COMP1008 Introduction to OOP using Java
Assignment#1 – The Essentials

Important Notes:

- **This assignment must be done by each group individually. Do not share your work with other groups**
- The assignment should be **submitted** through the **Blackboard** only **before the due date** in order not to lose marks
- Follow the **naming rules** and conventions for the assignment files as explained in the instructions
- Although Java code can be written, compiled, and run using different **IDEs (Integrated Development Environments)**, based on what we have been using in our class, it is recommended to answer the assignment using **Microsoft Visual Studio Code**. However, **you are allowed to use any other IDE (Integrated Development Environment) you prefer**.
- **At the end of the assignment** (after answering all the questions), you should have a:
- **folder named "Java-Assignment1"** which contains these **two java files** inside the application package folder:
  - **AnyDescriptiveName.java => The utility class that contains the required properties and methods**
  - **AnyDescriptiveName.java => The main class that contains the "main()" method to run the application**
- Submit your assignment files through the **Blackboard** official Assignment Link **(Emails are NOT accepted):**
- **Submit/upload** only the following files as explained below **(Zipping/Compressing the folder is NOT allowed)**:
  - **The two Java files *(Not the entire folder)*:**
    - The java main file for the application *(The entry point to launch the application)* ➔ **AnyName.java**
    - The java file for your custom (utility) class ➔ **AnyName.java**
  - **The PDF file that contains the screenshots/snapshots as explained below:**
    - Use **MS-Word** to capture all the assignment images (screenshots), put them all in one MS Word Document, then finally convert it to a PDF file and upload it with your submission.
    - Please consider the following screenshots (images):
      - The folder structure of your project inside Visual Studio Code or any other Editor/IDE that you are using that also shows the main java file
      - The Output of your code (after running the main Java application file) in the Console Window/Terminal for any IDE you are using
- **Online:** Each student must be ready with their Zoom Mic/Cam and Screen share feature to quickly show us the assignment content, Q/A, and run it to see the output (even if it is not working properly).
- **In-person:** I will make a tour to check the running application in the student's computer and asking the technical questions

***NOTE:** This presentation will be very quick without correction or feedback in order not to take the full lecture time! The assignment steps, feedback, and solutions will be completely explained to all students (after finishing all the presentations) in the next lecture*

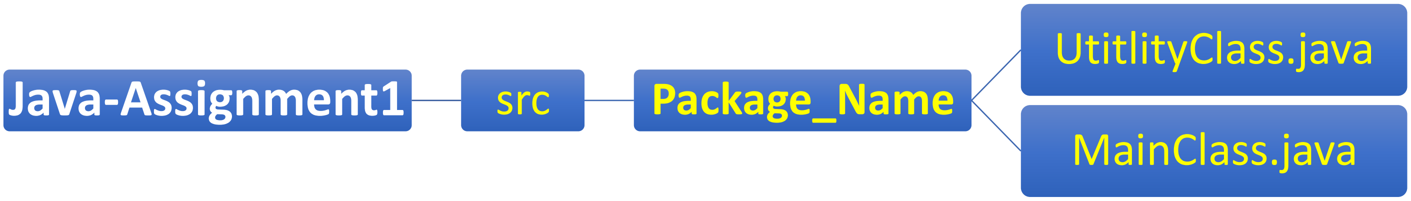
(20 Marks)

Assignment Contents:

This assignment will be a full demonstration of your skills in using:

- VS Code to setup a Java project *(or any other IDE you prefer)*
- Packages that contain Classes with multiple members (attributes and methods/behaviours)
- Constructors and Instances of classes (objects)
- Static Members
- Encapsulation with private members
- Conditional Structures with different paths to solve a problem
- Math and Formulas

Folder Structure and Contents:

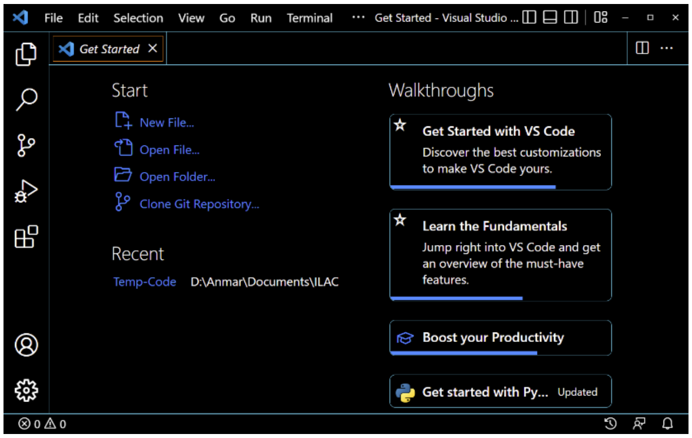


NOTE: Package and Classes Names shuld be any oher descriptive names you prefer

Assignment Instructions:

1. Using VS Code (or any other editor), you will create a new project named **"Java-Assignment1"** after choosing the folder location

   a. Open/run VS Code

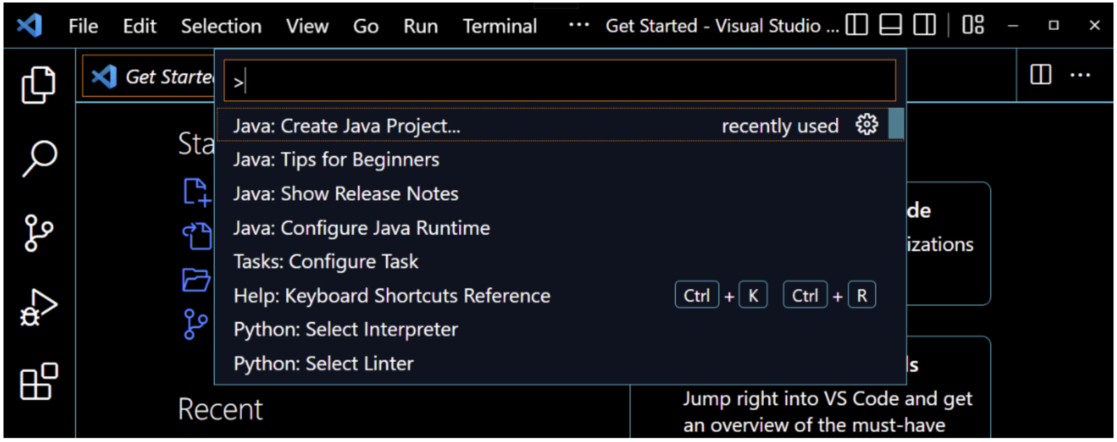


Discover the best customizations to make VS Code yours.

Jump right into VS Code and get an overview of the must-have features.

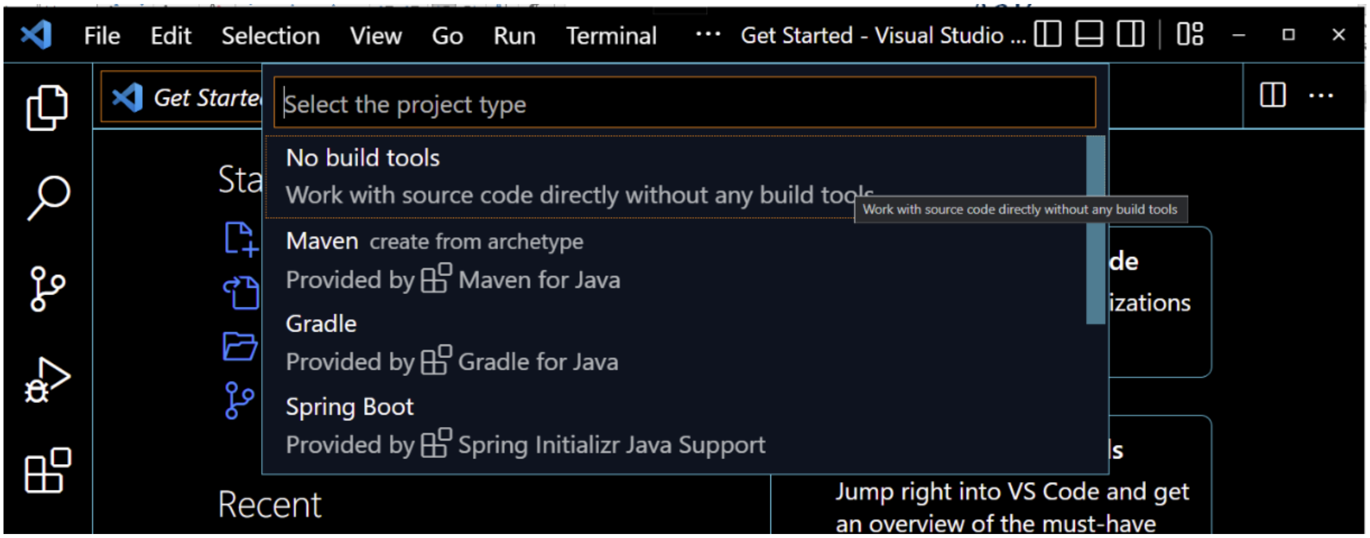
   b. CTRL + SHIFT + P (Windows Hotkeys) ➔ to open the command panel ➔ Then Select:

   - > **Java: Create Java Project**


File Edit Selection View Go Run Terminal ··· Get Started - Visual Studio ...
Get Starte
>|
Java: Create Java Project... recently used
Java: Tips for Beginners
Java: Show Release Notes
Java: Configure Java Runtime
Tasks: Configure Task
Help: Keyboard Shortcuts Reference Ctrl + K Ctrl + R
Python: Select Interpreter
Python: Select Linter
Sta
Recent
de
izations
s

Jump right into VS Code and get an overview of the must-have

   - Then select **"No Build tools":**


File Edit Selection View Go Run Terminal ··· Get Started - Visual Studio ...
Get Starte
Select the project type
No build tools
Maven create from archetype
Provided by Maven for Java
Gradle
Provided by Gradle for Java
Spring Boot
Provided by Spring Initializr Java Support
Sta
Recent
de
izations
s

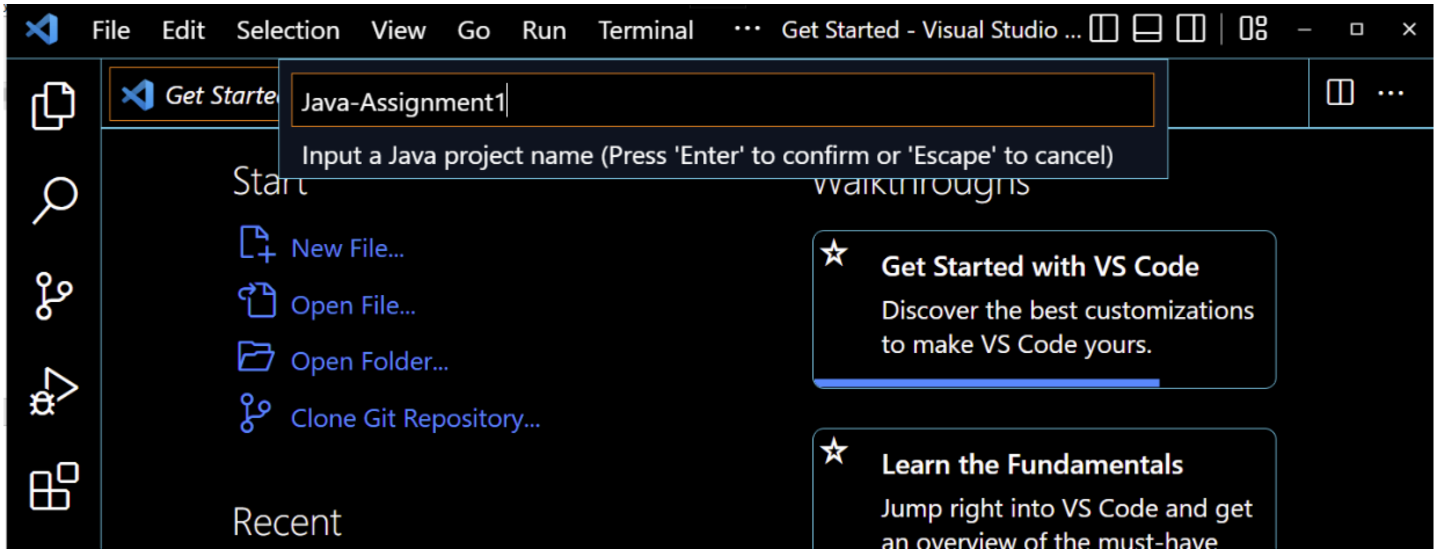
Work with source code directly without any build tools

Work with source code directly without any build tools

Jump right into VS Code and get an overview of the must-have

Then VS Code will open the **“Select Folder”** windows explorer to select the main folder where you want to create and build this assignment project folder. (It should be inside your current course, Georgian Java Semester 2 folder, or any other location you prefer)

c. After selecting the folder, **type the project name** in the box as shown below. Notice that the name you type inside the box **“Java-Assignment1”**, based on the instruction, will be the name of that actual folder of the assignment in your hard drive or your local drive. Press **“Enter”** key to confirm the name for your Java project/folder:



Java-Assignment1

Input a Java project name (Press 'Enter' to confirm or 'Escape' to cancel)

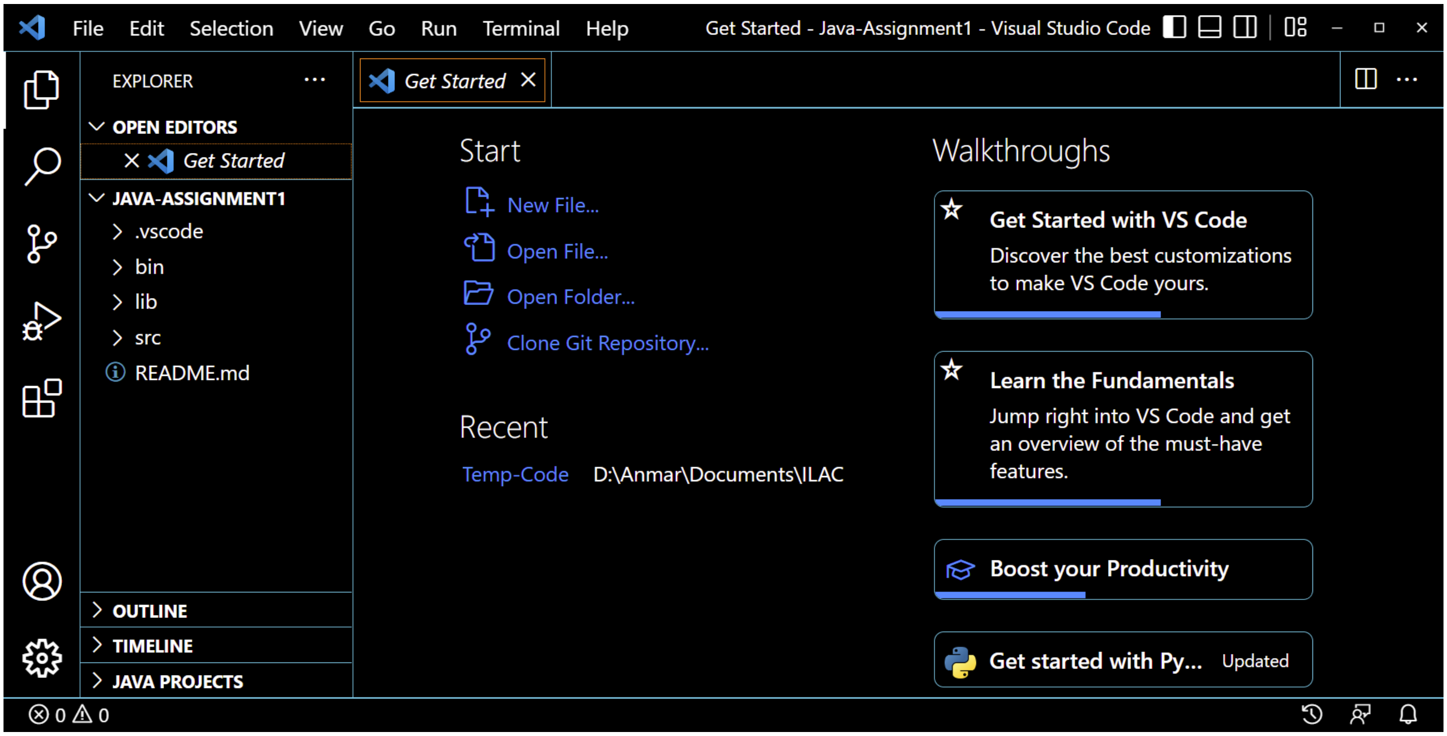
Get Started with VS Code

Discover the best customizations to make VS Code yours.

Learn the Fundamentals

Jump right into VS Code and get an overview of the must-have

d. After finishing these initial steps successfully, you should see your Assignment project “JAVA-ASSIGNMENT1) with the basic default content (folders and files) as shown below:



Get Started with VS Code

Discover the best customizations to make VS Code yours.

Learn the Fundamentals

Jump right into VS Code and get an overview of the must-have features.

Part 1: Convert.java Class

1. Create a **new package** with any name you prefer. I will use for example **“p1_converting”** just to demonstrate the step. This package/folder will contain the “Part1 Convert Java Code Files”.

   NOTES:

   - A package in Java Project is simply just a folder in Windows/Mac
   - Avoid using the **“.”** Symbol in the package name because it means that there is more than one package (folder) **[Please refer to my lecture and code if you need more details]**
     - **Examples:**
       - **“p1_converting”** ➔ one package/folder named “p1_converting”
       - **“p1.converting”** ➔ two packages/folders:
         - “p1” is the first package/folder
         - “converting” is the second package/folder which is inside “p1”

2. Create a new class (java) file with any name you prefer. I will use the name **“Convert”** just to demonstrate the steps clearly. This is just a utility class that will be used/called by the main class later to run your application, so **we do not add the “main()” method inside this class**. Now all your code starting from the following **point/question (3) will be written inside this “Convert” class**.

3. Create two class fields/variables/properties (without values):
   a) A field/variable named “**massResult**” of “**double**” data type and make it “**static**”
   b) A field/variable named “**tempResult**” of “**double**” data type and make it “**static**”

4. Create a method named “**getKilogramValue**()”. This method will be used to convert a mass (aka weight) in pound (lb) unit to kilogram (kg) unit:
   a) Make it “**static**” and “**private**”, with a returning value of “**double**” data type
   b) It accepts **one parameter** of “**double**” data type ➔ represents the mass value in pound
   c) It returns the result of the formula for converting the passing value which is in pound to kilogram

5. Create a method named “**getPoundValue**()”. This method will be used to convert a mass in kilogram (kg) unit to pound (lb) unit:
   a) Make it “**static**” and “**private**”, with a returning value of “**double**” data type
   b) It accepts **one parameter** of “**double**” data type ➔ represents the mass value in kilogram
   c) It returns the result of the formula for converting the passing value which is in kilogram to pound

   HINTS for Points 4 and 5:

   - **The approximation for converting kilograms (kg) to pounds (lb) is 1 kg = 2.2 lb.**
   - **To convert from kilogram to pound ➔ multiply (kg) value by 2.2**
   - **To convert from pound to kilogram ➔ divide (lb) value by 2.2**

6. Create a method named "**getFahrenheitValue** ()". This method will be used to convert a temperature value in Celsius unit to Fahrenheit unit:
   a) Make it "**static**" and "**private**", with a returning value of "**double**" data type
   b) It accepts **one parameter** of "**double**" data type ➔ represents the temp value in Celsius
   c) It returns the result of the formula for converting the passing value which is in Cel to Fah

7. Create a method named "**getCelsiusValue** ()". This method will be used to convert a temperature value in Fahrenheit unit to Celsius unit:
   d) Make it "**static**" and "**private**", with a returning value of "**double**" data type
   e) It accepts **one parameter** of "**double**" data type ➔ represents the temp value in Fahrenheit
   f) It returns the result of the formula for converting the passing value which is in Fah to Cel

HINTS for Points 6 and 7:

- **Celsius to Fahrenheit Formula Example: (12°C × 9/5) + 32 = 53.6°F**
  - **The formula in programming: value in F = (value in C * 9/5) + 32**
- **Fahrenheit to Celsius Formula: (12°F – 32) × 5/9 = -11.11°C**
  - **The formula in programming: value in C = (value in F - 32) * 5/9**

NOTES AND HINTS:

❖ By the end of answering these questions: 4, 5, 6, and 7 you will have these four methods:

getKilogramValue() => Converts Pound to Kilogram

getPoundValue() => Converts Kilogram to Pound

getFahrenheitValue() => Converts Celsius to Fahrenheit

getCelsiusValue() => Converts Fahernheit to Celsius

❖ Each method from all the list above has these same features:
- ➢ Static Member **"static"**: it belongs to the class, not to an instance (object)
- ➢ Access Modifier: **"private"** to be used/called within the class itself only
- ➢ **Accepts** a **double** value (argument) and **returns** a **double** data type value
- ➢ The returned value is the result of the formulas for converting a unit into another one

❖ Notice that all these methods are declared to be:
- ➢ **"static"** ➔ to be called inside another static methods, for this reason, they must be "static", otherwise, Java compiler will throw this error: **[Cannot make a static reference to the non-static method *methodName(double)* from the type Convert]**
- ➢ **"private"** ➔ We don't want to make them public to be used outside the class through its instances/objects which to emphasis the idea of "Encapsulation" in OOP

- ❖ Every pair of related methods will be called by a **public static method** that you will create in the coming questions as shown below:
  - ➢ "**getKilogramValue**()" & "**getPoundValue**()" are private methods:
    - i. will be called inside "**getMass**()" public **method**
  - ➢ "**getFahrenheit**()" & "**getCelsiusValue**()" are private methods:
    - i. will be called inside "**getTemperature**()" public **method**

  In the next questions (8) and (7), you will create these two methods: **getMass() & getTemperature()**

8. Create a method named "**getTemperature**()". This method will be used to convert between Celsius and Fahrenheit temperatures:
   - g) Make it "**static**" and "**public**", with a returning value of "**double**" data type
   - h) It accepts **two parameters**:
     - ▪ **First parameter** of "**double**" data type ➔ the temperature value
     - ▪ **Second parameter** of "**integer**" data type ➔ the user's choice either 1 or 2
       - a. If the passing value (argument) is 1 ➔ converting Cel to Fah
       - b. If the passing value (argument) is 2 ➔ converting Fah to Cel
   - i) You can use any type of conditional structure (if or switch) to check the choice variable value:
     - a. If the "**choice**" value is **1** ➔ you will call "**getFahrenheit**()" method and assign the result (returned value) of this method to the class variable (field) "**tempResult**"
     - b. If the "**choice**" value is **2** ➔ you will call "**getCelsius**()" method and assign the result (returned value) of this method to the class variable (field) "**tempResult**"
     - c. Otherwise (if the choice is neither 1 nor 2):
       - i. Print and error message (using **System.err**) ➔ "Invalid Choice! Only 1 or 2 is allowed"
       - ii. Assign the value of 0 to the "tempResult"
   - j) At the end, outside the conditional block, you will need to **return the value of the variable "tempResult"** ➔ This method has to have a numeric return data type

9. Create a method named "**getMass**()". This method will be used to convert between Pound and Kilogram mass (aka weight) unit:
   k) Make it "**static**" and "**public**", with a returning value of "**double**" data type
   l) It accepts **two parameters**:
      - **First parameter** of "**double**" data type ➔ the mass (aka weight) value
      - **Second parameter** of "**integer**" data type ➔ the user's choice either 1 or 2
         a. If the passing value (argument) is 1 ➔ converting Pound to Kilogram
         b. If the passing value (argument) is 2 ➔ converting Kilogram to Pound
   m) You can use any type of conditional structure (if or switch) to check the choice variable value:
      a. If the "**choice**" value is **1** ➔ you will call "**getKilogram**()" method and assign the result (returned value) of this method to the class variable (field) "**massResult**"
      b. If the "**choice**" value is **2** ➔ you will call "**getPound** ()" method and assign the result (returned value) of this method to the class variable (field) "**massResult**"
      c. Otherwise (if the choice is neither 1 nor 2):
         i. Print and error message (using **System.err**) ➔ "Invalid Choice! Only 1 or 2 is allowed"
         ii. Assign the value of 0 to the "massResult"
   n) At the end, outside the conditional block, you will need to **return the value of the variable "massResult"** ➔ this method has to have a numeric return data type

(50 Marks)

Part 2: ConvertTest.java Class

1. Create a new class file (Java file) named for example **“ConvertTest”** or any other name you prefer**.** This class will be used to run/launch your Java project so it **will contain the “main()” method as an entry point to run your application**. Now all your code starting from the following question (2) will be written **inside the main() method** of this “**ConvertTest**” **class**.

2. Ask the user to choose between either **Temperature Converting** or **Mass Converting:**
   Example: Enter '1' for weight converting, or '2' for temperature converting:
   Then save the user’s input “1” or “2” into an integer variable with any descriptive name you choose

3. Using “**if**” statement or “**switch**” statement condition to check:
   a) If the user input is **1** ➔ go to the if/case block to **calculate the wanted mass/weight value**
   b) If the user input is **2** ➔ go to the if/case block to **calculate the wanted temperature value**
   c) Else/Default output any simple message to show that the user’s input was invalid
   In the next **questions (4) and (5)**, you will write the full conditional structure code for calculating mass and the temperature based on the user’s choice (input)

4. Inside the **conditional block** for calculating the wanted **mass/weight** value:
   a) Ask the user to input a temperature value ➔ **Example**: **Enter a temperature value:**
   Then save the user’s input into a variable with any descriptive name you choose
   b) Ask the user again to choose between converting LB to KG or KG to LB
   Example: Enter '1' for converting LB to KG, or '2' for converting KG to LB:
   Then save the user’s input “1” or “2” into a variable with any descriptive name you choose
   c) Using “if” statement or “switch” statement condition to check:
      i. **If the user option is 1** you will go to the if/case block to **call the static public method “getMass”** and passing the required arguments for the required method’s parameter
      Hint:
         - You will pass the temperature value that you receive from the user’s input
         - You will need to pass the option value of 1 for converting LB to KG
      ii. Display the result in a nice format:
      Example: 33 LB is 15 KB
      iii. **If the user option is 2** you will go to the if/case block to **call the static public method “getMass”** and passing the required arguments for the required method’s parameter
      Hint:
         - You will pass the temperature value that you receive from the user’s input
         - You will pass the option value of 2 for converting KG to LB
      iv. Display the result in a nice format: ➔ **Example: 15 KB is 33 LB**
      v. **If the user option is something else** (neither 1 nor 2) ➔ go to the else/default block to display a warning message to notify the user about entering an invalid option

NOTES AND HINTS:

- To make your code easier to read and maintain, it's better to avoid using too many nested if conditions.
- In the previous questions (3) and (4):
  - You have first conditional structure block to check what type/kind of conversion: weight or temperature conversion
  - Then you have a second conditional structure block to check from which unit to which unit you want to convert:
    - lb to kg or kg to lb
    - C to F or F to C
  - With too many conditions like in this example, it's recommended for example to use "if" and "switch" statement, and it's up to you whether you prefer to have an "if" block first and place switch inside it or the other way
- Refer to the "Code Hint" in the next page for more clarifications and better understanding
- **These are the general guidelines to answer/solve this problem, you are free to follow any other logic you prefer since your logic will solve the same problem**

5. Inside the **conditional block** for calculating the wanted **temperature value**, repeat the same logic/idea as you have written in point/question (4) but this time you will calculate the temperature instead of the mass/weight, you will also have the option 1 for getting Fahrenheit temperature (instead of Kilogram) and the option 2 for Celsius temperature (instead of Pound)

Code Hints about mixing if statement and switch statement:

```
// Ask the user:
// to enter '1' for weight converting, or '2' for temperature converting:
// Save the user's choice/input:
// Main if Block:
If (choice is 1) {
            // this choice for calculating the mass (weight)
            // Ask the user to input a mass/weight value
            // Ask the user to input 1 for... or 2 for...

            /*
             * Build the switch statement with 3 cases:
             * Case 1 => for getting the KG value
             * Case 2 => for getting the LB value
             * Default => Output a warning/error message
             */
        } else if (choice is 2) {
            // this choice for calculating the temperature
            // Ask the user to input a temperature value
            // Ask the user to input 1 for... or 2 for...

            /*
             * Build the switch statement with 3 cases:
             * Case 1 => for getting the F temp value
             * Case 2 => for getting the C temp value
             * Default => Output a warning/error message
             */
        } else {
            // output a simple message to show that the user’s input was invalid
        }
```

(30 Marks)

Happy Coding ☺