Spring Term 2023

OOP (A and B section) BSE

Assignment 2

Total marks - 150

CLO

Note: Plagiarism is a serious offense and will not be tolerated. Any student found to have copied code from the internet, cross sections, or any other source, including ChatGPT or any other AI tool, will be caught and will receive a grade of zero for the assignment. It is important that you complete your assignments on your own and seek help from your TA if you are struggling with the material. Remember, the purpose of these assignments is to help you learn and develop your skills, and copying someone else's work defeats that purpose. No assignment will be accepted after the due date. Submit .cpp and .h Files by zipping them and you must follow the criteria of submission by renaming your zip file as your RollNo_A2.zip i.e 22L-1234_A2.zip.

1. A shop has a stack of chocolate boxes, each containing a positive number of chocolates. Initially, the stack is empty. During the next N minutes, either of these two things may happen: **[20]**

- The box of chocolates on top of the stack gets sold
- You receive a box of chocolates from the warehouse and put it on top of the stack.
- If C[i] = 0, he sells a box. If C[i] > 0, he receives a box containing C[i] chocolates.
- It is confirmed that he gets a buyer only when he has a non-empty stack.
- The capacity of the stack is infinite.
- You have to calculate the number of chocolates sold.

For example,

Input: N = 5 & C = {3, 0, 2, 1, 4}

Output: 3

Explanation: At the start, the stack is empty.

At minute 1, a box of 3 chocolates is received and put on top of the stack. Stack: [3]

At minute 2, the top box of chocolates (3 chocolates) is sold. Stack: []

At minute 3, a box of 2 chocolates is received and put on top of the stack. Stack: [2]

At minute 4, a box of 1 chocolate is received and put on top of the stack. Stack: [1, 2]

At minute 5, a box of 4 chocolates is received and put on top of the stack. Stack: [4, 1, 2]

In total, 3 chocolates were sold (3 from the first box). Therefore, the output is 3.

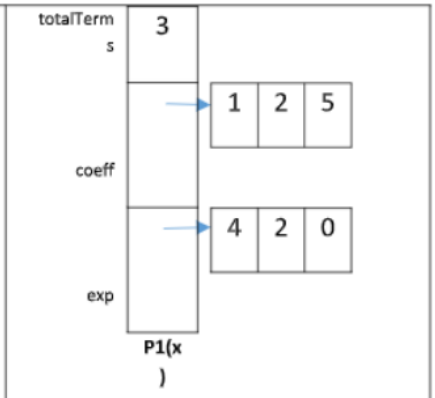
2. A university department consists of multiple teachers and offers various courses. Each course is taught by a teacher, and a teacher can teach multiple courses. Design a program to model this relationship using aggregation, where the Teacher class has a list of Course objects. Create classes Teacher and Course to represent teachers and courses, respectively. The Teacher class should contain attributes like id, name, and a list of courses they teach. The Course class should have attributes like id, title, and teacherId. Implement methods to add courses to a teacher and to display all courses taught by a teacher. **[20]**

3. A point in the x-y plane is represented by its x-coordinate and y-coordinate. Design a class, pointType, that can store and process a point in the x-y plane. You should then perform operations on the point, such as setting the coordinates of the point, printing the coordinates of the point, returning the x-coordinate, and returning the y-coordinate. Also, write a program to test various operations on the point. **[20]**

4. *The following question must use operator overloading concepts, use of any other libraries or hardcoding will not be accepted.*

A polynomial ***P1(x)*** $= x^4 + 2x^2 + 5$ has three terms: x^4, $2x^2$ and 5. **Coefficients** of these terms are 1, 2 and 5 respectively while **exponents** are 4, 2 and 0 respectively. To work with Polynomials, a definition of class Polynomial is given below and memory configuration for P1 is shown as follows:

```
class Polynomial
{
private:
int totalTerms;//Total terms in a Polynomial
int* coeff;//to save array of coefficients
int* exp; //to save array of exponents
};
```



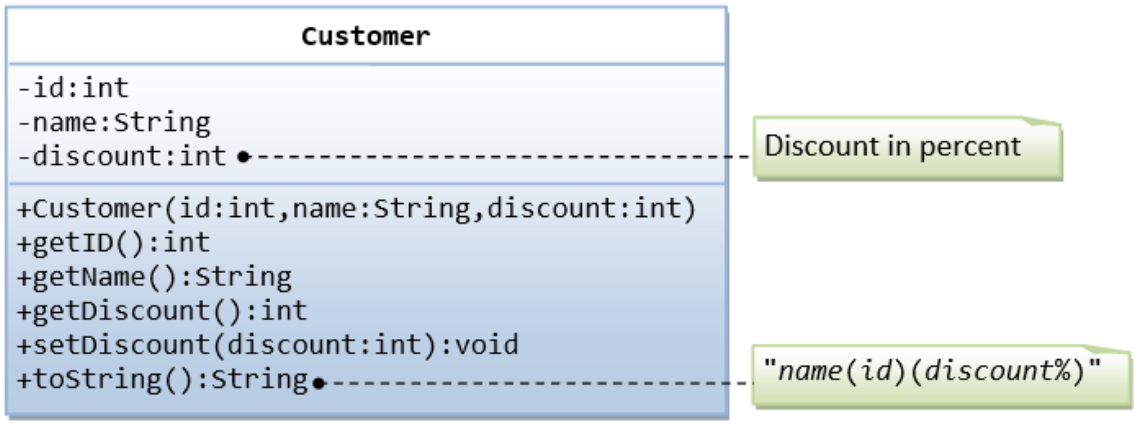
Your task is to complete the definition of Polynomial class such that the provided main program runs successfully without any errors. Make sure that your program doesn't consume extra memory space and it should not leak any memory. Use operator overloading like arithmetic, binary and assignment to complete this task. **[40]**

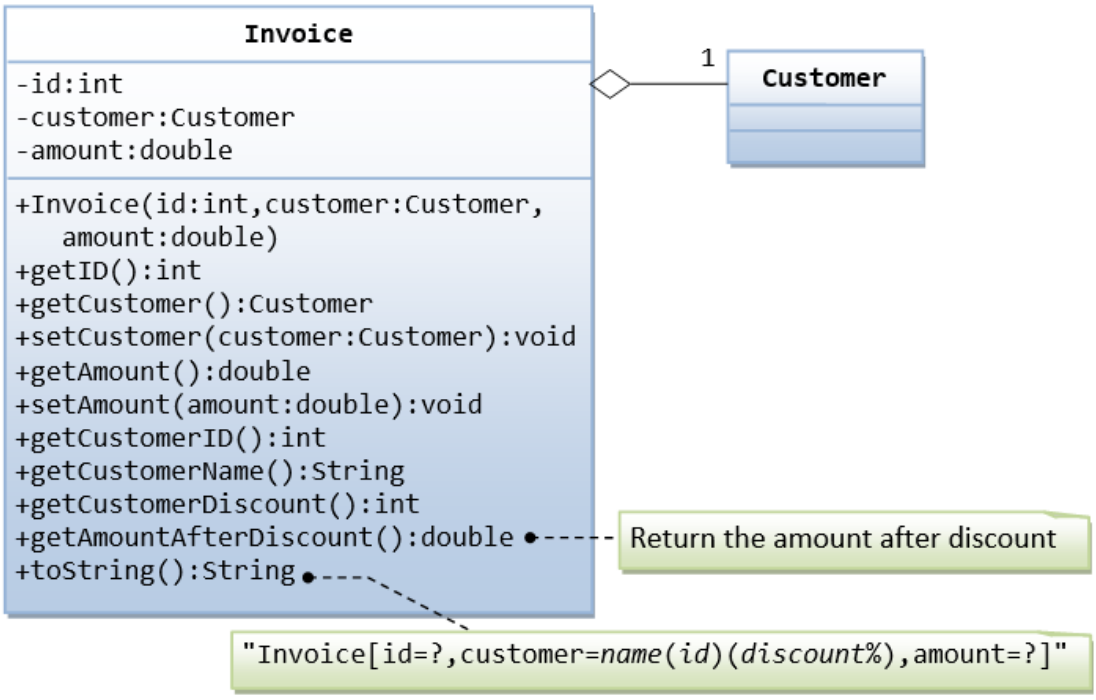
```
void main()
{
      int coeff_P1[] = {1,2,5};   //Coefficients for Polynomial P1
      int exp_P1[] = {4,2,0};     //Exponents for Polynomial P1

      int coeff_P2[] = {4,3};     //Coefficients for Polynomial P2
      int exp_P2[] = {6,2}; //Exponents for Polynomial P2

      Polynomial P1(3, coeff_P1, exp_P1);//Creates P1 with 3 terms (P1 =
1x^4 + 2x^2 + 5x^0 )
      Polynomial P2(2, coeff_P2, exp_P2);//Creates P2 with 2 terms (P2 =
4x^6 + 3x^2)

      cout<<"P1 = "<<P1<<endl; //Prints P1 = x^4+2x^2+5
      cout<<"P2 = "<<P2<<endl; //Prints P2 = 4x^6+3x^2
     if(!P1)
cout<<"P1 is zero"<<endl;/*if polynomial has only 1 term and its coeff
and exp are zero. i.e. if p1 = 0.*/
if(P1 != P2)
cout<<"P1 is Not Equal to P2"<<endl;
      Polynomial P3 = P1+P2;      //Adds P1 and P2 and saves result in
P3.You may consume extra space for resultant Polynomial in Add function
      cout<<"P3 = "<<P3<<endl;    //Prints P3 = 4x^6+x^4+5x^2+5
cout<<++P1<<endl;//adds 1 in all the coefficients.
cout<<P1<<endl;
cout<<P1++<<endl; //adds 1 in all the coefficients.
cout<<P1<<endl;

P3 = 2 + P1; //Assume P1 already has a constant term, add 2 in it.
cout<<"P3 = "<<P3<<endl;
}
```

This main function is only to be used for question 4. Your own code will not be considered.

Question 5,6,7 & 8 are related with each other and is of 50 Marks

5. Implement the given: **[10]**





6. Implement a superclass Appointment and subclasses Onetime, Daily, and Monthly. An appointment has a description (for example, "see the dentist") and a date. Write a method occursOn(year, month, day) that checks whether the appointment occurs on that date. For example, for a monthly appointment, you must check whether the day of the month matches. Then fill a list of Appointment objects with a mixture of appointments. Have the user enter a date and print out all appointments that occur on that date. **[10]**

7. Improve the appointment book program of Q6. Give the user the option to add new appointments. The user must specify the type of the appointment, the description, and the date. **[10]**

8. Improve the appointment book program by letting the user save the appointment data to a file

and reload the data from a file. The saving part is straightforward: Make a method save. Save the type, description, and date to a file. The loading part is not so easy. First determine the type of the appointment to be loaded, create an object of that type, and then call a load method to load the data. **[20]**

GOOD LUCK