Corso di Laurea Ingegneria Informatica

Fondamenti di Informatica 1

Dispensa E10

Esercizi su Array e Oggetti

C. Limongelli

Dicembre 2007

Contenuti

- La classe Libro e un esempio di gestione
- Verifiche universali ed esistenziali su array di stringhe
  - Verifica dell'esistenza di una stringa, in un array, formata tutta da caratteri maiuscoli
  - Verifica dell'esistenza di una stringa, in un array, formata sia da lettere che da cifre

Esercizi

- **Data la classe Libro per rappresentare oggetti libro con il nome dell'autore, il titolo e il numero di pagine e con i relativi metodi d'istanza**
  - **scrivere un metodo che, ricevendo come parametro un array di oggetti Libro, calcola e restituisce un array di oggetti Libro dello stesso autore**
  - **scrivere un metodo che, ricevendo come parametro un array di oggetti Libro, calcola e restituisce l'oggetto Libro con il massimo numero di pagine**

L’oggetto Libro

- Autore (autore)
- Titolo (titolo)
- Numero di pagine (pagine)

```
private String autore;
private String titolo;
private int pagine;

// Costruttore della classe
public Libro (String a, String t, int p){

    this.autore = a;
    this.titolo = t;
    this.pagine = p;
}
```

La classe Libro…

I metodi d'istanza della classe

- **String getAutore()**
  - **Restituisce la stringa corrispondente al nome dell'autore**
- **String getTitolo()**
  - **Restituisce la stringa corrispondente al titolo del Libro**
- **int getPagine()**
  - **Restituisce l'intero corrispondente al numero di pagine del Libro**
- **String toString()**
  - **Restituisce una stringa (su più righe) che contiene tutti I dati del libro**

… La classe Libro

- I metodi della classe:

```
public String getAutore(){
    return this.autore;
}

public String getTtitolo() {
    return this.titolo ;
}

public int getPagine() {
    return this.pagine ;
}

public  String toString(){
return    ("AUTORE:    " + this.autore + "\n" +
           "TITOLO:    " + this.titolo + "\n"+
           "N. PAGINE: " + this.pagine + "\n");
}
```

Creare un elenco di libri…

La classe GestioneLibro

- Crea e restituisce un array di oggetti Libro
- Restituisce il libro con il maggior numero di pagine
- Restituisce tutti i titoli dei libri che hanno un determinato autore (dato in input).
- … si possono aggiungere altri metodi ….

…Creare un array di libri…

- **Acquisisce i dati da input:**

```
//crea un array di n libri
public static Libro[] creaElenco(int n){
     Libro[] l;
     String a,t;
     int i;
     int p;
     l = new Libro[n];

     System.out.println("Dati del libro: ");
     for (i=0; i<n; i++){
            System.out.println("inserisci autore ");
            a = Lettore.in.leggiString();
            System.out.println("inserisci titolo ");
            t = Lettore.in.leggiString();
            System.out.println("inserisci numero pagine ");
            p = Lettore.in.leggiInt();

            l[i] = new Libro(a,t,p);
     }
     return l;
}
```

…Creare un array di libri…

- **Esplicitamente per la classe di test**

```
elencoLibri = new Libro[10];
elencoLibri[0] = new Libro("Esopo","Le storie dell'asino",20);
elencoLibri[1] = new Libro("Italo Calvino", "Il visconte
                                              dimezzato",158);
elencoLibri[2] = new Libro("Esopo","Le storie del cane",20);
elencoLibri[3] = new Libro("Esopo","Le favole piu' belle",98);
elencoLibri[4] = new Libro("Giovanni Rodari","Filastrocche
                                            lunghe e corte",80);
elencoLibri[5] = new Libro("Italo Calvino","le
                                             cosmicomiche",250);
elencoLibri[6] = new Libro("Gianni Rodari","Enciclopedia della
                                               favola",1120);
elencoLibri[7] = new Libro("Italo Calvino","Il barone
                                               rampante",135);
elencoLibri[8] = new Libro("Fratelli Grimm","Raperonzolo",36);
elencoLibri[9] = new Libro("Italo Calvino","Il cavaliere
                                              inesistente",124);
```

Calcolare i dati richiesti

- **Stampa il libro con il massimo numero di pagine**
  - Il metodo maxPagine fornisce l'indice dell'array relativo al libro con più pagine;
  - L'oggetto Libro con più pagine è identificato da:
    ```
    elencoLibri[maxPagine(elencoLibri)]
    ```
  - La stampa viene effettuata per mezzo del metodo toString
    ```
    System.out.println(elencoLibri[maxPagine(elencoLibri)].toString());
    ```

- **Stampa l'elenco dei libri con lo stesso autore**
  - Il metodo stessoAutore prende l'elenco dei libri, un autore e calcola un nuovo array che contiene i libri di quell'autore
    ```
    stessoAutore(elencoLibri,"Italo Calvino")
    ```
  - La stampa viene effettuata per mezzo del metodo stampaElenco
    ```
    stampaElenco(stessoAutore(elencoLibri,"Italo Calvino"))
    ```

- **Segnature dei metodi**
- **stampaElenco**
  - `stampaElenco(Libro[] e)`
- **stessoAutore**
  - `Libro[] stessoAutore(Libro[] e,String autore)`
- **maxPagine**
  - `int maxPagine(Libro[] e)`

stessoAutore…

- **Prima scorro l'array di libri per calcolare il numero di libri che hanno l'autore richiesto e per dimensionare il nuovo array da calcolare**
  - **Se l'attributo autore dell'i-esimio libro, cioè dell'oggetto e[i] è uguale all'autore dato come parametro al metodo, allora incremento il contatore k**

```
public static Libro[] stessoAutore(Libro[] e, String autore){
       int i,k;  //k rappresenta la lunghezza del nuovo array
       Libro[] app;

       /* scorro l'array solo per dimensionare app */
       k=0;
       for (i=0; i<e.length; i++)
              if (e[i].getAutore().equals(autore))
                     k++;
// …… continua …
```

… stessoAutore

- Creo l'array app e scorro nuovamente l'array e per inserire in app i k libri che hanno l'autore uguale alla stringa parametro del metodo
  - Bisogna usare il metodo equals definito per le stringhe

```
//…. continua …
/* creo l'array di k componenti */
    app = new Libro[k];

/* scorro nuovamente l'array e per inserire in app
          i k libri che hanno l'autore richiesto */
    k = 0;
    for (i=0; i<e.length; i++)
          if (e[i].getAutore().equals(autore)){
                app[k] = e[i];
                k++;
          }
    return app;
}
```

maxPagine

- Il metodo maxPagine() prende in input l'array e (elenco libri) e restituisce l'indice dell'array che corrisponde al libro che presenta il maggior numero di pagine in e

```
public static int maxPagine(Libro[] e){
int i,indMax,max;

max = e[0].getPagine();
indMax = 0;
for (i=1; i<e.length; i++){
       if (e[i].getPagine()>max){
              max = e[i].getPagine();
              indMax = i;
       }
}
return indMax;
}
```

La classe test

```
public static void testElencoLibri(){
   Libro[] elencoLibri;
   //…
   elencoLibri = new Libro[10];
   elencoLibri[0] = new Libro("Esopo","Le storie dell'asino",20);
   elencoLibri[1] = new Libro("Italo Calvino", "Il visconte dimezzato",158);
   elencoLibri[2] = new Libro("Esopo","Le storie del cane",20);
   elencoLibri[3] = new Libro("Esopo","Le favole piu' belle",98);
   elencoLibri[4] = new Libro("Giovanni Rodari","Filastrocche lunghe e corte",80);
   elencoLibri[5] = new Libro("Italo Calvino","le cosmicomiche",250);
   elencoLibri[6] = new Libro("Gianni Rodari","Enciclopedia della favola",1120);
   elencoLibri[7] = new Libro("Italo Calvino","Il barone rampante",135);
   elencoLibri[8] = new Libro("Fratelli Grimm","Raperonzolo",36);
   elencoLibri[9] = new Libro("Italo Calvino","Il cavaliere inesistente",124);
   stampaElenco(elencoLibri);
   System.out.println("il libro con il max. numero di pagine e':");
   System.out.println(elencoLibri[maxPagine(elencoLibri)].toString());
   //stampa l'elenco di libri con lo stesso autore
   System.out.println("stampa di tutti i libri di Italo Calvino");
   stampaElenco(stessoAutore(elencoLibri,"Italo Calvino"));
   System.out.println("stampa di tutti i libri di Esopo");
   stampaElenco(stessoAutore(elencoLibri,"Esopo"));
   System.out.println("stampa di tutti i libri di Wilde");
   stampaElenco(stessoAutore(elencoLibri,"Oscar Wilde"));
   }//end test
```

Esempi d’esecuzione…

```
==================================
========== ELENCO LIBRI ==========
==================================
AUTORE:    Esopo
TITOLO:    Le storie dell'asino
N. PAGINE: 20


il libro con il max. numero di pagine e':
AUTORE:    Esopo
TITOLO:    Le storie dell'asino
N. PAGINE: 20
```

```
==================================
========== ELENCO LIBRI ==========
==================================
AUTORE:    Esopo
TITOLO:    Le storie dell'asino
N. PAGINE: 20

AUTORE:    Italo Calvino
TITOLO:    Il visconte dimezzato
N. PAGINE: 158

AUTORE:    Esopo
TITOLO:    Le storie del cane
N. PAGINE: 20


il libro con il max. numero di pagine e':
AUTORE:    Italo Calvino
TITOLO:    Il visconte dimezzato
N. PAGINE: 158
```

… Esempi d'esecuzione

```
==================================
========== ELENCO LIBRI ==========
==================================
AUTORE:    Esopo
TITOLO:    Le storie dell'asino
N. PAGINE: 20

AUTORE:    Italo Calvino
TITOLO:    Il visconte dimezzato
N. PAGINE: 158

AUTORE:    Esopo
TITOLO:    Le storie del cane
N. PAGINE: 20

AUTORE:    Esopo
TITOLO:    Le favole piu' belle
N. PAGINE: 98

AUTORE:    Gianni Rodari
TITOLO:    Filastrocche lunghe e corte
N. PAGINE: 80

AUTORE:    Italo Calvino
TITOLO:    le cosmicomiche
N. PAGINE: 250
AUTORE:    Gianni Rodari
TITOLO:    Enciclopedia della favola
N. PAGINE: 1120

AUTORE:    Italo Calvino
TITOLO:    Il barone rampante
N. PAGINE: 135

AUTORE:    Fratelli Grimm
TITOLO:    Raperonzolo
N. PAGINE: 36
AUTORE:    Italo Calvino
TITOLO:    Il cavaliere inesistente
N. PAGINE: 124


il libro con il max. numero di pagine e':
AUTORE:    Gianni Rodari
TITOLO:    Enciclopedia della favola
N. PAGINE: 1120

stampa di tutti i libri di Esopo
==================================
========== ELENCO LIBRI ==========
==================================
AUTORE:    Esopo
TITOLO:    Le storie dell'asino
N. PAGINE: 20

AUTORE:    Esopo
TITOLO:    Le storie del cane
N. PAGINE: 20

AUTORE:    Esopo
TITOLO:    Le favole piu' belle
N. PAGINE: 98

stampa di tutti i libri di Italo Calvino
==================================
========== ELENCO LIBRI ==========
==================================
AUTORE:    Italo Calvino
TITOLO:    Il visconte dimezzato
N. PAGINE: 158

AUTORE:    Italo Calvino
TITOLO:    le cosmicomiche
N. PAGINE: 250

AUTORE:    Italo Calvino
TITOLO:    Il barone rampante
N. PAGINE: 135

AUTORE:    Italo Calvino
TITOLO:    Il cavaliere inesistente
N. PAGINE: 124


stampa di tutti i libri di Wilde
==================================
========== ELENCO LIBRI ==========
==================================

Press any key to continue . . .
```

Aggiungere altri metodi alla classe GestioneLibro

- Supponiamo che, data una stringa che rappresenta il nome di un autore, si voglia calcolare l'elenco dei titoli relativi a quell'autore
- Si definisce un array di stringhe in cui vengono memorizzati solo i titoli dei libri di quell'autore: titoliAutore. Di che tipo è l'array?

```
String[ ] titoliAutore
```

- Il metodo stampaElenco è ancora applicabile a questo array?
- NO, perché stampaElenco può essere solo applicato a un array di libri

Il metodo titoliAutore

```
public static String[] titoliAutore(Libro[] e, String autore){
     int i,k;       //k rappresenta la lunghezza del nuovo array
     String[] app;

     /* scorro l'array solo per calcolare il numero di libri
     che hanno l'autore richiesto e dimensionare app*/
     k=0;
     for (i=0; i<e.length; i++)
            if (e[i].getAutore().equals(autore))
                  k++;

     /* creo l'array di k componenti */
     app = new String[k];

     /* scorro nuovamente l'array e per inserie in app
            i k libri che hanno l'autore richiesto */
     k = 0;
     for (i=0; i<e.length; i++)
            if (e[i].getAutore().equals(autore)){
                  app[k] = e[i].getTitolo();
                  k++;
            }
     return app;}
```

Aggiungere altri metodi alla classe Libro

- Supponiamo di voler modificare il numero di pagine di un libro, o il nome di un autore
- Bisogna definire dei metodi d'istanza che modificano un dato oggetto Libro
- Esempio: modifica del numero di pagine

```
public void setPagine(int nuovoNum){
 this.pagine = nuovoNum;
}
```

- Esempio: modifica del nome dell'autore

```
public void setAutore(String nuovoAutore){
 this.autore = nuovoAutore;
}
```

Esercizi su array di stringhe

- Dato un array di stringhe di caratteri alfanumerici, verificare che esiste almeno una stringa nell'array che sia costituita da tutti caratteri alfabetici maiuscoli
- Classe di test. Esempi:
  - { }
    - La verifica da esito negativo
  - {"2100"}
    - La verifica da esito negativo
  - {"PIPPO"}
    - la verifica da esito positivo perché esiste la stringa "PIPPO"
  - {"Abcde00","ACCaEEbcde","ABC","pippO"}
    - la verifica da esito positivo perché esiste la stringa "ABC"
  - {"2001", "odissea", "nello", "spazio"}
    - La verifica da esito negativo

Algoritmo

- Una verifica esistenziale su un insieme di elementi (stringhe dell'array)
  - Almeno una stringa tutta di caratteri maiuscoli
- Ciascuno dei quali deve soddisfare una verifica universale
  - Data una stringa essa soddisfa la verifica se è composta da tutti caratteri maiuscoli

Il metodo tutteMaiuscole

```
public static boolean tutteMaiuscole(String s){
   int i;
   boolean tutteMaiuscole;
   char c;
```

- verifica universale: inizialmente la variabile booleana è vera finché non trovo un elelmento della sequenza che non soddisfa la proprietà richiesta

```
   tutteMaiuscole = true;
   i = 0;
   while (i<s.length() && tutteMaiuscole){
       c=s.charAt(i);
       if (!(c>='A'&& c<='Z'))
             tutteMaiuscole = false;
       i++;
   }
   return tutteMaiuscole;
}
```

La classe di test

```
public static void testTutteMaiuscole(){
     String[] a;

     a = new String[]{};
     System.out.println(almenoUnaTuttaMaiuscola(a) + " = FALSE");
     System.out.println();

     a = new String[]{"2100"};
     System.out.println(almenoUnaTuttaMaiuscola(a) + " = FALSE");
     System.out.println();

     a = new String[]{"PIPPO"};
     System.out.println(almenoUnaTuttaMaiuscola(a) + "  = TRUE");
     System.out.println();

     a = new String[]{"Abcde00","ACCaEEbcde","ABC","pippO"};
     System.out.println(almenoUnaTuttaMaiuscola(a) + "  = TRUE");
     System.out.println();

     a = new String[]{"2001", "odissea", "nello", "spazio"};
     System.out.println(almenoUnaTuttaMaiuscola(a) + " = FALSE");
     System.out.println();
}
```

Esempio d’esecuzione

```
false = FALSE

false = FALSE

true = TRUE

true = TRUE

false = FALSE

Press any key to continue . . .
```

…Schema generale della classe

```
class TutteMaiuscole{

   public static void main(String[] args){
        testTuttiMaiuscoli();
   }

   public static void testTutteMaiuscole(){
        …
        System.out.println(almenoUnaTuttaMaiuscola(a) + " = FALSE");
        …
   }

   public static boolean almenoUnaTuttaMaiuscola(String[] a){
   }

   public static boolean tutteMaiuscole(String s){
   }
}
```

Esercizio

- Dato un array di stringhe di caratteri alfanumerici, verificare che esiste almeno una stringa nell'array che contiene un'alternanza di lettere (maiuscole o minuscole) e cifre
- Esempio: a7B8d9c0, 4o5k6w
- Classe di test. Esempi:
  - { }
    - La verifica da esito negativo
  - {"2"}
    - La verifica da esito negativo
  - {"a"}
    - La verifica da esito negativo
  - {"P1"}
    - La verifica da esito positivo
  - {"a7B8d9c0"}
    - La verifica da esito positivo
  - {"A1B21C"}
    - La verifica esito negativo
  - {"4o5k6w"}
    - La verifica esito positivo
  - {"2","a","A1B2C","pippo"}
    - La verifica esito positivo
  - {"2","a","A1BeC","pippo"}
    - La verifica esito negativo

Schema generale della classe…

- Lo schema generale della classe AlternanzaLettereCifre è simile al precedente:
- Cambia il metodo per la verifica della proprietà, oltre ai test specifici di correttezza
- La proprietà, che è vera se nella stringa sono presenti in modo alternato lettere e cifre, che tipo di verifica comporta?
- Il metodo per la verifica esistenziale (sull'array) rimane invariato, a parte la chiamata del metodo specifico

Il metodo per la verifica dell'alternanza di lettere e cifre

- **Algoritmo:**
- **Una stringa di lunghezza 0 o 1 non presenta alternanze, quindi il metodo restituisce false**
- **Negli altri casi bisogna distingure se si parte con una cifra o con una lettera (maiuscola o minuscola)**
  - **Se il primo carattere è una lettera allora**
    - **Tutte le lettere si devono trovare in posizione d'indice pari nella stringa e tutte le cifre in posizione d'indice dispari**
  - **Se il primo carattere è una cifra allora**
    - **Tutte le lettere si devono trovare in posizione d'indice dispari nella stringa e tutte le cifre in posizione d'indice pari**
- **Inizialmente la variabile booleana è true e non appena una delle precedenti condizioni non è verificata diventa false e l'istruzione ripetitiva termina**

Il metodo alternanza…

- **Casi particolari e inizializzazioni**

```
public static boolean alternanza(String s){
int i;
boolean verifica;
char c, primo;

/* caso particolare lunghezza 0 o 1 */
if (s.length() == 0 || s.length() == 1)
    verifica = false;
    else {                          //caso generale
    /* verifica universale:
    inizialmente la variabile booleana e' true */
    verifica = true;
    primo = s.charAt(0);
```

…Il metodo alternanza…

- **Se il primo carattere è una lettera:**

```
/*se il primo carattere e' una lettera */
   if ((primo>='a' && primo<='z') ||
                (primo>='A' && primo<='Z')){
        i=1;
        /* tutti gli indici pari devono essere
        lettere e tutti i dispari cifre */
        while (i<s.length() && verifica){
                c = s.charAt(i);
                if (i%2 == 0 && !( (c>='a' && c<='z') ||
                                (c>='A' && c<='Z')))
                        verifica = false;
                if (i%2 == 1 && !(c>='0' && c<='9'))
                        verifica = false;
                i++;
        }
   }
```

…Il metodo alternanza…

- **Se il primo carattere è una cifra**

```
   else /* il primo carattere e' una cifra */
        if ((primo>='0' && primo<='9'))      {
                i=1;
                /* tutti gli indici pari devono essere
                cifre e tutti i dispari lettere */
                while (i<s.length() && verifica){
                        c = s.charAt(i);
                        if (i%2 ==1 && !( (c>='a' && c<='z') ||
                                          (c>='A' && c<='Z')))
                                verifica = false;
                        if (i%2 ==0 && !(c>='0' && c<='9'))
                                verifica = false;
                        i++;
                }
        }
   }    //chiude il primo else
   return verifica;
}
```

Il metodo unaAlternanza

Verifica esistenziale

```
public static boolean unaAlternanza(String[] a){
   int i;
   boolean trovata; //true se ho trovato una stringa con
   lettere e cifre

   /* verifica esistenziale: inizialmente la variabile
   booleana e' falsa: non appena trovo l'elemento che soddisfa
   la verifica, la varabile booleana diventa vera */

   trovata=false;
   i = 0;
   while (i<a.length && !trovata){
       if(alternanza(a[i]))
              trovata = true;
       }
   return trovata;
}
```

Il metodo di test

```
public static void testAlternanza(){
      String[] a;
      a = new String[]{};
      System.out.println(unaAlternanza(a) + " = FALSE");
      System.out.println();
      a = new String[]{"2"};
      System.out.println(unaAlternanza(a) + " = FALSE");
      System.out.println();
      a = new String[]{"a"};
      System.out.println(unaAlternanza(a) + " = FALSE");
      System.out.println();
      a = new String[]{"P1"};
      System.out.println(unaAlternanza(a) + "  = TRUE");
      System.out.println();
      a = new String[]{"a7B8d9c0"};
      System.out.println(unaAlternanza(a) + "  = TRUE");
      System.out.println();
      a = new String[]{"A1B21C"};
      System.out.println(unaAlternanza(a) + "  = FALSE");
      System.out.println();
      a = new String[]{"4o5k6w"};
      System.out.println(unaAlternanza(a) + "  = TRUE");
      System.out.println();
      a = new String[]{"2","a","A1B2C","pippo"} ;
      System.out.println(unaAlternanza(a) + "  = TRUE");
      System.out.println();
      a = new String[]{"2","a","A1BeC","pippo"} ;
      System.out.println(unaAlternanza(a) + "  = FALSE");
      System.out.println();}
```

Esempio d’esecuzione

```
false = FALSE
false = FALSE
false = FALSE
true  = TRUE
true  = TRUE
false  = FALSE
true  = TRUE
true  = TRUE
false  = FALSE
Press any key to continue . . .
```