Université Paris-Diderot - L1 Novembre 2012

IF1: Introduction à l'Informatique et à la programmation - Devoir sur Table

(Correction)

Documents non autorisés. Le barème est donné seulement à titre indicatif.

Exercice 1 (10 points)

1. *Un nombre de Armstrong est un entier naturel qui est égal à la somme des cubes des chiffres qui le composent. Par exemple,* $153 = 1^3 + 5^3 + 3^3 = 1 + 125 + 27$*, est un nombre de Armstrong. Écrire une fonction Java* `genereArmstrong` *qui genère tous les nombres de Armstrong contenant exactement 3 chiffres.*

   Correction :

```
static public void genereArmstrong(){
 int n, somcube;
 System.out.print("Nombres de Armstrong : ");
 for(int i = 1; i<=9; i++)
    for(int j = 0; j<=9; j++)
       for(int k = 0; k<=9; k++){
          n = 100*i + 10*j + k;
          somcube = i*i*i + j*j*j + k*k*k;
          if (somcube == n) System.out.print(n+" ");}
 System.out.println(" ");
 }
```

2. *Écrire une fonction Java* `estArmstrong` *qui teste si son argument est un nombre de Armstrong.*

   Correction :

```
static boolean estArmstrong(int n){
   int chi, a=n, acum=0;
   while (a!=0){
       chi = a%10;
       acum = acum + (chi*chi*chi);
       a=a/10;}
   return (n==acum);
}
```

Exercice 2 (10 points)

1. *Écrire une fonction* `minimum` *qui prend en argument un tableau d'entiers et renvoie comme résultat la position de son plus petit élément (ne pas oublier le cas du tableau vide).*

   Correction :

```
public static int minimum(int[] a){
   if (a== null) return -1;
      else { int min=a[0]; int ind=0;
             for (int i=1; i<a.length;i++)
                 if (a[i]<min) {min=a[i]; ind=i;};
             return ind; }
    }
```

2. *Écrire une fonction* `compresser` *qui prend en argument un tableau d'entiers a et qui renvoie comme résultat un tableau qui contient les éléments de a mais qui est privé de son plus petit élément. Le tableau résultat est donc plus petit que le tableau passé en argument.*

   Ainsi par exemple, l'appel de la fonction `compresse` *sur le tableau* `{4, 7, 9, 1, 3, 5}` *renvoie comme résultat le tableau* `{4, 7, 9, 3, 5}` .

   Correction :

```
public static int[] compresser(int[] a){
  int[] res = new int[a.length -1];
  int ind = minimum(a);
  for (int i=0; i<=ind-1;i++)
      res[i]=a[i];
  for (int i=ind; i<res.length;i++)
      res[i]=a[i+1];
  return res; }
```

3. *Écrire le programme principal qui demande a l'utilisateur de rentrer un entier n, puis de rentrer les n valeurs du tableau a. Le programme compresse le tableau a, puis imprime les éléments du tableau résultat dans l'ordre inverse.*

   Correction :

```
import java.util.*;

public class Compresser{
    .
    .
    .
  public static void main(String[] args){
     Scanner sc = new Scanner(System.in);
     System.out.print("Taille tableau : ");
     int n=sc.nextInt();
     int[] t= new int[n];
     for (int i=0;i<n;i++)
        { System.out.print("Un element : "); t[i]=sc.nextInt(); }

     int b = minimum(t);
     System.out.println("Minimum : "+b);
     int[] c = new int[n];
     c= compresser(t);
     System.out.print("Tableau Resultat : ");
     for (int j=c.length-1;j>=0;j=j-1)
          System.out.print(c[j]+" ");
     System.out.println("");
```

```
    }
  }
```

4. *Donner un exemple d'exécution du programme principal.*