Laboratorio di Programmazione
Laurea in Bioinformatica

10 marzo 2008

1 Soluzioni degli Esercizi di ricapitolazione (Java)

1.1 Esercizio 1

Si progetti una classe di nome CodiceIntero, i cui oggetti codificano numeri interi. Ogni oggetto della classe rappresenta un numero intero le cui cifre sono ordinate nel verso opposto a quello del numero con cui l'oggetto è stato creato. Se, ad esempio, il numero in questione è 74235 allora il corrispondente oggetto CodiceIntero rappresenta il codice 53247. Dopo avere previsto nella classe le necessarie variabili d'istanza, si progettino i seguenti metodi:

1. (costruttore/codificatore) public CodiceIntero (int numero)
2. (decodificatore) public int decodifica()
3. (stampa del codice) public String toString()

Infine, si dia un metodo main che a partire da un numero intero istanzia un oggetto CodiceIntero e successivamente stampa il valore del relativo codice e della sua decodifica.

Nota bene: soluzioni che si basino su preesistenti metodi di libreria che effettuano l'inversione di stringhe NON verranno accettate.

```
import prog.io.ConsoleOutputManager;

public class CodiceIntero {
    private int codice;

    // Costruttore
    public CodiceIntero(int num) {
        codice = num;
    }

    // Codifica
    public static CodiceIntero codifica(int numero) {
        return new CodiceIntero(converte(numero));
    }

    // Decodifica
    public int decodifica() {
        return converte(codice);
    }
```

```
    // Stampa
    public String toString() {
        return "" + codice;
    }

    // Convertitore dal/al codice
    private static int converte(int numero) {
        int risultato=0;
        while (numero>0) {
            int cifra = numero % 10;
            risultato = 10*risultato + cifra;
            numero /= 10;
        }
        return risultato;
    }

    // Main
    public static void main(String[] args) {
        ConsoleOutputManager schermo = new ConsoleOutputManager();

        int numero = 47532;

        schermo.println(numero);
        schermo.println(codifica(numero));
        schermo.println(codifica(numero).decodifica());
    }
}
```

1.2 Esercizio 2

Si progetti una classe di nome Numeri, i cui oggetti specificano un array a di stringhe di dimensione 100. Ciascun elemento dell'array contiene una stringa binaria il cui simbolo iniziale è 1, mentre tutti i restanti simboli sono 0.

Dopo avere previsto nella classe le necessarie variabili d'istanza, si progettino i seguenti metodi:

1. public Numeri (String numero), il quale attraverso l'uso di una procedura ricorsiva (che eventualmente si appoggi su un metodo ausiliario) assegna agli elementi dell'array stringhe ottenute estraendo via via dal parametro numero sottostringhe secondo il formato previsto. Se, ad esempio, numero = "10011000", allora si avrà a[0]="100", a[1]="1", a[2]="1000", a[3]=null, . . . , a[99]=null.
2. public String toString(), il quale restituisce una stringa formata concatenando gli elementi dell'array (esclusi i riferimenti null) attraverso la sequenza di escape newline. Nell'esempio precedente, toString() restituirà

   ```
   100
   1
   1000
   ```

Infine, si dia un metodo main che a partire da una stringa binaria obbligatoriamente iniziata dal simbolo 1 istanzia un oggetto Numeri e, successivamente, stampa l'oggetto.

Nota bene: soluzioni che si basino su algoritmi di tipo iterativo per la creazione dell'oggetto NON verranno accettate.

```
import prog.io.ConsoleOutputManager;

public class Numeri {
    private String[] numeri;

    // Costruttore
    public Numeri(String numero) {
```

```
        numeri = new String[100];
        int contatore = -1;
        creaNumero(numero,contatore);
    }

    // Metodo ricorsivo ausiliario
    private void creaNumero(String numero, int contatore) {
        if(numero.length()>0) {
            if (numero.charAt(0)=='1')
                numeri[++contatore] = "1";
            else
                numeri[contatore] = numeri[contatore] + "0";
        creaNumero(numero.substring(1),contatore);
        }
    }

    // Stampa
    public String toString() {
        String t = "";
        for(String s : numeri)
            if (s != null)
                t = t + s + "\n";
        return t;
    }

    // Main

    public static void main(String[] args) {
        ConsoleOutputManager schermo = new ConsoleOutputManager();
        Numeri numero = new Numeri("10000011100101000");
        schermo.println(numero);
    }
}
```

1.3 Esercizio 3

Si modifichi la classe CarteFrancesi in modo che sia possibile simulare la distribuzione di una mano di poker ad un unico giocatore. Per modellare la singola mano si consiglia di adoperare un ulteriore array di interi mano[i] di lunghezza 5.

I metodi aggiuntivi siano:

1. un metodo public void daiCarte(), che dà le 5 carte
2. un metodo public String vediCarte(), che visualizza le 5 carte in mano
3. un metodo public void accomodo(int[] cambio), che sostituisce le carte nelle posizioni indicate nel vettore cambio con altrettante carte prese dal mazzo
4. un metodo public String combinazione(), che individua la combinazione finale (coppia, doppia coppia, tris, scala, full, colore, poker, scala reale) .

[SUGGERIMENTO: per individuare la combinazione, è consigliabile ordinare modulo 13 le 5 carte (cioè in base al numero e non al seme) e ricavare un vettore delle differenze. Su questo vettore, si possono facilmente individuare i pattern delle combinazioni.

```
Es:
[#0##] -> coppia             [000#] -> poker
[00##] -> tris               [00#0] -> full
... etc.
```

La classe chiamante , estensione di UsaCarte, dà all'unico giocatore 5 carte dopo aver mescolato il mazzo, gli mostra le 5 carte, gli chiede quante e quali carte vuole cambiare e gli comunica infine la combinazione finale individuata.

```
package myclasses;
import prog.io.ConsoleOutputManager;

public class CarteFrancesi2 {
    private static final int NUM_CARDS = 52;
    private int[] carte = new int[NUM_CARDS];
    private static final int NUM_CARDS_MANO = 5;
    private int[] mano = new int[NUM_CARDS_MANO];

    public CarteFrancesi2() {
        for(int i=0; i<NUM_CARDS; i++)
          carte[i] = i+1;
        for(int j=0; j<NUM_CARDS_MANO; j++)
            mano[j] = 0;
    }

    public String vediTesta() {                      // stampa la testa del mazzo
        return toString(carte[0]);
    }

    public void spostaTesta() {                     // sposta la prima carta in fondo
        int temp = carte[0];
        for(int i=0; i<NUM_CARDS-1; i++)
            carte[i] = carte[i+1];
        carte[NUM_CARDS-1] = temp;
    }

    public void mescola(int volte) {
        int spessore_mazzetto;
        int[] mazzetto;
        for (int i=0; i<volte; i++) {
            spessore_mazzetto = 1+(int)(NUM_CARDS/2*Math.random());  // sceglie lo spessore del mazzetto
            mazzetto = new int[spessore_mazzetto];
            for(int j=0; j<spessore_mazzetto; j++)                // estrae il mazzetto dalla testa del mazzo
                mazzetto[j] = carte[j];
```

```
            for(int j=0; j<NUM_CARDS-spessore_mazzetto; j++)       // ridetermina la testa del mazzo
                carte[j] = carte[j+spessore_mazzetto];
            for(int j=1; j<=spessore_mazzetto; j++) {  // reinserisce il mazzetto in coda alternando le carte
                carte[NUM_CARDS-2*j+1] = mazzetto[spessore_mazzetto-j];
                carte[NUM_CARDS-2*j] = carte[NUM_CARDS-spessore_mazzetto-j];
            }
        }
    }

    private static String toString(int num) {
    String s;

    if (num == 0)
        s = "none";
    else {
        switch (num % (NUM_CARDS/4)) {
        case 1:
        s = "asso di ";
        break;
        case 11:
        s = "jack di ";
        break;
        case 12:
        s = "regina di ";
        break;
        case 0:
        s = "re di ";
        break;
        default:
        s = (num % (NUM_CARDS/4)) + " di ";
        }

        switch ((num-1) / (NUM_CARDS/4)) {
        case 0:
        s += "cuori";
        break;
        case 1:
        s += "quadri";
        break;
        case 2:
        s += "fiori";
        break;
        case 3:
        s += "picche";
        }
    }
    return s;
    }

public void daiCarte(){
    for(int i=0; i<NUM_CARDS_MANO; i++)
        mano[i] = carte[i];
}

public String vediCarte(){
    String s=toString(mano[0])+"\n";
    for(int i=1; i<NUM_CARDS_MANO; i++)
        s+=toString(mano[i])+"\n";
    return s;
}

public void accomodo(int[] cambio){
```

```
    int len = cambio.length;
    if (len!=0){
        for(int i=0; i<len; i++)
            mano[cambio[i]-1]=carte[NUM_CARDS_MANO+i];
    }
}

public String combinazione(){

    int coppie; // 0,1,2
    boolean tris, full, poker, scala, colore;
    int[] sorted_mano=new int[NUM_CARDS_MANO];
    int[] colori=new int[NUM_CARDS_MANO];

   /******** check colore***********/
    colore = true;
    colori[0]=mano[0]/(NUM_CARDS/4);
    for(int i=1; i<NUM_CARDS_MANO; i++){
        colori[i]=mano[i]/(NUM_CARDS/4);
        if(colori[i]!=colori[i-1])
            colore = false;
    }
   /*********************************/

   /******** sort delle carte ********/
     int temp;
     for(int i=0; i<NUM_CARDS_MANO; i++){
        for(int j=i+1; j<NUM_CARDS_MANO; j++){
            if(((mano[i]-1)%(NUM_CARDS/4))>((mano[j]-1)%(NUM_CARDS/4))){
                temp = mano[i];
                mano[i]=mano[j];
                mano[j]=temp;
            }

        }
        sorted_mano[i]=(mano[i]-1)%(NUM_CARDS/4)+1;
     }
   /*********************************/

   /******** check scala**************/
   scala=true;
   for(int i=1; i<NUM_CARDS_MANO; i++){
       if((sorted_mano[i]-sorted_mano[i-1])!=1){
            scala=false;
            break;
       }
    }
   /*********************************/

   /******** check poker,full,tris,dcoppia,coppie**************/
   poker=false;
   tris=false;
   full=false;
   coppie=0;
   int nzeriadiacenti=0;

   int[] diff=new int[NUM_CARDS_MANO-1];
   for(int i=0; i<NUM_CARDS_MANO-1; i++)
        diff[i]=(sorted_mano[i+1]-sorted_mano[i]);

   int i=0;
    while(i<NUM_CARDS_MANO-1){
```

```
    nzeriadiacenti=0;
    while(i<NUM_CARDS_MANO-1 && diff[i]==0){
         nzeriadiacenti+=1;
         i+=1;
    }
    if(nzeriadiacenti==3) poker=true;
        else if(nzeriadiacenti==2) tris=true;
            else if(nzeriadiacenti==1) coppie+=1;
    i+=1;
   }

   if(tris&&(coppie==1)) full=true;
   /*********************************/


    String s="Nessuna combinazione";
    if(colore&&scala)
        s="Scala Reale";
        else if(poker)
            s="Poker";
            else if(colore)
                s="Colore";
                else if(full)
                    s="Full";
                    else if(scala)
                        s="Scala";
                        else if(tris)
                           s="Tris";
                           else if(coppie==2)
                                s="Doppia coppia";
                                else if(coppie==1)
                                    s="Coppia";
    return s;
    }
}


import prog.io.*;
import myclasses.CarteFrancesi2;
class UsaCarte2 {

    public static void main(String[] args) {

    ConsoleInputManager in = new ConsoleInputManager();
    ConsoleOutputManager out = new ConsoleOutputManager();
    CarteFrancesi2 carte = new CarteFrancesi2();

    carte.mescola(200);
    carte.daiCarte();
    out.println("Carte in mano al giocatore:\n" + carte.vediCarte());

    int ncartecambio=in.readInt("Quante carte vuoi cambiare (0-5) ?: ");
    int cambio[]=new int[ncartecambio];
    for(int i=0; i<ncartecambio; i++)
        cambio[i]=in.readInt("Posizione della carta da cambiare (1-5): ");
    carte.accomodo(cambio);
    out.println("Carte in mano al giocatore:\n" + carte.vediCarte());
    out.println("Combinazione con valore maggiore: " + carte.combinazione());
    }
}
```