

### Gruppo 4

1) È necessario scrivere un metodo che si comporti come un predicato (restituisca vero o falso) e che analizzi un carattere passato in ingresso. Il predicato deve riconoscere se il carattere passato come argomento sia alfabetico, maiuscolo o minuscolo. Quale delle seguenti realizzazioni è corretta:

1. `public boolean isAlpha (char c) {  
    if ( (c >= 'a' && c <= 'z') || (c <= 'a' && c >= 'z') ) return true;  
    return false;}`
2. `public boolean isAlpha (char c) {  
    if ( (c >= 'A' || c <= 'Z') && (c >= 'a' && c <= 'z') ) return true;  
    return false;}`
3. `public boolean isAlpha (char c) {  
    if ( (c >= 'a' & c <= 'z') || (!c >= 'A' & !c <= 'Z') ) return true;  
    return false;}`
4. **`public boolean isAlpha (char c) {  
    if ( (c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z') ) return true;  
    return false;}`**

2) Se in un metodo compaiono le seguenti istruzioni:

```
try {file = new FileWriter (args[0]);}  
catch (IOException e) {  
    System.out.println("Errore. Problema nella apertura file."); }
```

quale delle seguenti affermazioni è falsa?

1. Il metodo deve prevedere una clausola throws IOException.
2. Il metodo può anche causare un'eccezione IOException ma non può terminare.
3. Il metodo restituisce sempre un oggetto del tipo previsto nella sua definizione.
4. **Il programma intero potrebbe terminare a causa di un'eccezione IOException.**

3) In un frammento di codice si usano le classi ClasseA, ClasseB (che eredita da ClasseA), e ClasseC (che eredita da ClasseB) con le seguenti istruzioni nell'ordine indicato:

```
ClasseA a; ClasseB b; ClasseC c;
```

```
a = new ClasseA (); // 1
```

```
b = new ClasseB (); // 2
```

```
a = b; // 3
```

```
c = null; // 4
```

Quale affermazione è corretta?

1. L'assegnamento 4 è errato.
2. L'assegnamento 3 è errato.
3. **L'assegnamento 3 rende inutile l'assegnamento 1.**
4. Gli assegnamenti 3 e 4 sono errati.

4) Se un metodo ha la seguente definizione:

```
public DListNode extract() throws ListEmptyException;
```

Quale fra le seguenti affermazioni è più corretta?

1. Il metodo extract restituisce un oggetto di tipo DListNode o un'eccezione.

2. **Il metodo extract restituisce un oggetto di una qualunque sottoclasse di DListNode o un'eccezione.**

3. Il metodo extract restituisce un oggetto di un tipo qualunque inclusi i primitivi.

4. Il metodo extract restituisce un oggetto di una qualunque sottoclasse di DListNode inclusi i primitivi o un'eccezione.

5) Indicare la risposta corretta per l'ambiente Java.

1. **L'uso del compilatore e dell'interprete del bytecode consente la massima portabilità tra architetture diverse.**

2. L'uso dell'interprete consente la massima efficienza anche per architetture diverse.

3. La esecuzione del bytecode consente di evitare errori che un compilatore non avrebbe potuto evitare.

4. La compilazione on-the-fly insieme con la interpretazione on-the-spot rendono Java non comparabile in efficienza con il C++.

6) Data la seguente classe:

```
public class Test extends Base {  
    public Test() {  
        super(); }  
    public Test(float j, int k) {  
        super(j, k); }  
}
```

Quale dei seguenti costruttori deve esistere esplicitamente nella definizione della classe Base?

1. `Base() { }`
2. `Base(int j) { }`
3. nessuno
4. **`Base(float x, int l) { }`**

7) Se in una classe trovate le seguenti linee di codice:

```
public void windowClosed(WindowEvent e){}  
public void windowClosing(WindowEvent e){}  
public void windowOpened(WindowEvent e) {}  
public void windowIconified(WindowEvent e) {}  
public void windowDeiconified(WindowEvent e) {}  
public void windowActivated(WindowEvent e) {}  
public void windowDeactivated(WindowEvent e) {}  
allora in tale classe
```

1. Si crea una finestra che viene sollecitata per ogni evento di finestra e che ha i metodi di gestione non ancora specificati.
2. Si intende gestire la finestra relativa in modo diverso dal default per ogni evento.
3. Si intende gestire la finestra relativa in modo diverso dal default per il caso di trasformazione della finestra in icona.
4. **Si crea una finestra che viene mostrata fino alla terminazione del programma e che ricorre al default per la gestione.**

8) Cosa esegue il main, invocato con la linea di comando

```
pack.MainClass pippo pluto
public class MainClass {
    public static void main(String[] args){
        int as = 0, x=4;
        if (args.length != 2){
            System.out.println ("Problemi di argomenti\n");
            System.exit(1);}
        else
            for (int i=0; i < args.length; i++)
                if (args[i].charAt(x) == 'o') as+= args[i].length();
        System.out.println (as); }
    }
```

1. Il processo di esecuzione stampa la stringa di errore e esce con una condizione di errore.
2. Il processo di esecuzione stampa una condizione di errore.
3. **Il processo di esecuzione stampa 10.**
4. Il processo di esecuzione stampa 0.

9) Quale affermazione è corretta, in relazione alla visibilità di una variabile priv dichiarata private e una variabile prot dichiarata protected? Per sottoclassi dirette si intendano solo le classi che ereditano esplicitamente da una classe; per sottoclassi si intendano tutte le possibili classi che ereditano sia in modo diretto che indiretto.

1. **La variabile priv può essere usata e modificata solo nella classe di definizione, la prot può essere usata e modificata solo nella classe di definizione e nelle sottoclassi.**
2. La variabile prot può essere usata e modificata solo nella classe di definizione, la priv può essere usata e modificata solo nella classe di definizione e nelle sue sottoclassi dirette.
3. La variabile priv può essere usata e modificata solo dalla classe di definizione, la prot può essere usata e modificata solo dalle sottoclassi dirette della classe di definizione.
4. La variabile prot non può essere usata e modificata neanche nella sottoclasse di definizione, la priv può essere usata e modificata solo nelle sottoclassi dirette della classe di definizione.

10) Quale delle seguenti definizioni di filtro è la più corretta?

1. Il filtro è un programma che gestisce un insieme di eccezioni, facendone filtrare a tempo di esecuzione verso le classi che lo hanno invocato solo il sottoinsieme di eccezioni non catturate e processate all'interno del filtro stesso.
2. **Il filtro è un programma che al momento dell'esecuzione deve controllare gli argomenti passati sulla linea di comando e lavora trattando tutto il file di ingresso fino a produrre un file di uscita; un filtro ben fatto tende a impegnare poche risorse di memoria e a portare le informazioni in uscita prima possibile.**
3. Il filtro è un programma che al momento dell'esecuzione deve controllare gli argomenti passati sulla linea di comando e lavora trattando tutto il file di ingresso fino a produrre un file di uscita che deve sempre essere di dimensione inferiore al file di ingresso; un filtro ben fatto tende a impegnare poche risorse di memoria e a portare le informazioni in uscita prima possibile.
4. Il filtro è un programma che al momento dell'esecuzione deve controllare gli argomenti passati sulla linea di comando e lavora processando il primo argomento di invocazione fino a produrre un file di uscita; un filtro ben fatto tende a impegnare poche risorse di memoria e a portare le informazioni in uscita prima possibile.

11) Dato il seguente codice:

```
class A {
    private int x; protected double y; public Object o;
    public int m1(boolean t) { x = 1; return x; }
    public int m2() { x = 2; return x; }
}
abstract class B extends A {
    public void m1() { y = 12.5; }
    public int m3() { y = 0.5; return (int)y; }
}
class C extends B {
    protected double x; private int y; public Object o1;
    public void m1() { ... }
    public int m3(int x) { this.x = m2(); return (int)this.x; }
}
```

che cosa stampa la linea di codice System.out.println(o.m3(5));

dove o è un oggetto di classe C?

1. 5.0.
2. **2.**
3. errore di compilazione perchè int m3(int) non è compatibile con int m3().
4. 2.0.

12) Un frammento di codice usa variabili di classi che implementano le interfacce IA, IB, e IC (le classi sono A, B, e C, rispettivamente). Tutte le interfacce ereditano dall'interfaccia Serializable.

```
A a; B b; C c; Serializable i;
```

1. Non si possono fare assegnamenti tra le variabili a, b, c, i.
2. Si possono fare assegnamenti solo tra le variabili a, b, c.
3. **Si possono fare assegnamenti diretti da a, b, c alla variabile i (p.es. i = a).**
4. Si possono fare assegnamenti diretti solo dalla variabile i alle variabili a, b, c (p.es. a = i).

13) Si considerino le clausole seguenti:

```
1) import myPackage.packA.*;
2) import java.lang.utilities.*;
3) import myPackage.packA.packB.*;
```

in un file sorgente:

1. le clausole sono inconsistenti e non corrette.
2. **le clausole possono essere inserite in qualunque ordine.**
3. la clausola 1 deve essere inserita prima della 3, l'altra è indifferente.
4. la clausola 2 non è significativa: è aggiunta a default.

14) Si considerino le linee di codice:

```
JFrame f = new JFrame("Esame");
Container c = f.getContentPane();
MyPanel p = new MyPanel();
c.add(p);
f.setSize(220,150);
f.addWindowListener(new Terminator());
```

1. La finestra relativa al frame f viene mostrata in un ciclo di esecuzioni del comando show.

2. La finestra relativa al pannello p viene mostrata solo se si attua un ciclo di show del pannello.
3. **La finestra relativa al frame f non viene mostrata fino alla esecuzione del comando show.**
4. La finestra relativa al pannello p viene mostrata solo se si attua un ciclo di show del container c.

15) Si considerino:

una classe A che definisce un costruttore A;

una sottoclasse B che eredita da A e definisce un costruttore che invoca il costruttore di A;

una sottoclasse C che eredita da A e definisce un costruttore che non invoca il costruttore di A.

Quale affermazione è vera?

1. I costruttori della classe B e della classe C non sono corretti: essendo dello stesso livello devono o invocare tutti il costruttore o non invocarli entrambi.
2. **Il costruttore della classe B deve invocare il costruttore di A in modo esplicito come prima istruzione del costruttore.**
3. Il costruttore della classe C deve invocare il costruttore di A in modo esplicito come prima istruzione del costruttore.
4. Il costruttore della sola classe B può ricorrere alla invocazione del costruttore della classe A in modo implicito.