République du Cameroun
Paix-Travail-Patrie
Ministère de l'Enseignement Supérieur
Université de Maroua
Faculté Des Sciences
Département de Mathématiques et Informatique

BP 814 Maroua

Republic of Cameroon
Peace-Work-Fatherland
Ministry of Higher Education
The University of Maroua
Faculty of Sciences
Department of Mathematics and Computer Science

Travaux Dirigés/Pratiques : Fiche 1
Unité d'Enseignement : Programmation orientée objet en JAVA
Enseignant : M. Mouga

Exercice 0_1 : Simulation du code

— 01 : Qu'affiche ce programme ?

```
class ExempleVariable
{
public static void main(String[] args)
{
int n = 4;
int nCarre;
nCarre = n * n;
System.out.println("La variable n contient " +
n);
System.out.println("Le carre de " + n + " est "
+ nCarre + ".");
System.out.println("Le double de n est " + 2 *
n + ".");
} }
```

— 02 : Qu'affiche ce bout de code ?

```
int a = 1;
int b = 2;
a = b;
b = a;

System.out.println(a + ", " + b);
```

— 021 Même question

```
int a = 1;
int b = 2;
if (a == b) {
System.out.println("Cas 1");
} else {
System.out.println("Cas 2");
}
if (2 * a == b){
System.out.println("b est egal au double de
a."); }
```

— 03 : Que fait ce programme?

```
import java.util.Scanner;
class DemanderVariable
{
public static void main(String [] args)
{
Scanner keyb = new Scanner(System.in);
System.out.println("Entrez une valeur pour n");
int n = keyb.nextInt();
int nCarre = n * n;
System.out.println ("La variable n contient " +
n);
System.out.println("Le carre de " + n + " est "
+ nCarre + ".");
System.out.println("Le double de n est " + 2 *
n + ".");
} }
```

— 031 : Qu'affiche ce bout de code ?

```
int a = 1, b = 2;
boolean c = true;
boolean d = (a == b);
boolean e = (d || (a < b));
if (e) {
System.out.println("e vaut true"); }
```

— 04 : Qu'affiche ce programme quand le user entre 1 et 2 ?

```
System.out.println("Entrez le premier
nombre:");
int n = scanner.nextInt();
System.out.println("Entrez le deuxieme
nombre:");
int p = scanner.nextInt();
if ((n < p) && (2 * n >= p)) {
System.out.print("1");
}
if ((n < p) || (2 * n >= p)) {
System.out.print("2");
}
if (n < p) {
if (2 * n >= p) {
System.out.print("3");
} else {
System.out.print("4");
}
}
System.out.println();
```

Même question pour 1 et 3 ?, 2 et 1 ?

Bon travail

— 04 : Que s'affiche-t-il quand on exécute le code :

```
for(int i = 0; i < 5; ++i) {
System.out.print(i);
if (i % 2 == 0) {
System.out.print("p");
}
System.out.print(" ");
}
System.out.println();
```

— 041 Même Question

```
for(int i = 0; i < 5; ++i)
System.out.println("i = " + i);
System.out.println("Bonjour");
```

— 051 : Proposez une nouvelle version qui enraille ce bug.

— 05 : Quel bug peut-on avoir au cours de l'exécution de bout de code ?

```
System.out.println("Entrez le nombre de
notes");
int nombre_de_notes = clavier.nextInt();
double somme = 0;
for(int i = 1; i <= nombre_de_notes; ++i) {
System.out.println("Entrez la note numero " +
i);
double note = clavier.nextDouble();
somme = somme + note;
}

System.out.println("Moyenne = " + somme /
nombre_de_notes);
```

— 06 : Qu'affiche ce bout de code ?

```
for(int j = 2; j <= 10; ++j) {
System.out.println("Table de multiplication par
" + j + ": ");
for(int i = 1; i <= 10; ++i) {
System.out.println(j + " multiplie par " + i +
" vaut " + j * i);
}
}
```

— 061 : Même quetsion

```
for(int i = 0; i < 3; ++i) {
for(int j = 0; j < 4; ++j) {
if (i == j) {
System.out.print("*");
} else {
System.out.print(j);
}
}
System.out.println("");
}
```

— 062 : Même question

```
for(int i = 0; i < 3; ++i) {
for(int j = 0; j < i; ++j) {
System.out.print(j);
}
System.out.println("");

}
```

— 063 : Même question

```
int i = 100;
do {
System.out.println("bonjour");
} while (i < 10);
```

— 064 :Même question

```
int i = 100;
while (i < 10) {
System.out.println("bonjour");
}
```

— 07 : Quaffiche ce bout de code ?

```
int[] a = new int[10]; // tableau de 10 entiers
int[] b = new int[10]; // tableau de 10 entiers
for (int i = 0; i < a.length; ++i) {
a[i] = i; // remplissage du tableau pointé par a
}
b = a; // opérateur = (affectation)
System.out.println("a[2] vaut " + a[2] + " et b[2] vaut " + b[2]);
a[2] = 42;
System.out.println("a[2] vaut " + a[2] + " et b[2] vaut " + b[2]);
```

— 071 : donnez une explication à cet affichage.

— 072 : dessinez les structures « a » et « b » à la fin de l'exécution.

— 071 : Qu'affichera ce bout de code où « a » et « b » sont les structures de la question « 07 »

```
if (a == null || b == null || a.length != b.length) {
System.out.println("contenus différents ou nuls");
}
else {
int i = 0;
while(i < a.length && (a[i] == b[i])) {
++i;
}
if (i >= a.length) {
System.out.println("contenus identiques");
}
else {
System.out.println("contenus différents")
}

}
```

Bon travail

— 08 : Qu'affiche ce bout de code ?

```
double[] tab = new double[10];
System.out.print("Le tableau contient : ");
for(double element : tab) {
System.out.print(element + " ");
}
System.out.println();
```

— 081 : dessinez la structure « array ».

```
int[][] array = {
{ 0, 1, 2, 3, 42 },
{ 4, 5, 6 },
{ 7, 8 },
{ 9, 0, 1 }
};
```

— 09 : Que contient chacune des variables ci-dessous définies ?

```
String s1 = "abcmbx";
int longueur = s1.length();
char c1 = s1.charAt(0);
char c2 = s1.charAt(longueur - 1);
int i = s1.indexOf('b');
```

— 091: Qu'affiche ce bout de code?

```
String essai = "essai";
String test = "";
for (int i = 1; i <= 3; ++i) {
test = test + essai.charAt(6-2*i);
test = essai.charAt(i) + test;
}
System.out.println(test);
```

— 011: Qu'affiche ce bout de code si le user entre sucessivement les valeurs « 3 5 2 0 7 2 -4 4 12 » ?

```
ArrayList<Integer> liste = new ArrayList<Integer>();
System.out.println("Donnez la taille voulue : ");
int taille = scanner.nextInt();
System.out.println("Saisie de " + taille + " valeurs :");
while (liste.size() < taille) {
System.out.println("Entrez la valeur " + liste.size() + " : ");
int val = scanner.nextInt();
if ((val < 0) && (!liste.isEmpty())) { liste.remove(liste.size() - 1); }
else if (val == 0) { liste.clear(); }
else if (val > 0) { liste.add(val); }
}
```

— 082 : à quoi correspond chacune des écritures ci-dessous ?

array[0]

array[1]

array[2]

— 083 : Peut-on écrire array[4] ? si oui que représente-elle ?

— 084 : Proposez le bout de code d'affichage du contenu de la structure « array ».

— 010: Qu'affiche ce programme?

```
import java.util.ArrayList;
class ArrayListSample {
public static void main(String[] args){
ArrayList<String> chaine = new ArrayList<String>();
chaine.add("un");
chaine.add("deux");
for(String part : chaine) {
System.out.print(part + " ");
}
System.out.println(chaine.get(1));
chaine.set(0, "first");
}
```

— 0101 : Dessinez cette structure en fin d'exécution.

— 0111 : Dessinez la structure « liste » à chaque étape de sa modification.

Bon travail

— 012 : Soit l'instruction de déclaration suivante :

```
int vecteur [] [] = { new int [3], new int [2] };
```

1) Schématiser cette structure
2) Pour chacune des instructions suivantes, associer sa sémantique
— la notation `vecteur[0]`
— la notation `vecteur[0][1]`
— la notation `vecteur[1]`
— la notation `vecteur[0][i-1]`
— l'expression `vecteur.length`
— l'expression `vecteur[0].length`
— l'expression `vecteur[1].length`

— 0121 : Soit maintenant :

```
int matrix [] [];
matrix = new int [2] [];
int [] vec1 = new int[3];
int [] vec2 = new int[2];
matrix[0] = vec1;

matrix[1] = vec2;
```

3) schématiser la structure matrix
4) schématiser les liens entre ces structures en mémoire
5) pour chacune des instructions ci-dessous, associer sa sémantique
— matrix[0]
— matrix[0].length
— matrix[1]
— matrix[1].length
6) pour chacune des lignes ci-dessous, compléter l'égalité d'après la séquence d'instructions précédentes

matrix[0] == ? matrix[0][1] == ? matrix[1][0] == ? matrix[1][2] == ? matrix[1][1] == ? matrix[0][2] == ?

— 013 : soit l'énoncé suivant :

Une tante fortunée vous envoie un jour une petite enveloppe, en francs tout ronds, pour vous aider à financer vos études. L'étudiant prévoyant et organisé que vous êtes décide de gérer cette somme de la façon suivante:

- Les trois quarts de cette somme seront dédiés à l'achat de livres et fournitures
- Le reste sera équitablement réparti entre les rubriques :
  - cafés,
  - abonnement au «club Informatique»,
  - Participation aux TDs spéciaux organisés par les ainés.

Vous décidez également de gérer cette somme en francs tout rond. Le reste de l'argent sera destiné à la nutrition. En supposant qu'un café vaut 2 Fcfa, qu'un numéro du «club Informatique » en vaut 4 et qu'un qu'une séance de TDs vaut 3 Fcfa, écrivez le programme `Gestion.java(classe)` vous permettant de produire la sortie suivante :

```
Combien avez-vous reçu d'argent (Fcfa)? 800
  Livre et Fournitures: 600 Fcfa
  Vous pouvez ensuite acheter:
   33 cafés
   16 numéros du club Informatique
   22 billets d'accès aux TDs
  et il vous restera 4 Fcfa pour votre nutrition.
```

— 0131 : Transformez votre précédente classe(Gestion.java) en une classe(GestionPOO.java)

Objectifs à atteindre :

- Le constructeur prendra en paramètre la somme reçue de votre tante.
- Si gestAcount est objet de la classe GestionPOO alors :
  - gestAcount.material : indique la somme réservée à l'achat des livres et fournitures.
  - gestAcount.billInf : spécifie le nombre de billets pour le club informatique
  - gestAcount.cofee : désignera le nombre de cafés possible
  - gestAcount.pocket : correspondra à la somme allouée à la nutrition
  - gestAcount.state() : une méthode affichant la répartition obtenue en fonction de la somme reçue, sous le modèle :

- argent reçu : somme Fcfa
- part pour livre et fournitures : montant Fcfa
- nombre de cafés couvert : nombre
- nombre d'abonnement au club informatique : nombre
- somme allouée à la nutrition : montant Fcfa.

— 0132 : Amélioration de la classe (GestionPOO) par ajout de méthodes.

Définissez une méthode `spendMoney(int )` qui reçoit un entier précisant la nature de la dépense et ampute la quantité associée à la rubrique d'une unité si café(0), club informatique(1) ou accès aux TDs(2). En ce qui concerne livre et fournitures(3), de même que la nutrition(4), la dépense est globale et atomique.

Toute exécution de **spendMoney** impacte l'état de l'objet sur lequel elle est appliquée.