1.6 Exercices

1.6.1 Création et utilisation d'une classe

Réaliser une classe *Mobile* permettant de représenter un téléphone mobile. Chaque mobile sera caractérisé par sa marque, son modèle et son système (de type String) et son prix (de type double). La classe devra contenir :

- un constructeur recevant en arguments la marque, le modèle, le système et le prix d'un téléphone mobile,
- une méthode *afficher* toutes les informations du mobile,
- une méthode *revendre* effectuant une modification du prix par la valeur de son argument.

Écrire une classe *Application* utilise la classe *Mobile* pour créer le *Iphone X*, et afficher toutes les caractéristiques de ce téléphone mobile. Afficher à nouveau les caractéristiques de ce téléphone après une revente.

1.6.2 Initialisation d'un objet

Que fournit l'exécution de la classe Application ci-dessous ?

```
class Nombre
{
private int n=2;
private int k;
public Nombre(int limit)
 { n *= limit;
   n += k;
 }
public void afficher()
 { System.out.println("n="+n+ "k="+k) ;
 }
}
public class Application
{ public static void main(String args[])
  { Nombre n=new Nombre(5);
    n.afficher();
  }
}
```

1.6.3 Affectation et comparaison d'objets

Que fournit l'exécution de la classe Application ci-dessous ?

```
class NombreEntier
{
  private int e;
  public NombreEntier(int n) {e=n;}
  public void decaler(int d) {e+=d;}
  public void afficher() {System.out.println(e);}
}

public class Application
{
public static void main (String args[])
{
  NombreEntier e1 = new NombreEntier(22);
  System.out.print("e1=␣");
  e1.afficher();
  NombreEntier e2 = new NombreEntier(5);
  System.out.print("e2=␣");
  e2.afficher();
  e2.decaler(17);
  System.out.print("e2␣=␣");
  e2.afficher();
  System.out.println("e1==e2␣est␣"+(e1==e2));
  e1=e2; e2.decaler(12);
  System.out.print ("e2=␣"); e2.afficher();
  System.out.print ("e1=␣"); e1.afficher();
  System.out.println ("e1==e2␣est␣"+(e1==e2));
}
}
```

1.6.4 Méthodes d'accès aux attributs privés

Soit les classes Mobile et Application ci-dessous :

```
class Mobile
{
 private String marque;
 private String modele;
 private String systeme;
 private double prix;
 public Mobile (String ma, String mo, String s, douple p)
 { marque=ma;
   modele=mo;
   systeme=s;
   prix=p;
 }
 public void revendre(double marge) {prix +=marge;}
 public void afficher()
 { System.out.println ("Mobile("+marque+"("+modele+")"+systeme+":"+prix);
 }
}
public class Apllication
{
  public static void main (String args[])
   {
    Mobile iphonex;
    iphonex= new Mobile("Apple","Iphone␣X","IOS","200000");
    iphonex.afficher() ;
    iphonex.revendre(10000); iphonex.afficher();
    galaxys9= new Mobile("Samsung","Galaxy␣S9","Android","150000");
   }
}
```

Modifier la définition de la classe *Mobile* en supprimant la méthode *afficher* et en introduisant quatre méthodes nommées getmarque, getModele, getSysteme et getPrix fournissant respectivement la marque, le modèle, le système et le prix d'un mobile. Adapter la classe Application pour quelle prendre en charge les nouvelles modifications.

1.6.5 Casting

On suppose qu'on dispose de la classe A ainsi définie :

```
class Casting
{
 void fx (int n, float x) {}
 void gx (byte b) {}
}
```

Soit la classe Application ci-dessous, donner le résulta de chaque appels de fx et de gx :

```
public class Apllication
{
  public static void main (String args[])
   {
     Casting c; int n; byte b; float f; double d;
         c.fx(n,f);
         c.fx(b+3,f);
         c.fx(b,f);
         c.fx(n,d);
         c.fx(n,(float)d);
         c.fx(n,2*f);
         c.fx(n+5,f+0.5);
         c.gx(b);
         c.gx(b+1);
         c.gx(b++);
         c.gx(3);
    }
}
```

1.6.6 Attributs et méthodes de classe

Trouver les erreurs dans la définition dans les classes ci-dessous ?

```
class Exemple
{
 static private final double d= 0.1;
 private int i;
 static int fx(int n)
  {
   i=n;
  }
 void gx(int n)
  {
   i=n;
   d=n;
  }
}
public class Application
{
 public static void main (String[] args)
 { Exemple e1=new Exemple(); int ii=5;
  e1.gx(ii);
  e1.fx(ii);
  Exemple.fx(ii);
  fx(ii);
 }
}
```

1.6.7 Attributs et méthodes de classe(2)

Réaliser une classe *Compte* qui permet des créer des comptes bancaires et d'attribuer un numéro unique à chaque nouveau compte bancaire créé. On ne cherchera pas à réutiliser les numéros de comptes détruits. Notre classe est dotée uniquement d'un constructeur, d'une méthode getIdentificateur fournissant le numéro attribué au compte et d'une méthode getIdentificateurMax fournissant le numéro du dernier compte créé. Écrire les classes Compte et Application.

1.6.8 Surdéfinition de méthodes

Trouver les erreurs figurant dans la classe ci-dessous ?

```
class SurDefinition
{
 public void fx(int i){}
 public int fx(int ii){}
 public void gx(float f){}
 public void gx(final double d){}
 public void hx(long l){}
 public int hx(final long ll){}
}
```

1.6.9 Surdéfinition de méthodes(2)

Soit la définition de la classe Definition et Application ci-dessous :

```
class
{
 public void fx(byte b){System.out.println("Je␣suis␣fx(byte)");}
 public void fx(int n){System.out.println("Je␣suis␣fx(int)");}
 public void fx(float x){System.out.println("Je␣suis␣fx(float)");}
 public void fx(double y){System.out.println("Je␣suis␣fx(double)")}
}
public class Application
{
 public static void main (String[] args)
 {
    Definition d; byte b; short s; int i; long l; float f; double d;
    d.fx(b);
        d.fx(s);
        d.fx(l);
        d.fx(f);
        d.fx(d);
        d.fx(2.*f);
        d.fx(b+1);
        d.fx(b++);
 }
}
```

Quelles sont les méthodes appelées par les instructions dans la méthodes *main* ?