

Introduction à la Programmation Java (IP1-Java)

Partiel – Durée : 2 heures

Université Paris-Diderot – Samedi 3 Novembre 2018

- Aucun document ni aucune machine ne sont autorisés. Les téléphones doivent être rangés.
- Les exercices sont tous indépendants.
- **Attention** : Indiquez au début de votre copie quel langage de programmation vous utiliserez dans le restant de votre copie. Il n'est pas possible de changer de langage une fois celui-ci choisi. Pour rappel, les filières INFO, MATHS-INFO, L1-L3-M1 MIASHS Linguistique doivent composer en JAVA mais si vous voulez composer en PYTHON, merci de nous l'indiquer.
- Une réponse peut utiliser les réponses attendues à une question précédente (même si elle est non traitée).
- Les fragments de code doivent être correctement indentés.
- Dans les énoncés, on propose parfois une liste de fonctions et procédures utilisables. Vous pouvez cependant utiliser tout équivalent JAVA (Par exemple, `s.length()` à la place de `stringLength(s)`). Attention cependant à ne pas utiliser des fonctions de la bibliothèque standard qui n'auraient pas été abordées en cours.

Exercice 1

1. Qu'écrire à la place de `A`, `B`, `C`, `D` et `E` pour que la fonction « sommePairImpair » suivante renvoie la somme des nombres pairs entre 0 et `a` inclus si `a` est pair et la somme des nombres impairs compris entre 1 et `a` inclus sinon ? Par exemple, `sommePairImpair(5)` renvoie 9 car $1 + 3 + 5 = 9$.

```
public static int sommePairImpair (int a) {
    int s = 0;
    if ( A ) {
        for (int i = 2; i <= B ; i = i + C ) {
            s = s + D ;
        }
    } else {
        for (int i = 1; i <= B ; i = i + C ) {
            s = s + D ;
        }
    }
    E ;
}
```

2. Quelles sont les valeurs des variables `x`, `y` et `z` à la fin de l'exécution des instructions suivantes ?

```
int x=3;
int y=1;
for (int i=1; i<5; i=i+1){
    y = 2 * x;
}
for (int j=0; j<=6; j=j+1){
    if(j%3 == 0){
        x = x - 1;
    } else{
        x = x + 2;
    }
}
String z = "AB";
if((x>y) || (x<y)){
    z = z + z + z ;
} else{
    z = z + z;
}
```

3. On considère la fonction `foo` dont la définition est donnée ci-dessous. Que renvoie `foo("100")` ?

```

public static String foo(String s){
    String a=""
    for(int i=0;i<s.length();i=i+1){
        a = a + "A";
    }
    return a;
}

```

□

Exercice 2

1. Écrire une procédure `rectanglePlein` qui prend comme arguments deux entiers `li` et `col` et affiche un rectangle de `li` lignes et `col` colonnes rempli du caractère `#`. Par exemple, `rectanglePlein(4,3)` affiche :

```

###
###
###
###

```

2. Écrire une procédure `rectangleAvecBords` qui prend comme arguments deux entiers `li` et `col` et affiche un rectangle de `li` lignes et `col` colonnes dont les bords sont faits du caractère `#` et l'intérieur du caractère `$`. Par exemple, `rectangleAvecBords(4,3)` affiche :

```

###
#$#
#$#
###

```

□

Exercice 3 Un jardinier veut se lancer dans la culture de tomates. Un type de tomates est donné par un facteur f . Chaque année une graine plantée de tomates donne alors $f + 3$ tomates et chaque tomate contient $5 * f$ graines.

1. Écrire une fonction `nbrTomates` qui prend comme arguments un entier f correspondant à un type de tomates et un entier g correspondant à un nombre de graines et qui renvoie le nombre tomates obtenues au bout d'un an pour le nombre de graines. Par exemple, `nbrTomates(2,3)` renvoie 15.
2. Écrire une fonction `nbrGraines` qui prend comme arguments un entier f correspondant à un type de tomates et un entier g correspondant à un nombre de graines initial et qui renvoie le nombre de graines obtenues au bout d'un an. On compte ici le nombre de graines fournies par les nouvelles tomates, les graines initiales ayant été plantées. Par exemple, `nbrGraines(2,3)` renvoie 150 car on a obtenu 15 tomates fournissant chacune 10 graines.
3. Écrire une fonction `nbrTomAnnee` qui prend comme arguments un entier f correspondant à un type de tomates, un entier g correspondant à un nombre de graines initiales et un entier n correspondant à un nombre d'années et qui renvoie le nombre de tomates obtenu l'année n , sachant que chaque année on plante les nouvelles graines des tomates obtenues l'année précédente. Par exemple, si on a 3 graines d'un type de tomates de facteur 2. au bout d'un an on a 150 graines, donc la deuxième année on aura 750 tomates qui produiront 7500 graines et la troisième année on aura alors 37500 tomates. Ainsi `nbrTomAnnee(2,3,2)` renvoie 750 et `nbrTomAnnee(2,3,3)` renvoie 37500.
4. Le jardinier hésite entre acheter 4 graines d'un type de tomates avec un facteur 2 ou acheter 2 graines d'un type de tomates avec un facteur 4. Il voudrait savoir avec quel choix il aura le plus de tomates la dixième année. Écrire un programme utilisant les fonctions précédentes qui affiche le message "Choix 1 meilleur" si la dixième année le premier choix produit plus de tomates et sinon le programme affiche "Choix 2 meilleur".

□

Exercice 4 En JAVA, pour les questions suivantes, vous pouvez utiliser par exemple :

- « `String characterAtPos (String s, int i)` » qui renvoie la chaîne de longueur 1 à la position i de s (on rappelle que la première position est la position 0).
- « `int stringLength (String s)` » qui renvoie la longueur de la chaîne s .
- « `boolean stringEquals (String s1, String s2)` » qui renvoie `true` si les deux chaînes $s1$ et $s2$ sont égales et `false` sinon.

1. Écrire une fonction `occurrence` qui prend en arguments une chaîne de caractères s et une chaîne de caractères a de longueur 1. Cette fonction renvoie le nombre de fois que le caractère présent dans a apparaît dans s . Par exemple, `occurrence("KAYAK","A")` renvoie 2.
2. Écrire une fonction `present` qui prend en arguments deux chaînes de caractères $s1$ et $s2$ et renvoie `true` si tous les caractères de $s2$ sont présents dans $s1$ et `false` sinon. Par exemple, `present("bob l'éponge","go ne")` renvoie `true`.

3. Écrire une fonction `battage` qui prend en arguments deux chaînes de caractères `s1` et `s2` et renvoie la chaîne de caractères produit de `battage` de ces deux chaînes, c'est-à-dire la chaîne de caractères contenant un caractère de `s1` puis un de `s2` puis un de `s1` etc jusqu'à ce que l'on arrive à la fin d'une des deux chaînes et dans ce cas on finit avec les dernières lettres de la chaîne la plus longue. Par exemple `battage("licence", "informatique")` renvoie "liincfeonrcmeatique".

□

Exercice 5 Dans cet exercice, on veut utiliser des tableaux pour représenter des ensembles d'entiers. On rappelle que dans un ensemble, un élément n'apparaît pas deux fois. Le tableau de taille 0 représentera l'ensemble vide.

1. Écrire une fonction `ajouteElem` qui prend comme arguments un tableau d'entiers `tab` représentant un ensemble et un entier `x` et renvoie un tableau dont les `tab.length` premiers éléments sont les mêmes que ceux de `tab` et dont le dernier élément est `x` si et seulement si `x` n'apparaît pas dans `tab` sinon la fonction renvoie une copie de `tab`. Par exemple, `ajouteElem(T,1)` avec `T={2, 4, 5, 1}` renverra `{2, 4, 5, 1}` mais si `T={2, 4, 5, 7}` alors `ajouteElem(T,1)` renverra `{2, 4, 5, 7, 1}`.
2. Écrire une fonction `estEnsemble` qui prend comme argument un tableau d'entiers `tab` et renvoie `true` si ce tableau correspond à un ensemble, c'est-à-dire si chaque entier apparaît au maximum une fois dans le tableau et `false` sinon. Par exemple, `estEnsemble(T)` avec `T={2, 4, 5, 1}` renvoie `true` mais si `T={5, 4, 5, 7}` alors elle renvoie `false`. Si le tableau est de taille 0, la fonction renvoie bien entendu `true`.
3. Écrire une fonction `cardinal` qui prend comme argument un tableau d'entiers `tab` et renvoie `-1` si ce tableau ne correspond pas à un ensemble et sinon elle renvoie la taille de l'ensemble. Par exemple, `cardinal(T)` avec `T={2, 4, 5, 1}` renvoie 4.
4. Écrire une fonction `intersection` qui prend comme arguments deux tableaux d'entiers et si les deux tableaux représentent des ensembles alors la fonction renvoie un tableau correspondant à l'intersection des deux sinon la fonction renvoie le tableau de taille 0. Par exemple, `intersection(T1,T2)` avec `T1={2, 4, 5, 1}` `T2={3, 1, 2}` renvoie le tableau `{2,1}`.
5. Écrire un fonction `egal` qui prend comme arguments deux tableaux d'entiers et si les deux tableaux représentent les mêmes ensembles alors la fonction renvoie `true` et sinon elle renvoie `false`. Par exemple, `egal(T1,T2)` avec `T1={1,3,4}` et `T2={3,1,4}` renvoie `true` mais si `T1={1,3,4}` et `T2={8,1,5}`, alors la fonction renvoie `false`.

□