University Transilvania Brașov
Faculty of Electrical Engineering and Computer Science
Study program Electrical Engineering and Computers in English Language

PROJECT No. 4
Custom Polynomials

Dan George Adrian, 2nd Year (2023-2024),4LF521A

1. Specification defining the problem

1.1 Description of the theme

Define the polynomial class to represent polynomials of any degree, with coefficients of any number (class will be defined generically (template)).
Class must implement: constructors (default, with parameter, copy, etc.) Destructor binary operators +, - (with result of polynomial type), the multiplication of two polynomials, left and right multiplication by a scalar, operator =, operator [] (return rate specified grade monomer), by reading the coefficients of the keyboard, display function, a get type method for finding the polynomial degree (int getGrd ()) function Calculate the polynomial for a given x. Use the above class to declare objects of type POLYNOMIAL and to illustrate the application of defined methods

1.2. Description of requirements

Overview

This project involves defining a generic `Polynomial` class to represent polynomials of any degree with coefficients of any numeric type. The class must provide a range of functionalities, including various constructors, binary operators, scalar multiplication, and methods for polynomial evaluation and degree determination.

Theme and Project Basis

The primary goal is to create a versatile and robust `Polynomial` class that can handle polynomials of varying degrees and coefficients of any numeric type. This will involve using templates in C++ to ensure the class is type-agnostic.

General Remarks

1. **Efficiency**: The implementation uses vectors for dynamic array handling, providing flexibility for polynomials of varying degrees.
2. **Error Handling**: Basic error handling for out-of-range access in the `operator[]`.
3. **Flexibility**: The use of templates ensures the class can handle different numeric types seamlessly.
4. **Scalability**: The class design supports easy extension for additional functionalities if required in the future.

This `Polynomial` class provides a comprehensive solution for representing and manipulating polynomials in a generic and efficient manner.

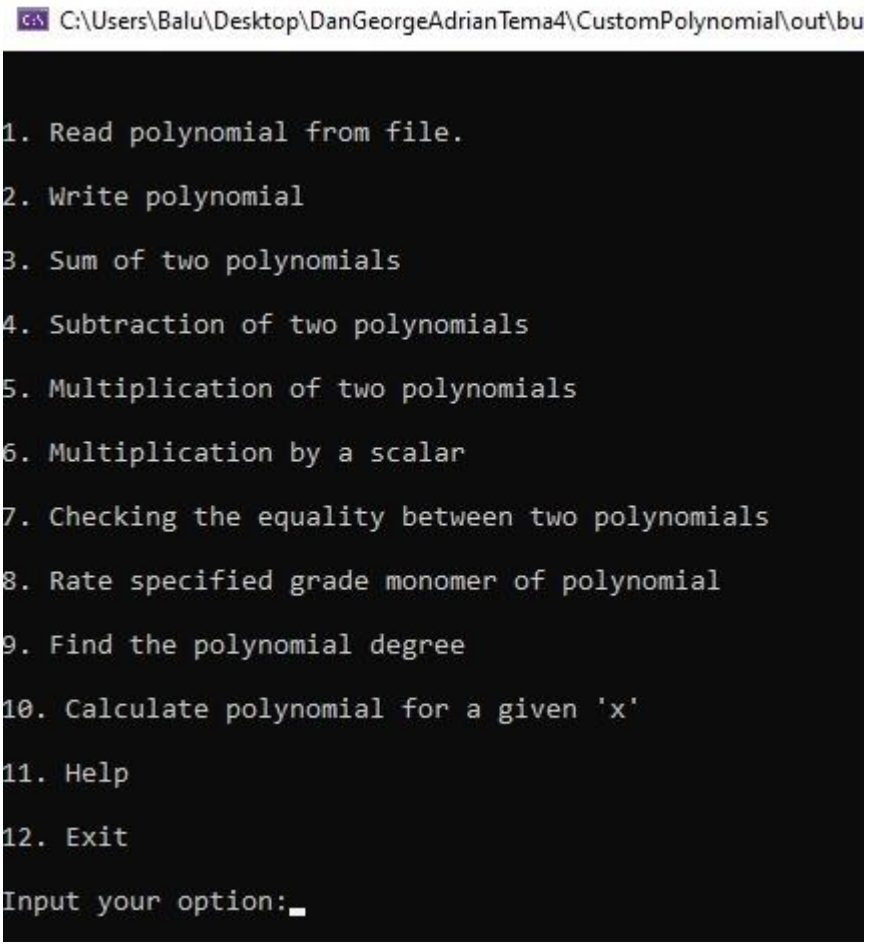
1.3. Functional specification

It specifies the functional requirements of the program. It presents all functional requirements of the program, relationships between them and their links with other requirements. Each request is presented separately including:

1. Input data
2. Output data
3. Functional diagram (in narrative form, logical flow charts or pseudo-code)

1.4. User Interface

It presents requirements for user interface (will be focused on a simple menu; it presents the proposed dialogue with the user for all menus, submenus, etc.)



2. Project Development

2.1. General objectives

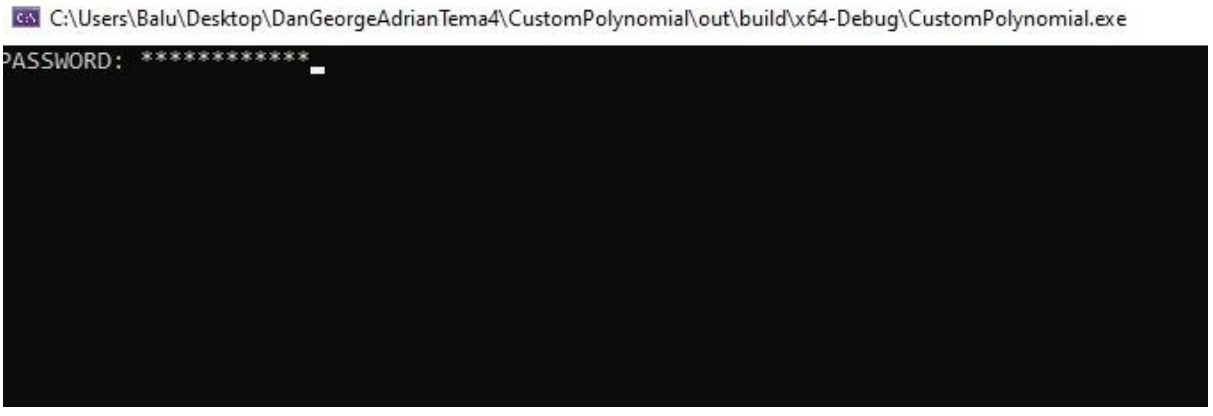
The primary objective of this project is to design and implement a generic `Polynomial` class in C++ capable of representing and manipulating polynomials of any degree with coefficients of any numeric type. The class aims to provide a comprehensive set of functionalities to perform various polynomial operations and evaluations. The following summarizes the key objectives:

1. **Robust Construction and Destruction**:
    - Implement constructors for initializing polynomials, including a default constructor, a parameterized constructor, and a copy constructor.
    - Ensure proper resource management with a destructor.
2. **Polynomial Operations**:
    - Define binary operators for polynomial addition, subtraction, and multiplication, returning results as polynomial objects.

   - Implement scalar multiplication for both left and right multiplication by a scalar value.
3. **Polynomial Access and Assignment**:
   - Provide an assignment operator for deep copying polynomial objects.
   - Implement an access operator to retrieve the coefficient of a specific degree.
4. **Degree and Evaluation**:
   - Develop a method to determine and return the degree of the polynomial.
   - Implement a method to evaluate the polynomial for a given value of x.
5. **User Interaction**:
   - Create methods for reading polynomial coefficients from the keyboard.
   - Create methods for reading polynomial coefficients from a file.
   - Develop a display function to print the polynomial in a human-readable format.
6. **Demonstration and Testing**:
   - Use the CustomPolynomial class to declare objects and illustrate the application of defined methods.
   - Ensure comprehensive testing to validate the functionality and correctness of the class.

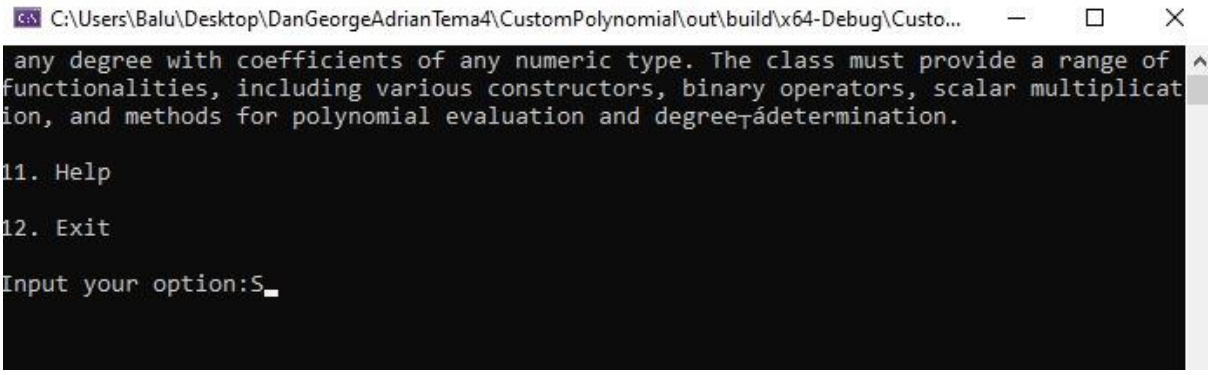
2.2. Description of data processed by the program

Description of input and output data and used formats (data structures/classes)



2.3. Description of program modules

functions, parameters, returned values, algorithms used, etc..



3. Conclusions

What conclusions were drawn from the project; Proposals for future developments of the project;
Elements that may take into account the implementation of other projects