SRI CHANDRASEKHARENDRA SARASWATHI VISWA MAHAVIDYALAYA

(UNIVERSITY ESTABLISHED UNDER SECTION 3 OF UGC ACT 1956)

ENATHUR, KANCHIPURAM – 631 561

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Name : _J . SHANMUKHA SAI__________________

Reg. No : 11249A344

Class : II B.E. (CSE)

Course Code:

Course Name:

SRI CHANDRASEKHARENDRA SARASWATHI VISWA MAHAVIDYALAYA

(UNIVERSITY ESTABLISHED UNDER SECTION 3 OF UGC ACT 1956)

ENATHUR, KANCHIPURAM – 631 561

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this is the bonafide record of work done by

Mr/Ms. **J . SHANMUKHA SAI**

with **Reg. No 11249A344**__________of **II-B.E.(CSE)** during the academic year 2025 – 2026.

Station:

Date:

Staff-in-charge **Head of the Department**

Submitted for the Practical examination held on________________.

Examiner-1 **Examiner-2**

EXP NO:01

STUDENT INFORMATION USING CLASS AND OBJECTS

DATE:22-08-2025

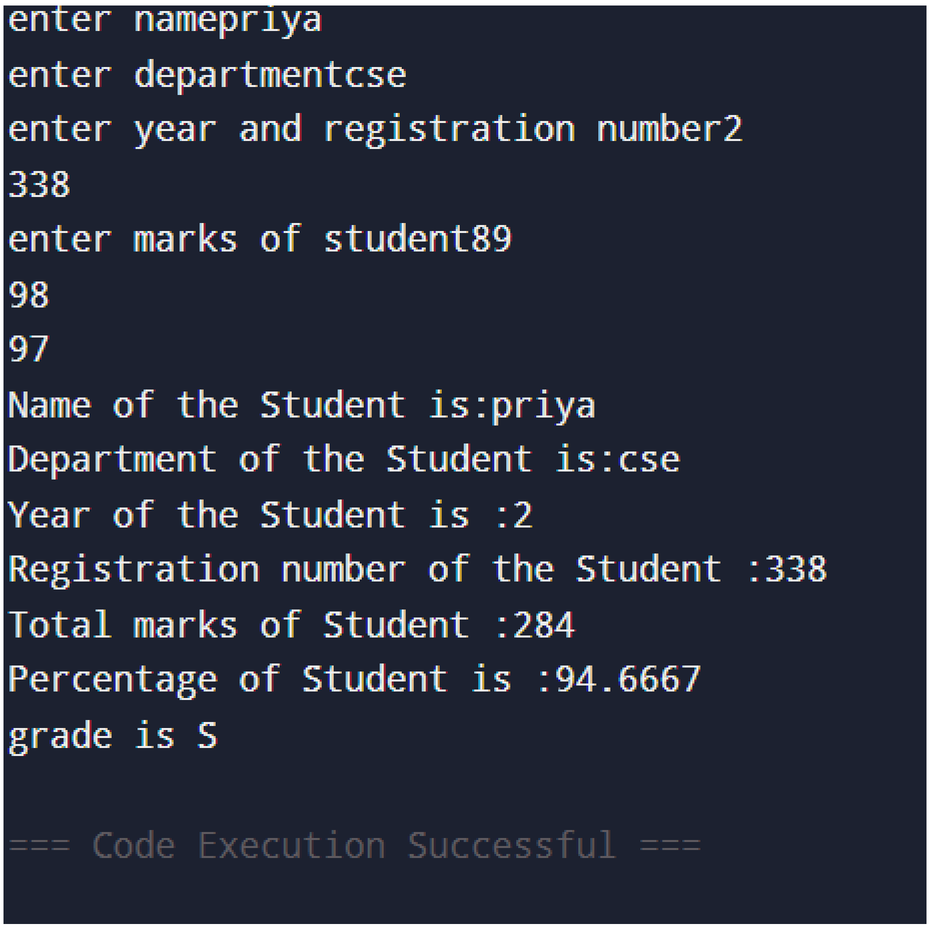
PROGRAM:

```
#include<iostream>
using namespace std;
class student
{
int  year, reg, m1, m2, m3;
char name[10],dep[6];
float total,avg,per;
public:
void read();
void cal();
void display();
};
void student :: read()
{
cout<< “enter name”;
cin>>name;
cout<< “enter department”;
cin>>dep;
cout<< “enter year and registration number”;
cin>>year>>reg;
cout<< “enter marks of student”;
```

```
cin>>m1>>m2>>m3;

}

void student :: cal()

{

total=m1+m2+m3;

avg=total/3;

per=(total/300)*100;

}

void  student :: display()

{

cout<< “Name of the Student is:”<<name<<endl;

cout<< “Department of the Student is:”<<dep<<endl;

cout<< “Year of the Student is :”<<year<<endl;

cout<< “Registration number of the Student :”<<reg<<endl;

cout<< “Total marks of Student :”<<total<<endl;

cout<< “Percentage of Student is :”<<per<<endl;

if (per>90 && per<100)

cout<< “grade is S”;

if (per>80 && per<91)

cout<< “grade is A”;

if (per>70 && per<81)

cout<< “grade is B”;

if (per>60 && per<71)

cout<< “grade is C”;

if (per>50 && per<61)

cout<< “grade is D”;

if(per<50)

cout<< “FAIL”;
```

```
}
int main ()
{
student a;
a.read();
a.cal();
a.display();
return 0;
}
```

OUTPUT:



```
enter namepriya
enter departmentcse
enter year and registration number2
338
enter marks of student89
98
97
Name of the Student is:priya
Department of the Student is:cse
Year of the Student is :2
Registration number of the Student :338
Total marks of Student :284
Percentage of Student is :94.6667
grade is S

=== Code Execution Successful ===
```

EXP NO:02

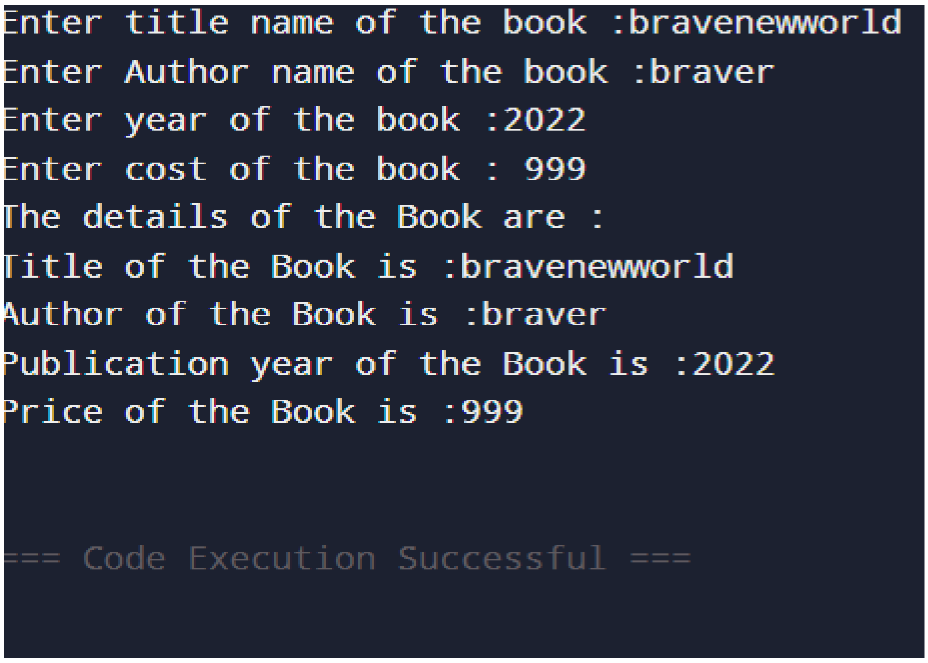
BOOK DETAILS USING CLASS AND OBJECTS AND DEFINING MEMBER FUNCTION OUTSIDE OF THE CLASS.

DATE:22-08-2025

PROGRAM:

```
#include<iostream>
using namespace std;
class bookdetails
{
 int year, cost;
char title[20],name[10];
public:
void read();
void display();
};
void  bookdetails :: read()
{
cout<< “Enter title name of the book :”;
cin>>title;
cout<< “Enter Author name of the book :”;
cin>>name;
cout<< “Enter year of the book :”;
cin>>year;
cout<< “Enter cost of the book : “;
cin>>cost;
}
void  bookdetails :: display()
{
cout<< “The details of the Book are :”<<endl;
cout<< “Title of the Book is :”<<title<<endl;
cout<< “Author of the Book is :”<<name<<endl;
cout<< “Publication year of the Book is :”<<year<<endl;
cout<< “Price of the Book is :”<<cost <<endl;
}
int main()
{
bookdetails a;
a.read();
a.display();
return 0;
}
```

OUTPUT:



```
Enter title name of the book :bravenewworld
Enter Author name of the book :braver
Enter year of the book :2022
Enter cost of the book : 999
The details of the Book are :
Title of the Book is :bravenewworld
Author of the Book is :braver
Publication year of the Book is :2022
Price of the Book is :999


=== Code Execution Successful ===
```

EXP NO:03

EMPLOYEE PAYSLIP USING CLASS AND OBJECTS.

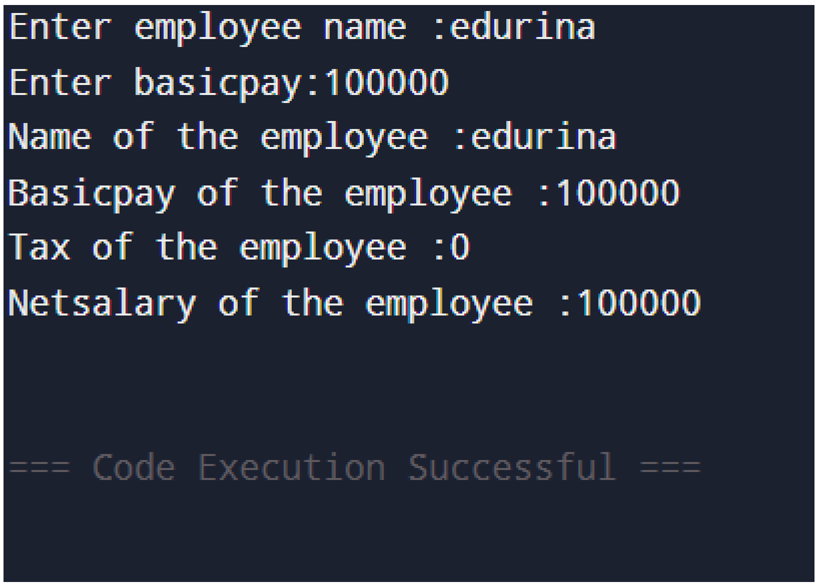
DATE:29-08-2025

PROGRAM:

```
#include<iostream>
using  namespace std;
class employee
{
int basicpay;
char name[10];
float da,hra,gs,tax,ns;
public:
void read();
void cal();
void display();
};
void  employee :: read()
{
cout<< “Enter employee name :”;
cin>>name;
cout<< “Enter basicpay:”;
cin>>basicpay;
}
void employee :: cal()
{
```

```
da=basicpay*(70/100);
hra=basicpay*(10/100);
gs=basicpay+da+hra;
tax=gs*(20/100);
ns=gs-tax;
}
void employee :: display()
{
cout<< “Name of the employee :”<<name<<endl;
cout<< “Basicpay of the employee :”<<basicpay<<endl;
cout<< “Tax of the employee :”<<tax<<endl;
cout<< “Netsalary of the employee :”<<ns<<endl;
}
int main()
{
employee a;
a.read();
a.cal();
a.display();
return 0;
}
```

OUTPUT:



```
Enter employee name :edurina
Enter basicpay:100000
Name of the employee :edurina
Basicpay of the employee :100000
Tax of the employee :0
Netsalary of the employee :100000


=== Code Execution Successful ===
```

EXP NO:04

GENERATING ELECTRICITY BILL USING CLASS AND OBJECTS

DATE:29-08-2025

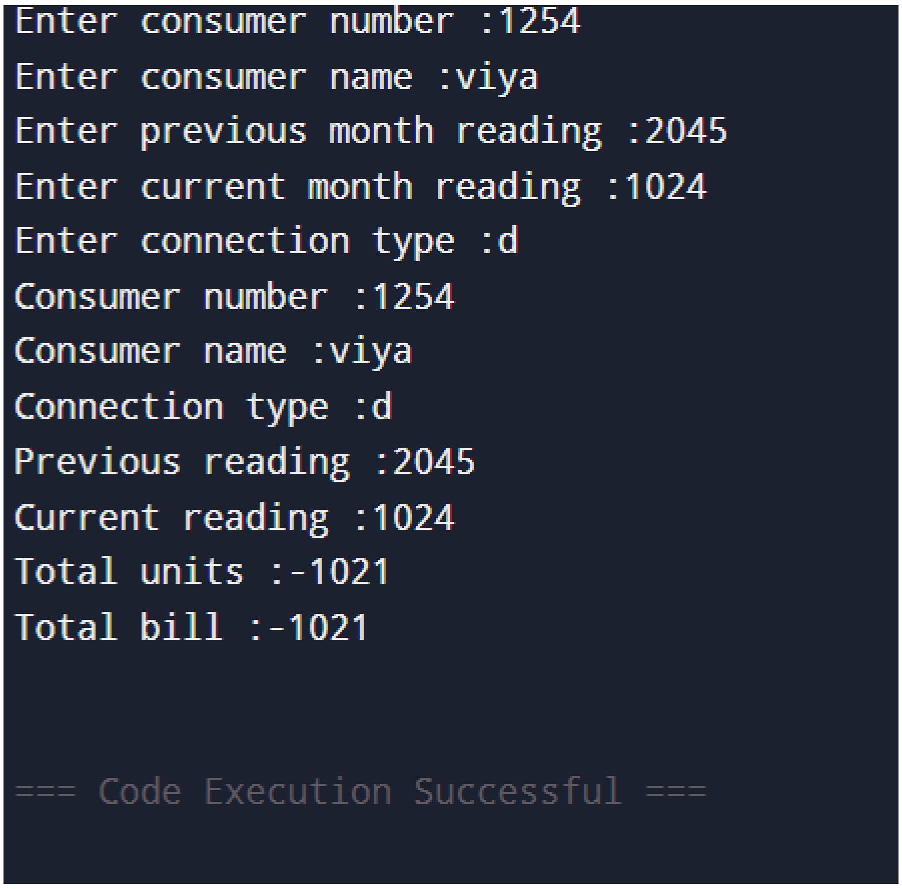
PROGRAM:

```
#include<iostream>
using  namespace std;
class electricity
{
int  con_num,pr;cr;
char name[20],type[10];
float total_units,bill_amount;
public:
void getdata()
{
cout<< “Enter consumer number :”;
cin>>con_num;
cout<< “Enter consumer name :”;
cin>>name;
cout<< “Enter previous month reading :”;
cin>>pr;
cout<< “Enter current month reading :”;
cin>>cr;
cout<< “Enter connection type :”;
cin>>type;
```

```
}
void  cal()
{
total_units=cr-pr;
bill_amount =0;
if(type[0] == ‘d’|| type[0]== ‘D’)
{
if (total_units<=100)
bill_amount=total_units*1;
else is(total_units<=200)
bill_amount=100*1+(total_units -100)*2.5;
else if (total_units<=500)
bill_amount =100*1+100*2.5+(total_units-200)*4;
else
bill_amount=100*1+100*2.5+300*4+(total_units-500)*6;
}
else
{
if(total_units<=100)
bill_amount+total_units*2;
else if(total_units<=200)
bill_amount=100*2+(total_units-100)*4.5;
else if(total_units<=500)
bill_amount=100*2+100*4.5+300*6+(total_units-500)*7;
}
}
void display()
{
```

```
cout<< “Consumer number :”<<con_num<<endl;
cout<< “Consumer name :”<<name<<endl;
cout<< “Connection type :”<<type<<endl;
cout<< “Previous reading :”<<pr<<endl;
cout<< “Current reading :”<<cr<<endl;
cout<< “Total units :”<<total_units<<endl;
cout<<Total bill :”<<bill_amount<<endl;
}
};
int main()
{
electricity e;
e.getdata();
e.cal();
e.display();
return 0;
}
```

OUTPUT:



```
Enter consumer number :1254
Enter consumer name :viya
Enter previous month reading :2045
Enter current month reading :1024
Enter connection type :d
Consumer number :1254
Consumer name :viya
Connection type :d
Previous reading :2045
Current reading :1024
Total units :-1021
Total bill :-1021


=== Code Execution Successful ===
```

EXP NO:05

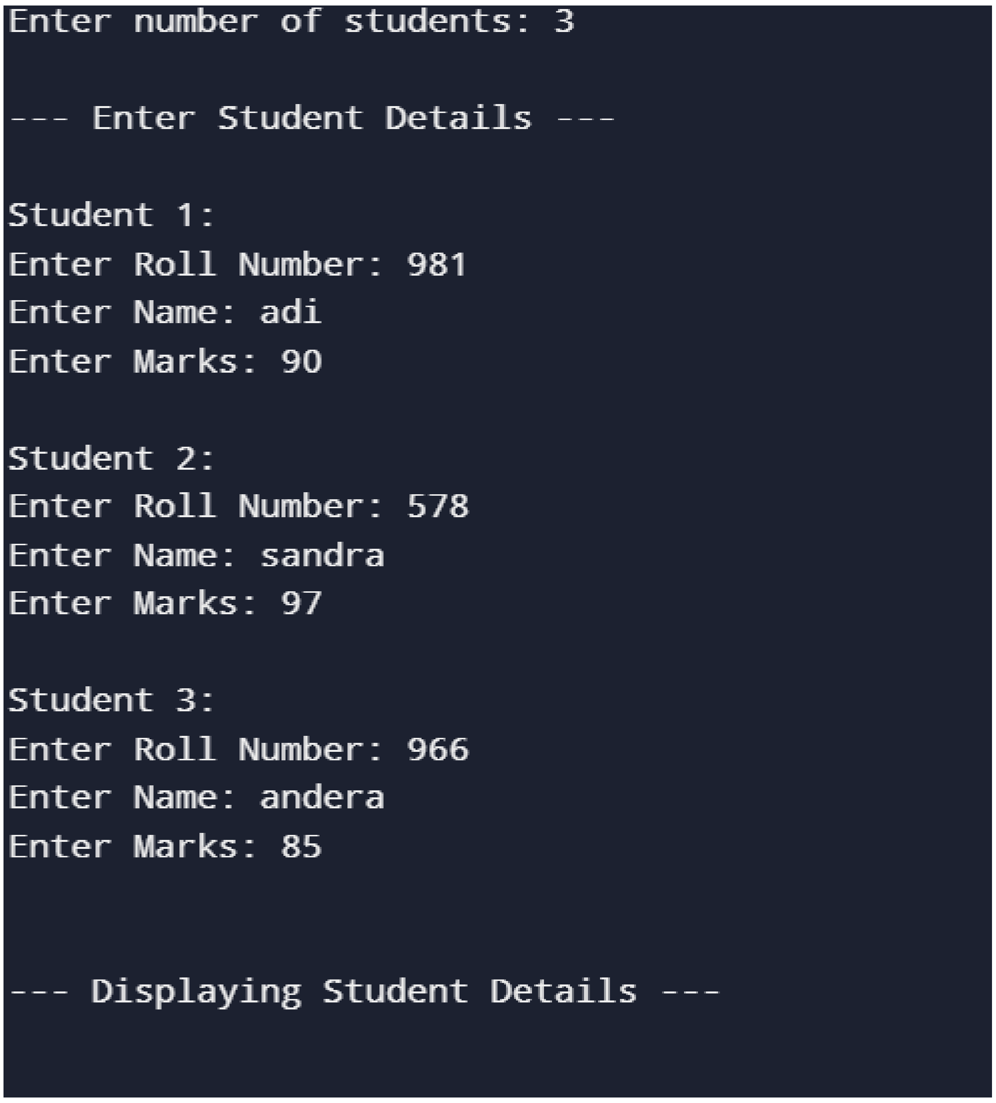
STUDENT INFORMATION USING CLASS AND OBJECTS FOR N STUDENTS.

DATE:05-09-2025

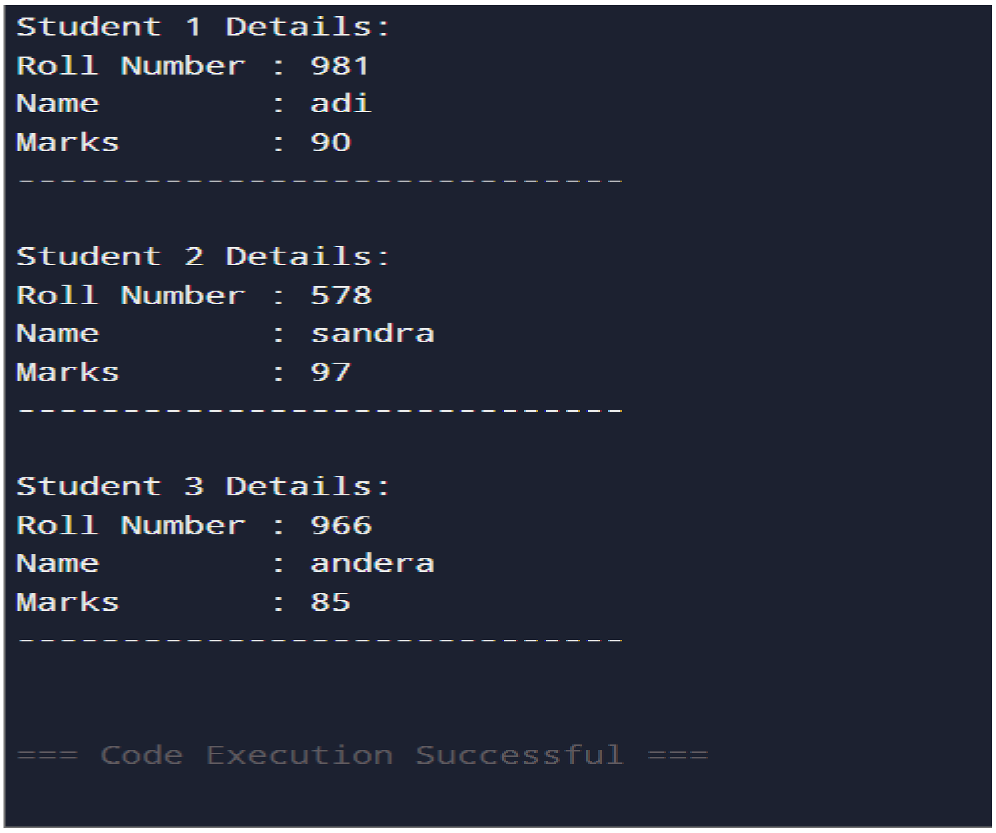
PROGRAM:

```
#include <iostream>
using namespace std;
class Student {
private:
   int roll;
   string name;
   float marks;
public:
   // Function to input student details
   void getData() {
      cout << "Enter Roll Number: ";
      cin >> roll;
      cout << "Enter Name: ";
      cin >> ws;        // to ignore newline
      getline(cin, name);
      cout << "Enter Marks: ";
      cin >> marks;
   }
   // Function to display student details
   void displayData() {
      cout << "\nRoll Number : " << roll;
      cout << "\nName       : " << name;
      cout << "\nMarks      : " << marks;
      cout << "\n----------------------------\n";
   }
};
int main() {
   int n;
   cout << "Enter number of students: ";
   cin >> n;
   // Create array of objects
   Student s[n];
   cout << "\n--- Enter Student Details ---\n";
   for (int i = 0; i < n; i++) {
      cout << "\nStudent " << i + 1 << ":\n";
      s[i].getData();
   }
   cout << "\n\n--- Displaying Student Details ---\n";
   for (int i = 0; i < n; i++) {
      cout << "\nStudent " << i + 1 << " Details:";
      s[i].displayData();
   }
   return 0;
}
```

OUTPUT:



```
Enter number of students: 3

--- Enter Student Details ---

Student 1:
Enter Roll Number: 981
Enter Name: adi
Enter Marks: 90

Student 2:
Enter Roll Number: 578
Enter Name: sandra
Enter Marks: 97

Student 3:
Enter Roll Number: 966
Enter Name: andera
Enter Marks: 85


--- Displaying Student Details ---
```



```
Student 1 Details:
Roll Number : 981
Name        : adi
Marks       : 90
------------------------------

Student 2 Details:
Roll Number : 578
Name        : sandra
Marks       : 97
------------------------------

Student 3 Details:
Roll Number : 966
Name        : andera
Marks       : 85
------------------------------


=== Code Execution Successful ===
```

EXP NO:06

DEMONSTRATE THE CONSTRUCTOR AND DESTRUCTOR.

DATE:05-09-2025

PROGRAM:

```
#include<iostream>
using namespace std;
class marks
{
public:
int maths,science;
//constructor
marks()
{
cout<< “Inside constructor”<<endl;
cout<< “C++ object created”<<endl;
}
//destructor
~marks()
{
cout<< “Inside destructor”<<endl;
cout<< “ C++ object destroyed”<<endl;
}
};
int main()
{
```

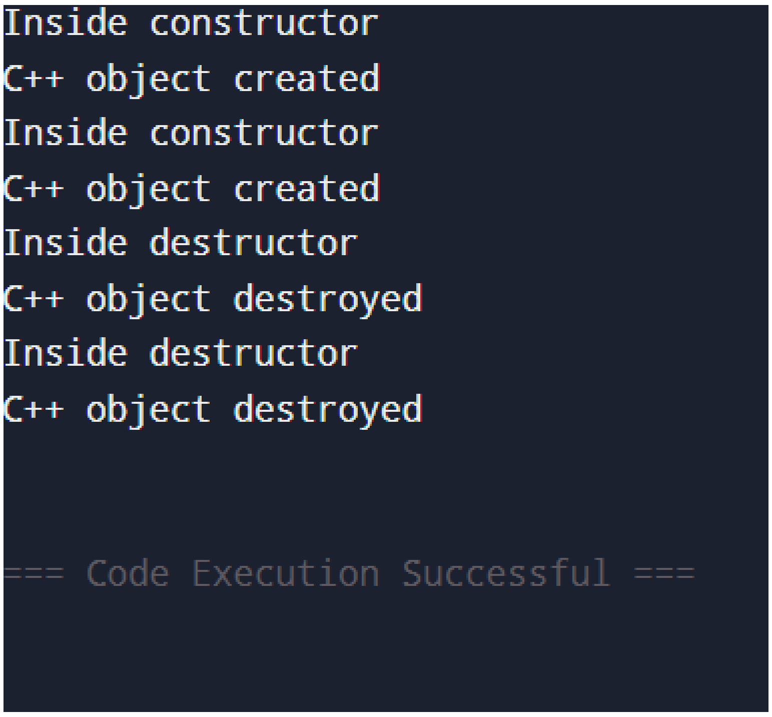
marks m1;

marks m2;

return 0;

}

OUTPUT:



```
Inside constructor
C++ object created
Inside constructor
C++ object created
Inside destructor
C++ object destroyed
Inside destructor
C++ object destroyed


=== Code Execution Successful ===
```

EXP NO:07

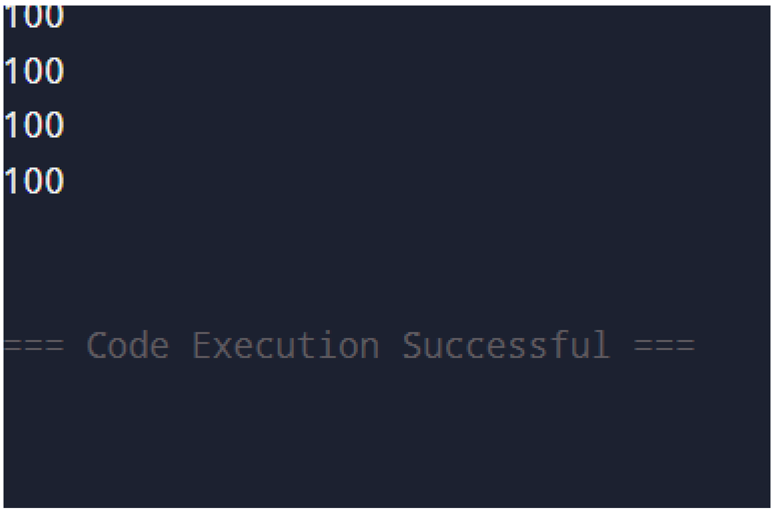
IMPLEMENT CONSTRUCTOR OVERLOADING.

DATE:12-09-2025

PROGRAM:

```
#include<iostream>

using  namespace std;

class sample

{

int n;

public:
sample() //default constructor

{

n=0;

}

sample( int a)//parameterized constructor

{

n=a;

}

sample(sample &x)//copy constructor

{

n=x.n;

}

void display()

{

cout<<n<<endl;

}
};
int main()
{
sample  a(100);
sample b(a);
sample c=a;
sample d;
d=a;
a.display();
b.display();
c.display();
d.display();
return 0;
}
```

OUTPUT:



```
100
100
100
100


=== Code Execution Successful ===
```

EXP NO:08

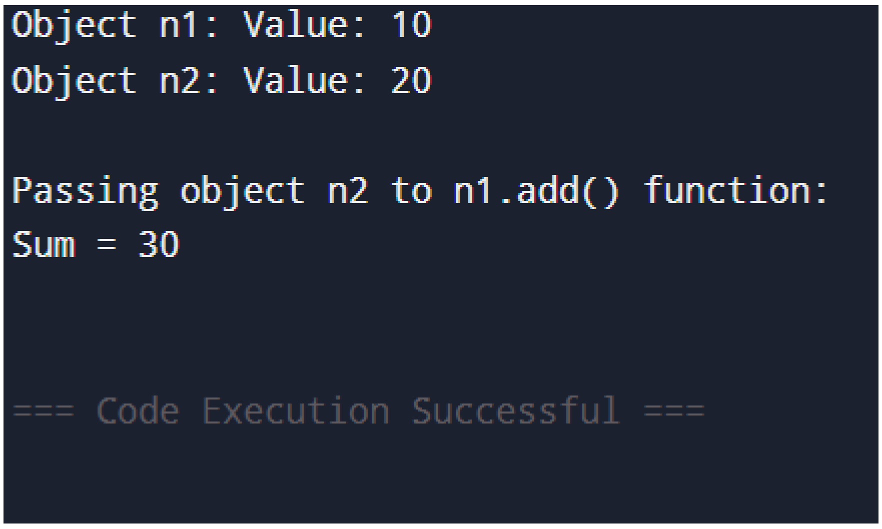
IMPLEMENT OBJECT AS FUNCTION ARGUMENTS.

DATE:12-09-2025

PROGRAM:

```
#include <iostream>
using namespace std;
class Number
 {
private:
int value;
public:
// Constructor
Number(int v = 0)
 {
      value = v;
 }
// Function that displays value
   void display()
 {
      cout << "Value: " << value << endl;
 }
   // Function that takes an object as argument
   void add(Number n) {   // object passed as argument
      cout << "Sum = " << value + n.value << endl;
   }
};


int main() {
    Number n1(10);
    Number n2(20);
  cout << "Object n1: ";
  n1.display();
  cout << "Object n2: ";
  n2.display();
  cout << "\nPassing object n2 to n1.add() function:\n";
   n1.add(n2);
   return 0;
}
```

OUTPUT:



```
Object n1: Value: 10
Object n2: Value: 20

Passing object n2 to n1.add() function:
Sum = 30


=== Code Execution Successful ===
```

EXP NO:09

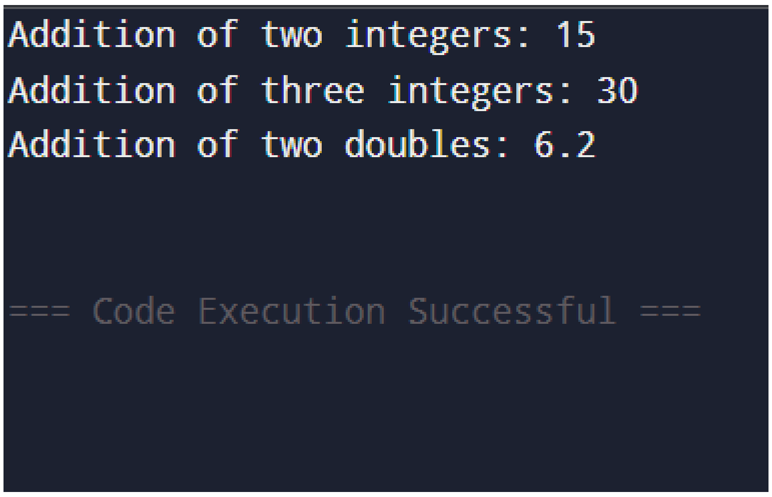
DEMONSTRATE FUNCTION OVERLOADING.

DATE:19-09-2025

PROGRAM:

```
#include <iostream>

using namespace std;

// Function to add two integers

int add(int a, int b)

 {

   return a + b;

}

// Function to add three integers

int add(int a, int b, int c)

 {

   return a + b + c;

}

// Function to add two double values

double add(double a, double b)

 {

   return a + b;

}

int main()

 {

   cout << "Addition of two integers: " << add(5, 10) << endl;

   cout << "Addition of three integers: " << add(5, 10, 15) << endl;

   cout << "Addition of two doubles: " << add(3.5, 2.7) << endl;
   return 0;
}
```

OUTPUT:



```
Addition of two integers: 15
Addition of three integers: 30
Addition of two doubles: 6.2


=== Code Execution Successful ===
```

EXP NO: 10

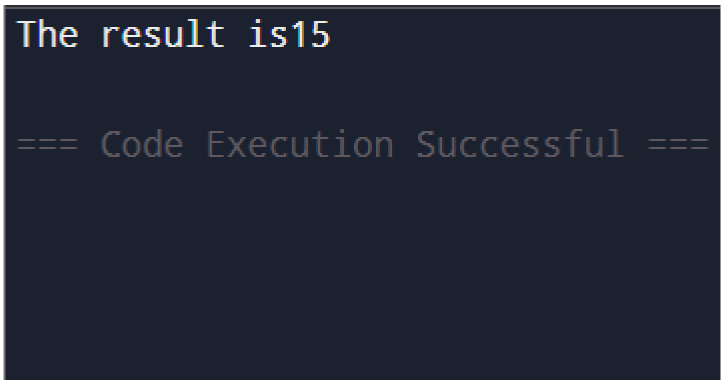
INLINE FUNCTION

DATE:19-09-2025

PROGRAM:

```
#include<iostream>

using namespace std;

inline int add (int a,int b)

{

return (a+b);

}

int main()

{

int result=add(5,10)

cout<<" The result is "<<result;

return o;

}
```

OUTPUT:



```
The result is15


=== Code Execution Successful ===
```

EXP NO:11

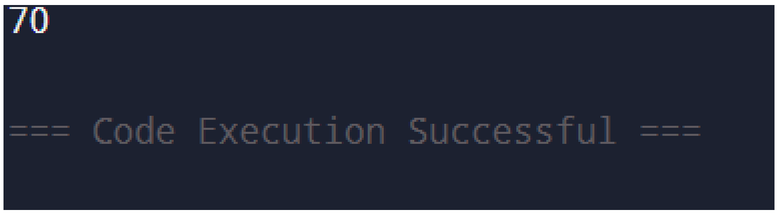
STATIC DATA MEMBERS AND STATIC MEMBER FUNCTION

DATE:19-09-2025

PROGRAM:

```
#include<iostream>
using namespace std;
Class A
{
public:
static int a,b;
static int add(int,int);
};
int A::add(int a,int b)
{
return(a+b);
}
int main()
{
A a,b,c;
int res;
res A::add(30,40);
cout<<res;
}
```

OUTPUT:



```
70


=== Code Execution Successful ===
```

EXP NO:12

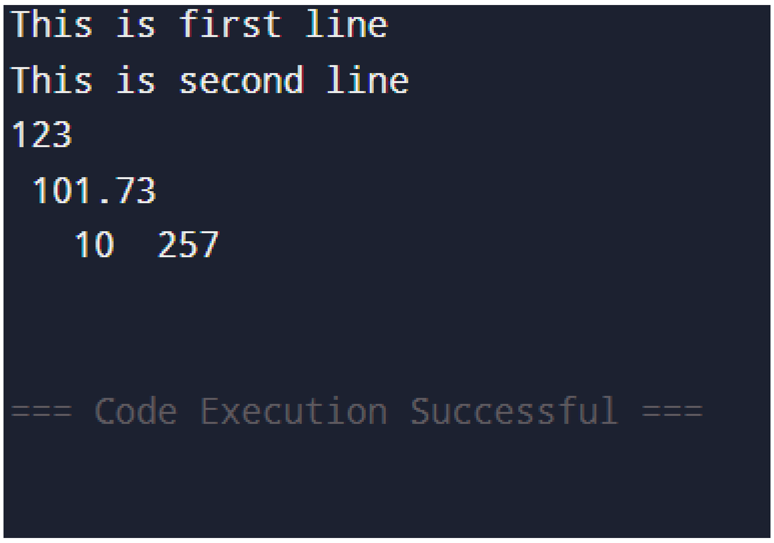
DEMONSTRATE THE USE OF MANIPULATORS

DATE:03-10-2025

PROGRAM:

```
#include<iostream>

#include<iomanip>

#include<math.h>

using namespace std;

int main()

{

cout<<”This is first line”<<endl;

cout<<”This is second line”<<endl;

cout<<123<<endl;

cout<<setw(3)<<10;

cout<<set precision(3);

cout<<sqrt(3)<<endl;

cout<<setw(5)<<10;

cout<<setw(5)<<257<<endl;

return o;

}
```

OUTPUT:



```
This is first line
This is second line
123
 101.73
   10   257


=== Code Execution Successful ===
```

EXP NO:13

IMPLEMENTATION OF UNARY OPERATOR OVERLOADING.

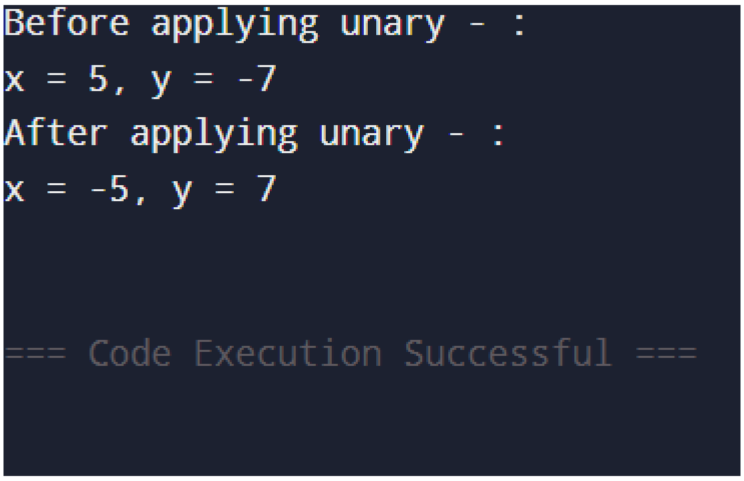
DATE:03-10-2025

PROGRAM:

```
#include <iostream>
using namespace std;
class Sample {
private:
 int x, y;
public:
 Sample(int a, int b)
 {
 x = a;
  y = b;
 }
 // Overload unary minus
 void operator - ()
 {
  x = -x;
   y = -y;
 }
 void display()
 {
    cout << "x = " << x << ", y = " << y << endl;
 }
};
```

```
int main()
{
 Sample s(5, -7);
 cout << "Before applying unary - :" << endl;
  s.display();
 -s;   // Calls operator-()
 cout << "After applying unary - :" << endl;
 s.display();
 return 0;
}
```

OUTPUT:



```
Before applying unary - :
x = 5, y = -7
After applying unary - :
x = -5, y = 7


=== Code Execution Successful ===
```

EXP NO:14

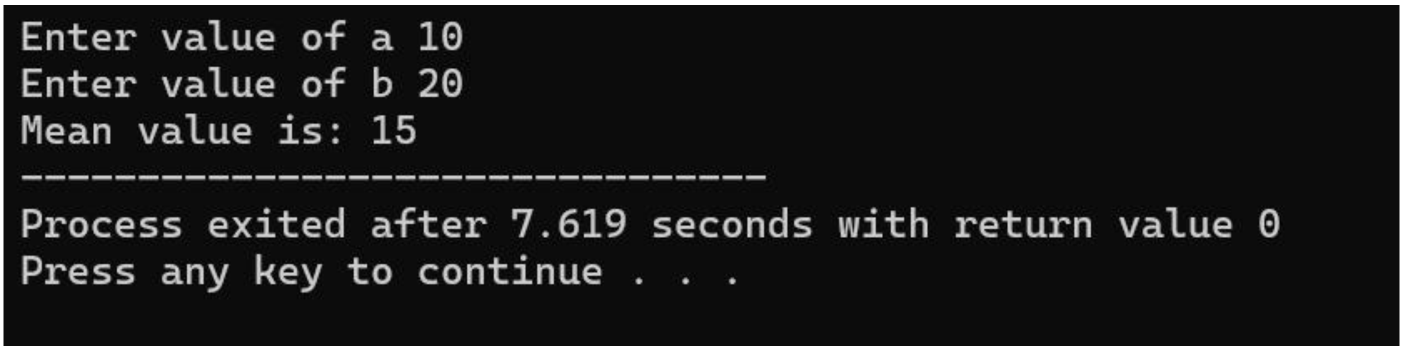
IMPLEMENTATION OF FRIEND FUNCTION.

DATE:10-10-2025

PROGRAM:

```
#include<iostream>

using namespace std;

 class sample

{

int a, b;

public:

void setvalue()

{

cout<<"Enter value of a ";

cin>>a;

cout<<"Enter value of b ";

cin>>b;

}

Friend float mean(sample);

float mean(sample s)

{

return float(s.a+s.b)/2.0;

}

};

int main()

{

clrscr();
sample x;

x.setvalue();

cout<<"Mean value is: "<<mean(x);

return 0;

}
```

OUTPUT:



```
Enter value of a 10
Enter value of b 20
Mean value is: 15
--------------------------------
Process exited after 7.619 seconds with return value 0
Press any key to continue . . .
```

EXP NO:15

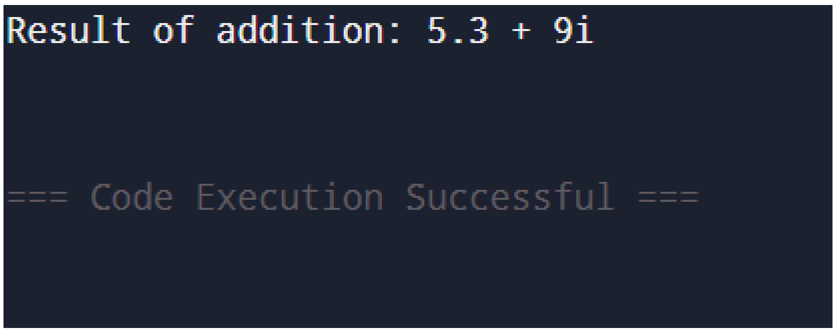
IMPLEMENTATION OF BINARY OPERATOR OVERLOADING.

DATE:10-10-2025

PROGRAM:

```
#include <iostream>
using namespace std;
class Complex
{
private:
 float real, imag;
public:
// Constructor
Complex(float r = 0, float i = 0)
{
real = r;
 imag = i;
 }
// Overloading the + operator
Complex operator + (const Complex &obj)
 {
      Complex temp;
      temp.real = real + obj.real;
      temp.imag = imag + obj.imag;
      return temp;
   }
 // Function to display result
   void display()
{
       cout << real << " + " << imag << "i" << endl;
   }
};
int main() {
Complex c1(3.2, 5.4);
 Complex c2(2.1, 3.6);
Complex c3 = c1 + c2;   // Calls operator+
 cout << "Result of addition: ";
 c3.display();
return 0;
}
```

OUTPUT:



```
Result of addition: 5.3 + 9i


=== Code Execution Successful ===
```

EXP NO:16

IMPLEMENTATION OF STRING CONCATENATION USING BINARY OVERLOADING (+).

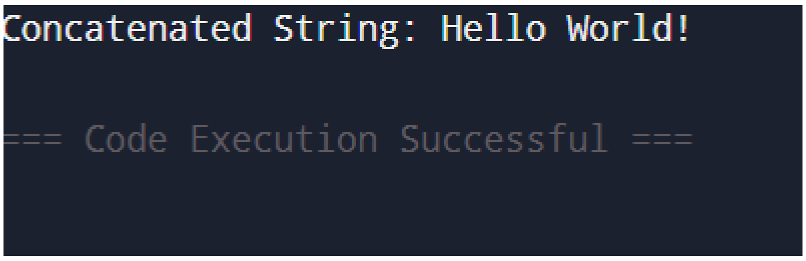
DATE:17-10-2025

PROGRAM:

```
#include <iostream>
using namespace std;
class MyString {
private:
 char str[100];
public:
// Constructor
 MyString(const char s[] = "")
 {
 strcpy(str, s);
}
 // Overloading + operator
MyString operator + (const MyString &obj)
 {
MyString temp;
strcpy(temp.str, str);        // Copy first string
strcat(temp.str, obj.str);     // Concatenate second string
return temp;
}
// Display function
void display()
```

```
 {
cout << str;
}
};
int main()
 {
MyString s1("Hello ");
MyString s2("World!");
MyString s3 = s1 + s2;  // String concatenation using + operator
cout << "Concatenated String: ";
s3.display();
return 0;
}
```

OUTPUT:



```
Concatenated String: Hello World!

=== Code Execution Successful ===
```

EXP NO:17

DEMONSTRATE SINGLE INHERITANCE.

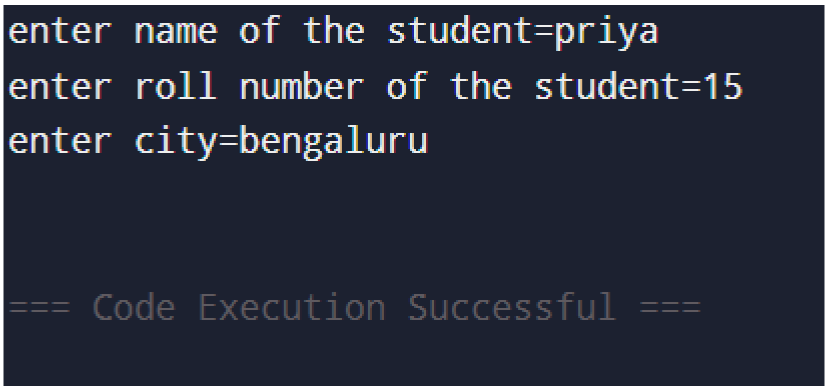
DATE:17-10-2025

PROGRAM:

```
#include<iostream.h>
using namespace std;
class student
{
private:
char name[20];
int rno;
public:
void getstudent()
{
cout<<”enter name of the student=”;
cin>>name;
cout<<”enter roll number of the student=”;
cin>>rno;
}
void displaystudent()
{
cout<<”name of the student=”<<name;
cout<<”\nroll number of the student=”<<rno;
}
}; // class termination
class address : public student
```

```
{
private:
char city[20];
public:
void getaddress()
{
getstudent();
cout<<”\nenter city=”;
cin>>city;
}
void displayaddress()
{
displaystudent();
cout<<”\ncity=”<<city;
}
};
int main()
{
address a1;
clrscr();
a1.getaddress();
return 0;
}
```

OUTPUT:



```
enter name of the student=priya
enter roll number of the student=15
enter city=bengaluru



=== Code Execution Successful ===
```

EXP NO:18

DEMONSTRATE MULTILEVEL INHERITANCE.

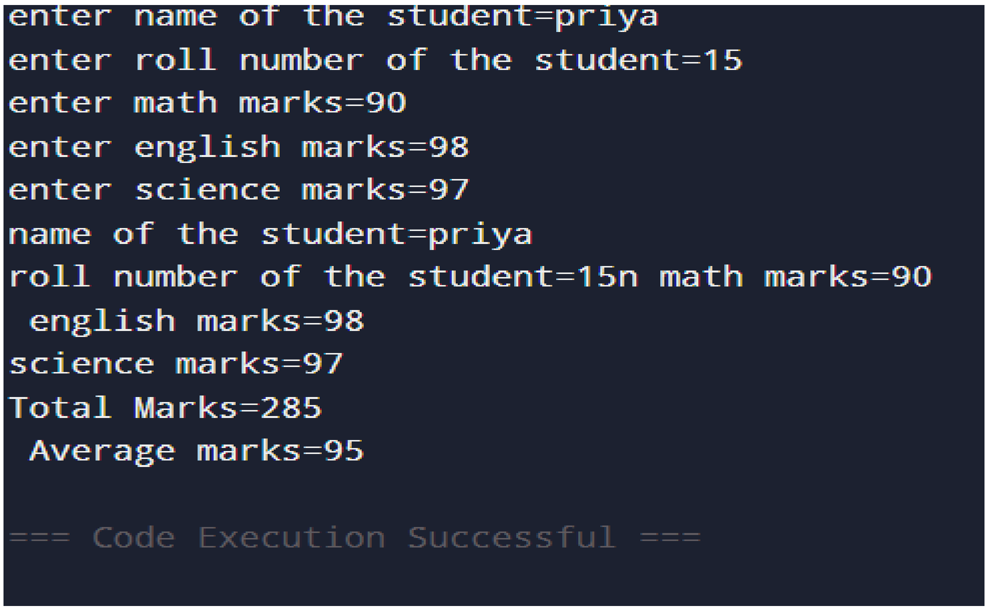
DATE:24-10-2025

PROGRAM:

```
#include<iostream.h>
using namespace std;
 class student
{
private:
char name[20];
int rno;
public:
void getstudent()
{
cout<<”enter name of the student=”;
cin>>name;
cout<<”enter roll number of the student=”;
cin>>rno;
}
void displaystudent()
{
cout<<”name of the student=”<<name;
cout<<”\nroll number of the student=”<<rno;
}
};
class address
```

```
{
private:
char city[20];
public:
void getaddress()
{
cout<<”\nenter city=”;
cin>>city;
}
void displayaddress()
{
cout<<”\ncity=”<<city;
}
};
class account: public student, public address
{
private:
int tfee,submit,balance;
public:
void getaccount()
{
getstudent();
getaddress();
cout<<“\nenter total fee=“;
cin>>tfee;
cout<<“\nenter submit fee=“ ;
cin>>submit;
}
};
void displayaccount()
{
displaystudent();
displayaddress();
cout<<”\ntotal fee=”<<tfee;
cout<<”\nsubmit fee=”<<submit;
balance=tfee-submit;
cout<<“\nbalance fee=“<<balance;
}
int main()
{
account a1;
a1.getaccount();
a1.displayaccount();
return 0;
}
```

OUTPUT:



```
enter name of the student=priya
enter roll number of the student=15
enter math marks=90
enter english marks=98
enter science marks=97
name of the student=priya
roll number of the student=15n math marks=90
 english marks=98
science marks=97
Total Marks=285
 Average marks=95

=== Code Execution Successful ===
```

EXP NO:19

DEMONSTRATE MULTILEVEL INHERITANCE.

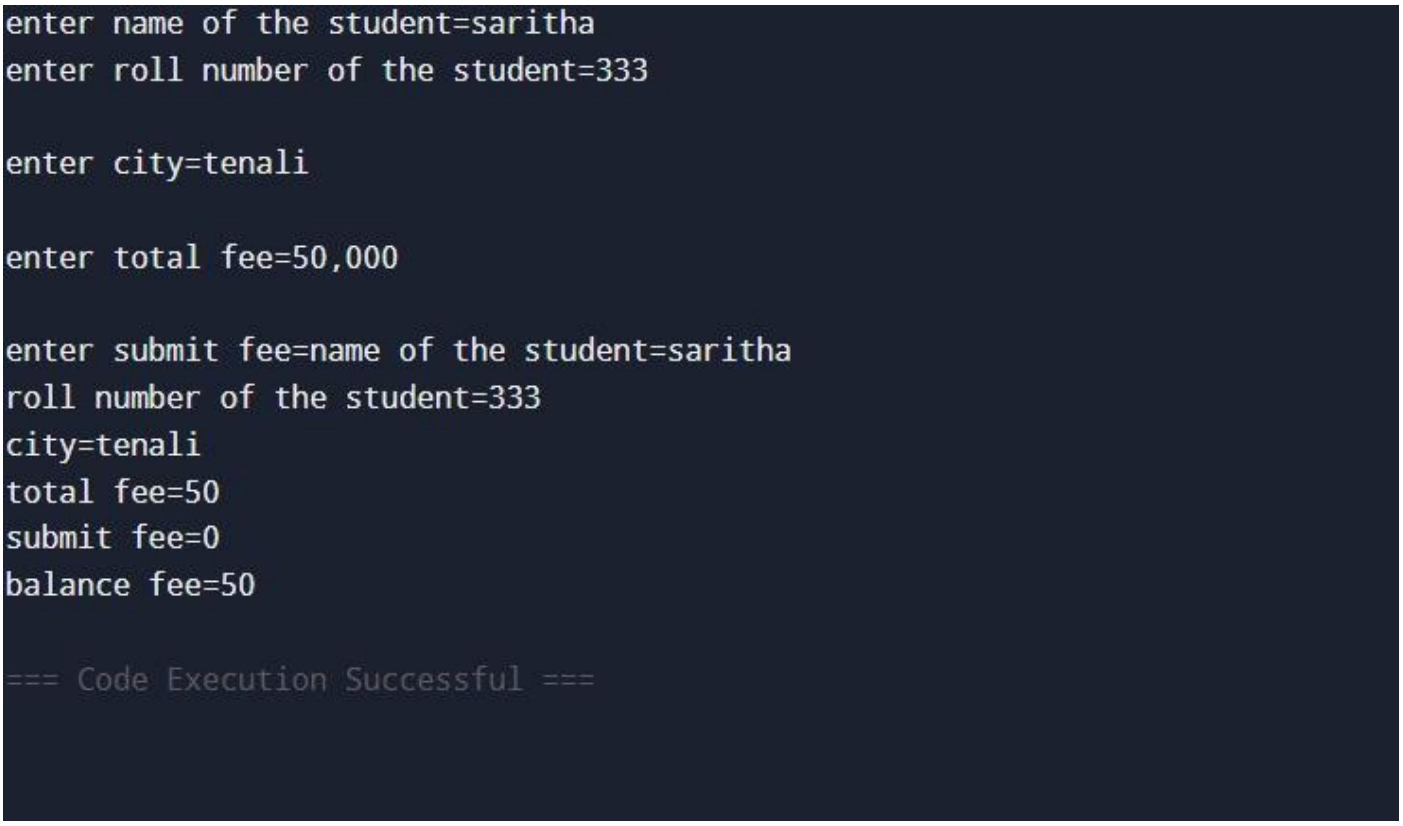
DATE:24-10-2025

PROGRAM:

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
class student
{
private:
char name[20];
int rno;
public:
void getstudent()
{
cout<<”enter name of the student=”;
cin>>name;
cout<<”enter roll number of the student=”;
cin>>rno;
}
void displaystudent()
{
cout<<”name of the student=”<<name;
cout<<”\nroll number of the student=”<<rno;
}
};
```

```
class test: public student
{
protected:
int math,eng,sci;
public:
void gettest()
{
getstudent();
cout<<”enter math marks=”;
cin>>math;
cout<<”enter english marks=”;
cin>>eng;
cout<<”enter science marks=”;
cin>>sci;
}
void displaytest()
{
displaystudent();
cout<<”\n math marks=”<<math;
cout<<”\n english marks=”<<eng;
cout<<”\nscience marks=”<<sci;
}
};
class result : public test
{
private:
int total,avg;
public:
void getresult()
{
gettest();
total=math+eng+sci;
avg=total/3;
}
void displayresult()
{
displaytest();
cout<<”\nTotal Marks=”<<total;
cout<<”\n Average marks=”<<avg;
}
};
int main()
{
result r1;
clrscr();
r1.getresult();
clrscr();
r1.displayresult();
return 0;
}
```

OUTPUT:



```
enter name of the student=saritha
enter roll number of the student=333

enter city=tenali

enter total fee=50,000

enter submit fee=name of the student=saritha
roll number of the student=333
city=tenali
total fee=50
submit fee=0
balance fee=50

=== Code Execution Successful ===
```

EXP NO:20

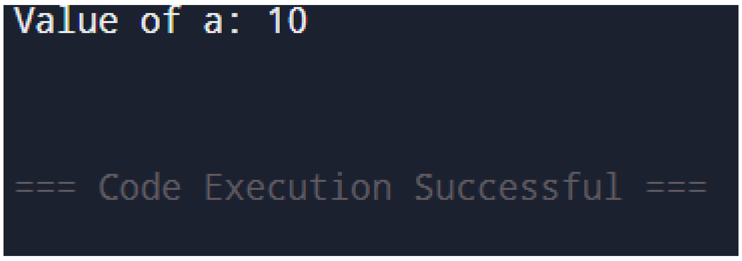
IMPLEMENTATION OF MEMORY MANAGEMENT OPERATOR

DATE:31-10-2025

PROGRAM:

```
#include <iostream>

using namespace std;

int main()

 {

   // Allocate memory for a single integer

   int *a = new int;

   *a = 10;

   cout << "Value of a: " << *a << endl;

   // Deallocate the memory

   delete a;

   // Allocate memory for an array of integers

   int *arr = new int[5];

   for (int i = 0; i < 5; ++i) {

      arr[i] = i * 2;

   }

   // Deallocate the array memory

   delete[] arr;

   return 0;

}
```

OUTPUT:



```
Value of a: 10


=== Code Execution Successful ===
```

EXP NO:21

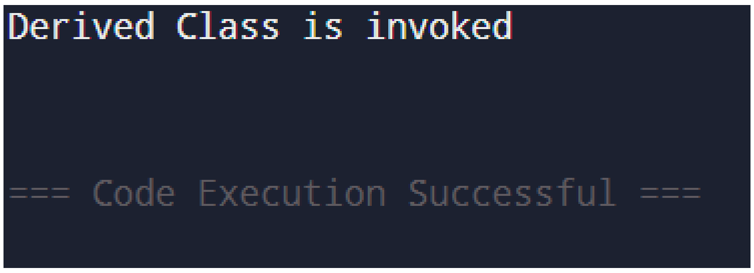
IMPLEMENTATION OF VIRTUAL FUNCTION.

DATE:31-10-2025

PROGRAM:

```
include <iostream>
using namespace std;
Class A
{
public:
virtual void display()
{
cout << "Base class is invoked"<<endl;
}
};
class B:public A
{
public:
void display()
{
cout << "Derived Class is invoked"<<endl;
}
};
int main()
{
A* a; //pointer of base class
B b; //object of derived class
a = &b;

a->display(); //Late Binding occurs

return 0;

}
```

OUTPUT:



```
Derived Class is invoked


=== Code Execution Successful ===
```

EXP NO:22

IMPLEMENTATION OF THIS POINTER

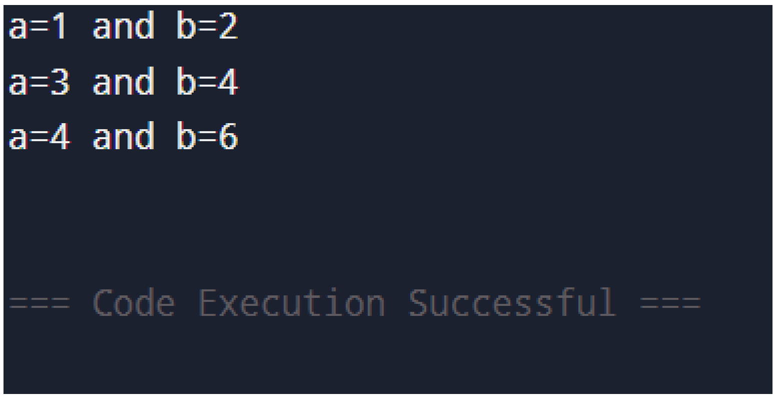
DATE:07-11-2025

PROGRAM:

```
#include<iostream>

using namespace std;

class num

{

int a,b;

public:

num(int x, int y)

{

a=x;b=y;

}

void display()

{

cout<<"a="<<a<<" and b="<<b<<endl;

}

num add(num);

};

num num:: add(num x)

{

a=a+x.a;

b=b+x.b;

return *this;

}
```

```
int main()

{

clrscr();

num obj1(1,2),obj2(3,4);

obj1.display();

obj2.display();

obj1.add(obj2);

obj1.display();

return 0;

}
```

OUTPUT:



```
a=1 and b=2
a=3 and b=4
a=4 and b=6


=== Code Execution Successful ===
```

EXP NO:23a

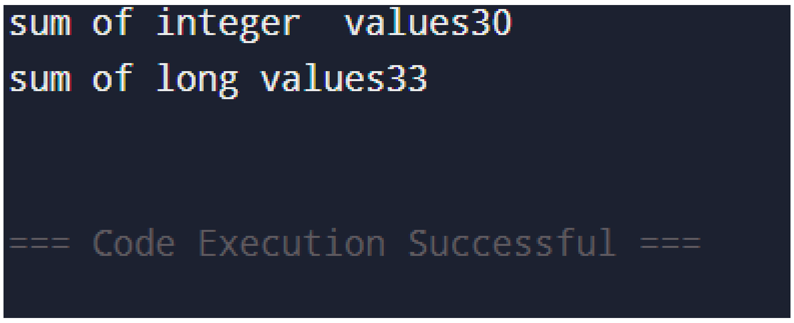
DEMONSATRATE THE FUNCTION TEMPLATE.

DATE: 07-11-2025

PROGRAM:

```
#include<iostream>

using namespace std;template<typename T>

T sum(T n1,T n2){

T rs;

rs=n1+n2;

return rs;

}

int main(){

int a=10,b=20,c;

long l=11,j=22,k;

c=sum(a,b);

cout<< “sum of integer  values”<<c<<endl;

k=sum(l,j);

cout<< “sum of long values”<<k<<endl;

return 0;

}
```

OUTPUT:



```
sum of integer  values30
sum of long values33



=== Code Execution Successful ===
```

EXP NO:23b

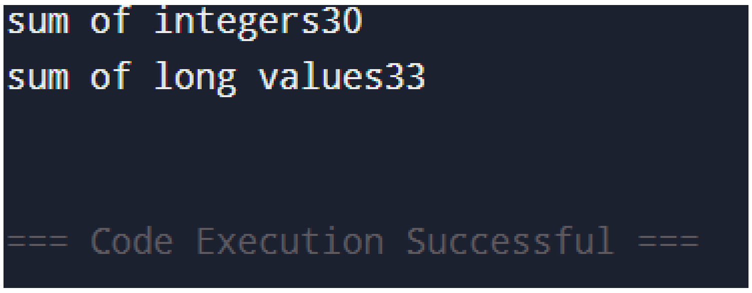
DEMONSTRATE CLASS TEMPLATE

DATE: 07-11-2025

PROGRAM:

```
#include<iostream>

using namespace std;

template<class T>

class addition

{

public:

T add(T,T);

};

template<class T>

T addition<T>::add(T n1, T n2)

{

T rs;

rs=n1+n2;

return rs;

}

int main()

{

addition<int>obj1;

addition<long>obj2;

int a=10,b=20,c;

long l=11,j=22,k;

c=obj1.add(a,b);
cout<< “sum of integers “<<c<<endl;

k=obj2.add(l,j);

cout<<  “sum of long values<<k<<endl;

return 0;

}
```

OUTPUT:



```
sum of integers30
sum of long values33


=== Code Execution Successful ===
```

EXP NO:24

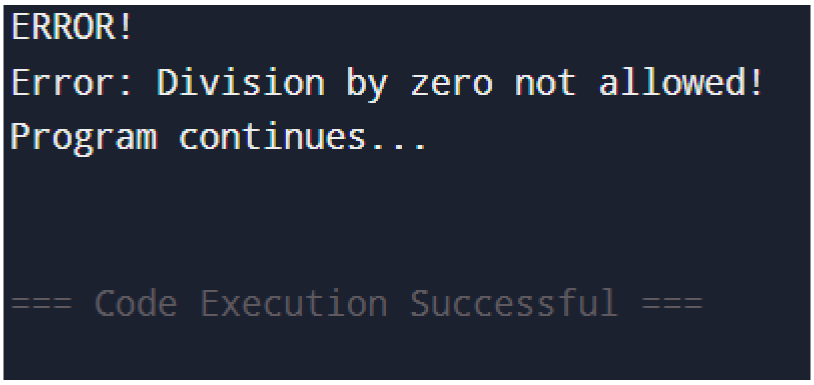
EXPECTION HANDLING

DATE: 07-11-2025

PROGRAM:

```
#include <iostream>

using namespace std;

int main()

 {

int a = 10, b = 0;

try

 {

if (b == 0)

throw "Division by zero not allowed!"; // throwing an exception

cout << "Result: " << a / b << endl;

}

catch (const char* msg) { // catching the exception

cout << "Error: " << msg << endl;

}

cout << "Program continues..." << endl;

return 0;

}
```

OUTPUT:



```
ERROR!
Error: Division by zero not allowed!
Program continues...


=== Code Execution Successful ===
```