Cognome _____ Nome _____
Matricola _____ Postazione PC _____

Corso di Laurea in Ingegneria Gestionale
Esame di Informatica
a.a. 2011-12
16 gennaio 2012

Testo

Il database di un videonoleggio è costituito da due vettori paralleli. Il primo è denominato "videoteca" e contiene oggetti di tipo "Film" che rappresentano i film presenti nell'archivio. Il secondo vettore è denominato "noleggi" e contiene oggetti di tipo "Noleggio" che rappresentano le informazioni di noleggio di un film. Le informazioni di noleggio del film presente in una determinata posizione nel vettore "videoteca", si trovano nella corrispondente posizione nel vettore "noleggi". Nel caso un dato film non sia stato noleggiato, nella sua posizione nel vettore "noleggi" sarà presente un riferimento "null". Entrambi i vettori hanno dimensione pari alla costante "MAX_FILM" (inizializzata a 256). Se il numero di film contenuti nell'archivio è inferiore a "MAX_FILM", i primi elementi del vettore conterranno gli oggetti di tipo "Film", mentre gli altri conterranno riferimenti "null". Tutti gli elementi "null" del vettore "videoteca" si devono trovare alla fine del vettore e non possono trovarsi in mezzo agli elementi validi.

Le classe Film contiene le informazioni relative ad un film ed un metodo per la stampa a video di queste informazioni:

```
public class Film {

    private int codice;
    public String titolo;
    public String regia;
    public int anno;
    public String[] cast;

    public Film(int co, String t, String r, int a, String[] ca) {
        codice = co;
        titolo = t;
        regia = r;
        anno = a;
        cast = ca;
    }

    public int getCodice() {
        return codice;
    }

    public void stampaInfo() {

        System.out.println("Film n° " + codice + " / " + titolo + " / " + anno
                + " / " + regia);

        if (cast != null && cast.length > 0) {
            System.out.print("\tCast:");
            for (int i = 0; i < cast.length - 1; i++) {
                System.out.print(" " + cast[i] + ",");
            }
            System.out.println(" " + cast[cast.length - 1]);
        }
    }

}
```

La classe Noleggio contiene il nome dell'utente che ha noleggiato un determinato film e la data di noleggio:

```
public class Noleggio {

    public String utente;
    public int giorno;
    public int mese;
    public int anno;

    public Noleggio(String u, int g, int m, int a) {
        utente = u;
        giorno = g;
        mese = m;
        anno = a;
    }
}
```

Si consiglia di procedere nel seguente modo: implementare un metodo e successivamente scrivere la parte del main che utilizza tale metodo, in modo da poterne verificare immediatamente la correttezza.

Le varie operazioni devono essere eseguite sulla porzione significativa dell'archivio, cioè la porzione di "videoteca" che non contiene riferimenti "null".

a) Scrivere il metodo statico:

```
public static void invertiOrdine(Film[] filmdb, Noleggio[] noldb)
```

che prende in ingresso i vettori "filmdb" e "noldb" costituenti il database di un videonoleggio ed inverte l'ordine dei film presenti in archivio (es. il primo film diventerà l'ultimo, il secondo diventerà il penultimo, etc.).

b) Scrivere il metodo statico:

```
public static Film[] noleggiatiDopo(Film[] filmdb, Noleggio[] noldb, int g,
        int m, int a)
```

che prende in ingresso i vettori "filmdb" e "noldb" costituenti il database di un videonoleggio ed una data rappresentata dai tre interi g (giorno), m (mese) ed a (anno). Il metodo deve ritornare un vettore di Film contenente tutti i film noleggiati dopo la data indicata.

c) Scrivere il metodo statico:

```
public static int restituisciUtente(Film[] filmdb, Noleggio[] noldb,
        String ut)
```

che prende in ingresso i vettori "filmdb" e "noldb" costituenti il database di un videonoleggio e la stringa "ut" rappresentante il nome di un utente. Il metodo deve rendere nuovamente disponibili tutti i film noleggiati dall'utente "ut" e deve ritornare il numero di film riconsegnati.

d) Scrivere il metodo statico:

```
public static String[] filmInfo(Film[] filmdb, int codice)
```

che prende in ingresso un vettore di Film "filmdb" ed un intero "codice". Il metodo deve cercare nel vettore "filmdb" il film avente il codice specificato e deve ritornare un vettore di stringhe così costruito: la prima stringa rappresenta il titolo del film, la seconda rappresenta il regista e le successive rappresentano gli attori del cast.

e) Scrivere il metodo main che:

- definisca ed inizializzi i vettori "videoteca" e "noleggi" secondo i valori riportati in tabella e stampi a video l'archivio. La stampa dell'archivio consiste nella stampa delle informazioni di ogni film (usando il metodo "stampaInfo" della classe "Film") e, se il film è noleggiato, deve essere indicato anche l'utente e la data di noleggio.

Codice	Titolo	Regia	Anno	Cast	Utente	Data
0	Le iene	Quentin Tarantino	1992	Harvey Keitel Tim Roth	Mario Rossi	5/1/2012
1	Pulp Fiction	Quentin Tarantino	1994	John Travolta Samuel L. Jackson		
2	Dal tramonto all'alba	Robert Rodríguez	1996	George Clooney	Mario Rossi	30/12/2011
3	Grindhouse – A prova di morte	Quentin Tarantino	2007	Kurt Russel	Gianni Verdi	13/1/2012

- inverta l'ordine dei vettori "videoteca" e "noleggi" utilizzando il metodo "invertiOrdine" e, poi, stampi nuovamente l'archivio.
- stampi l'elenco dei titoli dei film noleggiati dopo il 1/1/2012, usando il metodo "noleggiatiDopo".
- renda nuovamente disponibili per il noleggio tutti i film noleggiati da Mario Rossi, usando il metodo "restituisciUtente", stampi il risultato del metodo e stampi l'archivio aggiornato.
- stampi le informazioni recuperate dalla funzione "filmInfo" relative al film avente come codice "1".

Soluzione

```
public class Esame {

    public static final int MAX_FILM = 256;

    /*
     * Questo metodo conta i film presenti nell'archivio. Non è richiesto dal
     * testo, ma semplifica la scrittura degli altri metodi.
     */
    public static int contaFilm(Film[] filmdb) {
        int nFilm = 0;
        while (nFilm < MAX_FILM && filmdb[nFilm] != null) {
            nFilm++;
        }
        return nFilm;
    }

    /*
     * Questo metodo stampa le informazioni relative a tutti i film presenti
     * nell'archivio e, per i film noleggiati, indica l'utente che li ha
     * noleggiati e la data di noleggio. Non è richiesto dal testo, ma
     * semplifica la scrittura del main.
     */
    public static void stampaArchivio(Film[] filmdb, Noleggio[] noldb) {
        /* Conteggio dei film presenti nell'archivio */
        int nFilm = contaFilm(filmdb);

        for (int i = 0; i < nFilm; i++) {
            filmdb[i].stampaInfo();
            if (noldb[i] != null) {
                System.out.println("\tNoleggiato a " + noldb[i].utente + " il "
                        + noldb[i].giorno + "/" + noldb[i].mese + "/"
                        + noldb[i].anno);
            }
        }
    }

    /*
     * Questo metodo confronta due date, ciascuna espressa da tre interi che
     * indicano giorno, mese ed anno. Ritorna un valore: positivo se la prima
     * data è maggiore della seconda; negativo se la prima data è minore della
     * seconda; zero se le date sono uguali. Non è richiesto dal testo, ma
     * semplifica la scrittura dei metodi.
     */
    public static int confrontaData(int g1, int m1, int a1, int g2, int m2,
            int a2) {
        /*
         * Prima si confrontano gli anni. La differenza tra gli anni (qualora
         * sia diversa da 0) delle due date costituisce anche il valore di
         * ritorno, perché se è positiva allora la prima data è maggiore,
         * mentre se negativa la prima data è minore.
         */
        int ret = a1 - a2;
        /*
         * Nel caso in cui l'anno sia lo stesso, allora il confronto avviene
         * sul mese in maniera simile a quanto già visto.
         */
        if (ret == 0) {
            ret = m1 - m2;
            /*
             * Nel caso in cui anche il mese sia lo stesso, allora il confronto
             * avviene sul giorno.
             */
            if (ret == 0) {
                ret = g1 - g2;
            }
        }
        return ret;
    }
```

```
public static void invertiOrdine(Film[] filmdb, Noleggio[] noldb) {
    /* Conteggio dei film presenti nell'archivio */
    int nFilm = contaFilm(filmdb);

    /*
     * Per invertire l'ordinamento dell'archivio, bisogna scambiare il
     * primo film con l'ultimo, il secondo con il penultimo, e così via,
     * fino ad arrivare al centro dell'archivio.
     */
    for (int i = 0; i < nFilm / 2; i++) {
        /*
         * Calcolo dell'indice j dell'elemento da scambiare con i. Dato che
         * j parte dall'ultimo elemento (nFilm - 1) e viene decrementato
         * ogni volta, può essere facilmente calcolato a partire da i.
         */
        int j = nFilm - 1 - i;
        /* Scambio i due film nelle posizioni i e j */
        Film tmp = filmdb[i];
        filmdb[i] = filmdb[j];
        filmdb[j] = tmp;
        /*
         * Per mantenere la consistenza dei dati, vanno scambiati anche i
         * corrispondenti elementi del vettore parallelo.
         */
        Noleggio tmp2 = noldb[i];
        noldb[i] = noldb[j];
        noldb[j] = tmp2;
    }
}

public static Film[] noleggiatiDopo(Film[] filmdb, Noleggio[] noldb, int g,
        int m, int a) {
    /* Conteggio dei film presenti nell'archivio */
    int nFilm = contaFilm(filmdb);
    int nRis = 0;
    /* Conteggio degli elementi del vettore risultato */
    for (int i = 0; i < nFilm; i++) {
        if (noldb[i] != null
                && confrontaData(noldb[i].giorno, noldb[i].mese,
                        noldb[i].anno, g, m, a) > 0) {
            nRis++;
        }
    }
    /* Creazione del vettore risultato */
    Film[] ris = new Film[nRis];
    /* Popolamento del vettore risultato */
    int k = 0;
    for (int i = 0; i < nFilm; i++) {
        if (noldb[i] != null
                && confrontaData(noldb[i].giorno, noldb[i].mese,
                        noldb[i].anno, g, m, a) > 0) {
            ris[k] = filmdb[i];
            k++;
        }
    }
    return ris;
}

public static int restituisciUtente(Film[] filmdb, Noleggio[] noldb,
        String ut) {
    /* Conteggio dei film presenti nell'archivio */
    int nFilm = contaFilm(filmdb);
    int ris = 0;
    for (int i = 0; i < nFilm; i++) {
        /*
         * Prima di controllare il campo utente dell'oggetto riferito da
         * noldb[i] bisogna assicurarsi che il riferimento non sia nullo, o
         * si avrà un errore a tempo di esecuzione
         */
        if (noldb[i] != null && noldb[i].utente.equals(ut)) {
```

```
                ris++;
                /*
                 * Per rendere nuovamente disponibile il film bisogna
                 * cancellare le informazioni di noleggio.
                 */
                noldb[i] = null;
            }
        }
        return ris;
    }

    public static String[] filmInfo(Film[] filmdb, int codice) {
        /* Conteggio dei film presenti nell'archivio */
        int nFilm = contaFilm(filmdb);

        String[] ris = null;
        /* Ricerca del film desiderato */
        for (int i = 0; i < nFilm && ris == null; i++) {
            if (filmdb[i].getCodice() == codice) {
                /*
                 * Una volta trovato il film, bisogna costruire il vettore
                 * risultato. Innanzitutto va allocato. La dimensione del
                 * vettore di output è pari al numero degli attori nel cast del
                 * film aumentato di due, visto che deve contenere anche il
                 * regista ed il titolo.
                 */
                if (filmdb[i].cast != null) {
                    ris = new String[filmdb[i].cast.length + 2];
                } else {
                    ris = new String[2];
                }
                /*
                 * Dopo l'allocazione, si procede con il popolamento del
                 * vettore ris
                 */
                ris[0] = filmdb[i].titolo;
                ris[1] = filmdb[i].regia;
                if (filmdb[i].cast != null) {
                    for (int j = 0; j < filmdb[i].cast.length; j++) {
                        ris[2 + j] = filmdb[i].cast[j];
                    }
                }
            }
        }

        return ris;
    }

    public static void main(String[] args) {
        Film videoteca[] = new Film[MAX_FILM];
        Noleggio noleggi[] = new Noleggio[MAX_FILM];

        videoteca[0] = new Film(0, "Le iene", "Quentin Tarantino", 1992,
                new String[] { "Harvey Keitel", "Tim Roth" });

        videoteca[1] = new Film(1, "Pulp Fiction", "Quentin Tarantino", 1994,
                new String[] { "John Travolta", "Samuel L. Jackson" });

        videoteca[2] = new Film(2, "Dal tramonto all'alba", "Robert Rodríguez",
                1996, new String[] { "George Clooney" });

        videoteca[3] = new Film(3, "Grindhouse - A prova di morte",
                "Quentin Tarantino", 2007, new String[] { "Kurt Russel" });

        noleggi[0] = new Noleggio("Mario Rossi", 5, 1, 2012);
        noleggi[2] = new Noleggio("Mario Rossi", 30, 12, 2011);
        noleggi[3] = new Noleggio("Gianni Verdi", 13, 1, 2012);

        System.out.println("Archivio film:");
        stampaArchivio(videoteca, noleggi);
```

```
        System.out.println();

        invertiOrdine(videoteca, noleggi);
        System.out.println("Archivio film invertito:");
        stampaArchivio(videoteca, noleggi);

        System.out.println();

        Film[] ris = noleggiatiDopo(videoteca, noleggi, 1, 1, 2012);
        System.out.println("Film noleggiati dopo il 1/1/2012: ");
        for (int i = 0; i < ris.length; i++) {
            System.out.println(ris[i].titolo);
        }

        System.out.println();

        int ris2 = restituisciUtente(videoteca, noleggi, "Mario Rossi");
        System.out.println("Numero di film restituiti: " + ris2);
        System.out.println("Archivio aggiornato:");
        stampaArchivio(videoteca, noleggi);

        System.out.println();

        String[] ris3 = filmInfo(videoteca, 1);
        if (ris3 == null) {
            System.out.println("Il film non è presente in archivio");
        } else {
            System.out.println("Informazioni recuperate:");
            for (int i = 0; i < ris3.length; i++) {
                System.out.println(ris3[i]);
            }
        }

    }

}
```