Esercizi di riepilogo Java

10 Maggio 2023

Esercizio 1.

La classe **SequenzaDiStringheBinarie** rappresenta una sequenza di stringhe ciascuna delle quali è costituita soltanto a 0 e 1. Il suo scheletro è il seguente:

```
public class SequenzaDiStringheBinarie {

	private String[] seq;

	/* crea un oggetto SequenzaDiStringheBinarie il cui contenuto è
	rappresentato dall'array seq. Si assuma che tutte le stringhe di seq,
	siano effettivamente stringhe binarie. */
	public SequenzaDiStringheBinarie (String[] seq){...}

	/* Dato un indice i restituisce il valore numerico rappresentato dalla
	stringa binaria in posizione i se interpretata come un numero binario. Si
	assuma che i sia un indice valido. */
	public int valoreStringa(int i){...}

	/* Dato un intero k restituisce gli indici delle stringhe di seq il cui
	valore (se interpretate come numeri binari) è superiore a k */
	public int[] elementiSopraSoglia(int k){...}

	/* Restituisce una rappresentazione testuale della sequenza di stringhe.
	Per ogni elemento della sequenza va mostrata la stringa binaria e, tra
	parentesi, il valore numerico della stringa (se interpretata come numero
	binario) */
	public String toString(){...}

}
```

A titolo di esempio, si supponga che un oggetto **SequenzaDiStringheBinarie s** contenga le seguenti stringhe:

1001001
110
01100

L'invocazione **c.toString()** restituisce la seguente stringa:

```
1001001 (73)
110 (6)
01100 (12)
```

L'invocazione **s.valoreStringa(1)** restituisce **6** (valore del numero binario **110**). L'invocazione **c.elementiSopraSoglia(10)** restituisce l'array **{0,2}**, che contiene gli indici delle stringhe binarie 1001001 e 01100, che sono tutte quelle della sequenza il cui valore è maggiore di **10**.

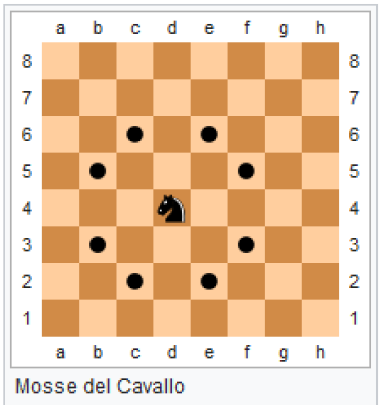
Si scriva la classe **SequenzaDiStringheBinarie** ed una classe **Prova SequenzaDiStringheBinarie** che contiene il solo metodo **main** e che esegue le seguenti azioni:

- Fa inserire all'utente una sequenza di stringhe binarie e crea un oggetto **SequenzaDiStringheBinarie s**. Durante l'inserimento bisogna assicurarsi che le stringhe inserite siano effettivamente stringhe binarie, cioè costituite da 0 e 1.
- Visualizza all'utente l'oggetto creato.
- Fa inserire ripetutamente all'utente un indice **i** e mostra all'utente il valore della stringa in posizione **i**. Per ogni indice inserito occorre verificare che esso sia un indice valido.
- Fa inserire ripetutamente all'utente un valore di soglia **k** e mostra all'utente gli indici delle stringhe i cui valori sono maggiori di **k.**

Esercizio 2.

Nel gioco degli scacchi il **cavallo** può muoversi su una delle caselle a lui più vicine che non appartengono alla riga, alla colonna e alle diagonali passanti per la sua casella di partenza. Un cavallo al centro della scacchiera ha a disposizione otto caselle ("rosa di cavallo") verso le quali muoversi (si veda la figura), mentre se si trova al bordo la sua mobilità è ridotta a quattro caselle e se si trova in un angolo a due. Il movimento del cavallo può essere immaginato come la somma di uno spostamento orizzontale di una casella e di uno verticale di due (o viceversa), disegnando una "L". Tale traiettoria è però "virtuale", nel senso che il cavallo, a differenza di tutti gli altri pezzi, si può portare direttamente sulla nuova casella senza che il percorso effettuato sia necessariamente sgombro, si dice che può "saltare".

R	D	A	C	T	P
re	donna	alfiere	cavallo	torre	pedone



Mosse del Cavallo

La classe **Pezzo** rappresenta i vari pezzi degli scacchi. La classe è la seguente:

```
public class Pezzo{
      private char tipo; // indica il tipo di pezzo secondo quanto indicato in
      tabella
      private char colore; // B=bianco, N=nero
      public Pezzo(char t, char c){
          tipo=t;
          colore=c;
      }
      public char getTipo(){ return tipo; }
```

```
        public char getColore(){ return colore; }
}
```

La classe **Scacchiera** rappresenta una scacchiera. La scacchiera è rappresentata da una matrice di oggetti **Pezzo** di dimensione 8×8 in cui le celle vuote memorizzano il valore **null**. La classe ha il seguente scheletro.

```
public class Scacchiera {
        private Pezzo[][] scacchiera;

        /* crea una scacchiera vuota, cioè senza pezzi. */
        public Scacchiera(){...}

        /* Posiziona il pezzo p in posizione (i,j). Se i e j non sono indici
        validi la scacchiera non viene modificata. Se la posizione (i,j) è già
        occupata da un altro pezzo la scacchiera non viene modificata. Se p è
        nullo la scacchiera non viene modificata. Il metodo restituisce true se il
        posizionamento è avvenuto con successo, false altrimenti */
        public boolean posizionaPezzo(int i, int j, Pezzo p){...}

        /* Restituisce true se il pezzo in posizione (i,j) è minacciato da un
        cavallo di colore c, false altrimenti. Un pezzo è minacciato da un cavallo
        se è di colore opposto a quello del cavallo e si trova in una delle celle
        su cui il cavallo può portarsi con una mossa. Se in posizione (i,j) non
        c'è un pezzo viene restituito false */
        public boolean minacciatoDaCavallo(int i, int j, char c){...}

        /* Restituisce un array di pezzi che sono minacciati da almeno un cavallo
        di colore c. Se nessun pezzo è minacciato da un cavallo di colore c viene
        restituito un array di dimensione 0. */
        public Pezzo[] minacciatiDaCavalli(char c){...}

        /* Restituisce una rappresentazione testuale della scacchiera. Per ogni
        cella occupata va indicato il tipo e il colore del pezzo che la occupa. */
        public String toString(){...}
}
```

Si scriva la classe Scacchiera ed una classe **ProvaScacchiera** che contiene il solo metodo **main** e che esegue le seguenti azioni:

- Crea una scacchiera e fa posizionare all'utente dei pezzi su di essa (numero, tipo e colore dei pezzi sono scelti dall'utente).
- Mostra all'utente la scacchiera popolata.
- Fa inserire ripetutamente all'utente due indici i e j ed un carattere c (pari a B o N) e dice all'utente se il pezzo posizionato in (i,j) è minacciato da un cavallo di colore c (se in (i,j) non c'è un pezzo la risposta sarà negativa). L'utente decide quando smettere con l'inserimento di indici.
- Fa inserire un colore c all'utente ed elenca tutti i pezzi che sono minacciati da almeno un cavallo di colore c.