Travaux pratiques d'informatique Nº 12

Le but principal de cette séance est de vous permettre de :

- tester vos connaissances sur l'héritage, le polymorphisme, les tableaux et la classe standard String ;
- mettre en place des notions élémentaires concernant les classes abstraites et les interfaces.

1. Choisir et encercler la (ou les) réponse(s) correcte(s) :

1.1 Indiquer les déclarations correctes. (Choisir trois réponses.)

A. *char c1 = 064770;*
B. *char c2 = 'face';*
C. *char c3 = 0xbeef;*
D. *char c4 = \u0022;*
E. *char c5 = '\iface';*
F. *char c6 = '\uface';*

1.2 Soit le programme suivant :

```
1.  public class Test {
2.  public static void main(String [ ] args) {
3.  int [ ] [ ] [ ] x = new int [3] [ ] [ ];
4.  int i,j;
5.  x[0] = new int[4][ ];
6.  x[1] = new int[2][ ];
7.  x[2] = new int[5][ ];
8.  for (i=0; i<x.length; i++)
9.  for (j=0; j<x[i].length; j++) {
10. x[i][j] = new int [i + j + 1];
11. System.out.println("size = " + x[i][j].length);
12. }}}
```

Quel est le résultat du programme ci-dessus ? (Choisir une réponse.)

A. 7 lignes affichées
B. 9 lignes affichées
C. 11 lignes affichées
D. 13 lignes affichées
E. Erreur à la compilation
F. Exception à l'exécution

1.3 Soit le programme suivant :

```
1.  public class Test {
2.  public static void main(String [ ] args) {
3.  byte [ ][ ] big = new byte [7][7];
4.  byte [ ][ ] b = new byte [2][1];
5.  byte b3 = 5;
6.  byte b2 [ ][ ][ ][ ] = new byte [2][3][1][2];
7.
8.  }}
```

Quelles lignes de code, introduites séparément à la ligne 7, permettent la compilation du programme ? (Choisir quatre réponses).

A. b2[0][1] = b;
B. b[0][0] = b3;
C. b2[1][1][0] = b[0][0];
D. b2[1][2][0] = b;
E. b2[0][1][0][0] = b[0][0];
F. b2[0][1] = big;

1.4 Soit le programme suivant :

```
1.  public class TestDogs {
2.  public static void main(String [ ] args) {
3.  Dog [ ][ ] theDogs = new Dog[3][ ];
4.  System.out.println(theDogs[2][0].toString( ));
5.  }}
6.  class Dog { }
```

Quel est le résultat du programme ci-dessus ? (Choisir une réponse.)

A. null
B. theDogs
C. Erreur à la compilation
D. Exception à l'exécution

1.5 Soit le programme suivant :

```
1.  public class CommandArgsThree {
2.  public static void main(String [ ] args) {
3.  String [ ][ ] argCopy = new String[2][2];
4.  int x;
5.  argCopy[0] = args;
6.  x = argCopy[0].length;
7.  for (int y = 0; y < x; y++) {
8.  System.out.print(" " + argCopy[0][y]);      }}}
```

Si le programme ci-dessus est lancé avec la ligne de commande ***java CommandArgsThree 1 2 3***, quel est son résultat ? (Choisir une réponse.)

A. 0 0
B. 1 2
C. 0 0 0
D. 1 2 3
E. Erreur à la compilation
F. Exception à l'exécution

1.6 Quels ne sont pas des bénéfices de l'encapsulation ? (Choisir deux réponses.)

A. Le code devient plus clair.
B. Le code devient plus efficace.
C. Des fonctionnalités peuvent être ajoutées plus tard.
D. Les modifications impliquent moins de changements du code.
E. Les modificateurs d'accès deviennent optionnels.

1.7 Soit le programme suivant :

```
1.  class B extends A {
2.  int getID( ) {
3.  return id;
4.  }}
5.  class C {
6.  public int name;
7.  }
8.  class A {
9.  C c = new C( );
10. public int id;
11. }
```

Indiquer les affirmations correctes. (Choisir deux réponses.)

A. A est un B (A is-a B)
B. C est un A (C is-a A)
C. A a un C (A has-a C)
D. B a un A (B has-a A)
E. B a un C (B has-a C)

1.8 Soit le programme suivant :

```
1.  class Over {
2.  int doStuff(int a, float b) {
3.  return 7;
4.  }}
```

5. class Over2 extends Over {

6. // introduire le code ici

7. }

Quelles méthodes, introduites séparément à la ligne 6, produisent une erreur de compilation ? (Choisir deux réponses).

A. *public int doStuff(int x, float y) { return 4; }*

B. *protected int doStuff(int x, float y) {return 4; }*

C. *private int doStuff(int x, float y) {return 4; }*

D. *private int doStuff(int x, double y) { return 4; }*

E. *long doStuff(int x, float y) { return 4; }*

F. *int doStuff(float x, int y) { return 4; }*

2. Créer un nouveau projet Eclipse appelé **PrTP12Exo2**, selon la démarche suivante :

- depuis le bureau virtuel du CMS, copier les fichiers **IAffichable.java**, **IAireCalculable.java**, **Quadrilatere.java**, **Rectangle.java**, **Carre.java** et **CP_TP12Exo2.java** dans le package **cms_tp12** du projet ;
- dans les fichiers sources appropriés, compléter les corps des classes dérivées **Rectangle** et **Carre**, en respectant les indications données sous forme de commentaires ;
- anticiper les résultats affichés suite à l'exécution du projet ;
- exécuter le projet et comparer les résultats obtenus avec ceux anticipés.

Indications :

*La classe abstraite **Quadrilatere** implémente l'interface **IAireCalculable** et contient la méthode abstraite **calculerPerimetre( )**.*

*La classe non abstraite **Rectangle** est la descendante directe de la classe de base **Quadrilatere** ; elle implémente l'interface **IAffichable** et permet d'instancier et de travailler avec des objets Java qui correspondent à des rectangles ; chaque tel rectangle est décrit par son nom, l'abscisse et l'ordonnée de son coin supérieur gauche ainsi que sa longueur et sa hauteur.*

*La classe non abstraite **Carre** dérive de la classe de base **Rectangle** et permet d'instancier et de travailler avec des objets Java qui correspondent à des carrés ; chaque tel carré est décrit par son nom, l'abscisse et l'ordonnée de son coin supérieur gauche et la longueur de son côté.*

On utilise un système de coordonnées cartésien tel que l'axe des abscisses est orienté vers la droite et l'axe des ordonnées vers le bas.

Remarque : Voici ci-dessous un diagramme UML pour le deuxième exercice de cette série :