Esame di Programmazione Internet

CdL in Informatica per il Management

Data Appello: 28/04/2021 - Appello Prova

Per chi ha superato il parziale: solo esercizio 4 - tempo a disposizione: 1 ora e ½

Per gli altri: tutti gli esercizi - tempo a disposizione: 3h

INFORMAZIONI PRELIMINARI

L'esame consiste nello scrivere uno o più file *.java compilabili. Elaborati che **non compilano il codice prodotto non** verranno presi in considerazione.

NON è consentito l'uso di testi e appunti.

La scelta dei nomi dei file, se non espressamente indicati, non è rilevante. I file possono essere salvati in directory diverse.

All'inizio di **ogni** file .java mettete come commento, Nome, Cognome, Num Matricola. Se **non** lo farete il compito sarà annullato. Esempio:

```
/*********************************
* Nome: Paolino
* Cognome: Paperino
* Num Matricola: 12345678
* email: paperino@unibo.it
*********************************/
```

Se il testo richiede effettuare lettura di file testuali, il candidato deve crearsene alcuni di esempio da solo (usando jEdit come editor e salvando il file con estensione .txt), **rispettando le specifiche date**.
E' **consentito** invocare qualunque metodo delle API di Java.

La prova intende valutare, oltre alla verifica che lo studente ha acquisito **competenze** nell'ambito della programmazione, la capacità del candidato di **comprendere** il testo del compito (i.e., programmi che girano ma **non** fanno quanto richiesto verranno considerati insufficienti), la capacità di inserire **commenti** al codice per facilitare la comprensione dell'elaborato, la capacità di scrivere **codice pulito e ordinato**, la capacità di **ottimizzare** la soluzione (soprattutto in termini di efficienza del programma).

NOTE TECNICHE

Nel caso (frequente) vi si dovesse bloccare il sistema di sviluppo (ex: per loop infinito), potete aprire un terminale (CTRL-ALT-t) e provare scrivere il comando "pkill java" o "pkill jedit". Se si dovesse chiudere jEdit (che e` quello che dovrebbe capitare), poi scrivere "jedit &" e il sistema dovrebbe ripartire.

NOTA LOGISTICA

Agli studenti affetti da disturbi quali dislessia, è richiesto di completare tutti gli esercizi in un tempo maggiorato del 30%.

Nota per lo studente: Per sicurezza, nei commenti iniziali si scriva "**studente con DSA**".

TESTO DEL COMPITO

Esercizio 1

Si crei un programma che legga in input (da tastiera) un valore intero ***x*** e il nome di un file. Si assume che il file da leggere sia così strutturato: il file è composto da una serie di righe, ognuna contenente due numeri interi. As esempio:

0 4

3 7

8 49

-1 13

Ogni coppia rappresenta un intervallo di valori interi (quindi, il primo numero sarà minore del secondo).

Il programma deve restituire l'intervallo di ampiezza maggiore, contenente il valore ***x***. Ad esempio, se ***x*** = 2 e il file contiene i valori sopra riportati, il programma deve restituire in output i valori -1, 13.

Testare il metodo in una classe demo.

Esercizio 2

Scrivere un metodo Java che, preso in input un array (di dimensione **n**) di valori numerici, crei una matrice (array bidimensionale), di dimensione **[n, n]**, con le seguenti proprietà:

- prima riga: è identica all'array letto in input;
- righe successive: ogni elemento è il risultato della somma di:
    - doppio del valore che sta nella stessa colonna e riga precedente (l'elemento "sopra")
    - doppio del valore sulla riga e colonna precedenti (elemento in alto a sinistra, se esiste)
    - doppio del valore sulla riga precedente e colonna successiva (elemento in alto a destra, se esiste):

Esempio: array in input {1, 2, 3}

Matrice risultante:
{ { 1, 2, 3 },
{ 6, 12, 10 },
{ 36, 56, 44 } }

Esercizio 3

Scrivere un metodo ricorsivo che prende in input un intero N e stampa in ordine decrescente i quadrati di tutti gli interi positivi dispari minori o uguali di N.

Esercizio 4

Si realizzi un'applicazione Java per la gestione di un garage pubblico con le seguenti caratteristiche:

- numero illimitato di posti disponibili
- politica di inserimento ed estrazione di tipo FIFO – First In First Out (garage surreale quindi, in cui per uscire bisogna aspettare che escano tutti i veicoli entrati prima)

Ogni posto del garage è identificato da un numero e può ospitare soltanto veicoli a motore, che sono di tre tipi: automobili, furgoni e motociclette.

Un veicolo a motore è caratterizzato da:

- anno di immatricolazione (intero)
- marca (stringa)
- tipo alimentazione (stringa)
- cilindrata (intero)

Inoltre ogni tipo di veicolo a motore ha alcune informazioni aggiuntive:

`Furgone`: capacità di carico

`Automobile`: numero di porte

`Motocicletta`: tipologia (stringa) e numero di tempi del motore (int)

Si crei un'interfaccia che descrive le operazioni possibili su un garage:

- Immissione di un nuovo veicolo nel garage in coda a tutti gli altri - restituire il numero del posto assegnato
- Estrazione dal garage del veicolo parcheggiato da più tempo - restituire l'istanza del veicolo stesso
- Stampa della situazione corrente dei posti nel garage: stampare per ogni veicolo tutte le informazioni disponibili
- Conteggio del numero di veicoli nel garage con una cilindrata maggiore di un valore X preso in input;
- Export su file delle informazioni relative agli ultimi N veicoli entrati nel garage, dove N è un valore passato in input (se nel garage ci sono meno di N veicoli si esportano le informazioni di tutti); il formato di salvataggio dei dati è arbitrario; gestire errori nell'apertura del file

Si implementi poi una classe `Garage` che modelli il garage sopra descritto e implementi questa interfaccia.

Si realizzi una classe demo con dati arbitrari per testare il corretto funzionamento del garage.