UNIVERSITÀ
DEGLI STUDI
DI TRIESTE

Fondamenti di Informatica (117IN)

A.A. 2022 / 2023

Lezione 8 - Funzioni

Sylvio Barbon Junior
sylvio.barbonjunior@units.it

Sommario:

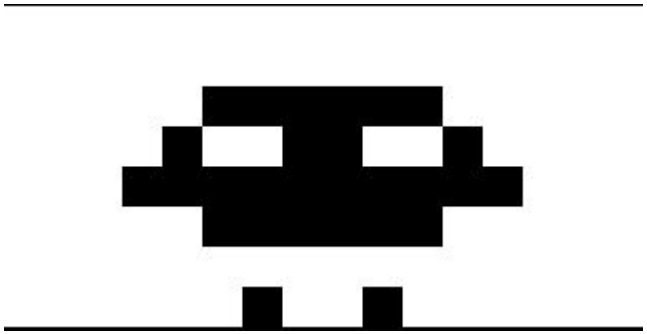
1) Funzioni
   a) Struttura
   b) Visibilità e Scope
   c) Overloading, Ricorsione

Sylvio Barbon Junior - sylvio.barbonjunior@units.it

L'ultima lezione…

```java
import java.io.File;
import java.awt.image.BufferedImage;
import javax.imageio.ImageIO;

public class Image{
    public static void main(String[] args)  throws Exception {
      File file= new File("Alien.png");
      BufferedImage img = ImageIO.read(file);
      System.out.println("Image (height):"+img.getHeight());
      System.out.println("Image (width):"+img.getWidth());
      for (int y = 0; y < img.getHeight(); y++) {
        System.out.println("");
         for (int x = 0; x < img.getWidth(); x++) {
            int pixel = img.getRGB(x,y);
            if(pixel==-1)
                System.out.print("B");
            else
                System.out.print("N");
         }
      }
      System.out.println("");
    }
}
```

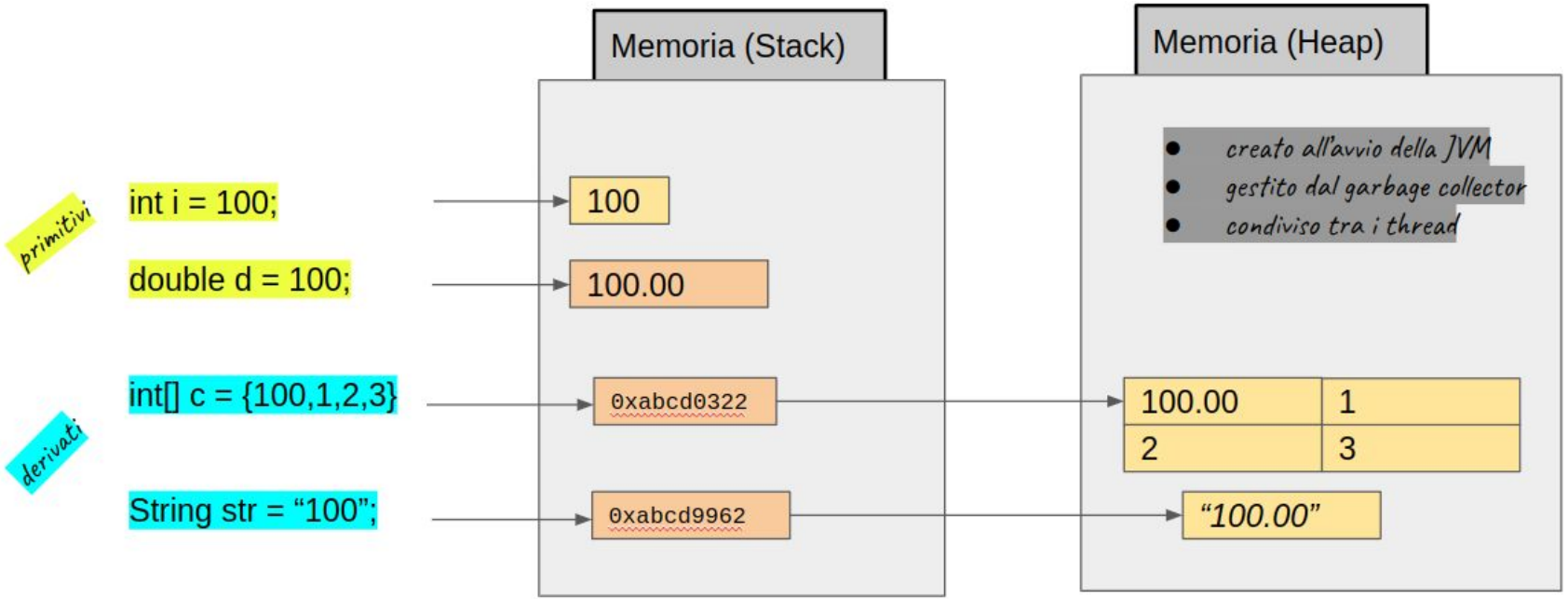


```
barbon@xps:~/Downloads/FI_2023/java$ java Image
Image (height):8
Image (width):16

BBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBB
BBBBBNNNNNNBBBBB
BBBBNBBNNBBNBBBB
BBBNNNNNNNNNNBBB
BBBBBNNNNNNBBBBB
BBBBBBBBBBBBBBBB
BBBBBBNBBNBBBBBB
barbon@xps:~/Downloads/FI_2023/java$
```


Sylvio Barbon Junior - sylvio.barbonjunior@units.it

Lezione precedente:



Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni
   - La classe principale di un programma è la funzione main();
   - Possiamo creare le nostre funzione che saranno richiamata dalla main() o altre classi;
   - Nella definizione di una funzione occorre indicare:
     i) tipo;
     ii) il tipo del risultato;
     iii) nome;
     iv) gli eventuali parametri;
     v) specificare quali azioni deve eseguire;

```
public static void main(String[] args) {
        System.out.println("Ciao!");
}
```

*Il metodo Java main() è sempre **statico**, in modo che il compilatore possa chiamarlo **senza la creazione di un oggetto** o prima della creazione di un oggetto della classe!!!*

Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni - struttura
   - Il corpo di una funzione (essendo un blocco) **non può contenere altre definizioni di funzione**, ma solo definizioni di:
     i) variabili;
     ii) classi;
   - Una funzione può restituire come valore un risultato, oppure non avere nessun risultato (parola chiave ***void***);
   - Il corpo deve prevedere almeno una istruzione ***return***, con la specifica dell'espressione il cui valore rappresenta il risultato stesso;

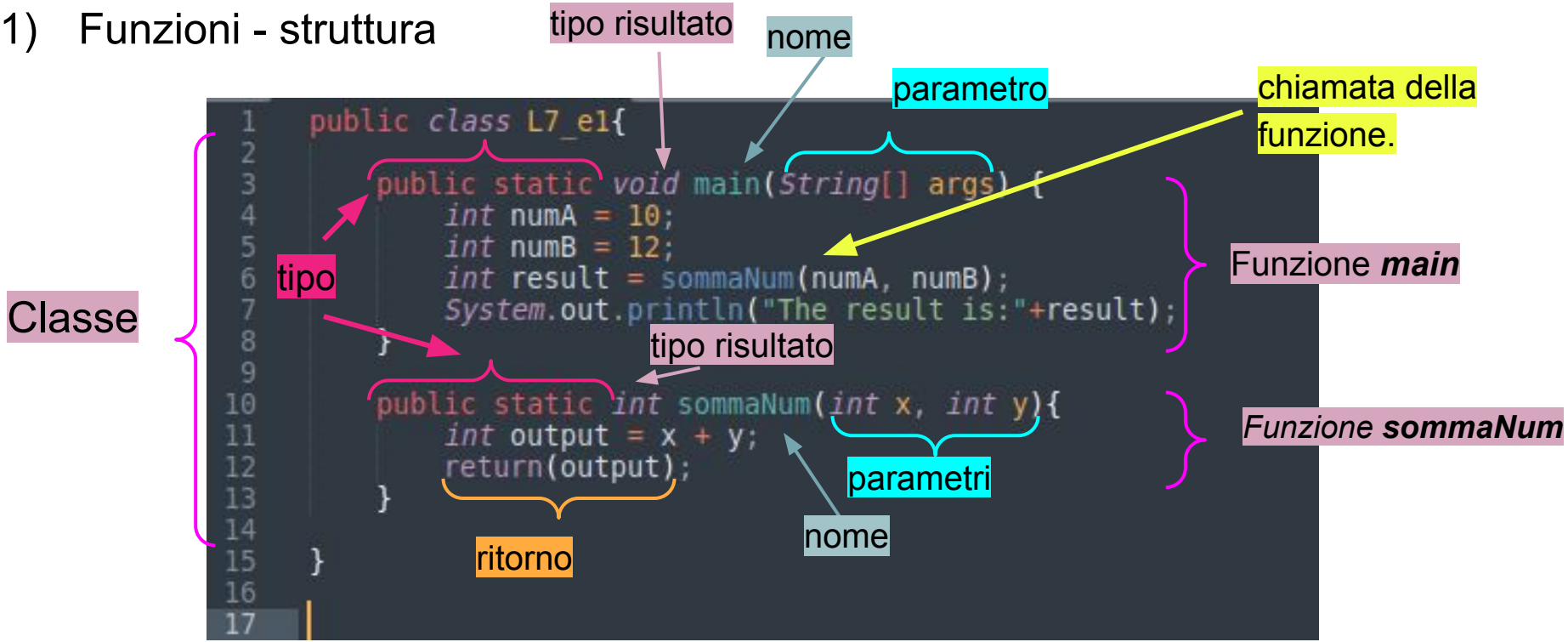
Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni - struttura

```
1   public class L7_e1{
2
3       public static void main(String[] args) {
4           int numA = 10;
5           int numB = 12;
6           int result = sommaNum(numA, numB);
7           System.out.println("The result is:"+result);
8       }
9
10      public static int sommaNum(int x, int y){
11          int output = x + y;
12          return(output);
13      }
14
15  }
16
17
```


Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni - struttura



Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni - Visibilità
   - La visibilità di proprietà e funzioni:
     i) **Public**: sono accessibili sia dall'interno che dall'esterno della classe;
     ii) **Protected**: utilizzati all'interno della classe stessa e all'interno delle classi derivate
     iii) **Private**: possono essere utilizzati soltanto all'interno della classe stessa.
     iv) **Default** (se non specifichiamo): accessibile solo da tutte le classi nel medesimo package.
   - Queste definizioni verranno trattate in modo approfondito quando si tratta di orientazione oggetti.
   - Le **variabile** possono essere dichiarati con le proprietà di visibilità;

Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni - Visibilità e Scope
   - Scope di una variabile l'area del codice nel quale un identificatore resta associato ad un indirizzo di memoria;
   - In Java si distinguono tre tipi di variabili:
     i) variabili locali (all'interno di un metodo);
     ii) variabili di istanza (all'esterno di ogni metodo;)
     iii) variabili di classe ('static');

sono anche chiamati come globali!

ogni blocco (cioè ogni gruppo di linee di codice racchiuso da parentesi graffe {}) definisce uno scope e ogni variabile locale ha come scope l'area di codice che inizia dalla definizione della variabile stessa e termina con il blocco corrente.

per le variabili statiche esiste un'unica ***locazione di memoria***

iv) **static** + **final**: **final** per dichiarare una variabile che potrà essere inizializzata una sola volta e con stati, si definiscono una **costanti;**

Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni - Visibilità e Scope

```
1   public class L7_e2{
2
3       public static final double PI = 3.141592;
4
5       public static void main(String args[]){
6           double raggio = Double.parseDouble(args[0]);
7
8           System.out.println("L'area della sphera è: "+volumeSphera(raggio));
9       }
10
11
12      public static double volumeSphera(double r){
13          return (4/3 * PI * r * r * r);
14      }
15
16  }
```

costanti

parametro del terminale

locale, solo nello scopo “main”

locale, solo nello scopo “volumeSphera”

```
barbon@barbon-PC:~/Scaricati/FI/codici$ java L7_e2 2
L'area della sphera è: 25.132736
```


Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni - Visibilità e Scope

visibilità default

variabile d'istanza

costanti

senza parametri

```
1  public class L7_e3{
2
3      public static final double PI = 3.141592;
4      static double raggio;
5
6      public static void main(String args[]){
7          raggio = Double.parseDouble(args[0]);
8
9          System.out.println("L'area della sphera è: "+volumeSphera());
10     }
11
12
13     public static double volumeSphera(){
14         return (4/3 * PI * raggio * raggio * raggio);
15     }
16
17 }
```

```
barbon@barbon-PC:~/Scaricati/FI/codici$ java L7_e3 2
L'area della sphera è: 25.132736
```


Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni - Overloading
   - Funzioni differenti possono avere lo stesso **identificatore** (sovrapposizione o overloading), esse devono però avere una differente firma (signature):
     i) parametri formali differenti in numero;
     ii) e/o tipo;
     iii) e/o ordine;

```
public static double volumeSphera(double raggio){
     return (4/3 * PI * raggio * raggio * raggio);}

public static double volumeSphera(int raggio){
     return (4/3 * PI * raggio * raggio * raggio);}

public static double volumeSphera(String str){
     double raggio = Double.parseDouble(str);
     return (4/3 * PI * raggio * raggio * raggio);}
```


Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni - Overloading

```
barbon@barbon-PC:~/Scaricati/FI/codici$ java L7_e4 25
L'area della sphera è: 49087.375
L'area della sphera è: 49087.375
barbon@barbon-PC:~/Scaricati/FI/codici$
```

```java
public class L7_e4{

    public static final double PI = 3.141592;

    public static void main(String args[]){
        double raggio = Double.parseDouble(args[0]);

        System.out.println("L'area della sphera è: "+volumeSphera(raggio));
        System.out.println("L'area della sphera è: "+volumeSphera(args[0]));
    }


    public static double volumeSphera(double raggio){
        return (4/3 * PI * raggio * raggio * raggio);
    }

    public static double volumeSphera(int raggio){
        return (4/3 * PI * raggio * raggio * raggio);
    }

    public static double volumeSphera(String str){
        double raggio = Double.parseDouble(str);
        return (4/3 * PI * raggio * raggio * raggio);
    }

}
```

Overloading

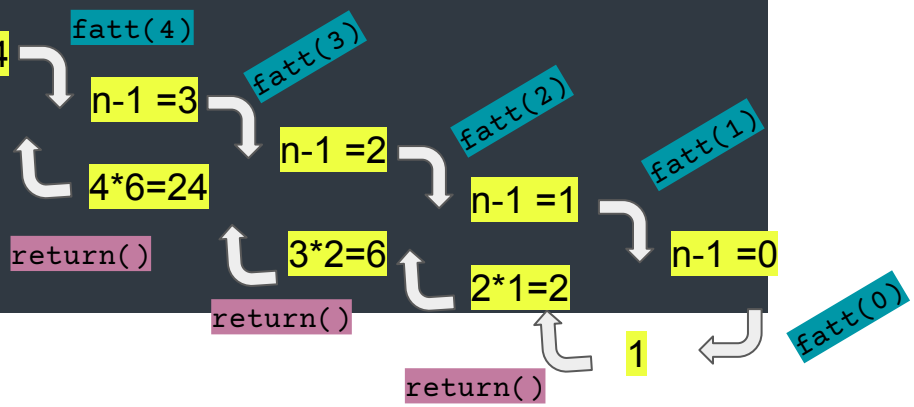
Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni - Ricorsione
   - Una funzione può invocare non solo un'altra funzione, ma anche se stessa;
   - La funzione è **ricorsiva** (si ha una nuova istanza della medesima funzione mentre l'istanza attuale non è ancora conclusa);
   - Un **classico esempio** è quello del calcolo del fattoriale di un numero intero positivo n, che può essere espresso ricorsivamente nel seguente modo:
     i) fattoriale (n) =
        (1) **1** se n = 0
        (2) **n** * fattoriale (n-1) se n > 0

Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni - Ricorsione

```
1   public class Ricorsione{
2       public static void main(String args[]){
3           int n = Integer.parseInt(args[0]);
4           System.out.println("Fattoriale di "+n+" è "+fatt(n));
5       }
6
7       static int fatt(int n){
8           if (n == 0){
9               return 1;
10          }
11          return n*fatt(n-1);
12      }
13  }
```




```
barbon@barbon-PC:~/Scaricati/FI/codici$ java Ricorsione  4
Fattoriale di 4 è 24
```


Sylvio Barbon Junior - sylvio.barbonjunior@units.it

1) Funzioni

```java
//algoritmo che restituisce il numero minore
public class L7_e5{

    public static void main(String args[]){
        int input[] = new int[args.length];

        for(int i=0;i<args.length;i++){
            System.out.println("Input:"+args[i]);
            input[i] = Integer.parseInt(args[i]);
        }


        int ouput = minore(input);
        System.out.println("Il minore è:"+ouput);
    }

    public static int minore(int input[]){

        int piccolo=9999;
        for(int i=0; i<input.length; i++){
            if (input[i]<piccolo){
                piccolo = input[i];
            }
        }
        return(piccolo);
    }
}
```

```
barbon@barbon-PC: ~/Scaricati/FI/codici
barbon@barbon-PC:~/Scaricati/FI/codici$ java L7_e5 19 5200 3 7
Input:19
Input:5200
Input:3
Input:7
Il minore è:3
barbon@barbon-PC:~/Scaricati/FI/codici$
```


Sylvio Barbon Junior - sylvio.barbonjunior@units.it

Grazie!

Sylvio Barbon Junior - sylvio.barbonjunior@units.it