PARADIGMI DI PROGRAMMAZIONE

A.A. 2024/2025

Laurea triennale in Informatica

E1: Tema dell'esercizio

TEMA DELL'ESERCIZIO

Lo scopo dell'esercizio è offrire un'occasione per dimostrare, nella scrittura di codice Java, la comprensione del ruolo delle strutture introdotte a lezione.

PATTINAGGIO DI FIGURA

Hanyu Yusuru

PyeongChang 2018

Il punteggio della prova viene raccolto in tempo reale, e mostrato nei canali interni all'arena, nella diretta televisiva e sul sito isuresults.org.

Semplificando la Rule 353 del regolamento ufficiale ISU possiamo dire che:

Per ogni elemento tecnico:

- il Technical Panel pubblica il *Base Value*
- ogni giudice pubblica un *Grade Of Execution*
- ogni giudice può chiedere la *Review*

A fine della prova:

- i giudici pubblicano i *GOE* per gli elementi rivisti
- i giudici pubblicano i giudizi complessivi dei *Components*

Un *Base Value* è un valore maggiore di 0.

Un *Grade Of Execution* va da -5 a 5 a passi di 1.

Un *Component* va da 0.25 a 10.0 a passi di 0.25

I component sono:

- Skating Skills
- Transitions
- Performance
- Composition
- Interpretation of Music

L'esercizio consiste nel completare un sistema per la raccolta e visualizzazione dei punteggi.

CLASSI IMPLEMENTATE

Alcune classi sono già implementate e non dovrebbero richiedere modifiche.

`it.unipd.pdp2024.ex1.Element`

```java
public record Element(int idx, String element)
```

Elemento che l'atleta esegue.

`it.unipd.pdp2024.ex1.Nation`

```
public enum Nation
```

Distingue i giudici. Ci sono 9 nazioni, quindi ci sono 9 giudici nel sistema.

`it.unipd.pdp2024.ex1.Vote`

```
public interface Vote
```

Accomuna i vari tipi di voti che giudici e pannello tecnico possono emettere.

`it.unipd.pdp2024.ex1.Athlete`

```
Athlete(final BlockingQueue<Element> rink,
  final List<Element> program)
```

Emette sul rink gli elementi del programma, e si ferma.

it.unipd.pdp2024.ex1.Judge

```
public Judge(Nation nation, int idx,
  BlockingQueue<Element> screen, BlockingQueue<Vote> votes)
```

Osserva gli elementi sullo schermo, emette i relativi voti. Quando vede l'ultimo elemento, emette i components e si ferma.

`it.unipd.pdp2024.ex1.TechnicalPanel`

```
public TechnicalPanel(BlockingQueue<Element> screen,
  BlockingQueue<Vote> values)
```

Osserva gli elementi sullo schermo, emette i relativi Base Values. Quando vede l'ultimo elemento si ferma.

`it.unipd.pdp2024.ex1.ScoreBoard`

```
public ScoreBoard(BlockingQueue<Vote> votes)
```

Osserva i voti e li somma. Espone un `AtomicReference` contenente il risultato parziale da stampare. Va fermato a conteggio concluso.

CLASSI DA IMPLEMENTARE

it.unipd.pdp2024.ex1.VideoSystem

```
public VideoSystem(BlockingQueue<Element> rink)
```

Fornisce delle code, che memorizza internamente. Una volta avviato, riporta su ogni coda fornita gli elementi che osserva sul rink, in modo che gli altri componenti del sistema li possano osservare. Si ferma quando osserva l'ultimo elemento.

`it.unipd.pdp2024.ex1.Printer`

```
Printer(ScoreBoard score)
```

Quando viene avviato, ogni secondo accede al risultato e lo stampa formattato. Va fermato a conteggio concluso.

`it.unipd.pdp2024.ex1.Main`

Inizializza e collega i vari oggetti. Avvia le parti del sistema. Rileva quando l'esercizio è concluso (cioè le parti che si fermano da sole si sono fermate) e ferma le parti che vanno fermate.

I punti in cui inserire il vostro codice sono evidenziati dai commenti TODO.

Aggiungete quanto ritenete necessario, e verificate il funzionamento del sistema.

CONSEGNA

```
PS> hg update default
PS> hg pull
PS> hg summary
genitore: 14:f4c041b25659 tip
 Added tag Esercizio for changeset fcf8a0db1a6d
branch: default
commit: (pulito)
update: (aggiornato)
```

```
PS > hg update {matricola}M
PS > hg merge default
PS > hg commit -m "Merge from default"
PS > .\gradlew ex1
```

La JVM deve terminare regolarmente. Significa cioè che ogni Thread deve fermarsi o essere fermato.

In caso di errori di formattazione, il sistema non parte, evidenziando i file non formattati. Correggete con il comando:

```
PS > .\gradlew spotlessApply
```

Quando siete soddisfatti del codice che avete scritto, riformattatelo con `spotlessApply` e consegnatelo rigenerando e aggiornando il bundle condiviso.

VALUTAZIONE

L'esercizio vale 3 punti. Il suo risultato sarà considerato valido fino all'ultimo appello della sessione di Febbraio 2026.

- 1 punto per la corretta esecuzione e formattazione del codice
- 1 punto per lo stile e le idee implementate
- 1 punto per il corretto uso delle classi usate a lezione

Non prendete soluzioni da altri senza averle esaminate criticamente: è statisticamente provato che sono probabilmente sbagliate.

Consegnate anche se non riuscite a far funzionare il sistema. Se l'errore è triviale ma l'idea corretta, posso comunque darvi qualche punto.

I bundle validi saranno quelli con data di ultima modifica non maggiore delle 23:59 del 26/04/2025.

Tale condizione è tassativa.

In caso di dubbi, chiedetemi pure via mail.

In caso di errata, segnalerò sugli annunci del corso il problema e come proseguire.