INFORMATICA GRAFICA - FONDAMENTI DI INFORMATICA
Ing. Civile - Edile/Architettura – Dott. Penzo

SOLUZIONE PROVA SCRITTA DEL 19/07/2002

```
import fiji.io.SimpleReader;

public class Programma {

  public static void main(String[] args) {

    SimpleReader in = new SimpleReader();
    final int NUM_CORSI = 2;
    String tipoCorso, giorno;
    int numMaxPartecipanti, costoCorso, numOrari, i, oraInizio, oraFine,
maxCosto, numPartecipanti, j;
    Orario[] orariCorso;
    Orario orario;
    Corso corso;
    Corso[] corsiDisponibili;

/* qui si assume che vengano caricati alcuni corsi (NUM_CORSI)
* nel vettore dei corsi della Piscina.
* Parte non richiesta dal testo */

    System.out.println("Fase di inizializzazione della piscina... ");

    for(j=0;j<NUM_CORSI;j++){
      System.out.println("Inserire il tipo di corso (principianti, intermedio,
avanzato): ");
      tipoCorso = in.readString();

      System.out.println("Inserire il numero massimo di partecipanti al corso:
");
      numMaxPartecipanti = in.readInt();


      /* Per semplicita` si chiede direttamente il numero
       * corrente di partecipanti */

      System.out.println("Inserire il numero corrente di partecipanti: ");
      numPartecipanti = in.readInt();

      System.out.println("Inserire il costo del corso: ");
      costoCorso = in.readInt();

      System.out.println("Inserire il numero di orari previsti per il corso: ");
      numOrari = in.readInt();
      orariCorso = new Orario[numOrari];

      for(i=0;i<numOrari;i++){
          System.out.println("Inserire il giorno " + (i+1) + " della settimana:
");
          giorno = in.readString();

          System.out.println("Inserire l'orario di inizio corso: ");
          oraInizio = in.readInt();

          System.out.println("Inserire l'orario di fine corso: ");
          oraFine = in.readInt();
```

```
          orario = new Orario(giorno, oraInizio, oraFine);

          orariCorso[i] = orario;
       }

      /* Si procede alla creazione del corso e al suo inserimento
       * fra i corsi della Piscina */

       corso = new Corso(tipoCorso, numMaxPartecipanti, costoCorso, orariCorso);

       /* Si aggiungono alcuni partecipanti al corso
        * (valore numPartecipanti acquisito da tastiera).
        * Parte non richiesta dal testo */

       for(i=0;i<numPartecipanti;i++) corso.aggiungiPartecipante();
       Piscina.aggiungiCorso(corso);
       }

    System.out.println("Fine inizializzazione... ");

      /* si procede con l'inserimento dei dati per la creazione
     * del corso da aggiungere alla Piscina */

    System.out.println("Inserire il tipo di corso da aggiungere alla Piscina
(principianti, intermedio, avanzato): ");
    tipoCorso = in.readString();

    System.out.println("Inserire il numero massimo di partecipanti al corso: ");
    numMaxPartecipanti = in.readInt();

    System.out.println("Inserire il costo del corso: ");
    costoCorso = in.readInt();

    System.out.println("Inserire il numero di orari previsti per il corso: ");
    numOrari = in.readInt();
    orariCorso = new Orario[numOrari];

    for(i=0;i<numOrari;i++){
        System.out.println("Inserire il giorno " + (i+1) + " della settimana:
");
        giorno = in.readString();

        System.out.println("Inserire l'orario di inizio corso: ");
        oraInizio = in.readInt();

        System.out.println("Inserire l'orario di fine corso: ");
        oraFine = in.readInt();

        orario = new Orario(giorno, oraInizio, oraFine);

        orariCorso[i] = orario;
    }

    /* Si procede alla creazione del corso e al suo inserimento
     * fra i corsi della Piscina */

    corso = new Corso(tipoCorso, numMaxPartecipanti, costoCorso, orariCorso);
    Piscina.aggiungiCorso(corso);

/* Si richiede al cliente il tipo di corso desiderato.
* Si procede poi a ricercare l'elenco dei corsi disponibili
* presso la Piscina per quella determinata tipologia. */
```

```
    System.out.println("Inserire la tipologia di corso desiderato (principianti,
intermedio, avanzato): ");
    tipoCorso = in.readString();

    corsiDisponibili = Piscina.corsiDisponibiliPerTipologia(tipoCorso);

    if (corsiDisponibili != null){
       System.out.println("Inserire il costo massimo del corso: ");
       maxCosto = in.readInt();

       for(i=0;i<corsiDisponibili.length;i++){
            corso = corsiDisponibili[i];
            if(corso.getCosto() < maxCosto){
               System.out.println("Codice del corso: " + corso.getCodice());
               corso.stampaOrari();
               System.out.println("Posti disponibili: " +
corso.numPostiDisponibili());
            }
       }

    }
    else System.out.println("Non ci sono corsi disponibili!");
}
}


public class Orario {

private String giornoSettimana;
private int oraInizio;
private int oraFine;

  /**
   * costruttore giorno, inizio, fine
   */
  public Orario(String giorno, int inizio, int fine) {
      this.giornoSettimana = giorno;
      this.oraInizio = inizio;
      this.oraFine = fine;
  }

  /**
   * Gets the giornoSettimana
   */
  public String getGiornoSettimana() {
      return giornoSettimana;
  }

  /**
   * Gets the oraInizio
   */
  public int getOraInizio() {
      return oraInizio;
  }

  /**
   * Gets the oraFine
   */
  public int getOraFine() {
      return oraFine;
  }

  /**
```

```
   * toString
   */
  public String toString() {
      return("Orario: " + giornoSettimana + ", " + oraInizio + "-" + oraFine);
  }

}


public class Corso {

  private static int codCorso = 0;

  private int codice;
  private String tipologia;
  private int numMaxPartecipanti;
  private int numPartecipanti;
  private int costo;
  private Orario[] orari;

  /**
   * costruttore
   */
  public Corso(String tipologia, int maxPartecipanti, int costo, Orario[] orari)
{
    this.codice = codCorso++;
    this.tipologia = tipologia;
    this.numMaxPartecipanti = maxPartecipanti;
    this.numPartecipanti = 0;
    this.costo = costo;
    this.orari = orari;
  }

  /**
   * Gets the codice
   */
  public int getCodice() {
      return codice;
  }

  /**
   * Gets the tipologia
   */
  public String getTipologia() {
      return tipologia;
  }

  /**
   * Gets the costo
   */
  public int getCosto() {
      return costo;
  }

  /**
   * numPostiDisponibili
   */
  public int numPostiDisponibili() {
      return(numMaxPartecipanti - numPartecipanti);
  }


  /**
```

```
   * aggiungiPartecipante
   */
  public void aggiungiPartecipante() {
      if (numPostiDisponibili() > 0)
            numPartecipanti++;
     else System.out.println("Posti disponibili esauriti!");
  }

  /**
   * stampaOrari
   */
  public void stampaOrari() {
      int i;
      for(i=0; i<orari.length; i++)
            System.out.println(orari[i]);
  }

}


public class Piscina {

  private static final int MAX_CORSI = 50;

  private static Corso[] corsi = new Corso[MAX_CORSI];
  private static int numCorsi = 0;


  /**
   * aggiungiCorso
   */
  public static void aggiungiCorso(Corso c) {
      if (numCorsi < MAX_CORSI){
            corsi[numCorsi] = c;
            numCorsi++;
      }
  }

  /**
   * corsiDisponibiliPerTipologia.
   * E` necessario verificare che ci sia posto per una nuova iscrizione.
   */
  public static Corso[] corsiDisponibiliPerTipologia(String tipologia) {
      Corso[] corsiDisponibili, temp = new Corso[numCorsi];
      int i, j;
      String tipo;

      j = 0;
      for(i=0; i<numCorsi; i++){
            tipo = corsi[i].getTipologia();
            if(tipo.equals(tipologia) && corsi[i].numPostiDisponibili() > 0){
                  temp[j] = corsi[i];
                  j++;
            }
      }
      if(j>0){
            corsiDisponibili = new Corso[j];
            for(i=0;i<j;i++) corsiDisponibili[i] = temp[i];
            return corsiDisponibili;
      }
      else return null;
  }
}
```