NAME : SHARMA AVINASH DHARMENDRA

SUB: OOPS GENERAL

ROLLNO. : 49

DIV : FYBSCIT

|  |  |
| --- | --- |
| SrNo. | Index |
| 1 | **Class and Methods** |
| a. | Design an employee class for reading and displaying the employee information, the getInfo() and displayInfo() methods will be used repectively. Where getInfo() will be private method |
| b. | Design the class student containing getData() and displayData() as two of its methods which will be used for reading and displaying the student information respectively.Where getData() will be private method. |
| c. | Write a program to demonstrate function definition outside class and accessing class members in function definition. |
| 2 | **Using friend functions.** |
| a. | Write a friend function for adding the two complex numbers, using a single class |
| b. | Write a friend function for adding the two different distances and display its sum, using two classes. |
| c. | Write a friend function for adding the two matrix from two different classes and display its sum. |
| 3 | **Constructors and method overloading.** |
| a. | Design a class Complex for adding the two complex numbers and also show the use of constructor. |
| b. | Design a class Geometry containing the methods area() and volume() and also overload the area() Function . |
| c. | Design a class StaticDemo to show the implementation of static variable and static function. |
| 4 | **Operator Overloading** |
| a. | Overload the operator unary(-) for demonstrating operator overloading. |
| b. | Overload the operator + for adding the timings of two clocks, And also pass objects as an argument. |
| c. | Overload the + for concatenating the two strings. For e.g “Py” + “thon” = Python |
| 5 | **Inheritance** |
| a. | Design a class for single level inheritance using public and private type derivation. |
| b. | Design a class for multiple inheritance. |
| c. | Implement the hierarchical inheritance. |
| 6 | **Virtual functions and abstract classes** |
| a. | Implement the concept of method overriding. |
| b. | Show the use of virtual function |
| c. | Show the implementation of abstract class. |
| 7 | **String handling** |
| a. | String operations for string length , string concatenation |
| b. | String operations for string reverse, string comparison, |
| c. | Console formatting functions. |
| 8 | **Exception handling** |
| a. | Show the implementation of exception handling |
| b. | Show the implementation for exception handling for strings |
| c. | Show the implementation of exception handling for using the pointers. |
| 9 | **File handling** |
| a. | Design a class FileDemo open a file in read mode and display the total number of words and lines in the file. |
| b. | Design a class to handle multiple files and file operations |
| c. | Design a editor for appending and editing the files |
| 10 | **Templates** |
| a. | Show the implementation for the following |
| b. | Show the implementation of template class library for swap function. |
| c. | Design the template class library for sorting ascending to descending and vice-versa |

**1.Class and Methods**

a. **Design an employee class for reading and displaying the employee information, the getInfo() and displayInfo() methods will be used repectively. Where getInfo() will be private method**

Ans:-

#include<iostream>

using namespace std;

class employeeś

{

char name[20];

int age;

float basic\_sal;

void getInfo()

{

cout<<endl<<"Enter name: ";

cin>>name;

cout<<endl<<"Enter age: ";

cin>>age;

cout<<endl<<"Enter basic salary: ";

cin>>basic\_sal;

}

public:

void displayInfo()

{

getInfo();

cout<<endl<<"\tEMPLOYEE INFORMATION\n";

cout<<"\t ";

cout<<endl<<" Name: "<<name;

cout<<endl<<" Age: "<<age;

cout<<endl<<" Basic Salary: "<<basic\_sal;

cout<<endl<<" Gross Salary: "<<basic\_sal + (0.6\*basic\_sal) +

(0.4\*basic\_sal);

}

};

int main()

{

employee e;

e.displayInfo();

return 0;

}

Output :

Enter name: Sanjeela

Enter age: 33

Enter basic salary: 8000EMPLOYEE INFORMATION

Name: Sanjeela

Age: 33

Basic Salary: 8000

Gross Salary: 16000

b.**Design the class student containing getData() and displayData() as two**

**of its methods which will be used for reading and displaying the student**

**information respectively. Where getData() will be private method**.

Ans:-

#include <iostream>

using namespace std;

class student

{

    char name[20];

    int age;

    float percentage;

    void getData()

    {

        cout << endl

             << "Enter name: ";

        cin >> name;

        cout << endl

             << "Enter age: ";

        cin >> age;

        cout << endl

             << "Enter percentage: ";

        cin >> percentage;

    }

public:

    void displayData()

    {

        getData();

        cout << endl

             << "\tSTUDENT INFORMATION\n";

        cout << "\t ";

        cout << endl

             << " Name: " << name;

        cout << endl

             << " Age: " << age;

        cout << endl

             << " Percentage: " << percentage;

    }

};

int main()

{

    student s;

    s.displayData();

    return 0;

}

Output :

Enter name: Sanjeela

Enter age: 46

Enter percentage: 89

c.**Design the class Demo which will contain the following methods:**

**readNo() ,factorial() for calculating he factorial of a number, reverseNo()**

**will reverse the given number, isPalindrome() will check the given number**

**is palindrome, isArmstrong() which will calculate the given number is**

**armStrong or not.Where readNo() will be private method.**

Ans:

#include<iostream>

using namespace std;

class Demo

{

int n,nn;

void readNo()

{

cout<<"\nEnter a number: ";

cin>>n;

}

public:

int factorial()

{

readNo();

int f=1;

while(n>0)

{

f=f\*n;

n--;

}

return f;

}

int reverseNo()

{

readNo();

int d,num=0;

nn=n;

while(n>0)

{

d=n%10;

num=num\*10+d;

n=n/10;

}

return num;

}

int isPalindrome()

{

int revnum=reverseNo();

if(nn==revnum)

return 1;

else

return 0;

}

int isArmstrong()

{

readNo();

int nn=n,sum=0,d;

while(n>0)

{

d=n%10;

sum=sum+(d\*d\*d);

n=n/10;

}

if(nn==sum)

return 1;

else

return 0;

}

};

int  main()

{

Demo d;

int res;

cout<<endl<<"To find out Factorial of Input Number\n";

cout<<"";

res=d.factorial();

cout<<endl<<"Factorial= "<<res;

cout<<endl<<"\nTo find out the reverse of Input Number\n";

cout<<"";

res=d.reverseNo();

cout<<endl<<"Reverse = "<<res;

cout<<endl<<"\nTo check whether input number is palindrome\n";

cout<<"";

if(d.isPalindrome())

cout<<"\nNumber is palindrome";

else

cout<<"\nNumber is not palindrome";

cout<<endl<<"\nTo check whether input number is Armstrong\n";

cout<<"";

if(d.isArmstrong())

cout<<"\nNumber is armstrong";

else

cout<<"\nNumber is not armstrong";

return 0;

}

Output :

To find out Factorial of Input Number

Enter a number: 5

Factorial= 120

To find out the reverse of Input Number

Enter a number: 1234

Reverse = 4321

To check whether input number is palindrome

Enter a number: 12321

Number is palindrome

To check whether input number is Armstrong

Enter a number: 154

Number is not Armstrong

**2.Using friend functions**

a.**Write a friend function for adding the two complex numbers, using a**

**single class**

**Ans:-**

#include<iostream>

using namespace std;

class complex

{

float n,m;

public:

void getData()

{

cout<<"\nEnter real number: ";

cin>>n;

cout<<"\nEnter imaginary number: ";

cin>>m;

}

void showData()

{ cout<< n <<" + i"<< m ;

}

friend complex sum(complex, complex);

};

complex sum(complex c1, complex c2)

{

complex c3;

c3.n=c1.n+c2.n;

c3.m=c1.m+c2.m;

return c3;

}

int main()

{

complex obj1, obj2, obj3;

cout<<"\nEnter Data for 1st Complex Number \n";

cout<<"";

obj1.getData();

cout<<"\nEnter Data for 2nd Complex Number \n";

cout<<"";

obj2.getData();

obj3=sum(obj1,obj2);

cout<<"\nComplex Number1: ";

obj1.showData();

cout<<"\nComplex Number2: ";

obj2.showData();

cout<<"\nComplex Number3: ";

obj3.showData();

return 0;

}

Output :

Enter Data for 1st Complex Number

Enter real number: 1.2

Enter imaginary number: 2.4

Enter Data for 2nd Complex Number

Enter real number: 4.5

Enter imaginary number: 6.4

Complex Number1: 1.2 + i2.4

Complex Number2: 4.5 + i6.4

Complex Number3: 5.7 + i8.8

b.**Write a friend function for adding the two different distances and display**

**its sum, using two classes.**

Ans:-

#include<iostream>

using namespace std;

class distance2;

class distance1

{

int feet;

int inch;

public:

void getData()

{

cout<<"\nEnter feet: ";

cin>>feet;

cout<<"\nEnter inches: ";

cin>>inch;

}

void showData()

{

cout<< feet <<"'-"<<inch<<"\"";

}

friend void sum(distance1, distance2);

};

class distance2

{

int feet,inch;

public:

void getData()

{

cout<<"\nEnter feet: ";

cin>>feet;

cout<<"\nEnter inches: ";

cin>>inch;

}

void showData()

{

cout<< feet<<"'-"<<inch <<"\"" ;

}

friend void sum(distance1, distance2);

};

void sum(distance1 d1, distance2 d2)

{

int f=d1.feet+d2.feet;

int i=d1.inch+d2.inch;

if(i>=12)

{

i=i-12;  f++;

}

cout<< f<<"'-"<<i <<"\"" ;

}

int main()

{

distance1 obj1;

distance2 obj2;

cout<<"\nEnter Data for 1st Distance \n";

cout<<"";

obj1.getData();

cout<<"\nEnter Data for 2nd Distance \n";

cout<<"";

obj2.getData();

cout<<"\nDistance1: ";

obj1.showData();

cout<<"\nDistance2: ";

obj2.showData();

cout<<"\nDistance3: ";

sum(obj1,obj2);

return 0;

}

Output :

Enter Data for 1st Distance

Enter feet: 12

Enter inches: 7

Enter Data for 2nd Distance

Enter feet: 2

Enter inches: 8

Distance1: 12'-7"

Distance2: 2'-8"

Distance3: 15'-3"

c.**Write a friend function for adding the two matrix from two different**

**classes and display its sum.**

Ans:-

#include<iostream>

using namespace std;

class matrix2;

class matrix1

{

int a[3][3];

public:

void getData()

{

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)

cin>>a[i][j];

}

}

void showData()

{

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)  cout<<a[i][j]<<"";

cout<<endl;

}

}

friend void sum(matrix1, matrix2);

};

class matrix2

{

int a[3][3];

public:

void getData()

{

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)

cin>>a[i][j];

}

}

void showData()

{

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)

cout<<a[i][j]<<"";

cout<<endl;

}

}

friend void sum(matrix1, matrix2);

};

void sum(matrix1 m1, matrix2 m2)

{

int a[3][3];

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)

{

a[i][j]=m1.a[i][j] + m2.a[i][j];

cout<<a[i][j]<<"";

}

cout<<endl;

}

}

int  main()

{

matrix1 obj1;

matrix2 obj2;

cout<<"\nEnter Data for 1st Matrix \n";

cout<<" \n";

obj1.getData();

cout<<"\nEnter Data for 2nd Matrix \n";

cout<<" \n";

obj2.getData();

cout<<"\nMatrix1: \n";

obj1.showData();

cout<<"\nMatrix2: \n";

obj2.showData();

cout<<"\nMatrix3: \n";

sum(obj1,obj2);

return 0;

}

Output :

Enter Data for 1st Matrix

1 2 3

4 5 6

7 8 9

Enter Data for 2nd Matrix

1 2 3

4 5 6

7 8 9

**3.Constructors and method overloading.**

a.**Design a class Complex for adding the two complex numbers and also show the use of**

**constructor.**

Ans :-

#include<iostream>

using namespace std;

class matrix2;

class matrix1

{

int a[3][3];

public:

void getData()

{

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)

cin>>a[i][j];

}

}

void showData()

{

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)  cout<<a[i][j]<<"";

cout<<endl;

}

}

friend void sum(matrix1, matrix2);

};

class matrix2

{

int a[3][3];

public:

void getData()

{

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)

cin>>a[i][j];

}

}

void showData()

{

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)

cout<<a[i][j]<<"";

cout<<endl;

}

}

friend void sum(matrix1, matrix2);

};

void sum(matrix1 m1, matrix2 m2)

{

int a[3][3];

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)

{

a[i][j]=m1.a[i][j] + m2.a[i][j];

cout<<a[i][j]<<"";

}

cout<<endl;

}

}

int  main()

{

matrix1 obj1;

matrix2 obj2;

cout<<"\nEnter Data for 1st Matrix \n";

cout<<" \n";

obj1.getData();

cout<<"\nEnter Data for 2nd Matrix \n";

cout<<" \n";

obj2.getData();

cout<<"\nMatrix1: \n";

obj1.showData();

cout<<"\nMatrix2: \n";

obj2.showData();

cout<<"\nMatrix3: \n";

sum(obj1,obj2);

return 0;

}

Output :

Complex Number1: 3 + i4

Complex Number2: 4 + i5

b.**Design a class Geometry containing the methods area() and volume()**

**and also overload the area() function .**

**Ans:-**

#include<iostream>

using namespace std;

class geometry

{

int l,b;

public:

int area(int x)

{

l=b=x;

return(l\*b);

 }

 int area(int x, int y)

{

l=x;

b=y;

return(l\*b);

}

int volume(int x)

{

l=x;

return(l\*l\*l);

}

};

int  main()

{

geometry g;

cout<<"\nArea of square= "<<g.area(10);

cout<<"\nArea of rectangle= "<<g.area(10,15);

cout<<"\nVolume of cube= "<<g.volume(6);

return 0;

}

Output:

Area of square= 100

Area of rectangle= 150

Volume of cube= 216

**c.Design a class Static Demo to show the implementation of**

**staticvariable and static function.**

**Ans:-**

#include <iostream>

//#include <cstdio>

using namespace std;

class student

{

    int roll\_no;

    char name[30];

    float percent;

    static int c;

public:

    void get()

    {

        cout << "\nEnter Name:";

        gets(name);

        cout<<"\nEnter percentage";

        cin>>percent;

        roll\_no = ++c;

    }

    void show()

    {

        cout <<"\nroll name"<< roll\_no;

        cout <<"\nName :"<< name;

        cout <<"\nPercentage :"<< percent;

        cout <<"\n\n\tTotal number of students admitted :"<< c;

    }

};

int student::c;

int main()

{

    student s1, s2;

    s1.get();

    s2.get();

    cout <<"\n Object 1 Data";

    cout <<"\n \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*";

    s1.show();

    cout <<"\nObject 2 Data";

    cout <<"\n \* \*\*\*\*\*\*\*\*\*\*\*\*\*";

    s2.show();

    return 0;

}

Output :

Enter Name: Sanjeela Sagar

Enter percentage: 78

Enter Name: Rakesh Sagar

Enter percentage: 96

Object 1 Data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Roll No: 1

Name: Sanjeela Sagar

Percentage: 78

Total number of students admitted: 2

Object 2 Data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Roll No: 2Name: Rakesh Sagar

Percentage: 96

Total number of students admitted: 2

**4.Operator Overloading**

**a.Overload the operator unary(-) for demonstrating operator overloading.**

**Ans:-**

#include<iostream>

using namespace std;

class geometry

{

int l,b;

public:

int area(int x)

{

l=b=x;

return(l\*b);

 }

 int area(int x, int y)

{

l=x;

b=y;

return(l\*b);

}

int volume(int x)

{

l=x;

return(l\*l\*l);

}

};

int  main()

{

geometry g;

cout<<"\nArea of square= "<<g.area(10);

cout<<"\nArea of rectangle= "<<g.area(10,15);

cout<<"\nVolume of cube= "<<g.volume(6);

return 0;

}

Output:

Enter three numbers: 5

-6

7

Original contents

A= 5 B= -6 C= 7

After Negation

A= -5 B= 6 C= -7

**b.Overload the operator + for adding the timings of two clocks, And**

**also pass objects as an argument.**

**Ans:-**

#include<iostream>

 using namespace std;

class time

{

    int hrs, min, sec;

public:

    void get()

    {

        cout << "\n\tEnter time (in hrs:minutes:seconds form): ";

        cin >> hrs >> min >> sec;

    }

    void show()

    {

        cout << "\n"

             << hrs << ":" << min << ":" << sec;

    }

    time operator+(time t2)

    {

        time t3;

        t3.sec = sec + t2.sec;

        t3.min = min + t2.min + (t3.sec / 60);

        t3.sec = t3.sec % 60;

        t3.hrs = hrs + t2.hrs + (t3.min / 60);

        t3.min = t3.min % 60;

        return t3;

    }

};

int main()

{

    time t1, t2, t3;

    t1.get();

    t2.get();

    t1.show();

    t2.show();

    t3 = t1 + t2;

    t3.show();

    return 0;

}

Output:

Enter time (in hrs:minutes:seconds form): 10 30 40 Enter time (in hrs:minutes:seconds form): 5 40 30 10:30:40 // time 1

**c.Overload the + for concatenating the two strings. For e.g “Py” + “thon” = Python**

**Ans:-**

#include<iostream>

#include<string.h>

using namespace std;

class strings

{

    char str[100];

    public:

    void input();

    void output();

    strings operator+(string s);

};

void strings::input()

{

    cout<<"\nenter the string :\t\t";

    cin>>str;

}

strings strings::operator+(string s)

{

    strings temp;

    strcpy(temp.str,str);

    strcat(temp.str,s.str);

    return temp;

}

void strings::output()

{

    cout<<"\nthe string is:\t\t";

    cout<<"\t"<<str;

}

int main()

{

    strings s1,s2,s3;

    s1.input();

    s2.input();

//  s3=s1+s2;

    s3.output();

    return 0;

}

Output

Enter a string: Sanjeel

Enter a string: Sagar

String 1

Sanjeela

String 2

Sagar

After concatenation String 3

SanjeelaSagar

**5.Inheritance**

**a.Design a class for single level inheritance using public and private type derivation.**

**Ans:-**

(1)Usingpublictypederivation:

    {

**Output:**

Entervalueforn:5

Entervalueforb:7

**N=5**

(2)Usingprivatetypederivation:

#include<iostream>

using namespace std;

class base

{

int n;

public:

void get()

{

cout<<"\nEntervalueforn:";

cin>>n;

}

void show()

{

cout<<"\n\t\tN="<<n;

}

};

class derived:private base

{

int b;

public:

void get()

{

base::get();

          cout<<"\nEntervalueforb:";

          cin>>b;

}

void display()

{

show();

}

};

int main()

{

derived d1;

d1.get();

d1.display();

return 0;

}

**Output:**

Entervalueforn:5

Entervalueforb:6

N=5

**b.Design a class for multiple inheritance.**

**Ans:-**

#include <iostream>

using namespace std;

class internal

{

    int n;

public:

    void get()

    {

        cout << "\nEntern:";

        cin >> n;

    }

    int n\_return()

    {

        return n;

    }

    void show()

    {

        cout << "\n\nInternalmarks:" << n;

    }

};

class external

{

    int m;

public:

    void get()

    {

        cout << "\nEnterm:";

        cin >> m;

    }

    int m\_return()

    {

        return m;

    }

    void show()

    {

        cout << "\nM:" << m;

    }

};

class final : public internal, public external

{

    float tot;

public:

    void get()

    {

        internal::get();

        external::get();

    }

    void show()

    {

        tot = internal::n\_return() + external::m\_return();

        cout << "\nTotal:" << tot;

    }

};

int main()

{

    final t;

    t.get();

    t.show();

    return 0;

}

**output:**

Entern:5

Enterm:4

**Total :9**

**c.Implement the hierarchical inheritance**

**Ans:-**

#include <iostream>

using namespace std;

class Number

{

private:

    int num;

public:

    void getNumber()

    {

        cout << "\n\nEnter an integer number: ";

        cin >> num;

    }

    int returnNumber(){

        return num;

    }

};

class Square : public Number

{

public:

    int getSquare()

    {

        int num1, sqr;

        num1 = returnNumber();

        sqr = num1 \* num1;

        return sqr;

    }

};

class Cube : public Number

{

public:

    int getCube()

    {

        int num2, cube;

        num2 = returnNumber();

        cube = num2 \* num2 \* num2;

        return cube;

    }

};

int main()

{

    Square objS;

    Cube objC;

    int sqr, cube;

    objS.getNumber();

    sqr = objS.getSquare();

    cout << "Square of " << objS.returnNumber() << " is: " << sqr << endl;

    objC.getNumber();

    cube = objC.getCube();

    cout << "Cube   of " << objC.returnNumber() << " is: " << cube << endl;

}

**output:**

Enterstudent'sdata

Enternameandage:Sanjeela33

Entermarks:72

Enteremployee'sdata

Enternameandage:Rakesh35

Enterdesignation:Manager

student'sData

Name:Sanjeela

Age:33

Marks:72

**6.Virtual functions and abstract classes**

**a. Implement the concept of method overriding.**

**Ans:**

#include <iostream>

using namespace std;

class employee

{

    int emp\_code, age;

    char name[30], qualification[30];

public:

    void get()

    {

        cout << "\nEnteremployeeid:";

        cin >>emp\_code;

        cout << "\nEnteremployeename:";

        cin >> name;

        cout << "\nEnteremployeeage:";

        cin >> age;

        cout << "\nEnteremployeequalification:";

        cin >> qualification;

    }

    void show()

    {

        cout << "\n\nEmployeeid:" << emp\_code;

        cout << "\tName:" << name;

        cout << "\nAge:" << age << "\t\tQualification:" << qualification;

    }

};

class contract\_employee : public employee

{

    int contract\_id;

public:

    void get()

    {

        cout << "\nEntercontract\_id:";

         cin >>contract\_id;

    }

    void show()

    {

        cout << "\nContractID:" << contract\_id;

    }

};

int main()

{

    contract\_employee ce;

    ce.get();

    ce.show();

    return 0;

}

**Output:-**

Entercontract\_id:101

Contact\_id:101

**b. Show the use of virtual function**

**Ans:**

#include<iostream>

using namespace std;

class base

{

public:

virtual void display()

{

cout<<"\nDisplay of base class called";

}

};

class derived:public base

{

public:

void display(){

cout<<"\nDisplay of derived class called";

}

};

int main()

{

base \*b;

derived d;

b= &d;

b->display();

return 0;

}

**Output:**

Displayofderivedclasscalled**c. Show the implementation of abstract class.**

**Ans:**

#include<iostream>

using namespace std;

class shape

{

public:

    virtual int getarea()=0;

    void setwidth(int w)

    {

        width=w;

    }

    void setheight(int h)

    {

        height=h;

    }

protected:

    int width;

    int height;

};

class rectangle:public shape

{

public:

    int getarea()

    {

        return(width\*height);

    }

};

class triangle:public shape

{

public:

    int getarea()

    {

        return(width\*height)/2;

    }

};

int main ()

{

    rectangle rect;

    triangle tri;

    rect.setwidth(5);

    rect.setheight(7);

    cout<<"total rectangle area:"<<rect.getarea()<<endl;

    tri.setwidth(5);

    tri.setheight(7);

    cout<<"total triangle area :"<<tri.getarea()<<endl;

    return 0;

}

**Output:**

total rectangle area:35

total triangle area :17**7. String handling**

**a. String operations for string length , string concatenation**

**Ans:**

#include<iostream>

#include<string.h>

using namespace std;

int main()

{

string str1="Sanjeela";

string str2="Sagar";

cout<<endl<<"Length of "<<str1<<": "<<str1.length();

string str3=str1+str2;

cout<<endl<<str3;

return 0;

}

**Output :**

Length of Sanjeela: 8

SanjeelaSagar

**b. String operations for string reverse, string comparison,**

**Ans:-**

#include<iostream>

#include<string.h>

#include<algorithm>

using namespace std;

int main()

{

string str="Hello,Its going to reverse";

reverse(str.begin(),str.end());

cout<<str;

string s1="Sanjeela";

string s2="Sagar";

if(s1<s2)

cout<<endl<<s1<<"comes before"<<s2;

else

cout<<endl<<s2<<"comes before"<<s1;

return 0;

}

**Output:-**

esreverotgniogstI,olleH

**c. Console formatting functions.**

**Ans:**

**EXAMPLE 1:Without using iomanip setw() method.**

#include<iostream>

using namespace std;

int main()

{

char s[10];

cout<<"Enter your name:";

cin>>s;

cout<<endl<<s;

return 0;

}

Output:

Enteryour name:SanjeelaSagarSanjeelaSagar

**With using iomanip setw() method.**

#include<iostream>

#include<iomanip>

using namespace std;

int main()

{

char s[10];

cout<<"Enter you rname:";

cin>>setw(10)>>s;

cout<<endl<<s;

return 0;

}

Output:

Entery our name:SanjeelaSagar

SanjeelaS

**Example2:**

#include<iostream>

using namespace std;

int main()

{

char c[]="Sanjeela Sagar";

cout.write(c,13).put('\n');//put('\n')isusedinplaceofendl

char ch[]="A";

cout<<"ch=";

cout.write(ch,1)<<endl;//writesonebyteofch.

char s[]="ABCDEFGHIJC";

cout.write(s,5)<<endl;//writes5bytesfromstrings

char name[15];

cout<<"Enter a name:";

cin.read(name,15);//reads15bytesfromname

cout.write(name,15)<<endl;//writes15bytesfromName

return 0;

}

Output:

SanjeelaSaga

ch=A

ABCDE

Enteraname:SanjeelaRakeshSagar

SanjeelaRakesh

**8.Exception handling**

**a. Show the implementation of exception handling**

**Ans:**

#include<iostream>

using namespace std;

int main()

{

float percent;

cout<<"Enter your percentage:";

cin>>percent;

try

{

if(percent<0||percent>100)

throw(percent);

else

cout<<endl<<"Your percentage:"<<percent;

}catch(int p)

{

cout<<endl<<"Invalid percentage:"<<p;

}

return 0;

}

**Output:**

Enter percentage:150

Invalid percentage:150

**b. Show the implementation for exception handling for strings.**

**Ans:**

#include <iostream>

#include <string.h>

using namespace std;

int main()

{

    string s;

    cout << "Enter the name of your course:";

    cin >> s;

    try

    {

        if (s == "B.Sc-IT" || s == "BMS" || s == "B.Com")

            cout << endl

                 << "Your have chosen Course:" << s;

        else

            throw(s);

    }

    catch (string ss)

    {

        cout << endl

             << "Oh!!!!!!!!! you have chosen the course that we don't provide:" << ss;

    }

    return 0;

}

**Output:**

1stRun:

Enterthenameofyourcourse:MCA

Oh!!!!!!!!!youhavechosenthecoursethatwedon'tprovide:MCA

2ndRun:

Enterthenameofyourcourse:BMS

**9.File handling**

**a. Design a class FileDemo open a file in read mode and display the total number of words and lines in the file.**

**Ans:**

#include<iostream>

#include<fstream>

using namespace std;

int main()

{

ofstream fwrite("1.txt");

fwrite<<"hello\nhi hey\nworld\n";

fwrite.close();

ifstream fread("1.txt");

int w=0,l=0;

char c;

while(fread)

{

fread.get(c);

if(c==' '|| c=='\n')

w++;

if(c=='\n')

l++;

}

fread.close();

cout<<"\n Total no. of words in the file:"<<w;

cout<<"\n Total no. of lines in the file:"<<l;

return 0;

}

**Output:**

Total no. of words in the file:5

Total no. of lines in the file:4

**b. Design a class to handle multiple files and file operations**

**Ans:**

#include<iostream>

#include<fstream>

using namespace std;

int main()

{

ofstream fwrite("Alphabets.txt");

fwrite<<"ABCDEFGHIJKLMNOPQRSTUVWXYZ";

fwrite.close();

ifstream fread("Alphabets.txt");

ofstream fwrite1("Vowels.txt");

ofstream fwrite2("Consonants.txt");

char c;

while(fread)

{

fread.get(c);

if(c=='A'||c=='E'||c=='I'||c=='O'||c=='U')

fwrite1<<c;

else

fwrite2<<c;

}

fread.close();

fwrite1.close();

fwrite2.close();

fread.open("Alphabets.txt");

ifstream fread1("Vowels.txt");

ifstream fread2("Consonants.txt");

cout<<"\n\nContents of Alphabets File\n";

cout<<"................\n";

while(fread)

{

fread.get(c);

cout<<c;

}

fread.close();

cout<<"\n\nContents of Vowels File\n";

cout<<"..................\n";

while(fread1)

{

fread1.get(c);

cout<<c;

}

fread1.close();

cout<<"\n\nContents of Consonants File\n";

cout<<"...............\n";

while(fread2)

{

fread2.get(c);

cout<<c;

}

fread2.close();

return 0;

}

**Output:**Contents of Consonants File

...............

BCDFGHJKLMNPQRSTVWXYZZZ

**c. Design a editor for appending and editing the files**

**Ans:**

#include<iostream>

#include<fstream>

using namespace std;

class student

{

char name[30];

int age;

float percent;

public:

void getdata()

{

cout<<endl<<"Enter name: ";

cin>>name;

cout<<endl<<"Enter age: ";

cin>>age;

cout<<endl<<"Enter percentage: ";

cin>>percent;

}

void showdata()

{

cout<<endl<<name;

cout<<"\t\t"<< age;

cout<<"\t\t"<<percent;

}

};

int main()

{

student st;

fstream freadwrite("Student.txt", ios::ate | ios::in | ios::out | ios::trunc);

for(int i=0;i<4;i++)

{

st.getdata();

freadwrite.write((char\*)&st,sizeof(st));

}

freadwrite.seekg(0,ios::beg);

cout<<endl<<"Current contents of file";

while(freadwrite.read((char\*)&st,sizeof(st)))

st.showdata();

freadwrite.clear();

cout<<endl<<"Enter details for one more student";

st.getdata();

//char c;

//cin.get(c);

freadwrite.write((char\*)&st, sizeof(st));

freadwrite.seekg(0);

cout<<endl<<"After addition of one more student";

cout<<endl<<"Name \t\t Age \t\t Percentage";

while(freadwrite.read((char\*)&st, sizeof(st)))

{

st.showdata();

}

int n = freadwrite.tellg() / sizeof(st);

cout<<endl<<"Total no. of student record: "<<n;

cout<<endl<<"Enter student number to be updated: ";

int num;

cin>>num;

//cin.get(c);

int l=(num-1) \* sizeof(st);

if(freadwrite.eof())

 freadwrite.clear();

 freadwrite.seekp(l);

 cout<<endl<<"Enter new values for the student";

st.getdata();

//cin.get(c);

 freadwrite.write((char\*)&st, sizeof(st))<<flush;

 freadwrite.seekg(0);

cout<<endl<<"After updation contents are";

cout<<endl<<"Name \t\t Age \t\t Percentage";

while(freadwrite.read((char\*)&st, sizeof(st)))

{

 st.showdata();

}

freadwrite.close();

return 0;

}

**Output:**

**Enter name: avinash**

**Enter age: 12**

**Enter percentage: 323**

**Enter name: 64**

**Enter age: 64**

**Enter percentage: 65**

**Enter name: 43**

**Enter age: 54**

**Enter percentage: 65**

**Enter name: 65**

**Enter age: 9**

**Enter percentage: 8**

**Current contents of file**

**avinash 12 323**

**64 64 65**

**43 54 65**

**65 9 8**

**Enter details for one more student**

**Enter name: 9**

**Enter age: 9**

**Enter percentage: 9**

**After addition of one more student**

**Name Age Percentage**

**avinash 12 323**

**64 64 65**

**43 54 65**

**65 9 8**

**9 9 9**

**Total no. of student record: 0**

**Enter student number to be updated:**