Corso: Fondamenti di Informatica 1 (gruppo 8-9)
Corsi di laurea: Area dell'Informazione
Questionario di Autovalutazione
Laboratorio 6

Domanda n. 1

Argomento: Compilatore e interprete

Peso: 1

Pubblica: N

Domanda: Con riferimento a Java, quale delle seguenti affermazioni e' corretta?

Risposte:

1. La memoria allocata per le variabili locali appartiene al java heap

   Punteggio: 0

2. Non e' mai possibile allocare memoria durante l'esecuzione di un programma

   Punteggio: 0

3. La memoria allocata per le variabili locali appartiene al run time stack (o java stack)

   Punteggio: 1

Spiegazione:
Le variabili locali sono allocate nel run time stack.

Domanda n. 2

Argomento: Variabili e tipi

Peso: 1

Pubblica: N

Domanda: Dal punto di vista sintattico, System.in e'

Risposte:

1. un metodo della classe System

   Punteggio: 0

2. una campo statico della classe System

   Punteggio: 1

3. una variabile globale

   Punteggio: 0

4. una costante intera

   Punteggio: 0

Spiegazione:
System.in è una variabile statica della classe System. Vedere documentazione della java platform API.

Domanda n. 3

Argomento: Classi

Peso: 1

Pubblica: N

Domanda: In Java, che cosa indica la parola chiave "this" all'interno di un metodo di esemplare?

Risposte:

1. l'utente che sta eseguendo il programma

   Punteggio: 0

2. la classe di cui viene eseguito il metodo

   Punteggio: 0

3. il parametro implicito del metodo

   Punteggio: 1

4. il risultato calcolato dal metodo

Punteggio: 0

Spiegazione:
"this" indica il parametro implicito.

Domanda n. 4

Argomento: Classi

Peso: 1

Pubblica: N

Domanda: Che tipo di errore provoca l'esecuzione della classe seguente?

```
public class MyComplex
{  private double real;
   private double imm;
   public MyComplex(double r, double i) { real = r; imm = i; }
   public static void main(String[] args)
   {  MyComplex[] v = new MyComplex[10];

      for(int i = 0; i < v.length; i++)
      {  v[i].real = 0;
         v[i].imm = 0;
      }
   }
}
```

Risposte:

1. NumberFormatException

   Punteggio: 0

2. ArrayIndexOutOfBoundsException

   Punteggio: 0

3. NullPointerException

   Punteggio: 1

4. Nessun errore

   Punteggio: 0

Spiegazione: l'enunciato

```
MyComplex[] v = new MyComplex[10];
```

crea e inizializza un array che contiene dieci riferimenti a oggetti di classe MyComplex inizializzati a null:

```
v -> {null,null,null,null,null,null,null,null,null,null}
```

Quando nel ciclo viene eseguito l'enunciato
```
v[0].real = 0;
```
viene generata l'eccezione
```
NullPointerException
```

Domanda n. 5

Argomento: Classi

Peso: 1

Pubblica: N

Domanda: Un metodo non statico

Risposte:

1. ha sempre un parametro implicito

   Punteggio: 1

2. ha un numero arbitrario di parametri impliciti, dipende da quanti ne sono definiti nella sua intestazione

   Punteggio: 0

3. ha un parametro implicito soltanto se il suo valore di ritorno non e' void

   Punteggio: 0

Spiegazione:
un metodo non statico (di esemplare) ha sempre un parametro implicito.

Domanda n. 6

Argomento: Iterazioni, array e stringhe

Peso: 1

Pubblica: N

Domanda: Quale e' il risultato dell'esecuzione del seguente metodo in Java?

```
public static int fattoriale(int n)
{  int p = 1;
   while (n > 0)
   { p = p * n;
     n = decrementa(n);
   }
   return p;
}
private static int decrementa(int k)
{ return k - 1; }
```

Risposte:

1. il fattoriale di n se n > 0, altrimenti 1

   Punteggio: 1

2. sempre 1, per qualsiasi valore di n

   Punteggio: 0

3. il metodo non termina a causa di un ciclo infinito

   Punteggio: 0

Spiegazione:
Il metodo calcola il fattoriale di n, se n > 0, 1 altrimenti. Eseguire il ciclo.

Domanda n. 7

Argomento: Variabili e tipi

Peso: 1

Pubblica: N

Domanda: Il seguente frammento di codice in Java

```
int k = 'A' + 1;
char c = k;
System.out.println(c);
```

Risposte:

1. genera un errore in compilazione all'enunciato int k = 'A' + 1;

Punteggio: 0

2. genera un errore in compilazione all'enunciato char c = k;

   Punteggio: 1

3. genera un errore in compilazione all'enunciato System.out.println(c);

   Punteggio: 0

4. non genera errori in compilazione ma lancia un'eccezione in esecuzione

   Punteggio: 0

5. non genera errori in compilazione e non lancia eccezioni in esecuzione

   Punteggio: 0

Spiegazione:
L'enunciato char c = k; genera in compilazione un errore dal momento che la conversione da int a char non puo' essere automatica (Possible Loss of Precision).

Domanda n. 8

Argomento: Variabili e tipi

Peso: 1

Pubblica: N

Domanda: Il seguente frammento di codice in Java

```
int k = 'A' + 1;
char c = (char)k;
System.out.println(c);
```

Risposte:

1. genera un errore in compilazione all'enunciato int k = 'A' + 1;

   Punteggio: 0

2. genera un errore in compilazione all'enunciato char c = (char) k;

Punteggio: 0

3. genera un errore in compilazione all'enunciato System.out.println(c);

   Punteggio: 0

4. non genera errori in compilazione ma lancia un'eccezione in esecuzione

   Punteggio: 0

5. non genera errori in compilazione e non lancia eccezioni in esecuzione

   Punteggio: 1

Spiegazione:
Il frammento di codice non contiene né errori sintattici né logici. In esecuzione non genera eccezioni, ma stampa il carattere successivo ad 'A' nel codice usato per rappresentare i caratteri.

Domanda n. 9

Argomento: Iterazioni, array e stringhe

Peso: 1

Pubblica: N

Domanda: Relativamente al frammento di codice seguente, dire quale affermazione e' corretta

```
int[] a = new int[10];
a[0] = 1;
int[] b = a;
b[0] = 2;
System.out.println(a[0]);
```

Risposte:

1. Il codice genera un errore in compilazione all'enunciato int[] b = a;

   Punteggio: 0

2. Il codice compila correttamente ma genera in esecuzione l'eccezione NullPointerException

Punteggio: 0

3. Il codice compila correttamente e in esecuzione stampa il numero 1

   Punteggio: 0

4. Il codice compila correttamente e in esecuzione stampa il numero 2

   Punteggio: 1

5. Nessuna delle precedenti affermazioni e' corretta

   Punteggio: 0

Spiegazione:
Il codice compila ed esegue stampando il numero 2. Gli enunciati

```
b[0] = 2;
System.out.println(a[0]);
```

sono, rispettivamente, un enunciato di assegnazione del primo elemento dell'array ed un enunciato di accesso al primo elemento dell'array.

Domanda n. 10

Argomento: Variabili e tipi

Peso: 1

Pubblica: N

Domanda: Relativamente al frammento di codice seguente, dire quale affermazione e' corretta

```
int n;
int k = 0;
while ( k < 10)
{
   n = k;
   k++;
}

System.out.println(n);
```

Risposte:

1. compila correttamente e in esecuzione stampa il numero 10

Punteggio: 0

2. compila correttamente e in esecuzione stampa il numero 9

   Punteggio: 0

3. compila correttamente ma in esecuzione genera l'eccezione ArithmeticException

   Punteggio: 0

4. genera in compilazione un errore per la possibile mancata inizializzazione della variabile n

   Punteggio: 1

5. nessuna delle precedenti risposte e' corretta

   Punteggio: 0

Spiegazione:
Il frammento di codice genere un errore in compilazione all'enunciato

```
   System.out.println(n);
(Variable n might not have been initialized)
```

per la possible mancata inizializzazione della variabile n.
Infatti, se il corpo del ciclo while non venisse eseguito, l'enunciato userebbe la variabile n senza che questa sia stata precedentemente inizializzata.

Domanda n. 11

Argomento: Variabili e tipi

Peso: 1

Pubblica: N

Domanda: Quale affermazione e' corretta relativamente al seguente metodo

```
private int findIndex(int[] a, int target)
{
   for (int i = 0; i < a.length; i++)
      if (a[i] == target)
         return i;


}
```

Risposte:

1. compila e restituisce l'indice dell'elemento target nell'array a

   Punteggio: 0

2. compila e restituisce l'elemento target dell'array a

   Punteggio: 0

3. compila ma in esecuzione genera sempre l'eccezione NullPointerException

   Punteggio: 0

4. compila ma in esecuzione genera l'eccezione ArrayIndexOutOfBoundsException

   Punteggio: 0

5. nessuna delle precedenti affermazioni e' corretta

   Punteggio: 1

Spiegazione:
In compilazione viene segnalato un errore perché manca un enunciato return. Infatti se il valore target non viene trovato nell vettore a, l'esecuzione del programma esce dal ciclo for. Poiché il metodo non ritorna void, deve essere esplicitamente programmato un enunciato return, ad esempio

```
return -1;
```

Domanda n. 12

Argomento: Variabili e tipi

Peso: 1

Pubblica: N

Domanda: Quale fra i seguenti e' l'enunciato corretto per generare un'eccezione quando n < 0:

Risposte:

1. if (n < 0) throw IllegalArgumentException();

   Punteggio: 0

2. if (n < 0) throw IllegalArgumentException;

   Punteggio: 0

3. if (n < 0) throw new IllegalArgumentException();

   Punteggio: 1

4. if (n < 0) System.out.println(new IllegalArgumentException());

   Punteggio: 0

5. nessuna delle precedenti

   Punteggio: 0

Spiegazione:

```
if (n < 0) throw new IllegalArgumentException();
```

Vedere la sintassi di Java.

Domanda n. 13

Argomento: Variabili e tipi

Peso: 1

Pubblica: N

Domanda: Quale affermazione e' corretta relativamente alla seguente classe

```
public class MyClass
{
   private int x;

   public MyClass(int unX)
   {
      int x = unX;
   }
}
```

Risposte:

1. il costruttore inizializza la variabile di esemplare al valore unX

Punteggio: 0

2. la variabile di esemplare e' sempre inizializzata a zero per ciascun esemplare della classe perche' la variabile di esemplare e' messa in ombre

   Punteggio: 1

3. non si possono inizializzare le variabili di esemplare nel costruttore

   Punteggio: 0

4. nessuna delle risposte precedenti e' corretta

   Punteggio: 0

Spiegazione:
L'enunciato

```
int x = unX;
```

del costruttore definisce una variabile locale x che mette in ombra la variabile di esemplare della classe con lo stesso nome.
Il risultato è che la variabile di esemplare di ciascun esemplare non viene inizializzata dal costruttore programmato, ma dal costruttore predefinito che la inizializza a 0.