```
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 : acc ();
- obj ;
```

O/P : Enter a numb : 5
Value = -5

4] Unary ++ operator ( Pre & post increment)

```
# include <iostream>
using namespace std;
class ob {
public:
int num;
void acc () {
cout << " Enter number :";
cin >> num;
}
void disp () {
cout << " rumber :" << num << endl;
}
void operator ++() {
+ + num ;
}
void operator ++ (int) {
num ++;
}};
int main () {
ob - obj ;
obj · acc ();
++ obj ;
obj ++ ;
obj · disp ();
return 0;
}
```

→ O/P:
Enter number : 5
rumbe : 7

```
    string name;
    int age;
    public:
    void getpersondata ()
    cout << " enter name ";
    cin >> name;
    cout << " enter age;
    cin >> age;
    }
    void show person data ()
    cout << " Name: " << name << endl;
    cout << " Age: " << age << endl;
    }
};
class student : public person {
protected:
int roll no;
public:
void get student data ()
cout << " enter roll no ";
cin >> roll no;
}
void show student data ()
cout << " roll no: " << roll no << end l;
}
};
class ~~syntax~~ sports {
protected:
int sports marks;
public:
void getsportsmarks ()
```

```
class manager : protected employee
{
private:
string dept;
public:
void accept ()
{
cout<<" enter emp name ";
cin>> name;
cout<<" enter emp id";
cin>> emp id;
cout<<" enter department";
cin>> dept;
}
void display ()
{
cout<<" emp name = "<< name;
cout<<" emp id = "<< emp id;
cout<<" department = "<< dept;
}
};
class developer : protected department
{
private:
string programming lang;
public:
void accept ()
{
cout<<" enter programming language ";
cin>> programming lang;
}
void display ()
{
cout<<" programming language = "<< programming lang;
}
};
int main ()
{
manager m1;
m1. accept ();
m1. display ();
developer d1;
d1. accept ();
d1. display ();
return 0;
}
```

O/P

enter name = Soham
enter emp id = 111
enter dept = HR
name = Soham emp id = 111 dept = HR
enter programming lang C++
programming lang = C++

5] Hybird inheritance.

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

```
private:
int total score ;
public:
void accept ()2
Cout 2c" enter student marks ";
cin >> marks ;
cout2c" enter ~~marks~~ sports score ";
cin >> score ;
}
void ~~accept~~ calculate()
2
total score = marks + score ;
}
void display () 2
cout 2c" Student marks =" 2c marks;
cout 2c" sports score =" 2c score;
cout 2c" total score =" 2c total score;
}
};
int main ()
2
result r1 ;
r1. accept ();
r1. ~~display ()~~ calculate ();
r1. display ();
return 0;
}
```

```
O/P : enter student marks : 90
      enter sports score : 80
      student marks = 90 , sports score = 80
      total score = 170
```

3] Multilevel inheritance

->
```
#include <iostream>
using namespace std;
class vehicle {
public:
string type;
};
class car: public vehicle
{
public:
string type;
};
class electronic: protected car
{
private:
int battery capacity;
```

->
```
#include <iostream>
using namespace std;
class vehicle {
public:
string brand;
string model;
};
```

```
Rectangle (int a)
{
l = a;
w = b;
}
void calculate ()
{
int a;
a = l * w;
cout << " Area =" << a;
}
};
int main()
{
Rectangle r1;
Rectangle r2 (5);
Rectangle r3 (2,5);
r1. calculate ();
r2. calculate ();
r3. calculate ();
}
```

O/P:

Area = 2 Area = 25 Area = 20

- Define Class "college" numbers as roll no, name, course. WAP using constructor with default value as "computer ~~engine~~ engineering" for course. Accept and display data for 2 objects.

```
#include <iostream>
using namespace std;
class college
{
int roll_no;
string name, course;
public:
college (int r, string n, string c = "computer
        engineering);
{
roll_no = r;
name = n;
course = c;
}
void display ()
{
cout << " name, roll no and course is:" << name <<
        roll no << course;
}};
int main ()
{
college c1 (1," Parth", "Cse");
college c2 (1," Parth");
c1. display;
c2. display ();
}
```

12/11

```
int per;
string name;
public:
student ()
{
per = 70;
name = "abc";
}
void display ()
{
cout<<" name = "<< name << end l;
cout<<" percentage = " << per << end l;
}
~ student ()
{
Cout<<" object destroyed ";
}
};
int main()
student s1;
S1. display ();
return 0;
}
```

O/P:

name = abc

percentage = 70

object destroyed.

```
#include <iostream>
using namespace std;
class student {
int per;
string name;
public:
student (int p, string n)
{
per = p;
name n;
}
void display ()2
cout<<" Percentage = "<< Per;
cout<<" Name = " << nome;
{
~ student ()
{
cout<<" object destroyed ";
}
};
int main() {
student s1 (80}, "Parth");
s1. display ();
return 0;
}
```

O/P:

Percentage = 80

Name = Parth

object destroyed

Exp-5

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

->
```
#include <iostream>
using namespace std;
class sum
{
int num;
int s;
public:
sum()
{
s = 0;
num = s;
for(int i=1; i<num; i++)
{
s = s+i;
}
}
void display()
{
cout<<" sum = "<<s;
}
~sum()
{
cout<<" object destroyed ";
}
};
int main()
{
sum s1;
s1.display();
return 0;
}
```

O/P

sum = 10

*
```
#include <iostream>
using namespace std;
class sum
{
int num;
public:
sum(int n)
{
num = n;
int s = 0;
for(int i=1; i<sum; i++)
{
s += i;
}
cout<<" sum = "<<s;
}
~sum()
{
cout<<" object destroyed ";
}
};
```

O/P
Sum : 30

2] WAP a program with a class Number that contains a private integer. Use a friend function swapNumbers (Number & Number &) to swap the private values of two Number projects.

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

class Number {
  int value;
public:
Number (int val): value(val) {}
void display () { cout << value << endl; }
friend void swapNumbers (Number &, Number &);
};

void swapNumbers (Number &n1, Number n2) {
int temp = n1.value;
n1.value = n2.value;
n2.value = temp;
}

int main () {
Number n1(5), n2(15);
cout << " Before swap: "; n1.display (); n2.display();
swapNumbers (n1, n2);
cout << " After swap: " n1.display (); n2.display ();
return 0;
}
```

O/P :
Befor swap : 5
15

After swap : 15
5

3] Define two classes Box and cube, each having a private volume. Write a friend function findGreater (Box, cube) that determines which object has a larger volume.

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

class cube;
class box {
  int volume;
public:
Box (int v): volume (v) {}
friend void findGreater (Box, Cube);
};

void findGreater (Box b, Cube c) {
cout << " Greater volume: " << (b.volume > c.volum
    b.volume : c.volume) << endl;
}
```

```
void gr (A a1, B b1);
};
void gr (A a1, B b1){
if (a1.a > b1.b){
cout << "First value is greater";
}
else{
cout << "Second value is greater";
}
int main(){
A x;
B y;
x.acc();
y.acc();
gr(x,y);
}
```

O/P:

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

- Practic questions on friend function

i) Create two classes, class A and class B, each with a private integer. Write a friend function sum() that can access private data from both classes and return the sum.

→

```
#include<iostream>
using namespace std;
class ClassB;
class ClassA {
  int a;
public:
ClassA(int val): a(val) {}
friend int sum(ClassA, ClassB);
};

Class ClassB {
 int b;
public:
ClassB(int val): b(val) {}
friend int sum(ClassA, ClassB);
};
int sum(ClassA obj_A, ClassB obj_B) {
 return obj_A.a + obj_B.b;
}
int main(){
 ClassA a(10);
 ClassB b(20);
cout << "sum:" << sum(a,b) << endl;
return 0;
}
```

3] Friend function swap 2 numbers different class

->
```
# include <stdio.h> <iostream>
using namespace std;
class CB;
class CA;
int num A;
public:
CA (int val) : num A (val) {}
void disp () {
cout << " Value in class A : " << num A << endl;
}
friend void swap (CA &, CB &);
};
Class CB {
private:
int num B;
CB (int val) : num B (val) {}
void display () {
cout << " Value in class B : " << num B << endl;
}
friend void swap (CA &, CB &);
};
void swap (CA &a , CB &b) {
{
int temp = a. num A;
a. num A = b. num B;
b. num B = temp;
}
```

```
int main () {
CA ob, A (10);
CB ob, B (20);
cout << " Before swapping: " << end l;
ob, A. disp ();
ob, B. disp ();
swap (ob, A , ob, B);
cout << " In After swapping: " << end l;
ob, A . disp ();
ob, B. disp ();
return 0;
}
```

O/P

Before swapping:

Value in Class A: 10

Value in Class B: 20

After swapping

value ~~of~~ in Class A: 20

Value in Class B: 10

O/P

| Before swap : | After swap : |
| --- | --- |
| value : 10 | value : 20 |
| Swap : 20 | value : 10 |

2] Swap 2 numbers from same class using friend function:

->
```
#include <iostream>
using namespace std;
class AB {
int a, b;
public:
void into() {
cout << " Enter 2 number: ";
cin >> a >> b;
}
friend void swap (AB &l);
};
void swap (AB &l) {
int temp;
temp = l.a;
l.a = l.b;
l.b = temp;
cout << " In Values after swapping: " << l.a << l.b;
}
int main() {
AB al;
al.into;
swap (al);
}
```

O/P:

Enter 2 numbers : 5
4

Value after swapping : 4
5

```
int main ()
{
book b1;
book *p;
p = &b1;
p -> accept ();
p -> display ();
return 0;
}
```

O/P:

price = 500
bookname = harry potter
authorname = Unknown

7] WAP to declare class student having data members as roll_no and percentage. Using this pointer involve members function to accept the data and display this data for 1 object of a class.

->

```
# include <iostream>
using namespace std;
class student
{
int roll_no;
float per;
public:
void accept ()
{
cout << " enter rollno ";
cin >> rollno;
cout << " enter percentage ";
cin >> per;
}
void display ()
{
this -> accept ();
cout << " roll_no = " << roll_no;
cout << " percentage = " << per;
}
};
int main ()
```

```
for (int i=0 ; i<5 ; i++)
{
cout << "\n enter details for staff " << i+1 <<
        "\n ";
  s[i]. accept ();
}
cout << "\n list of HODs \n";
 for (int i=0 ; i<5 ; i++)
 {
 s[i]. display ();
}
 return 0;
}
```

Output

```
Enter details for staff 1
Enter name and post Parth
HOD
Staff 2
Enter name and post Soham , lecturer
Staff 3
Enter name and post Yuvraj
 CEO
Staff 4
Enter name and post Viraj
 director
Staff 5
Enter name and post Ayush
 manager
 list of HOD's
    Parth
```

5] Write a Program to declare a class 'Account having data members as account number and balance. Accept this data for 5 accounts and gave interest of 10% where account balance is equal to greater than 5000 and display them.

```
#include <iostream>
using namespace std;
class account
{
public:
int acc_no;
int bal;
void accept ()
{
cout << " Enter the account no ";
cin >> acc_no;
cout << " Enter the balance :";
cin>> bal;
 }
};
  int main ()
{
 account a[5];
 int i;
 for (i=0; i<5; i++)
 {
```

```
b1.accept();
b2.accept();
cout<<"\n Book with greater Price :\n";
if (b1.price > b2.price)
{
b1.display();
}
else if (b2.price > b1.price)
{
b2.display();
}
else
{
cout<<"Both books have the same price:\n";
b1.accept display();
b2.display();
}
return 0;
}
```

Output

```
Enter value of name, price, pages : Bhagvatgita
500
300
Enter value of name, price, pages : Avengers
650
350
Book with greater price
Name : Avengers, price = 650, pages : 350
```

3] Write a program to declare a class time. Accept the time in HH:MM:SS and convert it into seconds and display them.

```
#include <iostream>
using namespace std;
class time
{
private:
int H, M, S;
public:
void accept()
{
cout<<"Enter Time";
cin>>H>>M>>S;
}
void display()
{
int total;
total = (H*3600) + (M*60) + S);
cout<<"Total time in seconds"<<total;
}
};
int main()
{
Time t1;
t1.accept();
t2.display();
return 0;
}
```

*
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6

```
# include <iostream>
using namespace std;
int main()
{
int rows = 6
for (i = 1 ; i <= rows ; i++)
{
for (int j = 1 ; j <= i ; j++)
{
cout << j << " ";
}
cout << endl;
}
return 0;
}
```

Output:

1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6

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

```
# include <iostream>
using namespace std;
int main()
{
int rows = 5;
for (int i = 1; i <= rows; i++)
{
for (int j = 1; j <= i ; j++)
{
cout << i << " ";
}
return 0; cout << endl;
}
return 0;
}
```

Output

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

* Write a C program check number is even or odd

```
#include <iostream>

int main()
{
int number;

std :: cout << "Enter an integer ";
std :: cin >> number;
if (number % 2 == 0)
{
std :: cout << "number << " is even" << std :: endl;
}
else
{
std :: cout << number << " is odd " << std :: endl;
}
return 0;
}
```

Output

Enter an integer 18

18 is even

* Write a C program to print 1 to 10 numbers using for loop.

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

Output

1

2

3

4

5

6

7

8

9

10

* Write a C program to add 2 no·s

```
#include <iostrem>
 int main()
{
 int a, b, c;
std :: cout <<" enter values of a and b";
std :: cin >> a >> b;
 c = a + b;
 std :: cout << "addition is " << c;
return 0;
}
```

Out put;

enter value of a and b

3

15

addition : 18

* Write a C program to print Arithematic operations (+/*) using switch.

```
# include <iostream>
 int main()
{
char op;
double num1, num2;
std :: cout << " Enter operator (+, -, *, /)
std :: cin >> op;
std :: cout << " Enter two numbers :";
std :: cin >> num1 >> num2;
 switch (op)
{
case '+';
std :: cout << num 1 << "+" << num2 << "="
      << num1 + num2;
  break;
case '-';
std :: cout << num 1 << "-" << num2 << "="
      << num1 - num2;
 break;
 case '*';
std :: cout << num1 << "*" << num2 << "="
      << num1 * num2;
 break;
case '/';
if (num2 != 0
std :: cout << num1 << "/" << num2 << "=" <
          num2;
 else
std :: cout << "Error! Division by zero";
break;
default;
std :: cout << " Invaild operator!";
 }
return 0;
}
```

Output:

Enter operator (+, -, *, /): +
Enter two numbers: 3
22
~~25 + 22 = 25~~
3 + 22 = 25

Enter operator (+, -, *, /): -
Enter two numbers: 45
18
45 - 18 = 27

Enter operator (+, -, *, /): *
Enter Enter two numbers: 3
18
3 * 18 = 54

Enter operator (+, -, *, /): /
Enter two numbers: 15
3
15 / 3 = 5

* Write a C program to print 1 to 10 nos using while loop.

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

Output:

1 2 3 4 5 6 7 8 9 10

* Write a C program to print the following pattern:

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

```
#include <iostream>
using namespace std;
int main ()
{
int i, j, space;
int rows = 3;
for (i = 1; i <= rows; i++)
{
for (space = 1; space <= rows - i; space
{
cout << " ";
}
for (j = 1; j <= i; j++)
{
cout << endl;
}
return 0;
}
```

Output:

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

* Write a C program for addition of 2 nos

```
#include <iostream>
using namespace std;
class number
{
int a, b, c;
public:
void add()
{
cout << "Enter any 2 numbers";
cin >> a >> b;
c = a+b;
cout << "addition =" << c;
}
};
int main()
{
number n1;
n1.add()
return 0;
};
```

31/7

Output

```
Enter any 2 numbers 3
18
addition = 21
```

Exp 1

Write a program to declare class students having data members as roll_no; name. Accept and display data for 1 object.

```
#include <iostream>
using namespace std;
class student
{
int roll;
string name;
public:
void accept()
{
cout << "Enter value of roll, name";
cin >> roll >> name;
}
void display()
{
cout << "roll:" << roll << ", name:" << name
     << endl;
}
};
int main()
{
student S1;
S1.accept();
S1.display();
return 0;
}
```

Output

Enter value of roll, name 62
Parth
roll: 62, Name: Parth.

2] Write a program to declare a class book having data members as Book name, price, no. of pages, accept for 2 objects and display the name of Book having greater price.

```
#include<iostream>
using namespace std;
 class book
{
int pages;
string name;
float price;
public:
void accept()
{
cout<<" Enter value of name,price,pages";
cin>> name >> price >> pages;
}
void display()
{
cout<<"Name:"<<name<<", Price:"<<price
        <<"Pages:"<<pages<<endl;
    }
};
int main()
{
  book b1,b2;
```

Output

Enter Time 12

56

45

Total time in seconds 46609

31/7

Exp 3

1] Write a C program 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.

```
#include <iostream>
using namespace std;
class city
{
int population;
string name;
public
void accept ()
{
cout <<" Enter city name and population";
cin >> name >> population;
}
};
int main()
{
city c[5];
int i, max;
for (i=0; i<=5; i++)
{
c[i].accept();
}
max = c[0].population;
for (i=0; i<5; i++)
```

```
{
if (c[i].population > max)
{
max = i;
}
c[max].display();
}
return 0;
}
```

Output

```
Enter city name and population : pune
25000
Enter city name and population : Mumbai
20000
Enter city name and population : Nashik
18000
Enter city name and population : Banglore
30000
Enter city name and population : goa
10000

City with highest population :
Name : Banglore , population : 30000
```

7] Write a C++ program to declare a class 'staff' having data members as name and post. Accept this data for 5 staff and display names of staff who are "HOD".

```
#include <iostream>
using namespace std;
class staff
{
string name;
string post;
public:
void accept()
{
cout << "Enter name and Post";
cin >> name >> post;
}
void display()
{
if (post == "HOD" || post == "hod" || Post == "
{
cout << "HOD" << name << endl;
}
}
};
int main()
{
staff s[5];
```

```
    a[i].accept();
}
for(i=0; i<5; i++)
{
  if(a[i].bal >= 5000)
  {
    a[i].bal = a[i].bal + (a[i].bal * 0.1);
cout<<" account number :" << a[i].acc_no <<",
    updated balance :" << a[i].bal;
  }
}
return 0;
}
```

3/7

Exp 3

1] Write a program to declare class book having data members as price, author name, book name. Accept and display data for 1 object using its object of that class

```
#include <iostream>
using namespace std;
class book
{
float price;
string price bookname;
string authorname;
public:
void accept()
{
cout<<" enter price";
cin >> price;
cout<< "enter bookname";
cin>> bookname;
cout<<" Enter * authorname";
cin >> authorname;
}
void display()
{
cout<<" price =" <<price;
cout<<" bookname =" <<bookname;
cout<<" authorname =" <<authorname;
}
};
```

```
{
student S1
S1.display();
return 0;
}
```

O/P :

```
Enter student
roll-no and percentage
56
80
roll-no = 56
per = 80
```

Nested Class

3)

```
#include <iostream>
using namespace std;
class marks {
public:
class percentage {
int marks, n;
float n, i;
public:
void info() {
cout << " Enter the marks you got :";
cin >> marks;
cout << " Enter total marks:";
cin >> n;
}
void display() {
i = (float) marks/n;
n = i * 100;
cout << " percentage :" << n;
}
};
int main() {
marks m1;
Marks :: Percentage n1;
n1.info();
n1.display();
}
```

O/P

Enter the marks you got : 67
Enter total marks : 70
Percentage : 95.71

14/8

Exp-4

1) Swap 2 numbers from same using object as function argument.

->
```
#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 <<end.l;
}};
int main () {
 number n1(10) ,n2(20);
 cout<<" Before Swap :" <<end l;
 n1.disp();
 n2.disp();
 n1.swap (n2);
 cout<<" \n After Swap :" <<end l;
 n1.disp();
 n2.disp();
 return 0;
}
```

4] Avg of two results

```
-> #include <iostream>
   using namespace std;
   class result 2;
   class result {
   int a ;
   public:
   void accept ()
   cout << " Enter marks out of 50:";
   cin >>a;
   }
   friend void cal (result r1 ,result 2 r2);
   };
   class result 22
   in b;
   public:
   void accept ()
   cout <<" Enter marks out of 50:");
   cin >>b;
   }
   friend void cal (result r1, result r2)
   float avg = (float) (r1.a + r2.b) /2;
   cout <<" In avg:" <<avg;
   }
   int main ()
   result x;
   result 2 y;
   2. accept ();
   y. accept ();
   cal (x,y);
   }
```

O/P : Enter marks out of 50 : 45
Enter marks out of 50 : 46
avg : 45.5

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

```
-> #include <iostream>
   using namespace std;
   class B;
   class A
   int a ;
   public:
   void acc ()
   cout <<" Enter value:";
   cin >> a;
   }
   friend void gr (A a1, B b1);
   };
   class B
   int b ;
   public:
   void accept ()
   cout <<" Enter a value:";
   cin >> b;
   public: }
```

```
int main() {
 Box box(118);
 cube cube(90);
 findGreater(box, Cube);
 return 0;
 }
```

O/P : Greater volume : 118

4] Create a class Complex with real and imaginary parts as private members. Use a friend function to add two complex numbers and return the result as a new complex object.

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

Class Complex {
 int real, imag;
public:
Complex(int r=0, int i=0): real(r), imag(i) {}
void display() { cout << real << "+" << imag << "i" <<
        end l; }
friend complex add (Complex, Complex);
};

Complex add (Complex c1, Complex (2) {
return Complex (c1.real + c2.real, c1.imag +
        c2.Imag);
}
```

```
int main() {
complex c1(2,5), c2(2,3);
Complex result = add (c1, c2);
cout << "sum of complex numbers:";
result.display();
return 0;
}
```

O/P
Sum of complex numbers : 6 + 8;

5] Create a class Student with private data members name and three subject marks. Write a friend function calculateAverage (Student) that calculate and displays the average marks.

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

class Student {
string name;
int m1, m2, m3;
public:
Student (string n, int a, int b, int c): name(n),
  m1(a), m2(b), m3(c) {}
friend void CalculateAverage (Student);
};

void calculateAverage (Student s) {
Float avg = (s.m1 + s.m2 + s.m3) / 3;
```

```
cout << " Average marks of " << s.name << ": " << avg
     << endl;
}

int main (){
Student S(" Parth", 90, 93, 91);
calculate Average (S);
return 0;
}
```

O/P :

Average marks of Parth: 91

6] Create three class : Alpha, Beta, Gamma, each with a private data member. Write a single friend function that can access all three and print their sum.

->
```
#include <iostream>
using namespace std;
class Beta ; class Gamma;
class Alpha {
int a ;
public:
Alpha (int val): a(val){}
friend void printsum (Alpha, Beta, Gamma);
};
Class Beta {
int b ;
public:
Beta (int val): b(val){}
friend void printsum (Alpha, Beta, Gamma);
};

Class Gamma {
int c ;
public:
Gamma (int val): c(val) {}
friend void printsum (Alpha, Beta, Gamma);
};

void printsum (Alpha x, Beta y, Gamma z) {
cout << " sum:" << x.a+y.b+z.c << endl;
}

int main (){
Alpha a(10); Beta b(20); Gamma c(30);
printsum (a, b, c);
return 0;
}
```

O/P ;

sum = 60

Qn
26/8

```
int main()
{
sum s1(5)
return 0;
}
```

O/P :

sum = 10
object destroyed

* 
```
#include <iostream>
using namespace std;
class sum
{
int n;
int total;
public:
sum(int num){
n = num;
total = 0;
for(int i=1;i<n;i++)
{
total += i;
}}
sum(const sum and object){
n = obj.n;
total = obj.total;
}
void display(){
cout<<" sum = "<<total;
}
~sum()
{
cout<<" object destroyed";
}
};
int main()
int num;
cout<<" enter a number";
cin>>num;
sum s1(num);
sum s2 = s1;
s2.display();
return 0;
}
```

O/P :

sum = 10
object destroyed
object destroyed.

- WAP to declare class 'student' having data members as name and percentage write constructor to initialize these data members.

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

```
#include <iostream>
using namespace std;
Class student {
int per;
string name;
public:
student()
{
per = 80;
name = "abc";
}
student (const student s)
{
per = s.per;
name = s.name;
}
void display()
{
cout << " name = " << name << endl;
cout << " percentage = " << per << endl;
}
};
int main()
{
Student S1;
Student S2 = S1;
cout << " student 1: " << endl;
S1.display();
cout << " Student S2: " << endl;
S2.display();
return 0;
}
```

O/P:

student 1:
name = abc
percentage = 80
student 2
name = abc
percentage = 80
object destroyed
object destroyed

* WAP

-> 

```
#include <iostream>
using namespace std;
class rectangle
{
int l, w;
public:
Rectangle()
{
l = 2;
w = 1;
}
```

Experiment -6

1] Single inheritance

Create a ~~base~~ class called person with name and age. Derive a class student that adds attribute roll no. Write functions to display all details of the student.

->
```
# includ <iostream>
using namespace std;
class person {
  protected:
string name;
int age;
};
class student : protected person {
private;
 int roll;
public:
void accept ();
cout << " enter name ";
cin >> name;
cout << " enter age ";
cin >> age;
cout << " enter roll no;
cin >> roll;
}
void display () {
cout << " name = " << name;
cout << " age = " << age;
cout << " roll no = " << roll;
 }
};
```

```
Int main () {
student S1;
S1. accept ();
S1. display ();
return 0;
}
```

O/P

enter name : parth
age : 17
roll no : 56
name : Parth
age : 17
roll no : 56

2] Multiple Inheritance

->
```
# include <iostream>
using namespace std;
class academic {
protected:
int marks;
};
class sports {
protected:
int score;
};
class result: protected academic, protected sports
{
```

```
class car : public vehicle
{
public:
 string type;
};
class car : public vehicle
{
public :
string type ;
};
class ~~electronics~~ electricars : protected car
{
private :
int battery capacity ;
public:
void accept ()
{
cout << " enter vehicle brand ";
cin >> brand;
cout << " enter vehicle model ";
cin >> model;
cout << " enter car type ;
cin >> type;
cout << " enter battery capacity ";
cin >> battery capacity;
}
void display ()
{
cout << " vehicle brand = " << brand;
cout << " vehicle model = " << to model;
cout << " car type = " << type;
cout << " battery capacity = " << battery capacity;
  }
};
int main ()
{
 eletriccar ecl ;
ecl . accept ();
ecl . display ();
 return 0;
}
```

O/P:

enter vehicle brand : ~~BMW~~ Tesla

enter vehicle model : ~~X1~~ 3

enter car type : ~~se~~ sidan

enter battery capacity : 57.9

vehicle brand = Tesla vehicle model = 3

car type = sidan battery capacity = 57.9

4] Hierarchial inheritance

```
#include <iostream>
using namespace std;
class employee:
{
protected:
string name;
int emp-id;
};
```

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) {
void l=len;
b = br;
a = l*b;
}
void calc ( int si ){
s = si;
a = s*s;
}
};
int main() {
xyz a;
cout<<" Enter side of square :";
cin>> b.s;
b.calc (b.s);
cout<<" Area of square :"<< b.a << endl;
return 0;
}
```

O/P : Enter length and breadth of rectangle 10
5
Area of rectangle : 50

Enter side of square : 5
Area of square : 25

2] Calcut 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 <<endl;
```

Q8] WAP to count digits of space using file handling.

```
#include <iostream>
#include <fstream>
using namespace std;
int main () {
fstream file ("file.txt");
if (!file) {
 cout << "unable to open file ";
 return 0;
}
char ch;
 int digit_count =0, space_count =0;
while (file.get (ch)) {
 if (ch >='0' && ch<='9') {
 digit_count ++;
}
else if (ch==' ') {
   space_count ++;
 }
}
cout << "digits :" << digit_count << endl;
cout << "spaces :" << space_count << endl;
file.close ()
 return 0;
}
```

O/P :

Q9] WAP to count words using file handling.

->
```
#include <fstream>
#include <iostream>
using namespace std;
int main () {
 fstream file;
file.open ("file.txt", ios::in);
if (!file) {
cout << "unable to open file ";
 return 1;
}
char ch;
int word_count =0,
int prev_type =0;
while (file.get (ch)) {
 int (current_type =(ch>='a' && ch<='z') ||
(ch>= 'A' && ch<='Z') || (ch>='0' &&
          ch<='9'), 1:0;
 if (current_type ==1 && prevtype==0) {
    wordcount ++;
}
prev_type =current type;
 }
cout<<"word count:" << word_count << endl;
file.close ();
 return 0,
 }
```

O/P :

Q] Write a C++ program to build simple calculator using class template.

```
#include <iostream>
#include <cmath>
using namespace std;
template <class T>
class calculation calculator {
private:
  T num1, num2;
public:
  Calculator (T n1, T n2) {
     num1 = n1;
     num 2 = n2;
  }
  T add () {
    return num2 + num 2;
  }
  T sub () {
    return num1 - num2;
  }
  T mul () {
    return num1 * num2;
  }
  T divide () {
  if (num 2 == 0) {
  cout << " error ! " << end l;
  return 0;
  } return num1 / num2;
  }
  T void display number () {
  cout << " Numbers : " << num1 << num2 << end l;
  };
int main () {
  double a, b;
int choice;
  cout << " enter 2 numbers : ";
  cin >> a >> b;
calculator < double > calc (a, b);
cout << " Choose an operation ",
cout << " 1. Addition \n";
cout << " 2. Subtraction \n";
cout << " 3. Multiplication \n";
cout << " 4. Division \n";
  cin >> choice;
  switch (choice) {
case 1:
  cout << " Result : " << calc. add (a, b) << end l;
  break;
case 2:
  cout << " Result : " << calc. sub (a; b) << end l;
case 3:
  cout << " Result : " << calc. mul (a, b) << end l;
  break;
  case 4:
  cout << " Result : " << calc. div (a, b) << end l;
  break;
  default;
    cout << " invalid choice " << end l;
}
```

```
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

E] To display the vector in 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 )

12/11

O/P:

10 pushed to stack
20 pushed to stack
30 pushed to stack
Stack elements : 10 20 30
30 poped
Stack elements 10 20

Experiment 11

a) Write a c++ program to implement generic vector To modify the value of a given element

→

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

6] Implement Sorting & searching with user defined records such as personal record (name, birth-date, telephone no) item record (item code, item name, quantity and cost).

```
-> #include <iostream>
   #include <vector>
   #include <algorithm>
   using namespace std;
   struct Person {
     string name;
     int age;
     Personal (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> people ={
        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 searchAge = 28;
   int foundIndex = -1;
    for (int i=0; i< (int) people.size (); i++){
       if (people [i].age == search_Age ? 1:0){
        foundIndex = i;
   break;
     }
   }
    if (foundIndex != -1)
       cout << " person with age " << search_Age <<
    " found " << people [ foundIndex.name << "\n";
    else
      cout << " person with age " << search_Age <<
                 " not found ";
    return 0;
   }
```

O/P:

```
Sorted records by Age :
Bob - 25
David - 28
Alice - 30
Charlie - 35
Person with age 28 found : David
```

```
return 0;
}
```

O/P:

```
enter two numbers: 2 4
Choose an operation : 1
1. Addition
2. Subtraction
3. Multiplication
4. Division
Result : 6

Choose an operation : 3
1. Addition
2. Subtraction
3. Multiplication
4. Division
Result : 8
```

d] Write a C++ program to implement push and pop methods from stack using class template.

```
#include <iostream>
#include <vector>
using namespace std;
template <typename T>
class Stack {
private:
    vector <T> elements;
public:
    void push (T item) {
        elements.push_back (item);
        cout << "item " << " pushed to stack \n";
    }
    void pop() {
        if elements.empty() {
            cout << " stack is empty. Cannot pop \n";
            return ;
        }
        cout << " elements.back () << " poped \n";
        elements.pop_back ();
    }
    void display () {
        cout << " stack elements :";
        for (auto item : elements)
            cout << item << " ";
        cout << endl;
    }};
int main() {
    Stack <int> s;
    s.push (10);
    s.push (20);
    s.push (30);
    s.display ();
    s.pop ();
    s.display ();
    return 0;
}
```

Experiment - 12

a] Write a 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 << " Top element 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";
}
return 0;
}
```

O/P:

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.push pop(20);
q.push (30);
cout << " Front elements : " << q.front () << endl;
cout << " back elements : " << q.back () << endl;
q.pop ();
cout << " after pop operation : " << 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 operation : 20
queue size : 2
queue is not empty

O/P:
Enter string : Hello
Enter string : World
Concatenated string : Hello World

1] Write a CPP code to create a base class Ilogin having data members name and password. Declare accept function virtual. Derive email login and membership login classes from Ilogin. Display email login and membership login details of the emp

->

```
#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 << endl;
  }
};
Class Membership login : virtual public Ilogin {
public:
```

Experiment - 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 << endl;
}};
int main() {
abc s1, s2, r;
s1.acc();
s2.acc();
r = s1 + s2;
r = disp();
return 0;
}
```

Experiment :-10

a] Write a program to find sum of array elements using function template (eg : pass integer, float & double array of 10 element)

->

```
#include <iostream>
using namespace std;
template <class T> T func (T a[], int n) {
  T sum = 0;
  int i;
  for (i=0 ; i<n ; i++) {
    sum = sum + a[i];
  }
  return sum;
}
int main () {
  int a1[3] = {2, 0, 2};
  float a2[3] = {1.2, 2.2, 2.4};
  double a3[3] = {10, 3, 20, 5, 30, 5};
  cout << "sum OF integer array is : " << func(a1,3)
  << endl;
  cout << "sum OF float array is : " << func (a2, 3)
  << endl;
  cout << "sum of double array is : " << func (a3,3)
  << endl;
  return 0;
}
```

O/P

Sum of integer array is : 4

Sum OF float array is : 4.8

Sum OF double array is : 61.5

b] Write a C++ program of square function using template specialization.
calculate the square of integer no. and a string.
Write a specialized functions for the square of a string

->

```
#include <iostream>
using namespace std;
template <class T> T square (T x) {
  T result;
  result = x * x;
  return result;
}
template <> string square <string> (string ss) {
  return (ss+ss);
}
int main () {
  int i = 2, ii;
  string ww ("Parth");
  ii = square <int> (i);
  cout << "i : " << ii << endl;
  cout << square <string> (ww) << endl;
  return 0;
}
```

O/P :

2 : 4

ParthParth

```
void showMembership ()
{
cout << "name :" << password << endl;
}
};
class Employee : public Email login , public Membership
                                                    Login
{
public:
  void input ()
void accept ();
}
void display ()
{
  showEmail ();
  showMembership ();
}
};
int main ()
{
Employee e;
e.input ();
e.display ();
return 0;
}
```

12/11

Experiment - 9

a) Write a program to copy the contents of one file into another. Open "first.txt" in read (ios::in) mode and "second.txt" file in write (ios::out) mode. Copy the contents of 'first.txt' into 'second.txt'. Assume 'first.txt' is already created

```
#include <fstream>
#include <iostream>
using namespace std;
int main ()
{
fstream file1, file2;
char ch;
file1.open ("first.txt", ios::in);
file2.open ("second.txt", ios::out);
while (file1.out (ch))
{
        file2.put (ch);
}
cout << " file copied successfully." << endl;
file1.close ();
file2.close ();
return 0;
}
```

O/P :

file copied successfully