Correction personnelle de l'examen d'Info 1 du 24 janvier 2007

FMdKdD
fmdkdd [à] free.fr

Université du Havre
Année 2007–2008

1.1 Une variable est un nom donné à un emplacement dans la mémoire centrale (verbatim du cours).

1.2 L'instruction :

```
int a = 3;
```

est une déclaration de variable avec affectation initiale. Alors que l'instruction :

```
a = 3;
```

n'est qu'une affectation, donc la variable a doit avoir été déclarée au préalable.

1.3 La condition $x \in [a, b]$ se traduit par :

```
(x >= a && x <= b)
```

Et sa négation par :

```
!(x >= a && x <= b)
```

qui devient :

```
!(x >= a) || !(x <= b)
```

soit :

```
(x < a || x > b)
```

2 Voici une solution qui répond correctement à l'énoncé de la question 2 :

```
1  import java.util.*;
2
3  public class Exam2007Q2 {
4      public static void main(String[] args) {
5          Scanner in = new Scanner(System.in);
```

```
6
7       System.out.println("Donner un pronom : ");
8       String pronom = in.next();
9       System.out.println("Donner un verbe : ");
10      String verbe = in.next();
11
12      String verbeConjugue = conjugue(verbe, pronom);
13      String phrase = pronom + " " + verbeConjugue;
14      System.out.println(phrase);
15  }
16
17  public static String conjugue(String verbe, String pronom) {
18      if (verbe.length() < 2) {
19          /* Si le verbe a moins de deux lettres, on aura un problème
20           * dans l'initialisation du radical :
21           *
22           *  verbe.length() - 2 < 0
23           * d'où
24           *  verbe.substring(0, verbe.length() - 2);
25           *
26           * lancera une exception...
27           * De plus, quel verbe français a moins de deux lettres ?
28           */
29          throw new IllegalArgumentException();
30      }
31
32      /*
33       * On conjugue les verbes du premier groupe en ôtant le
34       * suffixe -er à l'infinitif et en apposant les terminaisons
35       * appropriées suivant le pronom.
36       */
37
38      String verbeConjugue = verbe.substring(0, verbe.length() - 2);
39
40      /* Pour ne pas avoir à faire plusieurs conditions, et
41       * pour préserver les majuscules éventuelles
42       */
43      String pronomMinus = pronom.toLowerCase();
44
45      if (pronomMinus.equals("je")) {
46          verbeConjugue += "e";
47      }
48      else if (pronomMinus.equals("tu")) {
49          verbeConjugue += "es";
```

```
50          }
51          else if (pronomMinus.equals("il") || pronomMinus.equals("elle")
52                      || pronomMinus.equals("on")) {
53              verbeConjugue += "e";
54          }
55          else if (pronomMinus.equals("nous")) {
56              verbeConjugue += "ons";
57          }
58          else if (pronomMinus.equals("vous")) {
59              verbeConjugue += "ez";
60          }
61          else if (pronomMinus.equals("ils") || pronomMinus.equals("elles")) {
62              verbeConjugue += "ent";
63          }
64          else {
65              /* Si le pronom n'est aucun de ceux ci-dessus,
66               * on ne peut pas conjuguer. Plutôt que de
67               * retourner une chaîne vide, on signale
68               * que la fonction ne s'est pas déroulée correctement
69               */
70              throw new IllegalArgumentException();
71          }
72
73          return verbeConjugue;
74      }
75  }
```

Listing 1 – Exemple de solution de la question 2

3 Une solution de la question 3 :

```
1   import java.util.*;
2
3   public class Exam2007Q3 {
4       /* La fonction principale qui s'occupe des interactions avec
5        * l'utilisateur
6        */
7       public static void main(String[] args) {
8           Scanner in = new Scanner(System.in);
9
10          double xA, yA, xB, yB, xC, yC, xD, yD;
11
12          System.out.println("Coordonnées de A : ");
13          xA = in.nextDouble();
```

```
14          yA = in.nextDouble();
15          System.out.println("Coordonnées de B : ");
16          xB = in.nextDouble();
17          yB = in.nextDouble();
18          System.out.println("Coordonnées de C : ");
19          xC = in.nextDouble();
20          yC = in.nextDouble();
21          System.out.println("Coordonnées de D : ");
22          xD = in.nextDouble();
23          yD = in.nextDouble();
24
25          boolean croisement = croiseSegment(xA, yA, xB, yB, xC, yC, xD, yD);
26
27          if (croisement)
28              System.out.println("Les segments [AB] et [CD] se croisent");
29          else
30              System.out.println("Les segments [AB] et [CD] ne se croisent pas");
31      }
32
33      /**
34       * Détermine le demi-plan du point C par rapport au segment orienté AB.
35       * @param xA l'abscisse du point A
36       * @param yA l'ordonnée du point A
37       * @param xB l'abscisse du point B
38       * @param yB l'ordonnée du point B
39       * @param xC l'abscisse du point C
40       * @param yC l'ordonnée du point C
41       * @return
42       *    +1 si C est dans le demi-plan positif,
43       *    -1 si C est dans le demi-plan négatif,
44       *    0 si C est sur la droite AB.
45       */
46      public static int demiPlan(double xA, double yA, double xB, double yB,
47                                 double xC, double yC) {
48          double produitCroix = (xB - xA) * (yC - yA) - (xC - xA) * (yB - yA);
49          return (int)Math.signum(produitCroix);
50      }
51
52      /* Le prédicat qui utilise la fonction demiPlan pour déterminer
53       * si les segments donnés se croisent
54       */
55      public static boolean croiseSegment(double xA, double yA, double xB, double yB,
56                                          double xC, double yC, double xD, double yD) {
57          int AsurCD = demiPlan(xC, yC, xD, yD, xA, yA);
```

```
58          int BsurCD = demiPlan(xC, yC, xD, yD, xB, yB);
59
60          if (AsurCD == BsurCD) {
61              // [AB] dans le même demi-plan de (CD)
62              return false;
63          }
64
65          int CsurAB = demiPlan(xA, yA, xB, yB, xC, yC);
66          int DsurAB = demiPlan(xA, yA, xB, yB, xD, yD);
67
68          if (CsurAB == DsurAB) {
69              // [CD] dans le même demi-plan de (AB)
70              return false;
71          }
72
73          return true;
74      }
75  }
```

Listing 2 – Exemple de solution de la question 3