String, Array e Main

Lezione VII

Scopo della lezione

- Presentare la classe String ed il tipo stringa;
- presentare ed imparare a usare gli array in Java;
- approfondire la conoscenza con il metodo speciale `main`.

String

- Il tipo `String` contiene una stringa **costante** di caratteri;
- una stringa si denota delimitando i relativi caratteri con dei doppi apici, es `“Walter”;`
- usando una variabile `String` come argomento di `println` o `print`, la corrispondente stringa viene visualizzata.

Sequenze di escape

- Alcuni particolari elementi di una stringa non possono essere espressi tramite un carattere, come l'"a capo";
- per inserire questi elementi è possibile usare delle descrizioni mnemoniche dette **sequenze di escape**;
- le sequenze di escape sono costituite dal carattere di backslash (\) seguito da un particolare carattere;

Sequenze di escape

Carattere	Significato
`\n`	A capo (newline)
`\t`	Tabulazione
`\"`	Doppio apice
`\\`	Backslash

Concatenamento

- L'operatore di **concatemento** (+) giustappone una stringa a un'altra stringa
  - `"Ciao " + "mondo" = "Ciao mondo"`
- l'esito dell'operazione consiste nella creazione di una nuova stringa;

Esercizio

- Scrivere un programma Java che
  - Utilizzi il concatenamento di stringhe
  - Stampi i caratteri di newline, tabulazione, doppio apice, apice e backslash

Concatenamento

- I tipi primitivi sono implicitamente **promossi** a stringa quando usati in combinazione con l'operatore di concatenazione
  - `"La radice quadrata di " + 5`

Esercizio

- Ilseguente codice ha come esito la stampa di un numero

```
char c = ‘a’, d = ‘b’;
System.out.println(a+c);
```

- Perché?
- Verificare che invece il seguente codice stampa i contenuti di c e d

```
char c = ‘a’, d = ‘b’;
System.out.println(“”+a+c);
```

- Che cosa è “”?

String è una classe

- A differenza dei tipi di dati finora visti, String rappresenta una classe, ovvero un insieme di oggetti aventi caratteristiche comuni
- Gli oggetti, in questo caso, sono appunto le stringhe di caratteri. Per ora li chiameremo variabili di tipo String
- Più avanti studieremo in dettaglio le classi e gli oggetti in Java

String è una classe

- Le variabili di tipo String si dichiarano esattamente come le variabili di altri tipi
- Per fare riferimento ad un oggetto della classe string è possibile
  - Scrivere una stringa costante (cioè delimitata da doppi apici)
  - Utilizzare l'operatore new per creare una nuova stringa

Esempio


```
import prog.io.*;
class NuoveStringhe {
  public static void main(String args[]) {
    ConsoleOutputManager video
      = new ConsoleOutputManager();
    String messaggio = “io sono una stringa”;
    String nuovoMess = new String(“anche io”);
    video.print(messaggio);
    video.print(nuovoMess);
  }
}
```

`String` è una classe

- Le istanze di `String` incorporano una o più funzioni che operano su di esse (i.e. sulle stringhe che rappresentano);
- Per eseguire una funzione si scrive
  - il nome della variabile di tipo `String`,
  - il carattere di punto (`.`),
  - il nome della funzione seguito da parentesi tonde, contenenti eventuali argomenti separati da virgole.

Lunghezza di una stringa

- La funzione `length()` ritorna il numero di caratteri contenuti in una stringa;
- questa funzione non necessita di parametri.

Esempio

```
import prog.io.*;
class Stringa {
  public static void main(String args[]) {
    ConsoleOutputManager video
      = new ConsoleOutputManager();
    String messaggio;
    messaggio = “The quick brown fox ” +
      “jumps over the lazy dog”;
    video.println(messaggio);
    video.println(messaggio.length());
  }
}
malchiod% java Stringa
The quick browk fox jumps over the lazy dog
43
```

Esempio

- Length, come le altre funzioni, può essere calcolata sia su una variabile che contiene un oggetto della classe String che su una stringa costante
- Verificate che è possibile scrivere un programma Java in cui compare il seguente frammento di codice

```
“messaggio”.length()
```

Estrazione di caratteri

- La funzione `charAt()` ritorna il carattere contenuto in una precisa posizione della stringa;
- è necessario passare il numero di posizione come argomento della funzione;
- la prima posizione ha come numero `0`, la seconda `1` e così via.

Esempio

```
// Questo programma verra' compilato, ma...
import prog.io.*;
class PosizioneSbagliata {
  public static void main(String args[]) {
    ConsoleOutputManager video
      = new ConsoleOutputManager();
    String mess;
    mess = "The quick brown fox " +
      "jumps over the lazy dog";
    video.println(mess.charAt(0));
    video.println(mess.charAt(mess.length()));
  }
}
```

Esempio

- Verificare che il programma alla slide precedente viene correttamente compilato, ma durante la sua esecuzione si verifica un errore
- Analizzare il programma e spiegare perché si è verificato questo errore

Leggere una stringa

- La funzione readLine() della classe ConsoleInputManager legge una stringa da tastiera e ritorna il corrispondente oggetto della classe String

Sfida!

- Scrivere un programma che legga una stringa da tastiera e stabilisca se questa è o meno palindroma (cioè se non cambia qualora la si legga dall'ultimo al primo carattere)

Sottostringhe

- Si dice **sottostringa** una stringa che è parte di un'altra stringa;
- la funzione `substring()` ritorna una sottostringa di una stringa;
- è necessario passare come argomenti alla funzione:
  - la posizione del primo carattere della sottostringa;
  - la posizione che segue l'ultimo carattere.

Esempio

```
import prog.io.*;
class Sottostringa {
  public static void main(String args[]) {
    ConsoleOutputManager video
      = new ConsoleOutputManager();
    String mess;
    mess = “The quick brown fox ” +
      “jumps over the lazy dog”;
    video.println(mess.substring(4,15));
  }
}
malchiod% java Sottostringa
quick brown
```

Ricerca di sottostringhe

- La funzione `indexOf()` ritorna la posizione della prima occorrenza di una data sottostringa in una stringa;
- è necessario passare come argomento della funzione la sottostringa che si vuole ricercare;
- quando la ricerca ha esito negativo la funzione ritorna `-1`.

Ricerca di sottostringhe

- Una seconda versione di `indexOf()` effettua la ricerca a partire da una fissata posizione della stringa;
- è necessario passare come argomenti della funzione
  - la sottostringa che si vuole ricercare;
  - la posizione a partire da cui effettuare la ricerca.
- Se l'esito è negativo la funzione ritorna -1

Esempio

```
import prog.io.*;
class Ricerca {
  public static void main(String args[]) {
    ConsoleOutputManager video
      = new ConsoleOutputManager();
    String messaggio
      = “The quick brown fox jumps over the lazy dog”;
    video.print(messaggio.indexOf(“the”));
    video.print(messaggio.indexOf(“he”));
    video.print(messaggio.indexOf(“he”,5));
    video.println(messaggio.indexOf(“Thi”));
  }
}
malchiod% java Ricerca
31 1 32 -1
```

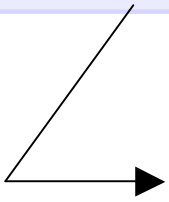
Sfida!

- Scrivere un programma che legga da tastiera due stringhe e stampi il numero totale di occorrenze della seconda stringa all'interno della prima
- Se ad esempio la prima stringa è "the quick brown fox jumps over the lazy dog" e la seconda stringa è "the", l'output del programma dovrà essere: 2

Confronto tra stringhe

- Le stringhe possono essere confrontate in base al loro ordine **lessicografico**. Caso speciale: la stringa vuota precede tutte le altre

```
public boolean equals(Object anObject);
// ridefinisce Object.equals()
public boolean equalsIgnoreCase(String  anotherString)
public int compareTo(String  anotherString)
```



>0 se l'argomento è minore dell'oggetto
=0 se l'argomento è uguale dell'oggetto
<0 se l'argomento è maggiore dell'oggetto

Confronto tra stringhe

- Due stringhe sono uguali se hanno le stesse lettere nello stesso ordine

```
String s1 = "hello“, s2 = "Hello";
s1.equals(s2)                 // false
s1.equals("hello”);           // true
```

- **Attenzione!** `s1==s2` controlla se i due oggetti sono identici e non se contengono la stessa stringa

Confronto di stringhe

```
String s1 = new String("hello");
String s2 = new String("hello");
String s3 = new String("Hello");
String s4 = s1;   // s1 ==  s4
String s5 = "hello";
String s6 = "hello";
```

s5 e s6 puntano allo stesso (identico) oggetto costante

Test di uguaglianza

```
s1.equals(s2) ➔ true
s1.equals(s3) ➔ false
s1.equals(s4) ➔ true
s1.equals(s5) ➔ true
s5.equals(s6) ➔ true
```

Test di identità

```
s1 == s2 ➔ false
s1 == s3 ➔ false
s1 == s4 ➔ true
s1 == s5 ➔ false
s5 == s6 ➔ true
```

Esempio

- Verificare, scrivendo un opportuno programma Java, che (tutte) le espressioni booleane descritte alla slide precedente assumano il valore indicato

Esercizio

- Scrivere un programma che legga due stringhe da tastiera e dica quale delle due è maggiore, sulla base del valore ritornato dalla funzione compareTo

StringBuffer

- Le stringhe Java sono **immutabili**. Se si assegna un nuovo valore a una stringa, Java **DEVE** creare un nuovo oggetto di tipo stringa e distruggere quello vecchio:

```
resultStr = resultStr + " " + ptr;
```

  Java creerà un nuovo oggetto che verrà puntato da resultStr;
- gli oggetti `StringBuffer` rappresentano stringhe modificabili.

ReverseString

- Scrivere una classe `ReverseString` che inverte una stringa introdotta da tastiera:
  - usando `StringBuffer`
  - non utlizzando la funzione `reverse`.
- Guardate la soluzione alla slide successiva solo DOPO aver risolto l'esercizio per conto vostro

Esempio

```
import prog.io.*;
class ReverseString {
 public static void main(String args[]) {
  ConsoleOutputManager video=new ConsoleOutputManager();
  ConsoleInputManager tasti=new ConsoleInputManager();
  char aux;
  StringBuffer msg = new StringBuffer(tasti.readLine(
   "Stringa da rovesciare: "));
  for(int i=0;i<=(msg.length()-1)/2;i++) {
   aux = msg.charAt(i);
   msg.setCharAt(i,msg.charAt(msg.length()-1-i));
   msg.setCharAt(msg.length()-1-i,aux);
  }
  video.println("Stringa rovesciata " + msg);
  }
}
```

Esempio

```
import prog.io.*;  // java.lang.* è incluso automaticam.
class ReverseString {
 public static void main(String args[]) {
  ConsoleOutputManager video=new ConsoleOutputManager();
  ConsoleInputManager tasti=new ConsoleInputManager();
  char aux;
  StringBuffer msg = new StringBuffer(tasti.readLine(
   "Stringa da rovesciare: "));
  for(int i=0;i<=(msg.length()-1)/2;i++) {
   aux = msg.charAt(i); // variabile di appoggio
   msg.setCharAt(i,msg.charAt(msg.length()-1-i));
   msg.setCharAt(msg.length()-1-i,aux); // scambio
  }
  video.println("Stringa rovesciata " + msg);
  }
}
```

Esempio

```
malchiod% java ReverseString
Stringa da rovesciare: dario
Stringa rovesciata oirad
```

Esercizio

- Scrivere la classe `Bizarre` che letta una stringa in input ne converta in loco (cioè senza creare un'altra stringa e senza stamapre direttamente il risultato) il case (cioè cambi ogni carattere maiuscolo in minuscolo e ogni carattere minuscolo in maiuscolo). Stampare alla fine, cioè dopo la conversione, la stringa convertita (la slide successiva contiene alcuni suggerimenti).

Suggerimenti

- Usare variabili di tipo `StringBuffer`.
- Consultare le API (http://java.sun.com, seguire il link APIs, poi il link J2SE1.5.0) relativamente alle funzioni
  - isUpperCase
  - isLowerCase
  - toUpperCase
  - toLowerCase

  della classe Character

Esercizio

- Scrivere la classe `ConcatenaStringhe` che lette due stringhe ne crea una terza contenente la loro concatenazione, il tutto usando solo i metodi `charAt` e `setCharAt()` della classe `StringBuffer`. Costruita la terza stringa stamparla a video. **Suggerimenti**: costruire la stringa risultato come una istanza della classe `StringBuffer` di dimensione la somma delle lunghezze delle due stringhe passate in input. Usare quindi `charAt()` e `setCharAt()` per copiare le stringhe nella stringa risultato.

Array

- Un array è una collezione di locazioni di memoria contigue contenenti dati dello stesso tipo
  - gli elementi degli array sono acceduti tramite la loro posizione nell'array piuttosto che usandone il nome;
  - gli n elementi di un array `a` sono riferiti come `a[0], a[1], a[2], ..., a[n-1]`

Array

- Gli array sono trattati come oggetti
  - vengono istanziati dall'operatore `new`,
  - hanno variabili di istanza (es. `length`),
  - le variabili di tipo array sono dei riferimenti
  - un riferimento ad un array è passato come un parametro
- Ma non c'è nessuna classe. Gli array non fanno parte della gerarchia delle classi di Java, come i tipi primitivi.

Terminologia

- Un array vuoto contiene zero valori;
- la lunghezza di un array è il numero di elementi che lo compongono;
- ogni componente di un array ha lo stesso tipo;
- gli elementi di un array possono essere di un tipo qualsiasi.

Dichiarazione e creazione

- Nel creare un array dobbiamo specificare sia il tipo degli elementi che la lunghezza

```
int arr[];     // Dichiara una variabile di tipo array
arr = new int[15];   // Crea l'array stesso
```

- Combinando i due passi:

```
int arr[] = new int[15];
```

Il nome dell'array è arr.

l'array può contenere 15 valori interi.

Inizializzazione di array

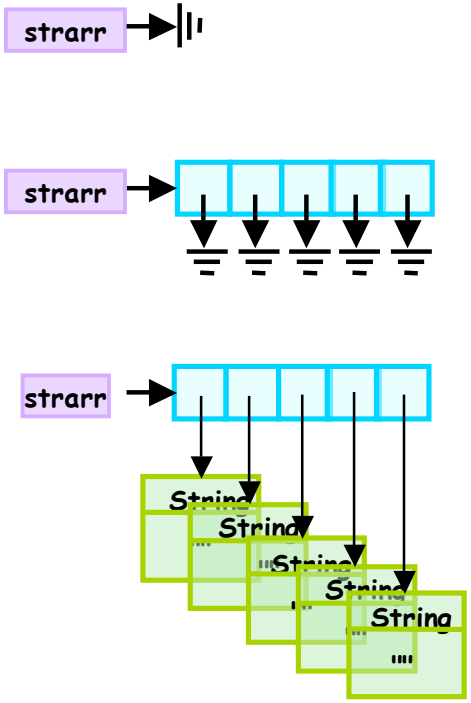
- Gli elementi di un array hanno un valore di default con cui vengono inizializzati:
  - `\0` per caratteri (NULL), interi e reali,
  - `false` per i booleani,
  - `null` per gli oggetti
- In alternativa, gli array possono essere inizializzati alla creazione

```
 int arr[]={-2,8,-1,-3,16,20,25,16,16,8,18,19,45,21,-2
};
 String strings[] = {“ciao", “mondo", “Walter"};
```

Esercizio

- Scrivere un programma che dichiara, senza inizializzarlo, un array
- Verificare che gli elementi dell'array vengono inizializzati automaticamente

Esempio: array di stringhe




```
(a) String strarr[];
    // array che punta a null

(b) strarr = new String[5];
    // crea un array di riferimenti
    // a stringhe nulle

(c) for (int k = 0; k < strarr.length; k++)
        strarr[k] = new String();
    /* Crea 5 stringhe e le assegna
       all'array */
```

Uso degli elementi

- Le variabili di tipo array, una volta indiciate, sono usate come altre variabili

```
arr[0] = 5;
arr[2] = 3;
strings[0] = “ciao";
strings[1] = strings[2] = “mondo";
```

- Si possono usare cicli for per scorrere gli array

```
for (int k = 0; k < arr.length; k++)
        arr[k] = (k+1) * (k+1);
```

length è una variabile di istanza, non un metodo.

Esercizio

- Scrivere un programma che
  - Richieda all'utente di immettere un numero intero n
  - Richieda all'utente di immettere, successivamente, n valori interi
  - Memorizzi questi valori in un array
  - Calcoli e stampi il massimo, il minimo, la somma e la media dei valori contenuti nell'array

Esercizio

- Scrivere un programma che
  - Legge da tastiera una stringa
  - Memorizza i suoi caratteri all'interno di un array
  - Crea un secondo array che contiene i caratteri in ordine inverso, dall'ultimo al primo, copiandoli dal primo array
- Provate a risolvere il problema usando un unico array

Ordinamento di array

- Bubble sort: ad ogni scansione dell'array, l'elemento più grande non ancora ordinata "galleggia" verso "l'alto".
- Per ordinare un array di N elementi sono necessarie N-1 passate

```
1. per ognuna delle N-1 scansioni dell'intero array
2.    per ognuna delle N-1 coppie di elementi adiacenti nell'array
3.       se l'elem. di indice minore è più grande di quello con indice maggiore
4.        scambia i due elementi
```

Array: ordinamento


```
 21  20  27  24  19    // array non ordinato
[20  21] 27  24  19    // confronta gli elementi adiacenti
 20  21 [24  27] 19    // li scambia, quando necessario
 20  21  24 [19 |27]   // 1ª passata, 27 galleggia verso
l'alto
 20  21  19 |24  27    // 2ª passata, 24 galleggia verso
l'alto
 20  19 |21  24  27    // 3ª passata, 21 galleggia verso
l'alto
 19 |20  21  24  27    // 4ª passata, l'array è ordinato
```

Esercizio

- Implementate l'algoritmo di bubble sort:
  - Inizializzate un array a valori scelti da voi
  - Ordinate l'array
  - Stampate l'array ordinato
- Le slide seguenti contengono la soluzione, ma voi guardatela solo DOPO aver risolto l'esercizio, e confrontate

Esempio


```
import prog.io.*;
class BubbleSort {
 public static void main(String args[]) {
  ConsoleOutputManager video=new
   ConsoleOutputManager();
  int arr[]={21,20,27,24,19},temp; // inizializza
  for (int i=0;i<arr.length;i++)
    video.print(arr[i] + " "); // stampa l'array
  video.println();     // prima che sia ordinato
```

Esempio


```
  for (int pass=1;pass<arr.length;pass++)
    for (int pair=1;pair<arr.length;pair++)
      if(arr[pair-1]>arr[pair]) {
        temp = arr[pair-1];
        arr[pair-1] = arr[pair];
        arr[pair] = temp;
      }
  for (int i=0;i<arr.length;i++)
    video.print(arr[i] + " "); // stampa l'array
  video.println();     // dopo l'ordinamento
  }
}
```

Esecuzione

```
malchiod% java BubbleSort
21 20 27 24 19
19 20 21 24 27
malchiod%
```

Ricerca sequenziale

- Problema: cercare un valore in un array.
- Se l'array non è ordinato, il valore va cercato sequenzialmente controllando ogni elemento dell'array

Esercizio

- Scrivete un programma che implementi l'algoritmo di ricerca sequenziale:
  - Inizializzate un array a dei valori fissi, scelti da voi
  - Richiedete all'utente di immettere un valore
  - Verificate se tale valore è presente o meno nell'array e comunicatelo all'utente
- A problema risolto, confrontate il codice con quello alla slide successiva

Esempio

```
import prog.io.*;
class SequentialSearch {
 public static void main(String args[]) {
  ConsoleOutputManager video=new ConsoleOutputManager();
  ConsoleInputManager tasti=new ConsoleInputManager();
  int arr[]={21,20,27,24,19}; // inizializza
  int k=0, key, temp;
  key = tasti.readInt("Chiave da cercare nell'array ");
  while((k<arr.length)&&(arr[k]!=key)) ++k;
  if(k<arr.length)
    video.println(key + " trovato al posto " + k);
  else video.println(key + " non trovato");
 }
}
```

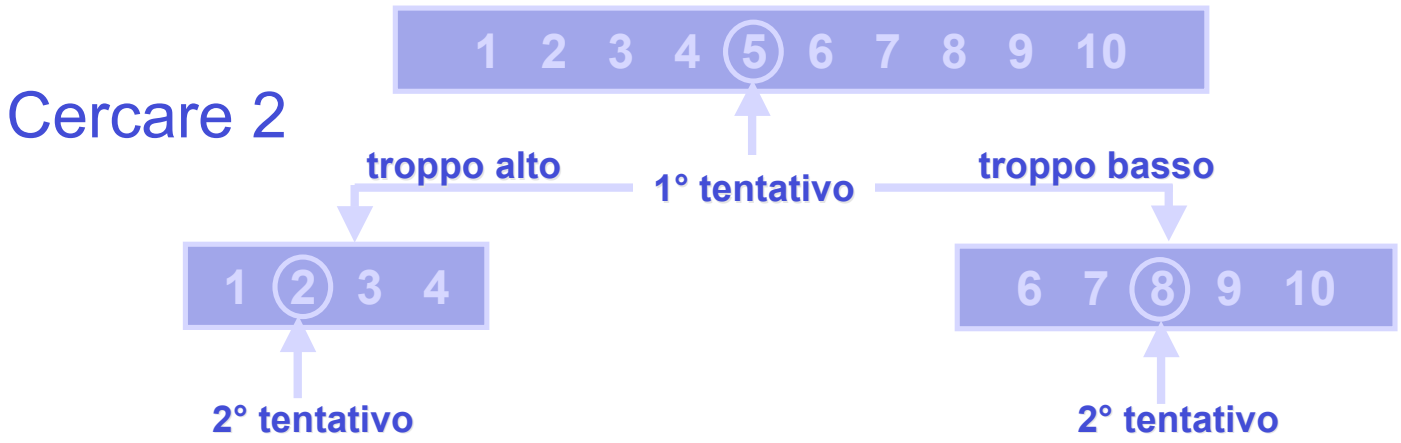
La ricerca fallisce se si arriva alla fine dell'array senza aver trovato il valore.

Esecuzione

```
malchiod% java SequentialSearch
Chiave da cercare nell'array 19
19 trovato al posto 4
malchiod% java SequentialSearch
Chiave da cercare nell'array 25
25 non trovato
malchiod% java SequentialSearch
Chiave da cercare nell'array 27
27 trovato al posto 2
malchiod%
```

Ricerca binaria

- Usa una strategia di tipo dividi e conquista in un array **ordinato**
- Ad ogni iterazione si divide l'array a metà, restringendo la ricerca alla metà che potrebbe contenere il valore cercato



Esercizio

- Scrivete un programma che
  - Legge una successione di valori interi e li memorizza in un array
  - Legge un valore intero e lo cerca all'interno dell'array, comunicando all'utente se l'elemento è stato trovato oppure no

  Ordinando l'array dopo che è stato creato e applicando la strategia di ricerca binaria (vedere la slide successiva)

Suggerimenti

- Memorizzare in due variabili intere le posizioni che delimitano la porzione di array che si sta analizzando:
  - Inizialmente queste variabili contengono 0 e n (la lunghezza dell'array)
  - Al primo passo della ricerca conterranno 0 e n/2 oppure n/2 e n, e così via
- A problema risolto confrontate la vostra soluzione con quella alla slide seguente

Esempio

```
import prog.io.*;
class BinarySearch {
 public static void main(String args[]) {
  ConsoleOutputManager video=new ConsoleOutputManager();
  ConsoleInputManager tastiera=new ConsoleInputManager();
  int key, arr[]={21,20,27,24,19};  // inizializza
  int low=0, high=arr.length-1;//porzione da usare
  key = tastiera.readInt("Chiave da cercare ");
```

Esempio


```
while(low <= high) {// finché non siamo in fondo
  int mid = (low+high)/2;          //spezza l'array
  if(arr[mid]==key) {
    video.println(key + " trovato al posto " + mid);
    break;
  }
  else if(arr[mid]<key)
         low = mid+1;          //cerca nella parte alta
       else high = mid -1; //cerca nella parte bassa
}
if(low > high)
   video.println(key + " non trovato");
 }
}
```

Esecuzione

```
malchiod% java BinarySearch
Valore da cercare 19
19 trovato al posto 0
malchiod% java BinarySearch
Valore da cercare 24
24 trovato al posto 3
malchiod% java BinarySearch
Valore da cercare 3
3 non trovato
malchiod%
```

Matrici

- Gli **array multidimensionali** o **matrici** sono array i cui componenti sono a loro volta array e così via.

Esempio: il calendario

```
double calendar[][] = new double[12][31];
```

- `calendar[7]` è agosto
- Il primo array rappresenta 12 mesi dell'anno. Ogni mese è rappresentato da un array di 31 giorni.

```
char c[][] = { {'a', 'b'}, {'c', 'd'} } ;
double d[][][] = { {1.0, 2.0, 3.0}, {4.0, 5.0}, {6.0,
   7.0, 8.0, 9.0} } ;
```

- Ogni dimensione di una matrice può avere una lunghezza diversa

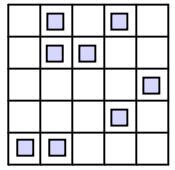
Battaglia navale

- Torniamo bambini e implementiamo la classe BattagliaNavale per il relativo gioco.
- Caratteristiche
  - campo di battaglia 5x5 predefinito;
  - controllo che le righe e le colonne introdotte da tastiera siano ammissibili;
  - finché manco le navi continuo a tentare.

Esempio

```
import prog.io.*;
class BattagliaNavale {
 public static void main(String args[]) {
  ConsoleOutputManager video=new ConsoleOutputManager();
  ConsoleInputManager tastiera=new ConsoleInputManager();
  char campo[][] = new char[5][5];
  //campo automaticamente inizializzato a '\0'

  int x,y;
  //introduco le navi sul campo di battaglia
  campo[0][1]='X'; campo[1][2]='X';
  campo[3][3]='X'; campo[4][0]='X';
  campo[1][1]='X'; campo[2][4]='X';
  campo[0][3]='X'; campo[4][1]='X';
```



Esempio


```
//mostro il campo
for(int i=0;i<5;i++) {
 video.print("[");
 for(int j=0;j<5;j++)
  video.print(((campo[i][j]=='\0')?
    '.':campo[i][j]) + " ");
 video.println("]");
}
do {
 do { //gestione introduzione coordinate
  x = tastiera.readInt("Introduci riga, da 0 a 4 ");
 } while ((x<0)||(x>4));
 do {
  y = tastiera.readInt("Introduci colonna, da 0 a 4 ");
 } while ((y<0)||(y>4));
```

Esempio

```
    if(campo[x][y]!='X')
     video.println("Acqua, riprova!");
   } while (campo[x][y]!='X');
   // gioco finche' non affondo qualcosa
   video.println("Nave colpita in ("+x+","+y+")");
  }
}
```

Esecuzione

```
java BattagliaNavale
[. X . X . ]
[. X X . . ]
[. . . . X ]
[. . . X . ]
[X X . . . ]
Introduci riga, da 0 a 4 0
Introduci colonna, da 0 a 4 5
Introduci colonna, da 0 a 4 0
Acqua, riprova!
Introduci riga, da 0 a 4 2
Introduci colonna, da 0 a 4 0
Acqua, riprova!
Introduci riga, da 0 a 4 0
Introduci colonna, da 0 a 4 1
Nave colpita in (0,1)
```

Esercizio

- Implementate il programma BattagliaNavale

`main`: questo sconosciuto

- Ogni programma deve specificare un punto da cui inizierà la propria esecuzione.
- La definizione del metodo `main` è uno dei modi che offre Java per informare la virtual machine del punto di inizio del programma.

`public static void main(String args[])`

- Esaminiamo la definizioine
  - main non ritorna alcun valore (void);
  - il metodo main viene attivato prima di tutti gli altri, compresi i metodi per la creazione di oggetti, e quindi non può e non deve far riferimento a istanze per poter essere attivato: deve essere un metodo statico;
  - non deve avere limiti di accesso (public)
  - è l'interfaccia del programma con l'esterno da cui prende degli argomenti (args).

Argomenti di main

- Ora sappiamo interpretare il significato del codice scritto tra parentesi dopo main
- Indica un array di stringhe dal nome args
- Se quando viene eseguito un programma Java vengono specificati dei valori addizionali dopo il nome della classe, questi vengono memorizzati nell'array args

Argomenti di main

- Se esiste una classe Argomenti e viene eseguita dal comando

```
Java Argomenti qui quo qua 8
```

- All'interno del metodo main sarà possibile fare riferimento all'array args, che conterrà le stringhe "qui", "quo", "qua" e "8" (NON il numero 8) in posizione 0, 1, 2 e 3, rispettivamente.

Esempio

```
import prog.io.*;
class ArgsFromCommandLine {
 public static void main(String args[]) {
  ConsoleOutputManager video=new ConsoleOutputManager();
  video.println("Il programma e' stato eseguito con "
   + args.length + " argomenti");

  if(args.length>0)
   for(int i=0;i<args.length;i++)
    video.println("Argomento " +i+ ": " +args[i]);
 }
}
```

Esecuzione


```
malchiod% java ArgsFromCommandLine
Il programma e' stato eseguito con 0 argomenti
malchiod% java ArgsFromCommandLine Qui Quo Qua 7
-0.7 e Paperino
Il programma e' stato eseguito con 7 argomenti
Argomento 0: Qui
Argomento 1: Quo
Argomento 2: Qua
Argomento 3: 7
Argomento 4: -0.7
Argomento 5: e
Argomento 6: Paperino
```

Esercizio

- Implementate la classe ArgsFromCommandLine