FACULTY OF COMPUTERS AND ARTIFICIAL INTELLIGENCE, CAIRO UNIVERSITY

CS213: Programming II
Year 2024-2025

First Semester

Assignment 1 - Version 3.0

Course Instructors:

Dr. Mohammad El-Ramly

Revision History

Version 1.0	By Dr Mohammed El-Ramly	26 Oct.2022	Main Doc
Version 2.0	By Dr Mohammed El-Ramly	26 Oct.2023	Updated for 22/23
Version 3.0	By Dr Mohammed El-Ramly	2 Oct.2024	Updated for 24/25

Objectives

This assignment aims to help you learn OOP concepts in C++ and how to use OOP modeling and design with C++ to build systems with intermediate complexity.

Instructions

1. **Your Learning Is YOUR Responsibility. أنت المسؤول عن تعلمك و عن نجاحك**
2. **Deadline for part 1 is 8 of Oct 2024 @ 11:00 pm (Tasks 1)**
3. **Deadline for part 2 is 15 of Oct 2024 @ 11:00 pm (Tasks 2 and 3)**
4. **Deadline for part 3 is 25 of Oct 2024 @ 11:00 pm (Task 4 and bonus)**
5. Students will forms teams of **three** students **from the same lab/section**.
6. Please submit **only work that you did yourself**. If you copy work from your friend or book the net or AI **you will fail the course, unless permission is given.**
7. **I use plagiarism tools to discover copied solutions and AI tools for AI-generated ones.**

Task 0 (0 marks)

1. Review OOP C++ concepts and syntax.
2. Review the slides and code examples uploaded in the class and understand them very well.
3. Read this quick tutorial https://www.codesdope.com/cpp-introduction/
4. Create a **private GitHub** repo for the project. **Use it for development.** Everyone should use it.

Task 1 (3 marks - Individual Work) - C++ Concepts and Problem Solving

Students will individually solve the problem in Sheet 1 individually. Student with the smallest ID will solve problems 1, 4, 7, 10. The next will solve 2, 5, 8, 11. And the student with the largest ID will solve problems 3, 6, 9, 12. Code should be in **standard C++** not using third-party libraries. **The program should have a menu that tells the user to choose which problem to solve.** You will uploaded **one .cpp** file in standard C++**. DO NOT USE AI.**

Each file will be named **A1_T1_SectionNum_StudentID.cpp.** Each file must have header :

```
// File: ......... .cpp
// Purpose: .........
// Author: ....................
// Section: ....................
// ID: ....................
// TA: ...................
// Date: 8 Oct 2024
// Detailed explanation of the file and how program works ...
```

Task 2 (2 marks - Group) - Classes, objects & operator overloading

In this assignment, you will develop a **`Polynomial`** class in C++. An object of this class represents a polynomial of arbitrary degree **using any AI assistant**. The public interface of your class will be as in this header file. You will develop **30 test cases** and write a program to run the test cases and see if AI did any mistakes. Then you will repeat with **a different AI** assistant. And **run test cases** again. **You will write a detailed report comparing the two AI** and explaining the difference between their code quality and which one did better and give them % from 100% and explain why. Report must be pdf and include the 2 codes and detailed comparison. Also zip cpp. Name file **A1_T2_T3_SectionNum_StudentID1_Id2_Id3.pdf** & **A1_T2_SecNum_Id1_Id2_Id3.zip** for code

```
#ifndef POLYNOMIAL_H
#define POLYNOMIAL_H

#include <iostream>
#include <vector>
#include <cmath>
#include <algorithm>

using namespace std;

class Polynomial {
private:
    vector<double> coeffs; // Store coefficients of the polynomial

public:
    // Constructors
    Polynomial();
    Polynomial(const vector<double>& coefficients);
    Polynomial(const Polynomial& other);

    // Destructor
    ~Polynomial();

    // Assignment operator
    Polynomial& operator=(const Polynomial& other);

    // Arithmetic operators
    Polynomial operator+(const Polynomial& other) const;
    Polynomial operator-(const Polynomial& other) const;
    Polynomial operator*(const Polynomial& other) const;

    // Equality operator
    bool operator==(const Polynomial& other) const;

    // Output operator
    friend ostream& operator<<(ostream& out, const Polynomial& poly);

    // Utility functions
    int degree() const; // Return the degree of the polynomial
    double evaluate(double x) const; // Evaluate the polynomial at x
    double evaluate(double x) const; // Evaluate the polynomial at x
    Polynomial compose(const Polynomial& q) const; // Composition
    Polynomial derivative() const; // Derivative of the polynomial
    Polynomial integral() const; // Return a polynomial of integration
    double integral(double x1, double x2) const; // Intergate from x1 to x2
    double getRoot(double guess=1, double tolerance=1e-6, int maxIter=100);
    void setCoefficients(const vector<double>& coefficients);
                      // Set coefficients
    double getCoefficient(int degree) const;
                      // Get coefficient of a specific degree
};
#endif // POLYNOMIAL_H
```

Example of using this class:

```
#include "Polynomial.h"

int main() {
    Polynomial p1({1, -2, 3}); // Represents 1 - 2x + 3x^2
    Polynomial p2({-1, 4});    // Represents -1 + 4x

    Polynomial sum = p1 + p2;
    Polynomial difference = p1 - p2;
    Polynomial product = p1 * p2;

    cout << "p1: " << p1 << endl;
    cout << "p2: " << p2 << endl;
    cout << "p1 + p2: " << sum << endl;
    cout << "p1 - p2: " << difference << endl;
    cout << "p1 * p2: " << product << endl;

    double x = 2.5;
    cout << "p1 evaluated at x = " << x << ": " << p1.evaluate(x) << endl;
    cout << "p2 evaluated at x = " << x << ": " << p2.evaluate(x) << endl;

    return 0;
}
```

Please note the following

- **See this** https://www.geeksforgeeks.org/copy-constructor-vs-assignment-operator-in-c/
- **See** https://ecomputernotes.com/cpp/classes-in-c/returning-object-from-function
- Team should help each other and support in each.
- Team should integrate the work together and submit one working report.
- Team should test the entire program and make sure that it works correctly in full.
- All team members **must fully understand all parts of the program.**

Task 3 (1 mark - Group / Individual) - Technology Training Opportunities

Different agencies and organization in Egypt and outside of**fer free technology training.**

In Egypt, this includes:

1. Information Technology Institute (ITI) 1month and 9 months programs
2. National Telecommunications Institute (NTI) 3 months programs and other programs
3. Digital Egypt Pioneers Initiative مبادرة رواد مصر الرقمية

There are also training opportunities like freeCodeCamp.com

The team will review in details three possible training opportunities, one of them outside Egypt.

They will write on integrated report about these three chances. For each one, they will write:
1- Who offers it and where it is located
2- What it offers
3- When it opens and how long it is and online or offline
4- The conditions for application and acceptance and the fees and conditions if any
5- The different learning track
6- Each member will select a track that he likes and will write why he wants this one.
7- Team will submit this task with T2 in file **A1_T2_T3_SectionNum_StudentID1_Id2_Id3.pdf**

Task 4 (4 marks - Group) - Classes, objects, abstraction and composition

Task 4.1 (0 marks - Individual) Review Vole Machine

Refer to CS111 materials and "CS an Overview Book" to review the Vole machine and language. And see the video in classroom.

Task 4.2 (0 marks - Group) Machine Simulator Design

Model and design a simulator for the Vole machine and its language that is capable of simulating its operation and running the program. This design will be in the form of a **UML class diagram** that shows different classes and their relations. The simulator will:

1- Offer a menu of choices
2- Allow the user to load a new program from a file
3- Fetch instructions a step by step to IR and validate it is a valid step and execute it
4- Allow the user to display the status of the registers, PC, IR, memory and screen at the end of program execution or after each step in a suitable format. (In text format)

All CS213 class can work together to design the simulator with OOP concepts like classes, inheritance, encapsulation and abstraction. Probably, the design will have classes like Machine, Instruction (can be multiple types of it), Register, Memory or Memory Manager, etc. Classes will have suitable attributes and operations and also inheritance or association relations between them. **Students can share design with or make it together BUT DO NOT SHARE CODE OR USE AI.**

Task 4.3 (4 marks - Group) Machine Simulator Development

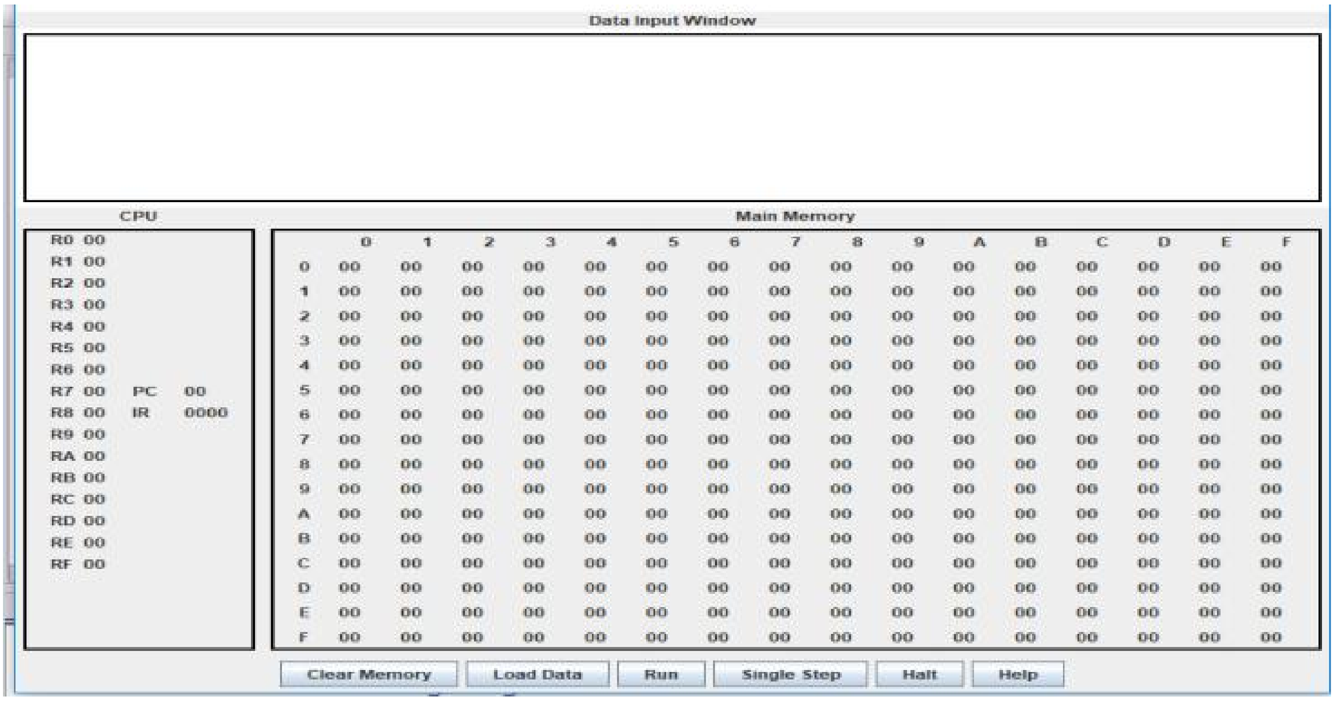
Team will implement its version of the simulator. Data is taken as a file of space separated chars representing the instructions, e.g. 10FF means load R0 with memory content in location 255_d. You need to support only the instructions below. Invalid instructions are ignored. Test your program on 20 real Vole programs like: “10E0 11E1 5201 32E2 C000” but longer. **All CS213 class can work together to develop & share the test cases.**

Machine Language

Op-code	Operand	Description
1	RXY	LOAD the register R with the bit pattern found in the memory cell whose address is XY. *Example:* 14A3 would cause the contents of the memory cell located at address A3 to be placed in register 4.
2	RXY	LOAD the register R with the bit pattern XY. *Example:* 20A3 would cause the value A3 to be placed in register 0.
3	RXY	STORE the bit pattern found in register R in the memory cell whose address is XY. *Example:* 35B1 would cause the contents of register 5 to be placed in the memory cell whose address is B1.
3	**R00**	**STORE to location 00, which is a memory mapping for the screen. Writing to 00 is writing to screen.**
4	0RS	MOVE the bit pattern found in register R to register S. *Example:* 40A4 would cause the contents of register A to be copied into register 4.
5	RST	ADD the bit patterns in registers S and T as though they were two’s complement representations and leave the result in register R. *Example:* 5726 would cause the binary values in registers 2 and 6 to be added and the sum placed in register 7.
6	RST	ADD the bit patterns in registers S and T as though they represented values in floating-point notation and leave the floating-point result in register R. *Example:* 634E would cause the values in registers 4 and E to be added as floating-point values and the result to be placed in register 3.
B	RXY	JUMP to the instruction located in the memory cell at address XY if the bit pattern in register R is equal to the bit pattern in register number 0. Otherwise, continue with the normal sequence of execution. (The jump is implemented by copying XY into the program counter during the execute phase.) *Example:* B43C would first compare the contents of register 4 with the contents of register 0. If the two were equal, the pattern 3C would be placed in the program counter so that the next instruction executed would be the one located at that memory address. Otherwise, nothing would be done and program execution would continue in its normal sequence.
C	000	HALT execution. *Example:* C000 would cause program execution to stop.

Bonus Task 5 (2 marks) - GUI for the Simulator & Competition

- Team will use a GUI library for C++ and develop a GUI for the simulator.
- Simulator will support the full list of instructions of the Vole machine.
- Be creative and develop GUI that is (1) Super easy to use and (2) Intuitive to understand. Here is a sample included of previous work but it is not the best design. DO NOT copy it.



R0	00		
R1	00		
R2	00		
R3	00		
R4	00		
R5	00		
R6	00		
R7	00	PC	00
R8	00	IR	0000
R9	00		
RA	00		
RB	00		
RC	00		
RD	00		
RE	00		
RF	00		

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
3	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
4	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
5	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
6	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
7	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
8	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
9	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
B	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
D	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
F	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

What to deliver for tasks 4 and 5?

I. Students should divide the work **as they like** and then **should integrate their code** together and make sure it works properly.

II. Use separate compilation and divide code to header, classes and app

III. **Working code in standard C++ not using third-party libraries if you are not doing GUI.** Name your files **A1-Task4-YourSection-YourIDs.cpp and .h and**

IV. A pdf report with the detailed design showing classes, relations & attributes. Include a **work break-down table** showing who did what & a cover page with team details.

V. Work on GitHub and add a screen shot for GitHub for shared projects.

VI. Zip everything in **A1-Task4-YourSection-YourIDs.zip**

If you are doing GUI

VII. Snapshots of the GUI of the simulator

VIII. **A link to public video** with FCAI logo and your name and prof name, explaining your application and showing how to use it in clear voice.

Marking Criterion

1. 3 0.75 x 4 individual problems - working solution

2. 2 Detailed report on AI tools code and comparison between them and 30 test cases

3. 1 Detailed report about training chances

4. 4 Working Vole machine simulator that can run different programs

5. 2 GUI for the machine that is clear and well designed .