

– Compito V Appello –

19/01/2022

Potranno essere consegnati al docente solo gli esercizi che avranno superato i nostri JUnit Test. I test svolti dagli studenti non saranno presi in considerazione per la valutazione. La consegna di entrambi gli esercizi (o solo del secondo esercizio, per coloro che vogliono far valere la prova intermedia superata) è condizione necessaria, ma non sufficiente, per passare la prova.

Per la valutazione saranno presi in considerazione aspetti di “buona programmazione”, “pulizia del codice” ed efficienza. Ad es.: formattazione corretta del codice, rendere il codice modulare aggiungendo ove necessario altri metodi rispetto a quelli richiesti dall’esercizio, soprattutto se questi rendono il codice più pulito e leggibile, o se evitano duplicazione di codice, evitare assegnamenti o inizializzazioni inutili, uso corretto dei modificatori di accessibilità, ecc. Inoltre, non ci devono essere warning nel codice scritto.

IMPORTANTE: seguire attentamente le specifiche per quanto riguarda i nomi (pacchetti, classi, metodi) e la firma dei metodi, altrimenti i test automatici falliranno rendendo il compito insufficiente. Invece, concetti come l’accessibilità di variabili di istanza o metodi, wildcards e tipi generici, non vengono testati automaticamente: dovete assicurarvi di seguire comunque attentamente le specifiche.

Allo stesso modo le stringhe create dai vostri metodi devono rispettare le specifiche, inclusi eventuali spazi, altrimenti, nuovamente i test automatici falliranno.

A meno che non sia specificato diversamente, i metodi e i costruttori richiesti dall’esercizio si intendono accessibili da qualsiasi classe.

Gli studenti possono aggiungere altri metodi alle classi, e sono invitati a farlo se questi rendono il codice più pulito e leggibile e soprattutto se questo aiuta a evitare duplicazione di codice. In tal caso, i metodi aggiunti devono essere accessibili solo alla classe stessa.

I puntini di sospensione tra parentesi angolari < ... > vanno sostituiti con il tipo più generico possibile rispetto alla specifica dell’esercizio.

Questa prova scritta è costituita da due esercizi.

- Esercizio n.1, in cui si dovrà svolgere un esercizio Java incentrato sugli argomenti visti nel primo semestre del corso. **Gli studenti che hanno superato e vogliono far valere la prova intermedia sono esentati da svolgere questo esercizio del compito**, mentre **deve essere svolto da tutti gli altri studenti**. Si ricorda che lo studente, per provare a migliorare la valutazione della sua prova intermedia, può comunque decidere di svolgere anche questa prima parte del compito. In questo caso, al momento della sua consegna, la prova intermedia viene annullata indipendentemente dalla valutazione del compito stesso.
- Esercizio n. 2, in cui si dovrà svolgere un esercizio Java incentrato sugli argomenti visti nel secondo semestre del corso. **Questa parte deve essere svolta da tutti gli studenti**.

ATTENZIONE: entrambi gli esercizi vanno svolti in ambiente Java utilizzando Eclipse come visto al secondo semestre, non utilizzando il pacchetto “java--”.

CONSEGNA ESERCIZI:

Gli studenti che hanno superato e vogliono far valere la prova intermedia, devono consegnare solo i file sorgenti (.java) relativi all'Esercizio n.2.

Gli studenti che non hanno superato o non vogliono far valere la prova intermedia, devono consegnare i file sorgenti (.java) relativi all'Esercizio n.1 e all'Esercizio n.2.

Entro il termine ultimo previsto per la consegna dello scritto, gli studenti che intendono consegnare provvedono a caricare il sorgente (o i sorgenti) .java attraverso l'apposita attività “compito”, disponibile nella pagina MOODLE del corso al seguente link:

<https://e-l.unifi.it/mod/assign/view.php?id=799505>


Fino allo scadere del tempo, lo studente potrà apportare modifiche al proprio lavoro. Non saranno accettate consegne effettuate in ritardo, o con modalità diverse da quelle definite dal docente. All'orario stabilito ad inizio compito il docente dichiara finita la prova e chiude la sessione.

Esercizio Java n. 1: Ribalta Matrice

Sia A una matrice di interi di dimensione $m \times n$, con $m > 0$ e $n > 0$, e sia r un array di caratteri di dimensione k , con $k > 0$, i cui elementi possono assumere solo il valore 'V' (Verticale) oppure 'O' (Orizzontale).


Un ribaltamento della matrice A rispetto all'asse Verticale genera una nuova matrice A' in cui la parte "sinistra" della matrice corrisponde ad A , mentre la parte "destra" è data dalla matrice A "ribaltata" in cui la prima colonna di A viene copiata nell'ultima colonna di A' , la seconda colonna di A viene copiata nella penultima colonna di A' , e così via.

Ad esempio, sia $A = \begin{pmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \end{pmatrix}$, il ribaltamento di A rispetto all'asse verticale ('V') genera la matrice $A' = \begin{pmatrix} 0 & 1 & 2 & 2 & 1 & 0 \\ 3 & 4 & 5 & 5 & 4 & 3 \end{pmatrix}$. (Esempio 1)



Un ribaltamento della matrice A rispetto all'asse Orizzontale genera una nuova matrice A' in cui la parte "alta" della matrice corrisponde ad A , mentre la parte "bassa" è data dalla matrice A "ribaltata" in cui la prima riga di A viene copiata nell'ultima riga di A' , la seconda riga di A viene copiata nella penultima riga di A' , e così via.

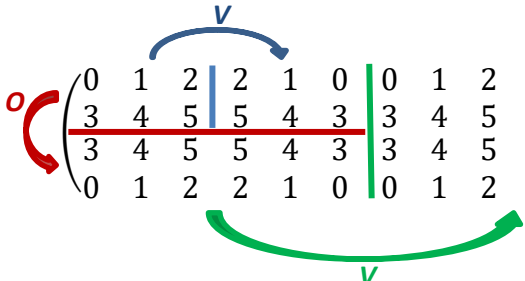
Ad esempio, sia $A = \begin{pmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \end{pmatrix}$, il ribaltamento di A rispetto all'asse orizzontale ('O') genera la matrice $A' = \begin{pmatrix} 3 & 4 & 5 \\ 0 & 1 & 2 \end{pmatrix}$. (Esempio 2)



Infine, un ribaltamento della matrice A rispetto all'array r genera una nuova matrice A' in cui si applicano in sequenza i ribaltamenti della matrice di partenza rispetto alle assi Verticale o Orizzontale così come definito dagli elementi dell'array r .

Ad esempio, sia $A = \begin{pmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \end{pmatrix}$, e sia $r = ('V', 'O', 'V')$, il ribaltamento di A rispetto ad r

genera la matrice $A' = \begin{pmatrix} 0 & 1 & 2 & 2 & 1 & 0 & 0 & 1 & 2 & 2 & 1 & 0 \\ 3 & 4 & 5 & 5 & 4 & 3 & 3 & 4 & 5 & 5 & 4 & 3 \\ 3 & 4 & 5 & 5 & 4 & 3 & 3 & 4 & 5 & 5 & 4 & 3 \\ 0 & 1 & 2 & 2 & 1 & 0 & 0 & 1 & 2 & 2 & 1 & 0 \end{pmatrix}$. (Esempio 3)



Consegna esercizio con difficoltà [BASE]:

Assumendo che l'array r sia sempre composto da 1 solo elemento (quindi si prevede 1 solo ribaltamento rispetto all'asse Verticale o Orizzontale – si veda Esempio 1 e 2), scrivere una classe `RibaltaMatrice` nel pacchetto `compito` contenente un metodo statico pubblico `ribalta` che, dati una matrice A ed un array r definiti come precedentemente descritto, restituisca una nuova matrice corrispondente al ribaltamento di A rispetto a r .

I JUnit Test che devono essere superati per la Consegna 1 sono quelli della classe **`RibaltaMatriceTest_Base`** (non considerare gli altri test che ovviamente falliranno).

Consegna esercizio con difficoltà [AVANZATA]: [← FACOLTATIVA](#)

Assumendo che l'array r sia composto da k elementi, con $k > 0$ (si veda Esempio 3), scrivere una classe `RibaltaMatrice` nel pacchetto `compito` contenente un metodo statico pubblico `ribalta` che, dati una matrice A ed un array r definiti come precedentemente descritto, restituisca una nuova matrice corrispondente al ribaltamento di A rispetto a r . E' possibile adottare indifferentemente sia una soluzione iterativa che una soluzione ricorsiva.

I JUnit Test che devono essere superati per la Consegna 2 sono quelli della classe **`RibaltaMatriceTest_Extra`**.

NOTA BENE:

- Gli studenti dovranno consegnare per questo esercizio solo 1 sorgente relativo alla consegna “base” oppure relativo alla consegna con difficoltà “avanzata”.

Nello svolgere l'esercizio NON devono essere utilizzati i metodi `clone`, o `arraycopy`, o metodi della classe `Arrays`. L'utilizzo di tali metodi renderà l'esercizio automaticamente insufficiente.

Esercizio Java n. 2:

Nel pacchetto `auxi` trovate le classi `Biglia`, `BigliaNumerata`.

Nel pacchetto `compito` trovate gli stub (da implementare) delle classi `Buca` e `Biliardo`.

La classe `Buca` ha

- **[necessario]** la seguente variabile di istanza non modificabile,

```
Set<Biglia> imbucate
```


con metodo `getters`,
- **[necessario]** costruttore `public Buca(Set<Biglia> imbucate)`
che inizializza le rispettive variabili di istanza con gli argomenti passati,

La classe `Biliardo` ha variabili di istanza non modificabili `List<Buca> buche;`

`Set<Biglia> tavolo;` ed i seguenti metodi e costruttore:

- **[necessario]** `public Biliardo()`
che inizializza `buche` inserendo 6 oggetti `Buca`, tutti diversi e tutti con la loro variabile di istanza `imbucate` vuota (cioè non ci sono biglie imbucate in nessuna buca).

Inizializza `tavolo` inserendo le biglie numerate da 1 a 15 (inclusi), dove per numero sulla biglia si intende la variabile di istanza `value` della classe `BigliaNumerata`. Inoltre viene inserito in `tavolo` una unica biglia non numerata, cioè un oggetto di classe `Biglia`.
- **[necessario]** `public List<Buca> getBuche()`
che ritorna `buche`,
- **[necessario]** `public Set<Biglia> getTavolo()`
che ritorna `tavolo`,
- **[opzionale]** `public Biliardo(List<Buca> buche, Set<Biglia> tavolo)`
che solleva l'eccezione `IllegalArgumentException` se (i) l'insieme di tutte le biglie (numerate e non numerate) contenute nelle `buche` e nel `tavolo` contiene almeno una biglia doppiata, oppure (ii) se esiste una biglia numerata con un numero fuori dall'intervallo che va da 1 a 15; altrimenti, inizializza le variabili di istanza con gli argomenti passati.
- **[opzionale]** `public boolean gameOver()`
che ritorna `true` se la biglia numerata con 8 è contenuta in una delle `buche`, `false` altrimenti;

- **[opzionale]** `public boolean pallinoImbucato()`

che ritorna `true` se la biglia non numerata (cioè di classe `Biglia`) è imbucata, `false` altrimenti;

- **[opzionale]** `public void reset()`

che toglie tutte le biglie contenute nelle `buche` e le aggiunge al `tavolo`; attenzione, non va creato nessun nuovo oggetto;

NOTA: i metodi necessari contengono test che devono essere superati per ottenere un voto almeno di fascia bassa nel secondo esercizio. I test dei metodi opzionali (denominati come opzionali) possono non essere superati. Solo nel caso in cui tutti i test necessari passano, lo svolgimento corretto di questi ulteriori metodi opzionali può aumentare la fascia di voto del secondo esercizio.