M T W T F S S

1) Write a program for addition of 2 numbers.

```
#include<iostream>
using namespace std;
int main() {
 int num1, num2, num3 sum;
 cout <<"Enter first number:";
 cin >> num1;
 cout << "Enter second number:";
 cin >> num2;
 sum = num1 + num2;
 cout <<"The sum is:" << sum << endl;
 return 0;
      }
```

2) Write a c++ program to perform arithmatic operations using switch case.

```
#include <iostream>
using namespace std;
int main () {
int num1, num2, result;
 char op;
cout <<"Enter first no.:";
 cin >> num1;
 cout << Enter second no.";
 cin >> num2;
 cout << "Enter second operator (+,-,*,/)";
```

```
cin >> op;
switch (op) {
case '+': result = num1 + num2;
          cout << "Result :" << result << endl;
          break;
case '-': result = num1 - num2;
          cout << "Result :" << result << endl;
          break;
case '*': result = num1 * num2;
          cout << "Result :" << result << endl;
          break;
case '/': result = num1 / num2;
          cout << "Result :" << result << endl;
    }
return 0;
    }
```

8) write a program to check if no is even/odd

```
#include <iostream>
using namespace std;
int main() {
int num;
cout << "Enter a number:";
cin >> num;
if (num % 2 == 0) {
cout << num << " is even." << endl;
  } else {
cout << num << " is odd." << endl;
  }
return 0;
```

4) Write a program to print 1-10 no.s usir for loop

```
# include <iostream>
using namespace std;
int main () {
for (int i = 1; i <= 10; i++) {
  cout << i << " ";
    }
cout << endl;
return 0;
  }
```

5) Write a code to print 10-1 nos. using while loop.

```
# include <iostream>
using namespace std;
int main() {
 int i = 10;
while (i >= 1) {
cout << i << " ";
  i--;
  }
cout << endl;
 return 0;
    }
```

~~Writ~~

6)
```
#include <iostream>
using namespace std;
int main () {
 cout << "Pattern a:" << endl;
 for (int i = 1; i <= 5; i++) {
  for (int j = 1; j <= i; j++) {
      }
   cout << endl;
     }
 cout << endl;
 cout << "patern b:" << endl;
 for (int i = 1; i <= 5; i++) {
 for (int j = 1; j <= i; j++) {
   cout << i << " ";
     }
   cout << endl;
      }
 cout << "Pattern c:" << endl;
  for (int i = 1; i <= 5; i++) {
  for (int j = 1; j <= i; j++) {
   cout << endl;
     }
   return 0;
     }
```

8/7/25

```
*                1                1
* *              2 2              1 2
* * *            3 3 3            1 2 3
* * * *          4 4 4 4          1 2 3 4
* * * * *        5 5 5 5 5        1 2 3 4 5
```

15/07/25

EXPERIMENT - 1

Page No.
Date

1) WAP to declare a class student having data members as name, roll no. accept & display data for one student.

Code:

```
# include <iostream>
using namespace std;
class student {
 string name;
 int roll-no;
  public:
void accept ();
void display ();
  };
void student :: accept () {
cout <<"Enter name and roll no.-";
cin >> name>> roll-no;
      }
void student :: display ()
   {
 cout << "\n student name;" << name;
 cout << "\n student roll no: " <<roll-no;
     }
int main () {
student SI;
SI accept ();
SI display ();
  return 0;
        }
```

OUTPUT:

Enter name and roll no - abc
23
student name: abc
student roll no: 23

3) WAP to declare a class ~~book~~ time having data members as ~~id, name, price~~ H, M, S. Accept data for ~~2 books~~ one object & display data of ~~book having greater price~~ total time in seconds.

Code:

```
# include <iostream>
using namespace std;
class time {
int H, M, S;
public:
void accept ();
void displayseconds ();
    };
void Time :: accept () {
cout << "\n Enter hours :";
cin >> H;
cout << " Enter minutes:";
 cin >> M;
cout << " Enter seconds:";
 cin >> S;
     }
void Time: displaySeconds () {
```

```
int total = (H * 3600) + (M * 60) + S;
cout << "\n Total time in seconds:" << total <<
endl;
    }
int main () {
time t1;
cout << "Enter time details:";
t1.accept();
t1.displayseconds();
return 0;
    }
```

OUTPUT:

Enter time details :
Enter hours: 2
Enter minutes: 10
Enter seconds: 45
Total time in seconds: 7845

2) WAP to declare a class book. having data members as ~~id~~, name, price. Accept data for 2 books & display data of book having greater price.

Code:

```
# include <iostream>
using namespace std;
class book {
int id;
string name;
float price;
public:
```

```
void accept ();
void display ();
float get price ();
  };
void book :: accept () {
cout << "\n Enter book ID: ";
cin >> id;
cout << "Enter book name :;
cin >> name;
cout << "Enter Price : ;
 cin >> price;
    }
void book :: display () {
c out << "\n Book ID : " << Pd << endl;
c out << "Price: ;" << price << endl;
   }
float book :: get price () {
  return price;
    }
int main () {
book b1, b2 ;
cout << "Enter details for book 1:;
 b1. accept ();
cout "\n Enter details for book 2;;
 b2. accept ();
cout << "\n Book with greater price is :";
if (b1. get price () >
b2. get price ()) {
 b1. display ():
   } else {
   b2. display (): }
     return 0; }
```

OUTPUT:

```
Enter details for book1 :
Enter book name : Twilight
Enter price : 320
Enter details for book 2:
Enter book ID : 12640
Enter book name : Divergent
Enter book price : 460
Book with greater price is:
Book ID: 12640
Book name: Divergent
Price: 460.
```

28/7/25

Experiment 2

1) WAP to declare a class 'city' having data members as name and population. Accept this data for 5 cities and display name of city having highest population.

Code:

```
#include <iostream>
using namespace std;
class city {
public:
 string name;
  int population;
    };
int main () {
city cities[5];
string max city;
 int maxPop = 0;
for (int i = 0; i<5; i++) {
cout << "Enter name of city " << i+1 << ": ";
cin >> cities[i].name;
cout << " Enter population: ";
 cin >> cities[i].population;
if (cities[i].population > maxPop) { {
maxPop = cities[i].population;
max city = cities[i].name;
  }
    }
cout << " city with highest population: " <<
max city << endl;
    return 0;
              }
```

OUTPUT:

```
Enter name of city 1: Pune
Enter population: 30000
Enter name of city 2: Nagpur
Enter population: 40000
Enter name of city 3: Mumbai
Enter population: 70000
Enter name of city 4: Hyderabad
Enter population: 60000
Enter name of city 5: Gujarat
Enter population: 45000
City with highest population: Mumbai
```

2) WAP to declare 'Account' having data members as account no. and balance. Accept this data for 10 accounts and give interest 10% where balance is equal or greater than 5000 & display them.

Code:

```
#include<iostream>
using namespace std;
class account {
public:
int accNO;
float balance;
    };
int main() {
```

```
Account acc[10];
for(int i=0; i<10; i++) {
cout<<"\n Enter details for Account"<<
i+1<<" "<<endl;
cout<<" Account no:";
cin>>acc[i].acc No;
cout<<" Balance :";
if (acc[i].balance>=5000) {
acc[i].balance += acc[i].balance*0.10;
        }
          }
cout<<"\n Accounts with Balance>=5000
(after 10% interest) \n";
for (int i=0; i<10; i++) {
cout<<"Account"<<i+1<<"Account No:"
acc[i].ace No
<<", Updated Balance:"<<acc[i].balance
<<endl;
     }
       }
return 0;
  }
```

OUTPUT:

Enter details for Account 1

Account no: 24567

Balance: 1212

Enter details for Account 2

Enter details for Account 3
Account no: 3456
Balance: 2121
Enter details for account 4
Account no: 76645
Balance: 5555
Enter details for account 5
Account no: 4356
Balance: 6666
Enter details for account 6
Account no: 88775
Balance: 7777
Enter details for account 7
Account no: 3635
Balance: 8888
Enter details for account 8
Account no: 454593
Balance: 9999
Enter details for account 9
Account no: 65778
Balance: 1424
Enter details for account 10
Account no: 65778
Balance: 1424
Accounts with Balance * (after 10% interest) =
Account 4 - account no: 76645, updated balance:
6110.5
Account 5 - account no: 4356, updated balance:
7332.6

Account 6 - account no: 88775. updated
balance : 8854.7
Account 7 - account no: 3635, updated
balance : 9776.8
Account 8 - Account no: 454543, updated
balance : 10998.9

3)
```
#include<iostream>
#include<cstring>
using namespace std;
class staff {
public:
char name[50];
char post[20];
    };
int main() {
staff s[5];
for(int i=0;i<5;i++) {
cout<<"Enter name of staff "<<i+1<<": ";
cin>>s[i].name;
cout<<"Enter post: ";
cin>>s[i].post;
    }
cout<<"staff with post 'HOD': \n";
for(int i=0; i<5; i++) {
if(strcmp(s[i].post,"HOD")==0) {
  cout<<s[i].name<<endl;
      }
        }
          }
return 0; }
```

OUTPUT:

```
Enter name of staff 1: Shruti
Enter post : HOD
Enter name of staff 2: Jaanvi
Enter post: Lecturer
Enter name of staff 3: Gayatry
Enter post: Dean
Enter name of staff 4: Vishal
Enter post: Principle
Enter name of staff 5: Amish
Enter post: HOD
Staff with Post 'HOD':
Shruti
Amish
```

25/7/25

1) Write a program to declare a class 'book' containing data members as book-title, author price.

Soln Code:

```
#include<iostream>
using namespace std;
class book
  { {
Public:
 string book-title;
 int price;
  string author_name;
 void accept ()
  {
cout << " enter book name : ";
cin >> book_title;
cout << " enter name of author: ";
cin >> author_name;
cout << " Price of the book : "&;
cin >> Price;
        }
void display()
   {
cout << " Book name is : ", book_title;
cout << " In author name : " << author_name;
cout << " In price is : " << price;
      }
        };
```

```
int main ()
 {
book b1;
 book p;
b1.accept();
b1.display();
 return 0;
    }
```

O/P:

```
Enter book name: Divergent
Enter author name: Anna
Enter price of the book: 799
Book name is: Divergent
Author name: Anna
Price is: 799
```

2) WAP to ~~draft~~ declare a class 'student' having data members roll_no using 'this' pointer invoke member functions to accept & display this data for one object of the class.

Code:

```
# include <iostream>
using namespace std;
class student
  {
int roll_no;
float percentage;
```

```
Public:
void accept ()
  {
cout <<"Enter student roll no:";
cin>> this-> roll-no;
cout <<"Enter student percentage:";
cin>> this -> percentage;
     }
void display ()
  {
cout <<"Roll number of student is: << rollno;
cout <<" Roll \n percentage is: << percentage;
     }
        };
int main ()
  {
student S1;
  Student * P;
   P = & S1;
S1. accept ();
S1. display ();
  return 0;
        }
```

O/P:

Enter the student roll number: 38
Enter the student percentage: 96%
Roll number of student is: 38
Percentage is: 96%

3) WAP to demonstrate use of nested class.

Code:

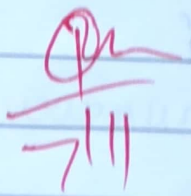
```
#include <iostream>
using namespace std;
class number
  {
Private :
 int num;
class operations
   {
Public :
int square (int n)
 {
return n*n;
    }
int cube (int n)
 {
return n*n*n;
    }
      };
 public:
void accept ()
   {
cout << "Enter an number:";
cin>> num;
    }
```

```
void display ()
  {

cout <<"Square of :" <<"=" << op.square
(num) << end l;
cout <<"cube of :" << = " << op. cube (num)
<< end l;
      }
        };
int main()
  {
Number n1;
n1. accept ();
n1. display ();
 return 0;
    }
```

O/P:

Enter an number : 2
Square of : 4
Cube of : 8



Q1) WAP to swap two numbers from same class using object as function argument. Write swap functions as member functions

Code:

```
#include<iostream>
using namespace std;
class number {
int a, b;
public:
void input () { {
cout << " enter two numbers : ";
cin >> a >> b;
    }
void display () {
cout <<"a:" << a << " b:" << b << endl;
      }
void swapnumbers (Number& obj) {
int temp = obj.a;
 obj.a = obj.b;
obj.b = temp;
        }
          };
int main ()  {
number n1;
 n1. input ();
cout << " Before swapping : ";
n1. display ();
n1. swapnumbers(n1);
cout << "After swaping.:";
```

```
n1.display();
return 0;
}
```

O/P:

Enter two numbers: 16 13
Before swapping: a=16, b=13
after swapping: a=13, b=16

Q2) WAP to swap two numbers from same class using concept of friend function.

Soln code:

```
#include <iostream>
using namespace std;
class Number {
int a, b;
Public:
void input () {
cout << " Enter two numbers:";
cin >> a >> b;
}
void display () {
cout << "a:" << a << "b:" << b << endl;
}
friend void swapnumbers (number & obj);
};
void swapnumbers (Number & obj) {
int temp = obj.a;
obj.a = obj.b;
obj.b = temp;
```

```
}
int main () {
number n1;
n1. input ();
cout << " before swapping :";
n1. display ();
swapnumbers (n1);
cout << "After swapping :";
n1. display ();
return 0;
}
```

O/P:

Enter two numbers: 1 4

Before swapping : a=1 , b=4

After swaping: a=4 , b=1

Page No.:
Date:

1) Sum from 1 to n using constructor

Code:

```
#include <iostream>
using namespace std;
class sum {
int n, total;
public:
sum(int num)
n = num;
total = 0;
for(int i=1; i<=n; i++)
 total += i;
      }
void display () {
cout << "Sum from 1 to " << n << " = " << total <<
endl;
      }
          };
int main () {
 int n;
cout << "Enter value of n: ";
cin >> n;
Sum s(n);
s.display();
  return 0;
      }
```

O/P:

```
Enter value of n: 32
Sum from 1 to 32 = 528
```

b) Create a class student with data members name and percentage, use a constructor to initialize them & display student details.

Code:

```
#include <iostream>
using namespace std;
class student {
string name;
float percentage;
public:
student (string n, float p) {
name = n;
percentage = p;
    }
void display () {
cout << "Name:" << Name << end l;
cout << "Percentage:" << percentage << "%." << endl;
        }
    };
int main () {
string name;
float per;
cout << "Enter name:";
cin >> name;
cout << "Enter percentage:";
cin >> per;
student s1 (name, per);
S1. display ();
  return 0;
            }
```

Engineering
Roll No: 2, Name: Sagar, course: Computer Engineering.

d) constructor overloading

Code:

```
#include <iostream>
using namespace std;
class Rectangle {
int length, breadth;
public:
Rectangle () {length = breadth = 0;}
Rectangle (int l) {length = breadth = l;}
Rectangle (int l, int b) {length = l; breadth = b;}
void area () {cout << "Area:" << length * breadth
<< endl;}
    };
int main () {
Rectangle r1, r2(5), r3 (4, 6);
  r1. area ();
r2. area ();
```

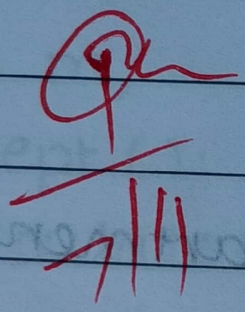
```
r3. area();
return 0;
    }
```

Output:

Area = 0

Area = 25

Area = 24



Experiment - 8

a) Operator overloading

```
#include<iostream>
using namespace std;
class MyString {
 string str;
 public:
void setstring () {
cout<<" Error string: " ;
 cin>> str;
    }
void disp () {
cout<<str<< endl;
    }
MyString operator + (MyString s) {
mystring temp;
temp.str=str+s.str;
   return temp;
        }
            };
int main () {
MyString s1, s2, s3;
cout << " First string ";
 s1. set string ();
cout <<" second string ";
 s2. setstring ();
s3 = s1+s2;
cout <<" concatenated string : ";
   s3. disp ();
   return 0;
        }
```

b. WAP to create a base class Ilogin having data members names, passwords. Declare accept() function virtual. Derieve Email login membership login classes from Ilogin. Display login details of employees.

Code:

```
#include <iostream>
using namespace std;
class Ilogin {
protected:
string name, password;
public:
virtual void accept (); {
cout << "Enter name:";
cin>> name;
cout << "Enter password:";
    }
virtual void disp() {
cout<<"Name:" << name << "Password:" << password;
        }
            };
class Emaillogin : public Ilogin {
 public:
void accept() override {
cout << "An Email login Details:";
Ilogin :: accept ();
   }
void display() override {
cout << " Email login:";
Ilogin :: display();
      }
         };
```

eklavya
PAGE NO.:
DATE: / /

```
class Membershiplogin : public Ilogin {
public:
void accept() override {
cout<<"Membership login→";
Ilogin:: display();
    }
      };
int main();
Emaillogin e;
Membershiplogin m;
e.accept();
m.accept();
cout<<"Displaying details:";
RETURN 0;
e.display();
m.display();
return 0;
```

O/P:-

```
Email login details:
Enter name: alice
Enter password: alice123
Membership login details:
Enter name: BOb
Enter password: bob123
Displaying login details:
Email login → Name: Alice
password: alice123
Membership login → Name: Bob
password: bob123
```

Experiment - 6.

Page No.:
Date:

Q1) Write a program to implement multilevel inheritence. Assume suitable data.

Code:

```
#include<iostream>
using namespace std;
class department {
protected :
string dname;
  };
class student : protected department {
protected :
 string sname;
 int roll;
   };
class marks : protected student {
int m1, m2, percentage;
public:
void accept() {
cout<<"Enter department:" <<endl;
cin >> dname;
cout<<"Enter name:" << endl;
cin>> cname;
cout<<"Enter marks1:" <<endl;
 cin>> m1;
cout<<"Enter marks 2:" <<endl;
cin>> m2;
```

```
void calculate () {
int per = (m1 + m2) /2
cout << "Department :" << dname << endl;
cout << "Name :" << sname << endl;
cout << "Percentage :" << per << endl;
        }
    };
int main () {
marks m;
m.accept ();
m.calculate ();
 return 0;
    }
```

Output :

```
Enter department: AIDS
Enter name : Shruti
Enter marks 1: 98
Enter marks 2: 91
Department : AIDS
Name : Shruti
Percentage: 92
```

Q2) WAP to implement multiple inheritence, assume suitable data.

Code :

```
# include <iostream>
using namespace std;
class department {
protected:
```

```
string dname;
   };
class student {
protected :
 string sname
   };
class marks : protected department, protected
  student {
int m1, int m2, percentage ;
public :
void accept () {
cout<<" Enter department : " endl;
cin>> dname;
cout <<" Enter name : " << endl;
cin>> sname ;
cout<<" Enter marks 1 : <<end l;
 cin>> m1;
cout <<" Enter marks 2 : << end l;
cin>> m2;
     }
void calculate () {
int per= (m1+m2)/2;
cout<<" Department : "<< dname << endl;
cout<<" Department Name : <<sname <<end l;
cout<<" percentage :"<< per << end l,
        }
          };
 int main {
marks m;
m.accept ();
m.calculate (); }
```

88) WAP to implement hierarchial inheritence

Code:

```
#include <iostream>
using namespace std;
class Person {
protected:
string name;
int age;
Public:
void acceptPer() {
cout<<" Enter name: "<< endl;
cin>>name;
cout<<" Enter age: " endl;
cin>> age;
  }
   };
class student : public Person {
 int roll;
 float per;
  public:
void accept stud() {
cout <<" Name: "<<name <<endl;
cout <<"Age: "<<age <<end l;
cout<<" Rollno: "<< roll<< end l;
cout<<" Percentage: "<<per<< end l;
     }
       };
class staff: public person {
int emp_id;
```

```
string subject;
public :
void accept staff() {
cout<<" Enter employee ID :" << end l;
cin>> emp-id;
cout<<"Enter subject :" << end l;
cin>> subject;
  }
void display staff () {
cout <<"Name :" <<name << end l;
cout <<"Age :" << age << end l;
cout <<" Emp ID :" << emp-id <<end l;
cout <<"subject taught :" <<subject <<end l;
  }
    };
int main () {
Students;
 staff t;
S. accept Per ();
s. accept stud ();
t. accept Per ();
t. accept stud ();
S. display stud ();
t. display staff ();
  return 0;
  }
```

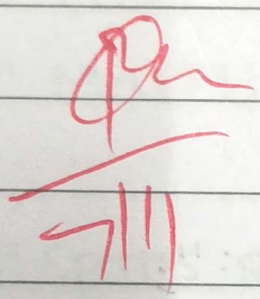
Q4) Write a program to implement hybrid inheritence.

Code:

```
#include <iostream>
using namespace std;
class college {
protected:
string cname;
  };
class Employee: Protected college {
 protected:
string emp_name;
 int id;
    };
 class staff: public Employee {
string name;
int dept ID;
public:
void accept Emp() {
cout<<"Enter clg name:"<< endl;
cin>>cname;
cout<<"Enter emp name:"<<endl;
cin>> emp_name;
cout>>"Enter ID:" endl;
 cin<<id;
cout <<"Enter staff name:"<endl;
 cin>>name;
cout <<"Enter dept id:" <<endl;
 cin>> dept Id;
```

```
void displayEmp () {
cout << "College:" << cname << endl;
cout << "Emp name:" << emp_name << endl;
cout << "ID:" << id << endl;
cout << "Staff Name:" << name << endl;
cout << "Department ID:" << dept-id << endl;
    }
};
class student: Protected college {
string stu-name;
int roll;
public:
void accept stud() {
cout << "Enter college name:" << endl;
cin >> cname;
cout << "Enter name:" << endl;
cin >> stu_name;
cout << "Enter roll no:" << endl;
cin >> roll;
    }
void display-stud() {
cout << "college:" << cname << endl;
cout << "student name:" << stu_name << endl;
cout << "Roll no:" << roll << endl;
    }
int main() {
staff staff1;
staff1. accept Emp();
staff 2. display Emp();
```

```
Student stud1;
stud1.acceptStud();
stud1.displayStud();
return 0;
}
```



e) virtual base class program:

Code:

```
#include <iostream>
using namespace std;
class student {
public:
int roll_no;
void getRoll() {
cout<<" Enter roll number:";
cin>>roll_no;
    }
      };
class Test : virtual public student {
public:
int marks1, marks2;
void getMarks() {
cout<<" Enter marks of two subjects:";
cin>> marks1 >> marks2;
    }
      };
class Sports: virtual public student {
public:
int score;
cout<<" Enter sports score:";
cin>> score;
      }
        };
```

```
class Result : Public Test , public Sports {
public:
void display () {
int total = Public Test, public Sports{ marks1 +
            marks2 + score ;
Public: cout << "\n Roll no:" << roll_no << endl;
cout << " Marks1:" <<marks1 <<", Marks2:" << marks2<<
  endl;
cout<<" Sports score :" << score <<endl;
cout<<"Total =" << total <<endl;
    }
      };
int main () {
Result r;
r.getRoll ();
r.getMarks ();
r.getScore ();
r.display ();
return 0;
      }
```

Experiment- 7

Page No.:
Date:

b)

Q WAP using function overloading to calculating area of laboratory (rectangle) classroom (square)

Code:

```
#include <iostream>
using namespace std;
class area {
public:
void calc (float len, float breadth) {
cout<<"Area of laboratory :"<<len* breadth<<endl;
   }
void calc (float len, float breadth) {
cout<<"Area of classroom:"<< side* side <<endl;
    }
      };
int main() {
area a;
cout<<"Laboratory:"<<endl;
a.calc(12.5, 10.00);
cout<<"classroom:"<<endl;
a.calc (8.0);
return 0;
      }
```

b) WAP using functions overloading to calculate sum of 5 float values sum of 10 interger values

```
#include <iostream>
using namespace std;
class sum {
public:
float sum(float a, float b, float c, float d,
 float e) { return a+b+c+d+e ; }
int sum (int arr[], int n) {
int s=0;
for (int i=0; i<n; i++)
  s+= arr[i];
  returns;
      }
        };
int main() {
sum s;
cout << "Sum of 5 float no. " << s.sum
(1.1, 2.2, 3.3, 4.5, 5.5);
int arr[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
cout << "Sum of 10 integers = " << sum(arr
,10);
return 0;
    }
```

O/P

Sum of 5 float nos = 16.5
Sum of 10 integers = 55

c) WAP to implement unary operator when used with the object so that the numeric data member of class.

Code:

```
#include <iostream>
using namespace std;
class number {
int num;
pubric:
void setdata(int n) { num=n;}
void display() { cout<<"value="<<num<<endl;}
void operator-() { num = - num }
    };
int main() {
number n;
n.setdata(10);
cout<<"original: "; n.display();
return 0;
   }
```

O/P:-

original : value =10

After negation : value = -10

d) WAP to implement the unary ++ operator (for pre & post increment)

```
#include <iostream>
using namespace std;
class count;
 public:
void setdata (int c) {count& = " << count <<endl;
void disp() {cout<< "count = "<< count << end l;
void operator++ (int) {count++;}
    };
int main() {
counter c;
c.setdata(5);
cout << "After pre-increment :"; c.disp();
 (++;
cout<<" original :"; c.disp();
 (++;
cout<< " After post increment :";
 return 0;
      }
```

O/P:

Original : count = 5
After pre increment : count = 6
After post increment: count = 7

Experiment 9

eklavya
PAGE NO :
DATE: / /

a) WAP to copy contents of one file into another. open First.txt in read (ios::in) mode & second .txt file in write (ios::out) mode copy elements of first into second. Assume "First" is already created.

Code:

```
#include <iostream>
# include <fstream>
using namespace std;
int main () {
ifstream fin ("First.txt");
ofstream fout ("second.txt");
if (!Fin) { cout << " First.txt not found"; return 0;}
string line ;{
while (getline (fin, line)) {
fout <<line << endl;
        }
fin. close();
fout. close();
fout <<"contents copied successfully ";
return 0;
   }
```

O/P:-

First.txt not found

b. WAP to count digits and spaces using file handling.

Code:

```
# include <iostream>
# include <fstream>
using namespace std;
int main () {
ifstream fin ("First.txt");
if (!fin) { cout << "File not found! \n" return 0; };
char ch;
int digits = 0, spaces = 0;
while (fin.get (ch)) {
if (isdigit (ch)) digits ++;
if (ch == ' ') spaces++;
    }
fin.close ();
cout << " Digits: " << digits << "spaces " << spaces;
return 0;
```

O/P:-

File not found!

c. WAP to count words using file handling

Code:

```
# include <iostream>
# include <fstream>
# include <sstream>
using namespace std;
int main () {
ifstream fin ("First.txt");
if (!fin) {
```

```
cout << "File not found" ;
return 0;
    }
string word;
int count = 0;
while (fin >> word) {
count++;
  }
fin.close();
cout << "Total words: " << count << endl;
return 0;
    }
```

O/P

File not found!

d) WAP to find occurence of a given word using file handling.

Code:

```
#include <iostream>
#include <fstream>
#include <sstream>
using namespace std;
int main();
ifstream fin("First.txt");
if (!fin) { cout << "File not found!"; return 0; }
int count=0;
cout << "Enter word to search";
cin >> target;
while fin >> word {
if (word == target) count++;
    }
```

```
fin.close();
cout<<"the word "<<target<<"occurred "<<count
<<"times";
return 0;
    }
```

O/P:

File not found.

Experiment 10.

eklavya
PAGE NO.:
DATE: / /

a) WAP to find sum of array elements using function template.

Code :-

```
#include <iostream>
using namespace std;
Template <class T>
T arrsum (T arr, int n) {
T sum = 0;
for (int i = 0; i<n; i++)
 sum += arr[i];
  return sum;
      }
int main () {
int Arr1[5] = {1, 2, 3, 4, 5};
cout<<" Sum of Array: " << arrsum (arr1, 5);
     }
```

O/P:

Sum of array : 15

b) WAP to square function using template specialization. calculate square of integer no. & string. Write a specialized function for string.

Code:

```
#include <iostream>
#include <string>
using namespace std;
template <class T>
T sq (T n) {
 return n * n;
          }
```

```
template <>
string sq<string> (string s) {
return s+s;
    }
int main(); {
int i=5;
String str="ABC";
cout<<"square of integer:" <<sq(i)<<endl;
cout<<"square of string:" <<sq(str)<<endl;
   }
```

O/P:-

Square of Integer : 25
Square of String : ABCABC

c. Code for calculator:

```
#include <iostream>
#include
using namespace std;
template <class T>
class calc { T num1, num2;
Public:
calc (Ta, Tb) {
num1 = a;
num2 = b;
  }
vois disp () {
cout <<" Addition:" << num1 + num2;
cout <<"Subtraction:" << num1 - num2;
cout <<"Multiplication:" << num1 * num2;
cout <<"Division:" << num1/num2;
```

```
cout << "Modulus :" << num1 % num 2;
cout << "square of num1 :" << num1 * num1;
cout << "square of num2 : << num2 * num2;
cout << "cube of num1" : " << num1 * num1 * num1;
cout << "cube of num2 : " << num2 * num2 * num2;
cout << "Average :" << (num1+num2) / 2;
cout << "Maximum :" << ((num1 > num2) ? num1 :
num 2);
cout << Minimum:" << ((num1 < num2)? num1 : num2;
    }
      };
int main (){
 int a, b;
cout << " Enter two integers :";
cin >> a >> b;
calc (int)cal (a, b).
cout << " Calculate :" << endl;
calc . disp ();
    }
```

O/P

Enter two integers : 10 5

Calculator Addition : 15

Subtraction : 5

Multiplication : 50

Division : 2

Modulus : 0

Square of num1 : 100

Square of num2 : 25

cube of num1 : 1000

Cube of num 2 : 125

Average : 7

Maximum : 10

Minimum : 5

d) WAP to implement push & pop methods from stack using class template.

code:

```
#include <iostream>
using namespace std;
template <class T>
class stack { T arr[5]; int top;
public:
stack() { top=-1;}
void push (T val { if (top==4) cout <<"stack
 overflow";
 else
arr [++top] = val;
    }
void pop() { (top==-1) cout <<"stack underflow";
  else
cout <<"Popped" << arr [top--] << endl;
void disp () { cout "Stack elements:";
 for (int i = 0; i <= top; i++)
 cout << arr [i] << " ";
  cout <<endl;
          }
            };
int main() {
stack <int> s;
s. push (10);
 s. push (20);
s. push ( 30);
s. display ();
s. pop ();
s. display();
```

O/P :-

stack elements: 10 20 30

popped: 30

stack elements: 10 20.

Experiment 11

eklavya
PAGE NO.:
DATE: / /

(without iterators)

8 WAP to create a vector, modify the value multiply by a scalar, display the vector.

code:

```
#include<iostream>
#include<vector>
using namespace std;
int main() {
vector<int> * v = {1,2,3,4,5,6,7,8,9,10};
cout<<" Initial vector:"<<endl;
for(int i=0; i<10; i++) {
cout<<v[i]<<" "<<endl;
    }
cout<<" Multiply by 10:"<<endl;
for (int i=0; i< 10; i++) {
v[i]=v[i] * 10;
        }
cout<<" New vector:"<<endl;
for ( int i=0; i< 10; i++) {
  cout<<v[i]<<" ,"<<endl;
        }
      return 0;
        }
```

O/P:

Intial vector:

1 2 3 4 5 6 7 8 9 10

Multiply by 10.

New vector:

10 20 30 40 50 60 70 80 90 100

(continued)

Experiment 11

eklavya
PAGE NO.:
DATE: / /

2) vector operations using iterators

Code:

```
#include <iostream>
#include <vector>
using namespace std;
int main () {
vector<int> v = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
cout<<"Initial vector:" <<endl;
for (auto it = v.begin (); it != v.end () ; ++it) {
} cout <<endl;
cout << "Multiplying each element by 10..." <<endl;
for (auto it = v.begin(); it != v.end(); ++it) {
*it = (*it) * 10;
        }
cout << "New vector after multiplication:" <<endl;
for (auto it = v.begin ();++it) {
  cout << *it << " ";
    }
cout <<endl;
    return 0;
```

O/P:-

Initial vector:
1 2 3 4 5 6 7 8 9 10
Multiplying each element by 10...
New vector after multiplication:
10 20 30 40 50 60 70 80 90 100

Experiment 12

eklavya
PAGE NO.:
DATE: / /

Q WAP to implement stack

Code:

```
#include <iostream>
#include <stack>
using namespace std;
stack<int> stack1;
void disp() {
stack<int> temp = stack1;
while (!temp.empty()) {
cout<<temp.top()<<" ";
temp.pop();
  }
cout<<endl;
   }
int main() {
 int num;
cout<<"Enter a no:"<<endl;
 cin>>num;
for(int i=0; i<num; i++) {
 int temp;
cout<<"Enter element at position"<<i+1<<endl;
 cin>>temp;
stack1.push(temp);
      }
cout<<endl;
cout<<"Top most element:"<<stack.top()<<endl;
 cout<<endl;
cout<<"Stack elements (top to bottom)--->"endl;
 disp()
cout<<endl;
cout<<"Pop function:"<<endl;
```

```
stack1.pop();
disp();
stack1.pop();
disp();
stack1.pop();
disp();
}
```

O/P:

```
Enter a no:
3
Enter element at position 0:
10
Enter element as position 1:
20
Enter element at position 2:
30
Top most element: 30
Stack elements (top to bottom) --->
30 20 10
Pop function:
20 10
10.
```

b) STL queue implementation

code:

```
#include <iostream>
#include <queue>
using namespace std;
int main() {
queue <int> q;
for (int i=0; i<=10; i++) {
q.push(i*10);
   }
cout<<endl;
cout<<"Front element:" <<
q.front() <<endl;
cout <<" Back element : " q.back()
<<endl;
cout <<endl;
q.pop();
cout <<" After one pop, front = " << q.front <<endl;
cout q.front() <<endl;
cout <<endl;
<<" Queue elements (front to back):";
while (!q.empty()) {
cout <<q.front() << " ";
   q.pop();
   }
   return 0;
   }
```

O/P:

Front element: 0

Back element 100.

After one pop, front = 10

Queue elements (front to back): 10 20 30 40 50 60 70 80 90 100