-: OOP LAB-12:-

12/11/2021

NAME : RAJSHREE VATSA
ROLL : 2006508

1) PROGRAM STATEMENT:-

Create a class shape. Derive three classes from it; Circle, Square and Triangle. Find area of each shape and display it, using virtual function.

PROGRAM CODE:-

```
#include<iostream>
#include<cstring>
using namespace std;
class Shape
{
	public:
	virtual void get_input()
	{
		cout<<"Shape's input fn called";
	}
	virtual void area()
	{
		cout<<"Shape's area";
	}
};
class Circle:public Shape
{
	int radius;
	public:
	void get_input()
	{
		cout<<"Enter radius of circle: ";
		cin>>radius;
	}
	void area()
	{
		cout<<"\nArea of Circle is:"<<3.14*radius*radius<<endl;
	}
};


class Triangle:public Shape
{
	int b,h;
	public:
	void get_input()
	{
		cout<<"Enter base of triangle: ";
		cin>>b;
```

```
            cout<<"Enter height of triangle: ";
            cin>>h;
        }
        void area()
        {
            cout<<"Area of triangle is: "<<0.5*h*b<<endl;
        }
};
class square:public Shape
{
        int l;
        public:
        void get_input()
        {

            cout<<"Enter length of square ";
            cin>>l;
        }
        void area()
        {
            cout<<"Area of square is: "<<l*l<<endl;
        }
};

int main()
{
        Shape *p1,*p2,*p3;
        Circle c;
        Triangle t;
        square r;
        p1=&c;
        p2=&t;
        p3=&r;
        p1->get_input();
        p2->get_input();
        p3->get_input();
        p1->area();
        p2->area();
        p3->area();
}
```

PROGRAM OUTPUT:-

```
Enter radius of circle: 4
Enter base of triangle: 2
Enter height of triangle: 3
Enter length of square 5

Area of Circle is:50.24
Area of triangle is: 3
Area of square is: 25
```

2) PROGRAM STATEMENT:-

Create a class which stores employee name,id and salary Derive two classes from 'Employee' class: 'Regular' and 'Part-Time'. The 'Regular' class stores DA, HRA and basic salary. The 'Part-Time' class stores the number of hours and pay per hour. Calculate the salary of a regular employee and a par-time employee, using pure virtual function.

PROGRAM CODE:-

```
#include <iostream>
using namespace std;
class employee
{
public:
	char name[25];
	int id, salary, DA, HRA, hr, pph;
	void info()
	{
		cout << "Enter name : ";
		cin >> name;
		cout << "Enter ID : ";
		cin >> id;
	}
	void regular()
	{
		cout << "Enter salary : ";
		cin >> salary;
		cout << "Enter DA : ";
		cin >> DA;
		cout << "Enter HRA : ";
		cin >> HRA;
	}
	void part()
	{
		cout << "Enter number of hours : ";
		cin >> hr;
		cout << "Enter pay per hour : ";
		cin >> pph;
	}
	virtual void sal() = 0;
};

class regular : public employee
{
public:
	void sal()
	{
		cout << "\nSalary of regular employee : " << salary + DA + HRA << endl;
	}
};
class part : public employee
{
public:
```

```
        void sal()
        {
                cout << "\nSalary of Part-time employee : " << pph * hr * 30 << endl;
        }
};
int main()
{
        regular r;
        employee *er = &r;
        er->info();
        er->regular();
        er->sal();
        part p;
        employee *ep = &p;
        ep->info();
        ep->part();
        ep->sal();

        return 0;
}
```

PROGRAM OUTPUT:-

```
Enter name : Sourish
Enter ID : 2006143
Enter salary : 100000
Enter DA : 25
Enter HRA : 255

Salary of regular employee : 100280
Enter name : ss
Enter ID : 33433
Enter number of hours : 4
Enter pay per hour : 200

Salary of Part-time employee : 24000
```

3) PROGRAM STATEMENT:-

Create a class which stores account number, customer name and balance. Derive two classes from 'Account' class: 'Savings' and 'Current'. The 'Savings' class stores minimum balance. The 'Current' class stores the over-due amount. Include member functions in the appropriate class for

-deposit money

-withdraw [For saving account minimum balance should be checked.]

[For current account overdue amount should be calculated.]

-display balance

Display data from each class using virtual function.

PROGRAM CODE:-

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

```
class account
{
public:
	int acn, balance, minbal, wd, dp, bal;
	char name[25];
	void info()
	{
		cout << "Enter account number : ";
		cin >> acn;
		cout << "Enter name : ";
		cin >> name;
		cout << "Enter balance : ";
		cin >> balance;
		cout << "Enter amount to withdraw : ";
		cin >> wd;
		cout << "Enter amount to deposit : ";
		cin >> dp;
	}
	void savings()
	{
		minbal = 1000;
		bal = balance - wd + dp;
		cout << "Minimum balance is : " << minbal << endl;
	}

	void current()
	{
		bal = balance - wd + dp;
		cout << "Current balance is : " << bal << endl;
	}

	virtual void data() = 0;
};
class savings : public account
{
public:
	void data()
	{
		cout << "Account number : " << acn << endl;
		cout << "Customer name : " << name << endl;
		if (bal < minbal)
			cout << "You cannow withdraw below minimum balance, which is Rs. " << minbal << endl;
		else
			cout << "Balance is : " << bal << endl;
	}
};
class current : public account
{
public:
	void data()
	{
		cout << "Account number : " << acn << endl;
```

```
            cout << "Customer name : " << name << endl;
            if (bal < 0)
            {
                  cout << "Amount Overdued." << endl;
            }
            else
            {
                  cout << "Balance is : " << bal << endl;
            }
      }
};

int main()
{
      int ch;
      savings s;
      account *as = &s;

      current c;
      account *ac = &c;

      while (1)
      {
            cout << "1. Savings" << endl;
            cout << "2. Current" << endl;
            cout << "3. Exit" << endl;

            cout << "Enter choice : ";
            cin >> ch;

            switch (ch)
            {
            case 1:
                  cout << "Savings Account." << endl;
                  as->info();
                  as->savings();
                  as->data();
                  break;

            case 2:
                  cout << "Current Account." << endl;

                  ac->info();
                  ac->current();
                  ac->data();
                  break;

            case 3:
                  return 0;
            default:
                  cout << "Wrong Choice!!" << endl;
            }
```

```
    }
}
```

PROGRAM OUTPUT:-

```
1. Savings
2. Current
3. Exit
Enter choice : 1
Savings Account.
Enter account number : 100
Enter name : Sourish
Enter balance : 10000
Enter amount to withdraw : 5000
Enter amount to deposit : 2000
Minimum balance is : 1000
Account number : 100
Customer name : Sourish
Balance is : 7000
1. Savings
2. Current
3. Exit
Enter choice : 2
Current Account.
Enter account number : 100
Enter name : Sourish
Enter balance : 50000
Enter amount to withdraw : 10000
Enter amount to deposit : 5000
Current balance is : 45000
Account number : 100
Customer name : Sourish
Balance is : 45000
1. Savings
2. Current
3. Exit
Enter choice : 3
```

4) PROGRAM STATEMENT:-

Write a program to demonstrate the use of virtual destructors.

PROGRAM CODE:-

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

class Base{
public:
   Base(){
      cout << "Base Constructor \n";
   }

   //virtual destructor
```

```
    virtual ~Base(){
        cout << "Base Destructor \n";
    }

};

class Derived: public Base{
public:
    int *n;
    Derived(){
        cout << "Derived Constructor \n";
        n = new int(10);
    }

    void display(){
        cout<< "Value: "<< *n << endl;
    }

    ~Derived(){
        cout << "Derived Destructor \n";
        delete(n);
    }
};

int main()
{
  Base *obj = new Derived();
  delete(obj);
  return 0;
}
```

PROGRAM OUTPUT:-

```
Base Constructor
Derived Constructor
Derived Destructor
Base Destructor
```