School Computing, Faculty of Engineering
Universiti Teknologi Malaysia

SCSJ1023 Programming Technique II
Semester 1, 2018/2019

Exercise 1
Class Constructors and Destructor

Duration : 60 minutes (including submission)

Exercise Materials

- Program templates are provided for this exercise. Please download from the elearning and extract the ZIP file to your local drive.
- The questions and tasks to be completed are stated in the program templates.
- You have the choice to use Microsoft **VS Code**, **Dev C++** or any other IDEs to write the code for this exercise.

Deliverable Item

- Submit only the s**ource code** file you have edited (i.e., **exercise1.cpp**)
- The submission must be done at elearning. Other than that, e.g., email, telegram, etc. will not be entertained.

Plagiarism Policy

- Discussions among the students are still possible during the exercise session.
- However, all works must be done individually.
- Any kind of plagiarism (e.g., copy and paste code by any mean) would lead to disqualification of submissions for both parties (i.e., students that copy others' code and students that give their code to others).

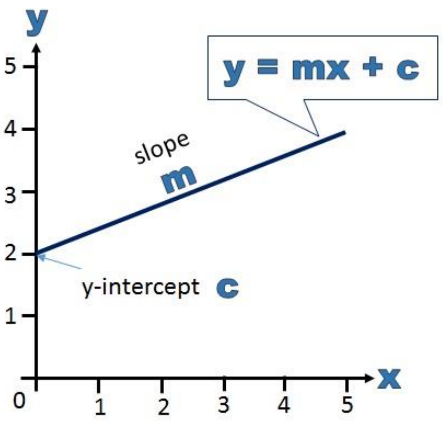
Late Submission Policy

- **10% deduction for every 5 minutes late**.
- For example: if the duration of the exercise is 60 minutes, and your submission is received on the 61st minute, you are only eligible to earn 90% at maximum of the total marks.

Question

In this exercise, you will be defining a class named `Line` to model a straight line equation. Each line can be expressed by the equation, **y = mx + c**

where *m* is the slope of the line and *c* is the y-intercept, i.e. the y coordinate of the location where the line crosses the *y*-axis (see the following figure)



Complete the program **exercise1.cpp** according to the following tasks:

Task 1:

Declare and define the class `Line`. You can use either inline or separation style for these tasks. Note that, the attributes for the class have been given in the class declaration.

Add the following methods to the class:

a. A default constructor which sets all the attributes to zero values.

b. A constructor that accepts two parameters to set the attributes *m* and *c*, respectively.

c. A constructor that accepts four parameters which are **`x1`**, **`y1`**, **`x2`**, and **`y2`**, representing the coordinates x and y of two points that the line passes through. The slope and y-intercept of the line are then calculated as follows:

$$m = \frac{y2 - y1}{x2 - x1}$$

$$c = y1 - mx1$$

d. A destructor which shows a message, for example **"The object is being destroyed"**.

e. Accessor (or getter) methods for all the attributes,.i.e. `getM()` and `getC()`

f. Mutator (or setter) methods for all the attributes.

g. A method named `printEquation()` that print the equation of the straight line. There are three types depending on the value of the y-intercept, *c*:

- `Eq.  y=mx` *(if the slope is zero)*
- `Eq.  y=mx-c` *(if the slope is negative)*
- `Eq.  y=mx+c` *(if the slope is positive)*

Task 2:

a. In the function `main()`, create a `Line` object named `line1` using an appropriate constructor to model the line: **y=2x-5**

b. and print the equation onto the screen using an appropriate method.

Task 3:

a. Create another `Line` object named `line2` using an appropriate constructor to model a line that passes through these two points: **(0,0)** and **(2,8)**.

b. and print the equation onto the screen.

Task 4:

a. Another `Line` object named `line3` has been created using the default constructor (already given in the code). Complete this task such that the slope, *m* and y-intercept, *c* of the line are specified with the values entered from the keyboard.

b. and print the equation onto the screen.

Output:

Expected result of the program is as shown in the following figure. **Bold** text indicates keyboard inputs.

```
Eq.  y=2x-5

Eq.  y=4x

Enter the slope and y-intercept of the line,  m  c => 3 10

Eq.  y=3x+10
```