Cognome _____ Nome _____
Matricola _____ Postazione PC _____

Corso di Laurea in Ingegneria Gestionale
Esame di Informatica - a.a. 2014
04 Luglio 2014

Testo

Il database di un bar è costituito da due vettori paralleli. Il primo è denominato “tables” e contiene oggetti di tipo “Tavolo” che rappresentano i tavoli presenti all'interno di un locale. Il secondo vettore è denominato “orders” e contiene oggetti di tipo “Ordine” che rappresentano le informazioni di ogni singolo ordine di un tavolo. Ad ogni tavolo può corrispondere più di un ordine, in quel caso le informazioni del tavolo saranno replicate.
Per ogni tavolo presente nella posizione i-esima del vettore “tables” si troveranno le informazioni relative ad un ordine nella corrispondente posizione del vettore “orders”. Nel caso che il tavolo in posizione i-esima non abbia alcun ordine associato, nella posizione corrispondente nel vettore “orders” sarà presente un riferimento *null*. Entrambi i vettori hanno dimensione pari alla costante “MAX_ELEM” (inizializzata a 1024). Se il numero di tavoli contenuti nell’archivio è inferiore a “MAX_ELEM”, i primi elementi del vettore conterranno gli oggetti di tipo “Tavolo”, mentre gli altri conterranno riferimenti *null*. Tutti gli elementi *null* del vettore “tables” si devono trovare alla fine del vettore e non possono trovarsi in mezzo agli elementi validi.

Le classe Tavolo contiene le informazioni relative ad un tavolo:

```
public class Tavolo {
	private int id;
	public String nomeCliente;
	public String cognomeCliente;
	public String posizione;

	public Tavolo(int myId, String nome, String cognome, String posizione){
		this.id = myId;
		this.nomeCliente = nome;
		this.cognomeCliente = cognome;
		this.posizione = posizione;
	}

	public int getId(){
		return id;
	}

	public String toString(){
		return "Tav.#" + id + " (" + posizione + ") "
			+ nomeCliente + " " + cognomeCliente;
	}
}
```

La classe Ordine contiene le informazioni relative ai singoli ordini di un tavolo.

```
public class Ordine {

	static private int numeroProgressivo = 0;
	private int numeroOrdine;

	public String prodotto;
	public int numero;
	public double prezzo;
	public boolean servito;
	public String orario;

	public Ordine(String product, int quantity, double price, String time, boolean served){
		numeroOrdine = numeroProgressivo++;
		prodotto = product;
		numero = quantity;
		prezzo = price;
		orario =  time;
		servito = served;
	}
```

```
        public String toString(){
                return "[ord." + numeroOrdine + "] "
                        + "num. " + numero + " " + prodotto;
        }
}
```

Si consiglia di procedere implementando un metodo e successivamente la parte del main che utilizza tale metodo. Le varie operazioni devono essere eseguite sulla porzione significativa dell'archivio, cioè la porzione di "tables" che non contiene riferimenti "null".

A) Scrivere il metodo statico:

```
public static int minuti(String orario)
```

Il metodo deve convertire la stringa orario (che può trovarsi sia nel formato *h:mm* che nel formato *hh:mm*) nel numero di minuti trascorsi a partire dalla mezzanotte (es. l'orario 1:10 sarà convertito nel'intero 70). Se si ha la necessità di convertire una stringa in intero, si può utilizzare la funzione di libreria *Integer.parseInt(s)* che converte la stringa *s* in un intero restituito come risultato.

B) Scrivere il metodo statico:

```
public static void ordinaArrivo(Tavolo[] tavoli, Ordine[] ordini)
```

Il metodo deve ordinare, nel vettore "ordini", gli elementi secondo l'ordine di arrivo, considerando ora e minuto di arrivo (es. chi arriva alle 11:15 si trova prima di chi arriva alle 11:20). L'orario può trovarsi sia nel formato *h:mm* che nel formato *hh:mm*. Se ci sono ordini nulli questi dovranno trovarsi in fondo al vettore "ordini" e non dovranno essere considerati per l'ordinamento. Lasciare l'intero archivio in stato consistente.

C) Scrivere il metodo statico:

```
public static double contoHappyHours(Tavolo[] tavoli,Ordine[] ordini, int id)
```

Il metodo deve restituire il conto totale relativo al tavolo il cui id è specificato come parametro. Nel computo si devono considerare solo gli ordini effettivamente serviti e si deve scontare del 20% tutti gli ordini effettuati tra le 19:00 e le 21:00.

D) Scrivere il metodo statico:

```
public static int[] modificaPosizione(Tavolo[] tavoli, String pos)
```

Per tutti i tavoli la cui posizione corrisponde al parametro pos, si modifichi il campo posizione dell'oggetto corrispondente, trasformando tutte le lettere in minuscolo. Il metodo deve restituire due parametri, il numero di modifiche effettuate e l'indice della prima posizione dell'array "tavoli" eventualmente modificata.

E) Scrivere il metodo main che:

- definisca ed inizializzi i vettori "tables" e "orders" secondo i valori riportati in tabella. La stampa dell'archivio consiste nello stampare le informazioni di ogni tavolo e gli ordini associati (se ve ne sono). Si utilizzino correttamente i relativi metodi toString() implementati nelle due classi. Per ogni ordine si stampi anche l'orario.

Id	Nome e Cognome	Posizione	OrarioOrdine	Prodotto	Quantità	Costo	Servito
0	Mario Rossi	Ristopub	20:56	Acqua	1	2	Sì
1	Pietro Rossi	Pizzeria					
0	Mario Rossi	Ristopub	21:05	Birra	2	3.50	Sì
2	Gio' Verdi	Veranda	20:55	Vino	1	10	Sì
2	Gio' Verdi	Veranda	21:10	Acqua	1	2	Sì

- Avvalendosi del metodo al punto A stampi tutti gli ordini con l'informazione dell'orario convertito.
- Ordini l'intero archivio utilizzando il metodo del punto B e stampi a video l'archivio prima e dopo l'ordinamento.
- Utilizzando il metodo C, stampi le informazioni relative al Tavolo con id 2.
- Modifichi il campo posizione dei vari tavoli con posizione "Ristopub", utilizzando il metodo del punto D. Al termine dell'operazione, se essa è avvenuta con successo si stampi l'archivio aggiornato, il numero di operazioni effettuate e l'indice del primo tavolo modificato; altrimenti si stampi un messaggio di errore.

Soluzione

```
public class Appello2 {

	static final int MAX_ELEM = 1024;

	// scambia l'elemento in posizione i-esima con quello in posizione j-esima
	public static void scambiaOrdine(Ordine[] v, int i, int j){
		Ordine tmp = v[i];
		v[i] = v[j];
		v[j] = tmp;
	}

	public static void scambiaTavolo(Tavolo[] v, int i, int j){
		Tavolo tmp = v[i];
		v[i] = v[j];
		v[j] = tmp;
	}

	public static int contaTavoli(Tavolo[] tavoli){
		int count = 0;
		while (count < MAX_ELEM && tavoli[count] != null){
			count++;
		}
		return count;
	}

	/**
	 * Restituisce la prima occorrenza del carattere c a partire
	 * da pos
	 */
	public static int indexof(String s, int pos, char c){
		int index = -1;
		boolean trovato = false;
		for (int i = pos; i<s.length() && !trovato; i++){
			if (s.charAt(i) == c){
				index = i;
				trovato = true;
			}
		}
		return index;
	}

	/**
	 * Restituisce la sottostringa di s a partire da pos1 incluso a pos2
	 * escluso
	 */
	public static String substring(String s, int pos1, int pos2){
		String sub = "";
		for (int i = pos1; i<pos2; i++){
			sub += s.charAt(i);
		}
		return sub;
	}

	/*
	 * A. Il metodo deve convertire la Stringa orario (che può
	 * trovarsi sia nel formato h:mm che nel formato hh:mm) nel
	 * numero di minuti trascorsi a partire dalla mezzanotte.
	 */
	public static int minuti(String orario){
		int o_s = indexof(orario, 0, ':');
```

```
		String ora_s = substring(orario, 0, o_s);
		int ora = Integer.parseInt(ora_s);

		String min_s = substring(orario, o_s+1, orario.length());
		int min = Integer.parseInt(min_s);

		return (ora * 60) + min;
	}

	public static int contaOrdini(Ordine[] ordini){
		int count = 0;
		for (int i=0; i<MAX_ELEM; i++)
			if (ordini[i]!= null)
				count++;
		return count;
	}

	/*
	 * B. Il metodo deve ordinare, nel vettore “ordini”, gli elementi
	 * secondo l'ordine di arrivo, considerando ora e minuto di arrivo
	 * (es. chi arriva alle 11:15 si trova prima di chi arriva alle 11:20).
	 * Se ci sono ordini nulli questi dovranno trovarsi in fondo al vettore
	 * “ordini” e non dovranno essere considerati per l'ordinamento.
	 * Lasciare l'archivio in stato consistente.
	 */
	public static void ordinaArrivo(Tavolo[] tavoli,
			Ordine[] ordini){
		int n = contaTavoli(tavoli);
		for (int i=0; i<n-1; i++){
			for (int j=i+1; j<n; j++){
				if (ordini[i]== null && ordini[j]!= null){
					scambiaTavolo(tavoli, i, j);
					scambiaOrdine(ordini, i, j);
				}
			}
		}

		n = contaOrdini(ordini);
		for (int i = 0; i < n-1; i++){
			int min = i;
			for (int j=i+1; j < n; j++){
				if (minuti(ordini[j].orario) <
						minuti(ordini[min].orario)) {
					min = j;
				}

			}
			scambiaTavolo(tavoli, i, min);
			scambiaOrdine(ordini, i, min);
		}
	}


	/*
	 * C. Il metodo deve restituire il conto totale relativo
	 * al tavolo il cui id è specificato come parametro. Nel computo si
	 * devono considerare solo gli ordini effettivamente serviti e
	 * si deve scontare del 20% tutti gli ordini effettuati tra le 19:00
	 * e le 21:00.
	 */
	public static double contoHappyHour(Tavolo[] tavoli,
			Ordine[] ordini, int id){
		int n = contaTavoli(tavoli);
		double tot = 0;
		final double SCONTO_HAPPY = 0.2;
		for (int i=0; i<n; i++){
			if (tavoli[i].getId() == id){
				if (ordini[i] != null && ordini[i].servito){
					if (minuti(ordini[i].orario) < minuti("19:00") ||
```

```
                                    minuti(ordini[i].orario) > minuti("21:00"))
                                    tot += ordini[i].prezzo * ordini[i].numero;
                              else {
                                    tot += (ordini[i].prezzo -
                                ordini[i].prezzo * SCONTO_HAPPY) * ordini[i].numero;
                              }
                        }
                  }
            }
            return tot;
}

/*
 * D. Per tutti i tavoli la cui posizione corrisponde al parametro
 * pos, si modifichi il campo posizione trasformando tutte le lettere
 * in minuscolo. Il metodo deve restituire due parametri, il numero
 * di modifiche effettuate e l'indice della prima posizione dell'array
 * "tavoli" eventualmente modificata.
 */
public static int[] modificaPosizione(Tavolo[] tavoli,
                                      String pos){
      int n = contaTavoli(tavoli);
      int counter = 0;
      int first_index = -1;
      for (int i=0; i < n; i++){
            if (tavoli[i].posizione.equals(pos)){
                  if (counter < 1)
                        first_index = i;
                  counter++;
                  tavoli[i].posizione = tavoli[i].posizione.toLowerCase();
            }
      }
      int [] myres = new int[2];
      myres[0] = counter;
      myres[1] = first_index;
      return myres;
}

public static void stampaDB(Tavolo[] tavoli,
            Ordine[] ordini){
      int n = contaTavoli(tavoli);
      for (int i=0; i < n; i++){
            if (ordini[i] != null){
                  System.out.println(tavoli[i] + " - " + ordini[i] + " "
                              + ordini[i].orario);
            }
            else {
                  System.out.println(tavoli[i]);
            }
      }
}

public static void main(String[] args) {
      Tavolo[] tables = new Tavolo[MAX_ELEM];
      Ordine[] orders = new Ordine[MAX_ELEM];

      tables[0] = new Tavolo(0, "Mario", "Rossi", "Ristopub");
      tables[1] = new Tavolo(1, "Pietro", "Rossi", "Pizzeria");
      tables[2] = new Tavolo(0, "Mario", "Rossi", "Ristopub");
      tables[3] = new Tavolo(2, "Gio'", "Verdi", "Veranda");
      tables[4] = new Tavolo(2, "Gio'", "Verdi", "Veranda");

      orders[0] = new Ordine("Acqua", 1, 2, "20:56", true);
      orders[2] = new Ordine("Birra", 2, 3.50, "21:05", true);
      orders[3] = new Ordine("Vino", 2, 10, "20:55", true);
      orders[4] = new Ordine("Acqua", 1, 2, "21:10", true);

      System.out.println("\nA.");
      for (int i=0; i < MAX_ELEM; i++){
```

```
            if (orders[i] != null)
                System.out.println("La conversione dell'orario per l'ordine "
                    + orders[i] + " è " + minuti(orders[i].orario));
        }
        System.out.println("\nB.");
        stampaDB(tables,orders);
        ordinaArrivo(tables,orders);
        System.out.println("\nDopo l'ordinamento:");
        stampaDB(tables,orders);

        System.out.println("\nC.");
        int tid = 2;
        double totTav0 = contoHappyHour(tables, orders, tid);
        System.out.println("Il totale del Tavolo #" + tid +" è " + totTav0 + "€");

        System.out.println("\nD.");
        int[] res = modificaPosizione(tables, "Ristopub");
        if (res[0] > 0 ){
            System.out.println("Modificati " + res[0] + " tavoli "
                + "di cui il primo in posizione " + res[1] + " dell'array");
            stampaDB(tables, orders);
        }
        else {
            System.out.println("Nessuna modifica");
        }
}
```