**Assignment -32 A Job Ready Bootcamp in C++, DSA and IOT MySirG**

**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.

Sol – 1.

#include<iostream>

using namespace std;

class FLOAT

{

private :

float a;

public :

void seta(float x)

{

a=x;

}

float geta()

{

return a;

}

FLOAT operator+(FLOAT F)

{

FLOAT temp;

temp.a=a+F.a;

return temp;

}

FLOAT operator-(FLOAT F)

{

FLOAT temp;

temp.a=a-F.a;

return temp;

}

FLOAT operator\*(FLOAT F)

{

FLOAT temp;

temp.a=a\*F.a;

return temp;

}

FLOAT operator/(FLOAT F)

{

FLOAT temp;

temp.a=a/F.a;

return temp;

}

};

int main()

{

FLOAT a1,a2;

a1.seta(5);

a2.seta(10);

cout<<"+ is "<<(a1+a2).geta()<<endl;

cout<<"- is "<<(a2-a1).geta()<<endl;

cout<<"\* is "<<(a1\*a2).geta()<<endl;

cout<<"/ is "<<(a1/a2).geta()<<endl;

return 0;

}

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

Sol – 2.

#include<iostream>

using namespace std;

class Rectangle

{

public :

int area(int l,int r)

{

return l\*r;

}

float area(int l,float r)

{

return l\*r;

}

float area(float l,float r)

{

return l\*r;

}

float area(float l,int r)

{

return l\*r;

}

};

int main()

{

Rectangle r;

cout<<r.area(5,5);

return 0;

}

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.

Sol – 3.

#include<iostream>

#include<string>

using namespace std;

class Animal

{

private :

string sound;

public :

string Sound()

{

return sound;

}

void setsound(string sound)

{

this->sound=sound;

}

};

class Dog : public Animal

{

private :

string dsound;

public :

string Sound()

{

return dsound;

}

void setsound(string sound)

{

dsound=sound;

}

};

int main()

{

Animal cat;

cat.setsound("Mew");

cout<<"Sound of Cat : "<<cat.Sound()<<endl;

Dog bruno;

bruno.setsound("Bhau-bhau");

cout<<"Sound of Dog : "<<bruno.Sound()<<endl;

return 0;

}

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

function overloading.

Sol – 4.

#include<iostream>

using namespace std;

class Addition

{

public :

int add(int a,int b)

{

return a+b;

}

int add(int a,int b,int c)

{

return a+b+c;

}

double add(int a,int b,double c)

{

return a+b+c;

}

double add(int a,double b)

{

return a+b;

}

double add(double a,int b)

{

return a+b;

}

double add(double a,double b)

{

return a+b;

}

double add(double a,double b,double c)

{

return a+b+c;

}

double add(double a,double b,int c)

{

return a+b+c;

}

double add(double a,int b,double c)

{

return a+b+c;

}

double add(int a,double b,double c)

{

return a+b+c;

}

};

int main()

{

Addition a;

cout<<a.add(5,10.5)<<endl;

cout<<a.add(2,2,2.5);

return 0;

}

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.

Sol – 5.

#include<iostream>

using namespace std;

class A

{

private :

int a;

public :

A(){}

A(int x)

{

a=x;

}

int geta()

{

return a;

}

};

class B : public A

{

private :

int b;

public :

B(){}

B(int x,int y):A(x)

{

b=y;

}

int getsum()

{

return geta()+b;

}

};

int main()

{

B b(5,10);

cout<<b.getsum();

return 0;

}

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

choice.

Sol – 6.

#include<iostream>

using namespace std;

class A

{

private :

int a=0;

public :

A(){cout<<"DC of A"<<endl;}

A(int x)

{

cout<<"PC of A"<<endl;

a=x;

}

int geta()

{

return a;

}

};

class B : public A

{

private :

int b=0;

public :

B(){cout<<"DC of B"<<endl;}

B(int x,int y):A(x)

{

cout<<"PC of B"<<endl;

b=y;

}

int getb()

{

return b;

}

};

class C : public B

{

private :

int c=0;

public :

C(){cout<<"DC of C"<<endl;}

C(int a,int b,int c):B(a,b)

{

cout<<"PC of C"<<endl;

this->c=c;

}

int getsum()

{

return geta()+getb()+c;

}

};

int main()

{

C c(4,89,7),v;

cout<<c.getsum()<<endl<<v.getsum();

return 0;

}

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

Sol – 7.

#include<iostream>

using namespace std;

class A

{

private :

int a;

public :

A(){}

A(int x)

{

a=x;

}

int geta()

{

return a;

}

};

class B : public A

{

private :

int b;

public :

B(){}

B(int x,int y):A(x)

{

b=y;

}

int getsum()

{

return geta()+b;

}

};

int main()

{

B b(5,10);

cout<<b.getsum();

return 0;

}

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

Sol – 8.

#include<iostream>

using namespace std;

class Fact

{

private :

int n,fact=1;

public :

Fact(int x)

{

n=x;

}

Fact(Fact &F)

{

n=F.n;

fact=F.fact;

}

void calculate()

{

for(int i=1;i<=n;i++)

{

fact=fact\*i;

}

}

void display()

{

cout<<"Factorial of "<<n<<" is "<<fact<<endl;

}

};

int main()

{

int x;

cout<<"Enter a no : ";

cin>>x;

Fact F1(x);

F1.calculate();

F1.display();

Fact F2(F1);

F2.display();

return 0;

}

9. Write a C++ program to calculate the area of triangle, rectangle and circle using

constructor overloading. The program should be menu driven.

Sol – 9.

#include<iostream>

#include<math.h>

using namespace std;

class Area

{

private :

float area;

public :

Area(float r)

{

area=3.14\*r\*r;

}

Area(int l,int b)

{

area=l\*b;

}

Area(int a,int b,int c)

{

float s=(a+b+c)/2.0;

area=sqrt(s\*(s-a)\*(s-b)\*(s-c));

}

float getarea()

{

return area;

}

};

int main()

{

Area Circle(5),Rectangle(9,15),Triangle(2,6,5);

cout<<"Area of Circle is "<<Circle.getarea()<<endl;

cout<<"Area of Rectangle is "<<Rectangle.getarea()<<endl;

if(Triangle.getarea()==0)

{

cout<<"Not a triangle"<<endl;

}

else

cout<<"Area of triangle is "<<Triangle.getarea()<<endl;

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 subjects and creates an array for number of goals and

number of matches dynamically.

Sol – 10.

#include<iostream>

#include<string>

using namespace std;

class Player

{

private :

int no,nom;

int \*nog;

string name;

public :

Player(int pn,int nom,string nm)

{

no=pn;

this->nom=nom;

name=nm;

nog=new int[nom];

}

void getgoals()

{

for(int i=0;i<nom;i++)

{

cout<<"Enter no of goals in match "<<i+1<<" : ";

cin>>nog[i];

}

}

void display()

{

cout<<"-----------------------------"<<endl;

cout<<"\*\*\*\*\*\*PLAYER INFO\*\*\*\*\*\*"<<endl;

cout<<"-----------------------------"<<endl;

cout<<"Play Number : "<<no<<endl;

cout<<"Player Name : "<<name<<endl;

cout<<"No of matches : "<<nom<<endl;

for(int i=0;i<nom;i++)

{

cout<<"No of goals in match "<<i+1<<" : "<<nog[i]<<endl;

}

}

};

int main()

{

Player P(11,5,"Chris Ronaldo");

P.getgoals();

P.display();

return 0;

}