

Matricola\_\_\_\_\_ Cognome\_\_\_\_\_ Nome\_\_\_\_\_

## Fondamenti di Informatica

### Prova d'esame del 22/1/2015

#### Regole d'esame:

1. È vietato parlare con altri studenti
2. È vietato consultare appunti, dispense, libri, in qualunque formato
3. È vietato tenere accesi i telefoni cellulari

#### Esercizio 1

Si consideri il seguente metodo che riceve in input un intero  $n$ . Indicare, spiegandola brevemente, la complessità asintotica di caso peggiore in funzione di  $n$ .

```
public static void esercizio(int n){
    int k=0;
    int i=0;
    while(i<n){
        k+=n*n;
        i++;
    }

    i=0;
    while(i<k){
        System.out.println("i: "+i);
        i++;
    }
}
```

Il primo ciclo while viene eseguito  $n$  volte (per  $i$  che assume tutti i valori interi da 0 a  $n-1$ ). Ad ogni iterazione di tale ciclo si aggiunge alla variabile  $k$  una quantità pari ad  $n*n$ . Poiché la variabile  $k$  viene inizializzata a 0, il valore di  $k$  al termine del primo ciclo for è  $k = \sum_{i=0}^{n-1} n^2 = n^2 \sum_{i=0}^{n-1} 1 = n^3 = O(n^3)$ . Il secondo ciclo for viene eseguito  $k$  volte (per  $i$  che assume tutti i valori interi da 0 a  $k-1$ ). Poiché  $k = O(n^3)$ , la complessità del secondo ciclo è  $O(n^3)$ .

#### Esercizio 2

Scrivere un metodo di classe ricorsivo che riceve in input una matrice quadrata di interi  $a$ , un indice  $j$  (che si può assumere valido), ed altri parametri se necessario, e restituisce la somma degli elementi pari appartenenti alla colonna  $j$ -esima.

#### Note.

1. Una soluzione non ricorsiva non sarà considerata corretta
2. Se ritenuto utile o necessario è possibile scrivere un metodo per l'avvio della ricorsione
3. Il metodo non deve utilizzare variabili che non siano locali

```

public static int sommaPariColonna(int[][] a, int j, int i){
    int somma;
    if(i==a.length)
        somma=0;
    else{
        if(a[i][j]%2==0)
            somma=a[i][j]+sommaPariColonna(a,j,i+1);
        else
            somma=sommaPariColonna(a,j,i+1);
        }
    return somma;
}

public static int sommaPariColonna(int[][] a, int j){
    return sommaPariColonna(a, j, 0);
}

```

### Esercizio 3

Il seguente codice contiene cinque errori. Indicarli spiegando brevemente in che cosa consiste l'errore.

```

public interface I{
    public I(int a, double b);
    public int metodo1();
    public double metodo2();
}

public class C implements I{
    private int a;
    private double b;

    public C(int a, double b){
        this.a=a;
        this.b=b;
    }
    public int metodo1(){
        return a;
    }
    public double metodo3(){
        return b;
    }
}

public class C2 extends C{
    private String c;

    public C2(int a, double b, String c){
        this.a=a;
        this.b=b;
        this.c=c;
    }
}

```

```

    }
    public String metodo4() {
        return c;
    }
}

```

1. Una interface non può avere costruttori
2. La classe C non implementa il metodo metodo2(). Poiché C implementa I, essa deve implementare tutti i metodi definiti in I.
3. L'istruzione this.a=a è errata perché la variabile di istanza a è definita private in C e quindi non può essere acceduta in C2.
4. L'istruzione this.b=b è errata perché la variabile di istanza b è definita private in C e quindi non può essere acceduta in C2.
5. La prima istruzione del costruttore della classe C2 deve essere un'invocazione di un costruttore della superclasse. Tale invocazione può non apparire esplicitamente soltanto a patto che la superclasse sia dotata di un costruttore senza argomenti, ma questo non è il caso della classe C. Si osservi che sostituendo le istruzioni this.a=a e this.b=b con l'invocazione super(a,b) si correggono gli errore 3., 4. e 5.

#### Esercizio 4

Con riferimento all'algoritmo di ordinamento MergeSort si mostri l'esecuzione del metodo merge (che realizza la fusione di due sottosequenze ordinate dell'array) assumendo che esso sia invocato nel seguente modo **merge(dati, temp, 0, 5, 11)** e che **dati** sia il seguente array:

5	8	12	22	23	25	10	13	15	16	17	20
---	---	----	----	----	----	----	----	----	----	----	----

**PER RISPONDERE ALLA DOMANDA SI DEVE MOSTRARE, PER OGNI PASSO, QUALI SONO GLI ELEMENTI (O L'ELEMENTO) CONSIDERATI E COME VIENE MODIFICATO L'ARRAY TEMP.**

#### Esercizio 5

Sia **n** il numero 118. Si rappresenti **n** in binario. Si rappresenti poi il numero **-n** in complemento a due con 8 bit.

118 : 2 = 59 e resto 0

59:2 = 29 e resta 1

29:2 = 14 e resta 1

14:2=7 e resta 0

7:2=3 e resta 1

3:2=1 e resta 1

1:2=0 e resta 1

118=1110110|2

*Questo compito è stato discusso e definito collegialmente dalla commissione d'esame di Fondamenti di Informatica*

118 con 8 bit è 01110110, invertendo tutti i bit si ottiene 118 in complemento a 1 con 8 bit, cioè 10001001. La rappresentazione in complemento a 2 con 8 bit si ottiene sommando 1 alla rappresentazione in complemento a 1, cioè:

$$\begin{array}{rcccccccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & + \\ & & & & & & & 1 & = \\ \hline 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & \end{array}$$