Examen de P00 (deuxième session)

(juin 2010)

durée deux heures ; les documents ne sont pas autorisés

3 pages

Le barème est donné à titre indicatif. (total sur 23)

1. On considère le programme suivant :

```
class F {
    void f(){System.out.print("F.f() ");this.g();}
    void g(){System.out.println("F.g() ");}
}
class Fbis extends F{
    void f(){System.out.print("Fbis.f() ");this.g();}
    void g(){System.out.println("Fbis.g() ");super.f();}
}
```

- (a) (0,5 pt) Que provoque (erreur à la compilation ? à l'exécution ? affichage ?) le code suivant ?
F f = new F() ; f.f() ;
- (b) (1,5 pt) Le code suivant :
Fbis f=new Fbis() ; f.f() ;
n'est pas correct, que provoque-t-il et pourquoi ?
- (c) (1 pt) Que provoque (erreur à la compilation ? à l'exécution ? affichage ?) le code suivant :
F f=new Fbis() ; ((F)f).f() ; ?

2. On considère les classes :

```
class A{public int i; public A(){i=0;}; public String toString(){return "i="+i;}}
class B extends A {public int i; public B(){i=10;}}
```

- (a) (2 point)

- i. Dans le code suivant quel est le type de l'objet référencé par `ta` ? Quel est le type de l'objet référencé par `ta[i]` ? Que produira le code suivant (erreur à la compilation, à l'exécution, et s'il n'y a pas d'erreur quelle en est la sortie).

```
A[] ta = new B[10];
for (int i = 0; i < ta.length; i++) {
    ta[i] = new A();
    ta[i].i = i;
}
System.out.println(ta[1]);
```

- ii. (1 point) Que produira le code suivant (erreur à la compilation, à l'exécution, et s'il n'y a pas d'erreur quelle en est la sortie).

```
A[] tab = new B[10];
for (int i = 0; i < tab.length; i++) {
    tab[i] = new B();
    tab[i].i = -i;
}
System.out.println(tab[1]);
```

iii. (1 point) Même question pour :

```
B[] tb=new B[10];
for (int i = 0; i < tb.length; i++) {
    tb[i]=new B();
    tb[i].i = -i;
}
A[] tac=tb;
System.out.println(tac[1]);
```

iv. (3 points) On rappelle qu'en Java si Y est une classe dérivée de X alors Y[] est aussi dérivée de X[]. Ainsi si tx est déclarée comme un tableau de X, tx peut référencer un tableau de Y. Quel problème pose alors le code :

```
Y[] y=new Y[1];
X[] x=y;
x[0]=new X();
System.out.println(y[1]);
```

S'il y a une erreur, celle-ci peut-elle être détectée à la compilation ?

Si au lieu de considérer des tableaux on considère des List<X> et des List<Y> a-t-on le même comportement (présence ou absence d'erreur) ?

3. Le but de cet exercice est de réaliser des *files*. Une *file* est une structure de données contenant des objets auxquels on peut accéder par la méthode *mettre* qui ajoute un objet dans la file et la méthode *retirer* qui retourne et supprime de la file l'objet mis le plus anciennement dans la file. Par exemple, partant d'une file vide, si on fait successivement :

```
mettre a
mettre b
mettre c
v = retirer
w = retirer
```

v contiendra a, w contiendra b et c sera le seul élément restant dans la file. Dans la suite on suppose que les files considérées peuvent avoir une taille maximum qui sera fixée au moment de leur création. Pour pouvoir manipuler des files il faut aussi pouvoir tester si une file est vide (*estVide*) et si elle est pleine (*estPleine*).

- (a) (1 point) Définir une interface java File correspondant à des files de Object. Dans le cas d'opérations impossibles à réaliser (retirer un élément d'une file vide ou mettre un élément dans une file pleine) on pourra jeter une exception.
- (b) (4 points) Dans cette question et la suivante, la classe à définir sera une implémentation de File et toutes les méthodes et variables définies dans cette classe sauf, bien sûr, celles de l'interface File seront private ou protected.

On veut implémenter les files d'objets à partir de listes. Pour cela on utilisera la classe :

```
static public class Cell{
    private Cell suivant = null;
    private Object element;
    public Cell(Object val) {
        this.element=val;
    }
    public Cell(Object val, Cell suivant){
        this(val);
        this.suivant=suivant;
    }
    public Object getElement(){
        return element;
    }
}
```

```

    public void setElement(Object v){
        element=v;
    }
    public Cell getSuivant(){
        return suivant;
    }
    public void setSuivant(Cell s){
        this.suivant=s;
    }
}

```

(Pour réaliser une file, on pourra définir deux références sur des Cell l'une vers la tête de la liste et l'autre vers la queue) On appellera la classe correspondant à cette implémentation des files FileListe. Ecrire le code Java correspondant à cette classe.

- (c) (4 points) Dans cette question on veut réaliser des files d'objets à partir de tableaux d'objets. Pour cela on va définir une classe FileTab qui implémente l'interface File précédente. La taille maximum de la File implémentée par FileTab pourra être un paramètre d'un constructeur de FileTab. Par défaut une FileTab aura une taille maximum de 100 objets. Cette implémentation utilisera un tableau de Object pour stocker les données. L'implémentation devra assurer que mettre ou retirer un élément ne déplace pas les autres éléments du tableau. (On pourra utiliser et gérer deux "pointeurs" dans le tableau, l'un pointant sur l'indice dans le tableau où mettre le prochain élément et l'autre pointant sur l'indice dans le tableau contenant le prochain élément à prendre).
- (d) (4 points) On veut maintenant définir une méthode statique affiche ayant en paramètre un objet de type File qui affiche le contenu d'une file, sans en modifier le contenu.
- i. On ajoute à l'interface File une méthode copie qui permet de dupliquer une file. Après avoir ajouté cette méthode à l'interface File, définir l'implémentation de copie dans FileTab et FileListe. Ecrire la méthode affiche
 - ii. Comment aurait-on pu procéder en redéfinissant la méthode clone de Object ?

1. Le code:

```
class A{
    int i=10; void f(){System.out.print(" i="+i);}
}
class B extends A{
    static int i=0; void f(){System.out.print("i="+i);super.f(); }
}
```

avec le morceau de code: `A a=new B();a.f();`

- (a) provoque une erreur à l'exécution
- (b) provoque une erreur à la compilation
- (c) affiche `i=10 i=10`
- (d) affiche `i=0 i=10`

2. On considère le code:

```
interface I{void f();}
class A implements I{public void f(){System.out.println("A");}}
class B implements I{public void f(){System.out.println("B");}}
```

Le code suivant: `"I[] tabl=new A[2]; tabl[0]= new A();tabl[1]= new B(); tabl[0].f();"`

- (a) provoque une erreur à la compilation
- (b) provoque une erreur à l'exécution
- (c) affiche A

3. On considère le code suivant:

```
class A {void g() {System.out.println("A:g()"); }}
class B extends A {void g() {System.out.println("B:g()"); }}
class C extends B {void g() {System.out.println("C:g()"); }}
```

et la séquence de code `A a = new B(); ((A)a).g();` quelle est la bonne réponse:

- (a) Le code provoque une erreur à la compilation ou à l'exécution
- (b) Il affiche `A:g()`
- (c) Il affiche `B:g()`

4. (Trouver la ou les bonnes réponses) On considère la classe:

```
class X{static int i=0; int j=2; /*...*/}
```

- (a) On peut ajouter à cette classe: `void f(){i=i+j;}`
- (b) On peut ajouter à cette classe: `static void g(){i=j+i;}`
- (c) On peut ajouter à cette classe: `static void h(){i=X.j+i;}`
- (d) On peut ajouter à cette classe: `static void i(){this.i=this.i+this.j;}`

5. La classe paramétrée: `class X <T>{public T val; X(T x){val=x;}}` avec la méthode:

```
public static void somme(X<? extends Number> x){System.out.println(x.val.doubleValue());}
```

et le code: `X<Integer> x= new X<Integer>(2); somme(x);`

- (a) provoque une erreur à la compilation ou à l'exécution
- (b) affiche `i=2.0`

6. On considère le code suivant:

```
class A {void g() {System.out.println("A:g()"); }}
class B extends A {void g() {System.out.println("B:g()"); }}
class C extends B {void g() {System.out.println("C:g()"); }}
```

et la séquence de code `C a = new A(); a.g();` quelle est la bonne réponse:

- (a) Le code provoque une erreur à la compilation
- (b) Le code provoque un erreur à l'exécution
- (c) Il affiche `C:g()`
- (d) Il affiche `B:g()`

7. On considère le code suivant:

```
class A {void f() {System.out.println("A:f()"); }}
class B extends A {void g() {System.out.println("B:g()"); }}
class C extends B {void g() {System.out.println("C:g()"); }}
```

et la séquence de code `A a = new C(); a.g();` quelle est la bonne réponse:

- (a) Le code provoque une erreur à la compilation
- (b) Le code provoque un erreur à l'exécution
- (c) Il affiche C:g()
- (d) Il affiche A:f()
- (e) Il affiche B:g()

8. On considère le code:

```
class A1 {void f(){System.out.println(i);}int i=0;}
class B1 extends A1{public void f(){System.out.println(i);}int i=1;}
class B2 extends A1{public void f(){System.out.println(i);}int i=2;}
```

Le code suivant: `"A1 a1= new B1(); B2 b2=(B2)a1;b2.f();"`

- (a) provoque une erreur à la compilation
- (b) provoque une erreur à l'exécution
- (c) affiche 1
- (d) affiche 2
- (e) affiche 0

9. On considère le code suivant:

```
class A {private void g() {System.out.println("A:f()"); }}
class B extends A {void g() {System.out.println("B:g()"); }}
class C extends B {void g() {System.out.println("C:g()"); }}
```

et la séquence de code `A a = new C(); ((C)a).g();` quelle est la bonne réponse:

- (a) Le code provoque une erreur à la compilation ou à l'exécution
- (b) Il affiche C:g()
- (c) Il affiche A:f()
- (d) Il affiche B:g()

10. Les classes:

```
class D {public int val; public D(int i){val=i;}}
class E extends D{public E(int i){super(i);}}
```

avec le code :

```
D[] t1= new E[4]; t1[0]=new D(1); t1[1]=new E(3); System.out.println(t1[0].val+" "+t1[1].val);
```

- (a) provoque une erreur à la compilation ou à l'exécution
- (b) affiche i 3

11. Soit:

```
class T{
    public int i=1;int j=2;
}
```

```
class U extends T{
    public int i=10;
}
```

On considère les instructions suivantes: `T x=new U();`
`System.out.println("x.i="+x.i+" ((T)x).i="+((T)x).i+" ((U)x).i="+((U)x).i);`
Trouver la ou les bonnes réponses:

- (a) ces instructions ne passeront pas à la compilation
 - (b) ces instructions provoqueront une erreur à l'exécution
 - (c) ces instructions provoqueront l'affichage: `x.i=1 ((T)x).i=1 ((U)x).i)=10`
 - (d) ces instructions provoqueront l'affichage: `x.i=10 ((T)x).i=1 ((U)x).i)=10`
 - (e) ces instructions provoqueront l'affichage: `x.i=10 ((T)x).i=10 ((U)x).i)=10`
12. L'interface: `interface I{int i;void f();}` avec le morceau de code:
`(new I){int i=0; public void f(){System.out.println("bravo "+ i);}}.f();`
- (a) provoque une erreur à la compilation
 - (b) affiche bravo 0
13. La classe paramétrée: `class X <T>{public T val; X(T x){val=x;}}`
avec la méthode: `public static void somme(X<Number> x){System.out.println(x.val.doubleValue());}`
et le code: `X<Number> x= new X<Number>(new Integer(2)); somme(x);`
- (a) provoque une erreur à la compilation ou à l'exécution
 - (b) affiche `i=2.0`
14. L'interface: `interface I{static int i=0; void f();}` avec le morceau de code:
`(new I){int i=2; public void f(){System.out.println("bravo"+ i);}}.f();`
- (a) provoque une erreur à la compilation
 - (b) affiche bravo 2
 - (c) affiche bravo 0
15. La classe paramétrée: `class X <T>{public T val; X(T x){val=x;}}` avec la méthode:
`public static void somme(X<Number> x){System.out.println(x.val.doubleValue());}`
et le code: `X<Integer> x= new X<Integer>(2); somme(x);`
- (a) provoque une erreur à la compilation ou à l'exécution
 - (b) affiche `i=2.0`

16. Pour la classe D définie comme suit:

```
class D {
    public int x;
    public D() {x=3; };
    public D( int a){this(); x=x+a;};
    public D( int a, int b){this(b); x= x-a;};
}
```

qu'affichera le code suivant?

```
D a=new D(5,6);
System.out.println(a.x);
```

- (a) 1
- (b) 2
- (c) 3
- (d) 4

17. On considère le code suivant:

```
class A {void g() {System.out.println("A:g()"); }}
class B extends A {void g() {System.out.println("B:g()"); }}
class C extends B {void g() {System.out.println("C:g()"); }}
```

et la séquence de code `A a = new C(); ((B)a).g();` quelle est la bonne réponse:

- (a) Le code provoque une erreur à la compilation
- (b) Le code provoque un erreur à l'exécution
- (c) Il affiche C:g()
- (d) Il affiche A:g()
- (e) Il affiche B:g()

18. On considère les classes:

```
class D {public int val; public D(int i){val=i;}}
class E extends D{public E(int i){super(i);}}
```

le code :

```
D[] t1= new D[4]; t1[0]=new D(1); t1[1]=new E(3); System.out.println(t1[0].val+" "+t1[1].val);
```

- (a) provoque une erreur à la compilation
- (b) affiche 1 3
- (c) provoquent une erreur à l'exécution

19. L'interface: `interface I{void f();}` avec le morceau de code:

```
(new I){public void f(){System.out.println("bravo");}}.f();
```

- (a) provoque une erreur à la compilation
- (b) affiche bravo

20. On considère le code suivant:

```
class A {
    void f() {System.out.println("f de A"); g(); }
    void g() {System.out.println("g de A"); }
}
class B extends A {void g() {System.out.println("g de B");super.f(); }}
class C extends B {void f() {super.f(); }}
```

et la séquence de code `A a = new C(); ((A)a).f();` quelle est la bonne réponse:

- (a) Le code provoque une erreur à la compilation ou à l'exécution
- (b) Il affiche f de A puis g de A