**LAB 5 : Inheritance**

1. **Student**

Create a class called Student with two member data to represent name and roll of the student. Use member functions to read and print these data. From this class derive a class boarder with a member data to represent room number. Derive another class called DayScholar from the class Student with the member data to represent the address and bus number of the student. In both derived classes, use member functions to read and print the respective data

Ans//

/\*program to read and display respective data of parent class and derived class\*/

#include <iostream>

using namespace std;

//declaring the class

class Student{

private:

string name;

int roll\_no;

public:

//mutator functions

getData(){

cout << "Name of student : ";

cin >> name;

cout<<"Roll number of student : ";

cin>>roll\_no;

}

//display method

displayData(){

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

cout << "Roll:" << roll\_no<< endl;

}

};//end of class

//derived class

class Boarder : public Student{

private:

int room\_no;

public:

getRoomNo(){

cout << "Room Number : ";

cin >> room\_no;

}

displayRoomNo(){

cout << "Room No:" << room\_no << endl;

}

};//end of class

//declaring child class

class Dayscholar : public Student{

private:

int bus\_no;

string address;

public:

getRecord(){

cout << "\nBus Number : ";

cin >> bus\_no;

cout<<"Address of student : ";

cin>>address;

}

displayRecord(){

cout << "Bus No:" << bus\_no << endl;

cout << "Address:" << address;

}

};//end of class

//main program

int main()

{ //creating object

Boarder d;

Dayscholar ds;

//for parent class

d.getData();

d.displayData();

d.getRoomNo();

d.displayRoomNo();

//for child class

ds.getRecord();

ds.displayRecord();

}

**2. Employee and ProductionWorker**

Design a class named Employee. The class should keep the following information in member variables:

• Employee name

• Employee number

• Hire date

Write one or more constructors and the appropriate accessor and mutator functions for the class.

Next, write a class named ProductionWorker that is derived from the Employee class. The ProductionWorker class

should have member variables to hold the following information:

• Shift (an integer)

• Hourly pay rate (a double)

The workday is divided into two shifts: day and night. The shift variable will hold an integer value representing the

shift that the employee works. The day shift is shift 1 and the night shift is shift 2. Write one or more constructors

and the appropriate accessor and mutator functions for the class. Demonstrate the classes by writing a program that

uses a ProductionWorker object.

Ans//

#include <iostream>

using namespace std;

//Declaring class

class Employee{

private:

string empname;

unsigned int number;

unsigned int date;

public:

Employee(){ //default constructor initializing the values

empname = " ";

number = 0;

date = 0;

}

getEmployeeData(){

cout << " Name of the Employee : ";

cin >> empname;

cout<<"Phone Number : ";

cin >> number;

cout<<" Hire Date : ";

cin>>date;

}

displayEmployeeData(){

cout << "\nName : " << empname << endl;

cout << "Number : " << number << endl;

cout << "Date of hire : " << date<<endl;

}

};//end of base class

//Declaring derived class

class ProductionWorker : public Employee{

private:

int shift;

double hourlyPayRate;

public:

ProductionWorker(){ //default constructor initializing the values of variales

shift = 0;

hourlypayRate = 0.00;

}

getData(){

cout << " Shift (1 for day 2 for night) : ";

cin >> shift;

cout << " Hourly Pay Rate : ";

cin >> hourlyPayRate;

}

displayData(){

if(shift == 1)

{

cout << "Shift :" <<"Day\n";

cout << "Hourly Pay Rate :" << hourlyRate;

}

else if(shift == 2)

{

cout << "Shift :" << "Night\n";

cout << "Hourly Pay Rate :" << hourlyRate;

}

}

};//end of child class

//main program

int main()

{

//Creating the object of derived class

ProductionWorker pd;

// calling get function from base

pd.getEmployeeData();

// calling get function from base

pd.getData();

// calling display function from base

pd.displayEmployeeData();

// calling display function from base

pd.displayData();

return 0;

}

**3. Person Data**

Design a class named PersonData with the following member variables:

• lastName

• firstName

• address

• city

• state

• zip

• phone

Write the appropriate accessor and mutator functions for these member variables. Next, design a class named

CustomerData, which is derived from the PersonData class. The CustomerData class should have the following

member variables:

• customerNumber

• mailingList

The customerNumber variable will be used to hold a unique integer for each customer. The mailingList variable

should be a bool. It will be set to true if the customer wishes to be on a mailing list, or false if the customer does

not wish to be on a mailing list. Write appropriate accessor and mutator functions for these member variables.

Demonstrate an object of the CustomerData class in a simple program.

Ans//

#include <iostream>

using namespace std;

//Declaring class

class PersonData{

private:

string fname;

string lname;

string address;

string city;

string state;

int zip;

long phone;

public:

void getData(){

cout << "Enter first name : ";

cin >> fname;

cout << "Enter the last name : ";

cin >> lname;

cout << "Enter the address : ";

cin >> address;

cout << "Enter the city : ";

cin >> city;

cout << "Enter the state : ";

cin >> state;

cout << "Enter the zip code : ";

cin >> zip;

cout << "Enter the phone number : ";

cin >> phone;

}

void DisplayData(){

cout << "First Name:" << fname << endl;

cout << "Last Name:" << lname << endl;

cout << "Address:" << address << endl;

cout << "City:" << city << endl;

cout << "State:" << state << endl;

cout << "Zip Code:" << zip << endl;

cout << "Phone No:" << phone ;

}

};//end of base class

//Derived class

class CustomerData : public PersonData{

private:

int customerNumber;

bool mailList;

public:

void getInformation(){

cout << "Enter Customer Number : ";

cin >> customerNumber;

cout << " Mailing List (1 for yes 0 for no) :";

cin >> mailList;

}

void displayInfo(){

cout << "\n Enter Custom Number : " << customerNumber;

if (mailList == 1)

cout << "\nMaillist : "<< true;

else if(mailList == 2)

cout << "\nMaillist :" << false;

else

cout<<"wrong input!! : ";

}

};//end of class

int main()

{ // creating the object of derived class

CustomerData cd;

//calling get function to get data

cd.getData();

cd.getInformation();

//calling display function

cd.DisplayData();

cd.displayInfo();

return 0;

}

**4. Ship, CruiseShip, and CargoShip Classes**

Design a Ship class that has the following members:

• A member variable for the name of the ship (a string)

• A member variable for the year that the ship was built (a string)

• A constructor and appropriate accessors and mutators

• A virtual print function that displays the ship’s name and the year it was built.

Design a CruiseShip class that is derived from the Ship class. The CruiseShip class should have the following

members:

• A member variable for the maximum number of passengers (an int)

• A constructor and appropriate accessors and mutators

• A print function that overrides the print function in the base class. The CruiseShip class’s print function should

display only the ship’s name and the maximum number of passengers.

Design a CargoShip class that is derived from the Ship class. The CargoShip class should have the following

members:

• A member variable for the cargo capacity in tonnage (an int).

• A constructor and appropriate accessors and mutators.

• A print function that overrides the print function in the base class. The CargoShip class’s print function should

display only the ship’s name and the ship’s cargo capacity.

Demonstrate the classes in a program that has CruiseShip, and CargoShip objects. The program should then step

through the objects, calling each object’s print function.

Ans//

#include <iostream>

#include <string>

using namespace std;

//declaring the class

class Ship{

protected:

string name;

int modelYear;

public:

//default constructor

Ship()

{

name = "";

modelYear = 1990;

}

//Parameterised constructor

Ship(string n, int y)

{

name = n;

modelYear = y;

}

//accessor functions

string getName()

{

return name;

}

int getYear()

{

return modelYear;

}

//mutator functions

void setName(string n)

{

name = n;

}

void setYear(int y)

{

modelYear = y;

}

//virtual print() function

virtual void print()

{

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

cout << "Build year: " << modelYear<< endl;

}

};//end of class

//Declaring child class

class CruiseShip : public Ship

{

private:

int maxPass;

public:

//default constructor set maximum number of passengers to 0

CruiseShip() : Ship()

{

maxPass = 0;

}

//Parameterised constructor

CruiseShip(int maxNum, string n, int y) :

Ship(n, y)

{

maxPass = maxNum;

}

//accessor function

int getMaxPassengers()

{

return maxPass;

}

//mutator function

void setMaxPassengers(int maxNum)

{

//validate input using while loop

if(maxNum<=0){

cout << " number of passengers must be greateer than zero";

}

else

maxPass = maxNum;

}

//override print() function

virtual void print()

{

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

cout << "Maximum Passenger(in number): " << maxPass << endl;

}

};//end of child class

//Declaring another child class

class CargoShip : public Ship{

private:

int maximum\_Tonnage;

public:

CargoShip() : Ship(){

maximum\_Tonnage = 0;

}

//overloaded constructor

CargoShip(int maxCapacity, string n, int y) :

Ship(n, y){

if(maxCapacity <= 0){

cout << "Invalid Capacity!! Maximum capacity must be greater than Zero";

exit(0);

}

else

maximum\_Tonnage = maxCapacity;

}

//accessor function

int getCapacity()

{

return maximum\_Tonnage;

}

//mutator function

void setCapacity(int maxCapcity)

{

while(maxCapcity < 0)

{

cout << "Maximum capacity cannot be negative number!";

cout << " Enter again: ";

cin >> maxCapcity;

}

maximum\_Tonnage = maxCapcity;

}

virtual void print()

{

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

cout << "Max Capacity: " << maximum\_Tonnage << endl;

}

};//end of child class

//main program

int main()

{

cout << "Creating Array of Ship Pointers To Reference Ship, CruiseShip and CargoShip objects...\n";

Ship \*ships[3] = {

new Ship(),

new CruiseShip(),

new CargoShip()

};//declaring the array

cout << "Displaying the data for each ship...\n";

//mutator functions to store new data into the objects

ships[0]->setName("TYCON\_19");

ships[0]->setYear(2008);

//using pointer to the derived class to access

CruiseShip \*cruiseship = static\_cast<CruiseShip\*>(ships[1]);

cruiseship->setName("TITNIC");

cruiseship->setMaxPassengers(1990);

//using pointer to the derived class to access

CargoShip \*cargoship = static\_cast<CargoShip\*>(ships[2]);

cargoship->setName("HAMMERHEAD CARGO");

cargoship->setCapacity(200000);

//using while to display

int i = 0;

while(i<3){

ships[i]->print();

i++;

cout << endl;

}

return 0;

}

**5. Teaching Assistant**

In the scenario below, a Teaching Assistant derives traits from Student and Employee class and Student derives

traits from its parent Engineering Stream class. Assume necessary data members and member function so that

information regarding a particular Teaching Assistant can be stored and retrieved when desired.

Write a program to implement the concept of hybrid inheritance to find the result of the student (consider there

are four classes : Student, Test, Sports and Result and assume the necessary data as well as functions)

Ans//

#include <iostream>

using namespace std;

//Declaring class

class Engineering{

protected:

string name;

public:

Engineering(){} //default constructor

void getName(){

cout << "Enter the name : ";

cin >> name;

}

};//end of base class

//declaring derived class

class Student : public Engineering{

protected:

float percentage;

public:

Student(){} //constructor

void getPer(){

cout << "Enter the percentage of Student : ";

cin >> percentage;

}

};//end of derived class

//Declaring another base class

class Teacher{

protected:

int year;

public:

Teacher(){} // default constructor

void getYear(){

cout << "Teaching Experience(in years) : ";

cin >> year;

}

};//end of base class

//Declaring derived class from two classes multiple inheritance

class TeacherAssist : public Student, public Teacher{

public:

TeacherAssist(){} //constructor

void Display(){

cout << "Name Of Teacher :" << name << endl;

cout << "Percentage Obtained :" << percentage << endl;

cout << "Worked Experience :" << year << " years";

}

};//end of derived class

//main program

int main()

{ //creating object of derived class

TeacherAssist t;

//Getting data

t.getName();

t.getPer();

t.getYear();

//displaying the data

t.Display();

return 0;

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*