**Assignment-32 Solution Name: Om Pant**

Overriding, overloading, constructor in inheritance

1. Create a class FLOAT that contains one float data member. Overload all the four arithmetic operators so that they can operate on the objects of FLOAT.

Ans-

// 1. Create a class FLOAT that contains one float data member. Overload all the four arithmetic operators so that they can operate on the objects of FLOAT.

#include<iostream>

using namespace std;

class FLOAT{

    public:

        float value;

    FLOAT(){

    }

    FLOAT(float x){

        value = x;

    }

    //Overloading Extraction Operator

    friend ostream & operator<<(ostream &c, FLOAT f){

        cout<<f.value;

    }

    //Overloading + operator

    FLOAT operator+(FLOAT temp){

        return this->value + temp.value;

    }

    //Overloading - operator

    FLOAT operator-(FLOAT temp){

        return this->value - temp.value;

    }

    //Overloading \* operator

    FLOAT operator\*(FLOAT temp){

        return this->value \* temp.value;

    }

    //Overloading / operator

    FLOAT operator/(FLOAT temp){

        return this->value / temp.value;

    }

};

int main(){

    FLOAT f1(5.5), f2(3.2),f3(1.1);

    FLOAT sum,diff,mul,div;

    sum = f1 + f2 + f3;

    cout<<"Sum is            : "<<sum<<endl;

    diff = f1 - f2 ;

    cout<<"Difference is     : "<<diff<<endl;

    mul = f1 \* f3;

    cout<<"Multiplication is : "<<mul<<endl;

    div = f1 / f3;

    cout<<"Divison is        : "<<div<<endl;

    return 0;

}

1. Define a class Rectangle and overload area function for different types of data type.

Ans-

// 2. Define a class Rectangle and overload area function for different types of data type.

#include<iostream>

using namespace std;

class Rectangle{

    public:

    float len, bred;

    Rectangle(int l, int b){

        len = l;

        bred = b;

    }

};

    void area(int l, int b){

        cout<<"Area : "<<l\*b<<endl;

    }

    void area(int l, float b){

        cout<<"Area : "<<l\*b<<endl;

    }

    void area(float l, int b){

        cout<<"Area : "<<l\*b<<endl;

    }

    void area(float l, float b){

        cout<<"Area : "<<l\*b<<endl;

    }

int main(){

    Rectangle r1(5,6), r2(10.2, 5), r3(5,3.1), r4(1.2,3.2);

    cout<<"Area of R1: ";

    area(r1.len, r1.bred);

    cout<<"Area of R2: ";

    area(r2.len, r2.bred);

    cout<<"Area of R3: ";

    area(r3.len, r3.bred);

    cout<<"Area of R4: ";

    area(r4.len, r4.bred);

    return 0;

}

1. Define a base class Animals having member function sound() . Define another derived class from Animals class named Dogs. You need to override the sound function of the base class in the derived class.

Ans-

// 3. Define a base class Animals having member function sound() . Define another derived  class from Animals class named Dogs. You need to override the sound function of the  base class in the derived class.

#include<iostream>

using namespace std;

class Animal{

    public:

    void sound(){

        cout<<"Animal Sound"<<endl;

    }

};

class Dog:public Animal{

    public:

    void sound(){

        cout<<"Dog Sound"<<endl;

    }

};

int main(){

    Animal a;

    Dog d;

    a.sound();

    d.sound();

    return 0;

}

1. Define a class Addition that can add 2 or 3 numbers of different data types using unction overloading.

Ans-

// 4. Define a class Addition that can add 2 or 3 numbers of different data types using  function overloading.

#include<iostream>

using namespace std;

class Addition{

    public:

    void add(int n1, int n2){

        cout<<n1+n2<<endl;

    }

    void add(int n1, float n2){

        cout<<n1+n2<<endl;

    }

    void add(float n1, int n2){

        cout<<n1+n2<<endl;

    }

    void add(float n1, float n2){

        cout<<n1+n2<<endl;

    }

    void add(int n1, int n2, int n3){

        cout<<n1+n2+n3<<endl;

    }

    void add(float n1, int n2, int n3){

        cout<<n1+n2+n3<<endl;

    }

    void add(int n1, float n2, int n3){

        cout<<n1+n2+n3<<endl;

    }

    void add(int n1, int n2, float n3){

        cout<<n1+n2+n3<<endl;

    }

    void add(float n1, int n2, float n3){

        cout<<n1+n2+n3<<endl;

    }

    void add(int n1, float n2, float n3){

        cout<<n1+n2+n3<<endl;

    }

    void add(float n1, float n2, int n3){

        cout<<n1+n2+n3<<endl;

    }

    void add(float n1, float n2, float n3){

        cout<<n1+n2+n3<<endl;

    }

};

int main(){

    Addition x;

    x.add(2,5);

    x.add(2,5,8);

    return 0;

}

1. Define a class A having multiple constructors. Define another class B derived from class A. Create derived class constructors and show use of constructor in this single inheritance.

Ans-

// 5. Define a class A having multiple constructors. Define another class B derived from class  A. Create derived class constructors and show use of constructor in this single  inheritance.

// 5. Define a class A having multiple constructors. Define another class B derived from class A. Create derived class constructors and show use of constructor in this single  inheritance.

#include<iostream>

using namespace std;

class A{

    int x;

    public:

        A(){

            cout<<"Default Consturctor of Class-A is called"<<endl;

        }

        A(int a){

            x = a;

            cout<<"Parametarized constructor Of Class-A is called"<<endl;

        }

};

class B:public A

{

    public:

        B():A(){

            cout<<"Default Consturctor of Class-B is called"<<endl;

        }

};

int main(){

    A a, a2(5);

    B b;

return 0;

}

1. C++ Program to illustrate the use of Constructors in multilevel inheritance of your choice.

Ans-

// 6. C++ Program to illustrate the use of Constructors in multilevel inheritance of your choice.

#include<iostream>

using namespace std;

class A{

    public:

        A(){

            cout<<"Default Consturctor of Class-A is called"<<endl;

        }

};

class B:public A

{

    public:

        B():A(){

            cout<<"Default Consturctor of Class-B is called"<<endl;

        }

};

class C:public B{

    public:

    C():B(){

        cout<<"Default Consturctor of Class-C is called"<<endl;

    }

};

int main(){

    C c;

    return 0;

}

1. C++ Program to illustrate the use of Constructors in single inheritance of your choice.

Ans-

// 7. C++ Program to illustrate the use of Constructors in single inheritance of your choice.

#include<iostream>

using namespace std;

class A{

    public:

        A(){

            cout<<"Default Consturctor of Class-A is called"<<endl;

        }

};

class B:public A

{

    public:

        B():A(){

            cout<<"Default Consturctor of Class-B is called"<<endl;

        }

};

int main(){

    B b;

    return 0;

}

1. Write a C++ program to find the factorial of a number using copy constructor.

Ans-

// 8. Write a C++ program to find the factorial of a number using copy constructor

#include<iostream>

using namespace std;

class Factorial{

    public:

    int fact = 1;

    Factorial(int x){

        for(int i=1; i<=x;i++){

            fact \*= i;

        }

    }

    void display(){

        cout<<"Factorial : "<<fact<<endl;

    }

};

int main(){

    Factorial f(5);

    f.display();

    return 0;

}

1. Write a C++ program to calculate the area of triangle, rectangle and circle using constructor overloading. The program should be menu driven.

Ans-

// 9. Write a C++ program to calculate the area of triangle, rectangle and circle using constructor overloading. The program should be menu driven.

#include<iostream>

using namespace std;

class Shape{

    public:

    int cirArea,recArea,triArea;

        Shape(int r){

            cirArea = 3.14\*r\*r;

        }

        Shape(int l, int b){

            recArea = l\*b;

            triArea = 0.5\*l\*b;

        }

};

int main(){

    int ch;

    cout<<"1 - Area of Circle"<<endl;

    cout<<"2 - Area of Rectangle"<<endl;

    cout<<"3 - Area of Triangle"<<endl;

    cout<<"Enter Your Choice: ";

    cin>>ch;

    switch (ch)

    {

        case 1:

            {int rad;

            cout<<"Enter Radius of Circle: ";

            cin>>rad;

            Shape s(rad);

            cout<<"Area : "<<s.cirArea<<endl;

            break;}

        case 2:

            {int l,b;

            cout<<"Enter length and Bredth of Rectangle: ";

            cin>>l>>b;

            Shape s2(l,b);

            cout<<"Area : "<<s2.recArea<<endl;

            break;

}

        case 3:

            {int bh,h;

            cout<<"Enter Base and height of Triangle: ";

            cin>>bh>>h;

            Shape s3(bh,h);

            cout<<"Area : "<<s3.triArea<<endl;

            break;

        }

        default:

            cout<<"Invalid Choice"<<endl;

            break;

        }

    return 0;

}

10.Create a C++ class for player objects with the following attributes: player no., name, number of matches and number of goals done in each match. The number of matches varies for each player. Write a parameterized constructor which initializes player no., name, number of matches and creates an array for number of goals and number of matches dynamically.

Ans-

// 10.Create a C++ class for player objects with the following attributes: player no., name, number of matches and number of goals done in each match. The number of matches varies for each player. Write a parameterized constructor which initializes player no., name, number of matches and creates an array for number of goals and number of matches dynamically.

#include<iostream>

using namespace std;

class Player{

    int player\_no;

    string name;

    int no\_of\_matches,no\_of\_goals;

    int \*matches;

    int \*goals;

    public:

        Player(int no, string name ,int matches, int goals){

            this->player\_no = no;

            this->name = name;

            this->no\_of\_matches = matches;

            this->no\_of\_goals = goals;

            this->goals = new int[goals];

            this->matches = new int[matches];

        }

        void setMatches(){

            cout<<"Enter "<<this->name<<"'s Matches --"<<endl;

            cout<<"Enter Matches :";

            for(int i=0; i<this->no\_of\_matches;i++){

                cin>>\*(matches + i);

            }

        }

        void setGoals(){

            cout<<"Enter "<<this->name<<"'s Goals --"<<endl;

            cout<<"Enter Goals  :";

            for(int i=0; i<this->no\_of\_goals;i++){

                cin>>\*(goals + i);

            }

        }

        void display(){

            cout<<"Player No   : "<<this->player\_no<<endl;

            cout<<"Player Name   : "<<this->name<<endl;

            cout<<"Goals    : [ ";

            for(int i=0; i<this->no\_of\_goals;i++){

                cout<<\*(goals + i)<<" , ";

            }

            cout<<" ]"<<endl;

            cout<<"Matches    : [ ";

            for(int i=0; i<this->no\_of\_matches;i++){

                cout<<\*(matches + i)<<" , ";

            }

            cout<<" ]"<<endl<<endl;

        }

};

int main(){

    Player p (07, "MSD", 5, 5);

    p.setGoals();

    p.setMatches();

    p.display();

    return 0;

}