```
/*
Lab 06
Question 1(a)
Default constructor, parameterized constructor, destructor
*/

#include<iostream>
using namespace std;
//define class
class MasterStudent
{       //private data members
        string name, title;
        int status;
        public: //public member functions
        //parameterized constructor
        MasterStudent(string n, string t, int x)
        {       name = n;
                title = t;
                status = x;
        }
```

Values from the main function is passed to parameterized constructor

```
        //default constructor
        MasterStudent()
        {       name = "Peter";
                title = "A Study on the Usability Factors of Mobile Apps.";
                status = 1;
        }
```

Constructor has the same name as class, default constructor is not parameterized

```
        //accessor functions
        int getStatus()
        {       return status;
        }
        string getName()
        {       return name;
        }
        string getTitle()
        {       return title;
        }
        //destructor
        ~MasterStudent()
        {       cout<<"\n\n~End of Details~Student~"<<name<<endl;
        }
};

int main()
{       MasterStudent MS1; //initialize class object using default
constructor
        cout<<"=================================="<<endl;
        cout<<" Masters Student Details "<<endl;
        cout<<"=================================="<<endl;
        //access and show private data members
        cout<<"Name \t: "<<MS1.getName()<<endl;
        cout<<"Title \t: "<<MS1.getTitle()<<endl;
        cout<<"Status \t: ";
        //check and print status
        if(MS1.getStatus() ==1) cout<<"Approved"<<endl;
        else cout<<"Pending"<<endl;
//Create another object passing the values
//"Aliana Mahmud" for name, "Customer Satisfaction towards Green Products"
for title, 0 for status
//Use the cout statements given earlier to display the content of the new
object.
//refer to label X for the output to be displayed
```

destructors are called in reverse order of their creation at program termination as part of the program's cleanup process. That explains why memory for MS2 class object is deallocated before MS1 class object.

```
        MasterStudent MS2("Aliana Mahmud","Customer Statisfaction towards 
Green Products",0);
        cout<<"=================================="<<endl;
        cout<<" Masters Student Details "<<endl;
        cout<<"=================================="<<endl;
        cout<<"Name \t: "<<MS2.getName()<<endl;
        cout<<"Title \t: "<<MS2.getTitle()<<endl;
        cout<<"Status \t: ";
        if(MS2.getStatus() ==1) cout<<"Approved"<<endl;
        else cout<<"Pending"<<endl;
}

//————————————————————————————————————————————————————————————————————————
/*
Lab 06
Question 1(b)
Constructor and destructor with array of objects
*/

#include<iostream>
using namespace std;
//define class
class MasterStudent
{       //private data members
        string name, title;
        int status;
        public: //public member functions
        //parameterized constructor
        MasterStudent(string n, string t, int x)
        {       name = n;
                title = t;
                status = x;
        }
        //default constructor
        MasterStudent()
        {       name = "Peter";
                title = "A Study on the Usability Factors of Mobile Apps.";
                status = 1;
        }
        //accessor functions
        int getStatus()
        {       return status;
        }
        string getName()
        {       return name;
        }
        string getTitle()
        {       return title;
        }
        //destructor
        ~MasterStudent()
        {       cout<<"\n\n~End of Details~Student~"<<name<<endl;
        }
};
```

Change the object to array

```
int main()
{       //initialize array of 4 objects with hardcoded data
        MasterStudent MS[4] = {MasterStudent("Philip Morales", "Working with 
Generation X employees: food industry", 1),
                                    MasterStudent("Cameron
Connor","Collective Co-Creation within the Open Source Software Community",
1),
                                    MasterStudent("Meriam Miles","What
Makes Online Video Advertisements Go Viral?",0),
                                    MasterStudent("Dory Dean","Social media
use for corporate communications",0)};
      //loop to access data members of each object in array and display
details
      for(int i=0;i<4;i++)
      {
      cout<<"\n================================="<<endl;
      cout<<" Masters Student Details "<<i+1<<endl;
      cout<<"================================="<<endl;
      cout<<"Name \t: "<<MS[i].getName()<<endl;
      cout<<"Title \t: "<<MS[i].getTitle()<<endl;
      cout<<"Status \t: ";
      if(MS[i].getStatus() ==1) cout<<"Approved"<<endl;
      else cout<<"Pending"<<endl;
      }
}
```

Use for loop to access array data members

```
//————————————————————————————————————————————————————————————————————
/*
Lab 06
Question 2
Overloaded constructors and mutator functions
*/
```

Overloaded constructors mean having multiple constructors with different parameters

Mutator a.k.a setter, modify class private data members values

```
#include<iostream>
using namespace std;
//define class
class Employee
{     //private data members
      string name, department, position;
      int idNumber;
      public://public member functions
            //parameterized constructor (4 parameters)
            Employee(string n,int id, string dept, string post)
            {
                  name = n;
                  idNumber = id;
                  department = dept;
                  position = post;
            }
            //parameterized constructor (2 parameters)
            Employee(string n, int id)
            {
                  name = n;
                  idNumber = id;
                  department = "";
                  position = "";
            }
            //default constructor
            Employee()
            {
                  name = "";
                  idNumber = 0;
                  department = "";
                  position = "";
            }
```

```
                //mutator functions
                void setName(string n){ name = n; }
                void setID(int id){ idNumber = id; }
                void setDept(string dept){ department = dept; }
                void setPost(string post){ position = post; }
                //accessor functions
                string getName(){return name;}
                int getID(){return idNumber;}
                string getDept(){return department;}
                string getPost(){return position;}
};

void displayData(Employee); //additional function to display data

int main()
{       //initialize object with parameterized constructor
        Employee SM("Susan Meyers",47899,"Accounting","Vice President");
        //initialize object with parameterized constructor and setting data
with mutator functions
        Employee MJ("Mark Jones",39119);
        MJ.setDept("IT");
        MJ.setPost("Programmer");
        //initialize object with default constructor and setting data with
mutator functions
        Employee JR;
        JR.setName("Joy Rogers");
        JR.setID(81774);
        JR.setDept("Manufacturing");
        JR.setPost("Engineer");
        //display data in objects
        displayData(SM);
        displayData(MJ);
        displayData(JR);
}
//display data function
void displayData(Employee E)
{       //access and display private data of object
        cout<<"\nName: "<<E.getName()<<endl;
        cout<<"ID Number: "<<E.getID()<<endl;
        cout<<"Department: "<<E.getDept()<<endl;
        cout<<"Position: "<<E.getPost()<<endl;
}

//————————————————————————————————————————————————————————————————————————
/*
Lab 06
Question 3
Constructor, destructor, mutator for array of objects
*/

#include<iostream>
#include<iomanip>
using namespace std;
//define class
class Books
{
        private: //private data members
                string isbnNo, title, author;
                float price, discountedprice, discountperc;
        public: //public member functions
                Books(); //default constructor
```

```
            Books(string,string,string,float,float); //parameterized
constructor
            void set_Data(); //mutator function
            void calcDiscountedPrice(); //for calculation
            void print(); //for display
            float getDiscountedPrice(); //accessor function
            ~Books(); //desctructor
};


void Books::set Data() //mutator function
{      //get user inputs and set to appropriate variables
       fflush(stdin); //clear buffer
       cout<<"\nEnter ISBN\t\t: ";
       getline(cin,isbnNo);
       cout<<"Enter Title\t\t: ";
       getline(cin,title);
       cout<<"Enter Author's name\t: ";
       getline(cin,author);
       cout<<"Enter price\t\t: RM ";
       cin>>price;
       cout<<"Enter discount (%)\t: ";
       cin>>discountperc;
}

void Books::calcDiscountedPrice() //void has no return
{      //calculate price after minus discounted amount
       discountedprice = price*(100-discountperc)/100;
}

void Books::print()
{      //display book details
       cout<<"\n--------------------------------------------------------
"<<endl;
       cout<<"\t\tBook Details"<<endl;
       cout<<"--------------------------------------------------------
"<<endl;
       cout<<"ISBN\t\t: "<<isbnNo<<endl;
       cout<<"Title\t\t: "<<title<<endl;
       cout<<"Author\t\t: "<<author<<endl;
       cout<<"Original Price\t: RM "<<fixed<<setprecision(2)<<price<<endl;
       cout<<"Discounted Price: RM "<<discountedprice<<endl;
}

float Books::getDiscountedPrice(){return discountedprice;} //accessor
function

Books::Books() //default constructor
{
       isbnNo =""; title=""; author="";
       price=0.00; discountperc=0;
}

Books::Books(string n, string t, string a, float p, float d)
{      //parameterized constructor
       isbnNo = n; title = t; author = a;
       price = p; discountperc = d;
}

Books::~Books(){cout<<"\nEnjoy reading "<<title<<endl;} //destructor

//function accept object by reference using a reference object as argument
```

```
void func(Books &B)
{       //call object functions
        B.set_Data();
        B.calcDiscountedPrice();
        B.print();
}

int main()
{       //initialize object with data values
        Books B1("102009912","7 Habits of Highly Effective People",
                     "Stephen Covey",400.00,30);
        //call functions of object to calculate and display data
        cout<<"........Book of the Month..... "<<endl;
        B1.calcDiscountedPrice();
        B1.print();
        //declare array of objects
        Books B2[3];
        float expensive = 0.00, discprc;
        int below = 0;

        cout<<"\nNow we shall enter and display data for 3 special
books... "<<endl;
        //loop to call functions for array of objects
        for(int i=0;i<3;i++)
        {
                func(B2[i]); //function calls object mutator, calculate
discount, display
                discprc = B2[i].getDiscountedPrice(); //access price after
discount
                //determine most expensive book
                if(expensive<discprc)
                {
                        expensive = discprc;
                }
                //accumulate count of books with price below 30 after discount
                if(discprc<30)
                        below++;
        }
        //display most expensive book price and number of books below 30
after discount
        cout<<"\n---------------------------------------------------------
"<<endl;
        cout<<"The most expensive book is RM "<<expensive<<endl;
        cout<<"The number of books that are below RM 30 are :"<<below<<endl;
        return 0;
}

//————————————————————————————————————————————————————————————————————————
/*
Lab 07
Question 1(a)
Copy constructor
*/
```

create a new object as a copy of an existing object of the same class

```
#include<iostream>
using namespace std;
//define class
class Bags
{       //private data members
        string brand;
        float height, length, width;
```

```
      public: //public member functions
            void setdata() //mutator function to get user input
            {
                  cout<<"Enter your bag's brand name : ";
                  getline(cin, brand);
                  cout<<"Enter value length , width and height of your bag
L, W, H ";
                  cin>>length>>width>>height;
            }

            void display() //showing data members
            {
                  cout<<"\nYour brand bag name is **"<<brand<<"** and the
dimensions are: "
                  <<length<<"L "<<width<<"W "<<height<<"H "<<endl;
            }

            Bags (const Bags &Bi) //copy constructor
            {
                  brand = Bi.brand;
                  length = Bi.length
                  width = Bi.width;
                  height = Bi.height;
                  cout<<"\nDo you have the same bag??"<<endl;
            }
```

Deep copying by constant reference to an existing Bags object.
In this case, copying the default constructor object values

```
            Bags() //default constructor
            {
                  brand = "Adidas";
                  length = 35;
                  width = 20;
                  height = 45;
            }
};
int main()

{     //need to developed by adding object K, L and M;
      Bags K; //declare object
      //call object member functions
      K.setdata();
      K.display();
```

Object values entered by user inputs

```
      Bags L; //declare object
      //call object member function
      L.display();
```

Object values initialized by default constructor

```
      //declare object as copy of another object
      Bags M(L); // or M = L;
      //call object member function
      M.display();
```

A shallow copying can be done by stating “`Bags M = L;`”

```
      //observation:    First object stores data entered by user, second
object initialized by default constructor
      //                Third object initialized by copying from
another existing object, constructor initialization
      //                indicated by "same bag?" message and contains
same data as copied object
}

//——————————————————————————————————————————————————————————————————————
/*
Lab 07
Question 1(b)
Friend function
*/
```

Friend functions are not member functions of the class.
They can access and modify the private and protected members of the class using the object reference and the dot operator (e.g., obj.private_member).

Check() function, a friend function can access the private data members of Bag's objects

```
#include<iostream>
using namespace std;
//define class
class Bags
{     //private data members
      string brand;
      float height, length, width;
      public: //public member functions
            void setdata() //mutator function to get user input
            {   cout<<"Enter your bag's brand name : ";
                  getline(cin, brand);
                  cout<<"Enter value length , width and height of your bag
L, W, H ";
                  cin>>length>>width>>height;
                  //fflush(stdin);
                  cin.ignore();
            }

            void display() //showing data members
            {
                  cout<<"\nYour brand bag name is **"<<brand<<"** and the
dimensions are: "
                  <<length<<"L "<<width<<"W "<<height<<"H "<<endl;
            }

            Bags (const Bags &bi) //copy constructor
            {
                  brand = bi.brand;
                  length = bi.length;
                  width = bi.width;
                  height = bi.height;
                  cout<<"\nDo you have the same bag??"<<endl;
            }
            Bags() //default constructor
            {
                  brand = "Adidas";
                  length = 35;
                  width = 20;
                  height = 45;
            }
            //declare as friend function using prototype
            friend void check(Bags,Bags,Bags);
```

Declare friend function within the class, reference to the class objects

```
};
//function to check value similarity and display
void check(Bags a, Bags b, Bags c)
{     //check for heigh similarity among 3 objects
      if(a.height==b.height && a.height==c.height)
            cout<<"\nCommon height for all 3 bags"<<endl;
      else
            cout<<"\n--not all bags have the same heights--"<<endl;
      cout<<"--------------------------------------------------";
}

int main()
{ //need to developed by adding object K, L and M;
      Bags K[3]; //declare array of 3 objects
      for(int i=0;i<3;i++) //loop to set data for each object
            K[i].setdata();
      check(K[0],K[1],K[2]); //call friend function by passing objects
```

This invoked the copy constructor as passing objects

```
}

//——————————————————————————————————————————————————————————————————————————
/*
Lab 07
Question 2
Friend function access private data members
*/

#include<iostream>
#include<iomanip>
using namespace std;
//define class
class ICE_CREAM
{
      private: //private data members
            string flavour;
            int number;
            float price;
      public: //public member functions
            void menu(); //member function
            void setflavour(); //mutator function
            void setHowMany(); //mutator function
            friend void display_receipt(ICE_CREAM); //friend function
            ICE_CREAM(); //default constructor
};

void ICE_CREAM::menu() //display menu information
{
      cout<<"\n=========================================="<<endl;
      cout<<"=== CHOOSE FLAVOUR ==="<<endl;
      cout<<"=========================================="<<endl;
      cout<<"[1] === Strawberry Flavour RM 3.50"<<endl;
      cout<<"[2] === Chocolate Flavour RM 2.50"<<endl;
      cout<<"[3] === Vanilla Flavour RM 1.50"<<endl;
      cout<<"[4] === Durian Flavour RM 0.50"<<endl;
}

void ICE_CREAM::setflavour()
{     //get user input
      int choice;
      cout<<"\nChoice of flavour: ";
      cin>>choice;
      //switch statement to set flavour and price
      switch(choice)
      {
            case 1: flavour = "Strawberry"; price = 3.50; break;
            case 2: flavour = "Chocolate"; price = 2.50; break;
            case 3: flavour = "Vanilla"; price = 1.50; break;
            case 4: flavour = "Durian"; price = 0.50; break;
      }
}

void ICE_CREAM::setHowMany()
{     //get user input to set amount
      cout<<"How many: ";
      cin>>number;
}

ICE_CREAM::ICE_CREAM()
{     //default constructor to display title
```

A friend function, reference to the class object can access the class object private data

```
	cout<<"BARNEY'S HOUSE OF ICE"<<endl;
}

void display_receipt(ICE_CREAM IC) //friend function accepting object
{	//display payment details by accessing class object
	cout<<"\n=========================================="<<endl;
	cout<<"=== PAYMENT ==="<<endl;
	cout<<"=========================================="<<endl;
	cout<<"Flavour\t\t: "<<IC.flavour<<endl;
	cout<<"Total Price\t: RM "<<fixed<<setprecision(2)
	<<IC.price*IC.number<<endl;
}

int main()
{
	ICE_CREAM IC; //declare class object
	IC.menu(); //call member function to show menu
	IC.setflavour(); //call member function to select flavour
	IC.setHowMany();//call member function to set amount
	display_receipt(IC); //call friend function to print receipt
}

//————————————————————————————————————————————————————————————————————————
/*
Lab 07
Question 3
Friend function call by reference using pointer
*/

#include<iostream>
using namespace std;
//define class
class NumberGame
{	int array[5]; //private data member
	public: //public member functions
		//--------------- (a)---------------
		NumberGame() //default constructor to initialize array
		{
			array[0]=15;
			array[1]=20;
			array[2]=33;
			array[3]=38;
			array[4]=100;
			//int array[5]={15,20,33,38,100};
		}
		//declare friend function with prototype
		friend void search(NumberGame , int*);
};
//--------------- (b)---------------
//function to find number in array
void search(NumberGame NG, int* num)
{	//two input parameters with call by reference pointer
	int i = 0, end = 0;
	//loop to go through array elements for search
	do
	{
		if(NG.array[i]==*num) //condition to find number
		{
			end = 1;
		}
		i++; //increment to index array
```

```
            if(i==5) //number not found at the end of search
            {
                  end=1;
            }
      } while(end!=1); //stop search if found/not found flag triggered

      // statements to display output message based on flag
      if(i==5)
            cout<<*num<<" is NOT found!"<<endl;
      else
            cout<<*num<<" is found!"<<endl;
}


int main()
{     NumberGame G ; //declare object
      int num;
      //prompt user number to be searched
      cout<<"Enter a number :";
      cin>>num;
      search(G, &num); //call function using address
      return 0;
}

//————————————————————————————————————————————————————————————————————————
/*
Lab 08
Question 1
Friend classes
*/

#include <iostream>
#include <cmath>
using namespace std;
//define class
class geometry
{
      private: //private data members
            float pi, height, radius;
      public: //public member functions
            //--------(1)--------
            //parameterized constructor with two float parameters
            geometry(float hg,float rd)
            {
                  pi = M_PI; height = hg; radius = rd;
            }
            //--------(2)--------
            //desctructor with display message
            ~geometry()
            {
                  cout<<"\n= END OF PROGRAM=";
            }
            //--------(3)--------
            //declare friend class
            friend class cylinder;
            //--------(4)--------
            //declare friend class
            friend class cone;
};
//define class
class cylinder
{
      private: //private data members
            float vol;
      public: //public member functions
            //--------(5)--------
            //calculate volume function
            void calc_vol(geometry *g) //pointer of class as parameter
            {     //calculate volume using dot and indirection operator
                  vol = g->pi * g->radius * g->radius * g->height;
                  //display output
                  cout<<"\nVolume of cylinder with radius ";
                  cout<<(*g).radius<<" and height ";
                  cout<<(*g).height<<" is : "<<vol<<endl;
            }
};
//define class
class cone
{
      private: //private data members
            float vol;
      public: //public member functions
            //--------(6)--------
            //calculate volumne function
            void calc_vol(geometry &g) //reference argument of class as
parameter
            {     //calculate volume using reference argument
                  vol = g.pi * g.radius * g.radius * g.height/3;
                  //display output
                  cout<<"\nVolume of cone with radius ";
                  cout<<g.radius<<" and height ";
                  cout<<g.height<<" is : "<<vol<<endl;
            }
};

int main()
{     //declare variables
      float hg, rd;
      //get user input
      cout << "Enter height: ";
      cin >> hg;
      cout << "Enter radius: ";
      cin >> rd;
      //--------(7)--------
      //declare object and pass parameter values
      geometry gmt(hg,rd);
      //--------(8)--------
      //declare object
      cylinder cyc;
      //--------(9)--------
      //call method by passing object address
      cyc.calc_vol(&gmt);
      //--------(10)--------
      //declare object
      cone cn;
      //--------(11)--------
      //call method by passing object
      cn.calc_vol(gmt);
}

//------------------------------------------------------------------------
/*
Lab 08
```

```
Question 2
Adding friend class
*/

#include <iostream>
#include <cmath>
using namespace std;
//define class
class geometry
{
        private: //private data members
                float pi, height, radius, length; //add length parameter
        public: //public member functions
                //parameterized constructor with three float parameters
                geometry(float hg,float rd, float lg)
                {
                        pi = M_PI; height = hg; radius = rd;
                        length = lg;
                }
                //desctructor with display message
                ~geometry()
                {
                        cout<<"\n= END OF PROGRAM=";
                }
                //declare friend classes
                friend class cylinder;
                friend class cone;
                //declare new friend class
                friend class cube;
};
//define class
class cylinder
{
        private: //private data members
                float vol;
        public: //public member functions
                //--------(5)--------
                //calculate volume function
                void calc_vol(geometry *g) //pointer of class as parameter
                {       //calculate volume using dot and indirection operator
                        vol = g->pi * g->radius * g->radius * g->height;
                        //display output
                        cout<<"\nVolume of cylinder with radius ";
                        cout<<(*g).radius<<" and height ";
                        cout<<(*g).height<<" is : "<<vol<<endl;
                }
};
//define class
class cone
{
        private: //private data members
                float vol;
        public: //public member functions
                //--------(6)--------
                //calculate volumne function
                void calc_vol(geometry &g) //reference argument of class as parameter
                {       //calculate volume using reference argument
                        vol = g.pi * g.radius * g.radius * g.height/3;
                        //display output
                        cout<<"\nVolume of cone with radius ";
```

```
                  cout<<g.radius<<" and height ";
                  cout<<g.height<<" is : "<<vol<<endl;
            }
};
//define class
class cube
{
      private: //private data members
            float vol;
      public: //public member functions
            //calculate volume function
            void calc_vol(geometry &g) //reference argument of class as
parameter
            {     //calculate volume using reference argument
                  vol = g.length * g.length * g.length;
                  //display output
                  cout<<"\nVolume of cube with length ";
                  cout<<g.length<<" is : "<<vol<<endl;
            }
};

int main()
{     //declare variables
      float hg, rd, lg;
      //get user input
      cout << "Enter height: ";
      cin >> hg;
      cout << "Enter radius: ";
      cin >> rd;
      //add prompt for user to enter length
      cout << "Enter length: ";
      cin >> lg;
      //declare object and pass parameter values
      geometry gmt(hg,rd,lg);
      //declare object
      cylinder cyc;
      //call method by passing object address
      cyc.calc_vol(&gmt);
      //declare object
      cone cn;
      //call method by passing object
      cn.calc_vol(gmt);
      //add declaration and method call for new class object
      cube cb; //declare object
      cb.calc_vol(gmt); //call method by passing object
}

//————————————————————————————————————————————————————————————————————
/*
Lab 08
Question 3
Calling friend class methods
*/
#include<iostream>
#include <limits>
using namespace std;
//define class
class Ticket
{
      private: //private data members
            int no; float price;
```

Remove comment '//' for friend class Student

```
            friend class Student; //declare friend class public:
        //public member functions
            Ticket() //default constructor
            { price = 10.00; }
            void setTickets()
            {       //prompt user input for number of tickets
                    cout<<"Please enter number of tickets to purchase: ";
                    cin>>no;
            }
};
//define class
class Student
{
        private: //private data members
            string id, name, purchase;
            Ticket p;
        public: //public member functions
            void setStudent()
            {
                    fflush(stdin); //clear buffer for input with [space]
                    //promp user input for name and id
                    cout<<"\nEnter ID: ";
                    getline(cin,id);
                    cout<<"Enter Name: ";
                    getline(cin,name);
            }
            void ticket_entry()
            {       //prompt user input for purchase decision
                    char sel;
                    cout<<"Do you want to purchase charity tickets? ";
                    cout<<"[Enter Y or N]: ";
                    cin>>sel;
                    //if-else statement to call function or displya message
                    if(sel=='Y')
                    {       //set purchase and call function of friend class
                            purchase = "Yes";
                            p.setTickets();
                    }
                    else
                            cout<<"--------No ticket purchased--------"<<endl;

                    cin.clear();cin.ignore(numeric_limits<streamsize>::max(),
'\n');
            }
            void display()
            {       //display student and additional details
                    cout<<"\n---------------------------------------------"<<endl;
                    cout<<"      STUDENT DETAILS"<<endl;
                    cout<<"---------------------------------------------"<<endl;
                    cout<<"ID   :"<<id<<endl;
                    cout<<"Name :"<<name<<endl;
                    cout<<"\n---------------------------------------------"<<endl;
                    cout<<"      ADDITIONAL DETAILS"<<endl;
                    cout<<"---------------------------------------------"<<endl;
                    //if-else statement to check purchase details
                    if(purchase=="Yes")
                    {
                            cout<<"You've purchased "<<p.no<<" Tickets"<<endl;
                            cout<<"Total amoun: RM "<<p.no*p.price<<endl;
                    }
                    else
```

```
                        {
                                cout<<"You've not purchased any tickets"<<endl;
                        }
                }
};

int main()
{

        Student s[3]; //declare array of 3 objects
        //loop to call functins of every array object
        for(int i = 0;i<3;i++)
        {
                s[i].setStudent();
                s[i].ticket_entry();
                s[i].display();
        }
}
```