IF1: Introduction à l'informatique et à la programmation

Premier devoir sur table du groupe MI2
17 octobre 2011 - Durée: 1h00

Documents non autorisés. Le barème est donné à titre indicatif.
La lisibilité du code Java sera prise en compte dans l'évaluation.

Exercice 1 (**6 points**)

A l'issue d'un match de rugby, les points de classement sont attribués aux deux équipes selon les règles suivantes:

- victoire: 4 points
- match nul: 2 points
- défaite: 0 points.
- au moins 4 essais marqués: 1 point (bonus offensif).
- défaite avec un écart de 7 points ou moins: 1 point (bonus défensif).

Par exemple, si un match entre les équipes A et B se termine sur le score de 35 points à 31 en faveur de A, et si A a marqué 5 essais et B a marqué 4 essais, alors A obtient 5 points de classement (victoire + bonus offensif) et B obtient 2 points de classement (bonus offensif + bonus défensif).

Écrire une méthode Java `matchDeRugby` qui prend comme paramètres 4 entiers désignant dans l'ordre: le nombre de points marqués au score par A, le nombre de points marqués au score par B, le nombre d'essais marqués par A, le nombre d'essais marqués par B, et qui affiche le nombre de points de classement obtenus par chaque équipe à l'issue du match.

Par exemple, l'appel `matchDeRugby(35,30,5,4)` affichera (aux détails d'implantation près):

```
L'équipe A obtient 5 points
L'équipe B obtient 2 points
```

Exercice 2 (**10 points**)

- Écrire une méthode Java `estDiviseur` qui prend comme arguments deux entiers, et qui renvoie la valeur `true` si le premier est un diviseur du deuxième, la valeur `false` sinon. Dans le corps de cette méthode, vous pouvez utiliser l'opération `%` de Java (rappel: si `n` et `m` désignent deux entiers, l'expression `n%m` désigne le reste de la division de `n` par `m`).
- Écrire une méthode `estDiviseur2`, qui a les mêmes arguments et renvoie le même résultats que `estDiviseur`, mais qui n'utilise pas l'opération `%`. L'algorithme à utiliser pour vérifier si le premier argument, `n`, divise le second, `m`, dans le corps de `estDiviseur2`, est le suivant:

  1. Si `n>m` alors renvoyer `false`, sinon:
  2. Si `n==m` alors renvoyer `true`, sinon:
  3. remplacer la valeur de la variable `m` par celle de l'expression `m-n`, et recommencer à l'étape 1.

La méthode `estDiviseur2` peut être écrite en Java ou en Langage Algorithmique Abstrait, au choix.

- Définir une classe Java `Diviseur` qui contient la méthode `estDiviseur`, et une méthode `main`, qui:
  - Demande à l'utilisateur d'entrer un entier positif au clavier.
  - Affiche tous les diviseur de cet entier.

  Voici un exemple de résultat d'un appel `java Diviseur`:

```
Entrer un nombre entier positif:
128
Les diviseurs de 128 sont:
 1, 2, 4, 8, 16, 32, 64, 128.
```

Exercice 3 (**4 points**)

On considère la classe suivante:

```
class Test{
    public static void etoile (int n){
        int i;
        for (i=0;i<n;i++)
            {System.out.print("*");}
        System.out.println("@");
    }
    public static void main(String[] args){
        int j;
        for (j=0;j<4;j++)
            {etoile(j);}

    }
}
```

Qu'affiche l'exécution du `main` de la classe `Test`?