Use the following outline to guide your self-assessment and notetaking

Week 1 – Introduction to Java and Problem Solving

The Java Programming Language (Ch 1.4)

A Java Program

* Defined using a class definition (public class + name of file)
* Inside the class definition, consists of methods
  + **Method:** Group of programming statements that are invoked when a program is executed
  + There must be a main method which is called once the program is executed
  + Main method preceded by (public static void)
* Delimited by curly braces

Comments

* Do not affect what a program does, but just makes understanding the code easier
* Should identify the purpose of the program
* **Documentation:** Written comments or diocuments, user guide, or technical references
* Comments are inline documentation
* // denotes a single line comment, /\* … \*/ denotes a comment block

Identifiers and Reserved Words

* Various words when writing programs
* Fall into 3 categories:
  + Made up when writing (ex: Lincoln, args)
  + Words that another programmer chose (String, System, out, println, main)
  + Reserved keywords (class, public, static, void)
* Java can use both upper and lowercase letters as an identifier, but they must start with a letter and follow only with more letters and digits (no spaces allowed)
* Should be readable and descriptive.

White Space

* Used to separate words and symbols in a program
* Consists of blanks, tabs, and newline characters
* Makes it easier to read and understand the program

Program Development (Ch 1.5)

Programming language levels

* There are 4 main groups for the levels of a language:
  + Machine Language
    - This is the lowest level of all
    - Expressed as a series of binary digits
    - All programs compile to this form in order to be executed by the computer
  + Assembly Language
    - Intended to be the next highest level from machine language, but still low level
    - Expressed with mnemonics
    - Easier for programmers to deal with but must first be translated to machine language
  + High Level Language
    - Allows the programmer to ignore the details of machine language
    - This is what most use to write software
    - Expressed as English like phrases
    - Java is high level
    - Also needs to be compiled to machine language
  + 4th Generation
    - Used to refer to languages that operate at a higher level than high level languages
    - Include special facilities for report generation or database interactions
    - Followed after the first 3 generations

Editors Compilers Interpreters

* **Editors:** Used as you type a program’s code. The text is then stored in a file pertaining to that programming language
* **Compiler:** A program that translates code in one language to equivalent code in another language. This is the program that allows your code to be run on the computer because it translates high level code into machine code
* **Interpreter:** This is similar to a compiler but it translates and executes the statements one by one instead of all at once.
  + **Advantage:** Only one phase required for compilation
  + **Disadvantage:** Translation process occurs during each execution and leads to slower runtime
* Java uses a compiler to translate code first into byte-code. The difference between byte-code and machine code is that byte-code does not need any specific processor to execute. This allows for portability.

Development Environments

* More advanced than editors
* Set of tools used to create, test, and modify a program
* **IDE:** Integrate these tools into one program and provide a GUI
* Include an interpreter, compiler, documentation generators, archivers, and a debugger that will help you find errors in your code

Syntax and Semantics

* Languages have rules that dictate how vocabulary should be used
  + Example in Java: Braces are used to begin and end classes and methods
* All syntax rules are checked during compilation
* Semantics define what happens when a statement is executed

Errors

* Programmers are responsible for accuracy and reliability of a program
* 3 kinds of errors in Java:
  + Compile-time error – identified before compilation
  + Run-time error – identified after compilation and causes the program to terminate
  + Logical error
* An example of compiler errors include syntax errors and incompatible data types
* An example of a runtime error is a divide by 0 error
* Logical errors are any errors that cause the program to function normally but not produce the correct result

Object Oriented Programming (Ch 1.6)

Problem Solving

1. Understand the problem
2. Design the solution
3. Consider alternatives and refine the solution
4. Implement the solution
5. Test the solution and fix any problems that exist

Object Oriented Software Principles

* Object – A fundamental element in a program. It has states and behaviors
* Method – Dictates the behavior of the object
* Attribute – The values it stores internally (define the state)
* Class – Definition of an object
* Encapsulation – The idea that an object should protect and manage its own information
* Inheritance – Object’s classes can be created from other classes
* Polymorphism – The idea that we can refer to multiple types of related objects in consistent ways

Character Strings (Ch 2.1)

print and println methods

**print** – display text and data onto a console but does not transition to the next line

**println** – display text and data onto a console and transition to the next line

Concatenation

* String literals cannot span multiple lines
* Allows the programmer chain one string to another
* Done using the + operator

Escape Sequences

* These are special characters that are used in Java that you need to escape with a backslash to indicate that they are part of the string

|  |  |
| --- | --- |
| \b | Backspace |
| \t | Tab |
| \n | New line |
| \r | Carriage return |
| \” | Double quote |
| \’ | Single quote |
| \\ | Backslash |

Variables and Assignment (Ch 2.2)

Variables – Name for a location in memory used to hold a data value

Assignment statement – Assigns a value to a variable

Constants – These values do not change throughout the program and are commonly written in all uppercase letters

Primitive Data Types (Ch 2.3)

Integers and floating points

* Integers are whole numbers and floating points are decimal numbers
* 4 integer types (byte, short, int, long)
* 2 floating point types (float, double)
* Difference between these is the amount of memory used to store the value

Characters

* Characters are basically like strings that have a length of 1
* Can be combined to form strings
* Indicated with a single quote

Booleans

* This is a logical type that is used to indicate a true or a false state
* Cannot be converted to any other data type and no other data types can be converted to a Boolean value

Expressions (Ch 2.4)

Arithmetic Operators

|  |  |
| --- | --- |
| + | Addition |
| - | Subtraction |
| \* | Multiplication |
| / | Division |
| % | Modulus (Remainder) |

* If either or both operands are floating point, the result is a floating point as well
  + **BUT:** If both operands are integers, the division operation will also yield an integer even if they cannot be divided evenly
  + Ex: 10/4 = 2 but 10.0/4 and 10.0/4.0 = 2.5

Operator precedence

* For complex expressions, the order of operations follow the same principles as in mathematics (PEMDAS)
* Multiplication has equal precedence as division and addition has equal precedence as subtraction

Increment and decrement operators

* Adds or subtracts 1 to any integer or float value
* Indicated as ++ or - - (Unary)
* Writing it before or after can yield different results
  + Ex: total = count++; will increment count by 1 and assign the original value of count to total
  + But total = ++count will increment both total and count

Assignment operators

* These are used to combine an operation with an assignment
  + Example : total += 5; is like stating total = total + 5;
* All operators can also be used as assignments

Interactive Programs (Ch 2.6)

Scanner

* Used to get user input
* Can convert a character string from the user into a primitive data type (int, double, float boolean)

The Math Class (Ch 3.5)

* Provide a large number of mathematical functions used to make calculations
* Methods in this class are static, so they can be invoked through the name of a class without instantiation