INFO1103

Introduction to Programing

Introduction to Programming

Programming Basics

Introduction to Computers & Java

Java: **Object-Oriented Programming Language** (OOP)

Source Code (Programming Language) > (**Compiler**) > **JVM** (**Java Virtual Machine**) > Machine Code (**Byte**/**Binary**/**Executable code**)

Compiler: Translates Java Code → Bytecode
JVM: **Interpreter** Translating & Executing Bytecode (**Java Bytecode**: Versatile and Run on any computer that has a JVM)

Applications: Regular programs
Applet: Run within a web browser
Package: Library of classes

Machine Code (byte/binary/executable code) = **Program**/Application

Java Compiler (JDK – Java Development Kit) = Compile a file from **.java** (Source Code) to **.class** (Executable Code)

Compiling a Java File
.java > .class (*class file* – **Bytecode**)
javac 'Class Name.java'

- Should NOT produce output ∴ no errors detected by compiler

Running – Invoking JVM to run the program
java 'Class Name'

Method Header (**Main Method**)
Method Body

'**=**' Assignment operator (assign a value)
'**int**': whole number
'**double**': fraction/decimal with numbers on both left and right hand side of decimal point

'**print**': print but what comes after stays on same line
'**println**': print a line but what comes after goes to the next line below

Program: Set of instructions that a computer can understand & execute
Compiler: Translate a programming language from high-level language (programming language/source code) we as users input into low-level language (machine code/ binary/ executable code)

- Write code (human can read)
- Compile code to executable code (computer can read)
- Run the program

Interpreter: Program that alternates Translation & Execution of statements in a program written in high-level language

Source code >>> exe > computer
Source code >>> exe > virtual machine > computer (with java)

Computer does 4 things?

1. Computer/process
2. Input
3. Output
4. Store Information

- Brackets come in pairs and section out a **block** **(), {}, []**
- End statements in semicolons**;**
- Upper/lowercase is important & distinct
- Word following a dot belongs to the former word (e.g. system.out)

Java Syntax

- *Reserved* words (E.g. **public**, **static**, **void**, **int**, **float**) – Can't be used for variable names
- **Variable** names cannot begin with numbers
- Blocks of code are presented with braces **{ }**
- Expressions presented with parentheses **( )** – **ARGUMENTS**
- Array items accessed with brackets **[ ]**
- Statements end with semi-colons**;**
- Strings limited by double quotes **" "** (not single/curly ones)

Errors

1. **Syntax error**: Grammatical errors
2. **Logic error**: Conceptual mistakes in the program/algorithm
3. **Run-time error**: Errors occurring during execution

E.g. in 'Hello World' Program