**Introduction**

**Class, Attributes, Setter & Getter:**

Class Student

{

Int student\_id;

String Student\_name;

String Student\_address;

Public:

Student()

{

Student\_id = 0;

Student\_name = “not set”;

Student\_address = “not set”;

}

Student(int id, string name, string address)

{

Student\_id = id;

Student\_name = name;

Student\_address = address;

}

//////////////////////Setter////////////////////////////////

Void set\_student\_id(int id)

{

Student\_id = id;

}

Void set\_student\_name(string name)

{

Student\_name = name;

}

Void set\_student\_address(string address)

{

Student\_address = address;

}

//////////////////////Getter////////////////////////////////

Int get\_student\_id()

{

Return student\_id;

}

String get\_student\_name()

{

Return Student\_name;

}

String get\_student\_addresss;

{

Return Student\_address;

}

};

**This -> Keyword & Const Class:**

#include <iostream>

#include<stdio.h>

#include<stdlib.h>

#define TOTAL 3

using namespace std;

class Person

{

string name;

string address;

public:

Person(string name, string address)

{

this->name = name;

this->address = address;

}

Person()

{

name = "not set";

address = "not set";

}

////////////Setter/////////////

void set\_name(string name)

{

this->name = name;

}

void set\_address(string address)

{

this->address = address;

}

//////////Getter////////////////

string get\_name() const

{

return name;

}

string get\_address() const

{

return address;

}

};

int main()

{

const Person p1("Nouman","Lahore");

cout<<p1.get\_name();

}

**Adding two classes using class name function:**

#include <iostream>

using namespace std;

class Person

{

int num;

public:

Person()

{

num = 0;

}

void set\_num(int a)

{

num = a;

}

void get\_num()

{

cout<<num;

}

Person sum\_obj(Person p1, Person p2)

{

Person p3;

p3.num = p1.num + p2.num;

return p3;

}

};

int main()

{

Person p1,p2,p3;

p1.set\_num(20);

p2.set\_num(30);

p3 = p3.sum\_obj(p2,p1);

p3.get\_num();

}

**Operators Overloading**

#include <iostream>

using namespace std;

class Person

{

int num;

public:

Person()

{

num = 0;

}

void set\_num(int a)

{

num = a;

}

void get\_num()

{

cout<<num;

}

///////////////Unary Overloading////////////////

/////for Prefix ++c

void operator ++()

{

++num;

}

/////for postfix c++

void operator ++(int)

{

++num;

}

//////////////Binary Operators Overloading//////////

Person operator + (const Person& obj)

{

Person temp;

temp.num = num + obj.num;

return temp;

}

Person operator - (const Person& obj)

{

Person temp;

temp.num = num - obj.num;

return temp;

}

/////////////////Stream Overloading////////////////

friend ostream &operator <<(ostream &output, const Person &obj)

{

output<<obj.num;

return output;

}

friend istream &operator >>(istream &input, Person &obj)

{

input>>obj.num;

return input;

}

};

int main()

{

Person p1,p2,p3;

p1.set\_num(10);

p2.set\_num(30);

p1++;

++p1;

//p1.get\_num();

///////////binary/////

p3 = p2+p1;

p3.get\_num();

cout<<endl;

p3 = p1-p2;

p3.get\_num();

//////stream overloading////////

Person p4;

cout<<endl;

cout<<"Enter the number is class : ";

cin>>p4;

cout<<"The number in class is : "<<p4<<endl;

}

**Static Variable, Object and Functions**

#include <iostream>

using namespace std;

class Student

{

int id;

string name;

static int count;

public:

Student()

{

id = 0;

name = "not set";

count++;

}

Student(int id, string name)

{

this->id = id;

this->name = name;

}

static int get\_count()

{

return count;

}

void display()

{

cout<<"Student ID : "<<id<<endl;

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

}

};

int Student:: count = 0;

void fun()

{

static int i =0;

i++;

cout<<"The value of i is : "<<i<<endl;

}

int main()

{

fun();

fun();

Student s1;

cout<<s1.get\_count()<<endl;

Student s2;

cout<<s2.get\_count()<<endl;

Student st1,st2,st3;

cout<<Student::get\_count();

}

**Association, Aggregation & Composition**

//Composition

#include <iostream>

using namespace std;

class Location

{

int x,y;

public:

Location()

{

x = y = 0;

}

Location(int x, int y)

{

this->x = x;

this->y = y;

}

////////////setter/////////

void set\_x(int a)

{

x = a;

}

void set\_y(int a)

{

y = a;

}

//////getter///////////

int get\_x()

{

return x;

}

int get\_y()

{

return y;

}

};

class Person

{

string name;

string email;

Location obj;

public:

Person()

{

name = "NOT SET";

email = "NOT Set";

}

Person(string n,string e,Location obj)

{

name = n;

email = e;

this->obj = obj;

}

///////////setter///////////

void set\_name(string n)

{

name = n;

}

void set\_email(string e)

{

email = e;

}

void set\_location(int x, int y)

{

obj.set\_x(x);

obj.set\_y(y);

}

///////////////getter///////////

string get\_name()

{

return name;

}

string get\_email()

{

return email;

}

Location get\_location()

{

return obj;

}

void display()

{

cout<<"The name of Person is "<<name<<endl;

cout<<"The Email of Person is "<<email<<endl;

cout<<"The location of Person is "<<obj.get\_x()<<" , "<<obj.get\_y()<<endl;

}

};

int main()

{

Person p1;

p1.set\_name("Nouman");

p1.set\_email("numan@gmail.com");

p1.set\_location(20,50);

p1.display();

}

//Aggregation

#include <iostream>

using namespace std;

class Employee

{

int id;

string name;

public:

Employee()

{

id = 0;

name = "NOT SET";

}

Employee(int id, string n)

{

this->id = id;

name = n;

}

///////Setters//////

void set\_id(int id)

{

this->id = id;

}

void set\_name(string name)

{

this->name = name;

}

//////////getter/////////

int get\_id()

{

return id;

}

string get\_name()

{

return name;

}

void display()

{

cout<<"The ID of Employee is : "<<id<<endl;

cout<<"The name of Employee is : "<<name<<endl;

}

};

class University

{

int id;

string name;

Employee \*ptr;

public:

University()

{

id = 0;

name = "NOT SET";

ptr = NULL;

}

University(int id, string name, Employee \*ptr)

{

this->id = id;

this->name = name;

this->ptr = ptr;

}

///////////setter////////////

void set\_id(int id)

{

this->id = id;

}

void set\_name(string name)

{

this->name = name;

}

void set\_employee(Employee \*ptr)

{

this->ptr = ptr;

}

////////getter////////////

int get\_id()

{

return id;

}

string get\_name()

{

return name;

}

Employee\* get\_employee()

{

return ptr;

}

void display()

{

cout<<"The ID of University is : "<<id<<endl;

cout<<"The Name of University is : "<<name<<endl;

ptr->display();

}

};

int main()

{

Employee e1;

e1.set\_id(1);

e1.set\_name("Nouman");

//e1.display();

University u1;

u1.set\_id(22);

u1.set\_name("UMT");

u1.set\_employee(&e1);

u1.display();

}

**Single Inheritance**

#include<iostream>

using namespace std;

class Person

{

protected:

int id ;

string name;

public:

Person()

{

id = 0;

}

Person(int id, string name)

{

this->id = id;

this->name = name;

}

void set\_id(int id)

{

this->id = id;

}

void set\_name(string name)

{

this->name = name;

}

int get\_id()

{

return id;

}

string get\_name()

{

return name;

}

};

class Employee:public Person

{

double salary;

public:

Employee()

{

salary = 0.0;

}

Employee(int id, string name, double salary):Person(id, name)

{

this->salary = salary;

}

void display()

{

cout<<"The ID of Employee is : "<<id<<endl;

cout<<"The Name of Employee is : "<<name<<endl;

cout<<"The Salary of Employee is : "<<salary<<endl;

}

};

int main()

{

Employee e1(1,"Nouman", 5000.00);

e1.display();

}