INDEX

Name : Mani Ratnam

Enrolment No. :

Branch : Computer Science and Technology

Group : 3CST1

S.No.	Name of Program	Date Performed	Date Checked	Remarks	Signature
1.	Write a program for multiplication of two matrices using OOPs.				
2.	Create a class Student, which have data members as name, branch, roll no., age, sex, five subjects. Display the name of student and his percentage who has more than 70%. Use array of objects.				
3.	Using the concept of function overloading, write function for calculating area of triangle, circle and rectangle.				
4.	Create a class TIME with members hours, minutes and seconds. Take input, add two-time objects and passing objects to function and display the result.				
5.	Write a program to find the greatest of two given numbers in two different classes using friend function.				
6.	Write a program to find the biggest of three numbers using Friend Function.				
7.	Write a program to demonstrate the use of friend function with Inline assignment.				

8.	Write a program to find the sum of two numbers declared in a class and display the numbers and sum using friend class.				
9.	Write a program to perform addition of two complex numbers using constructor overloading. The first constructor which takes no argument is used to create objects which are not initialized, second which takes one argument is used to initialize real and imag parts to equal values and third which takes two arguments is used to initialize real and imag to two different values.				
10.	Write a program to generate a fibonacci series using Copy Constructor.				
11.	Create a class which keep track of number of its instances. Use static data member, constructors and destructors to maintain updated information about active objects.				
12.	Implement a class String containing the following functions: i. Overload + operator to carry out the concatenation of strings. ii. Overload = operator to carry out string copy. iii. Overload <= operator to carry out the comparison of strings. iv. Function to display the length of the string. v. Function toLower() to convert upper case to lower case. vi. Function toUpper() to convert lower case to upper case.				
13.	Write a program to overload unary increment(++) operator.				
14.	Write a program to overload new and delete operator.				

15.	Create a class basic_info with data members name, roll no., sex and two member function get data and display. Derive a class physical_fit form basic_info which has data members height and weight and member functions get data and display. Display all the information using object of derived class.					
16.	Create class first with data members book no., book name and member function get data and put data. Create a class second with data members author name, publisher and member function get data and show data. Derive a class third from first and second with data member no. of pages and year of publication. Display all this information using array of objects of third class.					
17.	Design three classes STUDENT, EXAM and RESULT. The STUDENT class has data members such as roll no, name. Create a class EXAM by inheriting the STUDENT class. The EXAM class adds data members representing the marks scored in six subjects. Derive the RESULT from the EXAM class and has its own data members such as total marks. WAP to model this relationship.					
18.	Create a base class called SHAPE. Use this class to store two double type values. Derive two specific classes called TRIANGLE and RECTANGLE from the base class. Add to the base class, a member function get data to initialize base class data members and another member function display to compute and display the area of figures. Make display a virtual function and redefine this function in the derived class to suit their requirements. Using these three classes design a program that will accept driven of a TRIANGLE or RECTANGLE interactively and display the area.					

19.	Create a class LIST with two pure virtual function store() and retrieve(). To store a value call store and to retrieve call retrieve function. Derive two classes stack and queue from it and override store and retrieve.					
20.	Write a program to define the function template for swapping item of various data type such as integers, float and characters.					
21.	Write a program to define the function template for calculating the square of given numbers with different data types.					
22.	Write a program to illustrate how Template function can be overload.					
23.	Write a program to illustrate how define and declare a class template for reading two data items from the keyboard and two find their sum.					
24.	Write a program to demonstrate the use of special functions, constructor and destructor in the class template. The program is used to find the bigger of two entered numbers.					
25.	Write a program to raise an exception if any attempt is made to refer to an element whose index is beyond the array size.					
26.	Write a program to read a set of line from the keyboard and to store it on a specified file.					
27.	Write a program to read a text file and display its contents on the screen.					
28.	Write a program to copy the contents of a file to another.					
29.	Write a program to read two numbers and then divide the first number by second number and raise an exception if second number is zero.					
30.	Write a program to perform the deletion of white spaces such as horizontal tab, vertical tab, space, line feed, new line and carriage return from a text file and store the contents of					

	the file without the white spaces on another file.					
31	Write a program to read the class object of student info such as name, age, sex, height and weight from the keyboard and to store them on a specified file using read() and write() functions. Again the same file is opened for reading and displaying the contents of the file on the screen.					

EXPERIMENT 1

AIM:- Write a program for multiplication of two matrices using OOP.

Source Code:

```
#include <iostream>
using namespace std;

class Matrix
{
   int row;
   int col;
   int arr[50][50];
   public:
      void input()
      {
         /*
           take matrix input from the user.
         */
         cout<<"\nEnter Row size of Matrix: ";
         cin>>row;
         cout<<"Enter Column size of Matrix: ";
         cin>>col;
         if(row<1 || col<1 )
          {
            cout<<"Matrix can not be empty!!"<<endl;
         }
         if(row>=50 || col>50){
            cout<<"Matrix size is out of Range"<<endl;
         }
         else{
            cout<<"Enter the total "<<row*col<< " Values of the matrix :-"<<endl;
            for(int i=0; i<row; i++)
            {
               for(int j=0; j<col; j++)
               {
                  cin>>arr[i][j];
               }
            }
         }
```

```
    }
    void display()
        {
       /* function print the Matrix */
       cout<<"\n--Matrix --"<<endl;
       for(int i=0; i<row; i++)
        {
          for(int j=0; j<col; j++)
        {
              cout<<arr[i][j]<<" ";
          }
         cout<<endl;
       }
    }

    void mul(Matrix a, Matrix b)
        {
       /*
          mul function take 2 matrix as an input
          It multiplies matrix as AxB
          If columns of A are not equal to rows of B multiplication can not be done.
       */
       if(a.col != b.row){
          cout<<"Multiplication Can not be done."<<endl;
       }
       else{
          row = a.row;
          col = b.col;
          for(int i=0; i<a.row; i++)
          {
             for(int j=0; j<b.col; j++)
             {
                arr[i][j]=0;
                for(int k=0; k<a.row; k++)
                 {
                    arr[i][j]+= (a.arr[i][k]*b.arr[k][j]);
                }
             }
          }
       }
    }
};
```

```
int main() {
   Matrix M1,M2,M3;
   M1.input();
   M1.display();
   M2.input();
   M2.display();
   M3.mul(M1,M2);
   M3.display();

   return 0;
}
```

Output:

```
Enter Row size of Matrix: 3
Enter Column size of Matrix: 3
Enter the total 9 Values of the matrix :-
8 1 1 6 9 3 6 0 2
--Matrix --
8 1 1
6 9 3
6 0 2

Enter Row size of Matrix: 3
Enter Column size of Matrix: 3
Enter the total 9 Values of the matrix :-
9 8 1 1 6 9 3 6 0
--Matrix --
9 8 1
1 6 9
3 6 0

--Matrix --
76 76 17
72 120 87
60 60 6
```

EXPERIMENT 2

AIM:- Create a class Student, which have data members as name, branch, roll number, age, sex, five subjects. Display the name of the student & his percentage who has more than 70%. Use an array of objects. [Array of objects]

Source Code:

```
#include <iostream>
#include <iomanip> // setprecision
using namespace std;

class Student{
   string name;
   string branch;
   int rollNo;
   int age;
   char gen;
   int fiveSubj[5];
   public:
      float prcntg;
      void input(){
         cout<<"Student Name: ";
         cin>>name;
         cout<<"Student Branch: ";
         cin>>branch;
         cout<<"Student Roll No.: ";
         cin>>rollNo;
         cout<<"Student Age: ";
         cin>>age;
         cout<<"Student Gender[M/F]: ";
         cin>>gen;
         cout<<"Student Marks in Five Subject -"<<endl;
         int total = 0;
         for(int i=0; i<5; i++){
            cin>>fiveSubj[i];
            total += fiveSubj[i];
         }
         prcntg = total/5;
      }

      void display(){
         cout<<"Name: "<<name<<endl;
         cout<<"Branch: "<<branch<<endl;
         cout<<"Roll No.: "<<rollNo<<endl;
```

```
        cout<<"Age: "<<age<<endl;
        cout<<"Gender[M/F]: "<<gen<<endl;
        cout<<"Percentage: "<< fixed << setprecision(2) << prcntg<<endl;
    }
    void printNamePerc(){
        cout<<"Name: "<<name<<endl;
        cout<<"Percentage: "<< fixed << setprecision(2) << prcntg<<"%"<<endl;
    }
};

int main() {
    Student arr[10]; // array of Objects;
    int n;
    cout<<"Enter Total No. of Students: ";
    cin>>n;
    for(int i=0; i<n; i++){
        cout<<"Enter Student "<<i+1<< " details:"<<endl;
        arr[i].input();
        cout<<endl;
    }
    //Display the name & percentage of Student who has more than 70%
    cout<<"\nStudents with 70% plus marks :-"<<endl;
    for(int i=0; i<n; i++){
        if(arr[i].prcntg>70){
            arr[i].printNamePerc();
        }
    }

    return 0;
}
```

Output:

```
Enter Total No. of Students: 3
Enter Student 1 details:
Student Name: MANI
Student Branch: CST
Student Roll No.: 045
Student Age: 19
Student Gender[M/F]: M
Student Marks in Five Subject -
85
80
75
95
96
Enter Student 2 details:
Student Name: ANKIT
Student Branch: CST
Student Roll No.: 046
Student Age: 20
Student Gender[M/F]: M
Student Marks in Five Subject -
85
96
70
98
7
Enter Student 3 details:

Student Name: ANSHU
ANSHU
Student Branch: ITE
Student Roll No.: 050
Student Age: 21
Student Gender[M/F]: M
Student Marks in Five Subject -
45
65
75
71
33
Students with 70% plus marks :-
Name: MANI
Percentage: 86.00%
Name: ANKIT
Percentage: 71.00%
```

EXPERIMENT 3

AIM:- Write a program to find the area of a triangle, circle and rectangle using the concept of function overloading.

Source Code:

```
#include<iostream>
#include<cstdlib>
using namespace std;


float area(float r){
return(3.14 * r * r);
}
float area(int b,float h){
return(0.5 * b * h);
}
int area(int l,int b){
return (l * b);
}
int main()
{
float h,r;
int l,b;
int ch;
ch=0;
while(ch!=4)
{
cout<<"\n 1. Area of Circle \n 2. Area of Triangle \n 3. Area of Rectangle \n 4. Exit \n Enter Your Choice :  ";
cin>>ch;
switch(ch)
{
case 1: {
cout<<"\n Enter the Radius of Circle : ";
cin>>r;
cout<<"\n Area of Circle : "<<area(r);
break;
}
case 2:
{
cout<<"\n Enter the Base & Height of Triangle : ";
cin>>b>>h;
cout<<"\n Area of Triangle : "<<area(b,h);
```

```
break;
}
case 3:
{
cout<<"\n Enter the Length & Bredth of Rectangle : ";
cin>>l>>b;
cout<<"\n Area of Rectangle : "<<area(l,b);
break;
}
case 4:
exit(0);
default:
cout<<"\n Invalid Choice... ";
}
}
return 0;
}
```

Output:

Output

```
/tmp/aBcmXaMjqx.o
1. Area of Circle
 2. Area of Triangle
 3. Area of Rectangle
 4. Exit
 Enter Your Choice :  3
 Enter the Length & Bredth of Rectangle : 6
 5
 Area of Rectangle : 30
 1. Area of Circle
 2. Area of Triangle
 3. Area of Rectangle
 4. Exit
 Enter Your Choice :  2
 Enter the Base & Height of Triangle : 6
 5
 Area of Triangle : 15
 1. Area of Circle
 2. Area of Triangle
 3. Area of Rectangle
 4. Exit
 Enter Your Choice :  1
 Enter the Radius of Circle : 6
 Area of Circle : 113.04
 1. Area of Circle
 2. Area of Triangle
 3. Area of Rectangle
 4. Exit
 Enter Your Choice :  4
```

EXPERIMENT 4

AIM:- Create a class TIME with members hours, minutes and seconds. Take input, add two-time objects and passing objects to function and display the result.

Source Code:

```
#include <iostream>
using namespace std;
class Time{
private:
	int hours, minutes, seconds;
public:
	void input(){
	   cout<<"\nHours: ";
	   cin>>hours;
	   cout<<"Minutes: ";
	   cin>>minutes;
	   cout<<"Second: ";
	   cin>>seconds;
	}
	void display(){
		cout <<"("<< hours << ":" << minutes << ":" << seconds <<")"<< endl;
	}
	void addTwoTime(Time a, Time b){
		hours = a.hours + b.hours;
		minutes = a.minutes + b.minutes;
		seconds = a.seconds + b.seconds;

5		if(minutes>=60 || seconds>=60){
			int secUp = seconds/60;
			int secReminder = seconds%60;
			minutes = minutes+secUp;
			int minUp = minutes/60;
			int minReminder = minutes%60;

			hours = hours + minUp;
			minutes = minReminder;
			seconds = secReminder;
		}
	}
};
int main(){
	Time T1, T2, T3;
```

```
        T1.input();
        cout<<"\nTime 1: ";
        T1.display();
        T2.input();
        cout<<"\nTime 2: ";
        T2.display();

        T3.addTwoTime(T1,T2);
        cout<<"\nSum of Two Time is: ";
        T3.display();

        return 0;
}
```

OUTPUT:

```
Hours: 4
Minutes: 35
Second: 26
Time 1: (4:35:26)

Hours: 3
Minutes: 50
Second: 45
Time 2: (3:50:45)

Sum of Two Time is: (8:26:11)
```

EXPERIMENT 5

AIM:- Write a program to find the greatest of two given numbers in two different classes using the friend function.

Source Code:

```
#include<iostream>
using namespace std;

class First;// forward declaration

class Second{
   int b;
   public:
      Second(int x){ //Constructor
         b=x;
      }
   friend void max(First,Second);
};

class First{
   int a;
   public:
      First(int x){ //Constructor
         a=x;
      }
   friend void max(First,Second);
};

void max(First t1,Second t2){
   if(t1.a>t2.b)
      cout<<t1.a<<endl;
   else
      cout<<t2.b<<endl;
}

int main(){
   int m,n;
   cout<<"Enter Value (m): ";
   cin>>m;
   cout<<"Enter Value (n): ";
   cin>>n;

   First objF(m);
```

```
    Second objS(n);
    cout<<"Maximum value: ";
    max(objF,objS); //callin friend function

    return 0;
}
```

Output:

```
Enter Value (m): 30
Enter Value (n): 40
Maximum value: 40
```

EXPERIMENT 6

AIM:- Write a program to find the biggest of three number using friend function.

Source Code:

```
#include<iostream>
using namespace std;

class First;// forward declaration

class Second{
   int b;
   public:
      Second(int x){
         //Constructor
         b=x;
      }
   friend void max(First,Second);
};

class First{
   int a;
   public:
      First(int x){ //Constructor
         a=x;
      }
   friend void max(First,Second);
};

void max(First t1,Second t2){
   if(t1.a>t2.b)
      cout<<t1.a<<endl;
   else
      cout<<t2.b<<endl;
}

int main(){
   int m,n;
   cout<<"Enter Value (m): ";
   cin>>m;
   cout<<"Enter Value (n): ";
   cin>>n;
```

```
    First objF(m);
    Second objS(n);
    cout<<"Maximum value: ";
    max(objF,objS); //callin friend function

    return 0;
}
```

OUTPUT:

```
Output

/tmp/aBcmXaMjqx.o
Enter Value (m): 78
Enter Value (n): 88
Enter Value (o): 99
Maximum value: 99
```

EXPERIMENT 7

AIM:- Write a program to demonstrate the use of friend function using inline assignment.

Source Code:

```
#include<iostream>
using namespace std;
class Inline
{
int a,b;
public:
Inline(int x,int y)
{
a=x;
b=y;
}
friend  int sum(Inline);
};
int sum(Inline number)
{
return (number.a=10)+(number.b=10);
}
int main()
{
Inline obj(0,0);
cout<<"passed value for a is 0";
cout<<"\npassed value for b is 0";
cout<<"\nbut the sum after inline assignment is  "<<sum(obj);}
```

Output:

```
Output

/tmp/ZOKXR8giv5.o
passed value for a is 0
passed value for b is 0
but the sum after inline assignment is  20
```

EXPERIMENT 8

AIM: Write a Program to find the sum of two number declared in a class and display the number and sum using friend function

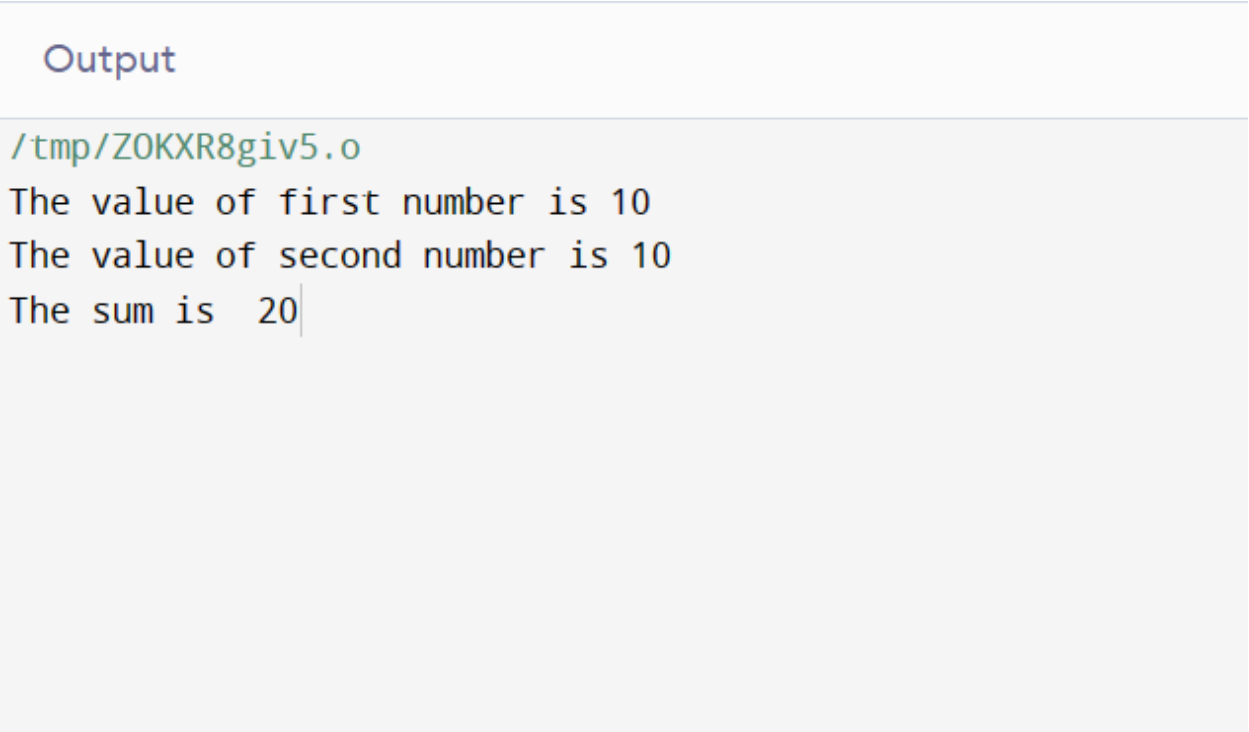
Source code:

```
#include<iostream>

using namespace std;

class Inline

{

int a,b;

public:

Inline(int x)

{

a=10;

b=10;

}

friend  int sum(Inline);

};

int sum(Inline number)

{

cout<<"The value of first number is "<<number.a;

cout<<"\nThe value of second number is "<<number.b;

cout<<"\nThe sum is  "<<(number.a+number.b);

}

int main()

{

Inline obj(0);
```

sum(obj);

}

Output:



EXPERIMENT 9

AIM:- Write a program to perform addition of two complex numbers using constructor overloading. The first constructor which takes no argument is used to create objects which are not initialized, second which takes one argument is used to initialize real and imag parts to equal values and third which takes two arguments is used to initialize real and imag to two different values.

Source Code:

```
#include <iostream>
using namespace std;

class Complex{
   int real;
   int imag;
   public:
      Complex(){
         /* Constructor Take No argument */
         cout<<"Constructor with No argument!"<<endl;
      }
      Complex(int a){
         /* Take One argument and initialize real and imag*/
         real = a;
         imag = a;
         cout<<"Constructor with One argument!"<<endl;
      }
      Complex(int a,int b){
         /* Takes Two arguments*/
         real = a;
         imag = b;
         cout<<"Constructor with Two arguments!"<<endl;
      }
      void display(){
         cout<<"("<<real<<" + "<<imag<<")"<<endl;
      }
      void add(Complex a, Complex b){
         real = a.real+b.real;
         imag = a.imag + b.imag;
      }
};

int main() {
   Complex C1(4,5);
```

```
    Complex C2(3);
    Complex C3;
    cout<<"C1: ";
    C1.display();
    cout<<"C2: ";
    C2.display();
    C3.add(C1,C2);
    cout<<"C3: ";
    C3.display();

    return 0;
}
```

Output:

```
Output

/tmp/aBcmXaMjqx.o
Constructor with Two arguments!
Constructor with One argument!
Constructor with No argument!
C1: (4 + 5)
C2: (3 + 3)
C3: (7 + 8)
```

EXPERIMENT 10

AIM: Write a program to generate a Fibonacci series using Copy Constructor.

Source Code:

```
#include <iostream>
using namespace std;

class Fibo
{
   int a,b,next;
   public:
      Fibo()
          {
      a=0;
      b=1;
      next=a+b;
      }

      Fibo (Fibo &temp){
      a=temp.a;
      b=temp.b;
      next=temp.next;
      }

      void nextNum(){
      a=b;
      b=next;
      next=a+b;
      }

      void display(){
         cout << next <<" ";
      }
};
int main (){
   Fibo number;
   int n;
   cout<<"Enter the Range: ";
   cin>>n;
   for (int i=0; i<n;i++){
      number.display();
      number.nextNum();
```

```
  }
}
```

Output:

```
Enter the Range: 15
1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
```

EXPERIMENT 11

AIM: Create a class which keep track of number of its instances. Use static data member, constructors and destructors to maintain updated information about active objects.

Source Code:

```
#include<iostream>
using namespace std;
class count
{
public:
static int inc;
static int dec;
count(int x)
{
inc+=1;
}
~count()
{
cout <<"\ninstance counting using destructor "<<dec++;
}
};
int count::inc=0;
int count::dec=1;
int main()
{
count obj1(1);
count obj2(2);
count obj3(3);
count obj4(4);
count obj5(5);
count obj6(6);
cout<<"no of instance counted using constructor= "<<count::inc;
}
```

Output:

Output

```
/tmp/ZOKXR8giv5.o
no of instance counted using constructor= 6
instance counting using destructor 1
instance counting using destructor 2
instance counting using destructor 3
instance counting using destructor 4
instance counting using destructor 5
instance counting using destructor 6
```

EXPERIMENT 12

AIM:- Implement a class string containing the following functions:

a. Overload + operator to carry out the concatenation of strings.
b. Overload = operator to carry out string copy.
c. Overload <= operator to carry out the comparison of strings.
d. Function to display the length of a string.
e. Function tolower( ) to convert upper case letters to lower case.
f. Function toupper( ) to convert lower case letters to upper case.

Source Code:

```
#include<iostream>
#include<conio.h>
#include<stdlib.h>
#include<string.h>
using namespace std;
class str{
public:
char str1[100], str2[100], str3[100];
void put_str_val(char s1[], char s2[]){
strcpy(this->str1, s1);
strcpy(this->str2, s2);
}
void operator+(){
cout<<"\n\nConcatenation of strings: "<<strcat(str1, str2);
}
int operator=(str s)
{
cout<<"Enter string: ";
cin>>str1;
strcpy(this->str3, str1);
strcpy(this->str2, str1);
cout<<"\nString 1 is copied in string 2 and 3: ";
cout<<"\nString1: "<<str1<<"\nString2: "<<str2<<"\nString3: "<<str3;
return 0;
}
void operator<=(str s)
{
   cout<<"Enter first String: ";
```

```
   cin>>str1;
   cout<<"Enter second string: ";
   cin>>str2;
   if(strlen(str1) <= strlen(str2)){
      cout<<str2<<" is greator than "<<str1;
   }
   else{
      cout<<str1<<" is greator than "<<str2;
   }}
void length_of_string(char st[]){
cout<<"\nString length: "<<strlen(st);
}
void lower_str(){
cout<<"Enter string: ";
cin>>str1;
for(int i=0; i<strlen(str1);i++){
 str1[i]=tolower(str1[i]);
}
cout<<"Lower case of string: "<<str1;
}
void upper_str(){
   cout<<"Enter string: ";
   cin>>str2;
   for(int i=0; i<strlen(str2);i++){
 str2[i]=toupper(str2[i]);
}
  cout<<"Lower case of string: "<<str2;
}};
int main(){
char a[100], b[100];
int ch, j=0;
//clrscr();
cout<<"1. Concatenate\n2.Copy\n3.Comparision\n4.length\n5.lowercase\n6.uppercase\n7.
Exit\n";
cin>>ch;
str s, r;
do{
switch(ch){
case 1:{
cout<<"Enter first string: ";
cin>>a;
```

```
cout<<"Enter second string: ";
cin>>b;
s.put_str_val(a, b);
+s;
break;
}
case 2:{
 r=s;
break;
}
case 3:{
r<=s;
break;
}
case 4:{
cout<<"Enter string: ";
cin>>a;
s.length_of_string(a);
break;
}
case 5:{
 s.lower_str();
break;
}
case 6:{
   s.upper_str();
   break;
}
case 7:{
cout<<"\nExiting....";
exit(0);
}}
cout<<"\nEnter choice: ";
cin>>ch;
j++;
}while(j!=8);
getch();
return 0;
}
```

Output:

```
Output
/tmp/aBcmXaMjqx.o
Enter your choise
1. Concatenate
2.Copy
3.Comparision
4.length
5.lowercase
6.uppercase
7. Exit
1
Enter first string: Mani
Enter second string:  Ratnam
Ratnam


Concatenation of strings: ManiRatnam
Enter choice: 2
Enter string: CST
String 1 is copied in string 2 and 3:
String1: CST
String2: CST
String3: CST
Enter choice: 3
Enter first String: C++
Enter second string: Oops
Oops is greator than C++
Enter choice: 4
Enter string: Overload
String length: 8
Enter choice: 5
Enter string: Operator
Lower case of string: operator
Enter choice: 6
Enter string: MANI
Lower case of string: MANI
Enter choice: 7
Exiting....
```

EXPERIMENT 13

AIM:-Write a program overload unary increment (++) operator.

Source Code:

```
#include <iostream>
using namespace std;
class Number{
private:
int n;
public:
void setData(int a){
n=a;
}
Number operator ++(){ // pre-increment
Number i;
i.n = ++n;
return i;
}
Number operator ++(int){ //Post -increment
Number i;
i.n = n++;
return i;
}
void display(){
cout << "Number: "<< n <<endl;
}
};
int main(){
Number C1,C2,C3,C4;
C1.setData(10);
C2=++C1; // pre
C1.display(); // 11
C2.display(); // 11
C3.setData(10);
C4=C3++; // post
C3.display(); // 11
C4.display(); // 10
}
```

Output:

```
Output

/tmp/ZOKXR8giv5.o
Number: 11
Number: 11
Number: 11
Number: 10
```

EXPERIMENT 14

AIM: Write a program to overload new and delete operator.

Source Code:

```
#include<iostream>
using namespace std;

class Student {
   string name;
   int age;
public:
   void display() {
      cout << "Name and Age of the Student is- " << name << " and " << age << endl;
   }
   Student(string nm,int ag) {
      name=nm;       age=ag;
   }
   void * operator new(size_t size) {
      cout << "Overloading new operator with size- " << size << endl;
      void *a= ::operator new(size);
      return a;
   }
   void operator delete(void *a) {
      cout << "Overloading delete operator" << endl;
      free(a);
   }
};

int main() {
   string nm;
   int ag;
   cout << "Enter name and age of the Student- ";
   cin >> nm >> ag;
   Student *s=new Student(nm,ag);
   s->display();
   delete s;
   return 0;
}
```

Output:

```
/tmp/arJD3mxCWE.o
Enter name and age of the Student- Mani 18
Overloading new operator with size- 40
Name and Age of the Student is- Mani and 18
Overloading delete operator
```

EXPERIMENT 15

AIM: Create a base class basic_info with data members name, roll no, sex and two member functions getdata and display. Derive a class physical fit from basic_info which has data members height and weight and member functions getdata and display. Display all the information using object of derived class. (Single Inheritance)

Source Code:

```
#include <iostream>
using namespace std;

class basic_info{
   string name;
   int rollNo;
   char gen;
   public:
      void getData1(){
         cout<<"Enter Name: ";
         cin>>name;
         cout<<"Enter Roll No.: ";
         cin>>rollNo;
         cout<<"Enter Gen(M/F): ";
         cin>>gen;
      }

      void display1(){
      cout<<"\nFrom basic_info class: "<<endl;
      cout<<"Name: \t\t"<<name<<endl;
      cout<<"Roll No: \t"<<rollNo<<endl;
      cout<<"Gender: \t"<<gen<<endl;
      }
};

class physical_fit: public basic_info{
   int height;
   int weight;
   public:
      void getData2(){
      cout<<"Enter Height (cm): ";
      cin>>height;
      cout<<"Enter (kg): ";
      cin>>weight;
```

```
    }

    void display2(){
        cout<<"From physical_fit class: "<<endl;
        cout<<"Height: \t"<<height<<endl;
        cout<<"Weight: \t"<<weight<<endl;
    }
};

int main(){
    physical_fit stu1;
    stu1.getData1();
    stu1.getData2();
    stu1.display1();
    stu1.display2();

    return 0;
}
```

Output:

Output

```
/tmp/q1hJ6T9jM3.o
Enter Name: Mani
Enter Roll No.: 45
Enter Gen(M/F): M
Enter Height (cm): 182
Enter (kg): 55
From basic_info class:
Name:       Mani
Roll No:    45
Gender:     M
From physical_fit class:
Height:     182
Weight:     55
```

EXPERIMENT 16

AIM: Create class first with data members book no, book name and member function getdata 16 and putdata. Create a class second with data members author name, publisher and members getdata and showdata. Derive a class third from first and second with data member no of pages and year of publication. Display all this information using array of objects of third class. (Multiple Inheritance)

Source code:

```
#include <iostream>
#include <string>
using namespace std;

class first
{
   int book_no;
   string book_name;
   public:
   void getdata1()
   {
      cout<<"\nEnter the book number ";
      cin>>book_no;
      cout<<"\nEnter the name of  book  ";
      cin>>book_name;
   }
      void putdata()
   {
      cout<<"\nBook no = "<<book_no;
      cout<<"\nName of book = "<<book_name;
   }
};
class second : public first
{
   string author_name;
   string publisher;
   public:
   void getdata2()
   {
      cout<<"\nEnter the name of author ";
```

```
      cin>>author_name;
      cout<<"\nEnter the name of publisher ";
      cin>>publisher;
   }
      void showdata()
   {
      cout<<"\nName of Author = "<<author_name;
      cout<<"\nName of publisher = "<<publisher;
   }
};
class third : public second
{
   int no_of_pages;
   int year_of_publication;
   public:
   void getdata3()
   {
      cout<<"\nEnter the total number of pages of book ";
      cin>>no_of_pages;
      cout<<"\nEnter the year of publiction of  book  ";
      cin>>year_of_publication;
   }
   void display()
   {
      cout<<"\nTotal number of pages = "<<no_of_pages;
      cout<<"\nyear of publication = "<<year_of_publication;
   }
};
int main()
{
   int n;
   cout<<"Enter the total number of books ";
   cin>>n;
   third obj[n];
   for(int i=0;i<n;i++)
   {
      cout<<"\nEnter the data of Book "<<i+1;
      obj[i].getdata1();
      obj[i].getdata2();
```

```
        obj[i].getdata3();
    }
    cout<<"-----------------------------------------------------------";
    for(int i=0;i<n;i++)
    {
        cout<<"\nDetails of Book "<<i+1;
        obj[i].putdata();
        obj[i].showdata();
        obj[i].display();
    }

}
```

Output:

Output

```
/tmp/HAPphzp1Y5.o
Enter the total number of books 1
Enter the data of Book 1
Enter the book number 456
Enter the name of  book  Oops
Enter the name of author Balaguruswamy
Enter the name of publisher Mct
Enter the total number of pages of book 885
Enter the year of publiction of  book  2007
-----------------------------------------------------------------
Details of Book 1
Book no = 456
Name of book = Oops
Name of Author = Balaguruswamy
Name of publisher = Mct
Total number of pages = 885
year of publication = 2007
```

EXPERIMENT 17

AIM: Design three classes STUDENT, EXAM, and RESULT. The STUDENT class has data members such as roll no, name. Create a class EXAM by inheriting the STUDENT class. The EXAM class adds data members representing the marks scored in six subjects. Derive the RESULT from the EXAM class and has its own data members such as total marks. WAP to model this relationship.(Multilevel Inheritance)

Source code:

```
#include <iostream>
#include <string>
using namespace std;

class STUDENT{
string name;
int rollNo;
public:
void getData1(){
cout<<"Enter Name: ";
cin>>name;
cout<<"Enter Roll No.: ";
cin>>rollNo;
}
void display1(){
cout<<"Name: \t\t"<<name<<endl;
cout<<"Roll No: \t"<<rollNo<<endl;
}
};

class EXAM {
int sixSubj[6];
public:
void input(){
for(int i=0; i<6; i++){
   cout<<"Enter Mark of Subject "<<i+1<<endl;
   cin>>sixSubj[i];
}
}
```

```
int add()
{
int t = 0;
for(int i=0; i<6; i++){
t+=sixSubj[i];
}
return t;
}
};

class RESULT : public EXAM, public STUDENT{
public:
int total=0;
int getresult()
{
total = EXAM::add();
}

};

int main(){
RESULT stu1;
stu1.getData1();
stu1.input();
stu1.display1();
stu1.add();
stu1.getresult();
cout<<"Total marks=    "<<stu1.total;

return 0;
}
```

Output:

```
/tmp/u9PrpRuEwl.o
Enter Name: Mani
Enter Roll No.: 45
Enter Mark of Subject 1
85
Enter Mark of Subject 2
75
Enter Mark of Subject 3
80
Enter Mark of Subject 4
90
Enter Mark of Subject 5
86
Enter Mark of Subject 6
78
Name:       Mani
Roll No:    45
Total marks=    494
```

EXPERIMENT 18

AIM: Create a base class called *SHAPE*. Use this class to store two double type values. Derive two specific classes called *TRIANGLE* and *RECTANGLE* from the base class. Add to the base class, a member function getdata to initialize base class data members and another member function display to compute and display the area of figures. Make display a virtual function and redefine this function in the derived classes to suit their requirements. Using these three classes, design a program that will accept driven of a *TRIANGLE* or *RECTANGLE* interactively and display the area.

Source Code:

```
#include <iostream>
using namespace std;

class Shape{
   public:
   double a,b;
   void getData(){
      cin>>a>>b;
   }
   virtual void display (){};
};

class Triangle:public Shape{
   public:
   void display (){
      cout<<"Area of triangle: "<<0.5*a*b<<endl;
   }
};

class Rectangle:public Shape{
   public:
   void display (){
      cout<<"Area of rectangle: "<<a*b<<endl;
   }
};

int main()
{
   Triangle t;
   cout<<"Enter base and altitude: ";
   t.getData();
   t.display();
```

```
    Rectangle r;
    cout<<"Enter length and breadth: ";
    r.getData();
    r.display();
    return 0;
}
```

Output:

```
Enter base and altitude: 5 10
Area of triangle: 25
Enter length and breadth: 5 10
Area of rectangle: 50
```

EXPERIMENT 19

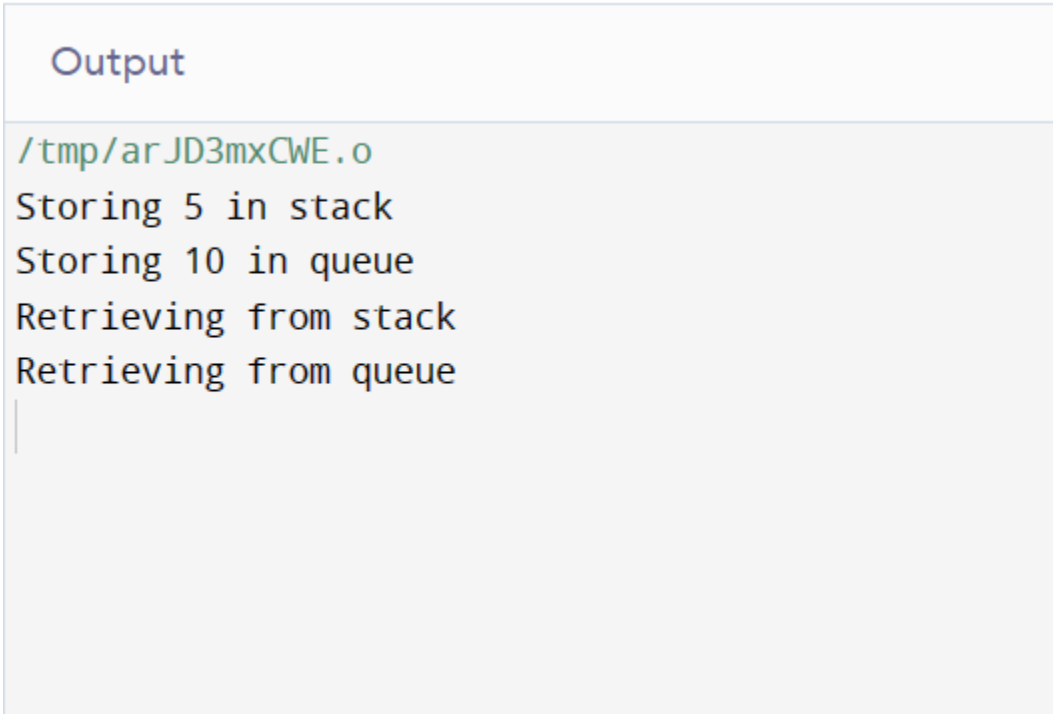
AIM: Create a class called LIST with two pure virtual functions store() and retrieve(), To store a value call store and to retrieve call retrieve function. Derive two classes stack and queue from it and override store and retrieve.

Source code:

```
#include <iostream>
using namespace std;

class List
{
public:
   virtual void store(int value) = 0;
   virtual int retrieve() = 0;
};

class Stack : public List
{
public:
   void store(int value)
   {
      cout << "Storing " << value << " in stack" << endl;
   }

   int retrieve()
   {
      cout << "Retrieving from stack" << endl;
      return 0;
   }
};

class Queue : public List
{
public:
   void store(int value)
   {
      cout << "Storing " << value << " in queue" << endl;
   }

   int retrieve()
   {
      cout << "Retrieving from queue" << endl;
      return 0;
   }
};

int main(){
```

```
    Stack stack;
Queue queue;

stack.store(5);
queue.store(10);
stack.retrieve();
queue.retrieve();

return 0;
}
```

Output:



EXPERIMENT 20

AIM: Write a program to define the function template for swapping item of various data type such as integers, float and characters.

Source code:

```
#include <iostream>
using namespace std;
template <class T>
int swap_numbers(T& x, T& y)
{
   T t;
   t = x;
   x = y;
   y = t;
   return 0;
}
int main()
{
   char ch1 = 'c', ch2 = 'd';
   swap_numbers(ch1, ch2);
   cout <<"character swapping"<< ch1 << " " << ch2 << endl;
   int m=10 ,n=20;
   swap_numbers(m, n);
   cout <<"integer swapping"<< m << " " << n << endl;
   float p=3.5,q=4.5;
   swap_numbers(p, q);
   cout<<"float swapping" << p << " " << q << endl;
   return 0;
}
```

Output:

Output

```
/tmp/arJD3mxCWE.o
character swapping : d c
integer swapping: 20 10
float swapping: 4.5 3.5
```

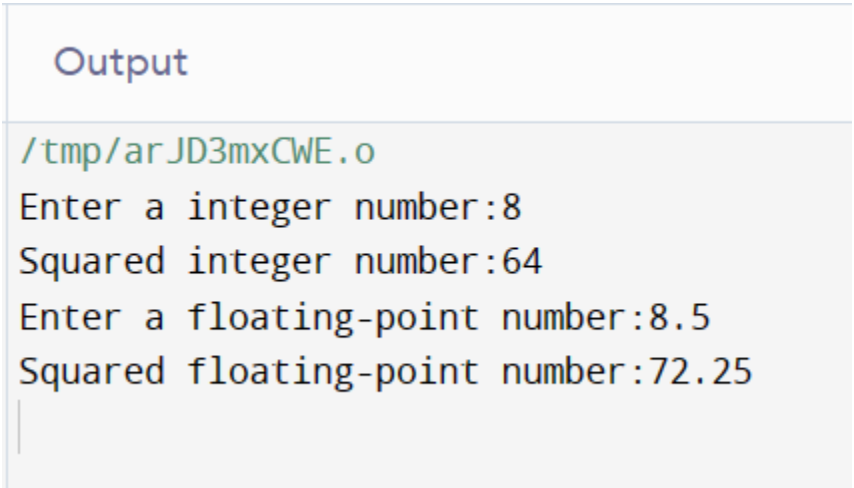
EXPERIMENT 21

Aim: Write a program to define the function template for calculating the square of given numbers with different data types.

Source code:

```
#include<iostream>
using namespace std;
template <class T>
T square(T num)
{
   return num * num;
}
int main()
{
   int int_num;
   float float_num;

   cout << "Enter a integer number:";
   cin >> int_num;
   cout << "Squared integer number:" << square(int_num) << endl;

   cout << "Enter a floating-point number:";
   cin >> float_num;
   cout << "Squared floating-point number:" << square(float_num) << endl;
   return 0;
}
```

Output:



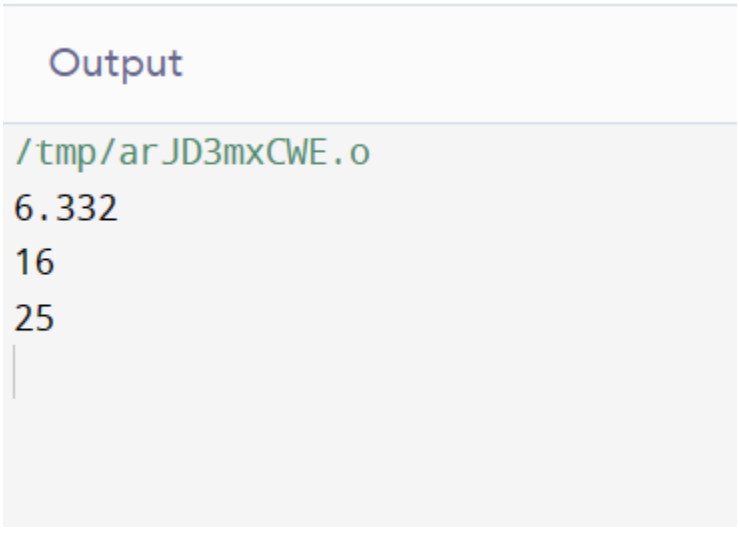
EXPERIMENT 22

AIM: Write a program to illustrate how Template function can be overload.

Source code:

```
#include<iostream>
using namespace std;
template <typename T>
T Add(T num1,T num2)
{
   return num1 + num2;
}
int Add(int num1,int num2,int num3)
{
   return num1+num2+num3;
}
int main()
{
   cout<<Add(3.3,3.032)<<endl;
   cout<<Add(3,3,10)<<endl;
   cout<<Add(15,10)<<endl;
   return 0;

}
```

Output:



EXPERIMENT 23

AIM: Write a program to illustrate how define and declare a class template for reading two data items from the keyboard and two find their sum.

Source code:

```
#include <iostream>
using namespace std;
template <typename t>
class Numberr
{
   t num1;
   t num2;

   public:
      Numberr(t num1, t num2)
      {
         this->num1 = num1;
         this->num2 = num2;
      }
      t add()
      {
         return this->num1 + this->num2;
      }
};
int main()
{
   int n1, n2;
   cout<<"Enter the two numbers: ";

   cin >> n1 >> n2;
   Numberr<int> N(n1, n2);
   int x;
   x = N.add();
   cout<<"The result is: ";
   cout << x;
   return 0;
}
```

Output:

```
Output

/tmp/arJD3mxCWE.o
Enter the two numbers: 4 5
The result is: 9
```

EXPERIMENT 24

AIM: Write a program to demonstrate the use of special functions, constructor and destructor in the class template. The program is used to find the bigger of two entered numbers.

Source code:

```
#include<iostream>
using namespace std;
template<typename t>
class Number
{
   t num1;
   t num2;
   public:
   Number(t num1,t num2)
   {
      this->num1=num1;
      this->num2=num2;

   }
   t getgreater()
   {
      return max(this->num1,this->num2);
   }
   ~Number()
   {
      cout<<"Destructor just called"<<endl;
   }
};
int main()
{
   int num1,num2;
   cout<<"Enter the two numbers: ";
   cin>>num1>>num2;

Number<int> n(num1,num2);

   cout<<"The greater num is: ";
   cout<<n.getgreater()<<endl;

}
```

Output:

```
Output

/tmp/arJD3mxCWE.o
Enter the two numbers: 45 56
The greater num is: 56
Destructor just called
```

EXPERIMENT 25

AIM: Write a program to raise an exception if any attempt is made to refer to an element whose index is beyond the array size.

Source code:

```
#include <iostream>

using namespace std;
int main()
{
int i, a[6] = {1, 2, 3,4, 5,6};
try
{
i = 0;
while (1)
{
if (i != 6)
{
cout << a[i] << endl;
i++;
}
else
{
throw i;
}
}
}
catch (int i)
{
cout << " Array Index out of Bounds Exception: " << i << endl;
}
return 0;
}
```

Output:

Output

```
/tmp/arJD3mxCWE.o
1
2
3
4
5
6
Array Index out of Bounds Exception: 6
```

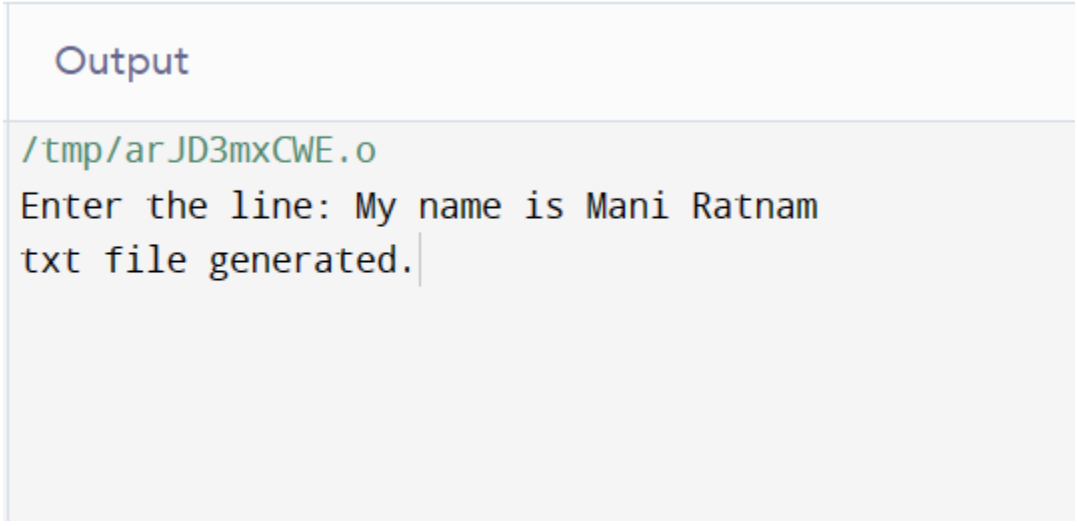
EXPERIMENT 26

AIM: Write a program to read a set of line from the keyboard and to store it on a specified file**.**

Source code:

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main(){
  fstream newfile;
  newfile.open("my.txt",ios::out);
  if(newfile.is_open())
  {
      string ln;
      cout<<"Enter the line: ";
      getline(cin,ln);
    newfile<<ln;
    newfile.close();
  }
   cout<<"txt file generated.";
}
```

Output:



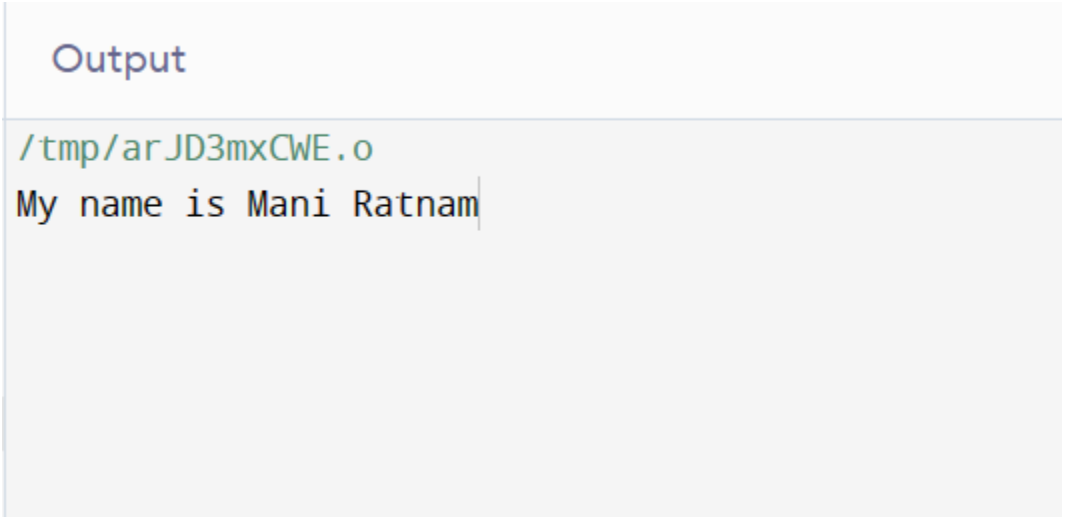
EXPERIMENT 27

AIM: Write a program to read a text file and display its contents on the screen.

Source code:

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main(){
  fstream newfile;
  newfile.open("my.txt",ios::in);
  if (newfile.is_open()){
    string st1;
    while(getline(newfile, st1)){
      cout << st1 << "\n";
    }
    newfile.close();
  }
  return 0;
}
```

Output:



EXPERIMENT 28

AIM: Write a program to copy the contents of a file to another**.**

Source code:

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

int main()
{
   fstream file01;
   fstream file02;

   string st;
   file01.open("my.txt", ios::in);
   file02.open("file2.txt", ios::out);

   while (!file01.eof()) {


      getline(file01, st);
      file02 << st << endl;
   }
   file01.close();
   file02.close();
   cout<<"File copied successfuly and the content is: ";
   file02.open("file2.txt", ios::in);
   while (!file02.eof()) {

      getline(file02, st);
      cout << st << endl;
   }
   file02.close();

   return 0;
}
```

Output:

```
Output

/tmp/arJD3mxCWE.o
File copied successfuly and the content is: My name is Mani Ratnam
```

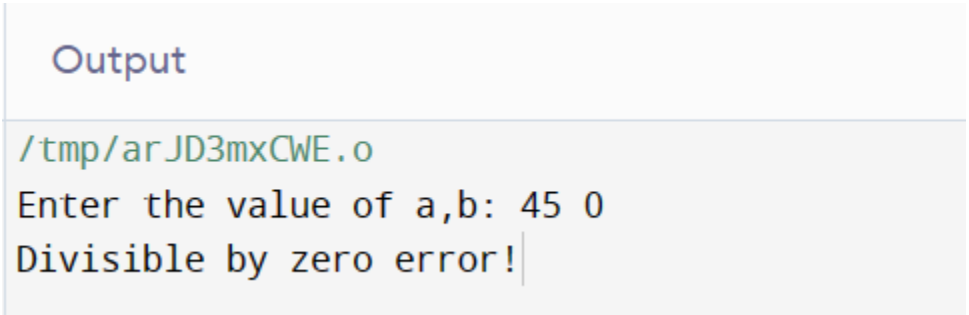
EXPERIMENT 29

AIM: Write a program to read two numbers and then divide the first number by second number and raise an exception if second number is zero.

Source code:

```
#include<iostream>
using namespace std;

int main(){
   int a,b;
   cout<<"Enter the value of a,b: ";
   cin>>a>>b;
   try
   {
      if(b!=0){
         float x=a/b;
         cout<<x;
      }else{
         throw b;
      }
   }
   catch(int i)
   {
      cout<<"Divisible by zero error!";
   }


   return 0;
}
```

Output:



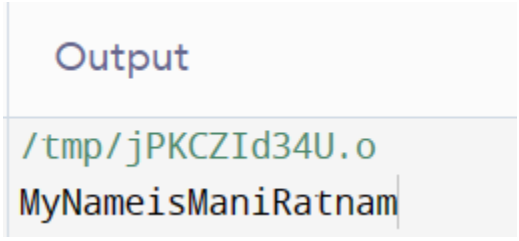
EXPERIMENT 30

AIM: Write a program to perform the deletion of white spaces such as horizontal tab, vertical tab, space, line feed, new line and carriage return from a text file and store the contents of the file without the white spaces on another file.

Source code:

```
#include <iostream>
#include <string>
#include <regex>
#include <fstream>
using namespace std;

int main()
{   fstream newfile;
   string s = "Vivek    \t\v\r\n\n Mishra";

   regex r("\\s+");
   s = regex_replace(s, r, "");
   cout << s;
   newfile.open("m.txt", ios::out);
   if (newfile.is_open())
   {
      string s;
      newfile << s;
      newfile.close();
   }

   return 0;
}
```

Output:



EXPERIMENT 31

AIM: Write a program to read the class object of student info such as name, age, sex, height and weight from the keyboard and to store them on a specified file using read() and write() functions. Again the same file is opened for reading and displaying the contents of the file on the screen.

Source code:

```
#include <iostream>
#include <string>
#include <fstream>
using namespace std;
class Student
{
public:
   string name;
   int age;
   int hight;
   int weight;
   char gender;
   Student()
   {
   }
   Student(string name, int age, int hight, int weight, char gender)
   {
      this->name = name;
      this->age = age;
      this->hight = hight;
      this->weight = weight;
      this->gender = gender;
   }
};
int main()
{
   string name;
   int age, hight, weight;
   char gender;
   cout << "Enter the name: ";
   getline(cin, name);
   cout << "Enter the age: ";
   cin >> age;
   cout << "Enter the hight: ";
```

```
    cin >> hight;
    cout << "Enter the weight: ";
    cin >> weight;
    cout << "Enter the gender: ";
    cin>>gender;
    Student St1(name, age,hight, weight, gender);
    fstream newfile;
    newfile.open("ex31.txt", ios::out);
    if (newfile.is_open())
    {
       newfile << name;
       newfile <<" "<< age;
       newfile <<" "<< hight;
       newfile <<" "<< weight;
       newfile <<" "<< gender;
       newfile.close();
    }
    newfile.open("ex31.txt",ios::in);
    if (newfile.is_open()){
      string st1;
      cout<<"File open: ";
      while(getline(newfile, st1)){
        cout << st1 << "\n";
      }
    }
    newfile.close();

    return 0;
}
```

Output:

```
/tmp/jPKCZId34U.o
Enter the name: Mani Ratnam
Enter the age: 18
Enter the hight: 6
Enter the weight: 55
Enter the gender: M
File open: Mani Ratnam 18 6 55 M
```