M.Sc. (Five Year Integrated) in Computer Science (Artificial Intelligence & Data Science)

First Semester

Laboratory Record

21-805-0107: C++ PROGRAMMING LAB

Submitted in partial fulfillment
of the requirements for the award of degree in
Master of Science (Five Year Integrated)
in Computer Science (Artificial Intelligence & Data Science) of
Cochin University of Science and Technology (CUSAT)
Kochi

Submitted by

OMAL S.
(80521015)

DEPARTMENT OF COMPUTER SCIENCE
COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY (CUSAT)
KOCHI-682022

MARCH 2022

DEPARTMENT OF COMPUTER SCIENCE

COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY (CUSAT)

KOCHI, KERALA-682022

This is to certify that the software laboratory record for ***21-805-0107: C++ Programming Lab*** *is a record of work carried out by* ***OMAL S.(80521015)****, in partial fulfillment of the requirements for the award of degree in* ***Master of Science (Five Year Integrated)*** *in* ***Computer Science (Artificial Intelligence & Data Science)*** *of Cochin University of Science and Technology (CUSAT), Kochi. The lab record has been approved as it satisfies the academic requirements in respect of the first semester laboratory prescribed for the Master of Science (Five Year Integrated) in Computer Science degree.*

Faculty Member in-charge

Dr. Madhu S. Nair
Professor
Department of Computer Science
CUSAT

Dr. Philip Samuel
Professor and Head
Department of Computer Science
CUSAT

Table of Contents

Sl.No.	Program	Pg.No.
1	Program to Calculate Students Grade	1
2	Program to find area using Overloaded Functions	4
3	Program using Classes for Bank Transactions of $'N'$ customers	7
4	Program to perform Operations on String Objects	12
5	Program to demonstrate execution order of Constructors & Destructors	16
6	Program to perform addition on Time Class objects	20
7	Program to perform operations on Matrix Class	22
8	Program to invoke Complex Class objects using constructor overloading	28
9	Program to design and implement Static Member Functions	32
10	Program to process Departmental Store list and perform operations	35
11	Program to Swap Private Data Members of classes using Friend Function	42
12	Program to perform addition on Complex Class Objects	44
13	Program to Overload Comparison operators for a Vector Object	46
14	Program to Overload Operators for Complex Class using Friend function	51
15	Program to overload operators like *,<<,>>for a Vector Object	54
16	Program to Overload '+' and '*' operators for a Matrix Class	56
17	Program to demonstrate Multiple and Multilevel Inheritance	60
18	Program to demonstrate Virtual Base class and Hybrid Inheritance	64
19	Program to show order of execution of Constructors for Multiple Inheritance	68
20	Program to find areas of shapes by Run Time Polymorphism and Abstract Classes	71
21	Program to demonstrate use of Pure Virtual Functions	76
22	Program to demonstrate use of Class Templates	79
23	Program to demonstrate use of Exception Handling	82

STUDENTS GRADE

AIM

To understand the use of class with member functions and calculate the grades of a list of students with attributes (Name, Roll_no, Marks of 3 subjects) with member functions input(), calcGrade(), display().

PROGRAM

```
#include<iostream>
using namespace std;
int n;
class student
{
 char name[20];
 int rollno;
 float m[3];
 public:
    void input();
    void calcGrade();
    void display();
};
void student::calcGrade()
{
 float mark=m[0]+m[1]+m[2];
 int grade=(mark/300)*100;
 cout<<grade<<"\t\t";
 if(grade>=95)
    cout<<"A+\n";
 else if(grade>=90&&grade<95)
    cout<<"A\n";
 else if(grade>=85&&grade<90)
    cout<<"B+\n";
 else if(grade>=80&&grade<85)
    cout<<"B\n";
 else if(grade>=75&&grade<80)
    cout<<"C+\n";
 else if(grade>=70&&grade<75)
    cout<<"c\n";
 else if(grade>=60&&grade<70)
    cout<<"D+\n";
```

```
 else if(grade>=50&&grade<60)
    cout<<"D\n";
 else
     cout<<"F\n";
}
void student:: input()
{cout<<endl;
 cout<<"Roll no.= ";
 cin>>rollno;
 cout<<"Name    = ";
 cin>>name;
  cout<<"Mark of Subject 1 out of 100: ";
  cin>>m[0];
  cout<<"Mark of Subject 2 out of 100: ";
  cin>>m[1];
  cout<<"Mark of Subject 3 out of 100: ";
  cin>>m[2];
}
void student:: display()
{
  cout<<"  "<<rollno<<"\t\t"<<name<<"\t\t "<<m[0]<<"\t\t "//
                       <<m[1]<<"\t\t "<<m[2]<<"\t\t ";
   calcGrade() ;
}
int main()
{
 int n;
 cout<<"Enter the number of Student in the list: ";
 cin>>n;
 student s[n];

 for(int i=0;i<n;i++)
 {s[i].input();}
 cout<<"\n-------------------------------------------------------------"//
                   "----------------------------------------------";
 cout<<"\n\t\tThe Mark List of Students\n";
 cout<<"-------------------------------------------------------------"//
                      "-----------------------------------------\n";
 cout<<"Roll no.\tName\t\tSubject 1\tSubject 2\tSubject 3\tpercentage "//
                        "\tGrade\n";
 for(int i=0;i<n;i++)
```

```
 {s[i].display();}
 cout<<endl;
 return 0;
}
```

SAMPLE INPUT-OUTPUT

```
Enter the number of Student in the list: 10

Roll no.= 1
Name    = Anjali
Mark of Subject 1 out of 100: 98
Mark of Subject 2 out of 100: 85
Mark of Subject 3 out of 100: 90

Roll no.= 2
Name    = Anna
Mark of Subject 1 out of 100: 88
Mark of Subject 2 out of 100: 80
Mark of Subject 3 out of 100: 84

Roll no.= 3
Name    = Bindhu
Mark of Subject 1 out of 100: 78
Mark of Subject 2 out of 100: 77
Mark of Subject 3 out of 100: 75

Roll no.= 4
Name    = Devika
Mark of Subject 1 out of 100: 99
Mark of Subject 2 out of 100: 97
Mark of Subject 3 out of 100: 98

Roll no.= 5
Name    = Hari
Mark of Subject 1 out of 100: 69
Mark of Subject 2 out of 100: 54
Mark of Subject 3 out of 100: 55

Roll no.= 6
Name    = Keerthi
Mark of Subject 1 out of 100: 82
Mark of Subject 2 out of 100: 85
Mark of Subject 3 out of 100: 84

Roll no.= 7
Name    = Manu
Mark of Subject 1 out of 100: 75
Mark of Subject 2 out of 100: 72
Mark of Subject 3 out of 100: 71

Roll no.= 8
Name    = Prasad
Mark of Subject 1 out of 100: 62
Mark of Subject 2 out of 100: 64
Mark of Subject 3 out of 100: 66

Roll no.= 9
Name    = Sunil
Mark of Subject 1 out of 100: 48
Mark of Subject 2 out of 100: 45
Mark of Subject 3 out of 100: 44

Roll no.= 10
Name    = Tharun
Mark of Subject 1 out of 100: 90
Mark of Subject 2 out of 100: 80
Mark of Subject 3 out of 100: 70

-------------------------------------------------------------------------------------------------
                The Mark List of Students
-------------------------------------------------------------------------------------------------
Roll no.        Name            Subject 1       Subject 2       Subject 3       percentage      Grade
  1             Anjali           98              85              90              91             A
  2             Anna             88              80              84              84             B
  3             Bindhu           78              77              75              76             C+
  4             Devika           99              97              98              98             A+
  5             Hari             69              54              55              59             D
  6             Keerthi          82              85              84              83             B
  7             Manu             75              72              71              72             c
  8             Prasad           62              64              66              64             D+
  9             Sunil            48              45              44              45             F
  10            Tharun           90              80              70              80             B
```

OVERLOADING FUNCTIONS TO FIND AREA

AIM

To understand overloaded function and calculate the area of different shapes like Square, Square,Triangle, Circle and Rhombus using function area().

PROGRAM

```
#include<iostream>
#include<cmath>
using namespace std;
float area(int a)
{   float area=1;
    area=a*a;
    return area;    }
float area(int a,int b)
{   float area= a*b;
    return area;    }
float area(float a,float b,float c)
{   float s=(a+b+c)/2;
    float s1=s*(s-a)*(s-b)*(s-c);
    s=sqrt(s1);
    float area=s;
    return area;    }
float area(float a)
{   float area=3.14*a*a;
    return area;    }
float area(float a,float b)
{   float area=(a*b)/2;
    return area;    }
int main()
{   int side1,side2;
    float side3,side4,side5;
    int ch,t;
    cout<<"\n\tAREA OF VARIOUS SHAPES.";
    do
    {   cout<<"\nChoose the shape to find the area.:\n1.Square\n2.Rectangle"//
                "\n3.Triangle\n4.Circle\n5.Rhombus\n6.Exit\n";
        cin>>ch;
        switch(ch)
        {   case 1:  {
```

```
        cout<<"Square: Enter the side: ";
        cin>>side1;
        cout<<"\tThe area of square: "<<area(side1)<<"\n";
        break;
        case 2:
        cout<<"Rectangle: Enter the breadth: ";
        cin>>side1;
        cout<<"\t   Enter the Height : ";
        cin>>side2;
        cout<<"\t   The area of Rectangle: "<<area(side1,side2)<<"\n";
        break;   }
        case 3:  {
        cout<<"Triangle: Enter the Side 1: ";
        cin>>side3;
        cout<<"\t  Enter the Side 2: ";
        cin>>side4;
        cout<<"\t  Enter the Side 3: ";
        cin>>side5;
        cout<<"\t  The area of Triangle: "<<area(side3,side4,side5)<<"\n";
        break;   }
        case 4:  {
        cout<<"Cricle: Enter the Radius: ";
        cin>>side3;
        cout<<"\tThe area of Circle: "<<area(side3)<<"\n";
        break;   }
        case 5:  {
        cout<<"Rhombus: Enter the diagonal 1: ";
        cin>>side4;
        cout<<"\t Enter the diagonal 2: ";
        cin>>side5;
        cout<<"\t The area of Rhombus: "<<area(side4,side5)<<"\n";
        break;   }
        case 6:
            break;
        default:cout<<"Made the wrong choice\n";
    }
    if(ch!=6)
    {   cout<<"\nPlease enter '1' to Continue and '0' to Exit : ";
        cin>>t;
        if(t==0)
            {   ch=6;   }
```

```
        }
    }while(ch!=6);
    cout<<"\tEND\n";
    return 0;
}
```

SAMPLE INPUT-OUTPUT

```
        AREA OF VARIOUS SHAPES.
Choose the shape to find the area.:
1.Square
2.Rectangle
3.Triangle
4.Circle
5.Rhombus
6.Exit
1
Square: Enter the side: 23
        The area of square: 529

Please enter '1' to Continue and '0' to Exit : 1

Choose the shape to find the area.:
1.Square
2.Rectangle
3.Triangle
4.Circle
5.Rhombus
6.Exit
2
Rectangle: Enter the breadth: 12
           Enter the Height : 20
           The area of Rectangle: 240

Please enter '1' to Continue and '0' to Exit : 1

Choose the shape to find the area.:
1.Square
2.Rectangle
3.Triangle
4.Circle
5.Rhombus
6.Exit
3
Triangle: Enter the Side 1: 12
          Enter the Side 2: 10
          Enter the Side 3: 11
          The area of Triangle: 51.5212

Please enter '1' to Continue and '0' to Exit : 1

Choose the shape to find the area.:
1.Square
2.Rectangle
3.Triangle
4.Circle
5.Rhombus
6.Exit
4
Cricle: Enter the Radius: 8
        The area of Circle: 200.96

Please enter '1' to Continue and '0' to Exit : 1

Choose the shape to find the area.:
1.Square
2.Rectangle
3.Triangle
4.Circle
5.Rhombus
6.Exit
5
Rhombus: Enter the diagonal 1: 14
         Enter the diagonal 2: 11
         The area of Rhombus: 77

Please enter '1' to Continue and '0' to Exit : 0
        END
```

BANK TRANSACTION

AIM

To understand the Classes with members and to perform bank transaction for n customers (cust_name, acc_no, acc_type, balance) as menu driven and with menus like adding new account, withdraw (keep a min balance of 500), deposit, balance enquiry and account statement (cust_name, acc_no, acc_type, balance).

PROGRAM

```
#include<iostream>
using namespace std;
int count=0;
class bankacc
{   char cust_name[50];
    char acc_type[30];
    float acc_no;
    float balance=0.0;
    public:
        void new_acc(void);
        void withdraw(void);
        void deposit(void);
        void balance_en(void);
        void acc_statement(void);
};
void bankacc::new_acc()
{   acc_no=count;
    cout<<"Enter the name    : ";
    cin>>cust_name;
    cout<<"Enter account type: ";
    cin>>acc_type;
    cout<<"Enter the balance : ";
    cin>>balance;
    cout<<"\n\tYour Bank Account is created successfully.\n";
    cout<<"Your Account number is : "<<acc_no<<"\nYour current balance is: "//
           <<balance<<endl;
}
void bankacc:: acc_statement()
{   cout<<"\n\tBank Account Details "<<endl;
    cout<<"Account number : "<<acc_no<<endl;
    cout<<"Holder name    : "<<cust_name<<endl;
```

```
    cout<<"Account type   : "<<acc_type <<endl;
    cout<<"Account balance: "<<balance<<endl;
}
void bankacc::balance_en()
{   cout<<"\nYour current Balance in the account number "<<acc_no<<" is "//
                        <<balance<<endl;     }
void bankacc::deposit()
{   float d;
    cout<<"Enter the amount to deposit: Rs.";
    cin>>d;
    balance=balance+d;
    cout<<"\n\tDeposited Successfully\nYour balance is Rs."<<balance<<endl;
}
void bankacc::withdraw()
{   float w,temp;
    cout<<"Account Balance: "<<balance<<endl;
    cout<<"Enter the amount to withdraw: Rs.";
    cin>>w;
    temp=balance-w;
    if(temp>=500)
    {
        balance=temp;
        cout<<"\n\tWithdrawal Successfull\nYour Current balance is Rs."//
                        <<balance<<endl;    }
    else
    {   cout<<"\nYour current balance is not sufficient to withdraw Rs."//
                    <<w<<" with a minimum balance of Rs.500. \n";   }
}
int main()
{   bankacc customer[100];
    int ch,accn,t;
    do
    {   cout<<"\n---------------------------------------------"//
                    "----------------------------------------"<<endl;
        cout<<"\t WELCOME TO THE BANK SERVICES \n";
        cout<<"-----------------------------------------------"//
                        "-------------------------------------"<<endl;
        cout<<"1.New account.\n2.Deposit Amount.\n3.Withdraw Amount.\n4."//
        "Balance enquiry.\n5.Account Statement.\n6.Exit.\n\nEnter your choice: ";
        cin>>ch;
        cout<<"\n*********************************************"//
```

```
                        "****************************************"<<endl;
    if(ch==1)
    {
        customer[count].new_acc();
        count=count+1;
        bankacc customer[count];
    }
    else if(ch>1&&ch<6)
    {
        cout<<"Enter the Account number :";
        cin>>accn;
        if(accn>(count-1))
            cout<<"Account number does not matches.\n";
        else
        {
            cout<<"Account number matches.\n";
            switch(ch)
            {
                case 2:     customer[accn].deposit();
                break;
                case 3:     customer[accn].withdraw();
                break;
                case 4:     customer[accn].balance_en();
                break;
                case 5:     customer[accn].acc_statement();
                break;
            }
        }
    }
    else if(ch==6)
    {   break;  }
    else
    {   cout<<"Your choice is wrong.\n";    }
         cout<<"*******************************************************"//
                                "****************************"<<endl;
            if(ch!=6)
            {   cout<<"\nPlease enter '1' to Continue and '0' to Exit : ";
                cin>>t;
                if(t==0)
                {   ch=6;   }  }
    }   while(ch!=6);
```

```
    cout<<"\tEND\n";
    return 0;
}
```

SAMPLE INPUT-OUTPUT

```
--------------------------------------------------------------------------------
        WELCOME TO THE BANK SERVICES
--------------------------------------------------------------------------------
1.New account.
2.Deposit Amount.
3.Withdraw Amount.
4.Balance enquiry.
5.Account Statement.
6.Exit.

Enter your choice: 1

********************************************************************************
Enter the name    : Manju
Enter account type: Saving
Enter the balance : 0

        Your Bank Account is created successfully.
Your Account number is : 0
Your current balance is: 0
********************************************************************************

Please enter '1' to Continue and '0' to Exit : 1

--------------------------------------------------------------------------------
        WELCOME TO THE BANK SERVICES
--------------------------------------------------------------------------------
1.New account.
2.Deposit Amount.
3.Withdraw Amount.
4.Balance enquiry.
5.Account Statement.
6.Exit.

Enter your choice: 3

********************************************************************************
Enter the Account number :0
Account number matches.
Account Balance: 0
Enter the amount to withdraw: Rs.500

Your current balance is not sufficient to withdraw Rs.500 with a minimum balance of Rs.500.
********************************************************************************

Please enter '1' to Continue and '0' to Exit : 1

--------------------------------------------------------------------------------
        WELCOME TO THE BANK SERVICES
--------------------------------------------------------------------------------
1.New account.
2.Deposit Amount.
3.Withdraw Amount.
4.Balance enquiry.
5.Account Statement.
6.Exit.

Enter your choice: 2

********************************************************************************
Enter the Account number :0
Account number matches.
Enter the amount to deposit: Rs.1000

        Deposited Successfully
Your balance is Rs.1000
********************************************************************************

Please enter '1' to Continue and '0' to Exit : 1

--------------------------------------------------------------------------------
        WELCOME TO THE BANK SERVICES
--------------------------------------------------------------------------------
1.New account.
2.Deposit Amount.
3.Withdraw Amount.
4.Balance enquiry.
5.Account Statement.
6.Exit.

Enter your choice: 3

********************************************************************************
Enter the Account number :0
Account number matches.
Account Balance: 1000
Enter the amount to withdraw: Rs.300

        Withdrawal Successfull
Your Current balance is Rs.700
********************************************************************************

Please enter '1' to Continue and '0' to Exit : 1
```

```
--------------------------------------------------------------------------------
         WELCOME TO THE BANK SERVICES
--------------------------------------------------------------------------------
1.New account.
2.Deposit Amount.
3.Withdraw Amount.
4.Balance enquiry.
5.Account Statement.
6.Exit.

Enter your choice: 1

********************************************************************************
Enter the name    : Leena
Enter account type: Saving
Enter the balance : 400

        Your Bank Account is created successfully.
Your Account number is : 1
Your current balance is: 400
********************************************************************************

Please enter '1' to Continue and '0' to Exit : 1

--------------------------------------------------------------------------------
         WELCOME TO THE BANK SERVICES
--------------------------------------------------------------------------------
1.New account.
2.Deposit Amount.
3.Withdraw Amount.
4.Balance enquiry.
5.Account Statement.
6.Exit.

Enter your choice: 5

********************************************************************************
Enter the Account number :0
Account number matches.

        Bank Account Details
Account number : 0
Holder name    : Manju
Account type   : Saving
Account balance: 700
********************************************************************************

Please enter '1' to Continue and '0' to Exit : 1

--------------------------------------------------------------------------------
         WELCOME TO THE BANK SERVICES
--------------------------------------------------------------------------------
1.New account.
2.Deposit Amount.
3.Withdraw Amount.
4.Balance enquiry.
5.Account Statement.
6.Exit.

Enter your choice: 5

********************************************************************************
Enter the Account number :1
Account number matches.

        Bank Account Details
Account number : 1
Holder name    : Leena
Account type   : Saving
Account balance: 400
********************************************************************************

Please enter '1' to Continue and '0' to Exit : 0
        END
```

STRING OPERATIONS

AIM

To understand the Objects and to perform operations such as compare, concatenate and length on String objects.

PROGRAM

```
#include<iostream>
#include<cstring>
using namespace std;
class String
{
 char *name;
 int length;
 public:
  String()
  {
   length=0;
   name=new char[length+1];
  }
  void input(void);
  void display(void);
  void concatenate(String a,String b);
  void compare( String &t);
  void stlength(void);
 };
 void String::input(void)
 {
  cout<<"Enter String: ";
  cin>>name;
  length=strlen(name);
 }
void String::display(void)
{
 cout<<name;
}
void String::concatenate(String a,String b)
{
 length=a.length+b.length;
 delete name;
```

```
 name=new char[length+1];
 strcpy(name,a.name);
 strcat(name," ");
 strcat(name,b.name);
}
void String::stlength(void)
{
 cout<<"The length: "<<strlen(name)<<"\n";
}
void String::compare( String &s)
{
 int m=strlen(s.name);
 int n=length;
 if(m>n)
 {
    cout<<"\n\t'";
    display();
    cout<<"' Smaller than '";
    s.display();
    cout<<"'\n";        }
 else if(m<n)
 {
    cout<<"\n\t'";
    s.display();
    cout<<"' Smaller than '";
    display();
    cout<<"'\n";        }
 else
    cout<<"\n\tThe two strings are of same size.\n";
}
int main()
{
 String s1, s2,s3;
 int ch,t;
 s1.input();
 s2.input();
 do
 {
 cout<<"\nPlease enter the required action to be performed with inputed "//
"string.\n1.Comparison \n2.Concatenation \n3.Length of the string \n4.Exit\n";
 cin>>ch;
```

```
 cout<<endl;
 switch(ch)
 {
  case 1:
    {   cout<<"The comparison of the String objects";
        s1.compare(s2) ;
        break;
    }
  case 2:
    {   cout<<"The Concatenation of String: \n\t\t";
        s3.concatenate(s1,s2);
        s3.display();
        cout<<"\n";
        break;
    }
  case 3:
    {   cout<<"\nThe Length of the String objects: \n\tString 1 :";
        s1.stlength();
        cout<<"\tString 2 :";
        s2.stlength();
        cout<<"\tString 3 :";
        s3.stlength();
        break;
    }
  case 4:
    {   break;  }
  default:
    {   cout<<"You have made the wrong choice.\n";
        break;
    }
 }
 if(ch!=4)
 {
  cout<<"\nPlease enter '1' to Continue and '0' to Exit.  ";
  cin>>t;
  if(t==0)
        {   ch=4;   }        }
}while(ch!=4);
cout<<"\tEND\n";
return 0;
}
```

SAMPLE INPUT-OUTPUT

```
Enter String: C++
Enter String: Programming_Lab

Please enter the required action to be performed with inputed string.
1.Comparison
2.Concatenation
3.Length of the string
4.Exit
1

The comparison of the String objects
        'C++' Smaller than 'Programming_Lab'

Please enter '1' to Continue and '0' to Exit.  1

Please enter the required action to be performed with inputed string.
1.Comparison
2.Concatenation
3.Length of the string
4.Exit
2

The Concatenation of String:
                C++ Programming_Lab

Please enter '1' to Continue and '0' to Exit.  1

Please enter the required action to be performed with inputed string.
1.Comparison
2.Concatenation
3.Length of the string
4.Exit
3


The Length of the String objects:
        String 1 :The length: 3
        String 2 :The length: 15
        String 3 :The length: 19

Please enter '1' to Continue and '0' to Exit.  1

Please enter the required action to be performed with inputed string.
1.Comparison
2.Concatenation
3.Length of the string
4.Exit
4

        END
```

CONSTRUCTORS AND DESTRUCTORS

AIM

To understand the order of execution of constructors and destructor and find the Sum of Matrices

PROGRAM

```
#include<iostream>
using namespace std;
int count=0;

class matrix
{
 int **m;
 int d1,d2;
 public:
  matrix(int x,int y);
  void input(int &i,int &j,int &value)
   {
        m[i][j]=value;
    }
  int get(int i,int j)
   {
         return(m[i][j]);
    }
  void matrix_add(matrix &, matrix &);

~matrix()
  {
    for(int i=0;i<d1;i++)
    {delete m[i];}
    delete m;
   cout<<"\n******** Object Destroyed : "<<count<<" ********\n"//
                "\tMemory Released\n"<<endl;
   count--;
  }
 };

 matrix::matrix(int x,int y)
 {
```

```
 d1=x;
 d2=y;

 m=new int *[d1];

 for(int i=0;i<d1;i++)
   {
       m[i]=new int [d2];
   }
 cout<<"\n******** Object Created : "<<++count<<" ********\n";
}

void matrix:: matrix_add(matrix &a,matrix &b)
{

 for(int i=0;i<d1;i++)
     { for(int j=0;j<d2;j++)
        {
         m[i][j]=a.m[i][j]+b.m[i][j];
         cout<<m[i][j]<<"  ";
        }
        cout<<endl;
     }
 }

 int main()
 {

 int r1,c1,r2,c2;
    cout<<"\tCONSTRUCOR AND DESTRUCTOR\n";
    cout<<"Enter the Rows and Columns of the 1st matrix: ";
    cin>>r1>>c1;
    cout<<"\n\tMain Constructors invoked\n";
    matrix A(r1,c1);
    cout<<"\nEnter the elements in the matrix row by row.\n";
    int i1,j1,value1;

    for(i1=0;i1<r1;i1++)
     {
       for(j1=0;j1<c1;j1++)
        {
```

```
          cin>>value1;
          A.input(i1,j1,value1);
         }
      }
     cout<<"Enter the Rows and Columns of the 2nd matrix: ";
     cin>>r2>>c2;

     matrix B(r2,c2);

     cout<<"\nEnter the elements in the matrix row by row.\n";
     int i,j,value;

     for(i=0;i<r2;i++)
      { for(j=0;j<c2;j++)
         {
          cin>>value;
          B.input(i,j,value);
         }
      }

    if(r1==r2 && c1==c2)
      {
       cout<<"\nThe constructor inside the block is invoked.\n";
       matrix C1(r1,c1);
       cout<<"\n    Number of objects created : "<<count<<"\n";
       cout<<"--------------------------------------------------"//
                   "-----------------"<<endl;
       cout<<"\tSum of the two matrix: \n";
       C1.matrix_add(A,B);
       cout<<"-------------------------------------------------"//
                   "--------------------"<<endl;
     }
    else
    {
       cout<<"Sum of the two matrices are no possible.\n"//
                   "They are with different dimensions.\n\n";
    }
     cout<<"\tMain Destructor invoked\n";
  return 0;
 }
```

SAMPLE INPUT-OUTPUT

```
        CONSTRUCOR AND DESTRUCTOR
Enter the Rows and Columns of the 1st matrix: 4 4

        Main Constructors invoked

******** Object Created : 1 ********

Enter the elements in the matrix row by row.
1 2 3 4
5 6 7 8
0 1 2 3
4 5 6 7
Enter the Rows and Columns of the 2nd matrix: 4 4

******** Object Created : 2 ********

Enter the elements in the matrix row by row.
7 6 5 4
3 2 1 0
8 7 6 5
4 3 2 1

The constructor inside the block is invoked.

******** Object Created : 3 ********

    Number of objects created : 3
-----------------------------------------------------------------
        Sum of the two matrix:
8  8  8  8
8  8  8  8
8  8  8  8
8  8  8  8
-----------------------------------------------------------------

******** Object Destroyed : 3 ********
        Memory Released

        Main Destructor invoked

******** Object Destroyed : 2 ********
        Memory Released


******** Object Destroyed : 1 ********
        Memory Released
```

TIME CLASS

AIM

To understand the passing objects to the functions and create a class TIME with members hours, minutes, seconds. Take input, add two time objects and display result.

PROGRAM

```
#include<iostream>
using namespace std;
class TIME
{
 int hours=0;
 int minutes=0;
 int seconds=0;
 public:
  void input(void);
  void add(TIME a,TIME b);
  void display(void);
 };
 void TIME::input(void)
 {
  cout<<"Enter hours  : ";
  cin>>hours;
  cout<<"Enter minutes: ";
  cin>>minutes;
  cout<<"Enter seconds: ";
  cin>>seconds;
 }
 void TIME::add(TIME a,TIME b)
 {
  seconds=a.seconds+b.seconds;
  minutes=seconds/60;
  seconds=seconds%60;
  minutes=minutes+a.minutes+b.minutes;
  hours=minutes/60;
  minutes=minutes%60;
  hours=hours+a.hours+b.hours;
 }
 void TIME::display(void)
 {
```

```
 cout<<hours<<" hours : "<<minutes<<" minutes : "<<seconds<<" seconds \n";
}
int main()
{
 TIME T1,T2,T3;
 cout<<"\n\tTIME ADDITION\n";
 cout<<"\tEnter the 1st TIME details.\n";
 T1.input();
 cout<<"\tEnter the 2nd TIME details.\n";
 T2.input();
 T3.add(T1,T2);
 cout<<"\n\tInputed Time details :\n 1st -\t";
 T1.display();
 cout<<" 2nd -\t";
 T2.display();
 cout<<"--------------------------------------------------\n";
 cout<<"Result: ";
 T3.display();
 cout<<endl;
 return 0;
 }
```

SAMPLE INPUT-OUTPUT

```
        TIME ADDITION
        Enter the 1st TIME details.
Enter hours  : 24
Enter minutes: 24
Enter seconds: 24
        Enter the 2nd TIME details.
Enter hours  : 12
Enter minutes: 12
Enter seconds: 12

        Inputed Time details :
 1st -  24 hours : 24 minutes : 24 seconds
 2nd -  12 hours : 12 minutes : 12 seconds
--------------------------------------------------
Result: 36 hours : 36 minutes : 36 seconds
```

MATRIX CLASS AND OPERATIONS

AIM

To understand class with members and implement a class MATRIX with member functions such as matrix_add, matrix_mult, matrix_transpose and matrix_trace

PROGRAM

```
#include<iostream>
using namespace std;
class matrix
{
 int **m;
 int d1,d2;
 public:
   matrix(){};
  matrix(int x,int y);
  void input(int &i,int &j,int valuer=0)
   {
  m[i][j]=valuer;
    }
  int get(int ,int );
  void matrix_add(matrix &, matrix &);
  void matrix_mult(matrix &, matrix &);
  void matrix_transpose(matrix &);
  int matrix_trace(int );
  ~matrix()
  {
    for(int i=0;i<d1;i++)
    {delete m[i];}
    delete m;
  }
 };
  matrix::matrix(int x,int y)
 {
  d1=x;
  d2=y;
  m=new int *[d1];
  for(int i=0;i<d1;i++)
    {m[i]=new int [d2];}
 }
```

```
int matrix::get(int i,int j)
{
 return(m[i][j]);
}
void matrix:: matrix_add(matrix &a,matrix &b)
{
 for(int i=0;i<d1;i++)
     { for(int j=0;j<d2;j++)
        {
         m[i][j]=a.m[i][j]+b.m[i][j];
         cout<<m[i][j]<<"  ";
        }
        cout<<endl;
     }
 }
 void matrix::matrix_mult(matrix &a,matrix &b)
 {

 int i,j,k;
  for(i=0;i<a.d1;i++)
  {
   for(j=0;j<b.d2;j++)
   {
    for(k=0;k<b.d1;k++)
    {
    m[i][j]=m[i][j]+a.m[i][k]*b.m[k][j];
    }
    cout<<m[i][j]<<"   ";
   }
   cout<<endl;
  }
  }
 void matrix::matrix_transpose(matrix &a)
 {
  for(int i=0;i<d1;i++)
  {
   for(int j=0;j<d2;j++)
   {
    int v=a.m[j][i];
     cout<<v<<"  ";
   }
```

```
    cout<<endl;
   }

  }

int matrix::matrix_trace(int n)
{int
sum=0;
 for(int i=0;i<n;i++)
 {
  sum=sum+m[i][i];
  }
return sum;
}

 int main()
    {
     int r1,c1,r2,c2;
     cout<<"\n\tMARTRIX OPERATION\n";
     cout<<"-----------------------------------------------------------------\n";
     cout<<"Enter the Rows and Columns of the 1st matrix: ";
     cin>>r1>>c1;
     matrix A(r1,c1);
     cout<<"Enter the elements in the matrix row by row.\n";
     int i1,j1,value1;
     for(i1=0;i1<r1;i1++)
      { for(j1=0;j1<c1;j1++)
         {
          cin>>value1;
          A.input(i1,j1,value1);
         }
         }
     cout<<"\nEnter the Rows and Columns of the 2nd matrix: ";
     cin>>r2>>c2;
     matrix B(r2,c2);
     cout<<"Enter the elements in the matrix row by row.\n";
     int i,j,value;
     for(i=0;i<r2;i++)
      { for(j=0;j<c2;j++)
         {
          cin>>value;
```

```
         B.input(i,j,value);
        }
        }
    int ch,t;
   matrix C1(r1,c1), C2(r1,c2),C3(c1,r1),C4(c2,r2);
  do{

   cout<<"\n_____________________________________________________________\n";
   cout<<"Choose the operation to be performed with the matrices.\n1.Addition"//
      "\n2.Product of matrices\n3.Transpose of each matrix\n"//
      "4.Trace of each matrix\n5.Exit\n";
  cout<<"_____________________________________________________________\n";
 cin>>ch;
 cout<<endl;
  cout<<"-------------------------------------------------------------\n";
switch(ch)
   {
 case 1:
     if(r1==r2 && c1==c2)
     {
      cout<<"\tSum of the two matrix \n";
      C1.matrix_add(A,B);}
    else{cout<<"Sum of the two matrices are not possible.\n"//
                  "They are with different dimensions.\n\n";
       }
    break;
 case 2:
    if(c1==r2)
     {
      cout<<"\tProduct of the two matrix \n";
      C2.matrix_mult(A,B);}
    else
    {
      cout<<"Product of the matices are not possible to compute.\n";
    }
    break;
 case 3:
     cout<<"\tTranspose of first matrix."<<endl;
     C3.matrix_transpose(A);

     cout<<"\tTranspose of second matrix."<<endl;
```

```
      C4.matrix_transpose(B);
      break;
  case 4:
    if(r1==c1)
     {
     cout<<"\tTrace of 1st matrix is: ";
     int trace=A.matrix_trace(r1);
     cout<<trace<<endl;
     }
     else
     cout<<"Computing Trace of 1st matrix is not possible.\n";
    if(r2==c2)
     {
     cout<<"\tTrace of 2nd matrix is: ";
     int trace=B.matrix_trace(r2);
     cout<<trace<<endl;
     }
     else
     cout<<"Computing Trace of 2nd matrix is not possible.\n";
     break;
  case 5:
      break;
  default:
      cout<<"You made the wrong choice.\n";
      break;
     }
  if(ch!=5)
  {
  cout<<"--------------------------------------------------"//
             "---------------\n";
  cout<<"\nPlease enter '1' to Continue and '0' to Exit.  ";
  cin>>t;
  if(t==0)
        {  ch=5;    }
 }

   }while(ch!=5);
   cout<<"\tEND\n";
      return 0;
 }
```

SAMPLE INPUT-OUTPUT

```
        MARTRIX OPERATION
-----------------------------------------------------------------
Enter the Rows and Columns of the 1st matrix: 4 4
Enter the elements in the matrix row by row.
1 2 3 4
5 6 7 8
0 1 2 3
4 5 6 7

Enter the Rows and Columns of the 2nd matrix: 4 4
Enter the elements in the matrix row by row.
7 6 5 4
3 2 1 0
8 7 6 5
4 3 2 1


_________________________________________________________________
Choose the operation to be performed with the matrices.
1.Addition
2.Product of matrices
3.Transpose of each matrix
4.Trace of each matrix
5.Exit
_________________________________________________________________
1

-----------------------------------------------------------------
        Sum of the two matrix
8  8  8  8
8  8  8  8
8  8  8  8
8  8  8  8
-----------------------------------------------------------------

Please enter '1' to Continue and '0' to Exit.  1


_________________________________________________________________
Choose the operation to be performed with the matrices.
1.Addition
2.Product of matrices
3.Transpose of each matrix
4.Trace of each matrix
5.Exit
_________________________________________________________________
2

-----------------------------------------------------------------
        Product of the two matrix
53  43  33  23
141  115 89  63
31  25  19  13
119  97  75  53
-----------------------------------------------------------------

Please enter '1' to Continue and '0' to Exit.  1


_________________________________________________________________
Choose the operation to be performed with the matrices.
1.Addition
2.Product of matrices
3.Transpose of each matrix
4.Trace of each matrix
5.Exit
_________________________________________________________________
3

-----------------------------------------------------------------
        Transpose of first matrix.
1  5  0  4
2  6  1  5
3  7  2  6
4  8  3  7
        Transpose of second matrix.
7  3  8  4
6  2  7  3
5  1  6  2
4  0  5  1
-----------------------------------------------------------------

Please enter '1' to Continue and '0' to Exit.  1


_________________________________________________________________
Choose the operation to be performed with the matrices.
1.Addition
2.Product of matrices
3.Transpose of each matrix
4.Trace of each matrix
5.Exit
_________________________________________________________________
4

-----------------------------------------------------------------
        Trace of 1st matrix is: 16
        Trace of 2nd matrix is: 16
-----------------------------------------------------------------

Please enter '1' to Continue and '0' to Exit.  0
        END
```

COMPLEX CLASS CONSTRUCTOR OVERLOADING

AIM

To understand the constructor overloading with default constructor and parameterized constructors and calculate the sum of two complex numbers.

PROGRAM

```
#include<iostream>
using namespace std;
class complexno
{
 int real;
 int imag;
 public:
  complexno()
  {real=0;
   imag=0;
  }
  complexno(int& a)
  {
   real=a;
   imag=a;
  }
  complexno(int& a,int& b)
  {
   real=a;
   imag=b;
  }
  void add(complexno&,complexno&);
  void display(void);
 };
 void complexno::add(complexno& a, complexno& b)
 {
  real=a.real+b.real;
  imag=a.imag+b.imag;
 }
 void complexno::display(void)
 {
  cout<<" "<<real;
  if(imag<0)
```

```
   cout<<imag<<"i"<<endl;
  else
   cout<<"+"<<imag<<"i"<<endl;
 }

int main()
 {
  int a,b,c,d,r1,m1,r2,m2,count1=0,count2=0,ch=1;
  cout<<"\n\tADDITION OF COMPLEX NUMBERS.\n\n";
  do{
 cout<<"First Complex number \n\tReal part     :";
 cin>>a;
 cout<<"\tImaginary part:";
 cin>>b;
 if(a==b)
    {r1=a;count1=1;
    }
 else
   {r1=a;
    m1=b;
   }

 cout<<"Second Complex number\n\tReal part     :";
 cin>>c;
 cout<<"\tImaginary part:";
 cin>>d;

 if(c==d)
   {r2=c;count2=1;
    }
 else
   {r2=c;
    m2=d;
    }

  complexno c5;
  cout<<"\nInput numbers\n\t1st-";
  if(count1==1 && count2==1)
    {complexno c1(r1);
     complexno c3(r2);
     c5.add(c1,c3);
```

```
    c1.display();
    cout<<"\t2nd-";
    c3.display();
   }

else if(count1==0 && count2==0)
   {complexno c2(r1,m1);
    complexno c4(r2,m2);
    c5.add(c2,c4);
    c2.display();
    cout<<"\t2nd-";
    c4.display();
    }

else if(count1==0 && count2==1)
   {complexno c2(r1,m1);
    complexno c3(r2);
    c5.add(c2,c3);
    c2.display();
    cout<<"\t2nd-";
    c3.display();
   }

else if(count1==1 && count2==0)
   {complexno c1(r1);
    complexno c4(r2,m2);
    c5.add(c1,c4);
    c1.display();
    cout<<"\t2nd-";
    c4.display();
   }

else
    {cout<<"Something went wrong.\n";}
cout<<"-------------------------------------------------------\n";
cout<<"   Result : ";
c5.display();
 cout<<"-------------------------------------------------------\n";
int t;
cout<<"\nPlease enter '1' to Continue and '0' to Exit : ";
cin>>t;
```

```
if(t==0)
      {ch=0;
       cout<<"\tEND\n";}
cout<<endl;
}while(ch!=0);
return 0;
}
```

SAMPLE INPUT-OUTPUT

```
        ADDITION OF COMPLEX NUMBERS.

First Complex number
        Real part     :23
        Imaginary part:-2
Second Complex number
        Real part     :-20
        Imaginary part:45

Input numbers
        1st- 23-2i
        2nd- -20+45i
-------------------------------------------------------
   Result :  3+43i
-------------------------------------------------------

Please enter '1' to Continue and '0' to Exit : 1

First Complex number
        Real part     :22
        Imaginary part:-1
Second Complex number
        Real part     :22
        Imaginary part:-1

Input numbers
        1st- 22-1i
        2nd- 22-1i
-------------------------------------------------------
   Result :  44-2i
-------------------------------------------------------

Please enter '1' to Continue and '0' to Exit : 0
        END
```

STATIC MEMBER FUNCTION

AIM

To understand the property of Static member function and to design a class having static member function named showcount() which has the property of displaying the number of objects created of the class.

PROGRAM

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

class staticcount
{
 int objnum;
 int num;
 string name;
 static int count;
 public:
  static void showcount()
  {
    cout<<"\tStatic count= "<<count<<endl;
  }
  void input()
  {
  objnum= ++count;
  cout<<"\tIndividual number "<<objnum<<endl;
  cout<<"ID number: ";
  cin>>num;
  cout<<"Name     : ";
  cin>>name;
  }
  void display()
  {
  cout<<"     "<<objnum<<"\t\t "<<num<<"\t\t "<<name<<endl;
  }
 };

 int staticcount :: count;
```

```
int main()
{
  int ch;
  cout<<"\n\tSTATIC MEMBERS AND FUNCTIONS.\n";
  cout<<"\nStatic Count before object creation statement.\n";

 int n;
 cout<<"_____________________________________________"//
              "_______________________\n\t";
    staticcount :: showcount();
 cout<<"_____________________________________________"//
              "_______________________\n";
  cout<<"\nNumber of individuals details to be stored: ";
  cin>>n;
 staticcount obj[n];
 cout<<"Enter the details \n\n";

 for( int i=0;i<n;i++)
 {
  obj[i].input();
 }
 cout<<"\nStatic count after object creation statement.\n";
 cout<<"_____________________________________________"//
                  "_____________________\n";
 staticcount :: showcount();
 cout<<"_____________________________________________"//
                  "___________________\n";
 cout<<"\n\tIndividual list of "<<n<<" objects.\n";
 cout<<"-----------------------------------------------"//
                  "____________________";
 cout<<"\n   Sl.no.\tID Number\tName\n";

  for( int i=0;i<n;i++)
 {
  obj[i].display();
  }
  cout<<endl;

 return 0;
}
```

SAMPLE INPUT-OUTPUT

```
        STATIC MEMBERS AND FUNCTIONS.

Static Count before object creation statement.
----------------------------------------------------------------------
                Static count= 0
----------------------------------------------------------------------

Number of individuals details to be stored: 3
Enter the details

        Individual number 1
ID number: 102310
Name     : Anju
        Individual number 2
ID number: 102311
Name     : Dhurga
        Individual number 3
ID number: 102312
Name     : Shyam

Static count after object creation statement.
----------------------------------------------------------------------
        Static count= 3
----------------------------------------------------------------------

        Individual list of 3 objects.
----------------------------------------------------------------------
   Sl.no.       ID Number       Name
     1           102310          Anju
     2           102311          Dhurga
     3           102312          Shyam
```

DEPARTMENTAL STORE

AIM

To understand the Class and to process shopping list for a Departmental Store. The list include details such as the code_no and price of each item and perform the operations like adding & deleting items to the list and printing the total value of an order.

PROGRAM

```
#include<iostream>
#include<cstring>
using namespace std;
int count=0,n1;
class list;
class deptstore
{
 float price;
 int tempa;
 int tempr;
 public:
 string name;
 int code;
 string note;
 int quantity=0;
 int extra=0;
  float total;
  void input(void);
  void display(int);
  void totalprice(int);
    void deleteitem(void);
  void displaystoke(void);
  friend class list;
};
void deptstore:: input(void)
{
  code=count;
  count++;
  cout<<"\tName\t\t   : ";
  cin>>name;
  cout<<"\tPrice of one item  : ";
  cin>>price;
```

```
  cout<<"\tAvailable quantity : ";
  cin>>quantity;
}
void deptstore:: totalprice(int q)
{total=0;
 tempr=q;
 if(quantity>=q)
    { total=q*price;
      tempa=quantity;
      quantity=quantity-q;
      }
 else if(q>quantity)
    {
       q=q-quantity;
       extra=q;
       tempa=quantity;
       total=quantity*price;
       quantity=0;
    }
 else
    {   cout<<"Stocks not found\n";   }
 }
void deptstore:: displaystoke(void)
{
  cout<<code<<"\t\t\t"<<name<<"\t\t\t"<<quantity<<"\t\t\t"<<price<<endl;

}
void deptstore:: display(int c)
{  if(c==1)
   cout<<" "<<code<<"\t\t"<<note<<"\t"<<name<<"\t\t\t";
  if(c==2)
   {cout<<"\t\t\t"<<tempa<<"\t\t\t   "<<price<<"\t\t\t "<<total<<"\t\t\t ";
   if(extra!=0)
     cout<<extra<<" Out of stoke.";
   cout<<endl;}
}
void deptstore:: deleteitem(void)
  {price=0;
   quantity=0;
   cout<<"\nDeleted quantity set to zero. \n";
   }
```

```
class list
{ public:
   int quantity;
   int code;
   string name;
   void input(void)
   {
      cout<<"Code\t : ";
      cin>>code;
      cout<<"\tQuantity : ";
      cin>>quantity;
    }
   void display(void)
   {
    cout<<quantity;
   }
};

int main()
{
  int n2;
  cout<<"\tDEPARTMENT STORE\n\nNumber Available items : ";
  cin>>n1;
  deptstore stoke[n1+10];
  cout<<"Enter the details of the items\n";
  for(int i=0;i<n1;i++)
  {
     cout<<i+1<<".\t";
     cout<<"Item Code : "<<i<<endl;
     stoke[i].input();
  }
  int ch,i=n1,a,j=0;
 do{
 cout<<"\n\tSERVICES\n";
 cout<<"-------------------------------------------------------------------"//
             "-------------------------------------\n";
 cout<<"1.Purchasing things\n2.Add an item to the store\n3.Delete an item"//
      "\n4.Display Available stoke list\n5.Exit\n";
 cin>>ch;
 switch(ch)
 {
```

```
case 1:
 {
 cout<<"\nPurchasing list \nNumber of items : ";
 cin>>n2;
 list item[n2];
 cout<<"\nEnter the details \n";

 for(int i=0;i<n2;i++)
 {
    cout<<i+1<<".\t";
    item[i].input();
 }
 cout<<"\n\t\tPURCHASE BILL\n";
 cout<<"-------------------------------------------------------"//
    "----------------------------------------------"//
    "-------------------------------------------------------\n";
 int stokecount=0,t=0;
 cout<<"  Code\t\tStatus\t\tName\t\tQuantity Required      "//
         "Quantity Available\t\tPer Price\t\tTotal\n";
 float s=0;
 for(int j=0;j<n2;j++)
 {
   for(int i=0;i<n1;i++)
   {
     if(item[j].code==stoke[i].code)
     {
       int m=item[j].quantity;
        stoke[i].totalprice(m);
        stoke[i].note="available";
        stoke[i].display(1);
         stokecount=0;
        item[j].display();
        stoke[i].display(2);
        s=s+stoke[i].total;
        t=item[j].code;
        break;
        }
    else
        stokecount=1;
    }
    stokecount=0;
```

```
    t=0;
  }
  cout<<"\tThe toal Amount : "<<s<<endl;
  }break;
  case 2:
  { cout<<"Enter the details of the items\n";
    cout<<i<<".\t";
     cout<<"Item Code : "<<i+1<<endl;
     stoke[i].input();
    i++;
     break;
  }
  case 3:
  { cout<<"Enter the item code: ";
              cin>>a;
              for(int k=0;k<i;k++)
              {
              if(stoke[k].code==a)
              {     j=1;
                    stoke[k].deleteitem();
              }  }
              if(j==0)
                cout<<"The item code not match.\n";
              break;
  }
  case 4:     n1=i;
   cout<<"\t\t\t\tAVAILABLE STOCKS\n";
   cout<<"-----------------------------------------------------"//
    "-----------------------------------------------------\n";
 cout<<"Code\t\t\tName\t\t\tQuatity\t\t\tPrice\n";
 for (int i=0;i<n1;i++)
 {stoke[i].displaystoke();
 }break;
  case 5:
      break;
  default: cout<<"Wrong choice\n";  break;
  }int t;
  if(ch!=5)
  {cout<<"\nDo you want to continue (1 - to continue of 0 - to exit) : ";
  cin>>t;
  if (t==0)
```

```
    ch=5;}
  }while(ch!=5);
  cout<<"\n\tEND\n";
  return 0;
  }
```

SAMPLE INPUT-OUTPUT

```
        DEPARTMENT STORE

Number Available items : 3
Enter the details of the items
1.      Item Code : 0
        Name               : Book
        Price of one item  : 50
        Available quantity : 50
2.      Item Code : 1
        Name               : Bag
        Price of one item  : 1000
        Available quantity : 80
3.      Item Code : 2
        Name               : Pen
        Price of one item  : 10
        Available quantity : 100

        SERVICES
-------------------------------------------------------------------------------------------
1.Purchasing things
2.Add an item to the store
3.Delete an item
4.Display Available stoke list
5.Exit
1

Purchasing list
Number of items : 2

Enter the details
1.      Code     : 2
        Quantity : 5
2.      Code     : 0
        Quantity : 2

                PURCHASE BILL
-------------------------------------------------------------------------------------------------------------------------------
  Code          Status          Name            Quantity Required      Quantity Available               Per Price               Total
 2              available       Pen                    5                      100                          10                     50
 0              available       Book                   2                      50                           50                     100
        The toal Amount : 150

Do you want to continue (1 - to continue of 0 - to exit) : 1

        SERVICES
-------------------------------------------------------------------------------------------
1.Purchasing things
2.Add an item to the store
3.Delete an item
4.Display Available stoke list
5.Exit
4
                                AVAILABLE STOCKS
-------------------------------------------------------------------------------------------
Code                    Name                    Quatity                 Price
0                       Book                    48                      50
1                       Bag                     80                      1000
2                       Pen                     95                      10

Do you want to continue (1 - to continue of 0 - to exit) : 1

        SERVICES
-------------------------------------------------------------------------------------------
1.Purchasing things
2.Add an item to the store
3.Delete an item
4.Display Available stoke list
5.Exit
2
Enter the details of the items
3.      Item Code : 4
        Name               : Shoes
        Price of one item  : 2000
        Available quantity : 50

Do you want to continue (1 - to continue of 0 - to exit) : 1
```

```
        SERVICES
-------------------------------------------------------------------------------------------
1.Purchasing things
2.Add an item to the store
3.Delete an item
4.Display Available stoke list
5.Exit
4
                                AVAILABLE STOCKS
-------------------------------------------------------------------------------------------
Code                    Name                    Quatity                 Price
0                       Book                    48                      50
1                       Bag                     80                      1000
2                       Pen                     95                      10
3                       Shoes                   50                      2000

Do you want to continue (1 - to continue of 0 - to exit) : 1

        SERVICES
-------------------------------------------------------------------------------------------
1.Purchasing things
2.Add an item to the store
3.Delete an item
4.Display Available stoke list
5.Exit
1

Purchasing list
Number of items : 3

Enter the details
1.      Code     : 1
        Quantity : 1
2.      Code     : 3
        Quantity : 2
3.      Code     : 2
        Quantity : 5

                PURCHASE BILL
-------------------------------------------------------------------------------------------------------------------------------
  Code          Status          Name            Quantity Required       Quantity Available              Per Price               Total
 1              available       Bag                     1                       80                      1000                    1000
 3              available       Shoes                   2                       50                      2000                    4000
 2              available       Pen                     5                       95                      10                      50
        The toal Amount : 5050

Do you want to continue (1 - to continue of 0 - to exit) : 1

        SERVICES
-------------------------------------------------------------------------------------------
1.Purchasing things
2.Add an item to the store
3.Delete an item
4.Display Available stoke list
5.Exit
4
                                AVAILABLE STOCKS
-------------------------------------------------------------------------------------------
Code                    Name                    Quatity                 Price
0                       Book                    48                      50
1                       Bag                     79                      1000
2                       Pen                     90                      10
3                       Shoes                   48                      2000

Do you want to continue (1 - to continue of 0 - to exit) : 1

        SERVICES
-------------------------------------------------------------------------------------------
1.Purchasing things
2.Add an item to the store
3.Delete an item
4.Display Available stoke list
5.Exit
3
Enter the item code: 1

Deleted quantity set to zero.

Do you want to continue (1 - to continue of 0 - to exit) : 1
```

```
        SERVICES
-------------------------------------------------------------------------------------------
1.Purchasing things
2.Add an item to the store
3.Delete an item
4.Display Available stoke list
5.Exit
4
                                AVAILABLE STOCKS
-------------------------------------------------------------------------------------------
Code                    Name                    Quatity                 Price
0                       Book                    48                      50
1                       Bag                     0                       0
2                       Pen                     90                      10
3                       Shoes                   48                      2000

Do you want to continue (1 - to continue of 0 - to exit) : 0

        END
```

SWAPPING PRIVATE DATA MEMBERS

AIM

To understand the private data members and swap private data members of classes named as class_1, class_2 using friend function.

PROGRAM

```
#include<iostream>
using namespace std;
class class_2;
class class_1
{
 int number1;
 public:
  class_1(int a)
  {
   number1=a;
   }
   void output(void)
   {
    cout<<number1<<endl;
   }
   friend void swap(class_1 &,class_2 &);
 };
 class class_2
 {
  int number2;
  public:
   class_2(int b)
   {
    number2=b;
   }
   void output(void)
   {
    cout<<number2<<endl;
   }
   friend void swap(class_1 &,class_2 &);
 };
 void swap(class_1 &a,class_2 &b)
 {
```

```
  int temp=a.number1;
  a.number1=b.number2;
  b.number2=temp;
 }
 int main()
 {int m,n;
  cout<<"\n\tEnter the values for Swapping between classes.\nClass_1 : ";
  cin>>m;
  class_1 A(m);
  cout<<"Class_2 : ";
  cin>>n;
  class_1 B(n);
  swap(A,B);
  cout<<"\n------------------------------------------------------------"//
                        "--------\n";
  cout<<"\tAfter Swapping the values.\n";
  cout<<"------------------------------------------------------------"//
                        "-------\n";
  cout<<"class_1 : ";
  A.output();
  cout<<"Class_2 : ";
  B.output();
  cout<<endl;
  return 0;
 }
```

SAMPLE INPUT-OUTPUT

```
        Enter the values for Swapping between classes.
Class_1 : 2453
Class_2 : 1000

-----------------------------------------------------------------
        After Swapping the values.
-----------------------------------------------------------------
class_1 : 1000
Class_2 : 2453
```

COMPLEX NUMBER ADDITION BY FRIEND FUNCTION

AIM

To understand the friend function returning an object and to design a class complex to use an external function to add two complex numbers.to design a class complex to represent complex numbers and to use a friend function to add two complex numbers. The function should return an object of type complex representing the sum of two complex numbers.

PROGRAM

```
#include<iostream>
using namespace std;
class complex
{
 int real;
 int imag;
 public:
  complex()
  {real=0;
   imag=0;
  }
  complex(int a,int b)
  {real=a;
   imag=b;
  }
  friend complex add(complex &,complex &);
  void display(void);
 };
 void complex::display(void)
 { cout<<real;
  if(imag<0)
   cout<<imag<<"i"<<endl;
  else
   cout<<"+"<<imag<<"i"<<endl;
 }
 complex add(complex &a, complex &b)
 {
  complex c;
  c.real=a.real+b.real;
  c.imag=a.imag+b.imag;
```

```
 return c;
}
int main()
{cout<<"\n\tCOMPLEX NUMBER ADDITION USING FRIEND FUNCTION\nEnter\n";
 int r1,m1,r2,m2;
 cout<<"\nFirst Complex number \n\tReal part     :";
 cin>>r1;
 cout<<"\tImaginary part:";
 cin>>m1;
 complex c1(r1,m1);
 cout<<"Second Complex number \n\tReal part     :";
cin>>r2;
cout<<"\tImaginary part:";
cin>>m2;
 complex c2(r2,m2);
 complex c3;
 c3=add(c1,c2);
 cout<<"Input numbers\n\t1st-  ";
 c1.display();
 cout<<"\t2nd-  ";
 c2.display();
 cout<<"-----------------------------------------------------"//
                          "------------------\n";
 cout<<"    Result :  ";
 c3.display();
   cout<<"-----------------------------------------------------"//
                          "------------------\n\n";
 return 0;
 }
```

SAMPLE INPUT-OUTPUT

```
        COMPLEX NUMBER ADDITION USING FRIEND FUNCTION
Enter

First Complex number
        Real part     :45
        Imaginary part:-7
Second Complex number
        Real part     :53
        Imaginary part:89
Input numbers
        1st-  45-7i
        2nd-  53+89i
-------------------------------------------------------------------
    Result :  98+82i
-------------------------------------------------------------------
```

OPERATOR OVERLOADED COMPARISON FOR VECTOR CLASS

AIM

To understand the operator overloading as the member function and to overload == , != , <, <=, ,> and >= operators as member operator functions for a vector object.

PROGRAM

```
#include<iostream>
#include<cmath>
using namespace std;
class vector
{
 int v[3];
 float magnitude;
 public:
  vector(){}
  void magnitude_(void);
  int operator ==(vector & );
  int operator !=(vector & );
  int operator <=(vector & );
  int operator >=(vector & );
  int operator >(vector & );
  int operator <(vector & );
  void input(int);
 };
 void vector::magnitude_(void)
  {float m=0;
   for(int i=0;i<3;i++)
     {m=m+pow(v[i],2);}
   magnitude=sqrt(m);
   cout<<magnitude<<endl;
  }
 int vector:: operator ==(vector & a)
 {
  if(magnitude==a.magnitude)
       return 1;
  else
    return 0;
 }
```

```
int vector::operator !=(vector & a)
{
 if(magnitude!=a.magnitude)
    return 1;
 else
    return 0;
}
int vector::operator <=(vector & a)
{
  if (magnitude<=a.magnitude)
    return 1;
  else
    return 0;
 }
int vector::operator >=(vector & a)
{
  if (magnitude>=a.magnitude)
    return 1;
  else
    return 0;
 }
int vector::operator <(vector & a)
{
  if (magnitude<a.magnitude)
    return 1;
  else
    return 0;
 }
int vector::operator >(vector & a)
{
  if (magnitude>a.magnitude)
    return 1;
  else
    return 0;
 }
void vector::input(int a)
{
   cout<<"Enter vector "<<a<<" : ";
 for(int i=0;i<3;i++)
 {
  cin>>v[i];
```

```
 }}
int main()
{
 cout<<"\n\tVECTOR COMPARISON\n";
 int ch;

 vector v1;
 v1.input(1);
 vector v2;
 v2.input(2);

 cout<<"Magnitude for vector 1 : ";
 v1.magnitude_();
 cout<<"Magnitude for vector 2 : ";
 v2.magnitude_();
 cout<<"\n\tCOMPARING THE VECTOR OBJECTS.\n";

 do{
  cout<<"Choose the option operartion to perform \n1.'=='\n2.'!='\n3.'<='"//
                    "\n4.'<'\n5.'>='\n6.'>'\n7.Exit\n";
  cin>>ch;

  cout<<"_____________________________________________________________\n";
  switch(ch)
   {
   case 1:cout<<"vector 1 == vector 2";
  if(v1==v2)
     cout<<"\t TRUE\n";
  else
     cout<<"\t FALSE\n";
        break;
    case 2:cout<<"vector 1 != vector 2";
  if(v1!=v2)
     cout<<"\t TRUE\n";
 else
     cout<<"\t FALSE\n";
         break;
    case 3:cout<<"vector 1 <= vector 2";
  if (v1<=v2)
      cout<<"\t TRUE\n";
  else
```

```
     cout<<"\t FALSE\n";
     break;
    case 4:cout<<"vector 1 < vector 2";
  if(v1<v2)
     cout<<"\t TRUE\n";
 else
     cout<<"\t FALSE\n";
         break;
    case 5:cout<<"vector 1 >= vector 2";
  if(v1>=v2)
   cout<<"\t TRUE\n";
  else
    cout<<"\t FALSE\n";
       break;
   case 6:cout<<"vector 1 > vector 2";
if(v1>v2)
    cout<<"\t TRUE\n";
 else
   cout<<"\t FALSE\n";
        break;
   case 7:
      break;
   default:cout<<"Wrong choice\n";
      break;
   }
  int t;
  if (ch!=7)
  {
  cout<<"____________________________________________________________\n";
  cout<<"\nDo you want to contiue.('1'-contiue or '0'-exit) : ";
   cin>>t;
   if (t==0)
     ch=7;
   }
   cout<<endl;
 }while(ch!=7);
cout<<"\tEND\n";
return 0;
}
```

SAMPLE INPUT-OUTPUT

```
        VECTOR COMPARISON
Enter vector 1 : 2 4 6
Enter vector 2 : 5 6 7
Magnitude for vector 1 : 7.48331
Magnitude for vector 2 : 10.4881

        COMPARING THE VECTOR OBJECTS.
Choose the option operartion to perform
1.'=='
2.'!='
3.'<='
4.'<'
5.'>='
6.'>'
7.Exit
1
____________________________________________________________
vector 1 == vector 2     FALSE
____________________________________________________________

Do you want to contiue.('1'-contiue or '0'-exit) : 1

Choose the option operartion to perform
1.'=='
2.'!='
3.'<='
4.'<'
5.'>='
6.'>'
7.Exit
2
____________________________________________________________
vector 1 != vector 2     TRUE
____________________________________________________________

Do you want to contiue.('1'-contiue or '0'-exit) : 1

Choose the option operartion to perform
1.'=='
2.'!='
3.'<='
4.'<'
5.'>='
6.'>'
7.Exit
3
____________________________________________________________
vector 1 <= vector 2     TRUE
____________________________________________________________

Do you want to contiue.('1'-contiue or '0'-exit) : 1

Choose the option operartion to perform
1.'=='
2.'!='
3.'<='
4.'<'
5.'>='
6.'>'
7.Exit
4
____________________________________________________________
vector 1 < vector 2      TRUE
____________________________________________________________

Do you want to contiue.('1'-contiue or '0'-exit) : 1

Choose the option operartion to perform
1.'=='
2.'!='
3.'<='
4.'<'
5.'>='
6.'>'
7.Exit
5
____________________________________________________________
vector 1 >= vector 2     FALSE
____________________________________________________________

Do you want to contiue.('1'-contiue or '0'-exit) : 1

Choose the option operartion to perform
1.'=='
2.'!='
3.'<='
4.'<'
5.'>='
6.'>'
7.Exit
6
____________________________________________________________
vector 1 > vector 2      FALSE
____________________________________________________________

Do you want to contiue.('1'-contiue or '0'-exit) : 0

        END
```

OPERATOR OVERLOADING FOR A COMPLEX CLASS

AIM

To understand the operator overloading using friend operator functions and to design a class representing complex numbers and having the functionality of performing addition & multiplication of two complex numbers.

PROGRAM

```
#include <iostream>
using namespace std;
class complex
{
 int i;
 int j;
 public:
 complex(){i=0;j=0;}
  complex(int a, int b)
  {i=a;
  j=b;}
  void print(void)
  {cout<<i;
   if(j<0)
     cout<<j<<"i"<<endl;
   else
     cout<<"+"<<j<<"i"<<endl;
  }
  friend complex operator+(complex &,complex &);
  friend complex operator*(complex &,complex &);
  };
 complex operator +(complex & a,complex & b)
 {
  complex c;
  c.i=a.i+b.i;
  c.j=a.j+b.j;
  return c;
  }
  complex operator *(complex & a,complex & b)
  {int p,q,r;
   complex c;
    c.i= (a.i*b.i)+((a.j*b.j)*(-1));
```

```
    c.j=(a.i*b.j)+(a.j*b.i);
   return c;
  }
  int main()
  {int i,j;
   cout<<"\n\tCOMPLEX NUMBER OPERARIONS\n\n";
   cout<<"Enter the first complex number :";
   cin>>i>>j ;
   complex A(i,j);
   cout<<"Enter the second complex number:";
   cin>>i>>j;
   complex B(i,j);
   complex C;
int ch;
cout<<"\nInput Complex number\n\t1st : ";
A.print();
cout<<"\t2nd : ";
B.print();
do{
   cout<<"\nChoice the operation to perform on complex numbers:"//
   "\n1.Addition\n2.multiplication\n3.Exit\n";
   cin>>ch;
   cout<<endl;
   cout<<"----------------------------------------"//
                "------------------------\n";
  switch(ch)
  {
  case 1:  C=A+B;
           cout<<"Sum of complex numbers: ";
           C.print();
           break;
  case 2:  C=A*B;
           cout<<"Product of Complex numbers: ";
           C.print();
         break;
 case 3: break;
  default:
  cout<<"Wrong choice.\n";
   break;
}
 if (ch!=3)
```

```
   {int t;
   cout<<"-------------------------------------"//
                "--------------------------\n";
   cout<<"\nDo you want to continue.'1' to"//
                "comntinue and '0' to exit : ";
   cin>>t;
    if (t==0)
      ch=3;
    }
}while(ch!=3);
cout<<"\tEND\n";
return 0;
}
```

SAMPLE INPUT-OUTPUT

```
        COMPLEX NUMBER OPERARIONS

Enter the first complex number :23 11
Enter the second complex number:-33 9

Input Complex number
        1st : 23+11i
        2nd : -33+9i

Choice the operation to perform on complex numbers:
1.Addition
2.multiplication
3.Exit
1

----------------------------------------------------------------
Sum of complex numbers: -10+20i
----------------------------------------------------------------

Do you want to continue.'1' to comntinue and '0' to exit : 1

Choice the operation to perform on complex numbers:
1.Addition
2.multiplication
3.Exit
2

----------------------------------------------------------------
Product of Complex numbers: -858-156i
----------------------------------------------------------------

Do you want to continue.'1' to comntinue and '0' to exit : 0
        END
```

OPERATOR OVERLOADING FOR VECTOR CLASS

AIM

To understand the operator overloading of *,>> and << for vector classes using friend function.

PROGRAM

```
#include <iostream>
using namespace std;
class vector
{
 int *v;
 public:
   vector()
   {
      v = new int[3];
      for(int i=0;i<3;i++)
          {v[i]=0;}
   }
   friend float operator*(vector &,vector &);
   friend void operator<<(int ,vector &);
   friend void operator>>(int , vector &);
};
void operator<<(int i,vector & a)
   { cout<<a.v[i]<<"i";
   if(a.v[1]<0)
     cout<<a.v[1]<<"j";
   else
     cout<<"+"<<a.v[1]<<"j";
   if(a.v[2]<0)
     cout<<a.v[2]<<"k";
   else
     cout<<"+"<<a.v[2]<<"k";
   }
void operator>>(int i,vector & a)
  {
   for(;i<3;i++)
   {     cin>>a.v[i];       }
  }
float operator*(vector & a,vector & b)
```

```
  { vector V;
    float dp=0;
    for(int i=0;i<3;i++)
    {
     dp=dp+(a.v[i]*b.v[i]);
     }
     return dp;
    }
int main()
{  int input=0,output=0,t=1;
   cout<<"\n\t 3D VECTOR MULTIPLICATION\n";
   vector A,B;
   float C;
   cout<<"\nEnter the first  vector: ";
   input>>A;
   cout<<"Enter the Second Vector: ";
   input>>B;
   cout<<"\nDot Product of ";
   C=A*B;
   cout<<"\tVector 1: ";
   output<<A;
   cout<<"\t and\tVector 2: ";
   output<<B;
   cout<<"\n-------------------------------------------"//
                   "----------------------";
   cout<<"\n\tProduct :";
   cout<<C;
   cout<<"\n---------------------------------------------"//
                   "---------------------\n";
   cout<<endl;
   return 0;
 }
```

SAMPLE INPUT-OUTPUT

```
         3D VECTOR MULTIPLICATION

Enter the first  vector: 4 5 6
Enter the Second Vector: 5 8 6

Dot Product of  Vector 1: 4i+5j+6k       and    Vector 2: 5i+8j+6k
-------------------------------------------------------------------
        Product :96
-------------------------------------------------------------------
```

OPERATOR OVERLOADING FOR MATRIX CLASS

AIM

C++ program for developing a matrix class which can handle integer matrices of different dimensions. Operators Overloaded for addition and multiplication of matrices. Double pointers used program to dynamically allocate memory for the matrices.

PROGRAM

```
#include<iostream>
using namespace std;
class matrix
{
 int **m;
 int d1,d2;
 public:
 matrix(){}
  matrix(int x,int y);
  void input(int &i,int &j,int &value)
   {       m[i][j]=value;      }
  int get(int ,int );
  void operator+(matrix &);
  void operator*(matrix &);
 };
  matrix::matrix(int x,int y)
 {
  d1=x;
  d2=y;
  m=new int *[d1];
  for(int i=0;i<d1;i++)
    {m[i]=new int [d2];}
 }
 int matrix::get(int i,int j)
 {          return(m[i][j]);         }
 void matrix:: operator+(matrix &a)
 {
 int s;
  for(int i=0;i<d1;i++)
      { for(int j=0;j<d2;j++)
         {
          s=m[i][j]+a.m[i][j];
```

```
          cout<<s<<" ";
         }
         cout<<endl;
      }
  }
  void matrix::operator*(matrix &a)
  {
  matrix c(d1,a.d2);
  int i,j,k;
   for(i=0;i<d1;i++)
   {
    for(j=0;j<a.d2;j++)
    {
     for(k=0;k<a.d1;k++)
     {  c.m[i][j]=c.m[i][j]+m[i][k]*a.m[k][j];      }
     cout<<c.m[i][j]<<"  ";
    }
    cout<<endl;
   }
   }
int main()
    {cout<<"\n\tMATIRX OPERATIONS\n";
     int r1,c1,r2,c2;
     cout<<"Enter the Rows and Columns of the 1st matrix: ";
     cin>>r1>>c1;
     matrix A(r1,c1);
     cout<<"Enter the elements in the matrix row by row.\n";
     int i1,j1,value1;
     for(i1=0;i1<r1;i1++)
      { for(j1=0;j1<c1;j1++)
         {
          cin>>value1;
          A.input(i1,j1,value1);
         }
         }
     cout<<"Enter the Rows and Columns of the 1st matrix: ";
     cin>>r2>>c2;
     matrix B(r2,c2);
     cout<<"Enter the elements in the matrix row by row.\n";
     int i,j,value;
     for(i=0;i<r2;i++)
```

```
    { for(j=0;j<c2;j++)
       {
        cin>>value;
        B.input(i,j,value);
       }
       }
   int ch;
   cout<<"\n\tMATRIX OPERATION\n";
  do{
   cout<<"Choose the operation to be performed with the matrices.\n1."//
                  "Addition\n2.Multiplication\n3.Exit\n";
   cin>>ch;
   cout<<"-------------------------------------------------------"//
                  "------------\n";
  switch(ch)
   {case 1:
     if(r1==r2 && c1==c2)
     {matrix C1(r1,c1);
      cout<<"Sum of the two matrix: \n";
      A+B;
      }
    else{cout<<"Computing Sum of the two matrices are no possible.\n"//
                  "They are with different dimensions.\n";}
    break;
   case 2:
    if(c1==r2)
     {matrix C2(r1,c2);
      cout<<"Product of the two matrix: \n";
      A*B;
      }
    else{cout<<"Computing Product of the matrices is not possible.\n";}
    break;
   case 3:
     break;
   default:
     cout<<"Wrong choice.\n";
     break;
    }
    int t;
    if(ch!=3)
     { cout<<"----------------------------------------------------"//
```

```
                                "-----------------\n";
      cout<<"\nDo you want to continue ('1'-continue or '0'-exit).\t";
       cin>>t;
       if(t==0)
          ch=3;}
   }while(ch!=3);
   cout<<"\tEND\n";
   return 0;
}
```

SAMPLE INPUT-OUTPUT

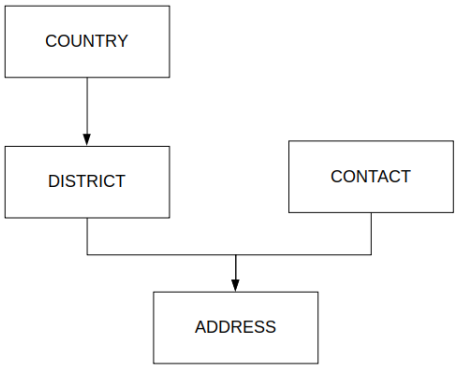
```
        MATIRX OPERATIONS
Enter the Rows and Columns of the 1st matrix: 4 4
Enter the elements in the matrix row by row.
4 5 6 3
4 5 6 3
4 5 6 3
4 5 6 3
Enter the Rows and Columns of the 1st matrix: 4 4
Enter the elements in the matrix row by row.
3 6 5 4
3 6 5 4
3 6 5 4
3 6 5 4

        MATRIX OPERATION
Choose the operation to be performed with the matrices.
1.Addition
2.Multiplication
3.Exit
1
-------------------------------------------------------------------
Sum of the two matrix:
7 11 11 7
7 11 11 7
7 11 11 7
7 11 11 7
-------------------------------------------------------------------

Do you want to continue ('1'-continue or '0'-exit).     1
Choose the operation to be performed with the matrices.
1.Addition
2.Multiplication
3.Exit
2
-------------------------------------------------------------------
Product of the two matrix:
54  108  90  72
54  108  90  72
54  108  90  72
54  108  90  72
-------------------------------------------------------------------

Do you want to continue ('1'-continue or '0'-exit).     0
        END
```

MULTIPLE AND MULTILEVEL INHERITANCE

AIM

To demonstrate the concept of Multiple and Multilevel inheritance including constructors with parameters.

INHERITANCE FLOWCHART



PROGRAM

```
#include<iostream>
#include<cstring>
#include<iomanip>
using namespace std;
class country
{
 string cname;
 string sname;
 public:
 country(){}
 country(string t1,string t2)
 {
   sname=t1;
   cname=t2;
 }
  void getc(void)
  {
     cout<<sname<<"\n"<<setw(25)<<right<<cname<<"\n"<<setw(25)<<right;
  }
};
class district:public country
```

```
{
 string dname;
 string poname;
 public:
  district(){}
  district(string t1,string t2,string t3,string t4):country(t3,t4)
  {
  dname=t2;
  poname=t1;
  }
  void getd(void)
  {
    cout<<poname<<"\n"<<setw(25)<<right<<dname<<"\n"<<setw(25)<<right;
  }
};
class contact
{
 long int n;
 public:
   contact(){}
   contact(long int t)
   {
       n=t;
   }
  void getcon(void)
  {
       cout<<n<<"\n";
  }
};
class address:public district , public contact
{
 string name;
 string hname;
 string pname;
 public:
 address(){}
  address(string t1,string t2,string t3,string t4,string t5, string t6,//
        string t7,long int num):contact(num),district(t4,t5,t6,t7)
  {
   name=t1;
   hname=t2;
```

```
   pname=t3;
  }
  void print(void)
  {
    cout<<name<<"\t\t";
   cout<<hname<<"\n"<<setw(25)<<right<<pname<<"\n"<<setw(25)<<right;
   getd();
   getc();
   getcon();
 }
};
int main()
{
 int n;
 string s1,s2,s3,s4,s5,s6,s7;
 long int num;
 cout<<"\nNumber of Address Details to be stored: ";
 cin>>n;
 address a[n];
 for(int i=0;i<n;i++)
 {
 cout<<"\nName        : ";
 cin>>s1;
 cout<<"House Name  : ";
 cin>>s2;
 cout<<"Place       : ";
 cin>>s3;
 cout<<"Post Office : ";
 cin>>s4;
 cout<<"District    : ";
 cin>>s5;
 cout<<"State       : ";
 cin>>s6;
 cout<<"Country     : ";
 cin>>s7;
 cout<<"Phone Number: ";
 cin>>num;
 a[i]=address(s1,s2,s3,s4,s5,s6,s7,num);
 }
 cout<<endl;
 cout<<"_________________________________________\n";
```

```
cout<<"\tDATA STORED\n";
cout<<"_________________________________________\n";
cout<<"Name\t\tAddress"<<endl;
 cout<<"----------------------------------------\n";
for(int i=0;i<n;i++)
{
   a[i].print();
}
cout<<endl;
return 0;
}
```

SAMPLE INPUT-OUTPUT

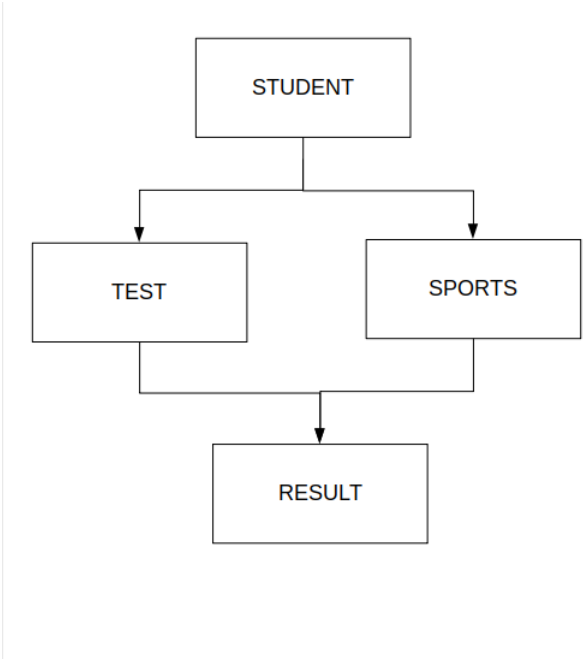
```
Number of Address Details to be stored: 2

Name        : Manju
House Name  : Lakshmi_Bhavan
Place       : Pattanakkad
Post Office : Pattanakkad
District    : Alappuzha
State       : Kerala
Country     : India
Phone Number: 8675463291

Name        : Manoj
House Name  : House_No:24
Place       : Edapally
Post Office : Kochi
District    : Ernakulam
State       : Kerala
Country     : India
Phone Number: 9456723461


_________________________________________
        DATA STORED
_________________________________________
Name            Address
----------------------------------------
Manju           Lakshmi_Bhavan
              Pattanakkad
              Pattanakkad
                Alappuzha
                   Kerala
                    India
               8675463291
Manoj           House_No:24
                 Edapally
                    Kochi
                Ernakulam
                   Kerala
                    India
               9456723461
```

VIRTUAL BASE CLASS

AIM

To understand the concept of Virtual base class on Hybrid inheritance and to design a student class representing student roll no. and a test class (derived class of student) representing the scores of the student in various subjects and sports class representing the score in sports. The sports and test class should be inherited by a result class having the functionality to add the scores and display the final result for a student.

INHERITANCE FLOWCHART



PROGRAM

```
#include<iostream>
#include<iomanip>
using namespace std;
int n;
class student
{
 protected:
   int roll_number;
 public:
  void inputr(int a)
  {
     roll_number=a;
  }
```

```
};
 class test:virtual public student
 {
  protected:
    float sub[3];
  public:
      int i=0;
      void input(float m)
       {
         sub[i]=m;
         i++;
       }
 };
class sports: virtual public student
{
  protected:
   float score;
  public:
    void inputs(int a)
    {
       score=a;
    }
 };
 class result:public test,public sports
 {
   float t=0;
   public:
    void total(void)
    {
       for (int i=0;i<3;i++)
     {
        t=t+sub[i];
     }
      t=t+score;
      cout<<t<<endl;
    }
    void display(void)
     {
     cout<<"  "<<roll_number<<setw(20)<<" ";
     for(int j=0;j<3;j++)
     {
```

```
        cout<<sub[j]<<setw(20)<<" ";
     }
     cout<<score<<setw(20)<<" ";total();}
};
int main()
{
 cout<<"\tHYBRID INHERITANCE\n";
 cout<<"Number of Students: ";
  cin>>n;
  int a;
  float b,c;
  result s[n];
  for(int i=0;i<n;i++)
  {
   cout<<"\nEnter the details:\n";
   cout<<"Roll Number: ";
   cin>>a;
   s[i].inputr(a);cout<<"\tSubject Marks "<<setw(25)<<right<<"Out of 100\n";
   for(int j=0;j<3;j++)
   {
       cout<<"subject "<<j+1<<": ";
       cin>>b;
       s[i].input(b);
   }
   cout<<"\tSports score  "<<setw(25)<<right<<"Out of 10\n"<<"Score: ";
   cin>>c;
   s[i].inputs(c);
   }
   cout<<"\n\t\tSTUDENTS MARKLIST\n";
   cout<<"-------------------------------------------------------------------"//
        "-------------------------------------------------------------\n";
   cout<<"Roll Number"<<setw(20)<<"Subject 1"<<setw(20)<<"Subject 2"<<setw(20)//
   <<"Subject 3"<<setw(25)<<"Sports Score"<<setw(30)<<"total out of 310\n";
   for(int k=0;k<n;k++)
   {
       s[k].display();
   }
   cout<<endl;
   return 0;
   }
```

SAMPLE INPUT-OUTPUT

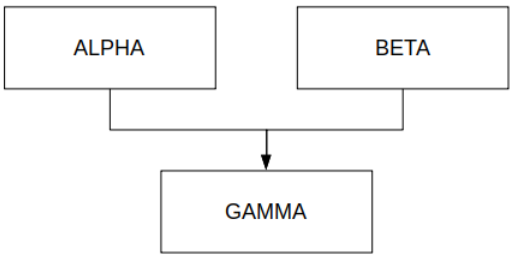
```
        HYBRID INHERITANCE
Number of Students: 5

Enter the details:
Roll Number: 8023
        Subject Marks               Out of 100
subject 1: 98
subject 2: 95
subject 3: 96
        Sports score                 Out of 10
Score: 8

Enter the details:
Roll Number: 8024
        Subject Marks               Out of 100
subject 1: 88
subject 2: 87
subject 3: 83
        Sports score                 Out of 10
Score: 9

Enter the details:
Roll Number: 8025
        Subject Marks               Out of 100
subject 1: 90
subject 2: 91
subject 3: 92
        Sports score                 Out of 10
Score: 8.5

Enter the details:
Roll Number: 8026
        Subject Marks               Out of 100
subject 1: 88
subject 2: 85
subject 3: 82
        Sports score                 Out of 10
Score: 7

Enter the details:
Roll Number: 8027
        Subject Marks               Out of 100
subject 1: 80
subject 2: 75
subject 3: 81
        Sports score                 Out of 10
Score: 8

                STUDENTS MARKLIST
-------------------------------------------------------------------------------------------------------------------------
Roll Number          Subject 1          Subject 2          Subject 3            Sports Score            total out of 310
  8023                  98                 95                 96                   8                       297
  8024                  88                 87                 83                   9                       267
  8025                  90                 91                 92                   8                       281
  8026                  88                 85                 82                   7                       262
  8027                  80                 75                 81                   8                       244
```

CONSTRUCTORS DURING INHERITANCE

AIM

Program to illustrating how the constructors are implemented and the order in which they are called when the classes are inherited. Use three classes named alpha, beta and gamma such that alpha and beta are base classes and gamma is a derived class inheriting alpha and beta.

INHERITANCE FLOWCHART



PROGRAM

```
#include<iostream>
#include<iomanip>
using namespace std;
class alpha
{
 int a;
 public:
  alpha(int &at)
  {
   a= at;
   cout<<setw(50)<<right<<"Alpha base class \n";
  }
   void displayalpha(void)
   {
      cout<<a;
   }
};
class beta
{
 int b;
 public:
  beta(int bt):b(bt)
```

```
  {
     cout<<setw(50)<<right<<"Beta base class \n";
  }
   void displaybeta(void)
   {
       cout<<b;
   }
};
class gamma:public beta,public alpha
{
 int g;
 public:
   gamma(int a1,int b1,int gt):beta(b1),alpha(a1)
   {
    g=gt;
    cout<<setw(50)<<right<<"Gamma derived class \n";
   }
   void displaygamma(void)
   {
        cout<<g<<endl;
   }
 };
int main()
{
 int a,b,g,ch=1;
 cout<<"\tRADIATION VALUES\n";
 do{
 cout<<"\nNumber of particle emitted\n\tAlpha : ";
 cin>>a;
 cout<<"\tBeta  : ";
 cin>>b;
 cout<<"\tGamma : ";
 cin>>g;
 gamma R(a,b,g);
 cout<<"--------------------------------------------------"//
                 "------------------\n";
 cout<<"Entered values\n\t A: ";
 R.displayalpha();
 cout<<"  B: ";
 R.displaybeta();
 cout<<"  G: ";
```

```
R.displaygamma();
cout<<"-----------------------------------------------------"//
                  "-----------------\n";
cout<<"\nDo you want to continue ('1'-continue or '0'-exit) : ";
cin>>ch;
}while(ch!=0);
return 0;
}
```

SAMPLE INPUT-OUTPUT

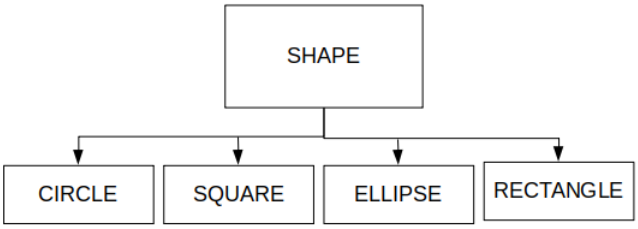
```
        RADIATION VALUES

Number of particle emitted
        Alpha : 1
        Beta  : 2
        Gamma : 0
                                Beta base class
                               Alpha base class
                             Gamma derived class
-----------------------------------------------------------------
Entered values
         A: 1  B: 2  G: 0
-----------------------------------------------------------------

Do you want to continue ('1'-continue or '0'-exit) : 1

Number of particle emitted
        Alpha : 2
        Beta  : 1
        Gamma : 1
                                Beta base class
                               Alpha base class
                             Gamma derived class
-----------------------------------------------------------------
Entered values
         A: 2  B: 1  G: 1
-----------------------------------------------------------------

Do you want to continue ('1'-continue or '0'-exit) : 0
```

RUN TIME POLYMORPHISM

AIM

Program to define classes Shapes, Circle, Square, Ellipse and Rectangle with member functions to get the values for finding corresponding areas and print the same. Utilize the concept of Abstract Class and Run time polymorphism to solve the problem.

INHERITANCE FLOWCHART



PROGRAM

```
#include<iostream>
using namespace std;
class shape
{
 protected:
  float side1;
  float side2;
  float area;
  public:
   virtual void calculatearea(void){};
   virtual void print(void){};
};
class circle:public shape
{
public:
  void input(float r)
  {
   side1=r;
   side2=0;
  }
  void calculatearea(void)
  {
     area=3.14*side1*side1;
```

```
  }
 void print(void)
  {
      cout<<"\tArea : "<<area<<endl;
  }
};
class square:public shape
{
public:
  void input(float s)
  {side1=s;
  side2=0;
  }
  void calculatearea(void)
  {
      area=side1*side1;
  }
  void print (void)
  {
        cout<<"\tArea : "<<area<<endl;
  }
};
class ellipse:public shape
{
public:
 void input(float a,float b)
  {
    side1=a;
    side2=b;
  }
  void calculatearea(void)
  {     area=3.14*side1*side2;   }
  void print (void)
  { cout<<"\tArea : "<<area<<endl;  }
};
class rectangle:public shape
{public:
  void input(float a,float b)
  {
    side1=a;
    side2=b;
```

```
  }
  void calculatearea(void)
  {     area=side1*side2;        }
  void print (void)
  {  cout<<"\tArea : "<<area<<endl;    }
};
int main()
{ cout<<"\n\tAREA OF FIGURES\n";
  float r,s;
  int ch;
   shape* shapes[4];
  do{
   cout<<"\nSelect the shape.\n1.Circle\n2.Square\n3.Ellipse\n4.Rectangle"//
                    "\n5.Exit\n";
   cin>>ch;
   cout<<"-------------------------------------------------------\n";
   switch(ch)
   {
     case 1:{
       cout<<"\tCIRCLE \n";
       cout<<"_______________________________________________________\n";
       cout<<"Enter Radius: ";
       cin>>r;
       circle C;
       C.input(r);
       shapes[0]=&C;
       shapes[0]->calculatearea();
       shapes[0]->print();
       break;           }
    case 2:{
       cout<<"\tSQUARE \n";
       cout<<"_______________________________________________________\n";
       cout<<"Enter Side: ";
       cin>>r;
       square S;
       S.input(r);
       shapes[1]=&S;
       shapes[1]->calculatearea();
       shapes[1]->print();
       break;       }
   case 3:{
```

```
        cout<<"\tELLIPSE \n";
        cout<<"_____________________________________________________\n";
        cout<<"Enter Side 1: ";
        cin>>r;
        cout<<"Enter Side 2: ";
        cin>>s;
        ellipse E;
        E.input(r,s);
        shapes[2]=&E;
        shapes[2]->calculatearea();
        shapes[2]->print();
        break;          }
    case 4:{
        cout<<"\tRECTANGLE \n";
        cout<<"_____________________________________________________\n";
        cout<<"Enter the length: ";
        cin>>r;
        cout<<"Enter the width : ";
        cin>>s;
        rectangle R;
        R.input(r,s);
        shapes[3]=&R;
        shapes[3]->calculatearea();
        shapes[3]->print();
        break;          }
   case 5:    break;
   default:     cout<<"Wrong choice.\n";
   }
   if(ch!=5)
   {
     int t;
     cout<<"-----------------------------------------------------\n";
     cout<<"\nDo you continue-1 or exit-0: ";
     cin>>t;
     if(t==0)
        ch=5;
    }
   }while(ch!=5);
cout<<"\tEND\n";
return 0;
}
```

SAMPLE INPUT-OUTPUT

```
        AREA OF FIGURES

Select the shape.
1.Circle
2.Square
3.Ellipse
4.Rectangle
5.Exit
1
-------------------------------------------------------
        CIRCLE
_______________________________________________________
Enter Radius: 8
        Area : 200.96
-------------------------------------------------------

Do you continue-1 or exit-0: 1

Select the shape.
1.Circle
2.Square
3.Ellipse
4.Rectangle
5.Exit
2
-------------------------------------------------------
        SQUARE
_______________________________________________________
Enter Side: 12
        Area : 144
-------------------------------------------------------

Do you continue-1 or exit-0: 1

Select the shape.
1.Circle
2.Square
3.Ellipse
4.Rectangle
5.Exit
3
-------------------------------------------------------
        ELLIPSE
_______________________________________________________
Enter Side 1: 11
Enter Side 2: 13
        Area : 449.02
-------------------------------------------------------

Do you continue-1 or exit-0: 1

Select the shape.
1.Circle
2.Square
3.Ellipse
4.Rectangle
5.Exit
4
-------------------------------------------------------
        RECTANGLE
_______________________________________________________
Enter the length: 8
Enter the width : 12
        Area : 96
-------------------------------------------------------

Do you continue-1 or exit-0: 0
        END
```

PURE VIRTUAL FUNCTIONS

AIM

To demonstrate the use of pure virtual functions and abstract base classes and find the Annual Salary of Employees

PROGRAM

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

class Employee
{
  protected:
    string name;
    int idno;
    float salary;
  public:
   virtual void input(void)=0;
   virtual void display(void)=0;
};

class developer:public Employee
{
 int yos;
 public:
  void input(void)
  {
  cout<<"Name           :";
  cin>>name;
  cout<<"ID Number      :";
  cin>>idno;
  cout<<"Monthly salary :";
  cin>>salary;
  cout<<"Service Years  :";
  cin>>yos;
  }
  void display(void)
  {
```

```
  cout<<name<<setw(18)<<right<<idno<<setw(19)<<right<<salary<<setw(20)//
                      <<right<<salary*12;
  cout<<setw(20)<<right<<yos<<endl;
  }
 };
int main()
 {
   cout<<"\nPURE VIRTUAL FUNCTION AND ABSTRACT BASE CLASSES\n";
   int n;
   cout<<"\nEnter number of employees: ";
   cin>>n;
   developer d[n];
   Employee* e[n];
   for(int i=0;i<n;i++)
   {
       cout<<"\nEmployee "<<i+1<<endl;
       e[i]=&d[i];
       e[i]->input();
   }
   cout<<"_____________________________________________________"//
        "_____________________________________________________\n";
   cout<<"\t\tINPUT DATA\n";
   cout<<"-------------------------------------------------------------------"//
   "-------------------------------------------------\n";
   cout<<"Sl.no"<<setw(12)<<right<<"Name"<<setw(20)<<right<<"ID number"//
        <<setw(20)<<right<<"Monthly Salary"<<setw(20)<<right<<//
        "Annual Salary"<<setw(20)<<right<<"Year of Service"<<endl;
   cout<<"-------------------------------------------------------------"//
        "-----------------------------------------------------\n";
   for(int i=0;i<n;i++)
     {
       cout<<" "<<i+1<<"."<<setw(15)<<right;
       e[i]->display();
     }
   cout<<"_____________________________________________________"//
        _____________________________________________________\n";
   cout<<endl;
   return 0;
 }
```

SAMPLE INPUT-OUTPUT

```
PURE VIRTUAL FUNCTION AND ABSTRACT BASE CLASSES

Enter number of employees: 3

Employee 1
Name           : Manju
ID Number      : 23041
Monthly salary : 25000
Service Years  : 1

Employee 2
Name           : Pranav
ID Number      : 23523
Monthly salary : 50000
Service Years  : 3

Employee 3
Name           : Surya
ID Number      : 23527
Monthly salary : 75000
Service Years  : 5
____________________________________________________________________________________________________
                 INPUT DATA
----------------------------------------------------------------------------------------------------
Sl.no        Name           ID number      Monthly Salary       Annual Salary     Year of Service
----------------------------------------------------------------------------------------------------
 1.          Manju            23041              25000              300000                  1
 2.         Pranav            23523              50000              600000                  3
 3.          Surya            23527              75000              900000                  5
____________________________________________________________________________________________________
```

TEMPLATE CLASS

AIM

To demonstrate the use of class templates and find the sum of list of numbers.

PROGRAM

```
#include<iostream>
using namespace std;
template <class T>
class list
{
 int length;
 T *p;
 T s=0.0;
 public:
  list()
  {
   length=0;
   p=new T[length+1];
  }
   void input(int l)
   {
    length=l;
    for(int i=0;i<length;i++)
     {
        cin>>p[i];
     }
   }
   void sum(void)
   {
     cout<<"--------------------------------------------------\n";
     cout<<"\tSum : ";
     for(int i=0;i<length;i++)
      {
         s=s+p[i];
      }
     cout<<s;
     cout<<"\n--------------------------------------------------\n";
     cout<<endl;
   }
```

```
};
int main()
{
 cout<<"\n\t      TEMPLATES\n";
 list <int> L1;
 list <float> L2;
 list <double> L3;
 int len,ch;
 do{
 cout<<"\n\tSUM of the LIST ELEMENTS\nselect the datatype of the List "//
      "\n1.Integer\n2.Float\n3.Exit\n";
 cin>>ch;
 cout<<endl;
 switch(ch)
 {
  case 1:
        { cout<<"Number of elements in the list : ";
          cin>>len;
          cout<<"Enter the elements : ";
          L1.input(len);
          L1.sum();
          break;
         }
  case 2:
        { cout<<"Number of elements in the list : ";
          cin>>len;
          cout<<"Enter the elements : ";
          L2.input(len);
          L2.sum();
          break;
        }
  case 3:   break;
  default:cout<<"Wrong choice.\n";break;
  }
  int t;
  if(ch!=3)
    {    cout<<"\nDo you want to continue-1 Or exit-0 :  ";
         cin>>t;
         if(t==0)
           ch=3;
    }
```

```
  }while(ch!=3);
  cout<<"\tEND\n";
  return 0;
 }
```

SAMPLE INPUT-OUTPUT

```
               TEMPLATES

        SUM of the LIST ELEMENTS
select the datatype of the List
1.Integer
2.Float
3.Exit
1

Number of elements in the list : 5
Enter the elements : 23 45 3 8 10
--------------------------------------------------
        Sum : 89
--------------------------------------------------


Do you want to continue-1 0r exit-0 :  1

        SUM of the LIST ELEMENTS
select the datatype of the List
1.Integer
2.Float
3.Exit
2

Number of elements in the list : 4
Enter the elements : 1.8 2.4 5.2 6.1
--------------------------------------------------
        Sum : 15.5
--------------------------------------------------


Do you want to continue-1 0r exit-0 :  0
        END
```

EXCEPTION HANDLING

AIM

To demonstrate the use of exception handling and to find the sum, difference, product and quotient of two numbers.

PROGRAM

```
#include<iostream>
#include<exception>
#include<cstring>
using namespace std;
void function(float i,float j)
{ float x;
 if(j==0)
    throw 1;
 else

    x=i/j;
     cout<<"Quotient   : "<<x;
}

int main()
{  int a,b,ch=1;
  cout<<"\n\t MATHEMATIC OPERATIONS\n";
  do{
  cout<<"\nEnter the two numbers to perform the operations.\n";
  cin>>a>>b;
  cout<<"\n-----------------------------------------------------\n";
  cout<<"Sum        : "<<a+b<<endl;
  cout<<"Difference : "<<a-b<<endl;
   try
      {
        function(a,b);

      }
    catch(int i)
      {
        cout<<"Division by 0 is not possible.";
      }
       cout<<"\n-----------------------------------------------------\n";
```

```
        cout<<"\nDo you want to continue ('1'-continue or '0'-exit) : ";
        cin>>ch;
        }while(ch!=0);
 cout<<endl;
 return 0;
 }
```

SAMPLE INPUT-OUTPUT

```
         MATHEMATIC OPERATIONS

Enter the two numbers to perform the operations.
45 23

----------------------------------------------------
Sum        : 68
Difference : 22
Quotient   : 1.95652
----------------------------------------------------

Do you want to continue ('1'-continue or '0'-exit) : 1

Enter the two numbers to perform the operations.
12 0

----------------------------------------------------
Sum        : 12
Difference : 12
Division by 0 is not possible.
----------------------------------------------------

Do you want to continue ('1'-continue or '0'-exit) : 0
```