Second Semester Examination
2022/2023 Academic Session

July/August 2023

CPT113 – Programming Methodology & Data Structures

Duration : 2 hours

Please check that this examination paper consists of SEVEN (7) pages of printed material before you begin the examination.

Instructions : Answer **ALL (4)** questions.

1. (a). The following is a definition of an object:

```
class myClass{
      private:
         string name;
         int *value;
};
```

(i). Demonstrate a suitable copy constructor definition for the above class by writing appropriate C++ code. Assume that the method is written outside of the class.

(5 marks)

(ii). Give an example of how to trigger this copy constructor?

(2 marks)

(b). Given the following program, answer the following without adding more **variables** into the respective classes. **Assume all methods/functions are defined outside of the class definition.**

```
class Shape {
   private:
      float x;
};

const double PI= 3.14159;
class Circle: public Shape {
   double radius;
};

class Square: public Shape {
   public:
      //write accessor & mutator
};
```

(i). Show suitable accessor and mutator for class `Square`. You may write assumptions (as comments) of any available methods from any of the classes.

(8 marks)

(ii). Show the methods `getArea` and `getPerimeter` by writing appropriate C++ code that belong to the class `Circle`. Given:

`area of a circle = PI * x * radius and`
`perimeter of a circle = 2 * PI * radius`

You may write assumption of available methods belong to any of the class.

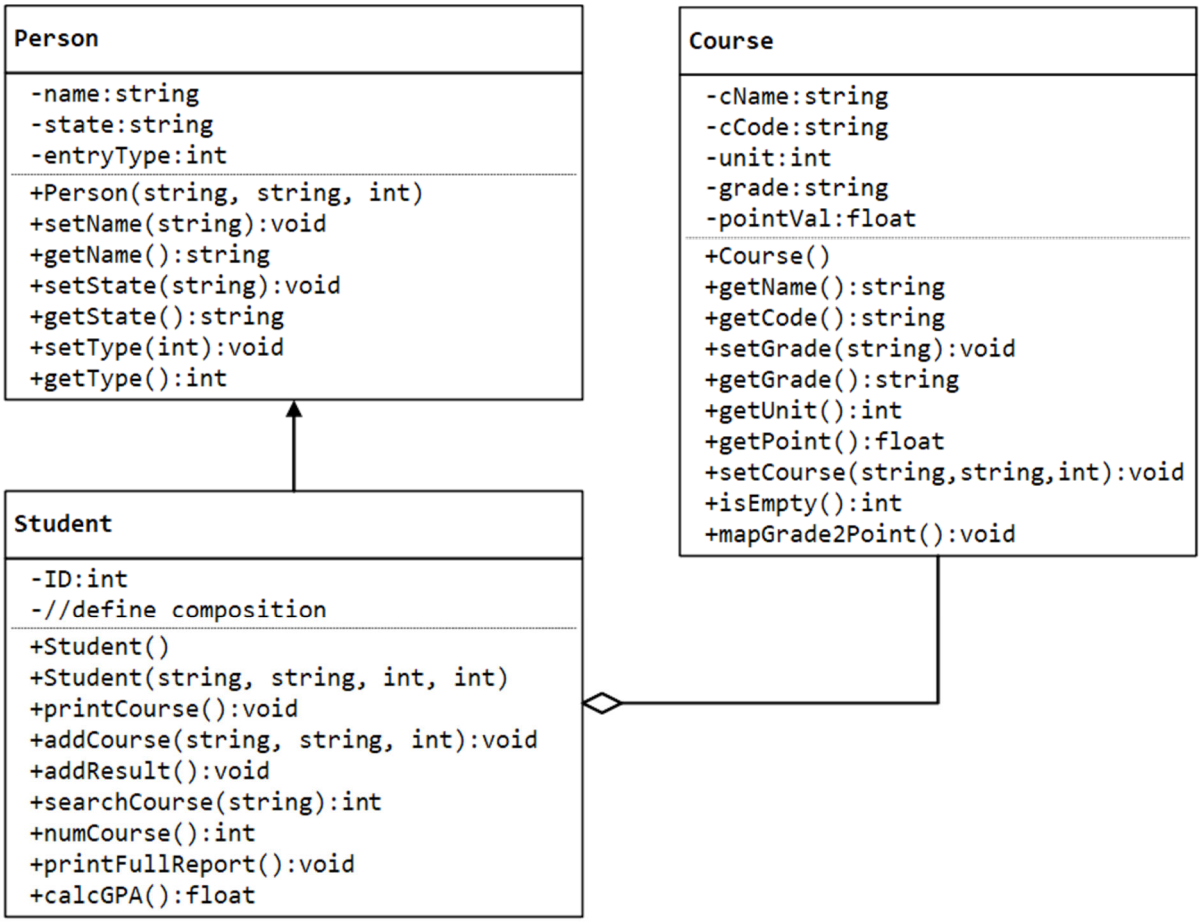
(7 marks)

(iii). Show overloading constructor for class `Circle` by writing appropriate C++ code.

(3 marks)

2. Analyse the following scenario:
There is a class `Person`, storing information available for a person which includes their full name, their state, and their entry qualification level (to university). There is also a class `Student` which is publicly inherited from class `Person`, which consists of Student `ID`, and is composed of the class `Course`.
Each student is allowed to register up to 5 courses only.
All method definitions must be non-inlined function.

The following UML diagram is provided to illustrate the structure of the classes and their relationships.



You are also given the following brief description to some methods.

`Student` members:

`printCourse()`: will check if the student has courses in their list and if it is not empty, display the details of the student's course.
`addCourse(string, string, int)`: accept course name, course code and unit to be added into the student's course list.
`addResult()`: will check if the course code is available in the student list, add student grade, then update grade point value (`pointVal` variable).

...4/-

`searchCourse(string)`: will accept course code and search if the code is available in the list of course belonging to a student. Return 1 if the course code is found.
`numCourse()`: will return the number of courses that a student currently has.
`printFullReport()`: print out the details of the student from person details, student and course list
`calcGPA()`: will calculate the GPA of the courses in the list.

`Course` members:
`setCourse(string,string,int)`: accept 3 values for course name, course code and unit of that course.
`isEmpty()`: checking if any course has been added into the student's course. Will return 1 if there is no course and will return 0 if there is already course assigned to a student.
`mapGrade2Point()`: map the grade assigned to a course to their respective grade point value for example grade A will obtain 4.00 point value, grade A- will receive 3.67 point value.

(a). Show how should the composition of class `Course` be declared under the `Student` class. Refer the UML diagram given.

(2 marks)

(b). Write a method called `isEmpty()` that belongs to the `Course` class. If any additional methods are required (apart from the ones declared in the UML), their definitions need to be written as well.

(2 marks)

(c). Write a method called `addCourse()` that belongs to `Student` class. If any additional methods are required (apart from the ones declared in the UML), their definitions need to be written as well.

(7 marks)

(d). Write a method called `calcGPA()` that belong to `Student` class. This method first ensures that grades are assigned to all courses belonging to a student and will then calculate the GPA based on the list. If any additional methods are required (apart from the ones declared in the UML), their definitions need to be written as well.

(7 marks)

(e). Write the `printFullReport()` method belonging to the `Student` class. The method will ensure that grades for all courses are already assigned before it can print full report. Report must display the student's details from the base class and the `Student` class, as well as the courses taken with each corresponding grade and point value. Finally, the method should be able to display the obtained GPA. If any additional methods are required (apart from the ones declared in the UML), their definitions need to be written as well.

(7 marks)

...5/-

3. (a). The following is a definition of a struct called `Player`:

```
struct Player{
        string characterClass;
        int maxHealth;
        string ability;
        int maxMana;
};
```

(i). For each of the following lines of code, identify if an error exists. If there is an error, correct it.

```
1:  Player *playerPtr = nullptr;
2:  playerPtr = new Player;
3:  playerPtr.characterClass = “Mage”;
4:  *playerPtr.maxHealth = 25;
5:  *playerPtr->ability = “Fireball”;
6:  (*playerPtr).maxMana = 30;
```

(2 marks)

(ii). Complete the following code to create a dynamic 2-dimensional array of Player objects called `players`. Then, create a nested for loop to initialise the `characterClass` for all Player objects in the 2-dimensional array.

```
int rows = 0, cols = 0;
cin >> rows >> cols;

//Your code snippet goes here
```

(5 marks)

(b). Bob wants to create a trading card game. In this game, each player will take turns to guess a number from 1 to 10, then draw a card from the **top** of a deck. If the card matches the guessed number, the player will keep the card. Otherwise, the card will be put on **top** of a discard pile. The player with the most cards wins.

Each card will have a pre-assigned number from 1 to 10. Bob is now deciding which data structure to use to represent the deck of cards in his game.

(i). Suggest a suitable abstract data structure (ADT) for Bob’s card game and explain why.

(2 marks)

(ii). Write the **class definition** called `CardDecks` that has the following requirements:

- Cards can be represented by a struct.
- Keep track of both the main deck and the discard pile.
- To allow more flexibility, the ADT must be dynamic in nature.
- The class definition must include at least member functions to **`add`** cards to the main deck, **`draw`** a card from the main deck (returns the value of the card) and **`discard`** cards (adds a card to the discard pile).

(4 marks)

(iii) Write the member function definitions for `add`, `draw` and `discard`.

(12 marks)

4. (a). Given the following formula:

$$nTermR(a, r, n) = \begin{cases} a & \text{if } n = 1 \\ r * nTerm(a, r, n-1) & \text{otherwise} \end{cases}$$

(i). Trace the given recursive function when a=3, r=2 and n=5. Provide the final answer.

(5 marks)

(ii). Analyse the formula in 4(a). What does `nTermR` do?

(3 marks)

(iii). The following function has a resemblance to the above formula. Assume that the recursive formula above has been converted into a function called `nTermR`. Examine the difference between the output of `nTermR` and the function below:

```
void nTerm (int a, int r, int n){
   if(n>0){
      nTerm (a,r,n-1);
      int An = a * pow(r,n-1);
      cout << "Term " << n << ": " << An <<endl;
   }
}
```

(2 marks)

(b). Analyse the following binary tree orders. Given bold **[I]** is the root of the tree:
Preorder: I U D I F N O Y A R G I I K
Inorder: F I D N U Y O **[I]** R A G I I K

(i). Construct the binary tree.

(8 marks)

(ii). List the nodes in postorder sequence.

(4 marks)

(iii). Given the nodes of a binary tree in preorder and postorder sequences, explain if it is possible to reconstruct a unique binary tree.

(3 marks)

- oooOooo -