

*****ASSIGNMENT 9*****

NAME: RAMAVATH SANTHOSH ROLL NO: 22MCF1R40

BRANCH: MCA(CSE) COURSE: PSP

1. A point in the x-y plane is represented by its x-coordinate and y-coordinate. Design a class, pointType, that can store and process a point in the x-y plane. You should then perform operations on the point, such as setting the coordinates of the point, printing the coordinates of the point, returning the x-coordinate, and returning the y-coordinate. Also, write a program to test various operations on the point.

```
#include <iostream>
#include <cmath>

using namespace std;

class pointType
{
protected:
    int x, y;
public:

    pointType(int x, int y)
    {
        this->x = x;
        this->y = y;
    }

    int distanceFromXAxis(){ return y; }
    int distanceFromYAxis(){ return x; }
    int getX(){ return x; }
    int getY(){ return y; }

    void set(int _x, int _y)
    {
        x = _x;
        y = _y;
    }

    void display()
    {
        cout << "(" << x << ", " << y << ")" << endl;
    }

    double distanceFromOrigin()
    {
        return sqrt(x*x + y*y);
    }
}
```

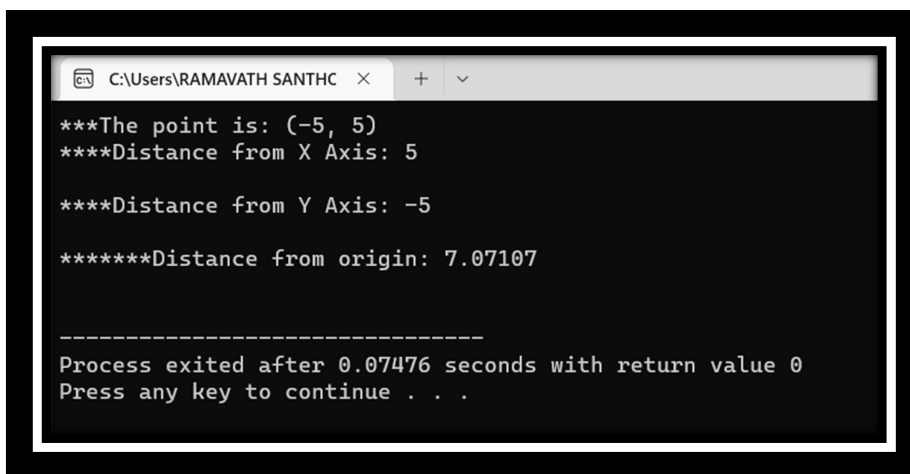
```

};

int main()
{
    pointType p(-5, 5);
    cout << "***The point is: ";
    p.display();
    cout << "****Distance from X Axis: " << p.distanceFromXAxis() << endl<<endl;
    cout << "****Distance from Y Axis: " << p.distanceFromYAxis() << endl<<endl;
    cout << "*****Distance from origin: " << p.distanceFromOrigin() << endl<<endl;

    return 0;
}

```



```

C:\Users\RAMAVATH SANTHC >
***The point is: (-5, 5)
****Distance from X Axis: 5
****Distance from Y Axis: -5
*****Distance from origin: 7.07107

-----
Process exited after 0.07476 seconds with return value 0
Press any key to continue . . .

```

2. Every circle has a center and a radius. Given the radius, we can determine the circle's area and circumference. Given the center, we can determine its position in the x-y plane. The center of the circle is a point in the x-y plane. Design a class, `circleType`, that can store the radius and center of the circle. Because the center is a point in the x-y plane and you designed the class to capture the properties of a point in Question 1 you must derive the class `circleType` from the class `pointType`. You should be able to perform the usual operations on the circle, such as setting the radius, printing the radius, calculating and printing the area and circumference, and carrying out the usual operations on the center. Also, write a program to test various operations on a circle.

```

#include <iostream>
#include <math.h>

using namespace std;

class pointType
{
protected:
    int x, y;
public:
    pointType()

```

```

{
    x = 0;
    y = 0;
}

pointType(int x, int y)
{
    this->x = x;
    this->y = y;
}

int distanceFromXAxis(){ return y; }
int distanceFromYAxis(){ return x; }
int getX(){ return x; }
int getY(){ return y; }

void set(int _x, int _y)
{
    x = _x;
    y = _y;
}

void display()
{
    cout << "(" << x << ", " << y << ")" << endl;
}

double distanceFromOrigin()
{
    return sqrt(x*x + y*y);
}

};

class circleType : public pointType
{
protected:
    int radius;
public:
    circleType() : pointType()
    {
        radius = 0;
    }

    circleType(int _r, int _x, int _y) : pointType(_x, _y)
    {
        radius = _r;
    }

    double getArea()

```

```

    {
        return M_PI * radius * radius;
    }

double getCircumference()
{
    return 2 * M_PI * radius;
}

void display()
{
    cout << "Circle radius is : " << radius << endl;
    cout << "Circle centre is : ";
    pointType::display();
}

void setCenter(int _x, int _y)
{
    set(_x, _y);
}

void setRadius(int _r)
{
    radius = _r;
}

int getRadius()
{
    return radius;
}

int getCentreX(){ return x; }
int getCentreY(){ return y; }
};

int main()
{
    circleType c(10, 5, 5);
    c.display();
    cout << "Area " << c.getArea() << endl;
    cout << "Circumference: " << c.getCircumference() << endl;
    return 0;
}

```

```
C:\Users\RAMAVATH SANTHC X + v
Circle radius is : 10
Circle centre is : (5, 5)
Area 314.159
Circumference: 62.8319

-----
Process exited after 0.07013 seconds with return value 0
Press any key to continue . . .
```

3. Every cylinder has a base and height, wherein the base is a circle. Design a class, cylinderType, that can capture the properties of a cylinder and perform the usual operations on the cylinder. Derive this class from the class circleType designed in Question 3. Some of the operations that can be performed on a cylinder are as follows: calculate and print the volume, calculate and print the surface area, set the height, set the radius of the base, and set the center of the base. Also, write a program to test various operations on a cylinder.

```
#include <iostream>
```

```
#include <math.h>
```

```
using namespace std;
```

```
class pointType
```

```
{
```

```
protected:
```

```
    int x, y;
```

```
public:
```

```
    pointType()
```

```
{
```

```
        x = 0;
```

```
        y = 0;
```

```
}
```

```
    pointType(int _x, int _y)
```

```
{
```

```
        x = _x;
```

```
        y = _y;
```

```
}
```

```
    double distanceFromOrigin()
```

```
{
```

```
        return sqrt(x*x + y*y);
```

```
}
```

```
    int distanceFromXAxis(){ return y; }
```

```
    int distanceFromYAxis(){ return x; }
```

```
    int getX(){ return x; }
```

```

int getY(){ return y; }

void set(int _x, int _y)
{
    x = _x;
    y = _y;
}

void display()
{
    cout << "(" << x << ", " << y << ")" << endl;
}

};

class circleType : public pointType
{
protected:
    int radius;
public:
    circleType() : pointType()
    {
        radius = 0;
    }

    circleType(int _r, int _x, int _y) : pointType(_x, _y)
    {
        radius = _r;
    }

    double getArea()
    {
        return M_PI * radius * radius;
    }

    double getCircumference()
    {
        return 2 * M_PI * radius;
    }

    void display()
    {
        cout << "Circle radius: " << radius << endl;
        cout << "Circle centre: ";
        pointType::display();
    }

    void setCenter(int _x, int _y)
    {
        set(_x, _y);
    }
}

```

```

    void setRadius(int _r)
    {
        radius = _r;
    }

    int getRadius()
    {
        return radius;
    }

    int getCentreX(){ return x; }
    int getCentreY(){ return y; }
};

class cylinderType : public circleType
{
protected:
    int height;
public:
    cylinderType() : circleType()
    {
        height = 0;
    }

    cylinderType(int _x, int _y, int radius, int _h) : circleType(radius, _x, _y)
    {
        height = _h;
    }

    double getVolume()
    {
        return height * getArea();
    }

    double getSurfaceArea()
    {
        return height * getCircumference() + 2 * getArea();
    }

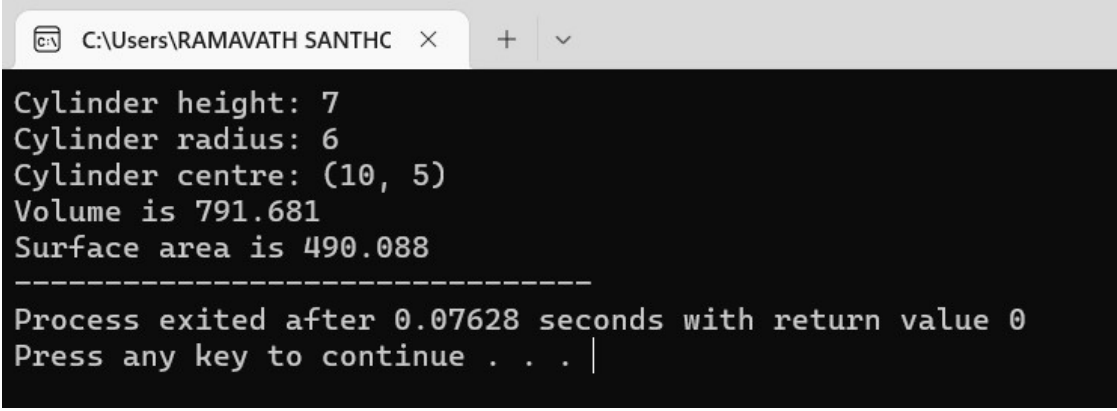
    void setHeight(int _h)
    {
        height = _h;
    }

    void display()
    {
        cout << "Cylinder height: " << height << endl;
        cout << "Cylinder radius: " << radius << endl;
        cout << "Cylinder centre: ";
        pointType::display();
    }
}

```

```
};

int main()
{
    cylinderType cyl(10, 5, 6, 7);
    cyl.display();
    cout << "Volume is " << cyl.getVolume() << endl;
    cout << "Surface area is " << cyl.getSurfaceArea();
    return 0;
}
```



```
Cylinder height: 7
Cylinder radius: 6
Cylinder centre: (10, 5)
Volume is 791.681
Surface area is 490.088
-----
Process exited after 0.07628 seconds with return value 0
Press any key to continue . . . |
```

4.A retail store has a preferred customer plan where customers may earn discounts on all their purchases. The amount of a customer's discount is determined by the amount of the customer's cumulative purchases in the store.

- When a preferred customer spends \$500, he or she gets a 5% discount on all future purchases.
- When a preferred customer spends \$1,000, he or she gets a 6% discount on all future purchases.
- When a preferred customer spends \$1,500, he or she gets a 7% discount on all future purchases.
- When a preferred customer spends \$2,000 or more, he or she gets a 10% discount on all future purchases.

Design a class named PreferredCustomer, which is derived from the CustomerData class you created in problem 7. The PreferredCustomer class should have the following member variables:

- purchasesAmount (a double)
- discountLevel (a double)

The purchasesAmount variable holds the total of a customer's purchases to date. The discountLevel variable should be set to the correct discount percentage, according to the store's preferred customer plan. Write appropriate member functions for this class and demonstrate it in a simple program.

```
#include<bits/stdc++.h>
using namespace std;
class CustomerData{
```



```

protected :
    int CustomerNumber;
    bool mailList;
public:
    void Getdata()
    {
        cout<<"Enter Customer Number:";
        cin>>CustomerNumber;
        cout<<"\nDo you want to be in the mailing List? 1->Y OR 0->N -->";
        cin>>mailList;
    }
    void displaydata()
    {
        cout<<"\n\nCustomerNumber is->"<<CustomerNumber;
        cout<<"\nMailing List Status->"<<((mailList?"Yes":"No"));
    }
};

class PreferredCustomer: public CustomerData{
private :
    double purchaseAmount;
    double discountLevel;

public:
    void purchasedata()
    {
        cout<<"\nEnter Total Purchase done by the customer:";
        cin>>purchaseAmount;

        if(purchaseAmount>=2000)
            discountLevel= 10;
        else if(purchaseAmount>=1500 && purchaseAmount<2000 )
            discountLevel= 7;
        else if(purchaseAmount>=1000 && purchaseAmount<1500 )
            discountLevel= 6;
        else
            discountLevel= 5;
    }
    void showdata()
    {
        cout<<"\n\nTotal Purchase is->"<<purchaseAmount;
        cout<<"\nDiscount to be applied on future purchases-
>"<<discountLevel<<"%";
    }
};

int main()
{
    PreferredCustomer pd;
    pd.Getdata();
}

```

```

        pd.purchasedata();
        pd.displaydata();
        pd.showdata();
    }

```

```

C:\Users\RAMAVATH SANTHC
Enter Customer Number:22
Do you want to be in the mailing List? 1->Y OR 0->N -->1
Enter Total Purchase done by the customer:524665

CustomerNumber is->22
Mailing List Status->Yes

Total Purchase is->524665
Discount to be applied on future purchases->10%
-----
Process exited after 11.01 seconds with return value 0
Press any key to continue . . .

```

5. Design a class named Person with the following member variables:

- FirstName
- LastName
- Address
- City
- State
- Zip
- PhoneNo
- Struct DOB (day, month, year)

Having a private constructor.

Write the appropriate accessor (getter) and mutator (setter) functions for these member variables.

In the main() function, create an array of objects of user defined size by calling a static function

for making objects.

Input data of all the persons and then display it by using a constant function

```

#include<iostream>
using namespace std;

struct DOB {
    int day, month, year;
};

class Person {
private:
    string firstName, lastName, address;

```

```

    string city, state, zip, phone;
    DOB d;
    Person() {
        d.day = 0;
        d.month = 0;
        d.year = 0;
    }
public:
    static Person *PersonArray(int n) {
        return new Person[n];
    }
    void getPersonDetails() {
        cout << "Enter First Name: ";
        cin >> firstName;
        cout << "Enter Last Name: ";
        cin >> lastName;
        cout << "Enter Address: ";
        cin.ignore();
        getline(cin, address);
        cout << "Enter City: ";
        cin >> city;
        cout << "Enter State: ";
        cin >> state;
        cout << "Zip Code: ";
        cin >> zip;
        cout << "Enter Phone: ";
        cin >> phone;
        cout << "Enter DOB(dd mm yyyy): ";
        cin >> d.day >> d.month >> d.year;
    }
    const void displayPersonDetails() {
        cout << "Full Name      : " << firstName << " " << lastName << endl;
        cout << "DOB(dd/mm/yyyy) : " << d.day << "/" << d.month << "/" << d.year << endl;
        cout << "Full Address   : " << address << ", " << city;
        cout << ", " << state << " (" << zip << ")" << endl;
        cout << "Phone No      : " << phone << endl;
    }
};

int main() {
    int n = 2;
    Person *p = Person::PersonArray(n);
    int i = 0;
    while(i < n) {
        cout << "----- Person - " << i + 1 << " -----" << endl;
        p[i].getPersonDetails();
        i++;
    }
    i = 0;
    system("cls");
    cout << "----- Persons List -----" << endl;

```

```

while(i < n) {
    p[i].displayPersonDetails();
    cout << endl;
    i++;
}
return 0;
}

```

```

C:\Users\RAMAVATH SANTHC >
----- Persons List -----
Full Name      : ramavath santhosh
DOB(dd/mm/yyyy) : 2/12/2022
Full Address    : 2-2-1055, Hyd, TS (50013)
Phone No       : 9568475241

Full Name      : bhukya chinna
DOB(dd/mm/yyyy) : 2/10/2325
Full Address    : 2-5-7, warangal, TS (546810)
Phone No       : 9874561230

-----
Process exited after 86.58 seconds with return value 0
Press any key to continue . . .

```

6. Design a class Car having attributes.

PersonData Driver (an object of class done in Q#5)

String carName

Int carID

String type (SUV, Wagon, Electrical etc)

Static int count

Const int wheels (assuming every car will have four wheels)

Having default, overloaded, copy constructors with initializer lists and a destructor,

Having functions addCar(), deleteCar(), updateCar(), printCar().

In the main(), you'll only have a pointer of size 1 at start.

Here, you'll have a constant object initialized explicitly like this const Car

constantObj (Person, cName, cID, cType);

Now, print this object at the start of the program and then print the menu. You will have a menu driven program to perform following functionalities.

1. Add Car

2. DeleteCar

3. Update Car Attributes

4. Print Details

5. Print List of cars

One thing, while printing the Driver details display only his name

7. Design a class named PersonData with the following member variables:

- FirstName
- LastName
- Address
- City
- State
- Zip
- PhoneNo

Write the appropriate accessor (getter) and mutator (setter) 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. CustomerData class will have the

- InputCustomerData member function which will Input all the data for customer. (use function over riding).
- DisplayCustomerData member function which will display all the data for customer. (use function over riding).

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

```
#include<bits/stdc++.h>
using namespace std;
class PersonData{
    protected:
        string FirstName,LastName,Address;
        string state,City;
        int ZIP,PhoneNo;

    public:
        void Getdata()
        {
            cout<<"\nEnter Customer FirstName:";
            cin>>FirstName;
            cout<<"\nEnter Customer LastName:";
            cin>>LastName;
            cout<<"\nEnter Customer Address:";
            cin>>Address;
            cout<<"\nEnter Customer State:";
            cin>>state;
```

```

        cout<<"\nEnter Customer City:";
        cin>>City;
        cout<<"\nEnter Customer Zip code:";
        cin>>ZIP;
        cout<<"\nEnter Customer PhoneNo:";
        cin>>PhoneNo;

    }
    void displaydata()
    {
        cout<<"\n\nFirst Name->"<<FirstName;
        cout<<"\nLast Name->"<<LastName;
        cout<<"\nAddress->"<<Address;
        cout<<"\nState ->"<<state;
        cout<<"\nCity->"<<City;
        cout<<"\nZIP->"<<ZIP;
        cout<<"\nPhoneNo->"<<PhoneNo;

    }
};

class CustomerData: public PersonData{
    private :
        int CustomerNumber;
        bool mailList;
    public:
        void Getdata()
        {
            cout<<"Enter Customer Number:";
            cin>>CustomerNumber;
            cout<<"\nDo you want to be in the mailing List? 1->Y OR 0->N -->";
            cin>>mailList;
        }
        void displaydata()
        {
            cout<<"\n\nCustomerNumber is->"<<CustomerNumber;
            cout<<"\nMailing List Status->"<<((mailList)?"Yes":"No");

        }

};

int main()
{
    CustomerData cd;
    cd.Getdata();
    cd.displaydata();
}

```

```
C:\Users\RAMAVATH SANTHC  X  +  v
Enter Customer Number:4685

Do you want to be in the mailing List? 1->Y OR 0->N -->1

CustomerNumber is->4685
Mailing List Status->Yes
-----
Process exited after 33.03 seconds with return value 0
Press any key to continue . . .
```

8. write a program to design a class representing the information regarding digital library (books, tape: book & tape should be separate classes having the base class as media). The class should have the functionality for adding new item, issuing, deposit etc. the program should use runtime polymorphism.

```
#include<iostream>
using namespace std;
struct date{
    int day;
    int month;
    int year;
};
ostream& operator<<(ostream& o,date &d)
{
    o<<d.day<<"/"<<d.month<<"/"<<d.year;
    return o;
}
istream& operator>>(istream& i, date &d)
{
    i>>d.day>>d.month>>d.year;
    return i;
}
class Media{
protected:
    int mediaId;
    string mediaName;
    date DOI;
    date DOD;
    bool isIssued;
public:
    void addMedia()
    {
        cout<<"Enter Media Id : ";
        cin>>mediaId;
        fflush(stdin);
        cout<<"Enter Media Name : ";
        getline(cin,mediaName);
        isIssued = false;
        DOI.day = DOD.day=DOI.month=DOD.month=DOD.year=DOI.year=0;
    }
}
```

```

int GetId()
{
    return mediald;
}
void issueMedia()
{
    cout<<"Enter date of issue : \n";
    cin>>DOI;
    isIssued = true;
}
void depositMedia()
{
    cout<<"Enter date of deposit : \n";
    cin>>DOD;
    isIssued = false;
}
void display()
{
    cout<<"Media Id : "<<this->mediald<<endl;
    cout<<"Media Name : "<<this->mediaName<<endl;
    cout<<"Media Issued? : ";
    if(isIssued)
    {
        cout<<"yes \n";
        cout<<"Date of Issue : "<<DOI<<endl;
    }
    else{
        cout<<"no \n";
        cout<<"Date of Deposit : "<<DOD<<endl;
    }
}
};
class Books:public Media{
private:
    int number_of_pages;
    string author;
public:
    void addMedia()
    {
        Media::addMedia();
        cout<<"Enter Number of pages in the book : ";
        cin>>number_of_pages;
        fflush(stdin);
        cout<<"Enter author's name : ";
        getline(cin,author);
    }
    void display()
    {
        Media::display();
        cout<<"Number of pages in the book : "<<number_of_pages<<endl;
        cout<<"Author's name : "<<author<<endl;
    }
};

```



```

        }
    };
    class Tape:public Media{
    private:
        int length_of_tape;
        string singer;
    public:
        void addMedia()
        {
            Media::addMedia();
            cout<<"Enter length of tape : ";
            cin>>length_of_tape;
            fflush(stdin);
            cout<<"Enter singer's name : ";
            getline(cin,singer);
        }
        void display()
        {
            Media::display();
            cout<<"Length of tape : "<<length_of_tape<<endl;
            cout<<"Singer's name : "<<singer<<endl;
        }
    };
    int main()
    {
        Books mybooks[100];
        Tape mytape[100];
        int b=0,t=0;
        while(1)
        {
            int ch,bid,tid;

            cout<<"Select Options from the menu : ";
            cout<<"\n1.Add a Book\n2.Add a Tape\n3.Display Book details\n4.Display Tape
details\n5.Issue Book\n6.Issue Tape\n7.Return Book\n8.Return Tape\n9.Quit\n:";
            cin>>ch;
            int count=0;
            switch(ch)
            {
                case 1:
                    mybooks[b].addMedia();
                    b++;
                    cout<<"\n Book added Successfully\n";
                    cin.get();
                    break;
                case 2:
                    mytape[t].addMedia();
                    t++;
                    cout<<"\n Tape added Successfully\n";
                    cin.get();
                    break;
            }
        }
    }
}

```

case 3:

```
cout<<"\nEnter the book Id : ";
cin>>bid;
for(int i=0;i<b;i++)
{
    if(mybooks[i].GetId() == bid)
    {
        mybooks[i].display();
        count++;
    }
}
if(count==0)
{
    cout<<"\nBook not found with id "<<bid<<endl;
}
fflush(stdin);
cin.get();
break;
```

case 4:

```
cout<<"\nEnter the Tape Id : ";
cin>>tid;
for(int i=0;i<t;i++)
{
    if(mytape[i].GetId() == tid)
    {
        mytape[i].display();
        count++;
    }
}
if(count==0)
{
    cout<<"\nTape not found with id "<<tid<<endl;
}
fflush(stdin);
cin.get();
break;
```

case 5:

```
cout<<"\nEnter book_id of the book you want to issue : ";
cin>>bid;
for(int i=0;i<b;i++)
{
    if(mybooks[i].GetId() == bid)
    {
        count++;
        mybooks[i].issueMedia();
        cout<<"\nBook Issued\n";
        break;
    }
}
if(count==0)
{
```

```

        cout<<"\nBook not found with id "<<bid<<endl;
    }
    fflush(stdin);
    cin.get();
    break;
case 6:
    cout<<"\nEnter tape_id of the tape you want to issue : ";
    cin>>tid;
    for(int i=0;i<t;i++)
    {
        if(mytape[i].GetId() == tid)
        {
            count++;
            mytape[i].issueMedia();
            cout<<"\nTape Issued\n";
            break;
        }
    }
    if(count==0)
    {
        cout<<"\nTape not found with id "<<tid<<endl;
    }
    fflush(stdin);
    cin.get();
    break;
case 7:
    cout<<"\nEnter book_id of the book you want to deposit : ";
    cin>>bid;
    for(int i=0;i<b;i++)
    {
        if(mybooks[i].GetId() == bid)
        {
            count++;
            mybooks[i].depositMedia();
            cout<<"\nBook Deposited\n";
            break;
        }
    }
    if(count==0)
    {
        cout<<"\nBook not found with id "<<bid<<endl;
    }
    fflush(stdin);
    cin.get();
    break;
case 8:
    cout<<"\nEnter tape_id of the tape you want to deposit : ";
    cin>>tid;
    for(int i=0;i<t;i++)
    {
        if(mytape[i].GetId() == tid)

```

```

        {
            count++;
            mytape[i].depositMedia();
            cout<<"\nTape Deposited\n";
            break;
        }
    }
    if(count==0)
    {
        cout<<"\nTape not found with id "<<tid<<endl;
    }
    fflush(stdin);
    cin.get();
    break;
case 9:
    exit(0);
default:
    cout<<"Please enter correct option";
    break;
    }
}
return 0;
}

```

```

C:\Users\RAMAVATH SANTHC
5.Issue Book
6.Issue Tape
7.Return Book
8.Return Tape
9.Quit
:1
Enter Media Id : 65
Enter Media Name : nhggx
Enter Number of pages in the book : 456
Enter author's name : jhgyt

Book added Successfully
9
Select Options from the menu :
1.Add a Book
2.Add a Tape
3.Display Book details
4.Display Tape details
5.Issue Book
6.Issue Tape
7.Return Book
8.Return Tape
9.Quit
:9

-----
Process exited after 38.44 seconds with return value 0
Press any key to continue . . . |

```