CPT106 C++ Programming and Software Engineering II

Assessment 1

Contribution to Overall Marks	70%
Release Date	May 5, 2022, 23:59
Submission Deadline	May 25, 2022, 23:59

How should the work be submitted?

SOFT COPY ONLY!

(This is an INDIVIDUAL WORK and MUST be submitted through Learning Mall so that we can run your programs during marking.)

1. Create a Solution and set your student ID as the solution name.

2. Create a project for each problem within the solution. The project name should be Problem1, Problem2 and so on for the problems below.

3. **All C++ source code files (including solution and project files)** for all tasks should be zipped into *a single file named with your student ID number* and submitted to Learning Mall.

4. Late Submission Policy: 5% of the total marks available for the assessment shall be deducted from the assessment mark for each working day after the submission date, up to a maximum of five working days.

Assessment Overview

This assessment aims at testing basic concepts of C++ programming, such as inheritance, information hiding, overloading, friend functions, vector, constructor and related software engineering concepts. The marking criteria are as follows:

Problem 1 (5 marks)

Write a function

   Int[] union_Array (int a[], int lenA, int b[], int lenB)

that constructs the union of two integer arrays and removes any element that occurs more than once.
For example, if two arrays of the union_Array are {1, 8, 16 } and {11, 8, 7, 1}, it will return an array
of {1, 8, 16, 11, 7}.
Tasks:
   1. The length of two arrays should be different and defined by constants using "#define". (1
      mark)
   2. Arrays are inputted from the keyboard.(1 mark)
   3. Call the function **union_Array** in your main function. Initialize two arrays. Print out the new
      array (there is no requirement for the order of the output elements). (1 marks)
   4. Compile and run correctly(2 marks)

Problem 2 (5 marks)

Given a student class to represent student information as follows:

class Student{

   string stuId;

   string name;

   double mathScore;

   double chineseScore;

   double englishScore;

};
Write a program to achieve the following tasks:
   1. Use an array to save student information. The length of the array should be defined by a
      constant using "const". (1 mark)
   2. Student information is inputted from the keyboard. (1 mark)
   3. Print out the student ID and average mark of three courses for each student in a descending
      sequence.(1 mark)
   4. Print out student ID whose average mark is smaller than the average mark of all students.(1
      mark)
   5. Compile and run correctly. (2 marks)

Problem 3 (5 marks)

Given a student structure to represent student information as follows:

 structure student{

   string stuId;

   string name;

   double mathScore;

   double chineseScore;

   double englishScore;

};

Write a program to achieve the following tasks:
1. Use a pointer and linked list to save student information. The length of the linked list (number of student objects) should be inputted from the keyboard. (1 mark)
2. Student information is inputted from the keyboard. (1 mark)
3. Print out the student ID and average mark of three courses for each student in a descending sequence. (1 mark)
4. Print out student ID whose average mark is smaller than the average mark of all students. (1 mark)
5. Compile and run correctly. (1 mark)

Problem 4 (5 marks)

Given a function header as **char *findC (char const *source, char const *obj)**:

Write a program to achieve the following tasks:

1. Write a function to search a character sequence pointed to by a pointer (called "obj"), in another character sequence (called "source"). Return the pointer pointing to the found character. If there is more than one target found in the source, return the pointer pointing to the first one. (2 marks)

   Eg1: search for "C" in "ABCDEF", return the pointer point to 'C'.

   Eg2: search for "Z" in "ABCDEF", return a NULL pointer.

   Eg3: search for "CD" in "ABCDEF", return the pointer point to 'C'.

   Eg4: search for "CF" in "ABCDEF", return a NULL pointer.

   Eg5: search for "A" in "ABCAFC", return the pointer point to the first 'A'.
2. Write a main function to input two strings by the keyboard and print the position of the pointer in the source string(print -1 if the function returns a NULL pointer). (1 mark)
3. Compile and run correctly. (2 marks)

Problem 5 (10 marks)

Given a circle class to represent circle information as follows:

class Circle{

    int x;

    int y;

    double radius;

};

Write a program to achieve the following tasks:
1. The centre point of (x,y) and radius should be inputted from the keyboard. (1 mark)
2. Define a method that can calculate and return the circle's area. (1 marks)
3. Define an overloading operator + to add two circle objects and get a new circle object. The centre point of the new circle object is the same as the centre point of the first circle object. The area of the new circle object should be the sum of the areas of two circle objects (You can calculate the radius of the new circle based on the sum of the area of two circles). (2 marks)

4. Define a function to print relations of two circles(interaction, tangency, separation and inside) (2 marks)
5. Define a main function to input two circle objects, get a new circle object based on these two circle objects and print the relations of these two circle objects. (2 marks)
6. Compile and run correctly. (2 marks)

Problem 6 (30 marks)

Given a set of x,y data points, it is often necessary to automatically calculate an equation which gives the best fit line through the data. This type of analysis is known as line fitting or regression analysis.

The mathematical basis behind line fitting is shown below. You have a set of data points represented by $(x_1,y_1), (x_2,y_2) ..... (x_n, y_n)$. You want an equation which represents this data, the exact type of equation will depend on the way in which the data varies. Given any mathematical relationship between x and y it should be possible to generate a line fitting algorithm. The line fitting algorithm is described below.

Straight line

The equation for a straight line is: $y = ax + b$. Suppose that we have n points of (x,y) values from the text file, the values of **a** and **b** for **the best fit line** that is the best approximation of the given set of data points can be obtained from the following equation:

$$a = \frac{\begin{vmatrix} b_1 & d_1 \\ b_2 & d_2 \end{vmatrix}}{\begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix}} \quad , b = -\frac{\begin{vmatrix} a_1 & d_1 \\ a_2 & d_2 \end{vmatrix}}{\begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix}}$$

where

$$a_1 = \sum_{i=1}^{n}(x_i * x_i), \qquad a_2 = \sum_{i=1}^{n} x_i ,$$

$$b_1 = \sum_{i=1}^{n} x_i , \qquad b_2 = n ,$$

$$d_1 = -\sum_{i=1}^{n}(x_i * y_i), \qquad d_2 = -\sum_{i=1}^{n} y_i$$

The value of a second order determinant $\begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix}$ is the difference between the two dot products

$(a_1 b_2 - a_2 b_1)$. Create a C++ program to calculate its value.

Distance

The distance from a point $P(x,y)$ to the line $ax+by+c=0$ is the length of the perpendicular line from the point P to the line:

$$\text{Distance} = \frac{|ax+by+c|}{\sqrt{a^2+b^2}}$$

where (x,y) is the coordinate value for point **P**.

Standard error of distance for all the points: $\text{SE}=\sqrt{\dfrac{\sum_{i=1}^{n}(x_i-\bar{x})^2}{n-1}}$

Where $\bar{x}$ is the mean distance of all points, x_i is the distance of Point P_i , n is the number of all points.

Please complete the assessment according to the following steps: (30 marks)
1. Create a file operation class "FileOp" with functions as follows: (12 marks, 3 marks per question)
    a) It can allow users to input points from the keyboard and save these points into a text file.
    b) It can read points from the text file.
    c) It can delete a point from the text file.
    d) It can check if a point is in the text file.
2. Create a class "LineFitting" with functions as follows: (10 marks, 2 marks per question)
    a) It can calculate the parameters of a, b and c for the best fit line ax+by+c=0.
    b) It can calculate the distance between each point in your file and the best fit line.
    c) It can find the best point from the file, which is not in the best fit line but has a minimal distance compared to other points.
    d) It can find the worst point from the file, which has the maximal distance compared to other points.
    e) It can calculate the standard error of distance for all the points.
3. Create a main method in your program to achieve the following functions: (8 marks)
    a) Create a main method. (2 mark)
    b) Create a text file using the "FileOp" class with some points. (1 mark)
    c) Print out the equation of the best fit line (e.g. 3x+2y+3=0). (1 mark)
    d) Ask the user to input a point and print out whether the point is in the text file or not. If it is in the file, then print out its position. (1 mark)
    e) Print out the worst point that has a maximal distance to the best fit line in all the points. (1 mark)
    f) Print out the standard error for all the points. (1 mark)
    g) Ask the user to input a point and delete it from the file if the point exists in the text file. (1 mark)
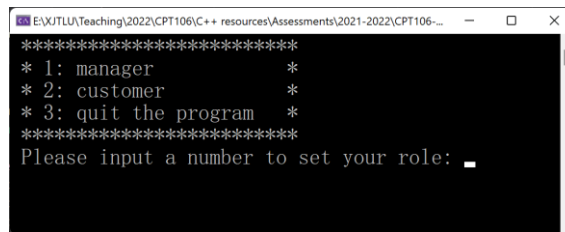
Problem 7 (40 marks)

Online trading systems can help customers buy goods from the internet. In this question, you are required to write a program to simulate the trading procedure and allow customers to query and buy some goods.

Tasks:

(1) Create a Product class with several private member variables, such as product id, product name, product amount and price. (2 marks)

(2) Create a base class Person with several private member variables, such as person name, gender, mobile number and address. The gender should be an enum. (2 marks)

(3) Create a subclass Customer based on the Person class. It has extra private member variables, such as customer id, capital and a vector of products the customer is purchasing. Each customer has 10,000 RMB initial capital. (2 marks)

(4) Create a subclass Manager based on Person class. The manager has extra private member variables, such as age and title. (2 marks)

(5) Create a MainProcess class. It has private member variables, such as a vector of products to save all product information, a vector of customers to save all customers, a manager who can manage customers and products, a customer who is purchasing products. In addition, it also has some functions to output operation menus both for managers and customers to choose from. (2 marks)

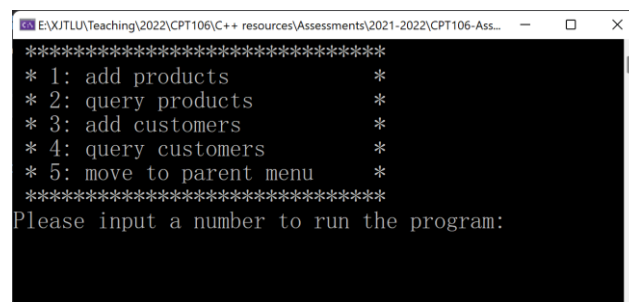(6) When the program runs, it shows a menu as follows: (2 marks)

Fig.1. Main operation menu

There are two roles in this online trading system (manager and customer). Different roles have different functions.

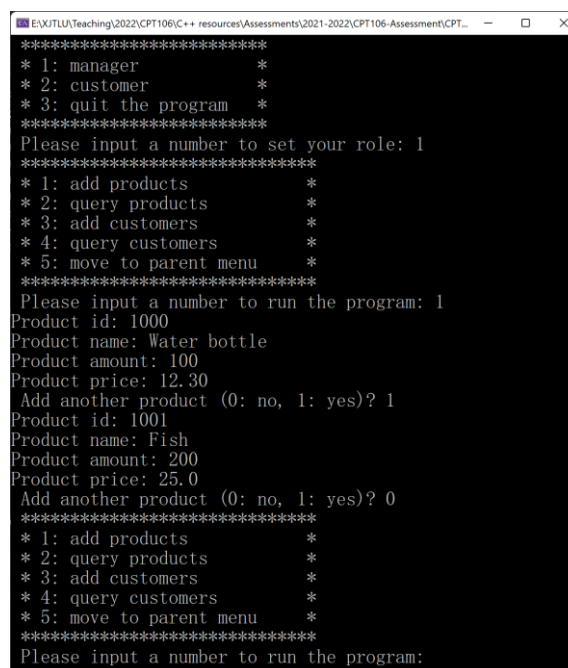(7) If the input is 1, then move to manager's menu as follows: (2 marks)

Fig.2. Manager's operation menu

The manager can add new products, query products, add new customers and query customers. If the user inputs 5, then the program will move to the parent menu shown in Figure 1 (main operation menu).

   a) Add products: manager can add new products into a product vector so that customers can buy these products. (2 marks)

Fig.3. Operation of adding products

The manager can continue to input product id, name, amount and price to create and add new products into the product vector. Duplicated product id cannot be added to the product vector. When the manager input 0 to stop adding, the manager operation menu will output on the window, and the manager can input another number to do other operations.

b) Query products: manager can input a product id to query a particular product or input "*" to list all products in the product vector. Output "No product found!" if there is no particular product. (2 marks)

Fig.4. Operation of querying products

c) Add customers: manager can add new customers to the customer vector. (2 marks)

Fig.5. Operation of adding customers

The manager can input customer id, name, gender, mobile number and address to create and add new customers into the customer vector. Duplicated customer id cannot be added to the customer vector. When the manager inputs 0 to stop adding, the manager operation menu will be outputted on the window, and the manager can input another number to do other operations.

d) Query customers: manager can input a customer id to query a particular customer or input "*" to output all customers in the customer vector. "No customer found!" will be outputted if there is no particular customer. (2 marks)

Fig.6. Operation of querying customers

e) Move to parent menu: if the manager inputs 5, then the program moves to the main operation menu shown in Figure 1. (2 marks)

(8) On the main operation menu, if the user inputs 2, then the system will ask the user to input a customer id to confirm which customer will purchase products. If the customer id exists in the customer vector, then the operation menu for customers will be outputted as Figure 7 shows. (2

marks)

Fig.7. Customer's operation menu

otherwise, ask users to input a new customer id or a special string "quit" to move to the parent operation menu shown in Figure 8 (it means that you should create at least one customer before you can enter into the customer operation menu and start to purchase products).

Fig.8. Failed to find the customer Id

a) Add products into cart: a customer can add products into the cart so that the customer can buy these products. Customer can input product id and the amount he/she wants to buy. The program will show a reminder if the customer does not have enough capital or the amount the customer wants is larger than the available amount of the product. The program should output successful information if the adding operation runs successfully. In addition, the amount of product in the product vector should be deducted accordingly. (2 marks)

Fig.8. Operation of adding products into cart

b) Query products in cart: customers can query all products added in the cart. The customer can input a specific product id to query one product or input "*" to list all products in the cart. (2 marks)

Fig.9. Operation of querying products in cart

c) Check up: customer can check up the bill for all products in the cart. It will list customer's id, name, gender and capital, all products in the cart and the total price. (2 marks)

Fig.10. checking up operation

If the cart is checked up successfully, the capital of the customer should be deducted based on the total price of products the customer has bought.

(9) Make sure your source code has some comments (3 marks) to help us understand and can be compiled successfully (5 marks).