CS352: Object Oriented Programming & Data Structures

Assigned on: 01-Oct-2025 (Wed)
Submission time: 17:00 Hrs, Wednesday, 15-Oct-2025
Instructors Dr. V. Vijaya Saradhi
Department of CSE, IIT Guwahati

1. Submit the assignment through MS Assignment
2. The system will not accept submissions after 17:00, 15-Oct-2025.
3. Plan your submission accordingly.

Question 1: (100 points)
Use of operator overloading concept. **Input file** description:

```
8   3 <-- number of polynomials, number of terms in each polynomial
2   2 <-- (rows x columns)
3   2 <-- first polynomial (coefficient degree)
6   1
7   0
5   2 <-- second polynomial (coefficient degree)
3   1
5   0
6   2 <-- third polynomial (coefficient degree)
2   1
9   0
1   2 <-- fourth polynomial (coefficient degree)
2   1
7   0
...
```

Input file `input-01.txt` consists of the following polynomials

1. polynomial p_1: $3x^2 + 6x^1 + 7x^0$
2. polynomial p_2: $5x^2 + 3x^1 + 5x^0$
3. polynomial p_3: $6x^2 + 2x^1 + 9x^0$
4. polynomial p_4: $1x^2 + 2x^1 + 7x^0$
5. polynomial p_5: $0x^2 + 9x^1 + 3x^0$
6. polynomial p_6: $6x^2 + 0x^1 + 6x^0$
7. polynomial p_7: $2x^2 + 6x^1 + 1x^0$
8. polynomial p_8: $8x^2 + 7x^1 + 9x^0$

Your task is to perform: Construct two matrices **A** and **B** using above polynomials as given below. That is matrix of polynomials:

$$\mathbf{A} = \begin{pmatrix} p_1 & p_2 \\ p_3 & p_4 \end{pmatrix} = \begin{pmatrix} 3x^2 + 6x^1 + 7x^0 & 5x^2 + 3x^1 + 5x^0 \\ 6x^2 + 2x^1 + 9x^0 & 1x^2 + 2x^1 + 7x^0 \end{pmatrix}$$

$$\mathbf{B} = \begin{pmatrix} p_5 & p_6 \\ p_7 & p_8 \end{pmatrix} = \begin{pmatrix} 0x^2 + 9x^1 + 3x^0 & 6x^2 + 0x^1 + 6x^0 \\ 2x^2 + 6x^1 + 1x^0 & 8x^2 + 7x^1 + 9x^0 \end{pmatrix}$$

1. Addition of two Matrices of polynomials **A** + **B**
2. Subtraction of two Matrices of polynomials **A** - **B**
3. Multiplication of two Matrices of polynomials **A** * **B**

To achieve this, implement the following tasks.

Task 01 (50 marks) Declare a class `polynomial` with the following specifications

1. (3 marks) `Private members`
   (a) `terms` of type `int` representing number of terms in a polynomial.
   (b) `coeff` of type pointer to an `int`.
   (c) `degree` of type pointer to an `int`.
2. (47 marks) `Public member functions`
   (a) (2 marks) A constructor with one default `int` argument. Default value taking 3. Initializes `coeff` and `degree` to 0.
   (b) (3 marks) A constructor with three arguments.
      i. First input: a pointer to `int`
      ii. Second input: a pointer to `int` and
      iii. Third input `int` argument.

      The first pointer refers to coefficient values, the second pointer refers to degree values. The integer specifies number of terms in the polynomial. This constructor initializes `coeff` with values in the first input argument. Initializes `degree` with values in the second input argument. Number of terms with third input argument.
   (c) (2 marks) A destructor to de-allocate the created memory
   (d) (5 marks) An overloaded >> `operator` with the following specification
      i. **Input argument 1**: `istream` reference
      ii. **Input argument 2**: `const polynomial` object
      iii. **Returns**: `istream` reference
      iv. **Functionality**: Read the coefficient and degree values from the input file (no need to use file I/O) into the given object
   (e) (5 marks) An overloaded << `operator` with the following specification
      i. **Input argument 1**: `ostream` reference

   ii. **Input argument 2**: `const polynomial` object
   iii. **Returns**: `ostream` reference
   iv. **Functionality**: Print to standard terminal the polynomial in the same format as given in the input file.

(f) (10 marks) An overloaded + `operator` with the following specification

   i. **Input argument 1**: `const polynomial` object
   ii. **Returns**: `polynomial` object
   iii. **Functionality**: Perform addition of two polynomials

(g) (10 marks) An overloaded − `operator` with the following specification

   i. **Input argument 1**: `const polynomial` object
   ii. **Returns**: `polynomial` object
   iii. **Functionality**: Perform subtraction of two polynomials

(h) (10 marks) An overloaded $*$ `operator` with the following specification

   i. **Input argument 1**: `const polynomial` object
   ii. **Returns**: `polynomial` object
   iii. **Functionality**: Perform multiplication of two polynomials.

Task 02 (35 marks) Declare a class `Matrix` with the following specifications

1. (3 marks) `Private members`
   (a) `rows` of type `int`
   (b) `columns` of type `int`
   (c) A pointer to pointer to `polynomial` object
2. `Public member functions`
   (a) (3 marks) A constructor with two `int`s taking default value 2. This constructor should initialize `rows` and `columns` with the first input argument and second input argument respectively. It should dynamically initialize memory for the polynomial object.
   (b) (2 marks) A destructor to de-allocate the created memory
   (c) (5 marks) An overloaded << `operator` with the following specification
      i. **Input argument 1**: `ostream` reference
      ii. **Input argument 2**: `const Matrix` object
      iii. **Returns**: `ostream` reference
      iv. **Functionality**: Print to standard terminal the matrix of polynomials as shown below (Refer to matrix **A** in Task 04) for output
   (d) (5 marks) An overloaded [] `operator` with the following specification
      i. **Input argument 1**: `int` data type
      ii. **Returns**: pointer to `polynomial`
      iii. **Functionality**: Performs matrix element assignment with `polynomial` object
   (e) (5 marks) An overloaded + `operator` with the following specification

        i. **Input argument 1**: `const Matrix` object
        ii. **Returns**: `Matrix` object
        iii. **Functionality**: Perform element-wise matrix addition

    (f) (5 marks) An overloaded − `operator` with the following specification

        i. **Input argument 1**: `const Matrix` object
        ii. **Returns**: `Matrix` object
        iii. **Functionality**: Perform element-wise matrix subtraction

    (g) (10 marks) An overloaded ∗ `operator` with the following specification

        i. **Input argument 1**: `const Matrix` object
        ii. **Returns**: `Matrix` object
        iii. **Functionality**: Perform matrix multiplication

Task 03 (15 marks) Perform the following in the main function.

1. (1 mark) Read the number of polynomials from the input file (`input-01.txt` or `input-02.txt`)
2. (1 mark) Read number of rows and number of columns from the input file (`input-01.txt` or `input-02.txt`)
3. (2 marks) Declare a two dimensional array `m1` of matrix objects of size `rows` × `columns`
4. (1 mark) Read each polynomial into a separate polynomial object
5. (2 marks) Assign the polynomial to each element of `m1` in the order in which you have read the polynomials

6. (1 mark) Declare a two dimensional array `m2` of matrix objects of size `rows` × `columns`
7. (1 mark) Read each polynomial into a separate polynomial object
8. (1 mark) Assign the polynomial to each element of `m2` in the order in which you have read the polynomials

9. (1 mark) Perform matrix addition and print the output
10. (1 mark) Perform matrix subtraction and print the output
11. (3 marks) Perform matrix multiplication and print the output