

Laboratorio di Programmazione 1 [Java]

Prova di esame - 7 Settembre 2010

Tempo massimo: 50 minuti

Si implementino in Java le classi **Tavolo** e **Ristorante**.

La *classe Tavolo* ha i seguenti attributi:

- **numero** (un intero contenente il numero del tavolo)
- **capienza** (un intero contenente il numero massimo di coperti del tavolo)

ed i seguenti metodi:

- *costruttore* che crea un oggetto della *classe Tavolo* assegnando numero e capienza.
- metodi "get" per tutti gli attributi, cioè metodi che restituiscono i valori di ciascun attributo;
- metodo "toString"

La *classe Ristorante* ha i seguente attributi:

- **tavoli** (un array di Tavolo)
- **nTavoli** (un intero contenente il numero di tavoli attualmente presenti nel ristorante)

ed i seguenti metodi:

- *costruttore* che crea un oggetto della *classe Ristorante* prendendo in input il numero massimo di tavoli che quel ristorante può contenere
- metodo "addTavolo", che preso in input un tavolo lo aggiunge ai tavoli del ristorante.
- metodo "toString", che sfruttando il metodo toString della classe Tavolo restituisce la descrizione di tutti i tavoli presenti.

Esempio di test:

```
public class Test {  
    public static void main(String [] args) {  
        Ristorante r = new Ristorante(40);  
        r.addTavolo(new Tavolo(1,10));  
        r.addTavolo(new Tavolo(5,8));  
        System.out.println (r);  
    }  
}
```

Laboratorio di Programmazione 1 [Java]

Prova di esame - 7 Settembre 2010

Cognome	Nome	Matricola

Tempo disponibile: 1 ora (più il tempo eventualmente risparmiato nella prova pratica)

Esercizio 1 (4 punti)

Dato il seguente codice Java:

```
public class Test {  
public static void main (String args []) {  
    int i=23, j=i-1;  
    if (!(i>=j)) {  
        i+= i+j;  
        System.out.println("i = "+ i++);  
    }  
    System.out.println ("i = " + (++i));  
}
```

Cosa stampa il programma ?

Esercizio 2 (4 punti)

Dato il seguente programma Java:

```
public class Calc {
public static void main (String args []) {
    int total = 0;
    for (int i = 0; total > -30 && total != 1; ++i) {
        int j = 4;
        total = 1;
    }
    System.out.println("j =" + j);
}
```

Quali delle seguenti affermazioni è vera ? **Cerchiare la risposta esatta**

1. Genera un errore a tempo di esecuzione
2. Genera un errore a tempo di compilazione
3. Visualizza "j =4"
4. Non termina mai la sua esecuzione

Esercizio 3 (4 punti)

Dato il seguente codice Java:

```
class Test {
public static void main(String args[]) {
int i,j,somma=0;
for (i = 0; i < 10; i++) {
    for(j = i; j < 10; j++) {
        if(i == j) somma+=i;
    }
}
System.out.println(somma);
}
}
```

Cosa viene visualizzato?

Esercizio 4 (5 punti)

Date le seguenti classi Java:

```
class B {
int x;
public B(int i){
x=i;
}
public int m1(A a){
return a.m1(this);
}
}
class A
{
int y;
public int m1(B b) {
b.x--;
y=b.x;
if (y==0) return 10;
return m1(b);
}
public int m1(A a) {
if (y==0) return 24;
y--;
return (1+a.m1(this));
}
}
public class Main{
public static void main(String args[]){
    B b=new B(18);
    A a=new A();
    System.out.println(b.m1(a));
}
}
```

Cosa viene visualizzato?

Esercizio 5 (18 punti) Rispondere sul foglio protocollo

Si consideri il seguente codice:

```
public class Tavolo {

    private int numero;
    private int capienza;

    public Tavolo (int n, int c){
        numero=n;
        capienza=c;
    }

}
```

```
public int getNumero(){
    return numero;
}

public int getCapienza(){
    return capienza;
}

public String toString(){
    return "Tavolo numero: "+numero+" capienza: "+capienza;
}
}

public class Ristorante {

    private int nTavoli;
    private Tavolo [] tavoli;

    public Ristorante (int max){
        tavoli=new Tavolo[max];
        nTavoli=0;
    }

    public void addTavolo (Tavolo t){
        if (nTavoli < tavoli.length) {
            tavoli[nTavoli++]=t;
        }
    }

    public String toString(){
        String ris ="Numero di tavoli presenti: "+nTavoli+" capienza tavoli:
"+tavoli.length;
        for (int i=0; i< nTavoli;i++){
            ris += "\n";
            ris+=tavoli[i].toString();
        }
        return ris;
    }
}
}
```

Completare il codice in questo modo:

- Scrivere una classe **Prenotazione** che contiene i campi Nome (di tipo stringa), giorno di prenotazione (di tipo intero), ora di inizio prenotazione (di tipo intero) ed ora di fine prenotazione (di tipo intero). [Nota: si assume dunque che sia le date che gli orari sono gestiti tramite numeri interi] **(5 punti)**
- Aggiungere alla classe Tavolo un array di Prenotazione di lunghezza 100 (modificando opportunamente anche il costruttore) **(3 punti)**
- Aggiungere **(10 punti)** alle classe Tavolo e Ristorante tutto ciò che è necessario per gestire l'arrivo di nuove prenotazioni; in particolare, nella classe Ristorante deve essere aggiunto
 - un metodo **addPrenotazione** che *presi in input* il nome (stringa) del prenotante e gli interi corrispondenti al giorno, all'ora di inizio e all'ora di fine della prenotazione e il numero di persone *cerca* un tavolo libero, lo *restituisce* in output come ritorno del metodo, e *aggiunge* la prenotazione opportunamente al tavolo in questione. Se nessun tavolo con le caratteristiche richieste è trovato, *restituisce* -1.