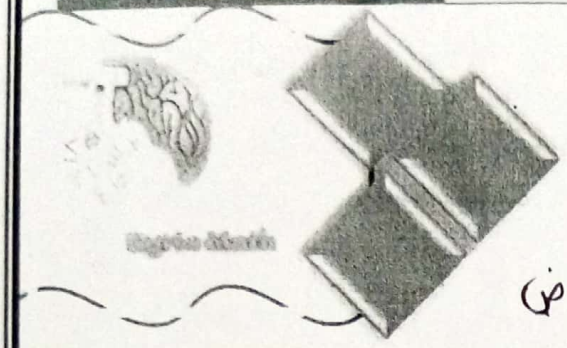


المحاضرة السادسة

◄ دكتور المادة: سمير جعفر

◄ عنوان المحاضرة: الصّفوف والأغراض

نظري ☑ عملي ☐

أهلاً بكم أصدقائي في المحاضرة السادسة من مقررنا البرمجة المتقدمة، وسنتناول مفهوم الصفوف في لغة جافا.

لنبدأ معاً...

- رأينا أنّ لغة الجافا لغة كائنية التوجه (object-oriented).

فهي مجموعة من الكائنات تتعاون مع بعضها البعض عن طريق الرسائل لحلّ مسألة ما.

- لغة الجافا لا تسمح باستخدام أي متغيّر دون تعريف نوعه (كلّ كائن يجب أن يكون من نوع معيّن)، لذلك سنتعرّف على الصّف فهو يمكننا من تعريف أنواع جديدة في اللّغة ممّا يسهّل علينا التعامل مع اللّغة.

الصّف: بنية معطيات تمكّننا من تعريف أنواع جديدة في اللّغة مع تحديد إمكانيات الوصول إلى محتوياته.

- إنّ الصّف يمكّننا من التصريح عن متغيرات من نوع صف، وهذه يصبح لدينا أنواع جديدة حسب حاجتنا لاستخدامها.

- يمكن تحديد إمكانيات الوصول إلى محتوياته:

وذلك عن طريق محدّدات الوصول لكلّ عضو في الصّف (( تعرّفنا على محدّدات الوصول في C++، وهي: افتراضي - عام - محمي - خاص)).

- إنّ الصّف في جافا يحتوي على:

1. معطيات أعضاء (أعضاء بيانية)
2. الطّرق (دوال أعضاء) Methods

- الشكل العام للصنف :

```
class  اسم الصنف {
        معطيات أعضاء ...
        طرق أعضاء ...
}
```

اسم الصنف Name of class : اسم اختياري على أن يحترم قواعد التسمية ويبدأ بحرف كبير (( الاتفاقية بين المبرمجين وليست قاعدة ))

- كلمة class كلمة مفتاحية .

- المعطيات الأعضاء : هي إما متغيرات بسيطة ( int, char, -- ) أو مركبة ( مثل المتجهة -- ) أو أنواع صديدة معرفة مسبقاً (فن بنيتها).

والتصريح عن متغيرات يكون كما تعلمنا سابقاً

مثلاً : التصريح عن متغير صحيح اسمه n : `int n ;`

- الدوال الأعضاء (الطرق) : عبارة عن دوال عادية تمثل العمليات التي نريد القيام بها في الصنف (تنفذ مهام محددة على متغيرات)

كتابة دالة : نوع الإرجاع ( قائمة الوسطاء ) اسم الدالة

{ -- -- -- }

- مثال عن صنف :

```
①
class Myclass{
② int n;
③ void myprint() { S.o.p(n); }
}  ← قوس الصنف
```

① سمينا الصنف Myclass (( كلمة واحدة ، أي لا نضع فاصلاً بين My و class لأن كلمة class كلمة محجوزة ، لكن إذا كانت مع My سيفهمها البرنامج على أنها اسم ، أما إذا كانت وحدها سيعطي خطأ ترجمة ))

② عرفنا متغير n من نوع صحيح . ③ دالة الطباعة اسمها myprint نوع إرجاعها void تطبع العدد الصحيح

- الطرق جزء منها يدعى طرقاً خاصة تسمى بواني :

البواني : طريقة (دالة) خاصة تكتب داخل الصنف ، لها اسم ويمكن أن تقبل مجموعة من الوسطاء ويجب أن يطابق اسم الباني اسم الصنف المعرّف عليه.

يختلف الباني عن الدوال العادية في أنه لا يملك نوع إرجاع .

مهمّة الباني : بناء الأغراض ضمن الصنف ، أي مهمته إعطاء قيم ابتدائية لمتحولات الصنف الخاصة بهذا الغرض.

- دون باني لا يمكن إنشاء غرض ، وبالتالي لا نستطيع استخدام الغرض.
- نظراً لإمكانية إعطاء قيم ابتدائية لمعطيات الصنف بأشكال مختلفة وبالاعتماد على التحميل الزائد للدوال يمكننا كتابة أكثر من باني للصنف الواحد.

تحميل الزائد للدوال : يمكننا من كتابة أكثر من دالة لها الاسم نفسه، على أن تختلف عن بعضها بقائمة الوسطاء ((العدد - النوع - الترتيب (إذا كانت من أنواع مختلفة))

ترويسة الدالة : هي اسم الدالة مع قائمة وسطائها ، وهذا ما يجب أن يكون مختلفاً كي لا تتشابه الدوال.

- التصريح عن كائن من صنف معيّن : يتم بمثل طريقة التصريح عن متغيّر عادي ، وله الشكل:

اسم الصنف اسم الغرض ;

مثلاً للتصريح عن كائن اسمه soso من الصنف Myclass نقوم بكتابة :

```
Myclass soso;
```

- لاستخدام كائن من الصنف لا يكفي فقط التصريح عنه ، بل يجب تخزينه في الذاكرة ، أي يجب أن يكون مبنياً بحجز مكان له في الذاكرة وإعطائه قيماً ابتدائية وذلك عن طريق استخدام الكلمة new ومن ثم النوع المراد حجزه للغرض ، وهذه هي طريقة الباني في الصنف.

مثلاً :

```
Myclass soso;
soso = new Myclass( );
```

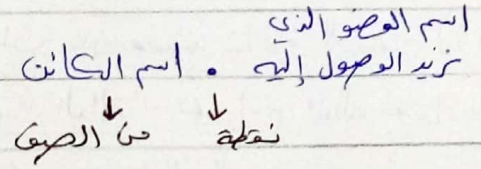
مرحلة البناء والحجز

- كما تعلمنا في C++ : في حال أن اللغة لم تجد باني في الصنف تقوم بتوليد باني افتراضي وهو الباني الخالي بحيث يعطي قيماً افتراضية للمتغيرات الأعضاء.

القيم الافتراضية للمتغيرات :

- int , float , long, double : 0
- char : المحرف الخالي
- boolean : false
- string : null

- للوصول إلى أي حقل أو عملية (دالة) داخل الصنف نقوم باستخدام النقطة (استدعاء متغير من نوع الصنف)




ملاحظة : من الأفضل كتابة باني خالي في الصنف دائماً.
في حال وجد باني واحد على الأقل فإن اللغة لن تولّد الباني الافتراضي.

مثال للتوضيح :

```
class Newtype {
int x;
char c;
Newtype ( ) {x = 0 ; }
Newtype (int a) { x = a ; }
Newtype (int a, char b) { x = a ;     c = b ; }

void print ( ) {        ← void حالة الطباعة نوع إرجاعها
```

```
System.out.print("x=" + x + "," + c);
return; } }
```

يمكن الاستغناء عن تعليمة return كون نوع إرجاع الدالة void

- عرّفنا ثلاث بواني:

الباني الأول لا يأخذ وسطاء ويعطي الـ x قيمة صفر.

الباني الثاني يأخذ وسيطاً صحياً a ويعطي قيمته لـ x.

الباني الثالث يأخذ وسيطين: صحيح a ومحرفي b، ويعطي قيمة a لـ x، و b لـ c

- نضع ماسبق في ملف مستقل وننهيه باسم الصنف نفسه (لسهولة الاستخدام)
- في حال وجدت كلمة public قبل الصنف عندها الجافا ستلزمك بتخزين الملف بنفس اسم الصنف، وفيما عدا ذلك أنت غير ملزم بتخزينه بنفس الاسم.
- لا يجب أن يكون هناك أكثر من صنف واحد عام ضمن الملف.
- إذا وجد أكثر من صنف أحدها عام يخزّن الملف بنفس اسمه.

مثال:

```
class UseNewtype {
public static void main(String args[]){
Newtype nt;          → تصريح وليس بناء وحجز
                       متحوّل من نوع صنف
nt = new Newtype();  → مرحلة البناء
nt.print();          → استدعاء الطريقة
nt = new Newtype(5, 'A');  → بناء الغرض مجدداً باستعمال الباني الثالث.
nt.print();
}
}
```

ملاحظة: new تريد دائماً بعدها نوع حتى تعرف الحجم الذي تريد حجزه.

انتهت المحاضرة.