



MehdiBouchachi
Web Developer

THE COMPLETE JAVA



BY MEHDI BCH

SECTION
JAVA ADVANCE

LECTURE
ALL IN ONE!

NOTE

NOTE

☞ ماتفوتوا حتى page خطرش كلش متسلسل انا نضمنلك اذا كملت كامل الـ PDF 1 و كنت قاري جافا من 2, راح تخرج ماستر في JAVA مور ماتكمل هذا الـ PDF (اقل شيء قدر المجهود لي درتوا باش بسطتك و وضحتك كلش خطوة بخطوة و تفصيلة بتفصيلة :)).

☞ امشي في غرفتك، ونتا تقرأ فيه اكتب الكود و جرب تفهم شوية منو وحدك منبعد شوف الشرح تاع الكود و اكتبوا في ورقة باش تفهم اكثر، و تولي تحل و تفهم اي كود مكتوب بالجافا :).

☞ مور هذو الثلاثة PDFs روح تحل الـ Exams لي بقاو (شوف الحل افهموا و ادي الفكرة من الكود و عاود حلوا وحدك).

☞ موفقين ان شاء الله :).

QUICK REVIEW

ARRAYLIST

هو نوع متطور من الـ **Array** يأخذ بالعادة مجموعة من الـ **Objects** بداخله. ↗
يمتاز بديناميكية اي ليس علينا تحديد طوله او عدد العناصر التي سوف تكون بداخله. ↗
يمكن ادخال عناصر جديدة بداخله حتى بعد تعريفيه بسبب الـ **Methods** التي يوفرها لنا الـ **ArrayList** ↗
اهم الـ **ArrayList Methods** هي **size** (طول) : **add, remove , clear , isEmpty, get** ↗
للحثاء على **ArrayList** ↗

```
ArrayList <Object> arrayName = new ArrayList <Object> ();
```

QUICK REVIEW

FINAL KEYWORD

تستخدم حتى تحصر المتغير او الميثود و تمنعها من التغيير. (يصبح **CONSTANT**)
اذا قمت بصناعة دالة **final** داخل **SuperClass**, لا يمكنك التعديل عليها في **SubClass**, اي لا يمكن عمل **override** لها.

EXAM

```
1 import java.util.ArrayList;
2
3 public class Rectangle{
4
5     private double longueur, largeur;
6     private static ArrayList <Rectangle> instances = new ArrayList <Rectangle>();
7
8     public Rectangle (double lo, double la){
9         longueur=lo;
10        largeur=la;
11        instances.add(this);
12    }
13
14    public void affiche () {
15        System.out.println("Rectangle la longueur : "+ longueur + " et la largueur : "+largeur );
16    }
17
18    public static ArrayList <Rectangle> getInstances(){
19        return instances;
20    }
21
22    public double surface (){
23        return longueur*largeur;
24    }
25
26    public double getLongueur() {
27        return longueur;
28    }
29
30    public double getLargeur() {
31        return largeur;
32    }
33
34    public void setLongueur (double Lon) {
35        longueur = lon;
36    }
37    public void setLargeur (double Larg) {
38        largeur = larg;
39    }
40}
41
42
43
44
```

```
1 public class Carré extends Rectangle {
2
3     public Carré (double cote) {
4         super(cote, cote);
5     }
6
7     public void affiche () {
8         System.out.println(" Carré la longueur du cote: "+ getLargeur () );
9     }
10
11    public static ArrayList <Rectangle> getInstances(){
12
13        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();
14
15        ArrayList <Rectangle> instances = Rectangle.getInstances();
16
17        for (int i=0;i<instances.size();i++){
18
19            if(instances.get(i) instanceof Carré){
20
21                instancesCarre.add(instances.get(i));
22            }
23        }
24    }
25    return instancesCarre;
26}
27
```

EXAM

MAIN



```
1 public class Main {  
2     public static void main(String[] args) {  
3  
4         Rectangle myInstances[] = new Rectangle [4];  
5  
6         for (int i=0; i < myInstances.length ; i++){  
7  
8             if(i % 2 == 0){  
9  
10                 myInstances[i] = new Rectangle (Math.random() ,Math.random());  
11             }else {  
12                 myInstances[i] = new Carré (Math.random());  
13             }  
14         }  
15         System.out.println("Les Carrés: "+Carré.getInstances().size());  
16         System.out.println("Les Rectangles: "+Rectangle.getInstances().size());  
17     }  
18 }  
19 }
```

EXAM

CODE EXPLANATION

قمنا بانشاء **класс** **Rectangle** الذي يحتوي على **متغيرين** **largeur** و **longueur** من **Rectangle instance** الذي سوف يحتوي بداخله على عدد **من** **ArrayList instances** قمنا بانشاء ايضا **ArrayList instances** في الـ 2 PDF (شرحنا في الـ 2 PDF) واش هو الـ **instance** ارجع ليه)

```
● ● ●
1 public class Rectangle{
2
3     private double longueur, largeur;
4     private static ArrayList <Rectangle> instances = new ArrayList <Rectangle>();
5
6     public Rectangle (double lo, double la){
7         longueur = lo;
8         largeur = la;
9         instances.add(this);
10    }
11
12    public void affiche () {
13        System.out.println("Rectangle la longueur : "+ longueur + " et la largueur : "+largeur );
14    }
15
16    public static ArrayList <Rectangle> getInstances(){
17        return instances;
18    }
}
```

EXAM

CODE EXPLANATION

قمنا بانشاء **Constructor Rectangle** ➔

نريد حفظ جميع الـ **instances** التي نصنعها في الـ **ArrayList instances** و قلنا في الدروس السابقة ان **عند صناعة instance**

لكلas ما دائما ينفذ الـ Constructor الخاص بالكلas مباشرة لذلك قمنا بوضع الـ **(Object Object)** داخل **instances.add(this)**

Construtor الـ

و **this** تشير الى هذا الـ **instance** الذي صنعته, اي ضف

هذا الـ **instance** الى داخل الـ **ArrayList instances**

```
1 public class Rectangle{  
2  
3     private double longueur, largeur;  
4     private static ArrayList <Rectangle> instances = new ArrayList <Rectangle>();  
5  
6     public Rectangle (double lo, double la){  
7         longueur = lo;  
8         largeur = la;  
9         instances.add(this);  
10    }  
11  
12    public void affiche () {  
13        System.out.println("Rectangle la longueur : "+ longueur + " et la largueur : "+largeur );  
14    }  
15  
16    public static ArrayList <Rectangle> getInstances(){  
17        return instances;  
18    }  
}
```

EXAM

CODE EXPLANATION

👉 قمنا بإنشاء الدالة **affiche** التي تقوم بالطباعة.

👉 قمنا بإنشاء الدالة **getInstances** التي تقوم بارجاع ال **ArrayList** المدخل بـ **.Rectangle instances**

```
 1 public class Rectangle{  
 2  
 3     private double longueur, largeur;  
 4     private static ArrayList <Rectangle> instances = new ArrayList <Rectangle>();  
 5  
 6     public Rectangle (double lo, double la){  
 7         longueur = lo;  
 8         largeur = la;  
 9         instances.add(this);  
10    }  
11  
12    public void affiche () {  
13        System.out.println("Rectangle la longueur : "+ longueur + " et la largueur : "+largeur );  
14    }  
15  
16    public static ArrayList <Rectangle> getInstances(){  
17        return instances;  
18    }  
}
```

EXAM

CODE EXPLANATION

→ قمنا بإنشاء الدالة **surface** التي تقوم بحساب المساحة.
→ قمنا بإنشاء الدوال **Setter g Getter** للمتغيرات.



```
1  public double surface (){
2      return longueur*largeur;
3  }
4
5  public double getLongueur() {
6      return longueur;
7  }
8
9  public double getLargeur() {
10     return largeur;
11 }
12
13 public void setLongueur (double lon) {
14     longueur = lon;
15 }
16 public void setLargeur (double larg) {
17     largeur = larg;
18 }
19 }
```

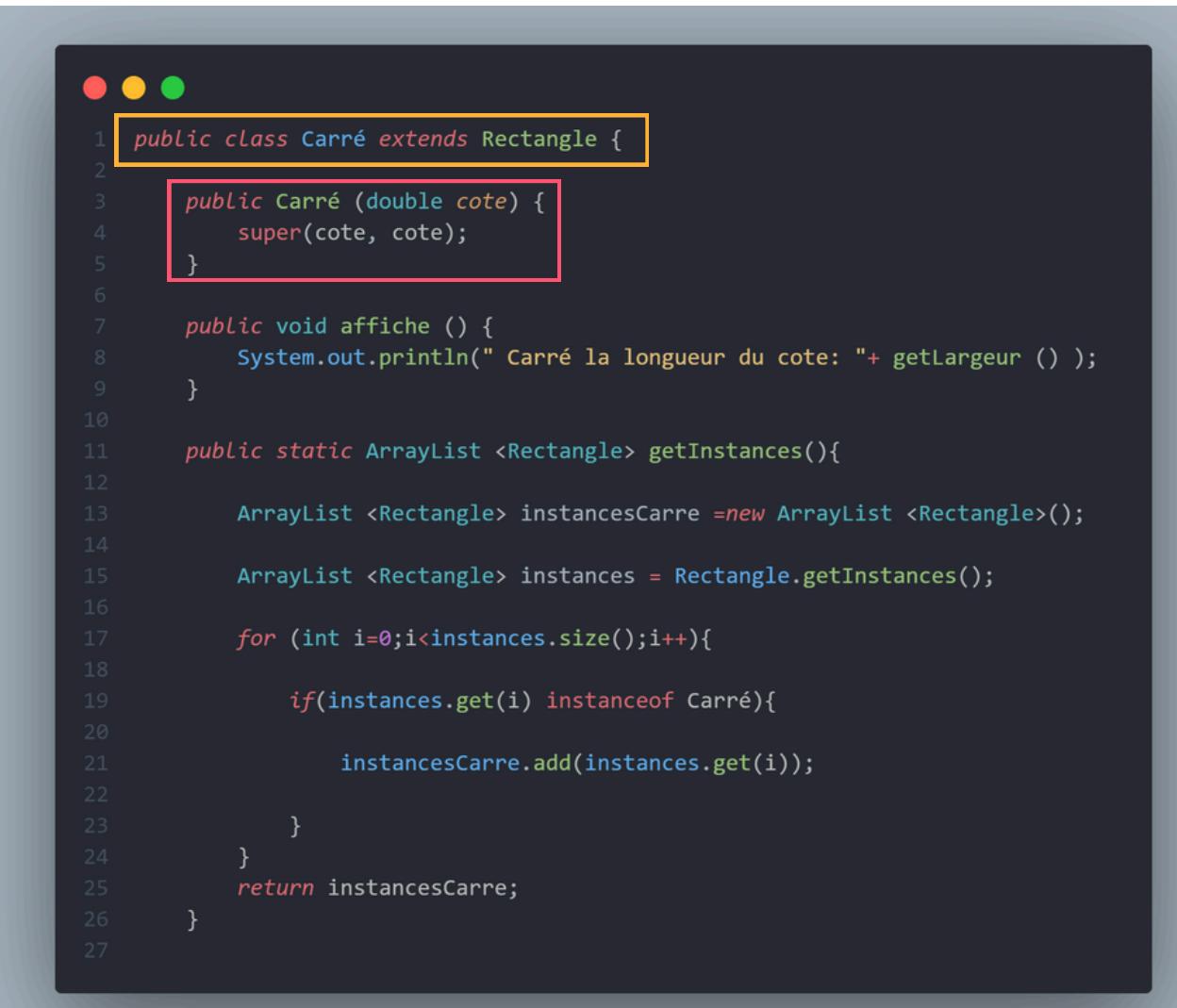
EXAM

CODE EXPLANATION

→ قمنا بإنشاء الكلاس **Carre** الذي يرث جميع الـ **Methods** و المتغيرات الموجودة (public / protected) بداخل الـ **Rectangle**

class

→ قمنا بإنشاء الـ **Constructor** الخاص بـ **Carre class**



```
1 public class Carré extends Rectangle {  
2  
3     public Carré (double cote) {  
4         super(cote, cote);  
5     }  
6  
7     public void affiche () {  
8         System.out.println(" Carré la longueur du cote: "+ getLargeur () );  
9     }  
10  
11    public static ArrayList <Rectangle> getInstances(){  
12  
13        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();  
14  
15        ArrayList <Rectangle> instances = Rectangle.getInstances();  
16  
17        for (int i=0;i<instances.size();i++){  
18  
19            if(instances.get(i) instanceof Carré){  
20  
21                instancesCarre.add(instances.get(i));  
22  
23            }  
24        }  
25        return instancesCarre;  
26    }  
27}
```

EXAM

CODE EXPLANATION

عند صناعة Carre instance بمعنى اخر عند صناعة Rectangle class الخاص ب Constructor يقوم بناء ال Super(cote, cote) ➔ Constructor يتم تنفيذ ال Carre class الخاص ب Constructor لـ Carre Object يتنفيذ ال Carre class الخاص ب Constructor من اجل اضافة Super بواسطة Rectangle class هذا الجيد الى داخل ال ArrayList instances وحفظه.

```
public class Carré extends Rectangle {  
    public Carré (double cote) {  
        super(cote, cote);  
    }  
  
    public void affiche () {  
        System.out.println(" Carré la longueur du cote: "+ getLargeur () );  
    }  
  
    public static ArrayList <Rectangle> getInstances(){  
  
        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();  
  
        ArrayList <Rectangle> instances = Rectangle.getInstances();  
  
        for (int i=0;i<instances.size();i++){  
  
            if(instances.get(i) instanceof Carré){  
  
                instancesCarre.add(instances.get(i));  
  
            }  
        }  
        return instancesCarre;  
    }  
}
```

EXAM

CODE EXPLANATION

→ قمنا بإنشاء الدالة **getInstances** التي تقوم بارجاع ال **ArrayList** المدخل ب **.Carre instances**

```
1 public class Carré extends Rectangle {  
2  
3     public Carré (double cote) {  
4         super(cote, cote);  
5     }  
6  
7     public void affiche () {  
8         System.out.println(" Carré la longueur du cote: "+ getLargeur () );  
9     }  
10  
11    public static ArrayList <Rectangle> getInstances(){  
12  
13        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();  
14  
15        ArrayList <Rectangle> instances = Rectangle.getInstances();  
16  
17        for (int i=0;i<instances.size();i++){  
18  
19            if(instances.get(i) instanceof Carré){  
20  
21                instancesCarre.add(instances.get(i));  
22  
23            }  
24        }  
25        return instancesCarre;  
26    }  
27}
```

EXAM

CODE EXPLANATION

قمنا بإنشاء الـ **ArrayList instancesCarre** الذي سيحمل نوع الـ **Rectangle** من نوع الـ **ArrayList instances** .
قمنا بإنشاء **ArrayList instances** الموجودة في **Rectangle** و قمنا بجلب جميع الـ **Rectangle** من النوع **ArrayList instances** .

Method **Static getInstance** الـ **Rectangle ArrayList instances** بواسطة الـ **Rectangle class** التي تم إنشائهما في الـ **Rectangle class** ووضعها بداخله.

```
public class Carré extends Rectangle {  
    public Carré (double cote) {  
        super(cote, cote);  
    }  
  
    public void affiche () {  
        System.out.println(" Carré la longueur du cote: "+ getLargeur () );  
    }  
  
    public static ArrayList <Rectangle> getInstances(){  
        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();  
        ArrayList <Rectangle> instances = Rectangle.getInstances();  
  
        for (int i=0;i<instances.size();i++){  
            if(instances.get(i) instanceof Carré){  
                instancesCarre.add(instances.get(i));  
            }  
        }  
        return instancesCarre;  
    }  
}
```

EXAM

CODE EXPLANATION

هنا تكمن فائدة الـ **static Method getInstance** من نوع **static Methods / Variables** حينها **صناعة** ↗ لو لم تكون الـ **static Methods / Variables** خاص بـ **Carre** (Rectangle instance) داخل **Carre** (Rectangle instance) و كما قلنا عند

```
public class Carré extends Rectangle {  
    public Carré (double cote) {  
        super(cote, cote);  
    }  
  
    public void affiche () {  
        System.out.println(" Carré la longueur du cote: "+ getLargeur () );  
    }  
  
    public static ArrayList <Rectangle> getInstances(){  
        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();  
        ArrayList <Rectangle> instances = Rectangle.getInstances();  
  
        for (int i=0;i<instances.size();i++){  
            if(instances.get(i) instanceof Carré){  
                instancesCarre.add(instances.get(i));  
            }  
        }  
        return instancesCarre;  
    }  
}
```

صناعة سيتوجب علينا اعطاء قيمة **Rectangle instance**

Constructor (new Rectangle (1,4)) | **Constructor** لـ **Carre** (1,4) مما يؤدي الى تنفيذ الـ

الخاص بـ **Rectangle instance** وبالتالي اضافة **Rectangle** جديد داخل **Carre** (Rectangle instance) مصنوع من طرف **ArrayList instances** الـ

.Main class وليس الـ **Carre class** الـ

اي كلما استخدمنا الـ **Carre** (Rectangle instance) س يتم

صناعة **Rectangle instance** جديد تلقائيا و هذا ما يسبب ارتفاع ضغط المبرمج وقلة بسبب زيادة عدد **Rectangle instance** بدون معرفة السبب : .

EXAM

CODE EXPLANATION

.ArrayList instances من أجل فصل ال Carre instances عن ال Rectangle instances انشائنا ال For loop ↗
.ArrayList instances من أجل معرفة طول او عدد ال instances.size() ↗

```
1 public class Carré extends Rectangle {  
2  
3     public Carré (double cote) {  
4         super(cote, cote);  
5     }  
6  
7     public void affiche () {  
8         System.out.println(" Carré la longueur du cote: "+ getLargeur () );  
9     }  
10  
11    public static ArrayList <Rectangle> getInstances(){  
12  
13        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();  
14  
15        ArrayList <Rectangle> instances = Rectangle.getInstances();  
16  
17        for (int i=0;i<instances.size();i++){  
18  
19            if(instances.get(i) instanceof Carré){  
20  
21                instancesCarre.add(instances.get(i));  
22  
23            }  
24        }  
25        return instancesCarre;  
26    }  
27}
```

EXAM

CODE EXPLANATION

.index i ذو ال **Object (instance)** تقوم بجلب ال **instance.get(i)** ➔
هي **كلمة محفوظة في جافا** تقوم بالتحقق من نوعية ال **instanceof** ➔
ما اذا كان تابعاً لل **Class** او ال **Object** او ال **instance**.

ذو **instance** يقوم بالتحقق من ال **instance.get(i)** **instanceof Carré** ➔
ال **i** الذي يدخل ال **ArrayList instances** ام لا.

```
public class Carré extends Rectangle {  
    public Carré (double cote) {  
        super(cote, cote);  
    }  
  
    public void affiche () {  
        System.out.println(" Carré la longueur du cote: "+ getLargeur () );  
    }  
  
    public static ArrayList <Rectangle> getInstances(){  
        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();  
        ArrayList <Rectangle> instances = Rectangle.getInstances();  
        for (int i=0;i<instances.size();i++){  
            if(instances.get(i) instanceof Carré){  
                instancesCarre.add(instances.get(i));  
            }  
        }  
        return instancesCarre;  
    }  
}
```

EXAM

CODE EXPLANATION

اذا كان **Carre class** فاننا نقوم تابع **ArrayList instances** داخل ال **index i** الذي يدخل **instanceJl** ↗
ArrayList instancesCarreJl

```
public class Carré extends Rectangle {  
    public Carré (double cote) {  
        super(cote, cote);  
    }  
  
    public void affiche () {  
        System.out.println(" Carré la longueur du cote: "+ getLargeur () );  
    }  
  
    public static ArrayList <Rectangle> getInstances(){  
        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();  
        ArrayList <Rectangle> instances = Rectangle.getInstances();  
        for (int i=0;i<instances.size();i++){  
            if(instances.get(i) instanceof Carré){  
                instancesCarre.add(instances.get(i));  
            }  
        }  
        return instancesCarre;  
    }  
}
```

EXAM

CODE EXPLANATION

فی الاخير نقوم بارجاع ال **ArrayList instancesCarre** لان ال **Method .get** من نوع **ArrayList** ↗

```
public class Carré extends Rectangle {  
    public Carré (double cote) {  
        super(cote, cote);  
    }  
  
    public void affiche () {  
        System.out.println(" Carré la longueur du cote: "+ getLargeur () );  
    }  
  
    public static ArrayList <Rectangle> getInstances(){  
        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();  
        ArrayList <Rectangle> instances = Rectangle.getInstances();  
        for (int i=0;i<instances.size();i++){  
            if(instances.get(i) instanceof Carré){  
                instancesCarre.add(instances.get(i));  
            }  
        }  
        return instancesCarre;  
    }  
}
```

EXAM

CODE EXPLANATION

قمنا بانشاء **instances** اي سيحمل بداخل ال **Rectangle** ذو النوع **Array myInstances** وحددنا طوله ب 4 .
قمنا بانشاء **For loop** التي تقوم التحقق ما اذا كان ال فردي ام زوجي.



```
public class Main {
    public static void main(String[] args) {
        Rectangle myInstances[] = new Rectangle[4];
        for (int i=0; i < myInstances.length ; i++){
            if(i % 2 == 0){
                myInstances[i] = new Rectangle(Math.random() ,Math.random());
            }else {
                myInstances[i] = new Carré(Math.random());
            }
        }
        System.out.println("Les Carrés: "+Carré.getInstances().size());
        System.out.println("Les Rectangles: "+Rectangle.getInstances().size());
    }
}
```

EXAM

CODE EXPLANATION

في حالة ما كان الـ **Carre instance** اما في حالة **الفردي** يقوم بصناعة الـ **Rectangle instance** ↗



```
public class Main {
    public static void main(String[] args) {
        Rectangle myInstances[] = new Rectangle[4];
        for (int i=0; i < myInstances.length ; i++){
            if(i % 2 == 0){
                myInstances[i] = new Rectangle(Math.random() ,Math.random());
            }else {
                myInstances[i] = new Carré(Math.random());
            }
        }
        System.out.println("Les Carrés: "+Carré.getInstances().size());
        System.out.println("Les Rectangles: "+Rectangle.getInstances().size());
    }
}
```

EXAM

CODE EXPLANATION

هنا ايضا تكمن فائدة ال static من نوع Method getInstances ، كما قلنا لو لم تكون ال static سيتوجب علينا instance و سيتم اضافة هذا الى Constructor سينفذ ال instance للستخدامها و عند مثلا (instance) Object مثابة instance و سيعمل على دخول الى داخل

instances 6 وفي هذه الحالة سيكون لدينا ArrayList instances وليس 4 .(Carre instance 1 و Rectangle instance 1)

```
● ● ●
1 public class Main {
2     public static void main(String[] args) {
3
4         Rectangle myInstances[] = new Rectangle [4];
5
6         for (int i=0; i < myInstances.length ; i++){
7
8             if(i % 2 == 0){
9
10                 myInstances[i] = new Rectangle (Math.random() ,Math.random());
11
12             }else {
13
14                 myInstances[i] = new Carré (Math.random());
15
16             }
17         }
18         System.out.println("Les Carrés: " + Carré.getInstances().size());
19         System.out.println("Les Rectangles: " + Rectangle.getInstances().size());
20
21     }
22 }
```

EXAM

CODE EXPLANATION

سيتم طباعة عدد الـ **instances** الموجودة داخل **ArrayList instancesCarre** و داخل **ArrayList instances** ↗

```
● ● ●
1 public class Main {
2     public static void main(String[] args) {
3
4         Rectangle myInstances[] = new Rectangle [4];
5
6         for (int i=0; i < myInstances.length ; i++){
7
8             if(i % 2 == 0){
9
10                 myInstances[i] = new Rectangle (Math.random() ,Math.random());
11
12             }else {
13
14                 myInstances[i] = new Carré (Math.random());
15
16             }
17         }
18         System.out.println("Les Carrés: " + Carré.getInstances().size());
19         System.out.println("Les Rectangles: " + Rectangle.getInstances().size());
20
21     }
22 }
```

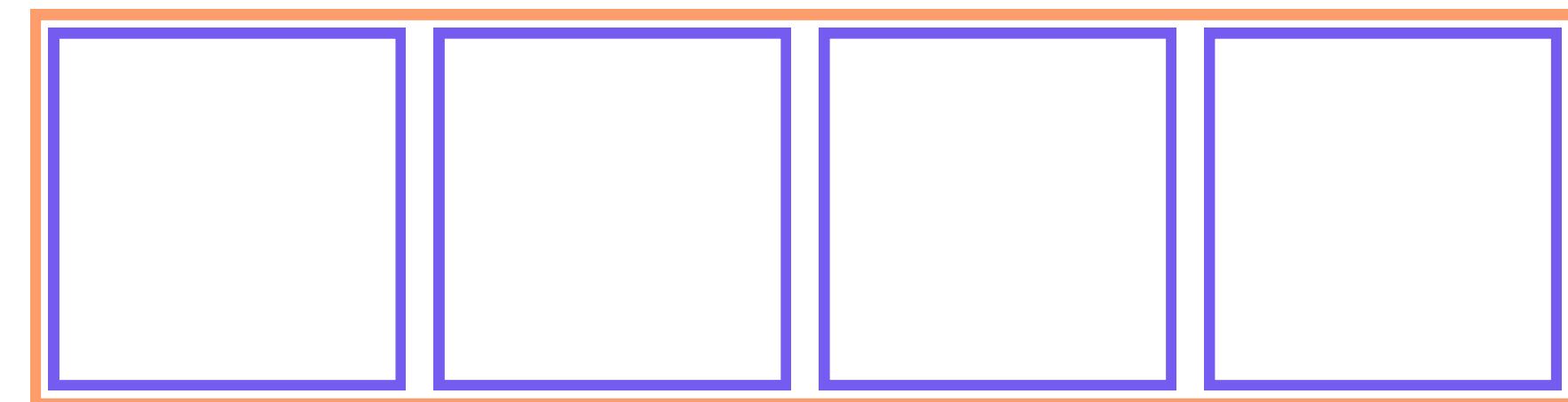
EXAM

BEHIND THE SCENES

قمنا بإنشاء `Array myInstances` ذو النوع `Rectangle` ذو طوله بـ 4.  اي سيحمل بداخل ال `instances` و حددنا طوله بـ 4.

`Rectangle myInstances [] = new Rectangle [4];`

`myInstances`



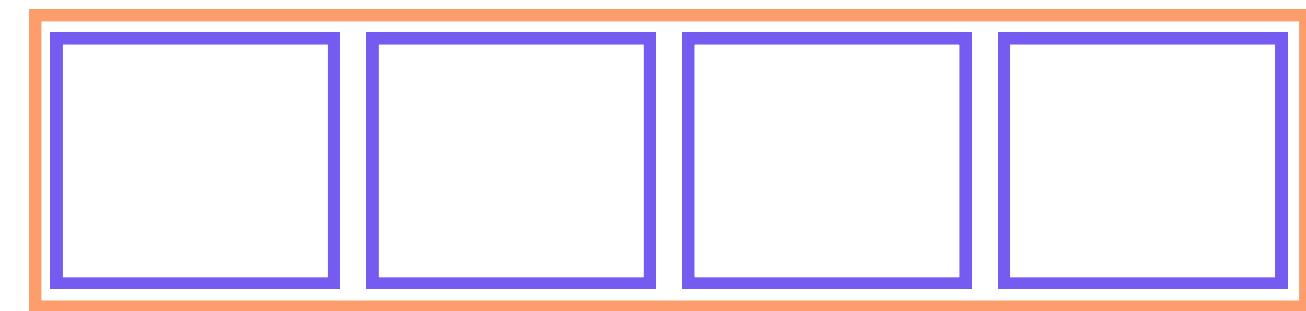
EXAM

BEHIND THE SCENES

طول الـ **Array myInstances** هو 4 اذا سيتم تنفيذ الكود الذي بداخل الـ **For loop** اربع مرات.

```
for (int i = 0; i < myInstances.length ; i++){  
    if(i % 2 == 0){  
        myInstances[ i ]= new Rectangle (2 ,5);  
    }else {  
        myInstances[ i ] = new Carré (4);  
    }  
}
```

myInstances



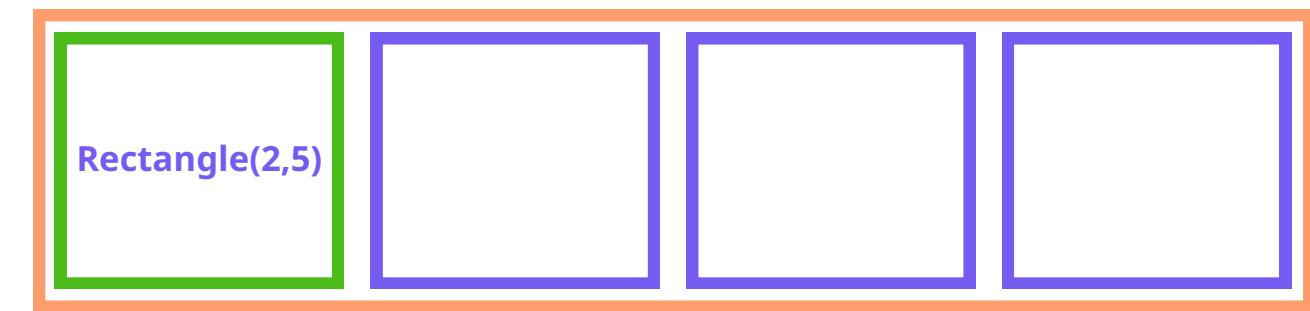
EXAM

BEHIND THE SCENES

EXECUTION OF FOR LOOP 01 : **i = 0**

```
for (int i = 0; i < myInstances.length ; i++){  
    if(i % 2 == 0){  
        myInstances[ 0 ]= new Rectangle (2 ,5);  
    }else {  
        myInstances[ i ] = new Carré (4);  
    }  
}
```

myInstances



EXAM

BEHIND THE SCENES

عند صناعة الى **instance** مما يؤدي الى اضافة هذا الـ **Constructor** الخاص بـ **Rectangle** سيتم تنفيذ الـ **Rectangle instance**  الموجود بالكلاس **ArrayList instances**.

```
myInstances[ 0 ]= new Rectangle (2 ,5);
```

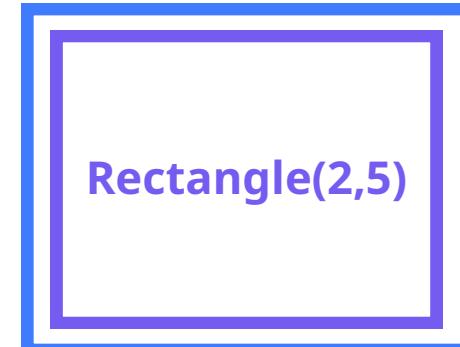


```
public class Rectangle{  
    private static ArrayList <Rectangle> instances = new ArrayList <Rectangle>();  
    public Rectangle (2, 5){  
        longueur = 2;  
        largeur = 5;  
        instances.add(this);  
    }
```

instances



instances



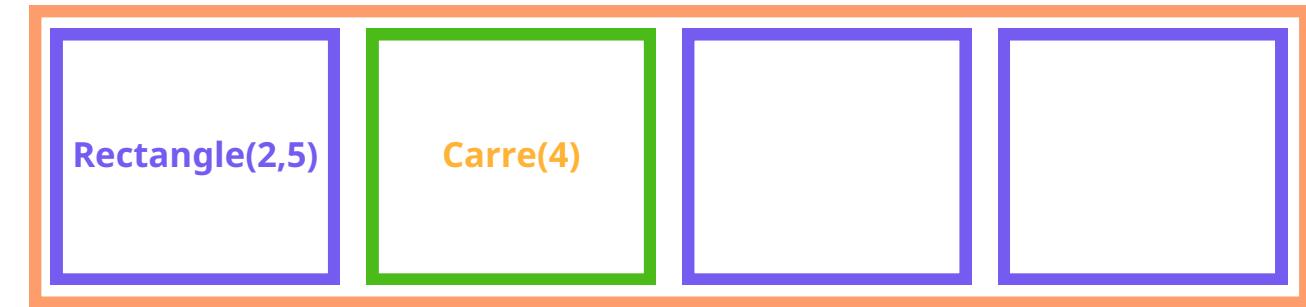
EXAM

BEHIND THE SCENES

EXECUTION OF FOR LOOP 02 : **i=1**

```
for (int i = 0; i < myInstances.length ; i++){  
    if(i % 2 == 0){  
        myInstances[ i ]= new Rectangle (2 ,5);  
    }  
    else {  
        myInstances[ 1 ] = new Carré (4);  
    }  
}
```

myInstances



EXAM

BEHIND THE SCENES

عند صناعة ↗ **Carre** س يتم تنفيذ ال **Constructor** الخاص بـ **Carre** مما يؤدي الى تنفيذ ايضاً ال **Constructor** الخاص **Caree instance** الموجود بالكلasse **ArrayList instances** مما يؤدي الى اضافة هذا ال **instance** الى **Super(4,4)** بسبب **Rectangle** .

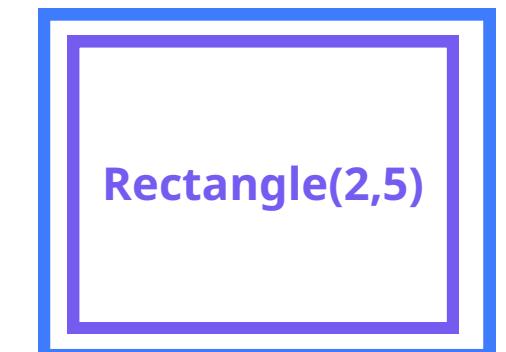
```
myInstances[ 1 ]= new Carre(4);
```



```
public class Carré extends Rectangle {
```

```
public Carré (4) {  
super(4, 4);  
}
```

instances



EXAM

BEHIND THE SCENES

يتم اضافة هذا الـ **instance** الى **ArrayList instances** الموجود بالكلasse **Rectangle** ↗

```
myInstances[ 1 ]= new Carre(4);
```

```
public class Rectangle{  
    private static ArrayList <Rectangle> instances = new ArrayList <Rectangle>();  
    public Rectangle (4, 4){  
        longueur = 4;  
        largeur = 4;  
        instances.add(this);  
    }  
}
```

instances



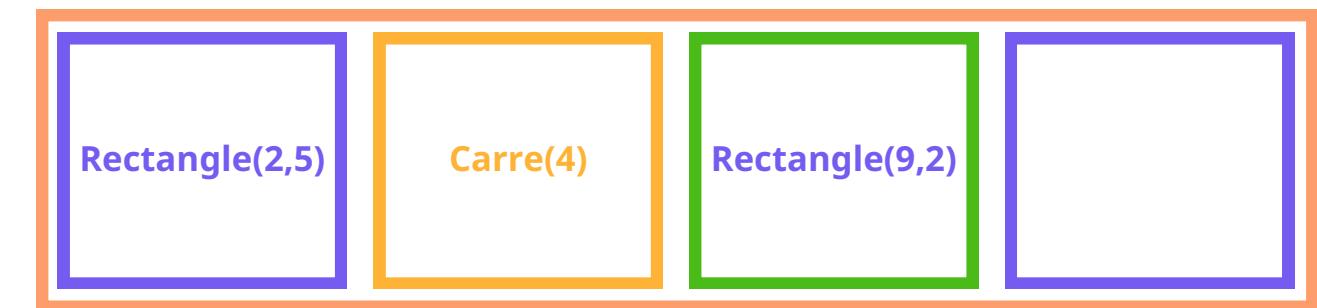
EXAM

BEHIND THE SCENES

EXECUTION OF FOR LOOP 03 : **i = 2**

```
for (int i = 0; i < myInstances.length ; i++){  
    if(i % 2 == 0){  
        myInstances[ 2 ]= new Rectangle (9 ,2);  
    }else {  
        myInstances[ i ] = new Carré (4);  
    }  
}
```

myInstances



EXAM

BEHIND THE SCENES

عند صناعة الى **instance** س يتم تنفيذ ال **Constructor** الخاص بـ **Rectangle** مما يؤدي الى اضافة هذا ال **Rectangle instance** الى **ArrayList instances** الموجود بالكلasse **ArrayList instances**.

```
myInstances[ 2 ]= new Rectangle (9 ,2);
```



```
public class Rectangle{  
    private static ArrayList <Rectangle> instances = new ArrayList <Rectangle>();  
    public Rectangle (9, 2){  
        longueur = 9;  
        largeur = 2;  
        instances.add(this);  
    }  
}
```

instances



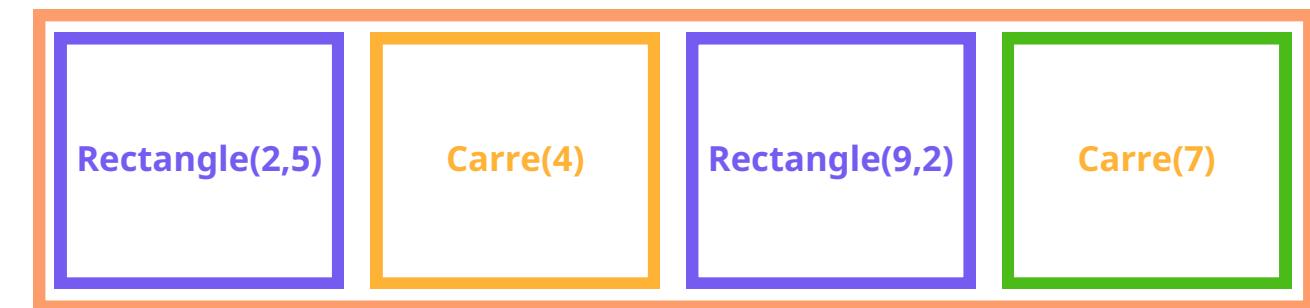
EXAM

BEHIND THE SCENES

EXECUTION OF FOR LOOP 04 : **i = 3**

```
for (int i = 0; i < myInstances.length ; i++){  
    if(i % 2 == 0){  
        myInstances[ i ]= new Rectangle (9 ,2);  
    }  
    else {  
        myInstances[ 3 ] = new Carré (7);  
    }  
}
```

myInstances



EXAM

BEHIND THE SCENES

عند صناعة ↗ **Carre instance** سيتم تنفيذ الـ **Constructor** الخاص بـ **Carre** مما يؤدي الى تنفيذ ايضاً الـ **Constructor** الخاص بالـ **ArrayList instances** الموجود بالكلasse **Super(7,7)** بسبب **Rectangle**.

```
myInstances[ 3 ]= new Carre(7);
```



```
public class Carré extends Rectangle {
```

```
public Carré (7) {  
    super(7, 7);  
}
```

instances



EXAM

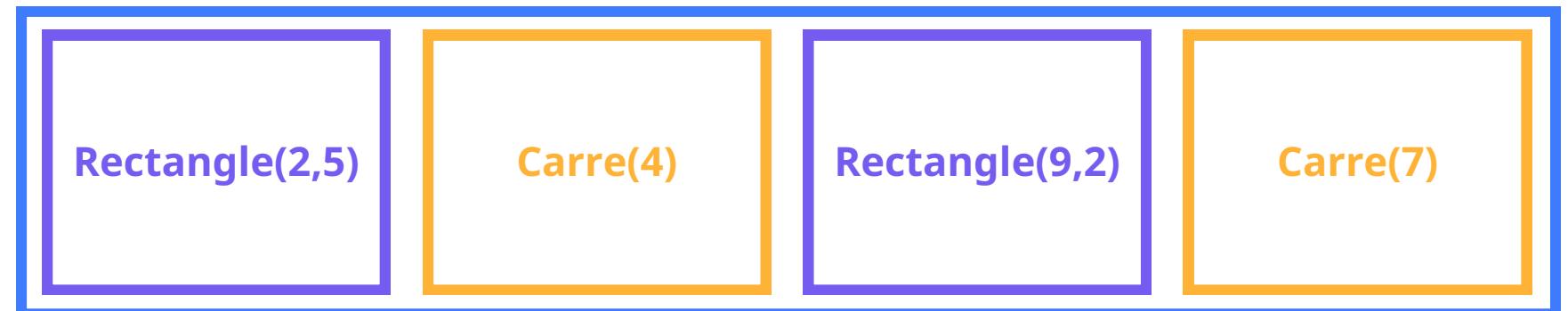
BEHIND THE SCENES

يتم اضافة هذا الـ **instance** الى **ArrayList instances** الموجود بالكلasse **Rectangle** ↗

```
myInstances[ 1 ]= new Carre(7);
```

```
public class Rectangle{  
    private static ArrayList <Rectangle> instances = new ArrayList <Rectangle>();  
    public Rectangle (7, 7){  
        longueur = 7;  
        largeur = 7;  
        instances.add(this);  
    }  
}
```

instances

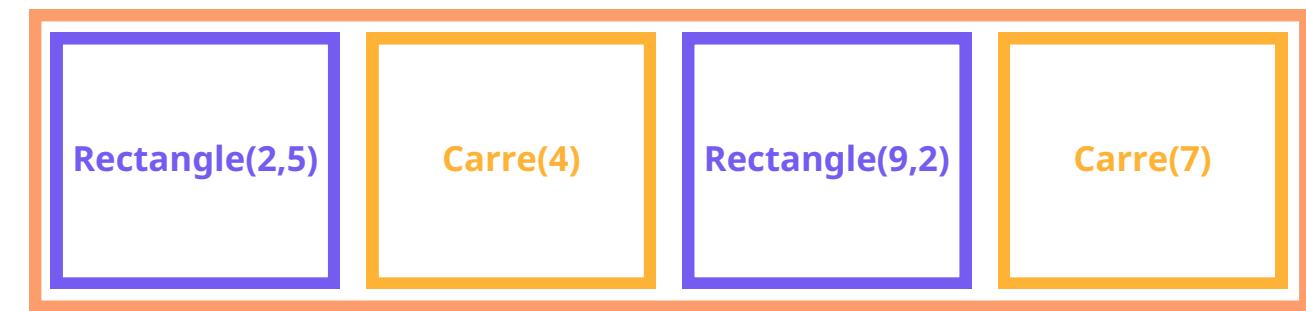


EXAM

BEHIND THE SCENES

EXECUTION OF FOR LOOP 05 : **i = 4 ? impossible -> 4 < 4**

```
for (int i = 0; i < myInstances.length ; i++){  
    if(i % 2 == 0){  
        myInstances[ i ]= new Rectangle (2 ,5);           myInstances  
    }else {  
        myInstances[ 3 ] = new Carré (4);  
    }  
}
```



EXAM

BEHIND THE SCENES

في نهاية الـ For loop ↗
instances الـ ArrayList instances الذي يحتويان على 4 myInstances

myInstances



instances



EXAM

BEHIND THE SCENES

```
System.out.println("Les Carrés: "+Carré.getInstances().size());  
  
class Carré {  
  
    public static ArrayList <Rectangle> getInstances(){  
  
        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();  
  
        ArrayList <Rectangle> instances = Rectangle.getInstances();  
  
        for (int i=0;i<instances.size();i++){  
  
            if(instances.get(i) instanceof Carré){  
  
                instancesCarre.add(instances.get(i));  
  
            }  
  
        }  
  
        return instancesCarre;  
    }
```

سيتم مساعدة الميثود **getInstances** الخاصة بكلاس **Carré** ↗

ArrayList **instancesCarre** صنع ↗

instancesCarre



EXAM

BEHIND THE SCENES

```
System.out.println("Les Carrés: "+Carré.getInstances().size());
```

```
class Carré {
```

```
    public static ArrayList <Rectangle> getInstances(){
```

```
        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();
```

```
        ArrayList <Rectangle> instances = Rectangle.getInstances();
```

```
        for (int i=0;i<instances.size();i++){
```

```
            if(instances.get(i) instanceof Carré){
```

```
                instancesCarre.add(instances.get(i));
```

```
}
```

```
        return instancesCarre;
```

```
}
```

سيتم صنع الـ **ArrayList instance** ووضع داخله كل الـ **Rectangle** الموجودين داخل الكلاس **ArrayList instances**

instances



instancesCarre

EXAM

BEHIND THE SCENES

طول ال **ArrayList instances** هو 4 اذا سيتم تنفيذ الكود الذي بداخل ال **For loop** اربع مرات.

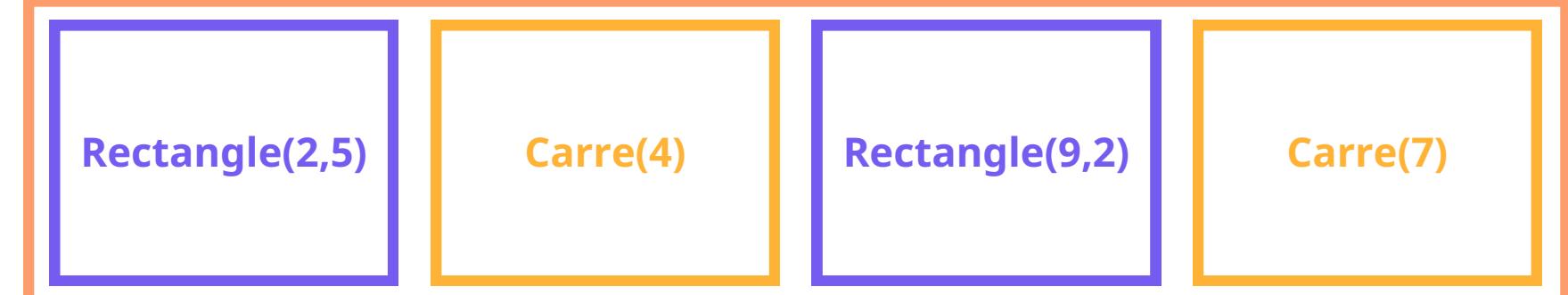
```
for (int i=0;i<instances.size();i++){
```

```
    if(instances.get(i) instanceof Carré){
```

```
        instancesCarre.add(instances.get(i));
```

```
}
```

instances



instancesCarre

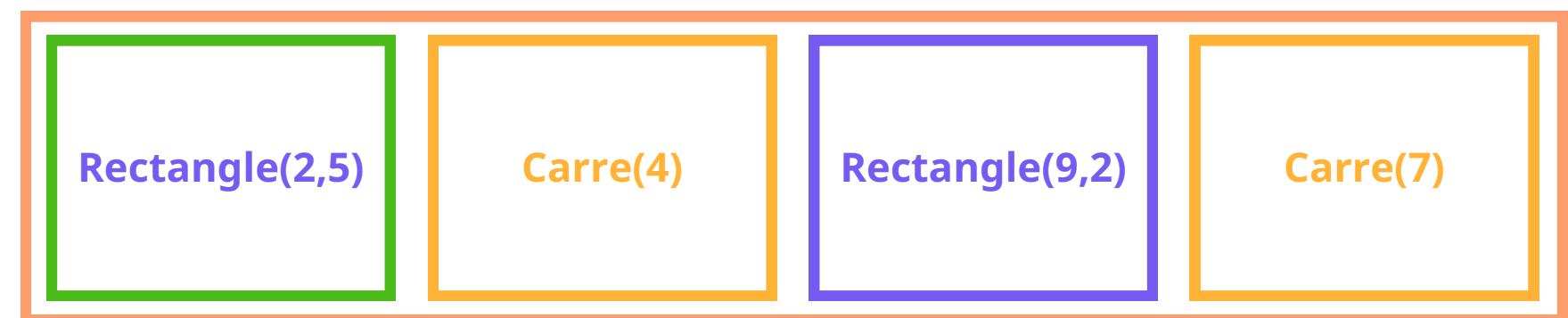
EXAM

BEHIND THE SCENES

EXECUTION OF FOR LOOP 01 : **i = 0**

```
for (int i=0;i<instances.size();i++){  
    if(instances.get(0) instanceof Carré){  
        instancesCarre.add(instances.get(i));  
    }  
}
```

instances



لا ينتمي الى الكلاس Carré

instancesCarre



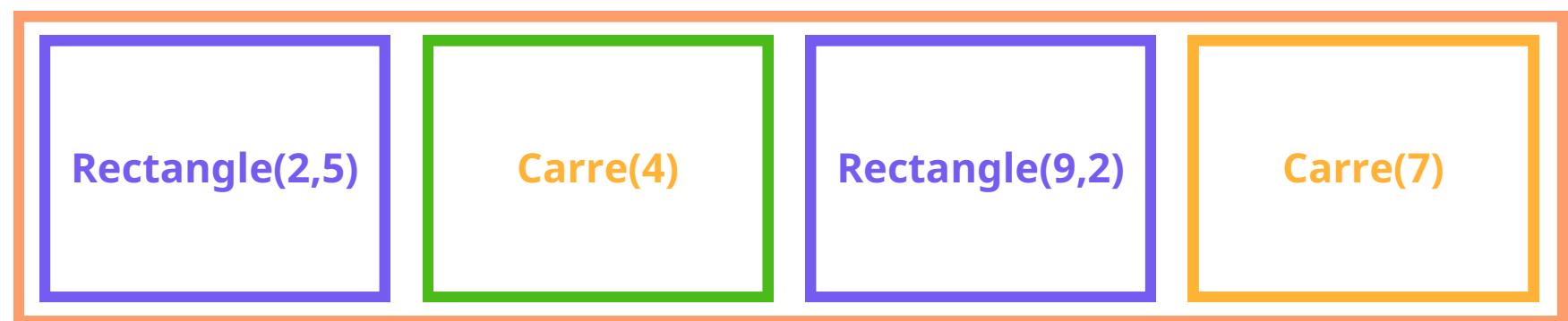
EXAM

BEHIND THE SCENES

EXECUTION OF FOR LOOP 02 : **i=1**

```
for (int i=0;i<instances.size();i++){  
    if(instances.get(1) instanceof Carré){  
        instancesCarre.add(instances.get(1));  
    }  
}
```

instances



ينتمي الى الكلاس Carré

instancesCarre



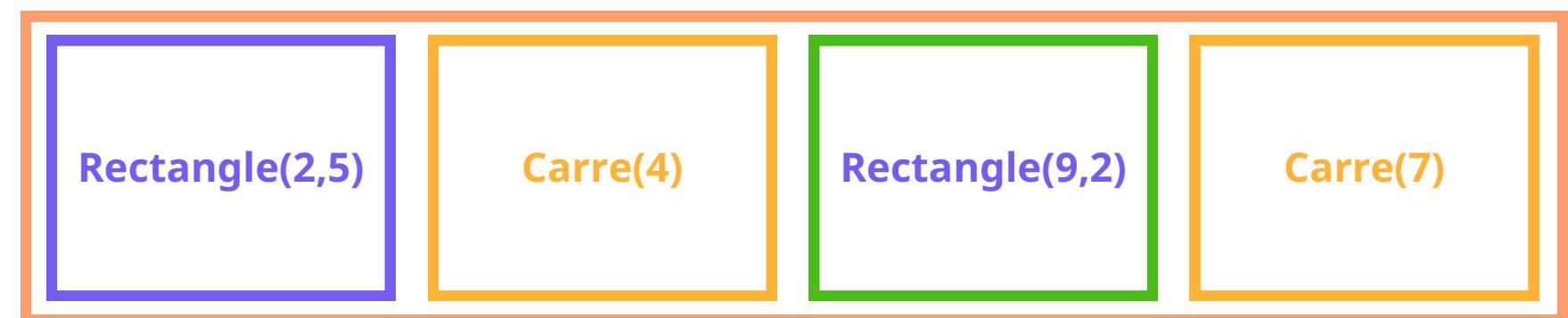
EXAM

BEHIND THE SCENES

EXECUTION OF FOR LOOP 03 : **i = 2**

```
for (int i=0;i<instances.size();i++){  
    if(instances.get(2) instanceof Carré){  
        instancesCarre.add(instances.get(i));  
    }  
}
```

instances



لا ينتمي الى الكلاس Carré

instancesCarre



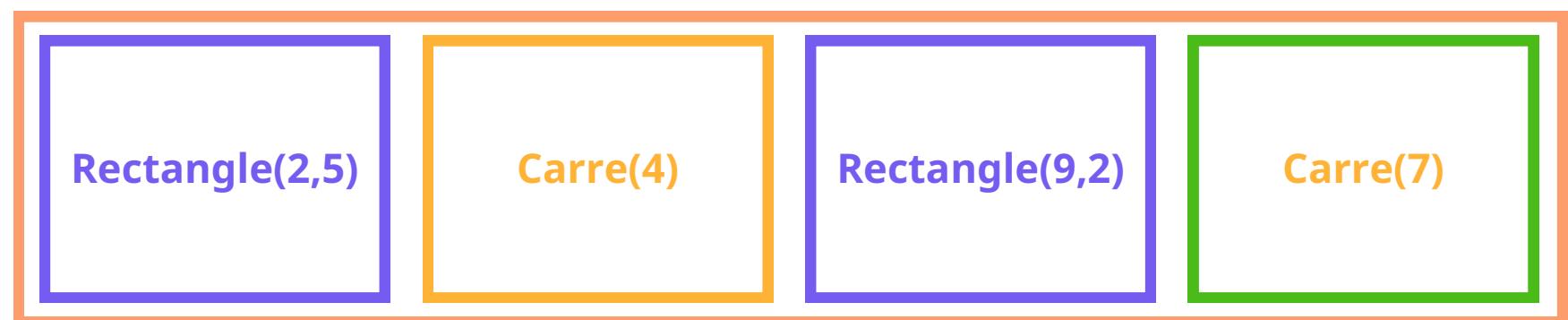
EXAM

BEHIND THE SCENES

EXECUTION OF FOR LOOP 04 : **i = 3**

```
for (int i=0;i<instances.size();i++){  
    if(instances.get(1) instanceof Carré){  
        instancesCarre.add(instances.get(1));  
    }  
}
```

instances



يُنتمي الى الكلاس Carré

instancesCarre



EXAM

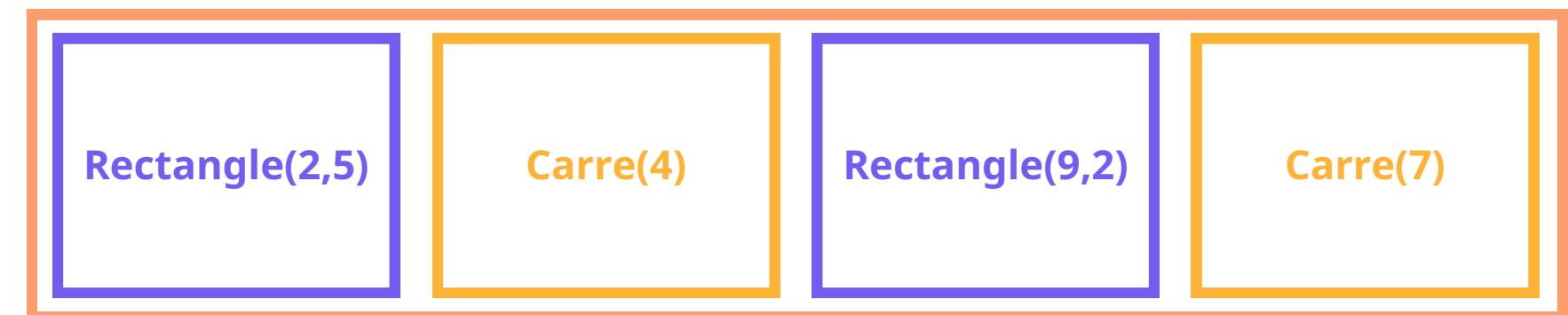
BEHIND THE SCENES

EXECUTION OF FOR LOOP 05 : **i = 4 impossible -> 4 < 4!**

```
for (int i=0;i<instances.size();i++){
```

instances

```
if(instances.get(1) instanceof Carré){
```



```
instancesCarre.add(instances.get(1));
```

instancesCarre

```
}
```

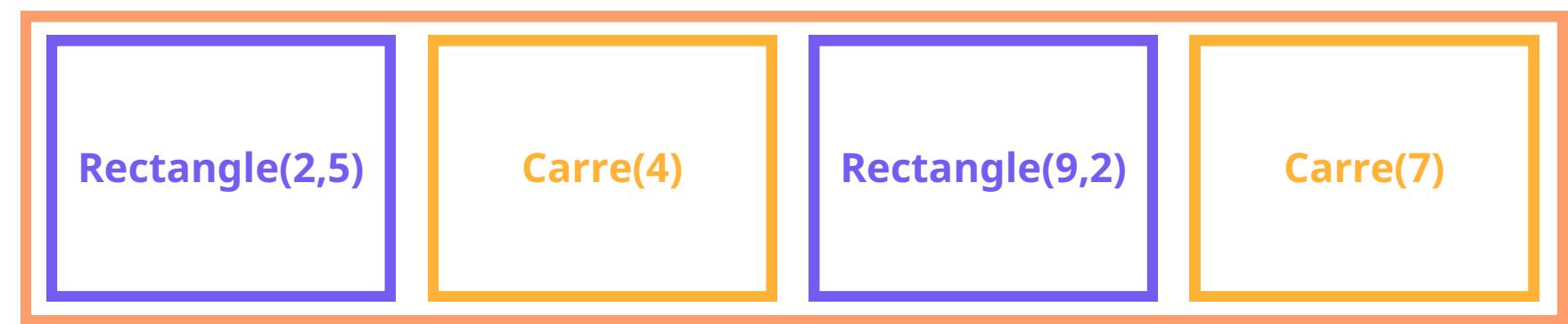


EXAM

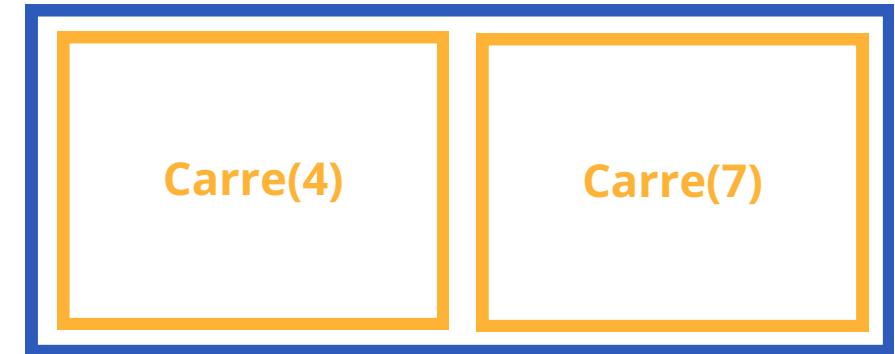
BEHIND THE SCENES

.instances الذي يحتوي على 2 ArrayList instancesCarre يتكون لنا ال For loop في نهاية ال 

instances



instancesCarre



EXAM

BEHIND THE SCENES

.**سيتم ارجاع** من **instanceCarre** ↗

```
class Carre {  
    public static ArrayList <Rectangle> getInstances(){  
        ArrayList <Rectangle> instancesCarre =new ArrayList <Rectangle>();  
        ArrayList <Rectangle> instances = Rectangle.getInstances();  
        for (int i=0;i<instances.size();i++){  
            if(instances.get(i) instanceof Carré){  
                instancesCarre.add(instances.get(i));  
            }  
        }  
        return instancesCarre;  
    }  
}
```

instances

instancesCarre

instances

instancesCarre

EXAM

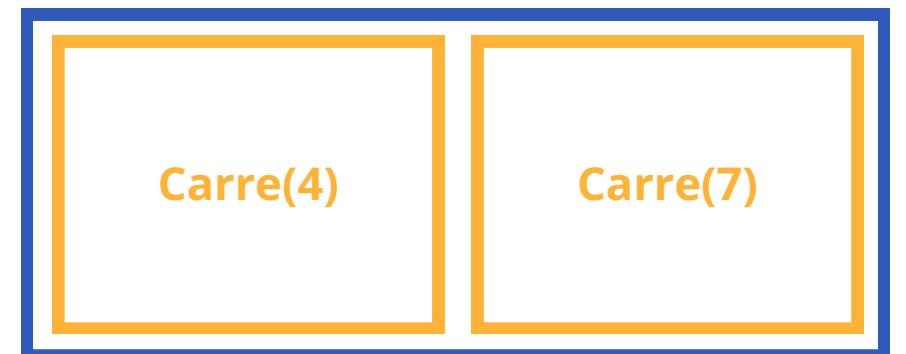
BEHIND THE SCENES

.2 گە instanceCarre ئىل جۇڭ

```
System.out.println("Les Carrés: "+Carré.getInstances().size());
```

Les Carres : 2

instancesCarre



EXAM

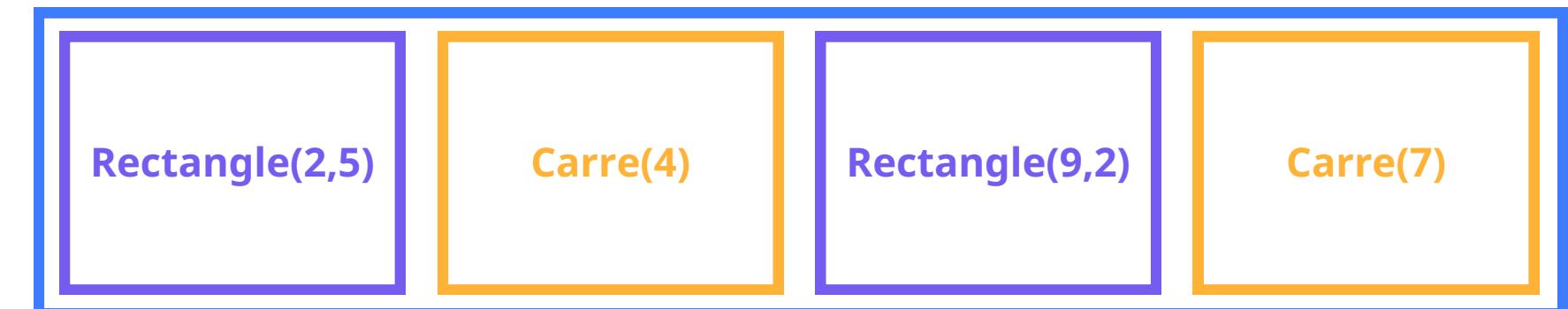
BEHIND THE SCENES

. 4 گە instances لى جول ظۇل ➔

```
System.out.println("Les Rectangles: "+Rectangle.getInstances().size());
```

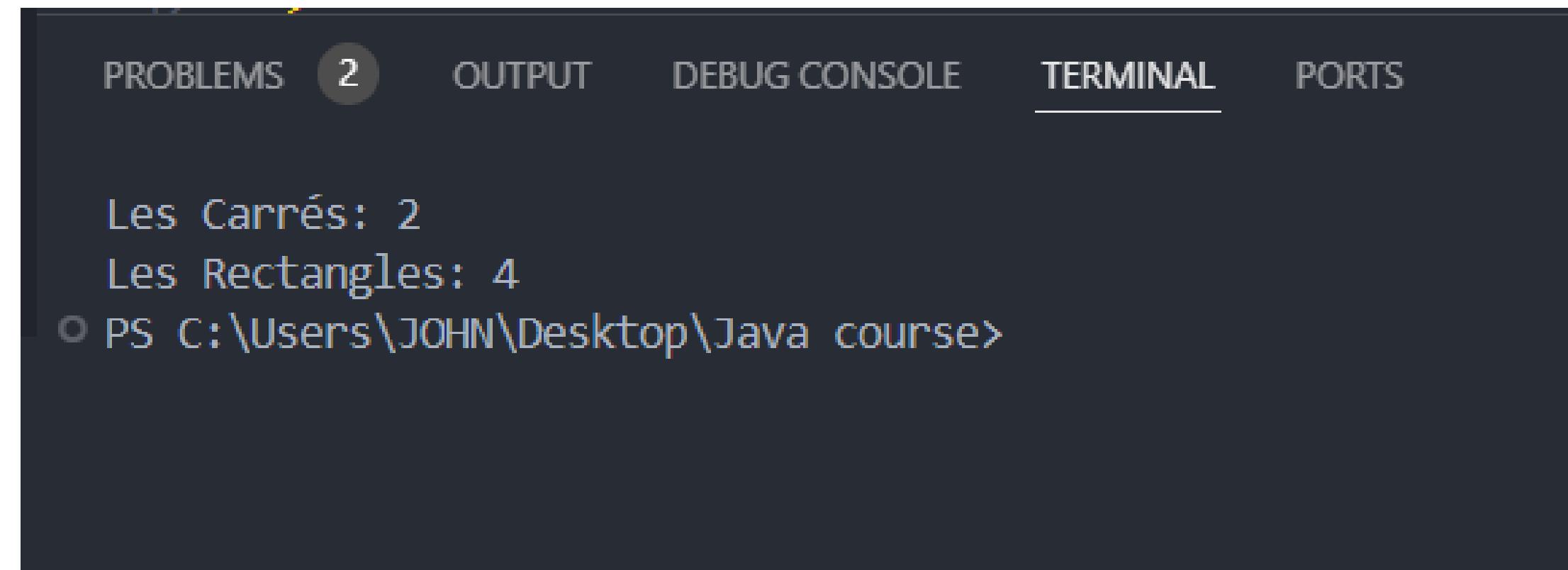
Les Rectangles : 4

instances



EXAM

OUTPUT



A screenshot of a terminal window with a dark background. At the top, there are tabs: PROBLEMS (with a red badge showing '2'), OUTPUT, DEBUG CONSOLE, TERMINAL (underlined), and PORTS. The terminal window displays the following text:

```
Les Carrés: 2
Les Rectangles: 4
○ PS C:\Users\JOHN\Desktop\Java course>
```

EXAM

OUTPUT

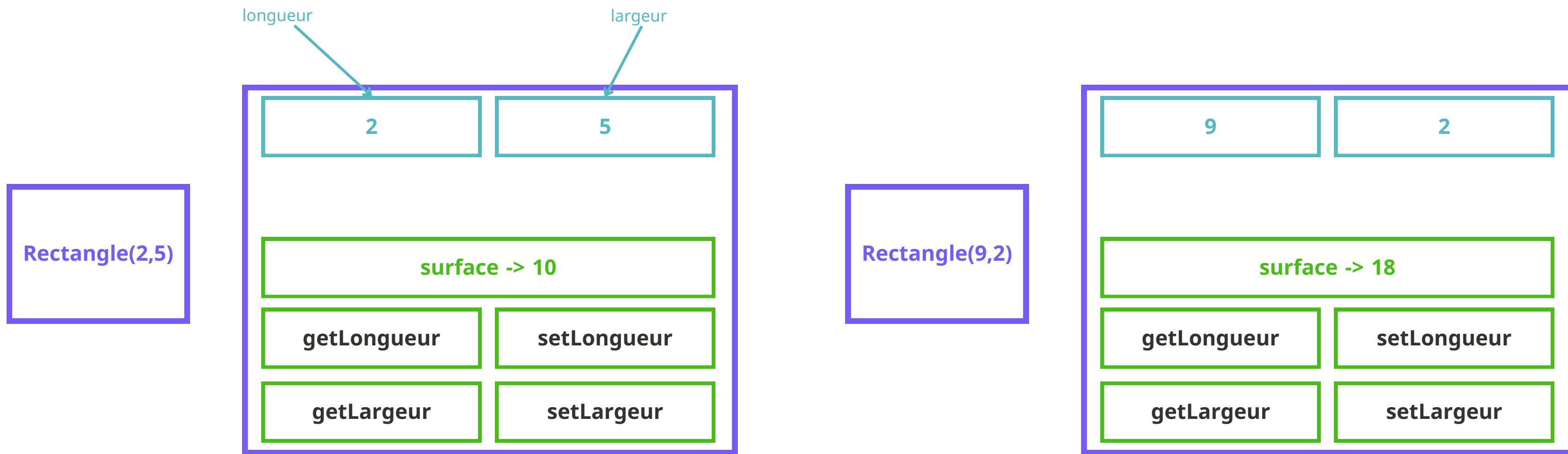


That's what I am talking
about

EXAM

ZOOM IN

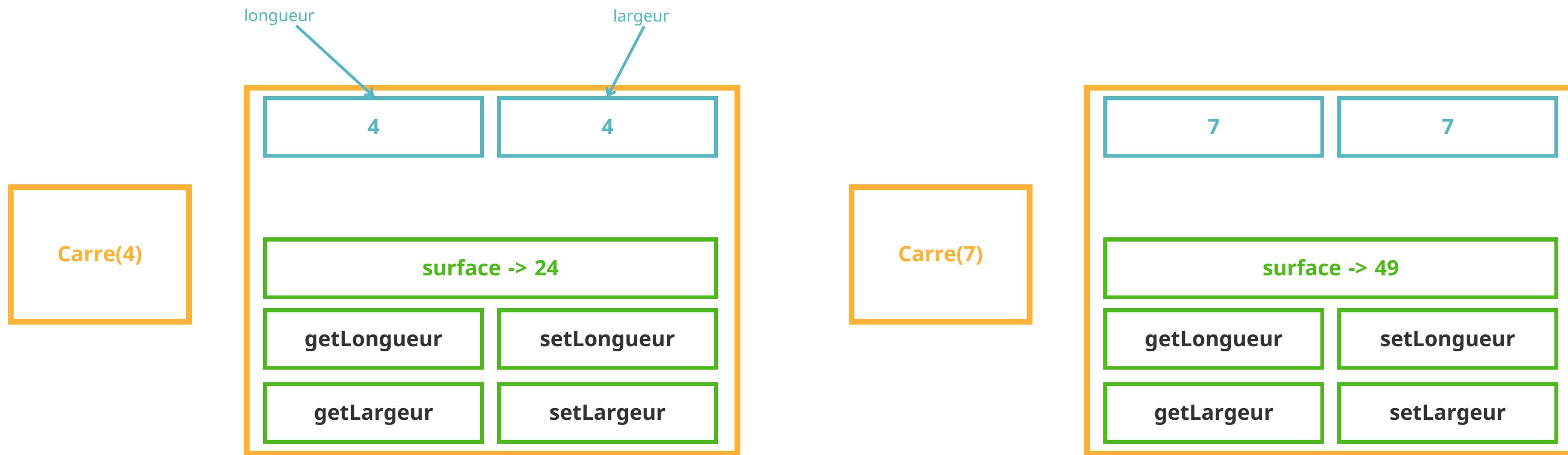
.instances نظرة عن قرب لـ 



EXAM

ZOOM IN

.instances نظرة عن قرب لـ 



EXAM

CONCLUSION

كل شيء في جافا عبارة عن **.object** ➔

THE END

ANY feedback?



MehdiBouchachi
Web Developer

SEE YOU SOON...