CS107
Introduction to Computer Science

Java Basics

Java

- Java was developed in the early 90s by Sun Microsystems
- Java is a high-level language
- Java programs are portable across platforms
  - Each program is translated into Java bytecode
  - Each machine has a Java Virtual Machine (JVM) which knows how to execute Java bytecode
- Java is object-oriented
  - We will not use objects in this class

A Java program

```
/*
Here you describe what your program does.

Laura Toma
Csci107
*/

public class CLASS-NAME {
    public static void main (String args[]) {

        // your program goes here

    } // end of main
} // end of class
```

Compiling and Running

In order to run a Java program:

- First you compile it
  - that is, you run a program called **compiler** that checks whether the program follows the Java syntax
  - if it finds errors, it lists them
  - If there are no errors, it translates the program into Java bytecode
  - Example: assume you created a program called Hello.java
    prompt>javac Hello.java
  - If successful, this creates a file Hello.class which contains the translation (Java bytecode) of Hello.java
- Then you execute it
  - That is, you call the Java Virtual Machine to interpret and execute the Java bytecode of your program
  - Example:
    prompt>java Hello

The infamous **Hello world** program

When learning a new language, the first program people usually write is one that salutes the world :). Here is the Hello world program in Java.

```
/*
This program prints out "hello world!" and terminates.
*/
public class Hello {
    public static void main (String args[]) {

        System.out.println("Hello world!");

    } // end of main
} // end of class
```

Notes

- Comments
  - what follows after // on the same line is considered comment
  - Or, what is in between /* this is a comment */
- Indentation
  - is for the convenience of the reader; compiler ignores all spaces and new lines ; the delimiter for the compiler is the semicolon
- All statements ended by semicolon
- Lower vs. upper case matters!!
  - Void is different than void
  - Main is different that main

Variable declaration

type variable-name;
Meaning: variable <variable-name> will be a variable of type <type>

Where type can be:

- int //integer
- double //real number
- char //character

Example:

```
int a, b, c;
double x;
int sum;
char my-character;
```

Input

```
/* this line should appear once in your program;  basically it declares a
   variable r which knows how read input from the user */
ReadStream r = new ReadStream();

/* Then you can use r.readInt(), r.readDouble() or r.readChar() to read
   integers, decimal and character value from the user.
*/

int a;
a= r.readInt();
r.readLine();
/* Meaning: read an integer from the user and store it into the variable
   called a
*/
```

Input

- Reading decimal numbers

```
double a;
a= r.readDouble();
r.readLine();
/* Meaning: read a decimal value from the user and store it into the variable called a*/
```

- Reading characters

```
char a;
c= r.readChar();
r.readLine();
/* Meaning: read a character from the user and store it into the variable called a*/
```

- Note that every read must be followed bythe r.readLine() instruction. This discards the rest of the line entered by the user.

Output

System.out.println(variable-name);
prints the value of variable <variable-name> to the user

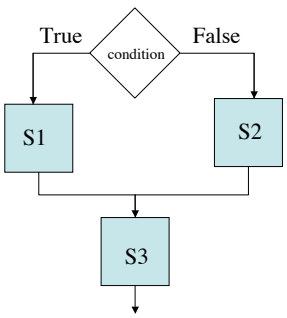
System.out.println("any message ");
prints the message within quotes to the user

Note: System.out.println() always prints on a new line.

System.out.println("hello" + "world" + a + "plus" + b);

If statements

```
if (condition) {
    S1;
}
else {
    S2;
}
S3;
```



Boolean conditions

..are built using

- Comparison operators
  - == equal
  - != not equal
  - < less than
  - > greater than
  - <= less than or equal
  - >= greater than or equal

- Boolean operators
  - && and
  - || or
  - ! not

Examples

Assume we declared the following variables:

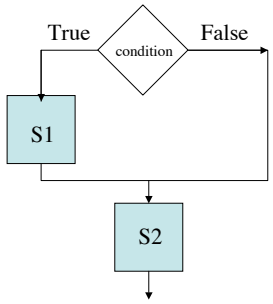
```
int a = 2, b=5, c=10;
```

Here are some examples of boolean conditions we can use:

- `if (a == b) ...`
- `if (a != b) ...`
- `if (a <= b+c) ...`
- `If ((a <= b) && (b <= c)) ...`
- `if !((a < b) && (b<c)) ...`

While statements

```
while (condition) {
   S1;
}
S2;
```



Example

```
/*
This program reads 100 numbers from the user and outputs their sum
*/
public class ComputeSum {
     public static void main (String args[]) {
               ReadStream r = new ReadStream();
               int i, sum, x;
               sum=0;
               i=1;
               while (i <= 100) {
                         x = r.readInt();
                         r.readLine();
                         sum = sum + x;
                         i = i+1;
               }
               System.out.println("sum is " + sum);
               System.out.println("Goodbye");
     } //end of main
}//end of class
```

Class exercise

- Write a program that asks the user
  - Do you want to use this program? (y/n)
- If the user says 'y' then the program terminates
- If the user says 'n' then the program asks
  - Are you really sure you do not want to use this program? (y/n)
  - If the user says 'n' it terminates, otherwise it prints again the message
  - Are you really really sure you do not want to use this program? (y/n)
  - And so on, every time adding one more "really".