Object Oriented Paradigm

Lab Manual (Lab 10)

Session: Spring 2016

Faculty of Information Technology

UCP Lahore Pakistan

TASK 1:

Create a base class, called BankAccount, and two additional classes (each derived from BankAccount), called SavingsAccount and CheckingAccount.

BankAccount:

- Title
- AccountNumber
- Balance
- Deposit()
- Withdraw()

SavingAccount:

- InterestRate
- CalculateInterest()

CheckingAccount:

- fee charged per transaction

Class CheckingAccount should redefine member functions withdraw and deposit so that they subtract the fee from the account balance whenever either transaction is performed successfully.

You will then test the operations of each class in function main() to simulate the transactions of both a checking account and a savings account.

Note :

In main you can only create a pointer of BankAccount type to call functions of both parent and child classes.

For example :

BankAccount * obj = new BankAccount();

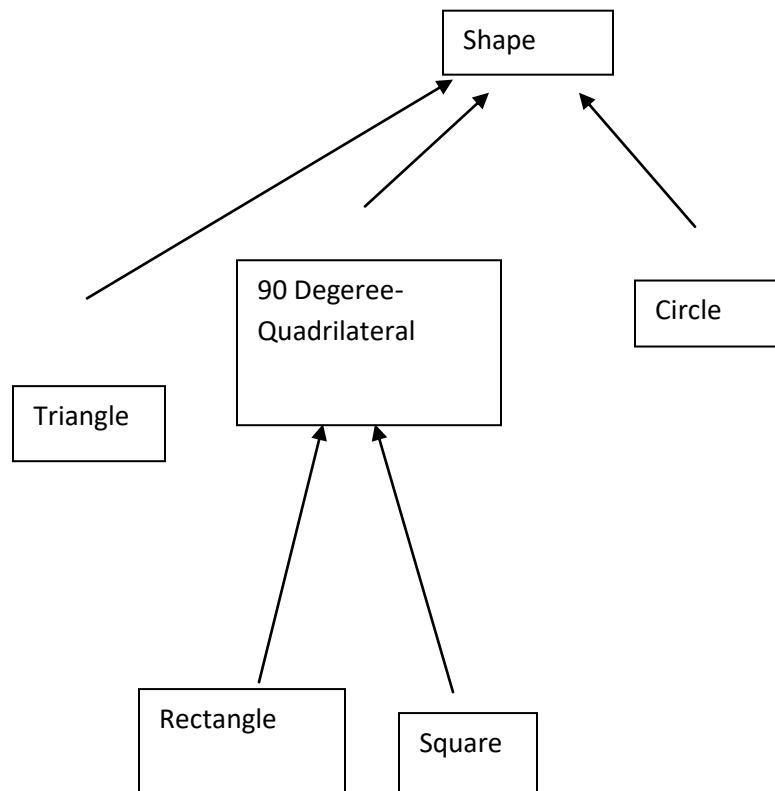BankAccount * obj1 = new CheckingAccount();

.

TASK 2:

Implement the following class hierarchy. Write functions for calculateArea to calculate area of each object of any shape.

Shape class has also calculateArea function which returns area of earth.

```
                          ┌──────────┐
                          │  Shape   │
                          └──────────┘
                         ↗    ↑    ↖

        ┌──────────────────┐        ┌──────────┐
        │ 90 Degeree-      │        │  Circle  │
        │ Quadrilateral    │        └──────────┘
        └──────────────────┘
  ┌──────────┐    ↑   ↖
  │ Triangle │
  └──────────┘

     ┌────────────┐      ┌──────────┐
     │ Rectangle  │      │  Square  │
     └────────────┘      └──────────┘
```

Note: Think about attributes and functions required for each class.

```
int main(){

Shapes *s[3];


s[0]= new Circle(0,10,"red",3);//x,y, color, radius

s[1]=new Triangle(12,30, "black",12,14);//x,y,color,base,height

s[2]=new Rectangle(20,30,"green",20,20);//x,y,color,length,width

cout<<s[0].calculateArea();

cout<<s[0].showAttributeValues();

cout<<s[1].calculateArea();

cout<<s[1]. showAttributeValues ();

cout<<s[2].calculateArea();

cout<<s[2]. showAttributeValues ();

Shape obj;

cout<<obj.calculateArea();

}
```