Name: Noor-ul-Ain Sagheer

Reg #: BCS223020

Date: 16-Oct-2023

Course: Object Oriented Programming

Section: '1'

Assignment No: '1'

Description: "Classes and Objects"

Submitted to: Sir Adnan Jelani

Capital University of Science & Technology

بِسْمِ اللّٰهِ الرَّحْمٰنِ الرَّحِيْمِ

Question # 01:
part (a)

Code:

```
#include<iostream>
using namespace std;

class Book
{
    Private:
        string title;
        string author;

    Public:
        Book(string Title, string Author)
        {
            title = Title;
            author = Author;
        }

        void display_book_info()
        {
            cout<<" The Title of Book is = "<< title;
            cout<<endl;
```

```
cout<< " The Name Of Author of Book = "<<author;
cout<< endl;
}
};
void main()
{
Book obj("Object Oriented Programming In
C++", "David parsons");
obj.display_book_info();
}
```

Part b:

Access Specifier:

The access specifiers are used to specify the access level of class members.

- Different access specifiers in C++ are;
  - Private.
  - Public.
  - Protected.

Private access specifier is used to completely hide class members. The class members declared with private access specifier cannot be accessed from

outside the class.

Protected: The class members declared with protected access specifier can only be accessed within the class or from the derived / Sub classes.

Public: The class members declared under public access specifier can be accessed from anywhere in the program.

Access	Public	Protected	private
• Access of class members in same class	Yes	Yes	Yes
• Access of class members in sub classes.	Yes	Yes	No
• Access of class members from outside class	Yes	No	No

Question # 02:

Part (a):

```
#include<iostream>
using namespace std;

class Student
{
  Private:
     Int student-id;
     string name;
```

```
Public:
    student(int student-ID, string Name)
    {
        student-id = student-ID;
        name = Name;
    }

    void display-info()
    {
      cout<<" The name of student = "<<name<<endl;

      cout<<" ID of a student = "<<student-id;
       cout<<endl;
    }

};

    void main()
{
    Student S1(12345, "Putin");
    S1.display-info();

    Student S2(67890, "Joe");
    S2.display-info();
}
```

Part (b):

Importance of Destructor:

A Destructor in a **student** class may not be necessary in this specific

case because the class does not manage any **dynamic resources** (memory allocation) or perform any cleanup tasks.

- Destructors are useful when you used resources that needs to be released (explicitly).

- The purpose of Destructor is to ensure resources are clearly cleaned up when an object goes out of scope.

- C++ will generate a default Destructor, when we don't define a custom destructor.

- In this case, we are relying on a default Destructor.

.... Why student-id private...

student-id is made private to hide the details of this data member, from external code.

- It ensures that the internal data of class "Student" object is not directly accessible or modified from outside class.

.... Why display-info public...

"display-info" is made public to make it easy for anyone to look at the information about student nicely and in organized way.

- It is a gate-way for accessing or modifying data in internal side of class, from outside.

Explain how .... why it's important in class design.

- Public means everyone can access that part of class.
- Private means only the class itself can use that part.

⇒ These rules are important in class design because these access specifiers control's who can use or change the class data and functions.

- This keeps our code safe.
- It helps make it work correctly.

Github account details:

Github email: chazeem19800@gmail.com
Github ID: NOOR-UL-AIN-SAGHEER