].username;
 cout<<"    "<<userProtection[i].password ;
 cout<<"\n";}

            fout.close(); }
```



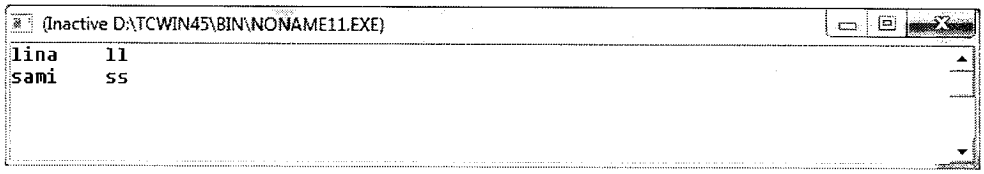
4-ضع برنامجا لإظهار محتويات سجلين فقط من الملف السابق:

البرنامج:

```
#include <fstream.h>
```

```
struct password_User
{char username[20];
 char password[20]; }userProtection[6];
main()
{
int i;

ifstream fout;
 fout.open("D:\\up.bin",ios::binary);
 fout.read((char *) & userProtection ,2*
sizeof(userProtection[0]));

fout.close();

for(i=0;i<6;i++){
 cout << userProtection[i].username;
 cout<<"    "<<userProtection[i].password ;
 cout<<"\n";}

                fout.close(); }
```

ويعد تنفيذ البرنامج تكون النتائج كما يأتي:

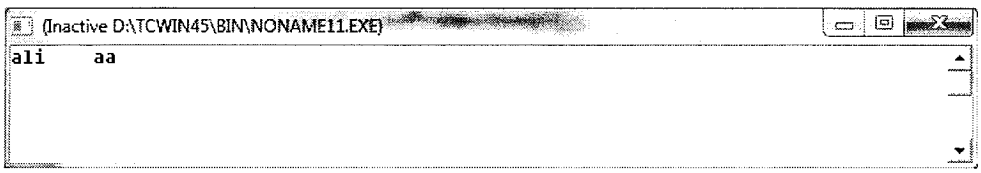


5- ضع برنامجا لإظهار محتويات السجل الرابع فقط:

البرنامج:

```
include <fstream.h>
struct password_User
{char username[20];
 char password[20]; }userProtection[6];
main()
{
int i;
ifstream fout;
 fout.open("D:\\up.bin",ios::binary);
 fout.seekg(3*sizeof(userProtection[0]));
fout .read((char *) & userProtection,
sizeof(userProtection[0]));
fout.close();
for(i=0;i<6;i++){
 cout << userProtection[i].username;
 cout<<"    "<<userProtection[i].password ;
 cout<<"\n";}
          fout.close(); }
```

وبعد تنفيذ البرنامج تكون النتائج كما يأتي:



تمارين9

1-ضع برنامجا لإنشاء ملف يتم إدخال مجموعة من السجلات بحيث يتألف السجل من الحقول االآتية:رقم العميل،اسم العميل،الرصيد.

2-ضع برنامجا لإظهار محتويات الملف السابق.

3-أنشئ ملفا" للموظفين يدعى employee على أن يحتوي كل سجل في الملف على الحقول االآتية:

- رقم الموظف.
- اسم الموظف.
- العنوان.

ثم قم بإدخال بيانات خمسة موظفين.

4-تأكد من إدخال البيانات في السؤال السابق بصورة صحيحة وذلك بكتابة برنامج لقراءة محتويات الملف.

5--قم بكتابة برنامج يقوم باستقبال معلومات عن طلاب كلية ويضعها في ملف يسمى Students، بحيث يحتوي ملف الطلاب على الآتي:

- رقم الطالب.
- اسم الطالب.
- تخصص الطالب.
- درجة الطالب.

6-ضع برنامجا يقوم بإظهار محتويات الملف السابق.

محتويات أشهر مكتبات السي Header files

1. محتويات مكتبة alloc.h.
2. محتويات مكتبة bios.h.
3. محتويات مكتبة conio.h.
4. محتويات مكتبة ctype.h.
5. محتويات مكتبة dir.h.
6. محتويات مكتبة dos.h.
7. محتويات مكتبة graphics.h.
8. محتويات مكتبة io.h.
9. محتويات مكتبة math.h.
10. محتويات مكتبة process.h.
11. محتويات مكتبة signal.h.
12. محتويات مكتبة stdarg.h.
13. محتويات مكتبة stdio.h.
14. محتويات مكتبة stdlib.h.
15. محتويات مكتبة string.h.
16. محتويات مكتبة time.h.
17. محتويات مكتبة iostream.h.
18. محتويات مكتبة iomanip.h.
19. محتويات مكتبة fstream.h.

(1)
alloc.h

brk( )	farcalloc( )	farmalloc( )	malloce( )
calloc( )	farcoreleft( )	farrealloc( )	realloc( )
coreleft( )	farfree( )	free( )	sbrk( )

(2)
bios.h

bioscom( ) biosdisk( ) biosequip( )	bioskey( ) biosprint( ) biostime( )

(3)
conio.h

cgets( ) clreol( ) clrscr( ) cprintf( ) cputs( ) cscanf( ) delline( )	getch( ) getche( ) getpass( ) gettext( ) gettextinfo( ) gotoxy( ) highvideo( )	insline( ) kbhit( ) lowvideo( ) movetext( ) mormvideo( ) putch( ) puttext( )	textattr( ) textcolor( ) textbackground( ) textmode( ) ungetch( ) wherex( ) wherey( ) window( )

(4)
ctype.h

tolower( ) toupper( ) isalnum( ) isalphal( )	isascii( ) iscntrl( ) isdigit( ) isgraph( )	islower( ) isprint( ) ispunct( ) isspace( )	isupper( ) toascii( ) tolower( ) toupper( )

(5)
dir.h

chdir( ) findfirst( ) findnext( )	fnmerge( ) fnsplit( ) getcurdir( )	getcwd( ) getdisk( ) mkdir( )	mktemp( ) rmdir( ) searchpath( ) setdisk( )

(6)
dos.h

absread( )	geninterrupt( )	importb( )	pokeb( )
abswrite( )	getcbrk( )	int89( )	randbrd( )
allocmem( )	getdate( )	int86x( )	randbwr( )
bdos( )	getdta( )	intdos( )	segread( )
bdosprt( )	getfat( )	intdosx( )	setblock( )
country( )	getfatd( )	intr( )	setcbrk( )
ctrlbrk( )	getfree( )	keep( )	setdate( )
delar( )	getpsp( )	MK_FP( )	setdta( )
disable( )	gettime( )	nosound( )	settime( )
dosexterr( )	getvect( )	output( )	setvect( )
dostounix( )	getverify( )	outputb( )	setverify( )
enable( )	harderr( )	parsfnm( )	sleep( )
FP-OFF( )	hardresume( )	peek( )	sound( )
FP_SEG( )	hardretn( )	peekb( )	unixtodos( )
freemem( )	inport( )	poke( )	unlink( )

(7)
graphics.h

_graphfreemen	getdrivername( )	grapherrormsg( )	sector( )
_graphgetmem( )	getfillpattern( )	graphresult( )	setactivepage( )
arc( )	getfillsettings( )	imagesize( )	setallpalette( )
bar( )	getgraphmode( )	initgraph( )	setaspectratio( )
bar3d( )	getimage( )	installuserdriver( )	setbkcolor( )
circle( )	getlinesettings( )	installuserfont( )	setcolor( )
cleardevice( )	getmaxcolor( )	line( )	setfillpattern( )
clearviewport( )	getmaxmode( )	linerel( )	setfillstyle( )
closegraph( )	getmaxx( )	lineto( )	setgraphbufsize( )
detectgraph( )	getmaxy( )	moverel( )	setgraphmode( )
drawpoly( )	getmodename( )	moveto( )	setlinestyle( )
ellipse( )	getmoderange( )	outtext( )	setpalette( )
fillellipse( )	getpalette( )	outtextxy( )	setrgbpalette( )
fillpoly( )	getpalettsize( )	pieslice( )	settextjustify( )
floodfill( )	getpixcel( )	putimage( )	settextstyle( )
getarccoords( )	gettextsettings( )	putpixel( )	setusercharsize( )
getaspectratio( )	getviewsettings( )	rectangle( )	setviewport( )

getbcolor( ) getcolor( ) getdefaultpalette( )	getx( ) gety( ) graphdefaults( )	registerbgidriver( ) registerbgifont( ) restorecrtmode( )	setvisualpage( ) setwritemode( ) textheight( ) textwidth( )

(8)
io.h

_chmod( ) _close( ) _creat( ) _open( ) _read( ) _write( ) access( ) chmod( )	chsize( ) close( ) creat( ) creatnew( ) creattemp( ) dup( ) dup2( ) eof( )	filelength( ) getftime( ) ioctl( ) isatty( ) lock( ) lseek( ) open( ) read( )	setmode( ) settime( ) sopen( ) tell( ) unlock( ) vsscanf( ) write( )

(9)
math.h

_matherr( ) acos( ) asin( ) atan( ) atan2( ) atof( ) bypot( ) cabs( )	ceil( ) cos( ) cosh( ) div( ) ecvt( ) exp( ) fabs( ) floor( )	fmod( ) frexp( ) ldexp( ) idiv( ) log( ) lob10( ) matherr( ) matherr( )	modf( ) poly( ) pow( ) pow10( ) sin( ) sinh( ) sqrt( ) tan( ) tanh( )

(10)
process.h

_exit( ) abort( ) execl( ) execle( ) execlp( )	execlpe( ) execve( ) execvp( ) execvpe( )	exit( ) spawnl( ) spawnle( ) spawnlp( ) spawnlpe( )	spawnv( ) spawnve( ) spawnvp( ) system( )

(11)
signal.h

raise( )	signal( )

(12)
stdarg.h

va_arg( )	va_end( )	va_start( )

(13)
stdio.h

_strerror( )	fprintf( )	gets( )	setvbuf( )
fclose( )	fputc( )	getw( )	sprintf( )
fcloseeall( )	fputchar( )	perror( )	sscanf( )
feof( )	fputs( )	printf( )	strerror( )
ferror( )	fread( )	putc( )	tmpfile( )
fflush( )	freeopen( )	putchar( )	tmpnam( )
fgetc( )	fscanf( )	puts( )	ungetc( )
fgetchar( )	fseek( )	putw( )	vfprintf( )
fgetpos( )	fsetpos( )	remove( )	vfscanf( )
fgets( )	ftell( )	rename( )	vprintf( )
fileno( )	fwrite( )	rewind( )	vscanf( )

flushall( ) fopen( )	getc( )	scanf( ) setbuf( )	vsprintf( )

(14)
stdlib.h

_exit( )	atoi( )	itoa( )	randomize( )
_lrotl( )	atol( )	labs( )	realloc( )
_trotr( )	bsearch( )	lfind( )	srand( )
_rotl( )	calloc( )	lsearch( )	strtod( )
_rotr( )	ecvt( )	ltoa( )	strtol( )
abort( )	exit( )	malloc( )	strtoul( )
abs( )	fcvt( )	putenv( )	swab( )
acvt( )	free( )	qsort( )	system( )
atexit( )	gcvt( )	rand( )	ultoa( )
atof( )	getenv( )	random( )	ultra( )

(15)
string.h

memchr( )	fmovmem( )	strdup( )	strnset( )
memcmp( )	setmem( )	strerror( )	strpbrk( )
memcpy( )	strcat( )	stricmp( )	strrchr( )
memcpy( )	strchr( )	strlen( )	strrev( )
memicmp( )	strcmp( )	strlwr( )	strset( )
memmove( )	strcmpi( )	strncat( )	strspn( )
memset( )	strcmp( )	strncmp( )	strstr( )
movedata( )	strcpy( )	strnicmp( )	strtok( )
			strupr( )

(16)
time.h

asctime( ) ctime( )	difftime( ) gmtime( )	localtime( ) stime( )	time( ) szset( )

(17)
iostream.h

endl ends	flush dec	hex oct	ws

(18)
iomanip.h

setw( ) setfill( )	setbase( ) setprecision( )	setiosflags( ) resetiosflag( )

(19)
fstream.h

cin cout	cerr cprn	caux

المراجع العربية:

1- عثمان، محمود . *الجزء العملي لمقرر التحليل العددي والبرمجة*. منشورات جامعة تشرين 2010-2009.

2- عثمان ، محمود . *الجزء النظري لمقرر التحليل العددي والبرمجة*. منشورات جامعة تشرين 2011-2010.

3- سليمان، علي . *البرمجة غرضية التوجه في لغة C++* . منشورات جامعة تشرين 2010-2009

4- دوجي، صلاح .*كيف تبرمج بلغة C++* . منشورات شعاع للنشر والعلوم 2004 .

5- C+++.doc ملف من الإنترنيت.

المراجع الأجنبية:

1- **Juan Soulié**, *C++ Language Tutorial Last revision* : **June, 2007.**

2- *Steve Oualline* ***Practical C++ Programming*** . Copyright © 1995 O'Reilly & Associates, Inc. All rights reserved Printed in the United States of America.

3- **Herbert Sehildt ,c++ the complete reference 1998 by meGraw -Hill**

دليل المصطلحات العلمية

الإنكليزية	العربية
A	
Abstract classes	الأصناف المجردة
action	نشيط
Address operator	عملية العنوان
Arithmetic operators	العمليات الحسابية
array	صف
arrays	المصفوفات
Arrays pointers	مصفوفات المؤشرات
B	
Base classes	الأصناف الأساسية
block	كتلة
C	
Cal-by-reference	الاستدعاء بالمرجع
Call-by-value	الاستدعاء بالقيمة
case	حالة
Character constants	الثوابت المحرفية
classes	أصناف
columns	أعمدة
compiler	المترجم
concepts	مفاهيم
Conditional operator	المعامل الشرطي
Constant variable	متغير ثابت
counter	عداد

D	
Data type	أنواع البيانات
declaration	التصريح
Declaring local variables	التصريح عن المتغيرات المحلية
Declaring pointer	التصريح عن المؤشر
destructor	المدمر
E	
events	أحداث
exponentiation	الرفع إلى قوة
expressions	التعابير
F	
File processing	التعامل مع الملفات
Floating-point numbers	أعداد الفاصلة العائمة
Friends classes	الاصناف الصديقة
H	
Header files	الملفات الرأسية
I	
include	إحتواء
inheritance	الوراثة
Inheriting multiple base classes	وراثة عدة أصناف أساس
Initial state	الحالة الابتدائية
Initialize variables	إعطاء قيم ابتدائية
Integer constant	الثوابت الصحيحة
interprter	المفسر
K	
keywords	كلمات مفتاحية
L	

links	روابط
loading	مرحلة التحميل
Local classes	الأصناف المحلية
Logical operators	المعاملات المنطقية
M	
Member access specifiers	محددات طريقة الوصول إلى الأعضاء
Memory	ذاكرة
Multiple inheritance relationships	علاقات التوريث المتعدد
N	
Nested classes	الأصناف المتداخلة
Nested loops	حلقات متداخلة
newline	سطر جديد
Number systems	نظم العد
O	
Objct-oriented programming	البرمجة غرضية التوجه
objects	أغراض
Opening and closing a file	فتح وإغلاق ملف
operations	عمليات
Operator overloding	التحميل الزائد للعمليات
Output stream	مجاري الخرج
P	
parameters	وسطاء
Poinrer ' operators	معاملات المؤشرات
Pointer 's expressions	تعابير المؤشرات
pointers	مؤشرات
polynomial	كثيرات الحدود
private	خاص

Protected	محمي
public	عام
R	
Random access	الوصول العشوائي
Random-access files	ملفات الوصول العشوائية
Reading and writing text files	القراءة والكتابة من وإلى الملفات النصية
Real constant	الثوابت الحقيقية
references	المراجع
Relational operators	المعاملات العلائقية
rows	أسطر
S	
Sequential execution	التنفيذ التسلسلي
size	حجم
statement	تعليمة
Static class members	الأعضاء الساكنة للصنف
Static data members	بيانات الأعضاء الساكنة
strings	السلاسل الحرفية
Strings arrays	مصفوفات السلاسل
Structured programming	البرمجة المهيكلة
switch	بنية الاختيار
T	
tables	جداول
templates	القوالب
The c++ stream classes	أصناف مجاري c++
The pointer arithmetic	العمليات الحسابية على المؤشرات
The pointer comparison	مقارن المؤشرات
type	نمط

V	
قيمة	value
المتغيرات	variables

اللجنة العلمية:

1- السيد الدكتور محمد حسن ،أستاذ في كلية العلوم ،جامعة تشرين،إختصاص نظرية البيان

2- السيد الدكتور المهندس رضوان دنده،أستاذ في كلية المعلوماتية ، جامعة تشرين،إختصاص أجهزة اتصالات

3- السيدالدكتور المهندس جبر حنا،أستاذ في كلية الهمك ،جامعة تشرين،إختصاص حاسبات وتحكم

مدقق اللغة العربية:

السيد الدكتور سمران متوج ، أستاذ في كلية الآداب،جامعة تشرين

www.facebook.com/groups/Applied.Mathematics.Tishreen.University/