TP – Premiers programmes

Exercice 1

Le but est de faire fonctionner le programme demandé dans l'exercice 7 du document « *Identificateurs – Instruction print* ». L'énoncé est le suivant :

Ecrire un programme qui demande à l'utilisateur de saisir un prix de base et le montant d'un bon d'achat (2 entiers). Le programme doit afficher ensuite le prix à payer, une fois le bon d'achat déduit. Il faudra respecter les affichages de l'exemple ci-dessous *(bien respecter les retours à la ligne).*

```
Entrez le prix initial de l'article : 79
Entrez le montant du bon d'achat : 15

Pour un prix de base de 79 euros, et un bon d'achat de 15 euros,
le prix à payer est de 64 euros.
```

1) Récupérer sur Moodle le fichier ***CalculAvecBonAchatACompleter***. Pour ce faire copier-coller le code dans un éditeur de texte (celui de votre choix : le bloc note, Notepad, …)

2) Enregistrer le fichier dans le dossier de votre choix. Bien faire attention au nom que vous allez attribuer au fichier !
   Puis mettre à jour la ligne ***package*** du programme pour qu'elle soit conforme à votre organisation de fichiers.

   Par exemple, si l'on décide de conserver la ligne ***package*** telle qu'écrite dans le programme, c'est-à-dire : ***package*** *notiondebase.affectation;*

   Et si vous envisagez de placer tous vos programmes Java dans un dossier nommé ***programmesJava*****,** vous devez créer dans celui-ci un dossier ***notiondebase***, puis à l'intérieur de ce dernier un autre dossier nommé ***affectation***. C'est dans ce dossier ***affectation*** que vous enregistrerez le fichier contenant le code source Java.

```
programmeJava
      =>  notiondebase
                  =>  affectation
                            => le fichier contenant le programme
```

3) Compilez le programme avec la commande **javac**. Par rapport, à l'exemple suivant, il faut exécuter la commande à partir du dossier ***programmeJava*** et ne pas oublier d'indiquer le chemin du fichier à compiler (ne pas écrire directement le nom du fichier, sinon la commande **javac** ira chercher le fichier dans le dossier courant, donc ***programmeJava,*** et ne le trouvera pas).

4) Exécutez le programme avec la commande **java**. La commande exécute le code contenu dans le fichier portant l'extension ***.class***. Ce fichier se trouve dans le dossier ***affectation***. Il ne faut pas oublier d'indiquer l'imbrication des paquetages conduisant à ce fichier (ne pas écrire directement le nom du fichier ayant l'extension ***.class***, sinon la commande **java** ira chercher ce fichier dans le dossier courant, donc ***programmJava***).

5) Vous constatez que le programme ne réalise pas tout à fait le traitement souhaité. Modifier le programme et le mettre au point pour qu'il réponde à l'énoncé (voir l'affichage souhaité dans l'encadré).

Exercice 2

Il s'agit maintenant de traiter l'exercice 1 du document « *Programmes avec l'instruction d'affectation* ». L'énoncé est le suivant :

Ecrire un programme qui demande à l'utilisateur de saisir un nombre d'heures, un nombre de minutes et un nombre de secondes. Le programme affiche ensuite la conversion de cette durée en une durée équivalente exprimée en secondes. Les heures, les minutes et les secondes sont des entiers.

Avant d'écrire le programme, répondre à la question suivante. Si l'utilisateur entre 2 pour les heures, 10 pour les minutes et 5 pour les secondes, quel résultat le programme doit-il afficher ?

Exemple d'exécution

```
Programme de conversion heures/minutes/secondes en secondes.
Heures ......... : 3
Minutes ........ : 10
Secondes ....... : 35

La duree equivalente est 11435 secondes.
```

1) Répondre à la question : Si l'utilisateur entre 2 pour les heures, 10 pour les minutes et 5 pour les secondes, quel résultat le programme doit-il afficher ?

2) Récupérer sur Moodle le fichier ***ConversionSeconde*ACompleter**. Pour ce faire copier-coller le code dans un éditeur de texte (celui de votre choix : le bloc note, Notepad, …)

3) Enregistrer le fichier dans le dossier de votre choix. Puis mettre à jour la ligne ***package*** du programme pour qu'elle soit conforme à votre organisation de fichiers.

4) Compilez le programme et l'exécuter. Vous constaterez que le programme ne fait rien.

5) Modifier le programme et le mettre au point pour qu'il réponde à l'énoncé (voir l'affichage souhaité dans l'encadré).

Exercice 3

Il s'agit maintenant de traiter l'exercice 2 du document « *Programmes avec l'instruction d'affectation* ». L'énoncé est le suivant :

Ecrire un programme qui demande à l'utilisateur de saisir une durée en secondes et qui affiche ensuite la conversion de celle-ci en heures, minutes et secondes. Toutes les valeurs sont des entiers. Les minutes et les secondes obtenues seront comprises entre 0 et 59.

Avant d'écrire le programme, répondre à la question suivante. Quelle est la conversion de 7805 secondes en une durée équivalente exprimée en heures, minutes et secondes ?

Exemple d'exécution

```
Programme de conversion secondes en heures/minutes/secondes.

Entrez les secondes à convertir : 11435

La duree equivalente est 3 heure(s) 10 minute(s) 35 seconde(s).
```

1) Répondre à la question : Quelle est la conversion de 7805 secondes en une durée équivalente exprimée en heures, minutes et secondes ?

2) Quels opérateurs Java allez-vous utiliser pour réaliser ce calcul ?

3) Ecrire le programme et le mettre au point pour qu'il réponde à l'énoncé (voir l'affichage souhaité dans l'encadré).

Exercice 4

Il s'agit maintenant de traiter l'exercice 3 du document « *Programmes avec l'instruction d'affectation* ». L'énoncé est le suivant :

Ecrire un programme qui demande à l'utilisateur d'entrer un nombre réel x, et un entier p. Le programme calcule et affiche ensuite la valeur x à la puissance p, x^p. On utilisera la fonction *Math.pow*. Par exemple, *Math.pow(x, p)* calcule x^p.

Exemple d'exécution

```
Ce programme fait un calcul de puissance.
Donnez un nombre réel ..... : 6,2
Donnez une puissance ..... : 5
6.2 à la puissance 5 est egal a 9161.32832
```

1) Pour cet exercice, les nombres réels seront stockés dans des variables de type ***double***. Pour effectuer une lecture au clavier d'une valeur de type ***double***, on n'utilisera pas l'opération ***nextInt***. Quel pourrait bien être le nom de cette opération ?

2) Ecrire le programme et le mettre au point pour qu'il réponde à l'énoncé (voir l'affichage souhaité dans l'encadré)