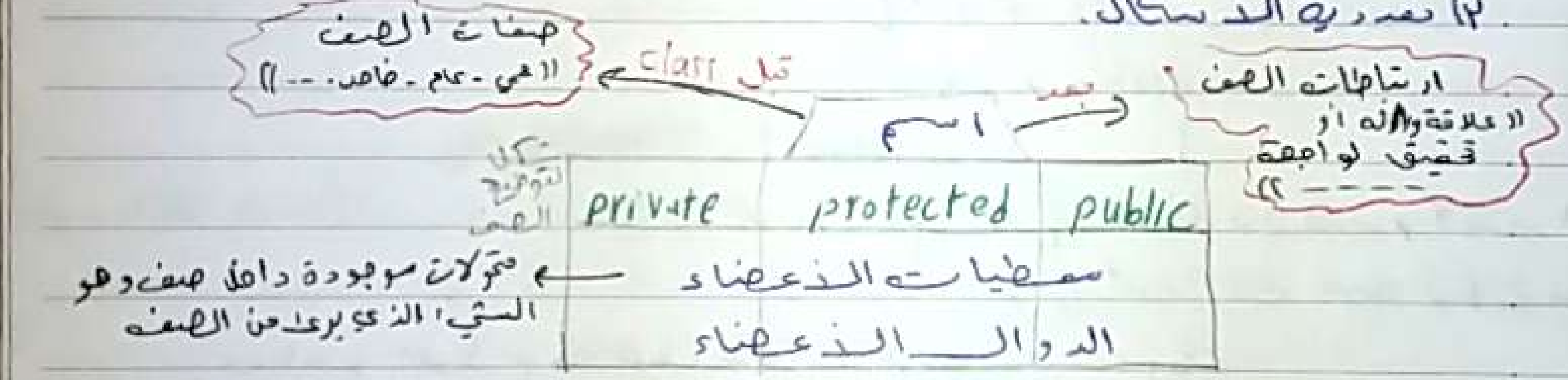
تمارين :

1) اكتب خوارزمية عودية تقوم بحساب جداء عددين .
2) " " " " ناتج قسمة عددين صحيحين .
3) " " " " " الجداء الداخلي لشعاعين من البعد n .
4) " " " " بإرجاع دليل أكبر عنصر في متجه .
5) " " " " أكبر عنصر في متجه .
6) " " " بيان فيما إذا كان عنصر x موجود أم لا في متجه معطاة

البرمجة غرضية التوجه Java
object oriented programming (o.o.p).

المبادئ الأساسية للبرمجة غرضية التوجه .

1) التجريد " للصفوف والأغراض object "
2) الوراثة
3) تعددية الأشكال .



// هامش : ممكن عدم وجود معطيات أعضاء ويوجد دوال أعضاء في الصف //

الصف السابق : "هو نوع جديد يقدم خدمات تنفذ" ولكي استفيد من هذه الخدمات يجب استخدام هذا النوع ← تعريفه أو التصريح عن متحولات من هذا النوع

المتحولات هي الأغراض object

بالمعطيات الأعضاء : نصرح من الأنواع القياسية بلغة جافا .

وكذلك المتجهات ويمكن استخدام كمعطى عضوي الصف أي نوع مركب معرف من قبل تعريف هذا الصف

مثال {
class complex {
double x ; // قسم حقيقي
double y ; // قسم تخيلي
}

//هامش: ذكر الدكتور أنه في الحالة العامة لا يشترط أن يكون اسم الملف «الذي تم حفظ الصف فيه» يملك نفس اسم الصف أي الملف نحفظه باسم Test أو First أو أي اسم كان.

```
clas Test {
   ...
}
```

ملاحظة: بلغة الجافا ضمن الملف الواحد يمكن أن يوجد أكثر من صف وجميع الطريقة (الدالة main).

- وحالة خاصة ((سابقة لأوانها)) إذا وضع أمام أحد الصفوف public أو pruivte ... «حددنا الصف محدد وصول». هنا يبقى الموضوع مياعي لأنه اسم الملف يجب أن يكون باسم الصف أي يكون اسم الملف واسم الصف First ولا يمكن وضع أكثر من صف عام ضمن هذا الصف.

```
public class First {
   ...
}
```

انتهى الهامش. //

بداية المحاضرة:

مثال: اكتب صفاً يمثل نقطة في الاحداثيات الديكارتية في المستوي؟

```
class point {
double x , y ;
}
```

point p_1 ; → object   or   point p_1 , p_2 ;
point p_2 ;

نوع جديد لا نستخدمه يجب تعريف متحولات منه.

- أي متحول نوعه صف يدعى غرض Object
- التصريح عن غرض لا يشمل أو لا يحجز مكان له في الذاكرة، ولحجز مكان له في الذاكرة نقوم ببناء هذا الغرض (object).

// هامش: الفرق بين متحول بسيط و object.

int x ; أي x متحول بسيط وهنا نصرح ونحجز في نفس الوقت بالذاكرة: x □

- أما object فيكون التصريح عنها بالشكل: point p_1 ,

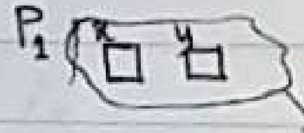
أي أن p_1 هو غرض (object) من النوع point وبقي هنا صرحنا ولم نقم بالحجز في الذاكرة. //

/ /

عملية البناء :

1) حجز مكان مناسب لهذا الغرض بالذاكرة .

2) إعطاء قيم ابتدائية للمعطيات الأعضاء .



// يعني P_1 لازم يحجز لـ x و y في ذاكرة // ⟵

الوصول للمتحول x التابع للغرض P_1 يتم كما يلي

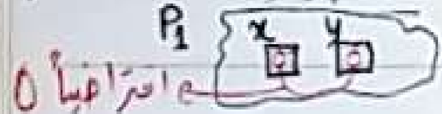
$P_1.x$

وأيضاً هكذا بالنسبة لـ y : $P_1.y$

تذكير بتعليمة NEW : تحجز مكان في ذاكرة يتسع لمتحول من النوع الذي يكتب بعدها

أصبحت دالة P_1 = New point ( ) ;

// point دالة تقوم بعملية بناء الغرض يعني بشكل أدق New الحجز ودالة البناء



// القيم الافتراضية بالنسبة للأرقام هي 0 // ⟵

الباقي هي الدالة point هنا أما ... نستطيع أن نعطي قيم ابتدائية الخاصة بالمتحول للغرض الصحيح

لو بدنا نقطة P_1 (3,2)

```
P1 = New point( ); P1 . x = 3 ;
                   P1 . y = 2 ;
                    P2 = New point ( );
                   P2 . x = 8 ;      P2 . y = 2 ;
```

// تنويه: وهذه القيم البدائية تكون بعد الحجز //

والنقطة P_2 (8, 2)

نعود للصف يلي بلشنا فيه المحاضرة

وبدنا نحسب المسافة بين نقطتين

```
class point {
double x ; double y;
double distand(double a , double b)
  {    return (Mat.sqrt(Math.pow(X-a,2) + (y-b)(y-b)) ;
  }
```

يتم استدعاء الدوال من خلال object

/ /

```
double d1 = p1 .distanc(p2.x , p2.y);
double d2 = p2 .distanc(p1.x, p1.y);
    System.out.print(p1.distanc(p2));

double distance(point p){                    أو :
return(Math.sqrt(Math.pow(x-p.x,2)+Math.pow(y-p.y,2)));
  }
  }// end of point class
```

انتهت

Scanned by CamScanner