Arya Sonawane

Roll no - 33 SY CSE (AIDS)

PRN - 12622402 73 Batch B

Experiment 1

a) WAP to declare a class student having data members as name, roll. no. Accept and display data for one student.

```
# include <iostream>
# include <string>
using namespace std;
class Student
{
    int r;
    string n;
public:
    void accept()
    {
        cout << "Enter name and roll number: ";
        cin >> n >> r;
    }
    void display()
    {
        cout << " Name = " << n << endl;
        cout << " Roll. no. = " << r << endl;
    }
};
int main()
{
    Student s1;
```

```
    s1.accept();
    s1.display();
    return 0;
}
```

Output -

```
Enter name and roll number : ABC
33
Name = ABC
Roll number = 33
```

b) WAP to declare a class Book having data members as id, name, price etc. Accept data for 2 books and display data of book having greater price.

```
# include <iostream>
# include <string>
using namespace std;
class Book;
{
public:
    int bp, bpgs;
    string n;
    void accept()
    {
        cout << "Enter name and price : ";
        cin >> n >> bp;
        cout << "Enter pages : ";
        cin >> bpgs;
    }
    void display()
    {
        cout << "Book name = " << n << endl;
        cout << "Book price = " << bp << endl;
        cout << "Book pages = " << bpgs << endl;
    }
};
```

```
int main()
{
    Book b1, b2;
    b1.accept();
    b2.accept();
    if (b1.bp > b2.bp) {
        cout << "Book with higher price:";
        b1.display();
    }
    else {
        cout << "Book with higher price:";
        b2.display();
    }
    return 0;
}
```

Output:

```
Enter name and price : ABC  50
Enter pages : 200
Enter name and price : XYZ  100
Enter pages : 300

Book with higher price :
Book name = XYZ
Book price = 100
Book pages = 300
```

c> WAP to declare a class time having data members as H, M & S. Accept data for one object and display total time in sec.

```
# include <iostream>
# include <string>
using namespace std;
class Time
{
    int h,m,s,t;
public:
    void accept()
    {
        cout << "Enter hours:";
        cin >> h;
        cout << "Enter minutes:";
        cin >> m;
        cout << "Enter seconds:";
        cin >> s;
    }
    void display()
    {
        t = (h * 3600) + (m * 60) + s;
        cout << "Total time = " << t;
    }
};
```

```
int main()
{
    Time t1;
    t1.accept();
    t1.display();
    return 0;
}
```

Output -

```
Enter hours : 4
Enter minutes : 6
Enter seconds : 89
Total time : 14849
```

Experiment 2

a) WAP to declare class CITY having data members as name and population. Accept data for 5 cities and display the one with higher population

→

```
#include <iostream>
#include <string>
using namespace std;
class City
{
public:
    string n;
    int p;
    void accept()
    {
        cout << "Enter city name and
        population : ";
        cin >> n >> p;
    }
    void disp() const
        cout << "City : " << n << ",Population : "<<p
};
int main()
{
    City c[5];
    cout << "Enter data of 5 cities :\n";
```

```
for ( int i=0 ; i<5 ; i++)
{
    cout << " City " <<i+1<< ":\n";
    c[i].accept();
}
int m = 0;
for (int i = 1; i<5 ; i++)
{
    if ( c[i].p > c[m].p)
    {
        m = i;
    }
}
cout << " City with the highest population:";
c[m].disp();
return 0;
}
```

Output -

```
Enter data for 5 cities :
City 1 :
Enter city name and population : ABC 100
City 2 :
Enter city name and population : DEF 250
City 3 :
Enter city name and population : GHI 60
City 4 :
Enter city name and population : JKL 470
City 5 :
Enter city name and population : XYZ 1000

City with the highest population :
City : XYZ , Population : 1000
```

b> WAP to declare a class 'Account' having data members: account no and balance. Accept data for 10 people

```
# include <iostream>
using namespace std;
class Account
{
  public :
    int a;
    double b;
    void input()
    {
        cout << " Enter accountno & balance:";
        cin >> a >> b;
    }
    void add()
    {
        if ( b >= 5000)
            b += b * 0.10;
    }
```

```
    void disp()
        cout << "Account no: " << a << "
        Balance : " << b << endl;
};
int main()
{
    Account A[10];
    for (int i = 0 ; i<10 ; i++)
    {
        cout << " Enter details for account "<<
        (i+1) << ":\n";
        A[i].input();
    }
    cout << " Added balance:" ;
    for (int i=0 ; i<10; i++)
    {
        A[i].b >= 5000)
            A[i].disp();
    }
    return 0;
}
```

Output -

```
Enter details for account -
1:
1
3456
2:
2
7890
3:
3
9475
4:
4
6578
5:
5
4980
6:
6
4321
7:
7
4999
8:
8
9768
9:
9
5678
10:
10
9000
Accounts with added balance:
Account no: 2, 8679
Account no: 3, 10422.5
Account no: 4, 7235.8
Account no: 8, 10744.8
Account no: 9, 6245.8
Account no: 10, 9900
```

c) WAP in C++ to define class Staff for 5 people with data-members as name and post and display data of those having post HOD.

→
```
# include <iostream>
# include <string>
using namespace std;
class Staff
{
    public:
    string n, p;
    void getData()
    {
        cout << "Enter name and post:";
        cin >> n >> p;
    }
    void disp()
        cout << "Name: " << n << ", Post: " << p;
};
int main()
{
    Staff S[10];
    cout << "Enter data for 5 people:";
    for (int i=0; i<5; i++)
    {
        cout << "Name" << i+1 << ":\n";
        S[i].getData();
    }
    int m=0;
    for (int i=0; i<5; i++)
    {
        if (S[i].p == "HOD")
        {
            m = i;
        }
        cout << "People who are HODS:\n";
        S[m].display();
        return 0;
    }
}
```

Output -

```
Enter data for 5 people:
Name 1:
Enter name: A
Post: Intern
Name 2:
Enter name: B
Post: Principal
Name 3:
Enter name: C
Post: HR
Name 4:
Enter name: D
Post: HOD
Name 5:
Enter name: E
Post: Intern
People who are HODS
Name: D, Post: HOD
```

29/7/25

Experiment 3

a) WAP to declare class 'book' containing data members as book title, author name and price. Accept and display the information for one object using pointer to that class.

→

```
# include <iostream>
using namespace std;
class Book
{
    string t, a;
    float p;
    public:
    void accept()
    {
        cout << " Enter book title:";
        getline (cin, t);
        cout << " Enter author name:";
        getline (cin, a);
        cout << " Enter price:";
        cin >> p;
    }
    void dis()
    {
        cout << " Book title :" << t;
        cout << " Book author :" << a;
        cout << " Price :" << p;
    }
};
int main()
{
    Book b1;
    Book * ptr = & b1;
    ptr -> accept();
    ptr -> dis();
    return 0;
}
```

Output -

```
Enter book title : ABC
Enter author name : XYZ
Enter price : 500

Book title : ABC
Book author : XYZ
Price : 500
```

b) WAP to declare a class 'student' having data members roll_no and percentage. Using 'this' pointer involve member functions to accept and display the data for one object of the class.

→

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

```
class Student
{
    int r;
    float p;
    public:
    void accept(int r, float p)
    {
        this -> r = r;
        this -> p = p;
    }
    void dis()
    {
        cout << " Roll no : " << r << endl;
        cout << " Percentage : " << p << "%d " << endl;
    }
};
int main()
{
    Student s;
    int ro;
    int pe;
    cout << " Enter roll no : ";
    cin >> ro;
    cout << " Enter percentage : ";
    cin >> pe;
    s.accept(ro, pe);
    s.dis();
    return 0;
}
```

Output -

```
Enter roll no : 33
Enter percentage : 92.2
Roll no : 33
Percentage : 92.2 %
```

c> WAP to demonstrate use of nested class

→

```
# include <iostream>
using namespace std;
class Number
{
    private;
    int num;
    class Operations
    {
        public:
        int square (int n)
            return n*n;
        int cube (int n)
            return n*n*n;
    };
    public;
        void accept()
        {
            cout << " Enter a number : "
            cin >> num;
        }
```

```
void display()
{
    Operations op;
    cout << " square of " << num <<
    " = " << op.square(num) <<
    endl;
    cout << " cube of " << num <<
    " = " << op.cube(num) << endl;
}
};
int main()
{
    Number n1;
    n1.accept();
    n1.display();
    return 0;
}
```

Output -

```
Enter a number 3
Square of 3 = 9
Cube of 3 = 27
```

24/7/25

Experiment 4

a) Swap 2 nos using same class, obj as fun^n argument

→

```
#include <iostream>
using namespace std;
class Number
{
    int num;
    public:
    void setData(int n)
    {
        num = n;
    }
    void display()
    {
        cout << " Number : " << num << endl;
    }
    void swap(Number &obj)
    {
        int t = num;
        num = obj.num;
        obj.num = t;
    }
};
int main()
{
    Number n1, n2;
```

```
n1.setData(6);
n2.setData(20);
cout << "Swap.";
n1.swap(n2);
n1.display();
n2.display();
return 0;
}
```

Output -

```
Number : 10
Number : 20
Swap.
Number : 20
Number : 10
```

b) Swap 2 nos from same class using friend function

→

```
#include <iostream>
using namespace std;
class Number
{
    int num;
    public:
    void input(int n)
    {
        num = n;
    }
    void dis()
    {
        cout << "Number : " << num;
    }
    friend void swap(Number &, Number &);
};
void swap(Number &a, Number &b)
{
    int t = a.num;
    a.num = b.num;
    b.num = t;
}
int main()
{
    Number n1, n2;
    n1.input(43);
    n2.input(67);
    swap(n1, n2);
    cout << "Swap";
    n1.display();
    n2.display();
    return 0;
}
```

```
Number : 43
Number : 67
Swap
Number : 67
Number : 43
```

Date / /

c) Swap 2 nos from different class using friend function

→

```
# include <iostream>
using namespace std;
class B;
Class A
{
    int num;
    public :
    void input (int n)
    {
        num = n;
    }
    void dis()
    {
        cout << " A Number : " << num;
    }
    friend void swap ( A &, B&) ;
};
class B
{
    int num;
    public :
    void input ( int n)
    {
        num = n;
    }
    void dis()
    {
        cout << " B Number : " << num;
    }
    friend void swap (A &, B&) ;
};
void swap ( A &a , B &b)
{
    int t = a.num;
    a.num = b.num;
    b.num = t;
}
Int main()
{
    A obj1;
    B obj2;
    obj1.input(65);
    obj2.input (9);
    swap (obj1, obj2)
    cout << "Swap " ;
    obj1.dis();
    obj2.dis();
    return 0;
}
```

Output -

A Number : 65
B Number : 9
A Number : 9
B Number : 65

d) WAP to make 2 classes for result and find average of results

→
```
# include <iostream>
using namespace std;
class R2;
class R1
{
    float m1;
    public:
    void input()
    {
        cout << "Enter marks";
        cin >> m1;
    }
    friend float avg(R1, R2);
};
class R2
{
    float m2;
    void input()
    {
        cout << "Enter marks";
        cin >> m2;
    }
    friend float avg(R1, R2);
};
float avg(R1 r1, R2 r2)
{
    return(r1.m1 + r2.m2)/2;
}
int main()
{
    R1 obj1;
    R2 obj2;
    obj1.input();
    obj2.input();
    float avgr = avg(obj1, obj2);
    cout << "Average:" << avg;
    return 0;
}
```

Output -

Enter marks : 78
Enter marks : 89
Average marks : 83.5

e) WAP to find greatest number among two numbers from two different classes using friend function

→
```
# include <iostream>
using namespace std;
class C2;
class C1
{
```

```
    int num1;
    public:
    void input(int n)
    {
        num1 = n;
    }
    friend void G(C1, C2);
};
class C2
{
    int num2;
    public:
    void input(int n)
    {
        num2 = n;
    }
    friend void G(C1, C2);
};
void G(C1 c1, C2 c2)
{
    if (c1.num1 > c2.num2)
        cout << c1.num1;
    else
        cout << c2.num2;
}
int main()
{
    C1 obj1;
    C2 obj2;
    obj1.input(371);
    obj2.input(99);
    G(obj1, obj2);
    return 0;
}
```

Output -

371

Experiment 5

a) WAP to find the sum of nos between 1 to n using a constructor where the value of n will be passed to the constructor.

i) Default constructor

```
# include <iostream>
using namespace std;
class A
{
    int n, s;
    public :
    A()
    {
        n = 5 ;
        s = 0;
        for ( int i = 1 ; i <= n ; i++)
            s += i ;
        cout << "Sum = "<< s ;
    }
};
int main()
{
    A a1;
    return 0;
}
```

Output - Sum = 15

ii) Parameterised constructor

```
# include <iostream>
using namespace std;
class A
{
    int n,s;
    public:
    A ( int x)
    {
        n = x;
        s = 0;
        for ( int i= 1 ; i<= n ; i++)
            s+ = i;
        cout << " Sum = " << s;
    }
};
int main()
{
    A a1(10);
    return 0;
}
```

Output -

Sum = 55

iii) Copy constructor

```
# include <iostream>
using namespace std;
class A
{
    int n, s;
    public:
    A ( int x)
    {
        n = x;
        s = 0;
        for ( int i = 1 ; i<= n ; i++)
            s + = i;
    }
    A ( A & a)
    {
        n = a.n;
        s = a.s;
        cout << " Sum = " << s;
    }
};
int main()
{
    A a1 (6);
    A a2 (a1);
}
```

Output -

Sum = 21

b)

```
# include <iostream>
using namespace std;
class Student
{
    string n;
    float p;
    public:
    Student ( string x , float y)
    {
        n = x;
        p = y;
    }
    void show()
    {
        cout << " Name = " << n << " Percent = " << p <<
        endl;
    }
};
int main()
{
    Student s (" Arya", 92.2);
    s.show();
}
```

Output -

Name = Arya Percent = 92.2

c) Default Constructor -

```
# Include <iostream>
using namespace std;
class College
{
    int r;
    string n, c;
    public:
    College ( int x , string y, string z = " CE
    {
        r = x;
        n = y;
        c = z;
    }
    void dis()
    {
        cout << " Roll : " << r << "Name = " << n <
        " Course : " << c << endl;
    }
};
int main()
{
    College a ( 1, " Arya"), b (2, " Shreya
    a.dis();
    b.dis();
}
```

Roll = 1 Name = Arya Course = CSE
Roll = 2 Name = Shreya Course = CSE

d) Constructor Overloading -

```
# include <iostream>
using namespace std;
class Box
{
    int l, b, h;
    public :
    Box()
    {
        l = b = h = 1;
    }
    Box (int x )
    {
        l = b = h = x ;
    }
    Box ( int x , int y , int z )
    {
        l = x ;
        b = y ;
        h = z ;
    }
void dis ()
```

```
int main()
{
    Box a , b(3) , c(2,3,4) ;
    a . dis();
    b . dis();
    c . dis();
}
```

Output -

Volume = 1
Volume = 27
Volume = 24

17/10

Experiment 6

a) Multi-level Inheritance:

```
#include <iostream>
using namespace std;
class A
{
    public:
    int x;
    void get()
    {
        x = 5;
    }
};
class B : public A
{
    public:
    int y;
    void set()
    {
        y = 10;
    }
};
class C : public B
{
    public:
    void show()
    {
        cout << "Sum : " << x + y;
    }
};
int main()
{
    C c1;
    c1.get();
    c1.set();
    c1.show();
}
```

Output -
Sum = 15

b) Multiple Inheritance -

```
#include <iostream>
using namespace std;
class A
{
    public:
    int x;
    void geta()
    {
        x = 3;
    }
};
class B
{
```

```
    public:
    int y;
    void gety()
    {
        y = 4;
    }
};
class C : public A, public B
{
    public:
    void show()
    {
        cout << "Mul :" << x * y;
    }
};
int main()
{
    C c1;
    c1.getx();
    c1.gety();
    c1.show();
}
```

Output -

Mul = 12

c) Hierarchical Inheritance

```
# include <iostream>
using namespace std;
class A
{
    protected:
    int n;
    public:
    void get()
    {
        n = 5;
    }
};
class B : private A
{
    public:
    void sq()
    {
        get();
        cout << "Square =" << n * n;
    }
};
class C : protected A
{
    public:
    void cube()
    {
        get();
```

```
        cout << " Cube = " << n*n*n ;
    }
};
int main()
{
    B x, C y;
    x.sq();
    y.cube();
}
```

Output -

Square = 25
Cube = 125

d) Hybrid -

```
# include <iostream>
using namespace std;
class A
{
    protected :
    int a;
    public :
    void geta()
    {
        a = 2;
    }
};
class B : protected A
{
    protected :
    int b;
    public :
    void getb()
    {
        b = 3;
    }
};
class C : protected A
{
    protected :
    int c;
    public :
    void getc()
    {
        c = 4;
    }
};
class D : protected B, protected C
{
    public :
    void show()
    {
        geta();
        getb();
        getc();
```

```
        cout << " Sum = " << a+b+c ;
    }
};
int main()
{
    D d1 ;
    d1. show();
}
```

Output -

Sum = 9

c) Virtual Base class -

```
# include <iostream>
using namespace std ;
class A
{
    protected :
    int a ;
    public :
    void geta()
    {
        a = 10 ;
    }
};
class B : virtual protected A
{
    protected :
    int b ;
    public :
    void getb()
    {
        b = 20 ;
    }
};
class C : virtual protected A
{
    protected :
    int c ;
    public :
    void getc()
    {
        c = 30 ;
    }
};
class D : protected B , protected C
{
    public :
    void show()
    {
        geta() ;
        getb() ;
        getc() ;
```

```
        cout << "Sum = " << a+b+c;
    }
};
int main()
{
    D d1;
    d1.show();
}
```

Output -
Sum = 60

17/10

Experiment 7

a) Function overloading to calculate the area of a laboratory and classroom.

→
```
#include <iostream>
using namespace std;
class Area
{
    public:
    int cal (int l, int b)
    {
        return l * b;
    }
    int cal (int a)
    {
        return a * a;
    }
};
int main()
{
    Area a1;
    cout << "Lab = " << a1.cal(5,4);
    cout << "Class = " << a1.cal(6);
}
```

Output -
Lab = 20

b) Function overloading to cal sum of 5 float values & sum of 10 int values

→
```
# include <iostream>
using namespace std;
class Sum
{
    public:
    float add( float a, float b, float c, float d,
    float e)
    {
        return a+b+c+d+e;
    }
    int add ( int a1, int a2, int a3, int a4, int a5,
            int a6, int a7, int a8, int a9, int a10)
    {
        return a1+a2+a3+a4+a5+a6+a7+a8+
        a9 + a10;
    }
};
int main()
{
    Sum s1;
    cout << " Float = " << s1. add (1.1, 2.2, 3.3, 4.4,
    5.5);
    cout << " Int = " << s1. add (1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
}
```

Output -
Float = 16.5
Int = 55

c) Unary (-) operator -

```
# include <iostream>
using namespace std;
class Num
{
    int n;
    public:
    Num (int x)
    {
        n = x;
    }
    void operator - ()
    {
        n = -n;
    }
    void show()
    {
        cout << " Val = " << n;
    }
};
int main()
{
    Num a(10);
    -a;
    a. show();
```

Output -
Val = -10

d) Unary (++) operator -

```
# include <iostream>
using namespace std;
class Num
{
    int n;
    public:
    Num (int x)
    {
        n = x;
    }
    void operator ++()
    {
        ++n;
    }
    void operator ++(int)
    {
        n++;
    }
    void show()
    {
        cout << "Val :" << n;
    }
};
```

```
int main()
{
    Num a(5);
    ++a;
    a.show();
    a++;
    a.show();
}
```

Output -
Val = 6
Val = 7

17/10

Experiment 8

a) Overload '+' operator to concatenate two strings

→
```
# include <iostream>
using namespace std;
# include <string>
class Str
{
    string s;
    public:
    str ( string x )
    {
        s = x;
    }
    Str operator + (Str t)
    {
        return Str ( s + t.s);
    }
    void show()
    {
        cout << s;
    }
};
int main()
{
    Str a("xyz") b(" pqr").
    Str c = a+b;
    c.show();
}
```

Output -

xyzpqr

b) Run - time Polymorphism -

```
# include <iostream>
# include <string>
using namespace std;
class B
{
    public:
    virtual void d()
    {
        cout << " Base call ";
    }
    virtual ~B()
    {
    }
};
class D1 : public B
{
    public:
    void d() override
```

```
        cout << " Login details ";
    }
};
class D2 : public B
{
    public:
    void d() override
    {
        cout << " Membership details ";
    }
};
int main()
{
    B * p;
    D1 d1;
    D2 d2;
    p = &d1;
    cout << " Display 1: ";
    p -> d();
    p = &d2;
    cout << " Display 2: ";
    p -> d();
    return 0;
}
```

Output -

```
Display 1: Login details
Display 2: Membership details
```

17/10

Experiment 9

a) Copy contents of one file to another.

→
```
# include <fstream>
using namespace std;
int main()
{
    if stream in ("First.txt", ios::in);
    of stream out ("Second.txt", ios::out);
    string s;
    while (getline (in, s))
        out << s << '\n';
    in.close();
    out.close();
    return 0;
}
```

Output -

No visible output.

b) Count digits and spaces using file handling

→
```
# include <fstream>
# include <iostream>
using namespace std;
int main()
{
    if stream f ("First.txt");
    char ch;
    int d = 0, sp = 0;
    while (f.get(ch))
    {
        if (is digit (ch))
            d++;
        if (ch == ' ')
            sp++;
    }
    cout << "Digits" << d << "Spaces" << sp;
    f.close();
    return 0;
}
```

Output -

Digits 0 Spaces 0

c) Count words using file Handling

```
# include <iostream>
# include <fstream>
# include <sstream>
using namespace std;
int main()
{
    ifstream f("First.txt");
    string s, w;
    int c=0;
    while ( getline(f,s))
    {
        stringstream ss(s);
        while (ss >> w)
            c++;
    }
    f.close();
    cout << " Words = " <<c;
}
```

Output -

```
Words = 12
```

d) Count occurrence of a given word.

```
# include <iostream>
# include <fstream>
# include <sstream>
using namespace std;
int main()
{
    ifstream f(" First.txt ");
    string s, w, t ;
    int c=0;
    cout << " Enter the word : " ;
    cin >> t ;
    while ( getline (f,s))
    {
        stringstream ss (s);
        while (ss >>w)
            if ( w == t)
                c++;
    }
    f.close();
    cout << " Occurrences = "<<c;
}
```

Output -

```
Enter the word : Hello
Occurrences = 3
```

17/10

Experiment 10

a) Find sum of array elements using function template

→

```
# include <iostream>
using namespace std;
template < class T > T sum ( T a[], int n)
{
    T s = 0;
    for ( int i = 0 ; i<n ; i++)
    {
        s += a[i];
    }
    return s;
}
int main()
{
    int ai[5] = { 1, 2, 3, 4, 5 };
    float af[5] = { 1.1, 2.2, 3.3, 4.4, 5.5 };
    double ad[5] = { 1.11, 2.22, 3.33, 4.44, 5.55 };
    cout << " Int sum = " << sum ( ai, 5) <<
    " In Float sum = " << sum (af, 5) << " Doubl
    sum = " << sum (ad, 5);
}
```

Output -

Int sum = 15
Float sum = 16.5

b) Square function using function specialization

→

```
# include <iostream>
using namespace std;
template < class T > T sq ( T x)
{
    return x * x ;
}
template < > string sq ( string s )
{
    return s+s ;
}
int main ()
{
    cout << sq (5) << endl;
    cout << sq ( string ("Hi")),
}
```

Output -

25
HiHi

c) Simple Calculator using class template

→

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

```
using namespace std;
template < class T >
class calculate
{
    T a, b;
    public:
    calculate ( T x, T y)
    {
        a = x;
        b = y;
    }
    void opp ()
    {
        cout<< "Sum : "<< a+b;
        cout<< "Sub : "<< a-b;
        cout<< "Mul : "<< a*b;
        cout<< "Div : "<< a/b;
        cout<< "Rem : "<< (int) a %(int) b;
        cout<< " sin(a) = "<< sin(a)<< " sin(b) =
        sin(b);
        cout<< " cos(a) = "<< cos(a)<< " cos(b) =
        cos(b);
        cout<< " min = "<< (a<b ? a : b);
        cout<< " max = "<< (a>b ? a : b);
        cout<< " Avg = "<< (a+b)/2.0;
    }
};
```

```
int main()
{
    int ch;
    double x, y;
    cout << " Enter nos : ";
    cin >> x >> y;
    calculate < double > c(x,y);
    cout << " 1. Sum, 2. Sub, 3. Mul, 4. Div,
    5. Rem, 6. Sin, 7. Cos, 8. Min, 9. Max,
    10. Avg ");
    cin >> ch;
    switch (ch)
    {
        case 1 :
            c.
        return 0;
}
```

Output -

```
Enter nos : 2 4
Sum : 6
Sub : 2
Mul : 8
Div : 2
Rem : 0
Sin(a) = 0.9   Sin(b) = -0.7
Cos(a) = -0.4  Cos(b) = -0.6
Min = 2
Max = 4
```

d) Stack using class template

→

```
# include <iostream>
using namespace std;
template <typename T>
class stack
{
    T * a;
    int t;
    int s;
    public:
    Stack ( int size)
    {
        s = size;
        a = new T[s];
        t = -1;
    }
    ~ Stack ()
    {
        delete[] a;
    }
    isEmpty ()
    {
        return t == -1;
    }
    isFull ()
    {
        return t == s-1;
    }
    T pop ()
    {
        if ( isEmpty ())
            return T();
        return a[t];
    }
    void push (T val)
    {
        if ( isFull() )
            return T();
        return a[t--];
    }
};
int main()
    void check ()
    {
        if ( isEmpty())
            return T();
        return a[t];
    }
};
int main()
{
    stack <int> s(5);
    s.push (10);
    s.push (20);
    s.push (30);
    cout << s.pop();
    cout << s.check();
    return 0;
}
```

Output -

30

20

DOMS | Page No. | Date / /

Experiment 11

a) WAP to implement generic vectors. Include following member functions.

i) To modify the value of a given element
ii) to multiply by scalar value.
iii) To display the vector in form (10,20,30)

→

```
# include <iostream>
using namespace std;
template <typename T>
class Vec
{
    T * v;
    int size;
    public:
    Vec ( int n) : size(n)
    {
        v = new T [size];
    }
    void set ( int idx, T val)
    {
        v[idx] = val;
    }
    void mue()
    {
        for ( int i = 0 ; i < size ; i++)
        v[i] *= 2;
    }
    void display()
    {
        cout << "( ";
        for (int i = 0 ; i < size ; ++i)
        {
            cout << v[i];
            if ( i != size - 1)
                cout << ", ";
        }
        cout << ")" << endl;
    }
};
int main()
{
    int n;
    cout << " Enter no of elements : ";
    cin >> n;
    Vec <int> v(n);
    for ( int i = 0; i < n; i++)
    {
        cout << " Enter value for element "<<
        i + 1 << ":";
        int val;
        cin >> val;
        v.set ( i, val);
```

```
v.display();
int idx, newval;
cout << " Enter index to modify "<<n-1<<"):";
cin >> idx;
cout<< " Enter new value : "<<idx;
cin >> newVal;
v.set (idx, newVal);
cout<< " Modified vector :";
v.display();
v.mul();
cout << " Vector after multi :";
v.display();
return 0;
}
```

Output -

```
Enter no of elements : 3
Enter value of element 1 : 10
Enter value of element 2 : 20
Enter value of element 3 : 30
(10, 20, 30)
Enter to modify : 3
Enter new value : 40
Modified vector: ( 10,20, 40)
Vector after multiplication : (20, 40, 80)
```

b) Using iterators:

→

```
# include <iostream>
# include <vector>
using namespace std;
template <class T>
class V
{
    vector<T> v;
    public:
    V (int n)
    {
        v = vector<T>(n);
    }
    void set (int i, T x){
    {
        v[i] = x;
    }
    void modify( int i, T x)
    {
        v[i] = x;
    }
    void mult( T s)
    {
        for ( typename vector<T>::
        iterator i = v.begin(); i != v.end();
        ++i)
```

```
    void disp()
    {
        for( typename vector<T>:: iterator
        i = v.begin(); i != v.end(); ++i)
        {
            cout << *i;
            if ( i+1 != v.end())
                cout << " ";
        }
        cout << "\n";
    }
};
int main()
{
    v<int> a(3);
    a.set (0, 10);
    a.set (1, 20);
    a.set (2, 30);
    a.disp();
    a.modify(1, 25);
    a.disp();
    a.mult(2);
    a.disp();
}
```

Output -

```
10 20 30
10 25 30
20 50 60
```

Experiment 12

a) Implement Stack -

```
# include <iostream>
# include <stack>
using namespace std;
int main()
{
    stack<int> s;
    s.push(10);
    s.push(20);
    s.push(30);
    cout << "Stack : ";
    while ( !s.empty())
    {
        cout << s.top();
        s.pop();
    }
}
```

Output -

```
Stack : 30 20 10
```

b) Implement Queue

→
```
# include <iostream>
# include <queue>
using namespace std;
int main()
{
    queue <int> q;
    q.push(10);
    q.push(20);
    q.push(30);
    cout << "Queue :";
    while ( !q.empty())
    {
        cout << q.front();
        q.pop();
    }
}
```

Output -

Queue : 10 20 30