Contrôle Continu de TP (durée 1h) IF1

1 À lire très attentivement avant de commencer

1.1 Instructions pour enregistrer votre travail

Lisez très attentivement les instructions suivantes. Nous vous demandons de les suivre *scrupuleusement*, sous peine de voir votre travail *non évalué*.

- Dans votre répertoire personnel, créez un nouveau répertoire `controletp_login` (où `login` est votre login Unix) dans lequel vous enregistrerez tout votre travail. Déplacez-vous dans ce répertoire avant de lancer `xemacs`. Exemple :
  ```
  $ mkdir controletp_pmaure98
  $ cd controletp_pmaure98
  ```
- Une fois votre travail fini, déplacez-vous de nouveau dans votre répertoire personnel (à l'aide de la commande `cd`). Vérifiez (à l'aide de la commande `ls`) que tous vos fichiers sont bien enregistrés dans le répertoire `controletp_login` et archivez ce répertoire (à l'aide de la commande `tar`). Exemple :
  ```
  $ cd ..
  $ ls controletp_pmaure98
  $ tar -cvf controletp_pmaure98.tar controletp_pmaure98
  ```
- Copiez votre archive dans le répertoire `/monits/monitIF1/controletp_B2/`. Exemple :
  ```
  $ cp controletp_pmaure98.tar /monits/monitIF1/controletp_B2/
  ```

1.2 Critères de notation

Un barême indicatif vous est fourni. Pour chaque exercice, la note maximale est donnée lorsque votre programme se compile sans erreur et qu'il fait bien ce qui est demandé.

Si toutefois votre programme ne peut pas être compilé ou s'il ne fait pas bien ce qui est demandé, vous n'aurez pas forcément zéro à la question : nous regarderons le code que vous aurez écrit. Nous accorderons une attention particulière à la présentation du code (ponctuation, indentation,...)

2 Exercices

Exercice 1 (Débogage I : 2 points)

Dans un fichier `debogage1.java`, recopiez et corrigez le programme suivant afin qu'il compile sans erreur :

```
import fr.jussieu.script.Deug;
public class debogage1{
   public static void main(String[] args){
      Deug.print("Bonjour, entrez un entier: ");
      i = Deug.readInt();
      if (i=0) {
         Deug.println(cet entier vaut zéro);
      }
      else if (i>0) {
         Deug.println(cet entier est positif);
      }
      else {
         Deug.println(cet entier est négatif);
      }
   }
```

```
}
```

Exercice 2 (Débogage II : 4 points)

Dans un fichier `debogage2.java`, recopiez et corrigez le programme suivant afin qu'il compile sans erreur **et** qu'il indique si le nombre entré par l'utilisateur est pair ou impair :

```
import fr.jussieu.script.Deug;
public class debogage2{

    public static boolean est_pair(int i){
        int j = 0;
        boolean pair = ((j/2) == 0);
        return pair;
    }

    public static void main(String[] args){
        Deug.print("Bonjour,␣entrez␣un␣entier:␣");
        int n = Deug.readInt();
        boolean pair;
        if (pair)
            Deug.println("cet␣entier␣est␣pair");
        else
            Deug.println("cet␣entier␣est␣impair");
    }
}
```

Exercice 3 (Chaînes de caractères, boucle : 6 points)

Vous répondrez aux questions de cette partie dans un fichier `palindrome.java`

a) Écrivez une fonction `boolean est_palindrome(String m)` qui détermine si le mot `m` est un palindrome. Un mot est un palindrome si la première lettre est la même que la dernière, la deuxième lettre est la même que l'avant-dernière, etc. (attention à la parité du nombre de lettres du mot...) Les mots suivants sont des palindromes : `alla, elle, ressasser, snobons`, etc.

b) Dans la fonction `main`, vous demandez à l'utisateur d'entrer un mot, vous lisez ce mot à l'aide de `Deug.readLine()`, puis vous affichez à l'écran un message indiquant si le mot entré par l'utilisateur est un palindrome ou non.

Exercice 4 (Tableaux : 8 points)

Vous répondrez aux questions de cette partie dans un fichier `tableau.java`

a) Écrivez une fonction `void affiche(double[] tab)` qui prend en argument un tableau de nombres réels `tab` et qui affiche ses éléments à l'écran.

b) Écrivez une fonction `void remplit(double[] tab)` qui prend en argument un tableau de nombres réels `tab` et qui le remplit avec les nombres $1, 2, \cdots, n$ où n est la taille du tableau `tab`.

c) Écrivez une fonction `void multiplie(double[] tab, double c)` qui prend en argument un tableau de nombres réels `tab` et un nombre réel `c` et qui multiplie tous les éléments de `tab` par `c`. *Cette fonction doit modifier le tableau `tab` et non pas en créer un nouveau.*

d) Dans la fonction `main`, vous demandez à l'utilisateur d'entrer un entier `n`. Vous créez ensuite un tableau `t` de taille `n`, tableau que vous remplissez à l'aide de la fonction `remplit()`. Puis vous multipliez les éléments de ce tableau par 3.14 à l'aide de la fonction `multiplie()` et enfin vous affichez le tableau multiplié à l'aide de la fonction `affiche()`.