King Fahd University of Petroleum and Minerals

###### Information and Computer Science Department

**ICS 102: Introduction to Computer Programming**

**Summer Semester 2019-2020 (Term 191)
Homework #2
[Posted: Sunday November 3rd 2019]
[Due Date: Saturday November 16, 2019 @ 11:59 PM (Midnight)]**

**Submission Guidelines:**

Submit a .java file containing the following files:

* **HW2\_XXXXXXXXX\_YourFamilyName**.java (Java source file) containing your program in java for the problem described here.

PLEASE DO NOT INCLUDE .class FILES IN YOUR SUBMISSION

where:

XXXXXXXXX is your 9 digit KFUPM ID.

YourFamilyName is your family name

Submission should be made through your ICS 102 Blackboard course page under **Assignments** submission link.

**Important Notes:**

* **Cheating is taken seriously**. Any cheating attempt will result in an F grade in the course.
* **EACH STUDENT IS REQUIRED TO DO THE HOMEWORK ALONE**. COPYING FROM ANY SOURCE IS REGARDED AS CHEATING.
* **Submission link will be available until 9:00am for late submission without penalty.**
* **Submissions via email are not accepted and will be simply ignored**.
* Submission of the homework solution should be in a zipped filed with the format specified above. **Any different formatting/naming will result in reducing the total homework score by half!**
* **You must use proper indentation and meaningful variable names in your programs.**

**Background:**

**Newton–Raphson method** is a method for finding better approximations to the roots (or zeroes) of a real-valued function.

A brief explanation of the Newton Raphson method is given in the algorithm below. For more details refer to the references given at the end of this document and see reference [1]

**Algorithm**:

|  |
| --- |
| For a continuous function *f*(*x*):**Step 1**: Choose an initial estimate, *xn* |
| **Step 2**: Compute *f*(*xn*) and *f’*(*xn*). Where *f’*(*xn*) is the first derivative of *f*(*xn*) |
| **Step 3**: Compute the new estimate:  |
| $$x\_{n+1}=x\_{n}-\frac{f(x\_{n})}{f'(x\_{n})}$$ |
| **Step 4:**Repeat steps 2 and 3**,** until |$x\_{n+1}- x\_{n}$| ≤ Ɛ**Note**: - Ɛ is a resolution that must be set by the user.  In the testing phase, this value may be set to 10-6. - The values $x\_{n} $and $x\_{n+1}$are respectively the previous and current values of the solution $x\_{ }$. |

 **Problem Statement***:*

The goal of this programming exercise is to write a java program to find the zero of a mathematical function f(x) around a given point x0, using the Newton Raphson method. We consider f(x) as a polynomial function $P\left(x\right)$ of degree *n,* such that *n* is less than or equal to 3.

$P\left(x\right)=a\_{3} x^{3 }+a\_{2} x^{2 }+a\_{1} x+a\_{0}$ (1)

The overall program MUST have the following 2 methods:

1. A java method **evalFun** (….), that computes P(x) for a given x. This method **evalFun()**, takes as input:
	* + the degree of the polynomial n, where n is an integer n ≤ 3,
		+ the real coefficients a­­3­, a­­2, a1­, a­0­ and x (all of type double), such that some coefficients may be equal to zero.

The method should return the value of P(x).

Note that since we are dealing with a polynomial function in equation (1), the same function code **evalFun()** can be used to evaluate both f(x) and its derivative f’(x) in equation (2), below:

$f'\left(x\right)=3a\_{3} x^{2 }+2a\_{2} x^{ }+a\_{1} $ (2)

1. Method **newtonRaphsonMethod**() is to find the zero of the polynomial function f(x) given an initial value **x0** using the Newton Raphson algorithm. This method (**newtonRaphsonMethod ()**) should have as input
	* the estimated initial value **x0**, and
	* the accuracy Ɛ.

Your program should use the private global variables maximumIterations, the three co-efficient (e.g., a0,a1,a2,a3) and the root of zero. Your method should change the values of **the root** of the function, and the **number of iterations required** to find this root. The number of iterations is number of times the loop has run. A condition should be set on the number of iterations such that if the number of iterations exceeds, the loop should stop and return. The program should check the value returned by the method and decides whether the result is valid or not.

**Requirements**:

* Your program should be general to handle all types of polynomial functions with degree n ≤ 3.
* For invalid input, the program should display an error message and prompts the user to enter a new choice or value again.
* The program should quit only when the user requests it.
* To make the polynomial coefficients (a3, a2, a1, a0) known in all functions they should be declared as ***global variables (static variables).***
* The main method should consist only of methods calls (you should not write long code in the main method).

**Sample Program Run**:

