Cognome	
Nome	
Matricola	
Anno di corso	

Linguaggi di Programmazione

a.a. 2013-2014

Esame del 13-01-2014

Istruzioni: Per ognuna delle seguenti 30 domande sono elencate 5 possibili risposte, di cui soltanto una è corretta. Marcare con una X la casella della risposta che si ritiene corretta. Non è consentito marcare piú di una risposta.

Ogni risposta corretta vale 1 punto; ogni risposta errata vale -0.25 punti; ogni risposta non fornita vale 0 punti.

Non è consentito l'uso di appunti, né di altre forme di informazione. Tempo a disposizione: 90 minuti.

---

1. Quale di questi non è un tipo fondamentale di Java ?

- [x] `string`
- [ ] `char`
- [ ] `boolean`
- [ ] `short`
- [ ] `byte`

2. Quale delle seguenti affermazioni è falsa ?

- [x] Una classe deve necessariamente avere lo stesso nome del file `.java` che la contiene
- [ ] Un file `.java` può contenere piú di una classe
- [ ] Un file `.java` non può contenere due classi pubbliche
- [ ] Se un file `.java` contiene una classe pubblica, questa deve avere lo stesso nome del file
- [ ] Una classe non può contenere un'altra classe al suo interno

3. Se per un elemento non viene specificato nessun modificatore di accesso (`private`, `protected`, `public`) allora quell'elemento:

- [x] è accessibile solo dalle classi dello stesso package
- [ ] è accessibile solo dalla classe che lo contiene e dalle sottoclassi di questa
- [ ] è accessibile solo dalla classe che lo contiene
- [ ] è accessibile da qualunque classe
- [ ] provoca un errore di compilazione

4. Quale delle seguenti affermazioni è falsa ?

- [x] Una variabile locale di tipo numerico è inizializzata per default a `0`
- [ ] Una variabile locale può contenere un oggetto
- [ ] Una variabile locale non può essere contrassegnata come `static`
- [ ] Le variabili locali cessano di esistere alla chiusura del metodo in cui sono dichiarate
- [ ] Un metodo non può accedere alla variabili locali di un altro metodo

5. Qual è la sintassi corretta per il metodo `main` ?

- [x] `public static void main (String[] args) { }`
- [ ] `public static void Main (String[] args) { }`
- [ ] `public static void main (String, args) { }`
- [ ] `public static void main (String[], args) { }`
- [ ] `public static void Main (String, args) { }`

6. Quale dei seguenti è un metodo costruttore ?

- [x] `public A () { }`
- [ ] `public static A () { }`
- [ ] `public void A () { }`
- [ ] `public static void A () { }`
- [ ] `public new A () { }`

7. All'interno di una stessa classe, quale tra le seguenti dichiarazioni del metodo `a` provoca un errore di compilazione ?

- [x] Vengono dichiarati sia `void a (int i)` che `void a (int j)`
- [ ] Vengono dichiarati sia `void a ()` che `void a (int i)`
- [ ] Vengono dichiarati sia `void a (int i)` che `void a (int i, int j)`
- [ ] Vengono dichiarati sia `void a (int j)` che `void a (int i, double j)`
- [ ] Vengono dichiarati sia `void a (double i)` che `void a (int i)`

8. Quale delle seguenti affermazioni è corretta ?

- [x] Un costruttore non può essere dichiarato con l'attributo `static`
- [ ] Un costruttore non può essere dichiarato senza l'attributo `public`
- [ ] Un costruttore non può essere dichiarato senza attributi modificatori
- [ ] Un costruttore non può essere dichiarato senza parametri
- [ ] Un costruttore non può essere dichiarato con un parametro di tipo non primitivo

9. Quale delle seguenti istruzioni di ciclo esegue sempre almeno una volta il proprio corpo?

- [x] `do while`
- [ ] `for`
- [ ] `while`
- [ ] `switch`
- [ ] Nessuna delle altre opzioni é corretta

10. Cosa stampa l'istruzione `System.out.println( +1+"+1" );` ?

- [x] `1+1`
- [ ] `11`
- [ ] Niente, perché provoca un errore di compilazione
- [ ] `+11`
- [ ] `1++1`

11. Quale delle seguenti affermazioni è falsa ?

- [x] Una stringa è un oggetto presente in qualunque classe
- [ ] Una stringa è un oggetto, ma può essere istanziata senza il `new`
- [ ] Una stringa può essere istanziata con il `new`
- [ ] Una stringa può essere istanziata con un letterale racchiuso tra apici doppi
- [ ] Una stringa non può essere istanziata con un letterale racchiuso tra apici singoli

12. Sia `s` un riferimento alla stringa "`prova 123`". Cosa restituisce l'espressione `s.substring(s.length()-3)` ?

- [x] La stringa "`123`"
- [ ] La stringa "` 123`"
- [ ] Un errore di compilazione
- [ ] La stringa "`23`"
- [ ] L'intero 6

13. Subito dopo l'istruzione `boolean [] b = new boolean [1]`, quanto vale l'espressione `b[0]` ?

- [x] Vale `false`
- [ ] Vale `null`
- [ ] Genera un errore perché la variabile `b[0]` non è stata inizializzata
- [ ] Vale `true`
- [ ] Vale `0`

14. Dopo aver dichiarato ed inizializzato un array a mediante le istruzioni `int[] a = new int [4]; for (int i = 3; i >= 0; i-) a[3-i] = 2*i;` qual è il valore ritornato da `a[2]` ?

- [x] `2`
- [ ] `4`
- [ ] `1`
- [ ] `3`
- [ ] `0`

15. Dato l'array `int [] n = {1,2,3,4,5,6}`, cosa stampa l'istruzione `System.out.println( (n[1] + n[3]) )` ?

- [x] `6`
- [ ] `4`
- [ ] `1+3`
- [ ] `2+4`
- [ ] Nulla, perché il compilatore segnala un errore

16. Sia `a` un ArrayList avente per parametro la classe `A`, e sia `b` un oggetto della classe `A`. Quale delle seguenti affermazioni è corretta ?

- [x] Per aggiungere `b` ad `a` si deve scrivere l'istruzione `a.add(b)`
- [ ] Per aggiungere `b` ad `a` si deve scrivere l'istruzione `add(a,b)`
- [ ] Per aggiungere `b` ad `a` si deve scrivere l'istruzione `Arrays.add(a,b)`
- [ ] Per aggiungere `b` ad `a` si deve scrivere l'istruzione `a.add(b,a.length)`
- [ ] Per aggiungere `b` ad `a` si deve scrivere l'istruzione `a = a.add(b)`

17. Quale delle seguenti affermazioni è corretta ?

- [x] Se `a` è una stringa, la sua dimensione è data da `a.length()`
- [ ] Se `a` è una stringa, la sua dimensione è data da `a.length`
- [ ] Se `a` è un array, la sua dimensione è data da `a.length()`
- [ ] Se `a` è un ArrayList, la sua dimensione è data da `a.length()`
- [ ] Se `a` è un ArrayList, la sua dimensione è data da `a.length`

18. Quale delle seguenti istruzioni non genera un errore di compilazione ?

- [x] `ArrayList<Integer> num = new ArrayList<Integer>();`
- [ ] `ArrayList<Int> num = new ArrayList<Int>();`
- [ ] `ArrayList<int> num = new ArrayList<int>();`
- [ ] `ArrayList<int> num = new ArrayList;`
- [ ] `ArrayList<Integer> num = new ArrayList;`

19. Sia `B` una sottoclasse di `A`, e sia `a` un attributo `private` di `A`. Quale delle seguenti affermazioni è corretta ?

- [x] `B` eredita `a`, ma non può accedervi direttamente
- [ ] `B` eredita `a` e può accedervi direttamente in quanto `B` è sottoclasse di `A`
- [ ] `B` non eredita `a`, perché eredita solo gli attributi `public` di `A`
- [ ] `B` non eredita `a`, perché eredita solo gli attributi `protected` di `A`
- [ ] `B` eredita `a` solo se `A` e `B` si trovano nello stesso package

20. Sia `B` una sottoclasse di `A`, e sia `a` un metodo `protected` di `A`. Quale delle seguenti affermazioni è corretta ?

☒ È sempre possibile invocare `a` su un oggetto di `B`

☐ Non è possibile invocare `a` su un oggetto di `B`

☐ È possibile invocare `a` su un oggetto di `B`, a meno che `a` non sia stato ridefinito in `B`

☐ È possibile invocare `a` su un oggetto di `B` solo se `a` è stato ridefinito in `B`

☐ Non è possibile invocare `a` su un oggetto di `B`, a meno che l'oggetto non sia contenuto in una variabile della classe `A`

21. Quale delle seguenti affermazioni è corretta ?

☒ Il modificatore `private` rende possibile l'accesso solo all'interno della classe

☐ Il modificatore `private` rende possibile l'accesso solo all'interno della classe e delle sue sottoclassi

☐ Il modificatore `private` rende possibile l'accesso solo all'interno della classe e del package che la contiene

☐ Il modificatore `private` rende possibile l'accesso da qualunque classe

☐ Il modificatore `private` rende impossibile l'accesso da parte di chiunque

22. Quale delle seguenti affermazioni è corretta ?

☒ Il modificatore `public` rende possibile l'accesso da qualunque classe

☐ Il modificatore `public` rende possibile l'accesso solo all'interno della classe

☐ Il modificatore `public` rende possibile l'accesso solo all'interno della classe e del package che la contiene

☐ Il modificatore `public` rende possibile l'accesso solo all'interno della classe e delle sue sottoclassi

☐ Il modificatore `public` rende impossibile l'accesso da parte di chiunque

23. Qual è il comportamento del seguente programma ?

```
class B {
   int i;
   void set (int j) {
      this.i = j;
   }
}
class A extends B {
    int i;
    public static void main (String[] args) {
        A a = new A();
        a.set(5);
        System.out.println(a.i);
    }
 }
```

☒ Stampa `0` perché la variabile `i` è stata adombrata in `A`

☐ Stampa `5`

☐ Dà un errore di compilazione perché la variabile `i` è stata ridefinita in `A`

☐ Dà un errore di compilazione perché la variabile `set` non è accessibile in `A`

☐ Dà un errore di compilazione perché la variabile `i` non è stata inizializzata

24. Quale delle seguenti affermazioni è falsa ?

☒ Un costruttore può essere dichiarato `final`; in tal caso, non è sovrascrivibile

☐ Una variabile di istanza può essere dichiarata `final`; in tal caso, non è modificabile

☐ Una variabile locale può essere dichiarata `final`; in tal caso, non è modificabile

☐ Una classe può essere dichiarata `final`; in tal caso, non è derivabile

☐ Un metodo può essere dichiarato `final`; in tal caso, non è sovrascrivibile

25. Quale delle seguenti affermazioni è falsa ?

☒ Una sottoclasse di una classe `abstract` deve definire tutti i metodi `abstract`

☐ Una classe può estendere una sola classe `abstract`

☐ Una classe `abstract` non può essere istanziata

☐ Oltre a metodi `abstract`, una classe `abstract` può contenere dati e metodi non `abstract`

☐ Una classe `abstract` può avere un costruttore, che potrà essere chiamato dai costruttori delle classi derivate

26. Quale delle seguenti affermazioni è falsa ?

☒ Una classe `abstract` non può implementare un'interfaccia

☐ Un'interfaccia può essere derivata

☐ Un'interfaccia deve essere `public`

☐ Un'interfaccia non può contenere un metodo `final`

☐ Se una classe implementa un'interfaccia, allora ogni sua sottoclasse implementa automaticamente la stessa interfaccia

27. Sia `a` un metodo suscettibile di lanciare un'eccezione, e nel quale non è stato installato un gestore per tale eccezione. Allora nella dichiarazione del metodo :

☒ bisogna dichiarare l'eccezione che il metodo potrebbe lanciare, per mezzo della clausola `throws`, solo se l'eccezione è di tipo controllato

☐ bisogna dichiarare l'eccezione che il metodo potrebbe lanciare, per mezzo della clausola `throws`, sia se l'eccezione è di tipo controllato che non controllato

☐ bisogna dichiarare l'eccezione che il metodo potrebbe lanciare, per mezzo della clausola `throw`, solo se l'eccezione è di tipo controllato

☐ bisogna dichiarare l'eccezione che il metodo potrebbe lanciare, per mezzo della clausola `throw`, sia se l'eccezione è di tipo controllato che non controllato

☐ non è necessario dichiarare l'eccezione che il metodo potrebbe lanciare, perché questa verrà propagata all'esterno

28. Quale delle seguenti affermazioni è vera ?

☒ Se in un metodo è presente un blocco `try`, questo deve essere seguito necessariamente da almeno un blocco `catch` oppure da un blocco `finally`

☐ In un metodo può essere presente al piú un blocco `try`

☐ In un metodo può essere presente al piú un blocco `finally`

☐ Se in un metodo è presente un blocco `try`, questo deve essere seguito necessariamente da almeno un blocco `catch`

☐ Se in un metodo è presente un blocco `try`, questo deve essere seguito necessariamente da un blocco `finally`

29. Qual è l'output del seguente programma ?

```
public class Test {
    public static void a() throws Exception {
        try { throw new Exception(); }
        finally { System.out.print("finally "); }
    }
    public static void main(String args[]) {
        try { a(); }
        catch (Exception e) { System.out.print("exception "); }
        System.out.print("finished");
    }
}
```

☒ `finally exception finished`

☐ `finally`

☐ Si ottiene un errore di compilazione

☐ `exception finished`

☐ `finally exception`

30. Quale delle seguenti affermazioni è vera ?

☒ L'istruzione `catch(X x)` può catturare sottoclassi di `X`, dove `X` è una sottoclasse di `Exception`

☐ La classe `Error` è una sottoclasse di `RuntimeException`

☐ Tutte le sottoclassi di `RuntimeException` sono classi di eccezioni controllate

☐ Ogni istruzione che può lanciare un'eccezione deve trovarsi all'interno di un blocco `try`

☐ Se non si installa un gestore per ogni eccezione controllata, si ottiene un errore di compilazione