Exercices sur les fonctions

Exercice 5.1.1 *adaptation de programme*

Le programme suivant calcule la division entière d'un nombre par un autre en utilisant la soustraction pour calculer le résultat sans utiliser l'opérateur `/`.

```
public class Exo8_1{
    public static void main(String[] args){
        int x,y,cour,res;
        Terminal.ecrireString("Entrez␣le␣nombre␣x:␣");
        x = Terminal.lireInt();
        Terminal.ecrireString("Entrez␣le␣nombre␣y:␣");
        y = Terminal.lireInt();
        res = 0;
        cour = x;
        while(cour>=y){
            cour=cour-y;
            res = res + 1;
        }
        Terminal.ecrireString("" + x + "/" +y + "=␣");
        Terminal.ecrireIntln(res);
    }
}
```

Question 1

Modifiez le programme pour que le calcul de la division soit réalisé par une fonction à deux paramètres, `x` et `y`.

Question 2

Que se passe-t-il dans le cas où le deuxième nombre entré vaut 0 ?

Modifiez le programme pour qu'une erreur soit déclenchée quand le diviseur y vaut 0.

Question 3

Modifiez le programme pour qu'il fonctionne également pour x ou y négatifs.

Exercice 5.1.2 *fonctions mathématiques*

Ecrire un programme avec les fonctions `carre` et `cube` qui calculent respectivement le carré et le cube (ou puissance 3) d'un nombre de type `double`. La méthode `main` doit tester ces deux fonctions sur plusieurs exemples.

Exercice 5.1.3 *égalité de tableaux*

On a vu dans les notes de cours sur les tableaux que les tests d'égalité `==` et d'inégalité `!=` de java ne teste pas l'égalité des valeurs contenues dans les tableaux mais l'identité des tableau en terme d'espace mémoire. Selon `==`, deux tableaux sont égaux si ils partagent le même espace mémoire.

Bien souvent, on veut comparer deux tableaux selon une autre égalité selon laquelle les deux tableaux `t1` et `t2` sont égaux si et seulement si ils ont la même longueur et les éléments de même indice sont égaux, c'est à dire que `t1[i] == t2[i]` pour tout indice `i`.

Ecrire une fonction appelée `estEgal` qui réalise ce test d'égalité pour des tableaux de type `int[]`.

Voici quelques exemples de résultats attendus pour des appels de `estEgal`.

```
int[] t1 = {4, 5, 6};
int[] t2 = {4, 5, 6};
int[] t3 = {4, 5};
int[] t4 = {6, 5, 4};
estEgal(t1,t2) --> true
t1 == t2       --> false
estEgal(t1,t3) --> false
t1 == t3       --> false
estEgal(t1,t4) --> false
t1 == t4       --> false
```

Exercice 5.1.4 *fonctions sur les tableaux*

Question 1

Ecrire une fonction qui cherche si un élément appartient à un tableau de `char`. Le caractère recherché et le tableau seront les deux paramètres de la fonction.

Question 2

Ecrire une fonction qui compte le nombre d'occurrences d'un caractère dans un tableau, c'est à dire le nombre de fois où un élément apparaît dans un tableau de caractères. Le caractère recherché et le tableau seront les deux paramètres de la fonction.

Exemples :

caractère	tableau	résultat
'z'	{'a','b','c'}	0
'a'	{'a','b','c'}	1
'b'	{'a','b','c','b','d','b','a','d'}	3

Question 3

Ecrire une fonction qui prend deux tableaux de caractères en paramètres et qui teste si tous les éléments du premier tableau apparaissent au moins une fois dans le deuxième tableau. Il est possible d'utiliser dans le corps de cette fonction la fonction écrite pour la réponse à la question 1.

Exercice 5.1.5 *concaténation de tableaux*

On appelle *concaténation* l'opération qui prend deux tableaux et calcule un tableau contenant les éléments du premier tableau, puis, à leur suite, les éléments du second tableau, dans le même ordre, mais avec un indice différent.

Par exemple, `t3` ci-dessous est la concaténation de `t1` et `t2`.

`t1`

0	1	2	3
12	17	15	10

`t2`

0	1	2
13	14	11

`t3`

0	1	2	3	4	5	6
12	17	15	10	13	14	11

Ecrire une fonction qui calcule la concaténation de deux tableaux d'entiers. Notez qu'une fonction peut renvoyer un tableau comme résultat aussi bien qu'un entier ou un booléen. Indice : il faut créer le tableau résultat dans le corps de la fonction.

Exercice 5.1.6 *affichage de tableau*

On donne un programme `AfficheTable` qui affiche le contenu d'un tableau avec un cadre.

```
class Exo8_6{
    static void afficheTable(int[] t){
        Terminal.ecrireChar('+');
        for (int i=0; i<t.length; i++){
            Terminal.ecrireString("---+");
        }
        Terminal.sautDeLigne();
        Terminal.ecrireChar('|');
        for (int i=0; i<t.length; i++){
            Terminal.ecrireString(" " + t[i] + " |");
        }
        Terminal.sautDeLigne();
        Terminal.ecrireChar('+');
        for (int i=0; i<t.length; i++){
            Terminal.ecrireString("---+");
        }
        Terminal.sautDeLigne();
    }
    public static void main(String[] args){
        int[] ex = {1,5,8,9,7};
        int[] ex2 = {12, 5, 8, 123};
        afficheTable(ex);
        afficheTable(ex2);
```

```
    }
}
```

L'exemple `ex2` montre que la méthode `afficheTable` ne fonctionne pas bien si le tableau contient des nombres à plusieurs chiffres. Le but de l'exercice est de corriger le programme pour que l'affichage du tableau se passe correctement dans tous les cas.

Question 1

Dans un premier temps, on ne s'intéressera qu'aux entiers strictement positifs. Le nombre de caractères `-` à afficher à chaque tour de boucle dépend du nombre de chiffres de la valeur contenue dans le tableau à l'indice `i`.

Question 2

Améliorez encore le programme pour qu'il fonctionne avec des entiers négatifs, positifs ou nuls.