OOP Practical

1) Write a C++ program for

a). Sum of 2 Numbers
b). Arithmetic operation by switch
c). check even or odd
d). print nos. 1 to 10 nos. using for loop
e). — " — while loop

f) Print below.

a).
```
    *
   * *
  * * *
 * * * *
```

b)
```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

c).
```
1
2 2
3 3 3
4 4 4
5 5 5
```

a).
```
#include<iostream>
int main()
{
int a,b,sum=0
cout<<"Enter 2 Numbers";
cin>>a>>b;
sum = a+b
cout<<"sum is \d"<<sum;
return 0
}
```

Output
Enter 2 Numbers
5
6

b). #include <iostream>
using namespace std;
int main()
{
float a,b;
char o;
cout << "+ - / *";
cin >> ope;
cout << "Enter 2 Numbers";
cin >> a >> b;
Switch(o)
{
case '+' cout << "sum is" << a+b;
~~case~~ break;
case '-' cout << "Difference is" << a-b;
break;
case '*' cout << "Product is " << ~~a*b~~ a*b;
break;
case '/' cout << " Division is" << a/b;
else
cout << "Can't ~~obtain~~ divide";
break
}
return 0;
}

Output
+, -, *, / :/
Enter two Numbers : 3
6
Division is 0.5

d) #include <iostream>
int main()
{
int i;
for(i=1; i<=10; i++)
{
std:: cout << "\n" << i;
}
return 0;
}

Output
1
2
3
4
5
6
7
8
9
10

e) #include <iostream>
int main()
{
int i=1
while (int<=10)
{
std:: cout << "\n" << i;
itt;

```
}
return 0;
}
```

Output

```
1
2
3
4
5
6
7
8
9
10
```

f) i)

```
#include <iostream>
int main()
{
int i,j,s;
for(i=1;i<=3;i++)
{
for(s=1;s<=3-i;s++)
{
std::cout<<" ";
}
for(j=1;j<=i;j++)
{
std::cout<<"*";
std::cout<<" ";
```

```
}
std::cout<<"\n";
}
return 0;
}
```

Output

```
    *
  * *
* * *
```

ii)

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

```
#include <iostream
int main()
{
int i,j;
for(i=1;i<=5;i++)
{
for(j=1;j<=i;j++)
{
std::cout<<j;
}
std::cout<<"\n";
}
return 0;
}
```

c).
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5

```
#include<iostream>
int main()
{
int i,j;
for (i=1;i<=5;i++)
{
for(j=1;j<=i;j++)
{
std::cout<<i;
}
std::cout<<"\n";
}
return 0;
}
```

Experiment 1

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

```
#include<iostream>
using namespace std;
class student
{
private:
string name
float roll;
public:
void accept()
{
cout<<"Enter Student Name \n";
cin>>name;
cout<<"Enter Roll no. \n";
cin>>roll;
}
void display()
{
cout<<"Student Name:"<<name<<endl;
cout<<"Roll no.:"<<roll<<endl;
}
};
int main()
{
student S1
S1.accept();
```

```
s1.display();
}
```

O/P

```
Enter StudentName:
ABC
Enter RollNo.
15
StudentName: ABC
Roll no: 1
```

2)

```
#include <iostream>
using namespace std;
class book
{
private;
string name;
float pages;
public:
float price;
void accept()
{
cout<<"Enter Bookname:\n";
cin>> name;
cout<<"Enter Price:\n";
cin>> price;
cout<<"Enter no. of pages:\n";
cin>> pages;
}
Void display()
{
cout<<"Bookname:"<<name<<endl;
cout<<"Price:"<<price<<endl;
cout<<"Pages:"<<pages<<endl;
}};
Int main()
{
book b1,b2;
b1 accept();
b2 accept();
b1 display();
b2 display();
if (b1.price>b2.price)
cout<<"book1";
else
cout<<"Book2";
}
```

Output

```
Enter book Name:
xyz
Enter price
120
Enter no. of pages
12
Enter book Name
ABC
Enter Price
1015
Enter No. of Pages
512
```

3) #include<iostream>
using namespace std;
class time
{
Public:
float hr,min,sec;
float sum;
void accept()
{
Cout<<"Enter hour";
Cin>>hr;
Cout<<"Enter Minutes";
Cin>>min;
Cout<<"Enter Seconds";
Cin>>sec;
}
void display()
{
cout<<"Hours:"<<hr<<endl;
cout<<"Minutes:"<<min<<endl;
cout<<"Seconds"<<sec<<endl;
}
void tosecond()
{
Sum=hr*3600+min*60+sec;
Cout<<"Total seconds:"<<sum;
};
int main()
{
time t1;
t1.accept()
t1 display();
t1 tosecond();
return 0;
}

Output

Enter Hour:1
Enter Minutes:1
Enter Seconds:1

Hours:1
Minutes:1
Seconds:1

Total Seconds:3661

31/7/25

Practical 2

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

```
#include<iostream>
using namespace std;
class city
{
string name
public:
int pop;
void accept()
{
cout<<"Enter city name"<<endl;
cin>>name;
cout<<"Enter population"<<endl;
cin>>pop;
}
void display()
{
cout<<"The city which has highest population:"<<endl;
cout<<"City Name:"<<name<<endl
cout<<"City Population:"<<pop<<endl;
};
int main()
{
city c[5];
int i,max;
for(i=0;i<5;i++)
{if(c[i].pop>max)
max=i;}
c[max].display();
return 0;
}
```

Output

```
Enter City Name
a
Enter Population
1
Enter City Name
b
Enter Population
2
Enter City Name
c
Enter Population
3
Enter City Name
d
Enter Population
4
Enter City Name
e
Enter Population
5
The city which has highest population
City Name:e
City Population:5
```

2) WAP to declare a class 'staff' having data members name and post. Accept this data for 5 staff and display name of staff who are 'HOD'

```
#include<iostream>
using namespace std;
class staff
{
String name,post;
public:
void accept()
{
cout<<"Enter staff name:";
cin>>name;
cout<<"Enter Post:";
cin>>post;
}
void display()
{if(post=="HOD"||post=="hod")
{cout<<"\n Staff name :"<<name<<endl;staff post:
<<post<<endl;
cout<<name<<" is the head of department"<<endl;
}};
int main()
{
staff s[5];
int i;
for(i=0;i<5;i++)
{s[i].accept()}
for(i=0;i<5;i++)
{s[i].display()}
return 0;
}
```

Output

```
Enter staff name:a
Enter post:salesman
Enter staff name:b
Enter post:manager
Enter staff name:c
Enter post:Hod
Enter staff name:d
Enter post:tally
Enter staff name:e
Enter post:random

Staff name:c
Staff post:hod
c is the head of department.
```

3) WAP to declare a class 'account' have data members account no. and balance. Accept this data for 10 accounts and give interest of 10% when balance is equal or greater than 5000 and display them

```
#include<stdiostream>
using namespace std;
class acc
{
int acc;
float bal,bon=0;
public:
void accept()
{
```

```
cout<<"Enter Acc No.:";
cin >> acc;
cout<<" Enter balance :";
cin >> bal;
}
void calcbonus()
{if (bal>=5000)
{bon = bal * 10/100;}
else
{bon =0;}
}
void display()
{if (bon>0)
{ cout<<"\n Account No. :"<<acc;
cout<<"\n Balance :"<<bal;
cout<<"\n Bonus :"<<bon;
cout<<"\n Total (Balance + Bonus) :"<<bal+bon<<endl
}}};
int main()
{
acc a[10]
int i;
for (i=0; i<10; i++)
{
a[i].accept;
}
for (i=0; i<10 ; i++)
{
a[i].calcbonus();}
for (i=0; i<10; i++)
{ a[i].display();}
return 0;
```

Output

```
Enter Acc No. :1
Enter Balance :1000
Enter Acc No. :2
Enter Balance :2000
Enter Acc No. :- 3
Enter Balance :- 3000
Enter Acc No. :- 4
Enter Balance :- 4000
Enter Acc No. :- 5
Enter Balance :- 5000
Enter Acc No. :- 6
Enter Balance :- 1500
Enter Acc No. :- 7
Enter Balance :- 2400
Enter Acc No. :- 8
Enter Balance :- 3700
Enter Acc No. :- 9
Enter Balance :- 4500
Enter Acc No. :- 10
Enter Balance :- 5500

Account No. :- 5
Balance :- 5000
Bonus :- 500
Total (Balance + Bonus) :- 5500

Account No. :- 10
Balance :- 550
Bonus :- 650
Total (Balance + Bonus) =
7200
```

31/7/25

Practical 3

① Pointer to that object :

```
#include <iostream>
using namespace std;
class book {
String title;
String a_name;
 int price;
 public:
 Voidinfo () {
cout << "Enter the book title, author name and price
        of that book : ";
cin >> title >> a_name >> price;
}
Void display () {
cout << "Book Name : " << title << endl;
cout << "Author Name : " << a_name << endl;
  }
 };
int main () {
  book * p;
  book b1;
  p = & b1;
  p = info ();
  p = display();
 }
```

Output

```
Enter the book title, author name & price of that
book : test
       ABC
       100

Book Name : test
Author name : ABC
Price : 100
```

② This pointer

```
#include <iostream.h>
using namespace std;
class student {
 int roll;
 float perc;
 public:
 void info (int roll, float percen) {
 this -> roll = roll n;
 this -> perc = percen;
 }
 void display () {
 cout << "Roll no :- " << roll << " Marks : " << perc;
 }
 };
 int main () {
  student S1;
  S1.info (3, 89.7);
  S1. display;
```

Output
Roll no:- 3
Marks : 89.7

③ Nested Class.

```
#include <iostream>
using namespace std;
class marks {
 public:
 class percentage {
  int m, n;
  float p, i;
  public:
 void info () {
 cout<<"Enter the marks you got & total marks :";
 cin >>m>>n;
  }
  void disp() {
  i = (float) m/n;
   p = i*100;
   cout << "Percentage" <<p;
   33;3;
  int main(){
  marks: percentage p1;
   p1.info();
   p1. disp();
   }
```

Output

Enter the marks you got & total marks: 56
60
Percentage : 93.33

Q.
1/18

Experiment 4

1)
```
#include <iostream>
using namespace std;

class Number
{
 int value;
 public:
  Number (int v =0)
  {
    value =v;
  }
   void swap(Number &other)
   {
     int temp=value;
         value= other value;
         other value =temp;
   }
    void display()
    {
     cout <<"Value" << value;
    }
   };
      int main()
      {
      Number n1(10), n2(20);
      cout <<"Before swap";
       n1. display();
       n2. display();
       n1. swap(n2);
cout<<"After swap";
  n1. display()
  n2. display();
   return 0;
}
```

Output:

```
Before Swap
value:10
value:20
After swap
 value:20
  value:10
```

2. Friend Function (swap same class)

```
#include <iostream>
Using namespace Std;

class AB
{
 int a, b;
 public:
  void accept()
  {
    cout <<"Enter 2numbers";
    cin>>a>>b;
  }
    friend void swap (AB a1);
};
```

```
Void swap (AB a1)
{
int temp;
 temp = a1.a;
  a1.a = a1.b;
   a1.b = temp;
 cout<<"Values after swapping"<<a1.a<<a1.b;
}
 int main()
 {
 AB.a1;
 a1.info;
 swap (a1);
 }
```

Output

Enter 2 Numbers : 5
4

Values after swapping : 4
5

3. Friend Function swap 2 numbers different class

```
#include <iostream>
using namespace std;
class CB;
class CA;
 int numA;
 public:
CA(int val) : numA(val) {}
void display()
{
cout<<"Value of class A"<<numA;
}
 friend void swap (CA&, CB&);
};
 class CB
{
 private :
  int numB;
 CB (int val) : numB (val) {}
  void disp()
 {
  cout<<"Values in Class B"<<numB;
 }
  friend void swap(CA&, CB&)
 };
  void Swap (CA&a, CB&b);
  {
   int temp = a.numA;
    a.numA = b.numB;
     b.numB = temp;
  }
  int main()
  {
   CA objA (10);
   CA objB(20);
 Cout<<"Before Swapping";
   objA.disp();
   objB disp();
   swap (obj.A, objB);
```

```
cout<<"Afterswapping";
ObjA.displ();
ObjB.displ();
return 0;
}
```

Output

Before Swapping:
Value in ClassA:10
Value in Class B:20

After swapping:
value in Class A:20
Value in classB:10

4. Avg of 2 results

```
#include <iostream>
using namespace std;
class result2;
class result
{
 int a;
 public:
 void accept()
 {
 cout<<"Enter marks out of 50";
 cin>>a;
 }
 friend void(a) (result r1,result2 r2);
};
```

```
class result2
{
 int b;
 public:
 void accept()
 {
 cout<<"Enter marks out of 50:;"
 cin>>b;
 }
 friend void cal(result r1,result r2)
};
 void cal (result r1,result r2)
 {
 float avg = (float) (r1.a+r2.b)/2
 cout<<"Average"<<avg;
 }
  int main()
  {
  result x;
  result2 y;
  x. accept();
  y. accept();
  cal (x,y);
  }
```

Output
Enter marks out of 50:45
Enter marks out of 50:46
Average :-45.5

5. Greatest among 2 numbers (Diff class) (friend)

```
#include<iostream>
using namespace std;
class B;
class A
{
 int a
 public:
 void acc()
 {
 cout<<"Enter avalue";
 cin>>a;
 }
 friend void gr(Aal,Bal);
};
void gr(Aal,Bal)
 {
  if(al.a>bl.b)
 {
  cout<<"first value is greater";
 }
  else
  {
 cout<<"second value is greater";
 }
 int main()
 {
  Ax;
  By;
  x.acc()
  y.acc();
  gr(x,y);
 }
```

O/P

```
Enter value :10
Enter value :100
Second value is greater.
```

Pg 26/8

Experiment-5

Write a C++ program to implement type of constructor

a) Write a program to find the sum of numbers between 1 to n using a constructor where the value of n will be passed to the constructor.

Default Constructor

```
#include <iostream>
using namespace std;
class add {
    int total = 0, n, i;
  public:
   add() {
       n = 5
    }
    Void calculate() {
      total = 0
      for (i = 1; i <= n; i++) {
        total = total + i;
      }
    void calculate() { cout << "Sum of first 5 numbers is: " << total;
};
int main()
  add s1;
  s1. Calculate();
    return 0;
  }
```

Output :- Sum of first 5 numbers is 15

Copy Constructor

```
#include <iostream>
using namespace std;
class add {
  int total = 0, n, i;
 public:
  add (int num) {
      n = num; }
 add (add &obj) {
       n = obj. n;
  }
 Void calculate () {
       total = a;
       for (i = 1; i <= n; i++) {
       total = total + i; }

 cout << "Sum of 1 to n numbers is: " << total;
 };
int main() {
  int n;
  cout << "Enter value of n: ";
    cin >> n;
  add s1 (n);
  add s2 (s1);
   s2. calculate();
   return 0; }
```

Output: Enter the value of n: 5
Sum of 1 to n numbers is: 15

Parametrized Constructor

```
#include<iostream>
using namespace std;
Class add {
    int total=0,n,i;
  public:
  add(int num) {
       n=num; }
  void calculate(){
       total=0;
       for (i=1; i<=n; i++) {
            total = total+i;
       }
       cout<<"Sum of 1ton numbers is:"<<total<<end;
       }};
 int main() {
    int n;
  cout<<"Enter the value of n:";
     cin>>n;
     add s1(n);
     S1.calculate();
      return 0;
    }
```

Output:- Enter the value of n:5
Sum of 1 to n numbers is:15

6) WAP to declare a class student having data members: as name and percentage. Write a constructor to initialize these data members. Accept and display data for one student
Default Constructor

```
#include<iostream>
using namespace std;
class student {
  int per;
  string name;
  public:
   student() {
       per=90;
       name="Arnav";
    }
  void display() {
     cout<<"Name:"<<name;
     cout<<"Percentage:"<<per;
  }};
 int main() {
     int per;
    string name;
     student S1;
      S1.display();
      return 0;
    }
```

Output:
Name = Arnav
Percentage = 90

Copy Constructor

```
#include <iostream>
Using namespace std;
Class Student {
        int per;
        String name;
Public :
  Student (int p, String n) {
        per = p;
        name = n;}

  Student (student &s) {
        per = s.per;
        name = s.name; }
Void display () {
    Cout<<"Name :" <<name;
    Cout<<"Percentage :" <<per;
};
int main ()
  int per:
  String name:
    Couf <<"Enter name :";
     cin>> name;
    Cout<<"Enter percentage :";
     cin>> per;
     Student s1 (per, name);
      Student s2 (s1);
      s2. display ()
      return 0;
      }
```

Output

Enter Name : Arnav
Enter Percentage : 97

Name : Arnav
Percentage : 97

Parametrized Constructor

```
#include <iostream>
Using namespace std;
Class student {
    int per;
    String name;
    public :
    Student (int p, String n) {
        per = p;
        name = n; }
Void display () {
   Cout<<"Name :" <<name ;
   cout << "Percentage :" <<per;
};
int main () {
     int per;
     String name;
    Cout <<"Enter Name :";
    cin >> name;
    Cout <<"Enter Percentage :";
     cin >> per;
     Student S1 (per, name);
       S1. display ();
      return 0;
      }
```

Output :- Enter Name : Arnav
Enter Percentage : 67

Name : Arnav
Percentage : 67

c) WAP to demostrate constructor overloading

```
#include <iostream>
using namespace std;
class Rectangle {
        int l,w;
        public:
        Rectangle() {
            l=1;
            w=2; }
        Rectangle 2 (int a) {
            l=a;
            w=a; }
        Rectangle (int a, int b) {
            l=a;
            w=b; }
    void calculate() {
        int a;
        a=l*w
        Cout<<"Area="<<a;
        }};
int main(){
Rectangle r1;
Rectangle r2(5);
Rectangle r3(4,5);
r1. calculate();
r2. calculate();
r3. Calculate();
}
```

Output:-

Area = 2
Area = 25
Area = 20

Experiment 6

Write C++ programs to implement inheritance

Single Inheritance

```
#include <iostream>
Using namespace std;
 class person {
    protected:
     String name;
      int age;
    }
   class student : protected person {
    int roll_no;
    public:
    void accept() {
    Cout<<"Enter name:";
     Cin>> name;
     Cout<< "Enter age:";
      Cin>> age;
      Cout<<"Enter roll_no:";
       Cin>>roll_no;}
       Void display(){
       Cout<<"Name:"<<name;
       Cout<<"Age:"<<age;
       Cout<<"Rollno.:"<<roll_no;
      }};
    int main(){
     Student s;
      s. accept();
```

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

Output:

```
Enter Name :- Arnav
Enter Age :17
Enter roll_no :36
  Name :- Arnav
  Age :- 17
  Roll_no :- 36
```

Multiple Inheritance

```
#include <iostream>
 using namespace std;
  class academic {
   protected:
    string name;
     int mark; };
  class sports {
      protected:
         int score; };
 class student : protected academic , protected sports {
  int total;
  Public:
  void accept() {
  Cout<< "Enter name:";
  cin>>name;
  Cout << "Enter marks:";
   Cin>> marks;
   Cout<< "Enter score:";
    Cin>> score;
    total = marks+score; }
void display() {
 Cout<<"Name:"<<name;
 Cout<< "Marks:" << marks;
 Cout<< "Score:" <<score;
 Cout<< "Total:" << total;
 }};
int main() {
   student S;
    S. accept();
    S. display();
    return 0;
  }
```

Output

```
Enter name :- Arnav
Enter marks :- 68
Enter Score :- 7
Name :- Arnav
 Marks :- 68
 Score :- 7
 Total :- 75
```

3) Multilevel Inheritance

```
#include <iostream>
using namespace std;
 class vehicle {
     Public:
      String model;
      String brand; };
```

```
class car : public vehicle {
    protected:
      int type; };
class electric car: public car {
  public:
  int battery = capacity;
     void accept () {
     cout<<"Enter model:";
     cin>> ~~brand~~ model;
     cout<<"Enter brand:";
     cin>> brand;
     cout<<"Enter type (1 for sedan, 2 for SUV):";
      cin>> type;
      cout<<"Enter battery capacity (in KWh):";
       cin>> battery_capacity; }
 void display(){
  cout<<"Model:"<<model;
  cout<<"Brand:"<<brand;
  cout<<"Type:"<<(type==1? "Sedan": "SUV");
  cout<<"Battery Capacity:"<<battery_capacity<<"Kwh";
  }};
int main(){
electricity C:
  C.accept();
  C.display();
  return 0;
}
```

Output: Enter Model = M++
Enter Brand: XYZ
Enter type (1 for sedan, 2 for SUV): 2
Enter Battery Capacity (in KWh): 44
Model: M++
Type: SUV
Battery Capacity: 44 Kwh

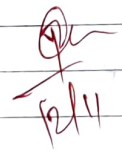
4) Hierarchial Inheritance

```
#include <iostream>
using namespace std;
class employee {
protected:
  string name;
  int id; };
class manager : public employee {
      private:
        string dept; };
class Developer : public employee {
 public:
  string lang;
  string dept;
  void accept(){
   cout<<"Enter emp name:";
      cin>> name;
    cout<<"Enter emp ID:";
    cin>> id;
    cout<<"Enter programming language:";
     cin>> lang;
     cout<<"Enter Department:";
       cin>> dept; }
    void display(){
    cout<<"Name:"<<name;
    cout<<"ID:"<<id;
    cout<<"Programming language:"<<lang;
    cout<<"Department:"<<dept;
    }};
int main(){
 Developer d;
```

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

Output

```
Enter emp name: Arnav
Enter emp ID: 001
Enter Programming language: C++
Enter department: Technical
Name: Arnav
  ID: 1
Programming language: C++
 Department: Technical
```



Experiment 7

① Area of laboratory (rectangle) and area of classroom (square) (constructor overloading).

```
#include <iostream>
using namespace std;
class xyz {
public:
 int l,b,s,a;
void calc (int len,int br) {
 l = len;
 b = br;
 a = l*b;
}
void calc (int si) {
 s = si;
 a = s*s;
}};
int main() {
 xyz a;
cout <<"Enter length and breadth of rectangle:";
cin>> a.l >>a.b;
 a.calc (a.l,a.b);
cout<<"Area of rectangle:" <<a.a <<endl;
 xyz b;
cout <<"Enter side of square:";
 cin>> b.s;
 b.calc (b.s);
cout <<"Area of square:" <<b.a <<endl;
 return 0;
}
```

Output :- Enter length and breadth of rectangle : 10
5

Area of rectangle : 50
Enter side of square : 5
Area of square : 25

② Constructor Overloading (Sum of 10 integer and 5 float)

```
#include<iostream>
using namespace std;
class sum {
public:
 float fsum = 0;
 int isum = 0;
 void calc (float a) {
  fsum = fsum + a;
 }
 void calc (int a) {
  isum = isum + a;
 }};
int main() {
 sum s;
 float f;
 int i;
 cout << "Enter 5 float values:";
 for (int j=0; j<5; j++)
  cin >> f;
  s.calc();
 }
cout << "Sum of float values:" << s.fsum
cout << "Enter 10 integer values";
for (int j=0; j<10; j++) {
 cin >> i;
 s.calc(i);
}
cout << "Sum of integer values:" << s.isum << endl;
return 0;
}
```

③ Unary Operator

```
#include<iostream
using namespace std;
class numb {
public:
 int val;
 void acc() {
 cout << "Enter a numb";
 cin >> val;
 }
 Void disp() {
 cout << "Value" << val;
 }
 void operator - () {
 val = -val;
 }};
 int main() {
 numb obj;
 -obj;
 obj.disp;
 }
```

Output :-
Enter a num
Value : -

④ Unary ++ operator (Pre & Post increment)

```
#include<iostream>
using namespace std;
Class ab {
public:
int num;
void acc(){
cout<<"Enter Number:";
cin>>num;
}
void disp(){
cout<<"Number:"<<num;
}
void operator ++(){
 ++ num;
}
void operator ++(int){
num++;
};
int main(){
ab obj;
obj.acc();
++obj;
obj++;
obj.disp();
return 0;
}
```

Output:
Enter Number: 5
Number: 7

Practical 8

① Overload '+' operator so that two string can be concatenated

```
#include <iostream>
#include<string>
using namespace std;
class abc{
public:
string str;
void acc(){
cout<<"Enter string:";
cin>>str;
}
abc operator+(abc s){
abc temp;
temp.str = str+s.str;
return temp;
}
void disp(){
cout<<"Concatenated String:"<<str;
};
int main(){
abc s1,s2,r;
s1.acc();
s2.acc();
r = s1+s2;
r.disp();
return 0;
}
```

Output:-
Enter String: Hello
Enter String: World
Concatenated String: HelloW

② WAP to create a base class ILogin having data member name and password. Declare accept() function virtual. Derive EmailLogin and MembershipLogin classes from ILogin. Display Email login details and membership login details of employee.

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

class ILogin {
protected;
 String name,password;
 public:
 Void accept() {
 cout << "Name:";
  cin>> name;
  Cout<< "Password:";
   cin>> password;
};
class EmailLogin : Virtual
public ILogin {
public:
 void showEmail() {
 Cout << name << " " << password;
};
class MembershipLogin: virtual public ILogin {
public:
 void MembershipLogin() {
  cout <<name << " " << password << endl;
};
class Employee : public EmailLogin, public MembershipLogin {
EmailLogin public:
void input() {
 accept();
}
void display() {
 showEmail();
 showMembership();
};
int main() {
Employee e;
e.input();
e.display();
return 0;
}
```

12/11

Experiment 9

a). WAP to copy the contents of one file another, open "First.txt" in read (ios mode). Copy the contents of "First.txt" into "Second.txt".

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

int main() {
  if stream fin ("First.txt");
  ofstream fout("Second.txt");
   char ch;
  while (fin.get(ch)) {
   fout.put(ch);
    }
   cout <<"File copied successfully!";
   fin.clou();
   fout.clou();
   return 0;
   }
```

b). Count digit & space using file handling

```
#include <iostream>
#include <fstream>
using namespace std;
int main(){
ifstream fin("Data.txt");
char ch;
int digits=0, space=0;
while (fin.get(ch)) {
if (is digit(ch)) digits++;
 if (ch==' ') spaces++;
}
cout <<"Digits" << digits <<endl;
cout <<"Spaces" <<spaces <<endl;
fin close();
return 0;
}
```

c) Count words using file handling

```
#include<iostream>
#include <fstream>
#include <string>
using namespace std;
int main() {
if stream fin ("Data.txt");
string word;
int count=0;
while (fin >> word) {
   count++;
  }
cout<<"Total words:" <<count<<endl;
fin close ();
return 0;
}
```

d) Count occurance of a given word using file handling

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main(){
ifstream fin("Data.txt");
string word, target;
int count = 0;
cout<<"Enter word to search: ";
cin>>target;
while (fin>>word) {
  if (word == target) {
    count++; }}

cout<<" Occurence of "<<target<<" "<<count<<endl;
fin.close();
return 0;
}
```

12/11

Experiment 10

a. Sum of array elements using function template

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

template <typename>
T sumArray(T arr[], int size);
    T sum = 0;
    for (int i=0; i<size; i++) {
      sum += arr[i];
    }
    return sum;
    }
int main() {
const int size = 10;
int Array[size] = {1,2,3,4,5,6,7,8,9,10};
int intsum = sumArray(intArray, Size);
cout<<"Sum of integer array: "<<int sum;
return 0;
}
```

O/P:

55
59.6

b). Square Function (Template Specialization)

```
#include <iostream>
#include <string>
using namespace std;
template <typename T>
 T square (T value){
  return value * value;
  }
 template <>
 String Square <string> (string str){
 return str+str;
 }
 int main(){
 int intNum=5;
 Cout<< "Square of integer"<<int Num <<":"<<Square (um)
 String mystr = "Hello";
 cout<< "Square of string "<< 'mystr" <<":" << square (Byshl)
 return0;
 }
```

O/P
25
Aaa Aaa

c). Simple Calculator (Usestemplate)

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

template<typename T>
class calculator {
private:
  Tnum1;
  Tnum2;
public:
 Calculator (Tn1, Tn2): num1(n1), num2 (n2) Tadd(-)
                                              const
return num1-Inum2;
 }
 T subtract () const {
  return num1-num2;
   }
 T multiply() const{
  return num1* num2;
  }
 T divide () const {
  return num1/num2;
  return0;
 }
```

d) Stack implementation (class template)

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

template <typename T>
class stack {
private:
vecctor <T> elements;

public:
void push (T val) {
elements.push_back(val);
cout<< "Pushed:" <<val;
} }
 T.pop()
 {
 if (elements.empty()) {
cout<< "Error: Stack is empty" << endl;
return T();
T topElement = elements.back();
elements.pop_back();
return topElement;
}
bool isEmpty() const {
return element.empty()
 }
 };
 return 0;
 }
```

O/P: 20

Experiment 11

WAP to implement generic vector

a) To modify the value of a given element.

```
#include<iostream>
 template<typename T>
  class vector {
   T a[100];
   int size;
    public:
   vector(int s) : size(s) {}
   void set (int i, T val) {
   if (i>=0 && i<size)
    a[i] = val;
    else
    cout<< "invalid";
   }
   T get (int i) {
    if (i>=0 && i<size)
    return a[i];
    cout<<"invalid";
    return T;
    }
    void display () {
    for (int i=0; i<size; i++)
    cout << a[i] <<" ";
    cout<< endl;
    }};
    int main() {
```

```
Vector<int> v(5);
for(int i=0;i<5;i++) {
v.set (i,i*10);
  v.display();
  v.set (2,99);
  Cout<<"After modification";
  v.display();
  return 0; }
```

O/P:

0 10 20 30 40

After Modification: 0 10 99 30 40

b). To multiply by a scalar value

```
#include<iostream>
#include<vector>
using namespace std;
int main(){
Vector<int> vec = {1,2,3,4,5};
int scalar = 3;
for (int value:vec){
   val=val*scalar;}
 for (int& value:vec){
   Cout<<val<<" ";}
   Cout<<endl;
   return 0;
   }
```

O/P

3 6 9 12 15

c) To display the vector is the form (10,20,30...)

```
#include<iostream>
#include<vector>
using namespace std;
int main(){
vector<int> vec = {10,20,30,40,50};
Cout<<"(";
for(int i=0;i<vec.size();i++) {
 Cout<<vec[i];
 if(i!=vec.size()-1)
 Cout<<",";
 }
Cout<<")"<<endl;
return 0;
}
```

O/P

(10,20,30,40,50)

Experiment 12

Q Write C++ program using STL:

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<<"Top element:"<<s.top()<<endl;
      s.pop();
cout<<"Topelement after pop:"<<s.top()<<endl;
cout<<"Stack Size:"<<s.size()<<endl;
if(s.empty()){
cout<<"Stack is empty"<<endl;
}else{
cout<<"Stack is not empty"<<endl;
}
return 0;
}
```

Output:

Top element: 30
Top element after pop: 20
Stack size: 2
Stack is not empty.

b) Implement queue

```
#include<iostream>
#include<queue>
using namespace std;
int main(){
queue<int> q;
q.push(10);
q.pop(20);
q.push(30);
cout<<"front elements:"<<q.front()<<endl;
cout<<"back elements:"<<q.back()<<endl;
q.pop()
cout<<"After pop operations"<<endl;
cout<<"Front element"<<q.front()<<endl;
cout<<"queue.size"<<q.size()<<endl;
if(q.empty){
cout<<"Queue is empty:"<<endl;
}else{
cout<<"Queue is not empty:"<<endl;
}
return 0;
}
```

O/P

front element: 10
back element: 30
after pop;
front element: 20
queue: 2
queue is not empty.

12/11

c).

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;
struct person {
string name;
int age;
Person(string n, int a): name(n), age(a) {}
};
int compare_age(const person &p1, const person &p2);
return (p1.age < p2.age)?1:0;}
int main(){
Vector<person> person = {
                    person("Alice", 30);
                    Person("Bob", 25);
                    person("Charlie", 35);
                    person("David", 28);};
Sort(people begin(), people.end());
[](const person &a const Person &b) {
    return compare_age(a,b);}
Cout<<"Sorted records by age:/n";
for(auto &p.people){
Cout<<p:name"-"<<p.age<<"\n";}
int sample age = 28;
int found index = -1;
for(int i=0;i<(int) people.size();itt){
if(people[i].age = search.age?!:0){found index = i;
break;}}
if(found != index)
Cout<<"person with age"<<search_age<<endl;
else
Cout<<"person with age"<<search
```

Output

Sorted records by Age

Bob = 25

David = 28

Alice = 30

Charlie = 35