ITO4131 Java Programming - week1 worksheet

Week 1 Synchronisation Session – Welcome & familiarising with the Tools

Expected runtime 30~40min

Pre-session checklist

- You have access to Ed platform
- You have access to Teams
- You have gone through Lesson 1.1-1.3 and in a process of going through Lesson 1.4

Reminder

- Enable the instant email notification from Ed

Rundown

1. Connect with everyone using MS Teams

   - In the Teams, say Hi to everyone by posting your name and briefing introducing 1~2 movies / TV series you have watched in the past 6 months
     - Feel free to respond to others' posts
   - Send a private message to me - Hi to keep the conversation going.

2. Using workspaces in Ed

   - Create a Java file called "*PrePostOperator.java"* in the workspace shared with you in the Ed platform and include the following code in the file.
   - Open the terminal in the workspace, and combine the knowledge from Lesson 2, compile the file and run the file.

```
public class PrePostOperator
{
   public static void main(String[] args)
   {
      int x = 10;
      int w = (x++ + x++ - (--x - --x * 2)) / 4;
      System.out.println(w);
   }
}
```

3. Using code snippets in Ed discussion

   - Include the following *"TypeConversionTest"* code as code snippet and reply to the question "W1-S1 testing code snippets" in Ed discussion. **Make sure your program can run in the code snippet.**
   - Combining the contents from Lesson 3, discuss how you make sense of the result and include your discussion in the answer.

```
public class TypeConversionTest
{
   public static void main(String[] args)
   {
      System.out.println('a' + 100 + "abc");
   }
}
```

4. Data types in Java
   a. Primitive data types - based on the values below, think about what data type they represent and **post your answers in Teams group channel**
      i. What is the difference between 10.0 and 10.0F, and the difference between 'a' and "a"?

Value	10	10.0	10.0F	'a'	"a"	true	"true"
Data type							

5. Create a Java file in workspace

   Go to the workspace "RolllingProject_XXXX" that is shared with you
   -> Use **terminal command "mkdir w1"** to create a folder called "w1"
   -> then use **terminal command "cd w1"** to change the working directory to "w1" (more details see appendix)
   -> create a Java file "W1S2.java" inside the "w1" folder
   -> copy and paste the following class skeleton into the "W1S2.java" file

```
public class W1S2
{
        public void testDataType()
        {

        }
}
```

6. Using variables and type casting inside the above method testDataType()

    a. Write code to **declare a variable "num" as int type**

    b. Write code to **initialise the "num" variable as 2 in a separate line of code**.

    c. Write code to **declare a variable "strValue" as String type**, and initialise the value as "\u0065"

    d. Write code to **explicit type cast the variable "num" into double format** and pass it to another variable "numDouble"

    e. Use System.out.println to **print out the sum of the two variables "num" and "numDouble"**

    f. Use System.out.println to **print out the variable "strValue"**

7. Write a **main method** to print out the exact result of the following calculation. Then compile and run your program.
    a. the total sum of the numbers 1, 2, 3, 4, 5 and then divided by 2

8. Writing a method named "sumDivideByTwo" with public visibility modifier in the W1S2 class. The method should **accept two parameters of int type**, **add them up and divided by 2**, then **return the exact result in double format**.

9. Writing a method named "sumDivideByTwo" with public visibility modifier in the W1S2 class. The method should **accept two parameters of double type**, **add them up and divided by 2**, and **return the exact results in double format.**
    a. Is this method the same as the above?
    b. What is the difference between this method and the above method? Can you also relate this phenomenon to the System.out.println method in task 3?

10. Write the method header for the following part of method body

```
{
        return (double) num1 * num2;
}
```

Note: Please also download W1 folder to your own machine

11. OLA to discuss code challenge in Lesson 1.5

12. OLA to discuss the concept of class, constructor, accessor, mutator.

Extra exercises if you have more capacity. Once you have finished task 13-14, post your codes as code snippets or screenshot and reply to the question “W1extra exercises” in Ed discussion.

13. Can you make the following program compile? Post what you changed and the corrected code as code snippet in Ed discussion thread

```
public class TestMinusOne {
    public static int minusOne(int number) {
        return number - 1;
    }

    public static void callMinusOne() {
        System.out.println(minusOne(4));
        System.out.println(minusOne("hello"));
    }

    public static void main(String[] args) {
        System.out.println("Running the code...");
        callMinusOne();
    }
}
```

14. Fill in the result for each line of calculation in the table, take a screenshot of the table and post it in Ed discussion thread

```
public class Variable3 {
    public static void main(String[] args) {
        System.out.println("Start");
        int x, y = 32;
        x = y % 3;
        int z = (x+y)/6;

        x = --z + ++y;
        z += (y = ++x);
        y = x%3 + ++z/5;
        z /= z;
        x = ++z;
        y = x--;

        z += ++x + ++y;
        z %= 2;
        x %= z;
    }

}
```

Statement	Value of x	Value of y	Value of z
int x, y = 32;			
x = y % 3;			
int z = (x+y)/6;			
x = --z + ++y;			
z += (y = ++x);			
y = x%3 + ++z/5;			
z /= z;			
x = ++z;			
y = x--;			
z += ++x + ++y;			
z %= 2;			
x %= z;			

Appendix

Basic terminal commands

- To create a folder, use “mkdir” command, followed by the new folder name. Example below to create a folder named “newfolder” under the current directory.

mkdir newfolder

- To change the current directory, use “cd” command, followed by the folder directory you want to use. Example below to change the current directory into a sub-folder named “newfolder”

cd newfolder

- Another example below to change the current directory to the parent folder, where the “..” (two dots) represents the parent directory.

cd ..

- Another example below to change to the current directory to your home directory “/home”, where the “~” (tilde) represents your home directory.

cd ~

Method header recap

A method header is defined as follows:

(Visibility modifier) (return type) (method name) ([optional parameters])

For Example:

public void sampleMethod()

Publicly accessible method which returns nothing.

public void sampleMethod(int inputA, int inputB)

Publicly accessible method which returns nothing but accepts two inputs

private String sampleMethod(int inputA, int inputB, String inputC)

Private method which returns a String as an output and accepts three inputs

private Student sampleMethod()

Private method which returns a Student object as an output and accepts no inputs

Visibility Modifier

Can private or public or protected – Determines WHO can see the method.

Return Type

Can be void (i.e returns nothing) or must be one of the following: int, boolean, char, float, double, String, or any other Class Object Type – Provides a response to the calling object

Method Names

CANNOT have the same signature – Allows Java to know which method to execute.

Parameters

Can potentially accept any number of parameters as required – Allows Java to pass values to a method in order for it to perform its task.