Object Oriented Programming Lab Task 8

Name: Muhammad Ibad Saleem
Roll No: 19K-0220

MCQS

1-The scope-resolution operator usually
a. limits the visibility of variables to a certain function.
b. tells what base class a class is derived from.
c. specifies a particular class.
d. resolves ambiguities
.

2-. **True** or false: It is sometimes useful to specify a class from which no objects will ever be created.

3- Assume that there is a class Derv that is derived from a base class Base. Write the declarator for a derived-class constructor that takes one argument and passes this argument along to the constructor in the base class.

```
=> Derv (int x): Base(x) {}
```

4- Assume a class Derv that is privately derived from class Base. An object of class Derv located in main() can access
a. public members of Derv. **// can be this as well**
b. protected members of Derv.
c. private members of Derv.
d. public members of Base.
e. protected members of Base.
f. private members of Base.

5- **True** or false: A class D can be derived from a class C, which is derived from a class B, which is derived from a class A.

6-. A class hierarchy
a. shows the same relationships as an organization chart.
b. describes "has a" relationships.
c. describes "is a kind of" relationships.
d. shows the same relationships as a family tree.
18. Write the first line of a specifier for a class Tire that is derived from class Wheel and from class Rubber.

7-Assume a class Derv derived from a base class Base. Both classes contain a member function func() that takes no arguments. Write a statement to go in a member function of Derv that calls func() in the base class.

```
Base::Func();
```

8-True or **false**: It is illegal to make objects of one class members of another class.

9- In the UML, inheritance is called Sub/Super Class.

10- Aggregation is
a. a stronger form of instantiation.
b. a stronger form of generalization.
c. a stronger form of composition.
d. a "has a" relationship.

11- **True** or false: the arrow representing generalization points to the more specific class.

Consider the following statements:

```
class yClass
{
public:
    void one();
    void two(int, int);
    yClass();

private:
    int a;
    int b;
};
```

```
class xClass: public yClass
{
public:
    void one();
    xClass();

private:
    int z;

};
```

Suppose the following statements are in a user program (client code):

```
yClass y;
xClass x;
```

a. The `private` members of `yClass` are `public` members of `xClass`. True or False?

b. Mark the following statements as valid or invalid. If a statement is invalid, explain why.

i.
```
void yClass::one()
{
    cout << a + b << endl;
}
```

ii.
```
y.a = 15;
x.b = 30;
```

iii.
```
void xClass::one()
{
    a = 10;
    b = 15;
    z = 30;
    cout << a + b + z << endl;
}
```

iv.
```
cout << y.a << " " << y.b << " " << x.z << endl;
```

a) False
b) (i) Valid.
b) (ii) Invalid--- Both variable are private.
b) (iii) Invalid--- a and b are private
b) (iv) Invalid--- private data cannot be accessed by the main.

PROGRAMMING TASKS

Task 1:

Imagine a publishing company that markets both book and audiocassette versions of its works. Create a class publication that stores the title (a string) and price (type float) of a publication. From this class derive two classes: book, which adds a page count (type int), and tape, which adds a playing time in minutes (type float). Each of these three classes should have a getdata() function to get its data from the user at the keyboard, and a putdata() function to display its data.
Write a main() program to test the book and tape classes by creating instances of them, asking the user to fill in data with getdata(), and then displaying the data with putdata().

Start with the publication, book, and tape classes of task1. Add a base class sales that holds an array of three floats so that it can record the dollar sales of a particular publication for the last three months. Include a getdata() function to get three sales amounts from the user, and a putdata() function to display the sales figures. Alter the book and tape classes so they are derived from both publication and sales. An object of class book or tape should input and output sales data along with its other data. Write a main() function to create a book object and a tape object and exercise their input/output capabilities.

4. Assume that the publisher in task1 and 3 decides to add a third way to distribute books: on computer disk, for those who like to do their reading on their laptop. Add a disk class that, like book and tape, is derived from publication. The disk class should incorporate the same member functions as the other classes. The data item unique to this class is the disk type: either CD or DVD. You can use an enum type to store this item. The user could select the appropriate type by typing c or d.

Task 2:

a-Define the class bankAccount to store a bank customer's account number and balance. Suppose that account number is of type int, and balance is of type double. Your class should, at least, provide the following operations: set the account number, retrieve the account number, retrieve the balance, deposit and withdraw money, and print account information. Add appropriate constructors.

b- Every bank offers a checking account. Derive the class checkingAccount from the class bankAccount This class inherits members to store the account number and the balance from the base class. A customer with a checking account typically receives interest, maintains a minimum balance,and pays service charges if the balance falls below the minimum balance. Add member variables to store this additional information.

In addition to the operations inherited from the base class, this class should provide the following operations: set interest rate, retrieve interest rate, set minimum balance, retrieve minimum balance, set service charges, retrieve service charges, post interest, verify if the balance is less than the minimum balance, write a check, withdraw
(override the method of the base class), and print account information.
Add appropriate constructors.

c- Every bank offers a savings account. Derive the class savingsAccount from the class bankAccount (designed in part(a)). This class inherits members to store the account number and the balance from the base class. A customer with a savings account typically receives interest, makes deposits, and withdraws money. In addition to the operations inherited from the base class, this class should provide the following operations: set interest rate, retrieve interest rate, post interest, withdraw (override the method of the base class), and print account information. Add appropriate constructors.

d- Write a program to test your classes designed in parts (b) and (c).