

Programmazione II

A.A. 2022-23

Prof. Maria Tortorella



Dal C al Java
Caratteristiche lessicali
Tipi
Strutture di controllo

Da C a Java

Cominciamo ad analizzare delle differenze attraverso un piccolo programma, implementato in C ed in Java. **Si mettano a confronto le soluzioni**

Problema:

- Data una funzione reale di una variabile reale f , definita per punti nell'intervallo x_1 - x_2 , ed un numero reale x_0 tale che $x_1 < x_0 < x_2$, scrivere un programma che calcoli il valore di f nel punto x_0 mediante interpolazione lineare.

```
#include <stdio.h>

int main (int argc, char *argv[]) {
    float x[10];
    float y[10];
    float x0=0,y0=0;
    for (int i=0;i<10;i++) {
        printf("inserisci x:");
        scanf("%d", &x[i]);
        printf("inserisci y:");
        scanf("%d", &y[i]);
    }
    printf("inserisci valore x0:");
    scanf("%d", &x0);
    int j=0;
    while (x[j]<x0 && j<10)
        j++;
    if (j>0 && j<9) {
        y0=(x0-x[j])*(y[j+1]-y[j])/(x[j+1]-x[j]) + y[j];
        printf("valore in f(%d) = %d \n) " , x0, y0 );
    }
}
```

```

import java.io.*;
public class Esempio {
    public static void main(String[] args) throws Exception{
        BufferedReader stantardin =
            new BufferedReader(new InputStreamReader(System.in));
        float[] x=new float[10];
        float[] y=new float[10];
        float x0=0,y0=0;
        for (int i=0;i<10;i++) {
            System.out.print("inserisci x:");
            x[i] = Float.parseFloat(stantardin.readLine());
            System.out.print("inserisci y:");
            y[i] = Float.parseFloat(stantardin.readLine());
        }
        System.out.print("inserisci valore x0:");
        x0 = Float.parseFloat(stantardin.readLine());
        int j=0;
        while (x[j] < x0 && j < 10)
            j++;
        if (j > 0 && j < 9) {
            y0=(x0-x[j])*(y[j+1]-y[j])/(x[j+1]-x[j]) + y[j];
            System.out.println("valore in f("+x0+")="+y0);
        }
    }
}

```

Java

Confronto tra le due versioni ...

C

```
#include <iostream>

int main (int argc, char *argv[]) {
    float x[10];
    float y[10];
    float x0=0,y0=0;

    ...

}
```

Java

```
import java.io.*;
public class Esempio {
    public static void main(String[] args) throws Exception{
        BufferedReader stantardin = new
            BufferedReader(new InputStreamReader(System.in));
        float[] x=new float[10];
        float[] y=new float[10];
        float x0=0,y0=0;
        ....
    }
}
```

... confronto tra le due versioni ...

C

```
...  
for (int i=0;i<10;i++) {  
    printf("inserisci x:");  
    scanf("%d", &x[i]);  
    printf("inserisci y:");  
    scanf("%d", &y[i]);}  
}  
printf("inserisci valore x0:");  
scanf("%d", &x0);
```

Java

```
...  
for (int i=0;i<10;i++) {  
    System.out.print("inserisci x:");  
    x[i] = Float.parseFloat(stantardin.readLine());  
    System.out.print("inserisci y:");  
    y[i] = Float.parseFloat(stantardin.readLine());  
}  
System.out.print("inserisci valore x0:");  
x0 = Float.parseFloat(stantardin.readLine());  
...
```

... confronto tra le due versioni

C

```
...
int j=0;
while (x[j]<x0 && j<10)
    j++;
if (j>0 && j<9) {
    y0=(x0-x[j])*(y[j+1]-y[j])/(x[j+1]-x[j]) + y[j];
    printf("valore in f(%d) = %d \n" , x0, y0 );
}
}
```

Java

```
...
int j=0;
while (x[j] < x0 && j < 10)
    j++;
if (j > 0 && j < 9) {
    y0=(x0-x[j])*(y[j+1]-y[j])/(x[j+1]-x[j]) + y[j];
    System.out.println("valore in f("+x0+")="+y0);
}
}
```

Alfabeto

- I sorgenti Java utilizzano il codice standard internazionale a 16 bit Unicode, che comprende gli alfabeti più diffusi: arabo, greco, ebraico, cirillico, thai, katakana, hiragana, cinese, coreano, e molti altri (*<http://www.unicode.org>*)
- I programmi Java possono essere scritti con altre codifiche (es. ASCII), ma sono convertiti in Unicode prima di essere compilati

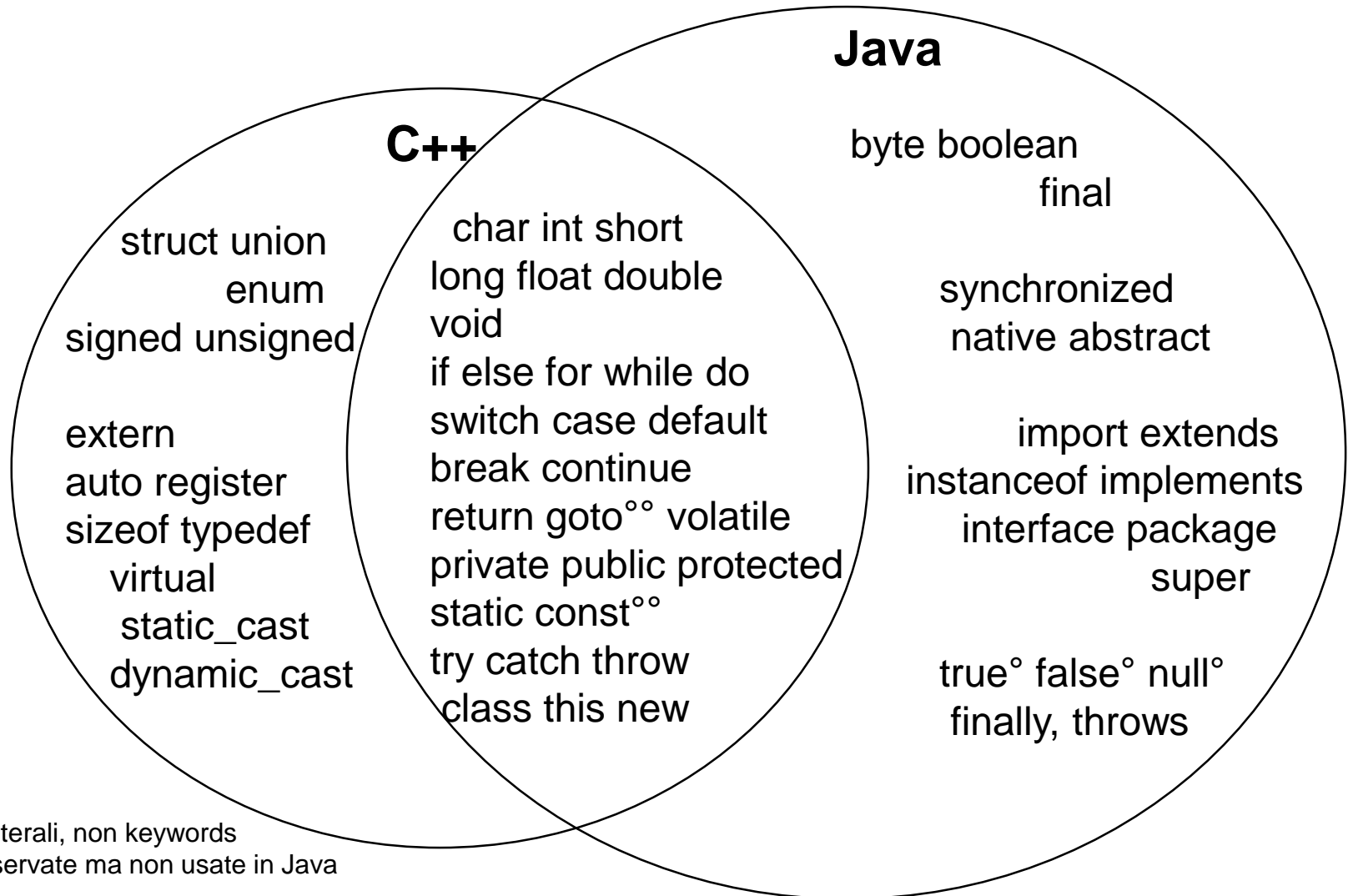
Parole riservate

Le seguenti keyword non possono essere usate come identificatori:

<i>abstract</i>	<i>double</i>	<i>int</i>	<i>super</i>
<i>boolean</i>	<i>else</i>	<i>interface</i>	<i>switch</i>
<i>break</i>	<i>extends</i>	<i>long</i>	<i>synchronized</i>
<i>byte</i>	<i>final</i>	<i>native</i>	<i>this</i>
<i>case</i>	<i>finally</i>	<i>new</i>	<i>throw</i>
<i>catch</i>	<i>float</i>	<i>package</i>	<i>throws</i>
<i>char</i>	<i>for</i>	<i>private</i>	<i>transient</i>
<i>class</i>	<i>(goto)</i>	<i>protected</i>	<i>try</i>
<i>(const)</i>	<i>if</i>	<i>public</i>	<i>void</i>
<i>continue</i>	<i>implements</i>	<i>return</i>	<i>volatile</i>
<i>default</i>	<i>import</i>	<i>short</i>	<i>while</i>
<i>do</i>	<i>instanceof</i>	<i>static</i>	

Note: - *const* e *goto* sono riservate, ma non usate
- anche i letterali *null*, *true*, *false* sono riservati

Parole riservate Java e C++



Formato e commenti

- Il formato di un sorgente Java è libero: gli spazi (**blank, tab, newline, form feed**) non sono significativi, tranne che per separare le parole fra loro

Sono possibili tre diversi stili di commento:

➤ **/* Commento tradizionale, eventualmente su più linee, non nidificato */**

➤ **// Commento su di una sola linea**

➤ **/** Commento di documentazione". */**

Solo immediatamente prima di una dichiarazione di classe, interfaccia, metodo o campo, e viene incluso nella documentazione generabile automaticamente a partire dal codice sorgente (es.: **javadoc**)

Tipi

- Linguaggio fortemente tipato:
il tipo di una espressione è sempre noto a tempo di compilazione
- Linguaggio a oggetti non puro:
non tutti i tipi sono classi; esistono tipi predefiniti
- Rispetto al C, non esistono:
signed, unsigned, long double, enum, puntatori, struct, union, typedef

Formato dei dati

- Il formato dei dati è specificato esattamente (strong type-checking):

✓ byte	8 bit complemento a 2
✓ short	16 bit "
✓ int	32 bit "
✓ long	64 bit "
✓ float	32 bit IEEE 754 floating
✓ double	64 bit "
✓ char	16 bit Unicode

Classificazione dei tipi

	TIPI		KEYWORD	NOTE
Primitivi	booleani		boolean	true, false
	numerici	interi	byte short int long	8 bit interi in compl. a 2 16 bit 32 bit 64 bit
		floating-point	float double	32 bit IEEE 754 64 bit
	caratteri		char	16 bit unicode
Reference	classi		class	
	interfacce		interface	
	array			
Null				

Tipi primitivi

- Dichiarazioni: `int i; float f; char a, b;`
- Inizializzazioni: `double pi = 3.14;`
- Espressioni: `i + 8`
`j = i++`

Essenzialmente, gli stessi operatori del C

Nota:

byte e ***short*** vengono sempre promossi a ***int*** prima di essere valutati

- Assegnamenti: `i = j + 5;`

Classi “wrapper”

- Tutti i tipi primitivi hanno una classe corrispondente in un package ***java.lang***:
Boolean, Integer, Short, Byte, Long, Character, Float, Double,
(“wrapper class”)
- Ogni classe definisce metodi e costanti utili per quel tipo, ad esempio:
 - Classe: ***Character***
 - Costanti: ***MIN_VALUE, MAX_VALUE, ...***
 - Metodi (***static***): ne parleremo quando li incontreremo

Letterali

■ Ogni tipo Java ha i propri letterali

- ✓ Tipi reference: *null*
- ✓ Tipo boolean: *false* e *true*
- ✓ Tipi interi: *40* *046* *0x4F*
- ✓ Tipi floating point: *20.* *2.0e1* *0.20E2*
- ✓ Tipo char: *\n* *\t* *\b* *\r* *\f* ** *\'* *\"* *\ddd*
- ✓ Tipo stringa: sequenze di letterali di tipi char

Operatori

Si riportano gli operatori del linguaggio Java in ordine decrescente di priorità

[]	expr++	expr--	
++expr	--expr	+expr	-expr
new	(tipo)expr		
* /	%		
+ -			
< >	>=	<=	
== !=			
&&			
?:			
= += -= *= /=			

Alcune strutture di controllo

Non c'è goto!

Sequenza		
Selezione	<i>if else</i> <i>switch</i>	
Iterazione	<i>for</i> <i>while</i> <i>do-while</i>	
Salto	<i>break</i> <i>continue</i> <i>return</i>	uscita da un blocco continua un loop da un metodo
Gestione eccezioni	<i>try-catch- finally-throw</i>	

Sintassi delle strutture di controllo

boolean,
non int!

```
if ( condition )  
    statement  
[else  
    statement ]
```

```
switch ( intexpr ) {  
    case intexpr : statement  
    [case intexpr : statement  
        ...  
    default : statement ]  
}
```

```
while ( condition )  
    statement  
  
do  
    statement  
while ( condition );
```

```
for ( init; condition; increment )  
  
    statement
```

Lo statement può essere un'istruzione o un blocco di istruzioni contenute tra { }

Uscite

- ❑ *break [label];*
- ❑ *continue [label];*
- ❑ *return expr;*
- ❑ *label: statement*

```
label: for (i=0 , j=0 ; i<10 && j<20; i++ , j++) {  
    for (z=0 ; z<100 ; z++) {  
        switch (expr) {  
            case tag:  statements; break; /* esce da switch */  
            ....  
            default:  statements; break label; /* esce da label */  
        }  
        if (z==15) continue; /* prosegue il for z */  
        if (z==i+j) continue label; /* prosegue da label */  
    }  
}
```

Forma di un programma (1)

- Almeno per le prime lezioni:

```
import java.io.*;

public class ProgramName {
    public static void main (String[] arg) {
        statement
        statement
        ...
        statement
    }
}
```

Ricordare che:

- Ogni programma Java è costituito da una o più classi
 - Ogni file Java include una o più classi
 - Componente indispensabile dell'infrastruttura:

```
public class ProgramName {  
    ...  
}
```

- Ciascuna classe può contenere uno o più metodi
- Ogni programma Java deve avere un metodo main
 - Componente indispensabile dell'infrastruttura:

```
public static void main (String[] arg) {  
    ...  
}
```

Un primo esercizio

- Scrivere un programma che annunci il vostro primo programma e l'intenzione di continuare



Elementi del linguaggio Java

- Regole per la costruzione delle frasi
- Identificatori
- Parole chiave (keywords)
 - Significati predefiniti
 - Es: import, class, public, static, void
 - NON PrintStream
- Ordine delle istruzioni
 - Ordine lessicografico coincidente con l'ordine di esecuzione

Elementi del linguaggio Java

- Il formato del testo del programma è libero
 - Indentazione
 - Convenzioni
 - Una istruzione per linea
 - Usare i tab e non gli spazi per l'indentazione
- Commenti
 - Racchiusi da delimitatori: `/* ...*/`
 - Su linea singola: `//`

Elementi del linguaggio Java

```
/* Il mio primo commento */
```

```
/*  
 * Il mio primo commento  
 */
```

```
// Il mio primo commento
```

```
Statement // Il mio primo commento
```

Linee guida

- Inserire un commento prima di ogni classe per spiegarne scopo e comportamento
- NON spiegare come funziona Java
- Dare informazioni che non possono essere facilmente derivate dal codice
- Mai interrompere una istruzione con un commento

Esempio

```
import java.io.*;

/*
 *Program1: Stampa un saluto
 */
class Program1 {
    public static void main (String[] arg) {
        System.out.println("Benvenuti al corso");
    }
}
```

Errori

- ❑ A tempo di compilazione
 - Scoperti dal compilatore
 - Es: mancanza del punto e virgola, parentesi aperte e non chiuse
- ❑ A tempo di esecuzione (run time)
 - Scoperti solo all'atto dell'esecuzione del programma
 - Es: output difforme da quello atteso

Esempio

- Tre tipi di errori sul nostro programma:

```
System.out.println("Benvenuti al corso")
```

```
System.out.println("Benvenuti al corso);
```

```
class Program1 {  
    public static void Main (String[] arg) {  
        System.out.println("Benvenuti al corso");  
    }  
}
```

Gli strumenti di sviluppo di Java

Java Development kit (JDK)

- Dove recuperare i file del JDK?

<http://www.oracle.com>

<https://www.oracle.com/java/technologies/downloads/>

Diverse versioni disponibili – fino alla 17 – Java 17
minimo Java 11 (Java SE Development Kit 11.0.12)
massimo Java 16 (Java SE Development Kit 16.0.2)

- Cosa contiene il JDK :

- FILE SORGENTI (`src.zip`)
- STRUMENTI
- DOCUMENTAZIONE E DEMO

Editing

PariDispari.java

```
import java.util.Scanner;

public class PariDispari {

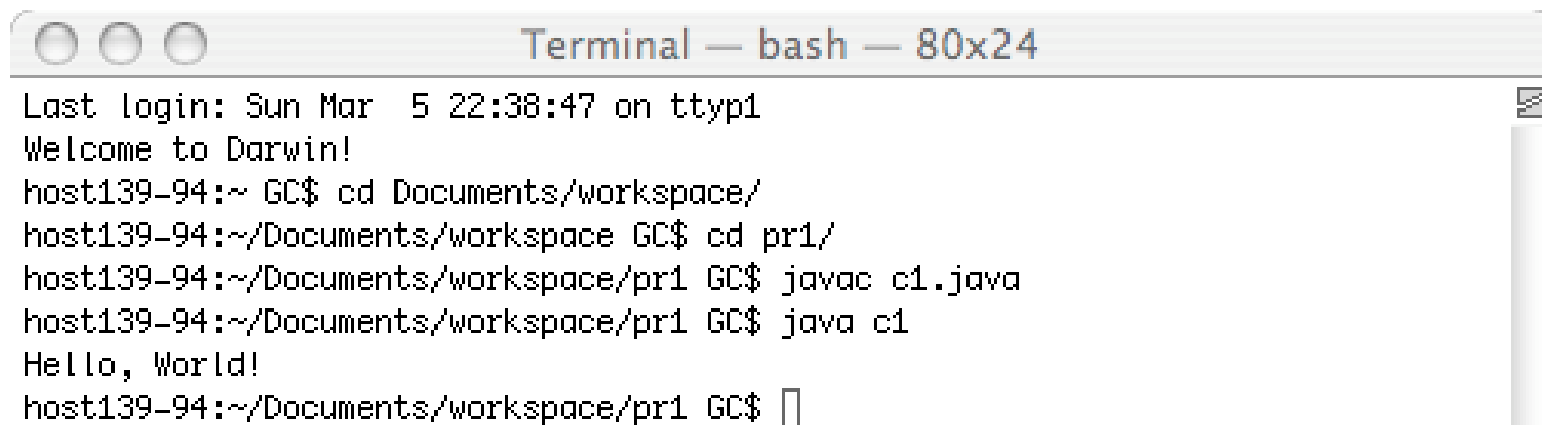
    public static void main(String[] args) {

        System.out.print("Inserisci un numero: ");

        Scanner s= new Scanner(System.in);
        int value=s.nextInt();
        s.close();

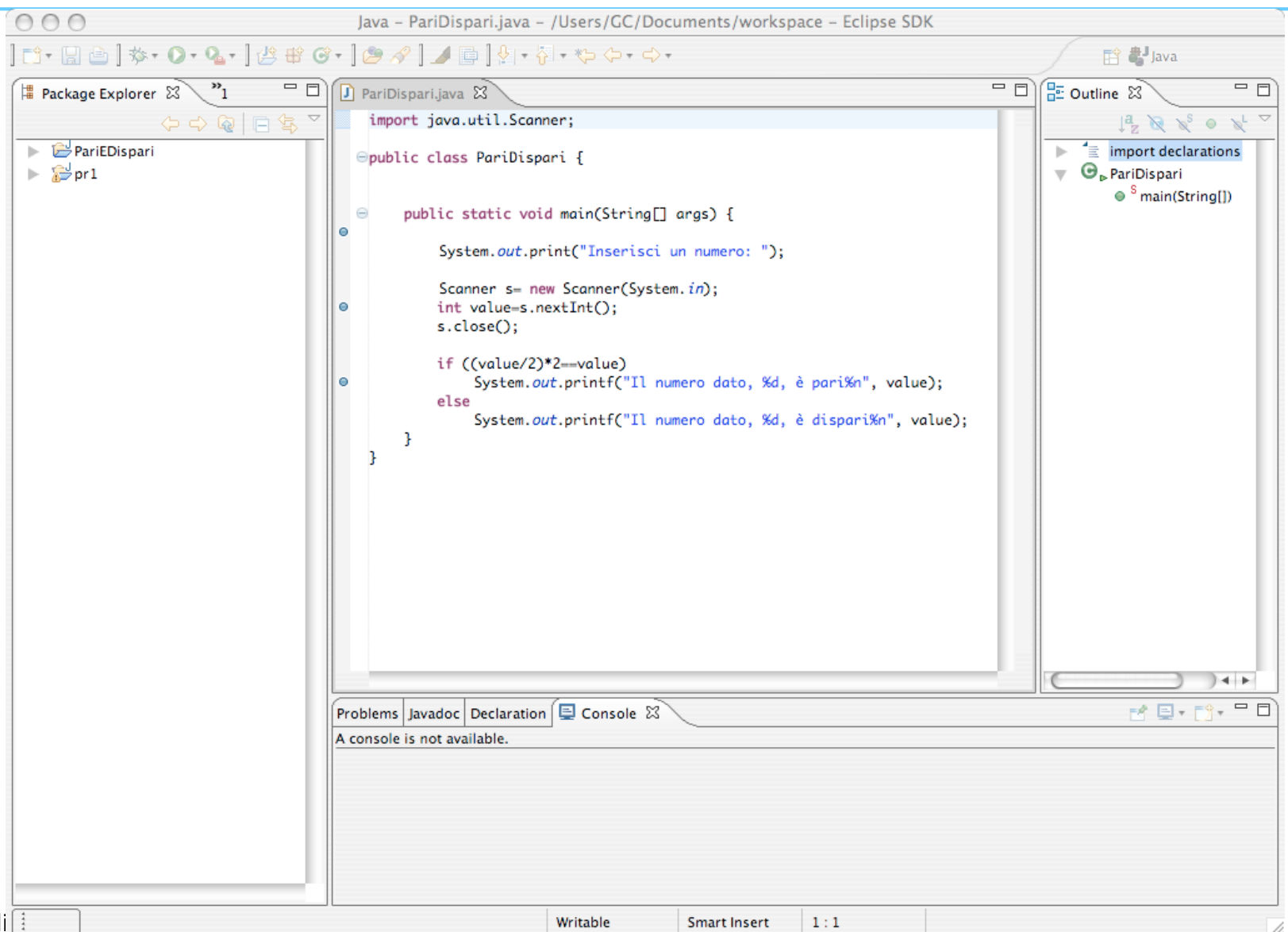
        if ((value/2)*2==value)
            System.out.printf("Il numero dato, %d, è pari\n", value);
        else
            System.out.printf("Il numero dato, %d, è dispari\n", value);
    }
}
```

Compilare ed eseguire



```
Terminal — bash — 80x24
Last login: Sun Mar  5 22:38:47 on ttty1
Welcome to Darwin!
host139-94:~ GC$ cd Documents/workspace/
host139-94:~/Documents/workspace GC$ cd pr1/
host139-94:~/Documents/workspace/pr1 GC$ javac c1.java
host139-94:~/Documents/workspace/pr1 GC$ java c1
Hello, World!
host139-94:~/Documents/workspace/pr1 GC$
```

Editing



Compilare ed eseguire

