

Contrôle de TP du 08/12/2006

Durée : 2h

Quelques conseils :

- Commencez le programme par une zone de commentaires indiquant le nom de votre groupe (A3), votre nom, la date et le but du programme
- Indentez correctement votre programme
- Utilisez des noms de variables significatifs (évités les noms de variables d'une lettre)
- Toute utilisation d'une fonction `Deug.read` doit être précédée d'une instruction `Deug.println` qui indique à l'utilisateur ce qu'on attend de lui.

À chaque exercice doit correspondre un ou plusieurs fichier(s) **.java**. Ces fichiers seront stockés dans un répertoire nommé **exam**. À la fin de l'examen, venez nous voir pour que nous enregistrions vos fichiers.

Exercice 1 (Théorème de Pythagore)

1. Écrire un programme **Pyth.java** qui demande à l'utilisateur les longueurs a , b , c des côtés d'un triangle, en commençant par le plus grand côté, et qui indique si le triangle est rectangle en utilisant le théorème de Pythagore, c'est-à-dire en vérifiant si $a^2 = b^2 + c^2$.

Par exemple :

```
Longueur du plus grand côté ?
5
Longueur du 2ème côté ?
4
Longueur du 3ème côté ?
3
C'est bien un triangle rectangle.
```

2. À partir du programme précédent, écrire un programme **Pyth2.java** qui n'impose plus de rentrer le plus grand côté en premier, et qui décide si le triangle est rectangle en testant si l'une des 3 égalités possibles est vraie ($a^2 = b^2 + c^2$ ou $b^2 = a^2 + c^2$ ou $c^2 = a^2 + b^2$).

Exercice 2 (Week-end)

Écrire un programme **WeekEnd.java** qui demande à l'utilisateur le jour de la semaine, et qui renvoie le nombre de jours avant le week-end.

Par exemple :

```
Quel jour sommes-nous ?
mardi
Courage, plus que 4 jours avant le week-end !
```

Si l'utilisateur entre **samedi** ou **dimanche**, le programme répondra **Chouette, c'est le week-end !**

Exercice 3 (plus grand préfixe commun)

Écrire un programme **Prefixe.java** qui demande deux mots à l'utilisateur, puis renvoie le plus grand préfixe commun à ces deux mots. Par exemple :

```
Premier mot ?
chocolat
Deuxième mot ?
choucroute
Le plus grand préfixe commun est : ch.
```

Exercice 4 (Scrabble)

Le Scrabble est un jeu qui attribue une valeur à chaque lettre et qui demande aux joueurs de construire des mots de valeur élevée. La valeur d'un mot est la somme des valeurs des lettres qui le composent.

Toutes les méthodes devront être écrites dans la classe **Scrabble**.

1. Créer la méthode **public static int numeroLettre (char lettre)** qui prend un caractère et qui renvoie le numéro de la lettre (c'est à dire 'a' = 0, 'b' = 1 ...), calculé par :
`(int)lettre - 97`.
Explication : `(int)'a'` renvoie le code Ascii de 'a' c'est à dire 97, le code Ascii de `(int)'b'` renvoie 98, etc.
2. Créer une méthode **public static int valeurMot (String text, int[] val)** qui prend 2 arguments : un mot et un tableau de 26 valeurs, et qui renvoie la valeur du mot. La valeur de chaque lettre est indiquée par le tableau : le premier entier représente la valeur de 'a', le second la valeur de 'b', etc.
3. Initialiser le tableau de valeurs par 'a' = 2, 'b' = 4, 'c' = 6 ... et vérifier que les mots *scrabble* et *groquick* valent respectivement 124 et 202 points.

Exercice 5 (calculatrice)

L'objectif de cet exercice est de réaliser une mini-calculatrice infixe (l'opérateur se situe entre les opérandes, comme dans l'expression $3 + 4$). L'utilisateur rentrera donc une liste de nombres et d'opérateurs, et le programme retournera le résultat du calcul.

Toutes les méthodes devront être écrites dans la classe **Calculatrice**.

1. Écrire la méthode **public static double calcul (double op1, char operateur, double op2)**. Cette fonction va retourner le résultat du calcul entre *op1* et *op2* avec *operateur*. L'argument *operateur* peut prendre les valeurs + (addition), - (soustraction), * (multiplication) et \ (division).
Attention à bien écrire le caractère correspondant à \.
2. Écrire une méthode **public static String [] litVecteurString ()** qui demande à l'utilisateur la longueur de l'expression à calculer, puis lui fait rentrer successivement dans un tableau chaque nombre ou opérateur (sous la forme d'une chaîne de caractère), et pour finir retourne le tableau ainsi rempli.
3. Écrire la méthode **public static double calculatrice (String [] exp)**. Cette dernière devra retourner le résultat de l'expression fournie par le tableau *exp*. Ne vous souciez pas de la priorité des opérandes : $2 + 3 * 5 - 4$ sera interprété par le programme comme $((2 + 3) * 5) - 4$
4. Écrire la méthode **main** afin de vérifier que le programme calcule $2 + 3 * 5 - 4 = 21$