ALGONQUIN COLLEGE

# CST8110 INTRODUCTION TO PROGRAMMING

## Week 2

## Introduction to Java

# Programming

- Writing instructions line by line which will be translated to machine code and executed by the CPU
- Computer program
  - A set/list of instructions that tell a computer what to do
  - A collection of instructions that performs a specific task when executed by a computer

# Programming Language

- A programming language is a vocabulary and a set of grammatical rules for instructing a computer to perform specific tasks
- Examples: C, C++, **Java**, COBOL, Python

# Programming Software

- What is programming software?
  - Is a **software** which helps the **programmer** in developing other **software**.
  - Examples: Compilers, assemblers, interpreters etc.
  - a **compiler** or an **interpreter** is a program that converts program written in high-level language into machine code understood by the computer.
- Integrated Development Environments (IDEs) are combinations of all these **software.**
  - **Example: Eclipse**

# Useful videos

- Compiler/Interpreter - https://www.youtube.com/watch?v=_C5AHaS1mOA

- High level languages - https://www.youtube.com/watch?v=kil2Z3ij-JA

# Advantages of Java

- Platform independent
- Portable “byte code” – write once, run anywhere
- A rich language which uses many foundational programming methods such as classes, inheritance, interfaces, polymorphism etc.
- Widely used in industry
- Once you learn Java, you can easily learn any other programming language

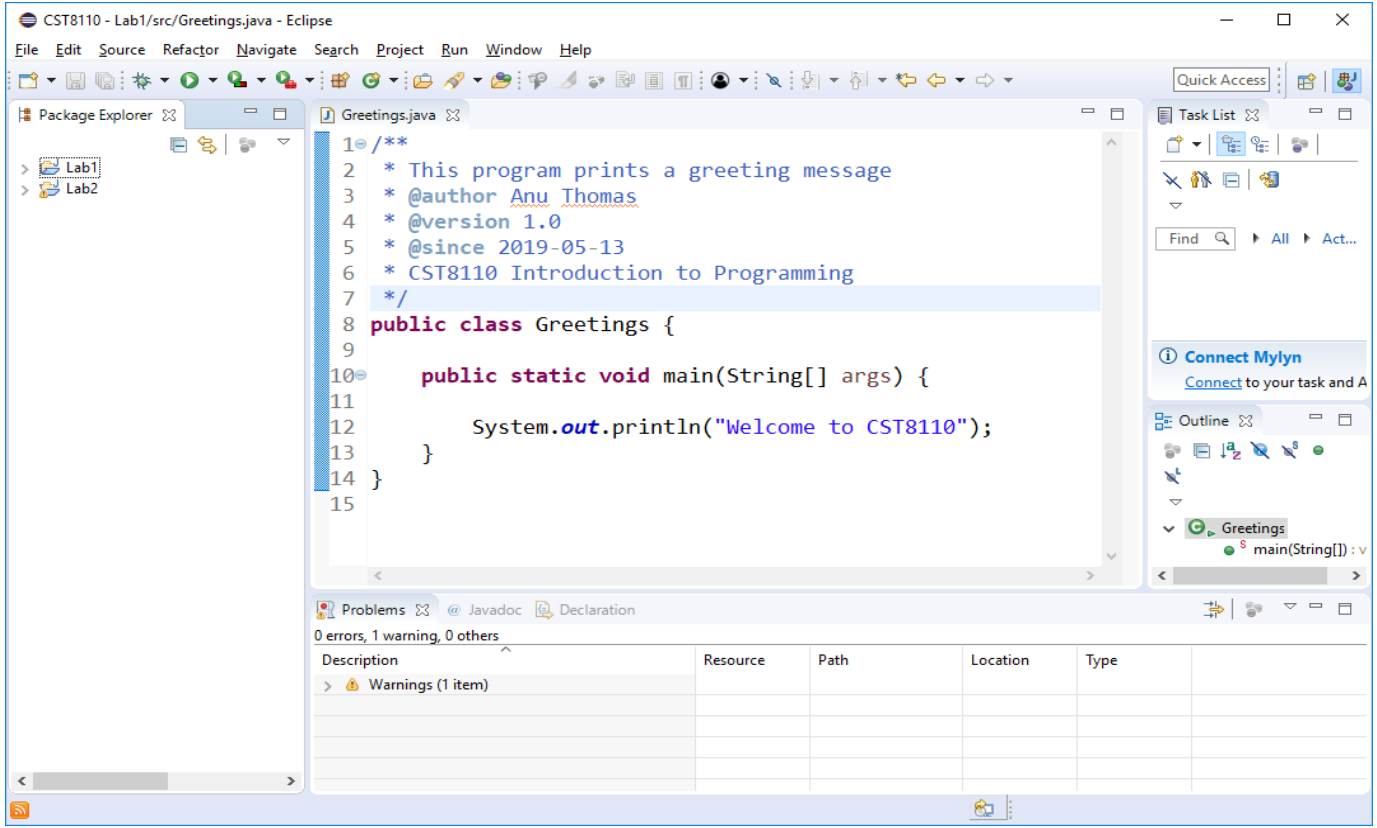
# Java Versions

- Version is not important, changes between versions are minor and usually in advanced features and can be easily learned
- Basics remain the same
- Saying you know how to program in Java 9 is like saying you know how to drive a 2017 Honda Civic

# Development Cycle for Java

- Step 1 - Editor
  - Type in our instructions/statements according to rules of Java Language
  - File from editor is called a source file and will have extension **.java**
- Step 2 - Compiler
  - Use compiler to translate the statements in source file to bytecode (which are portable – not dependent on hardware platform)
  - File from compiler is called bytecode file and will have extension **.class**
  - Use command> javac ProgramName.java
- Step 3 – Class Loader
  - class loader loads bytecode files into memory (includes all the .class files that you have written/produced and those provided by Java that you are using)
  - Use command> java ProgramName to invoke Loader, Verification, Execution
- Step 4 – Bytecode verifier
  - Use bytecode verifier to ensure bytecodes are valid and do not violate Java's security restrictions
- Step 5 – Execution
  - Use JVM (Java Virtual Machine) to execute program's bytecodes (to perform the actions specified by your program)

# Sample Java Class

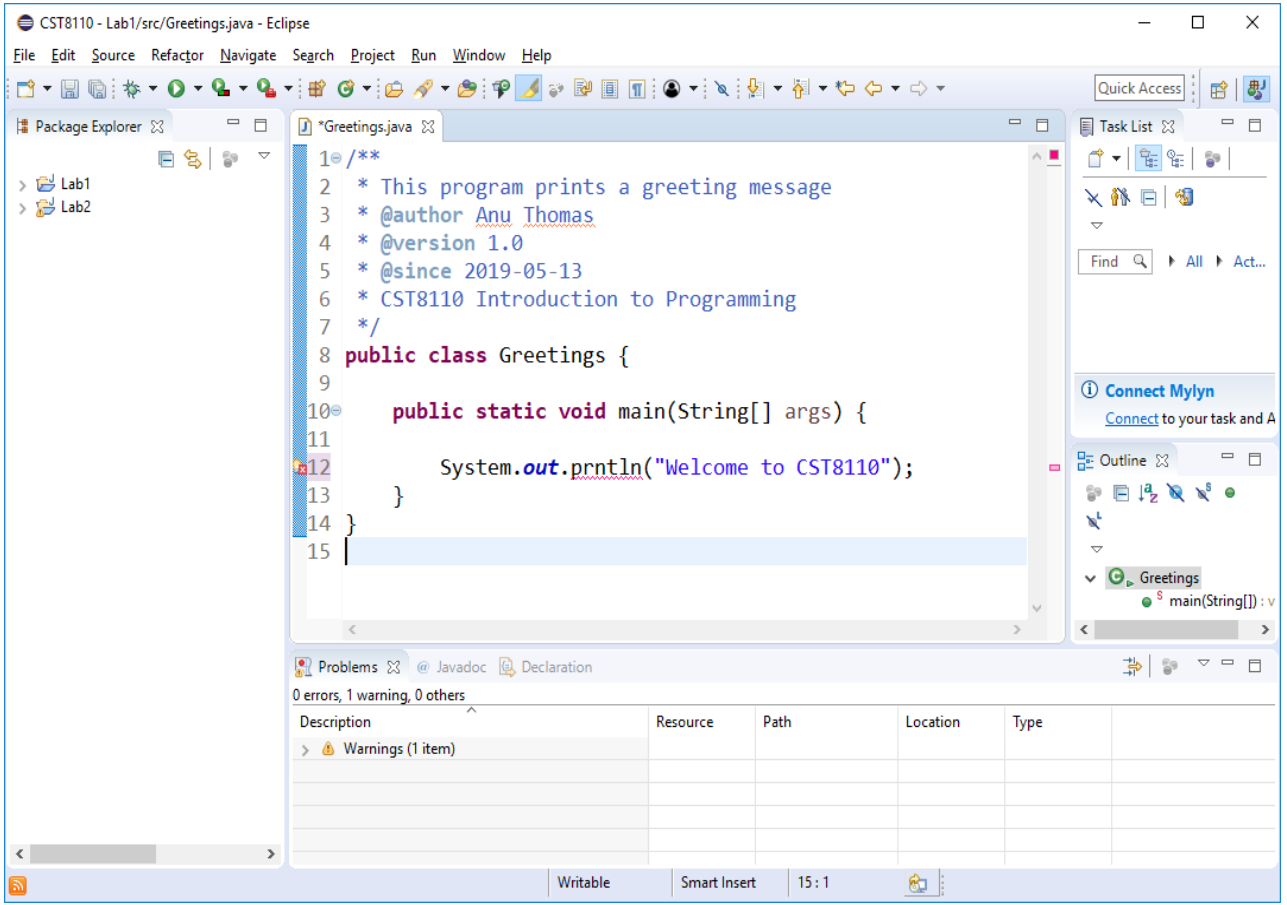


# General Java Notes

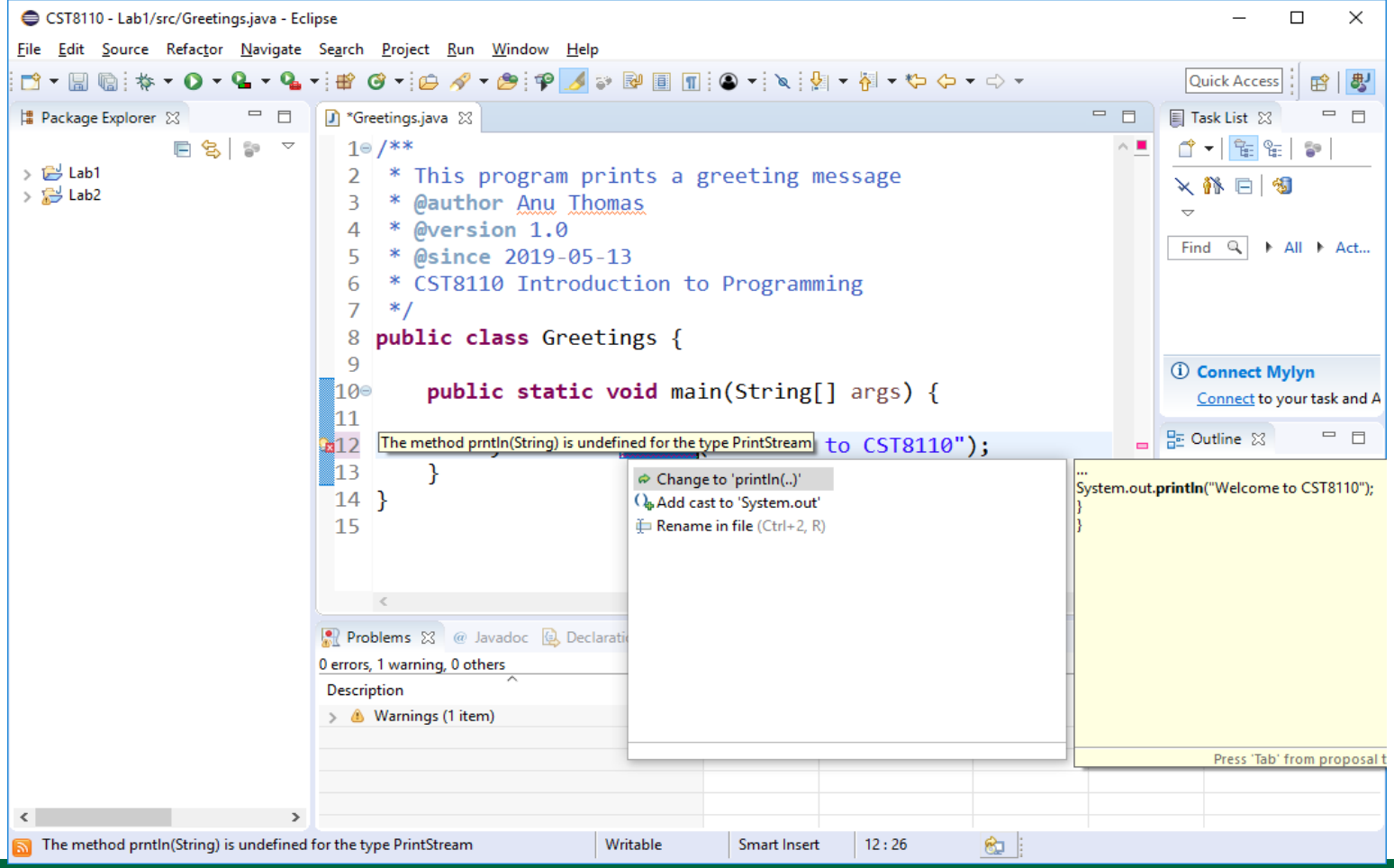
- Everything in Java is case sensitive…
- Statements end in semicolon ;
- Comments – ignored by java compiler:
  - //  means ignore to end of line
  - /*  means ignore until you hit  */
  - /**  special comment used to generate documentation */
- Blocks are bounded by {  }
- Must have one and only one…

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

# Error!


CST8110 - Lab1/src/Greetings.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
Lab1
Lab2
*Greetings.java
1 /**
2 * This program prints a greeting message
3 * @author Anu Thomas
4 * @version 1.0
5 * @since 2019-05-13
6 * CST8110 Introduction to Programming
7 */
8 public class Greetings {
9
10 public static void main(String[] args) {
11
12 System.out.prntln("Welcome to CST8110");
13 }
14 }
15
Task List
Find All Act...
Connect Mylyn
Connect to your task and A
Outline
Greetings
main(String[]) : v
Problems
Javadoc
Declaration
0 errors, 1 warning, 0 others
Description
Resource
Path
Location
Type
Warnings (1 item)
Writable
Smart Insert
15 : 1

# Error


CST8110 - Lab1/src/Greetings.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
Lab1
Lab2
*Greetings.java
1 /**
2 * This program prints a greeting message
3 * @author Anu Thomas
4 * @version 1.0
5 * @since 2019-05-13
6 * CST8110 Introduction to Programming
7 */
8 public class Greetings {
9
10 public static void main(String[] args) {
11
12 The method prntln(String) is undefined for the type PrintStream to CST8110");
13 }
14 }
15
Task List
Find All Act...
Connect Mylyn
Connect to your task and A
Outline
Change to 'println(..)'
Add cast to 'System.out'
Rename in file (Ctrl+2, R)
System.out.println("Welcome to CST8110");
}
}
Press 'Tab' from proposal t
Problems
Javadoc
0 errors, 1 warning, 0 others
Description
Warnings (1 item)
The method prntln(String) is undefined for the type PrintStream
Writable
Smart Insert
12 : 26

# print, println and \n

- Newline characters indicate to System.out's print and println methods when to position the output cursor at the beginning of the next line in the command window.
- Newline characters are whitespace characters.
- The backslash (\) is called an escape character.
  - Indicates a "special character"
- Backslash is combined with the next character to form an escape sequence—\n represents the newline character.

# Example

```
1  // Fig. 2.3: Welcome2.java
2  // Printing a line of text with multiple statements.
3
4  public class Welcome2
5  {
6     // main method begins execution of Java application
7     public static void main(String[] args)
8     {
9        System.out.print("Welcome to ");
10       System.out.println("Java Programming!");
11    } // end method main
12 } // end class Welcome2
```

```
Welcome to Java Programming!
```

**Fig. 2.3** | Printing a line of text with multiple statements.

# Example

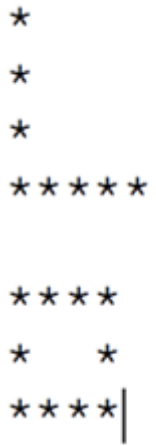
```
1  // Fig. 2.4: Welcome3.java
2  // Printing multiple lines of text with a single statement.
3
4  public class Welcome3
5  {
6     // main method begins execution of Java application
7     public static void main(String[] args)
8     {
9        System.out.println("Welcome\nto\nJava\nProgramming!");
10    } // end method main
11 } // end class Welcome3
```

```
Welcome
to
Java
Programming!
```

**Fig. 2.4** | Printing multiple lines of text with a single statement.

# Practice

1. Write a program to print “Hello John Doe” (or whatever your name is)
2. Write a program to print a box like this



# Forward slash vs back slash

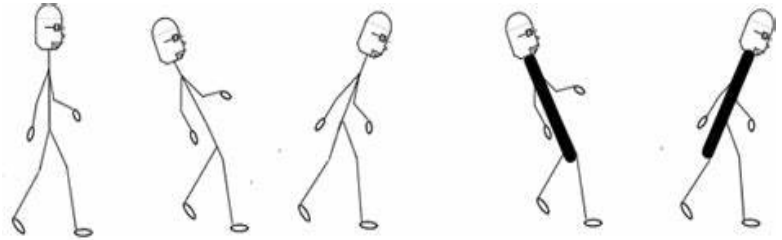
\

**This is a backslash.**

/

**This is a forward slash.**

Let us all memorize this, and end the confusion.



# Everything matters

- Uppercase lowercase matters ie. int x not the same as int X
- / not the same as \
- ; not the same as :
- Spaces generally don’t matter
- Brackets and quotes must be in pairs () {} “”
- Usually it is a small problem (one character)

# Engineering notebook

- Keep a notebook or google doc
- Every time you get an error and figure it out write down the details
- You will thank yourself!

| Error Message | Cause | Fix |
|---|---|---|
| String literal is not properly closed by a double-quote | Did not have a matching pair of quotes | Add missing quote. Strings should be enclosed in double-quotes |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |