Q1 Write a C++ program for addition of two numbers.

⇒
```
#include <iostream.h>

void main()
{
int a, b, c ;
std::cout << "enter value of a & b ";
std::cin >> a >> b;
c = a+b;
std::cout << "Addition = " << c;
}
```

Q2 Program to check number is even or odd

=>
```
#include <iostream>

int main() {
int a;
std :: cout << "Enter a number :" ;
std :: cin >> a ;
if (a % 2 == 0)
{std :: cout << "The number is even" ;}
 else
{std :: cout << " The number is odd ", }}
```

Q.3 C++ code to print 1 to 10 numbers using for loop.

=>
```
#include <iostream>

int main()
{int i;
for (i=1; i<=10; i++)
{std::cout<< "\n" i; }}
```

Q.4 C++ code to print 10 to 1 using while loop.

=>
```
#include <iostream>

int main(){
int i=11;
while(i>1)
{i--;
std::cout<<"\n"<<i;}}
```

Q5. C++ code to print below pattern.

a)
```
#include <iostream>

using namespace std;
int main(){
int r= 3;
for (int i=1; i<=r; i++)~~}~~{
for (int j=1; j<=i; j++){
std::cout << "*" ; }
cout << rnd;}
```

b)
```
# include <iostream>

using namespace std;
int main (){
int r = 5;
for (int  i=1; i<=r ; i++){
for (int j =1; j<=i , j++){
~~sd~~cout << i << " "  ;}}
```

c)
```
#include <iostream>

using namespace std;
int main (){
int r=5;
for (int i =1; i<=r; i++)
{
for (int j =1; j<=i ;j++)
{
~~to~~ std :: cout <<i<< "  "; }}
```

[illegible]
8/7/25

Date
Page

EXPERIMENT-1.

1. Write a C++ program to declare a class student having data members as roll_no & name. Accept & display data for single student.

⇒
```
# include <iostream>
using namespace std;
class student
{
int roll_no;
string name;
public:
void accept()
{
cout <<"Enter student name & Roll_No : ";
cin>> name >>roll_no; }
void disp ()
{
cout <<"Name of student : " <<name ;
cout <<"Roll_No of student : " << Roll_No;}};
int main ()
{ student s1;
 s1.accept();
 s1.disp();
 return0; }
```

O/P-

```
Enter student Name & Roll. No : Palak 85
Name of student : Palak
Roll. No of student : 85
```

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

⇒
```
# include <iostream>
using namespace std;
class book {
public :
int bookid;
string bookname;
float bookprice;
public:
void accept () {
cout << "Enter bookid , book name & book price:";
cin >> bookid >> bookname >> book price; }
void disp() {
cout << "Enter book id :" << bookid;
cout << "\n Enter book name :" << bookname;
cout << "\n Enter book price :" << bookprice; }};
int main() {
book b1, b2 ;
b1. accept ();
b2. accept () ;
if (b1.bookprice > b2.bookprice){
b1.disp();}
else{
b2.disp();}
return 0;}
```

O/P-

```
Enter book id, book name & book price:1720 XYZ
2000
Enter book id, book name & book price:0508 ABC
4000
Enter book id : 508
Enter book name : ABC
Enter book price : 4000
```

3 Write a program to declare a class time having data members as Hours, minutes & seconds. Accept data for one object & display total time in seconds.

⇒
```
#include<iostream>
using namespace std;
class Time{
public:
int H, M, S;
void accept(){
cout<<"Enter time in Hours, Minutes & Seconds:"
cin>>H>>M>>S;
H = H * 3600
M=M*60;
S=S+H+M;}
void disp(){
cout << "Total time in seconds:" <<S;}};
int main(){
Time T1;
T1.accept();
T1.disp();
return 0;}
```

O/P-

```
Enter time in Hours, Minutes & Seconds:1 20 17
Total time in seconds: 4817.
```

[illegible]
15/7

22.7.25 EXPERIMENT-2

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

→
```
#include <iostream>
using namespace std;
class city
{public:
 string name;
 int population;
 void accept()
 {cout << "Enter city name:";
  cin >> name;
  cout << "Enter city population:";
  cin>> population; }
  void disp()
 {cout <<"City having highest population:"<<
  name; }};
 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].disp();  return 0; }
```

O/P-

```
Enter city Name :   Mumbai
  "    "   Population: 789
  "    "   Name : Pune
  "    "   Population : 567
  "    "   Name : Jaipur
  "    "   Population: 389
  "    "   Name : Kharagpur
  "    "   Population : 456
  "    "   Name : Kolkata
  "    "   Population: 987
City having ~~Name~~ highest population Kolkata.
```

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

→
```
# include <iostream>
using namespace std;
class account
{public :
 int acc_no ;
 float balance;
 void accept
 {cout << "Enter Account no : ";
   cin >> acc_no;
  cout << "Enteraccount balance : ";
cin >>balance; }
void disp()
{cout<<"\n Account no. :"<<acc_no;
 cout<<"\n Account balance :"<<balance ;}};
int main()
{account A[10];
 int i;
 for(i=0; i<10; i++)
 {A [i].accept();}
 for(i=0; i<10; i++)
 { if(A[i].balance >=5000)
 { A[i].balance = A[i].balance+(0.1 *A [i].
   balance);
   A[i].disp();}}
   return 0; }
```

O/P-

```
Enter Account No : 2564
Enter Account Balance: 2095
   "        "        No. : 4879
   "        "        Balance : 7890
   "        "         No. : 6789
   "        "         Balance : 4567
   "        "          No. : 3578
   "        "          Balance 2478
   "        "           No. 3678
   "        "           Balance : 2634
   "        "            No. : 9705
   "        "            Balance 2634.
"    "    No. : 9705
"    "    Balance 5738
"    "    No. : 57228
"    "    Balance 5835
"    "    No. : 3789
"    "    Balance : 4627
"    "    No. : 8906
"    "    Balance : 4689.
```

```
Account no. :
"     Balance : 790
"      no. : 7467.9
"      balance : 2634
"      no. : 10675.5
"      balance : 5738
"      no. : 629508
"      balance : 4627.
```

3. WAP to declare a class 'Staff' having data members as name & post. Accept this data for 5 staff & display names of staff who are "HOD".

→
```
#include <iostream>
using namespace std;
class staff
{ string name;
  public:
  string post;
void accept()
{cout << "Enter the staff name :";
 getline(cin, name);
 cout << " Enter the staff post :";
 getline(cin, post); }
 void disp()
 {cout << "\n Staff Name :" << name;
  cout << "\n Staff Post :" << post; }};
  int main()
  {staff s[5];
   int i;
   for (i=0; i<5; i++)
   {s[i].accept(); }
     For(i=0, i<5; i++)
   {if (s[i].post == "HOD")
   {s[i].disp(); }}
   return 0; }
```

o/p :

```
Enter the staff name : sheetal
  "     "     "   post : lecturer
  "     "     "   name : Sushma
  "     "     "   post : lecturer
  "     "     "   name : Jayshree
  "     "     "   post  HOD
  "     "     "   name : Hemlata
  "     "     "   post : Dean
  "     "     "   name : Mugdha
  "     "     "   post : HOD
```

Staff Name: Jayshree
" Post: HOD
" Name: Mugdha
" Post: HOD

[illegible]
5/8

25.7.25 EXPERIMENT-3

1 WAP to declare a class 'book' containing data members as book title, authorname & price, Accept & display the info for one object using a ptr. to that object.

→
```
#include <iostream>
using namespace std;
class book
{string book_title;
 string author_name;
  int price;
  Public:
 void accept()
 {cout<<"Enter Book Title:";
  getline (cin, Book title);
  cout << " Enter Author Name:";
   getline (cin, author_name);
   cout<<" Enter book price:";
   cin >> price;}
  void disp()
 {cout << " Book Title:" << book_title;
  cout<<"\n Author Name:"<< Author_name;
  cout<<"\n Book price :" << ~~boo~~ price;}};
  int main()
  {book B1;
   book * p;
   p=&B1;
   p→accept();
   p→disp();
    return 0;}
```

O/P-

```
Enter Book Title: Atomic Habits
Enter Author Name: James Clear
Enter Book price: 599
Book title: Atomic Habits
~~Bo~~ Author Name: James Clear
Book price: 599
```

2. WAP to declare a class 'student' having datamembers as roll_no. & percentage. Using 'this' pointer invoke member functions to accept & display this data for one object of the class.

→
```
#include<~~st~~iostream>
using namespace std;
class student
{int roll_no;
 float percentage;
 public:
  void accept()
  {cout <<"Enter the Student Roll No:")
   cin >>roll_no;
  cout << "Enter student Percentage:";
   cin >> percentage;}
  void disp()
  {this → accept();
   cout<<"\n Roll No of the student :"<<roll_no;
   cout << "\n Percentage of the student :"<<
    percentage;}};
```

```
int main()
{ Student S1;
  S1.disp();
  Return 0;}
```

O/P -

```
Enter student Roll No. 85
Enter student percentage: 98.5

Roll No. of the student: 85
Percentage of the student: 98.5
```

3. Write a program to demonstrate the use of nested class.

```
→ #include <iostream>
using namespace std;
class Student {
string name;
int roll_no;
public:
void accept() {
cout << "Enter the Student name: ";
getline (cin, name);
cout << "Enter the Student roll no.: ";
cin>>roll_no;
}
class marks {
  public:
```

```
int phy.
int maths;
int total_p;
int total_m;
void accept(){
cout<<"Enter the total marks for physics: ";
cin>>total_p;
cout<<"Enter the marks obtained in physics: ";
cin>>phy;
cout<<"Enter the total marks for maths: ";
cin>>total_m;
cout<<"Enter the marks obtained in maths: ";
cin>>maths;
}
void disp(){
float total=0;
float percentage;
total=total_p+total_m;
percentage=((phy+maths)/total)*100;
cout<<"\n Percentage obtained by the student ="
<<percentage<<" / "; }};
};
int main(){
student s;
s.accept();
student::marks m;
m.accept();
m.disp();
return 0;}
```

O/P-

```
Enter the student name: Palak Navani
Enter the student roll no. 85
Enter the total marks for physics :100
Enter the marks obtained in physics :99
Enter the total marks for maths :100
Enter the marks obtained in maths :98

Percentage obtained by the student =98.5%
```

[illegible]
5/8

Date________
Page________

EXPERIMENT- 4

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

→
```
#include <iostream>
using namespace std;
class numbers {
      private:
      int a, b;
      public:
      int temp ;
      void accept {
         cout<<"Enter the first number: ";
         cin>>a;
         cout<<"Enter the second number: ";
         cin>>b;
      }
      void disp() {
           cout<<"\n After swapping: "<<endl;
           cout<<" First number =  "<<a<<endl;
           cout<<" Second  number= "<<b;
      }
      void swap(Numbers &n) {
              n.temp = n.a;
               n.a= n.b;
               n.b= n.temp;
       }
};
```

```
int main () {
        numbers n;
        n.accept();
        ~~n.swap();~~ swap(n);
        n.disp();
        return 0;
}
```

→ O/P- Enter the first number : 9
Enter the second number: 2
After swapping:
First Number = 2
Second Number = 9

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

→
```
# include <iostream>
using namespace std;
class number {
        private:
        int a, b, temp;
        public:
        void accept() {
            cout<<"Enter the first number :";
            cin>>a;
            cout<<"Enter the second number:";
            cin>>b;
        }
        void display() {
```

```
    cout<<"\n After swapping:"<<endl;
    cout<<"First Number : " <<a <<endl;
    cout<<" Second Number: "<<b;
  }
  friend void swap(number &n);
};
void swap(number &~~x~~n) {
        n.temp = n.a;
        n.a = n.b;
        n.b = n.temp;
}
int main() {
      number n;
      n.accept();
      swap(n);
      n.disp();
      return 0;
}
```

→ O/P- Enter the first number : 7
Enter the second number: 4
After swapping :
First number = 4
Second number = 7

3) WAP to swap two numbers from different class using friend function.

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

```
class num2;
class num1 {
    private:
    int a;
    public:
    void accept() {
        cout<<"Enter the first number: ";
        cin>>a;
    }
    friend void swap(num1 &, num2 &);
};
class num2 {
    private:
    int b;
    public:
    void accept() {
        cout<<"Enter the second number : ";
        cin>>b;
    }
    friend void swap(num1 &, num2 &);
};
void swap(num1 &x , num2 &y) {
    int temp;
    temp=x.a;
    x.a=y.b;
    y.b= temp;
    cout<<"\n After swapping: "<<endl;
    cout<<" First number = "<<x.a<<endl;
    cout<<" Second number ="<<y.b;
}
```

```
int main() {
    num1 n1;
    num2 n2;
    n1 accept();
    n2 accept();
    swap(n1, n2);
    return 0;
}
```

→ O/P-
```
Enter the first number: 5
Enter the second number: 6
After swapping:
First Number = 6
Second Number = 5
```

4) WAP to create two classes Result1 & Result2 which stores the marks of the students. Read the value of marks for both the class objects & compute the average of 2 results.

→
```
#include <iostream.h>
using namespace std;
class result2;
class result1 {
  private:
   string name;
   float marks;
   public:
   void accept() {
      cout<<"Enter student name:";
```

```
    getline (cin, name);
    cout<<" Enter the total marks obtained in
    first sem out of 100:";
    cin>> marks;
  }
 friend void average(result1, result2);
};
class result2 {
    private:
               marks
    float ~~average~~;
    public:
    void accept() {
        cout << "Enter the total marks obtained in
        second sem out of 100:";
        cin >> marks ;~~)~~
    }
    friend void average(result1, result2);
};
  void average(result1 x, result2 y) {
     float avg;
     avg =(x.marks + y.marks)/2 ;
     cout<<"\n Average of both the results="<<
     avg;
}
int main() {
    result r1;
    result r2;
    r1.accept();
    r2.accept()
    average (r1, r2);
    return 0;
}
```

O/P → Enter student name: Palak Novani
Enter the total marks obtained in first sem out of 100 98
Enter the total marks obtained in second sem out of 100: 96
Average of both the results = 97.

5) WAP to find the greatest number among the two numbers from two different classes using friend function.

→
```
#include <iostream>
using namespace std;
class num2;
class num1 {
    private:
    int a;
    public:
    void accept() {
      cout<<" Enter the first number:";
      cin>>a;
    }
    friend void grt (num1,num2);
};
class num2 {
     private:
     int b;
     public:
     void accept() {
        cout<<"Enter the second number:";
        cin>>b;
    }
    friend void grt(num1, num2);
};
void grt(num1 x, num2 y){
    if (x.a >y.b) {
        cout<<x.a <<" is greater than" <<y.b;
    }
    else {
        cout<<y.b<<" is greater than"<<x.a;
    }
};
int main() {
        num1 n1;
        num 2 n2;
        n1 accept();
        n2 accept();
        grt(n1, n2);
        return 0;
}
```

→ O/P- Enter the first number: 8

```
Enter the second number:10
10 is Greater than 8.
```

FRIEND FUNCTION PRACTICE.

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

→
```
#include <iostream>
using namespace std;
class B;
class A {
   private:
   int a;
   public:
   void accept() {
      cout << "Enter the first number: ";
      cin >> a;
  }
  friend void ~~swap~~ sum (A, B);
};
class B {
   private:
   int b;
   public:
   void accept() {
      cout<<"Enter the second number: ";
      cin>>b;
   }
   friend void sum(A,B) ;
};
```

```
void sum(A x, B y) {
    int sum,
    sum=x.a+y.b;
    cout<<"\n Sum of the two numbers ="<<sum;
}
int main() {
    A n1;
    B n2;
    n1.accept();
    n2.accept();
    sum(n1,n2);
}
```

O/P-
```
Enter the first number: 5
Enter the second number: 6
Sum of two numbers: 11
```

2) WAP with a class Number that contains a pvt. integer. Use a friend function to swap number to swap private values of two number objects

→
```
# include <iostream>
using namespace std,
class Number {
        private:
        int a, b;
        public:
        void accept() {
            cout<<" Enter the first no. : ";
            cin>>a;
    }
    void accept1() {
        cout<<" Enter the second no ";
        cin>>b;
    }
    void disp() {
        cout <<"\n After swapping : "<<endl;
        cout <<" First no = "<<a <<endl;
    }
    void disp1() {
        cout<<" Second no. = "<<b;
    }
    Friend void swap(Number & x, Number &y);
};
void swap(Number &x, Number &y) {
    int temp;
    temp=x.a;
    x.a=y.b;
    y.b=temp;
}
int main() {
    Number n1;
    Number n2;
    n1.accept();
    n2.accept1();
    swap(n1, n2);
    n1.disp();
    n2.disp1();
    return 0;
}
```

```
O/P- Enter first no : 7
     Enter second no. 4
     After swapping:
     First no. = 4
     Second no. = 7
```

3) Define two classes Box & Cube, each having a private volume WA friend function. Find (Greater box cube) that determines which object has a larger volume

→
```
#include <iostream>
using namespace std;
class Box;
class Cube {
        private:
        float volume;
        public:
        void accept () {
          cout << "Enter the vol. of a cube:";
          cin>>volume;
        }
        friend void find grt (cube, box);
};
class Box {
        private:
        float vol;
        public:
        void accept () {
          cout << " Enter the vol. of a box:";
          cin>>vol;
        }
        friend void ~~gr~~ find grt (Cube, Box);
};
```

```
void find grt (Cube x, Box y) {
    if (x. volume > y. vol) {
      cout << "\n Cube has a larger vol "
    }
    else {
      cout << "\n Box has larger vol. "
    }
}
int main() {
    Cube c;
    Box b;
    c. accept();
    b. accept();
    find grt(c, b);
    return 0;
}
```

O/P →
```
Enter the vol. of a cube: 567.88
Enter the vol. of a box: 985.66
Box has larger vol
```

4) ~~WAP~~ Create a class Complex with real & imaginary parts as private members. Use a frnd func. to add 2 complex no.'s & return the result as a new complex object.

```
→ #include <iostream>
using namespace std;
class complex {
    private:
    int r, i;
    public:
    void accept() {
     cout << "Enter the real part : ";
      cin >> r;
      cout << "Enter the imaginary part : ";
      cin >> i;
    }
    void disp() {
     cout << r << " + " << i << "i" << endl;
    }
    friend complex sum (complex, complex)
};
complex sum (complex x , complex y) {
    complex temp;
    temp.r = x.r + y.r;
    temp.i = x.i + y.i;
    return temp;
}
int main() {
   complex c1, c2, c3;
   cout << "Enter the first complex no : ";
   c1.accept ();
   cout << "   "   "  second  "  " : ";
   c2.accept ();
cout <<"# Complex No 1: ";
C1 disp ();
cout << "  "   "  "  2 : ";
C2 disp ()
C3 = sum (C1 , C2 );
cout <<" In sum of the two complex
  no's = ";
C3 disp
}
```

5> Create a class student with private data members. name & 3 subject marks. Write a friend func. calculate avg (student) that calculates & displays the avg marks

```
→ #include <iostream>
using namespace std;
Class Student {
    string name;
     int math;
     int chem;
     int phy;
     public:
     cout <<" Enter stud. name : ";
     getline (cin , name);
     similarly for marks obtained in
      maths, physics.
     cout <<" Enter marks obtained in chem : ";
     cin>> chem ;
}
```

```
friend void calculate avg(student); {
};
void calculate avg (student x) {
  float avg;
  avg = (x math + x phy + x chem) /3;
  cout <<"\n Avg marks = "<<avg;
}
int main() {
     student s;
      s accept();
      calculate avg (s);
      return 0;
}
```

[illegible]
23/9/25

EXPERIMENT-5

1) 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

→ parameterized constructor

```
→ #include <iostream>
using namespace std;
class sum {
    int n;
    public:
    sum(int num) {
        int s=0;
        int i=0;
        n=num;
        for(int i=1; i<=n; i++){
            s= s+i;
        }
        ~~void disp(){~~
            cout<<"Sum ="<<s;
        }
    }
};
int main() {
   sum s1(5);
}
```

→ copy constructor

```
→ #include <iostream>
using namespace std;
class sum {
    int n, s=0;
    public:
    sum() {
       n=4.
    }
    sum(sum &s) {
       n= s.n;
       int i,
       for (i=1, i<=n, i++) {
            s = s+i;
       }
       void display() {
           cout << " Sum= " <<s~~um~~;
       }
    }
};
int main() {
   sum s1;
   sum s2(s1);
   return 0;  → s2. display();
}
```

```
→ #include <iostream> → default.
using namespace std;
class sum {
   int n, s;
   public:
   sum() {
      n=5;
      s=0;
```

```
    for(i=1; i<=n, i++){
    s=s+i;
    }
 }
 void display(){
   cout<< "sum="<<s;
 }
};
int main(){
   sum s1;
   s1.disp();
}
```

2) WAP to declare a class "Student" having data members as ~~roll no~~ name & percentage. Write a constructor to initialize these data members. Accept and display data for one student.

```
→ #include <iostream>
  #include <string>
  using namespace std;
  class Student {
     private:
     string name;
     float percentage;
     public:
     Student(String n, float p) {
        name=n;
        percentage=p;
        cout<<"Name:";
        getline(cin, Name);
        cout<<"Percentage:";
        cin>>percentage;
      }
      void display() {
         cout<<"Student name: "<<name<<endl;
         cout<<"Student percentage: "<<percentage
      }
};
int main() {
      Student s1("Palak", 90);
      s1.display();
}
```

O/P-

```
Name: Palak
Percentage: 90
Student Name: Palak
Student Percentage: 90
```

3) Define a class 'CollegeMembers' variables as roll no, name, course WAP using constructor with default value as computer engineering for course. Accept this data for 2 objs. of class & display the data.

→
```
#include <iostream>
#include <string>
using namespace std;
class college {
    private:
```

```
    string course_name, stud_name;
    int roll_no;
    public: ~~x~~
    ~~v~~ college() {
        roll_no = 36;
        stud_name = "Palak";
        course_name = "AIDS";
    }
    void display() {
      cout << "Roll no : " <<roll_no <<endl;
      cout << " stud name : << stud_name <<endl,
      cout <<" course name : " << course_name <<endl;
    }
};
int main() {
    college c1;
    c1. display
    return 0;
}
```
O/P -

```
Roll no. = 36
Stud_name = Palak
Course_name = AIDS
```

4) WAP to demonstrate constructor overloading.

→
```
#include <iostream>
using namespace std;
class rectangle {
      int l,b;
      public:
      rectangle(){
          l=2;
          b=5;
      }
       rectangle(int x){
           l=x;
       }   b=x;
        rectangle(int x, int y){
            l=x;
            b=y;
        }
        void calculate(){
           cout<<"The area is: "<<(l*b)<<endl;
        }
    };
    int main(){
        rectangle r1;
        r1. calculate();
        rectangle r2;
        r2. calculate();
        rectangle r3;
        r3. calculate();
    }
   O/P-
  The area is- 10
  The area is- 9
  The area is- 36.
```

classmate
Date
Page

EXPERIMENT-6

→ multilevel inheritance

```
1. # include <iostream>
# include <string>
using namespace std;
class Person {
    protected:
    int age;
    string name;
};
Class student : public person {
        private:
        int roll_no ;
        public:
        void accept() {
            cout<< "Enter your name: ";
            cin>> name;
            cout<<"Enter your age:";
            cin>>age;
            cout << "Enter your roll no : ";
            cin>> roll_no ;
        }
        void display () {
            cout << "Name\n" <<name;
            cout << " Age:\n " << age;
            cout << " Roll_no : \n " <<roll_no;
        }
    };
    int main() {
        student S1;
          S1.accept();
  }       S1.display();
```

O/P-

```
Enter your name Palak
Enter your age: 17
Enter your roll. no 36
Name: Palak
Age : 17
Roll no : 36
```

2 Multiple inheritance

```
→ #include <iostream>
using namespace std;
class Academic {
      protected:
      int marks;
};
class Sports {
   protected:
   int score;
};
class Result {: protected Academic, protected sports{
        int total score = 0;
        public:
        void accept() {
            cout<<"Enter marks of student:";
            cin>>marks;
            cout<<"Enter sports score of student:";
            cin>> ~~sports~~ score;
        }
        void calculate () {
```

```
        total score = marks + score;
        cout << " The marks of the student is "<<marks;
        cout << " The sportsscore of the student is: "<<
                                                     score;

        cout << " The total score of the student is:"<<
        total score;
      }
};
int main(){
    Result r;
    r.accept();
    r.calculate();
}
```

O/P-

```
Enter the marks of the student 99
Enter the sports score of the student :90
The marks of the student : 99
The sportscore of the student :90
The totalscore of the student: 189
```

3) Heirarchial inheritance.

```
→ #include <iostream>
using namespace std;
class vehicle {
      protected:
       string brand;
       int model;
};
```

```
class car : protected vehicle{
  protected:
  string type;
};
class Electriccar : protected car {
    int batterycapacity;
    public:
    void accept() {
      cout<<"Enter the brand, model, type, batter
      capacity : ";
      cin >> brand>>model >> type >> batterycapaci
    }
    void display () {
      cout<<"\n The brand of the car : "<<brand
      endl;
      cout<<"\n The model of the car : "<<model
      endl;
      cout<<" The type of the car : "<< type<<
      endl;
      cout<<" The battery capacity of the car"<<
      battery capacity;
    }
};
int main() {
    Electriccar e;
    e.accept();
    e.display();
}
```

O/P

Classmate
Date
Page

4> hybrid inheritance

```
→ # include <iostream>
# include <string>
using namespace std;
Class Person {
    Public:
    string name;
    ~~st~~ int age;
    void getperson Details() {
        cout << "Enter name : ";
        cin>> name;
        cout<<" Enter Age : ";
        cin>>age;
    }
    void showdetails () {
        cout << "Name : " << name <<" Age : " <<age<<
                                              endl;
    }
};
class Student : public person {
    Public:
    String course;
    void getDetails() {
        cout << " Enter course : ";
        cin>>course;
    }
    void showdetails () {
        cout<< " Course : " <<course <<endl;
    }
};
```

```
{void showDetails()
    cout <<"Course : "<<course <<endl;
 }
};
class Employee : public Person
{
   public:
   string company;
   {
     void getEmployee details()
     cout <<"Enter company    ";
     cin >> company;
   }
   void show Employeedetails()
  {
    cout<<"Company    "<< company<<endl;
  }
};
class Intern : public Student, public Employee{
   public:
   void show Interndetails   ()
   {
     cout<<"\n -- Intern Details --\n ";
     student :: showPersonDetails();
     showEmployeeDetail ();
   }
};
```

```
intmain()
{
  Intern it;
  it.Student::getPersonDetails();
  it.getdetails();
  it.getemployeeDetails();
  it.showinterndetails();
  return 0;
}
```

O/P -

```
Enter name: Palak
Enter age: 21
Enter course : Computer Science
Enter company Microsoft
-- Intern Details --
Name: Palak, Age: 21
Course: Computerscience
Company: Microsoft
```

[illegible]
11/11

Date
Page

EXPERIMENT-~~8~~7

```
1 # include <iostream>
Using namespace std;
class Area ;
{
    private:
    float l, b;
    public:
    void area(float l)
    {
        float area;
        area = l* l;
        cout<<"Area of square: "<<area<<endl;
    }
    void area (float l, float b)
    {
        float area;
        area = l* b;
        cout<<"Area of rectangle: "<<area<<endl;
    }
};
int main ()
{
    Area a1;
    a1.Area(8);
    a1.area(4, 12);
}
```

O/P-

```
Area of square : 64
Area of rectangle: 848
```

2.
```
#include <iostream>
using namespace std;
class Sum {
    private:
    int a, b, c, d, e, f, g, h, i, j;
    float k, l, m, n, o;
    public:
    void sum ( float k, float l, float m, float n,
                float o) }
     float sum;
     sum = k + l + m + n + o;
     cout << " Sum of 5 floating no's  "<< sum;
     void sum (int a, int b, int c, int d, int e, int f,
       int g, int h, int i, int j) ;
       int sum;
       sum = a + b + c + d + e + f + g + h + i + j )
       cout << "sum of 10 integers  "<< sum;
    }
};
int main () {
   sum s1;
   s1 sum (1.2, 3.5, 5.6, 8.4, 1.5);
   s1. sum (1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
}
```
O/P -
```
Sum of 10 integers : 55
sum of 5 floating no.s : 10.2
```

3.
```
#include <iostream>
using namespace std;
class Number {
  private:
  int x;
  public:
  void accept() {
    cout<<" Enter a number : ";
    cin>>x;
  }
  voidoperator - () {
      x = -x;
  }
  void display() {
    cout "Negeted Number : "<<x<<endl
  }
};
int main() {
  Number n1;
   n1 accept();
   -n1;
    n1.display();
}
```
O/P:-
```
Enter a no : 5
Negated no. :-5
```

```
1. # include <iostream>
using namespace std;
class Number {
  private: int x ;
   public:
   void accept() {
     cout << "Enter a no.:";
     cin >> x;
   }
   temp = x;
   void operator ++ () {
       x = ++x;
   }
   void reset () {
       x=temp;
   }
   void operator ++(int) {
      x =x++;
   }
   void display1() {
     cout << "(pre) The no. is:"<< x <<endl;
   }
   void display 2() {
     cout<< "(post) The no is: "<<x;
   }
};
int main() {
 Number n1;
 n1.accept();
  ++n1;
 n1.display();
 n1.rest();
  n1++;
 n1.display 2();
}
```

[illegible]

EXPERIMENT-8

```
1. #include <iostream>
   #include <string>
   using namespace std;
   class Mystring {
     public:
     string str;
     My string operator +(My String& other) {
         MyString temp;
         temp.str = str + other.str;
         return temp;
     }
     void display() {
        cout<<str<<endl;
     }
   };
   int main() {
     Mystring S1, S2, S3;
     S1. str = "xyz";
     S2. str = "pqr";
     S3 = S1 + S2;
     cout<< "Concatenated string :";
     S3.display();
   }

2. #include <iostream>
   #include <string>
   using namespace std;
   class login {
```

```
protected: string name, password;
public: ~~virt~~
virtual void accept() {
    cout<<"Enter name: ";
    cin>>name;
    cout<<"Password: ";
    cin>>password;
  }
};
class Emaillogin: Public login {
    string email;
    public:
    void accept() override {
      login::accept()
      cout<<"Enter email: ";
      cin>>email;
    }
    void display(){
      cout<<"\n--- Email login details---\n";
      cout<<"Name: "<<name<<"Password: "<<
      password<<"Email "<<email<<endl;
   }
};
class Membershiplogin: public login {
     string Membership ID;
     public:
     void accept() override {
     login::accept();
     cout<<"Enter membership ID";
     cin>>membership_ID;
    }
```

```
void display() {
  cout << "\n _ _ Membership login Details _ _ \n";
  cout << "Name: " << name << "\n Password : " <<
  password << "Membership Id : " << membershipID;
}
};
int main() {
  Email login e;
  Membershiplogin m;
  e.accept();
  m.accept();
  e.display();
  m.display();
}
```

[illegible]

EXPERIMENT - 9

1
```
#include <iostream>
#include <fstream>
using namespace std,
int main() {
   fstream first_file, second_file;
   first_file.open ("first.txt", ios::in);
   if (!first_file) ;
   cout<<"Error opening first ~~text~~ !"<<endl ;
   return 1 ;
  }
second_file.open ("second.txt", ios::out );
if(!second_file) {
   cout<<" Error opening second.txt! "<<endl;
   return 1;
}
char ch;
while(first_file.get(ch)) {
 second_file.put(ch);
}
first_file.close(),
second_file.close(),
cout<<"File copied successfully!";
}
```

2
```
#include <iostream>
#include <fstream>
#include <cctype>
using namespace std;
int main() {
   fstream new_file;
   newFile.open("first.txt", ios::in);
   if(!new_file){
      cout<<"Error occured while opening!"<<
      endl;
      return 1;
  }
  int digits_count=0;
  int spaces_count=0;
  char ch;
  while(newFile.get(ch)){
    if (isdigit(ch)){
        digit_count++;
    }
    if(isspace(ch)){
        space_count++;
  }
  new_file.close();
  cout <<"No of digits: "<<digits_count;
  cout <<"No. of spaces: "<<spaces_count;
}
```

```
4. #include <iostream>
   #include <fstream>
   #include <string>
   using namespace std;
   int main() {
```

classmate
Date
Page

```
fstream new_file;
new_file.open("first.txt", ios::in);
if(!new_file) {
    cout<<"Error in opening first.txt "<<endl;
    return 1;
  }
  string target = "Pronoun";
  string word;
  int count = 0;
  while(new_file>>word) {
     if(word=target) {
       count++;
   }
   new file.close();
   cout<<"The word "<<target<<"\""occured"
   << count << " times. << endl;
}
```

[illegible]
11|11

Date ______
Page ______

EXPERIMENT-10

1.
```
#include <iostream>
Using namespace std;
template <typename T>
void sumArr(T arr[], int n) {
  Tsum = 0;
  for(int i=0; i<n; i++) {
     sum += arr[i];
  }
  cout<<"sum 1: " <<sum;
}
int main() {
 int n=7;
 int arr[n] = {1,2,3,4,5,6,7};
 Float arr[n] = {1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7};
 double arr[n] = {1.234, 2.789, 3.456, 4.987,
                  5.999, 6.787, 7.854};
 sumArr(arr, n);
 sumArr(arr1, n);
 sumArr(arr2, n);
}
```

2.
```
# include <iostream>
Using namespace std;
template < typename T>
class stack {
   arr[100];
   int top;
   public:
   stack()  { top = -1; }
```

```
void push (T val) {
if (top ==99) {
cout<<"Stack Overflow! "<<endl; }
else {
  arr[++top] = val;
  cout<<val<<" puched into stack;
  }
}
void pop() {
   if (top == -1) {
     cout<<"stack underflow"); }
     else {
       cout<<arr[top--] <<" popped from stack"
   }
 }
void display() {
  if (top == -1) {
    cout << "Stack is empty"; }
    else {
      cout <<" Stack elements: ";
      for (int i=top; i>=0; i--) {
        cout<<arr[i]<<"    "; }
        cout<<endl; }}};
int main () {
    stack <int> s;
    int choice, vals;
    do {
      cout<<"\n --- Stack menu -- \n";
      cout<<"1. Push \n 2. Pop \n 3. Display
      Exit\n";
```

```
cout<<" Enter your choice:" ;
cin>>choice ;
switch(choice) {
    case 1:
   cout<<" Enter values to push: " ;
    cin>> val;
    s.push(val);
    break;
    case 2:
     s.pop() ;
      break;
    case 3:
     s.display ();
     break;
     case 4:
     cout<<" Exiting... "<<endl;
      break;
     default:
      cout<<" Invalid choice! "<<endl;
  }}
  while(choice!=4);
 }
```

[illegible]
1/11

EXPERIMENT-11

```
1. #include <bits/stdc++.h>
using namespace std;
class vect {
   public:
   vector <int> vec= {10,20,30,40,50,60};
   void modify () {
   int idx, val;
   cout << "Enter pos: ";
   cin>> idx;
   cout << "Enter value: ";
   cin >> val;
   for (int i=0; i<vec.size(); i++) {
      if (vec[idx]= =vec[i]) {
      vec[i] = val;
   }
}
}
void ~~modif~~ multiply() {
   int scal;
   cout<<"Enter value to multiply: ";
   cin>> scal;
   for (int& i : vec) {
      i* = scal;
   }
}
void display() {
  for(int i : vec) {
     cout <<i<< " ";
  }
}
};
```

```
int main() {
  vect a;
  a.display();
  cout<<endl;
  a.modify();
  cout<<endl;
  a.multiply();
  cout<<endl;
}
```

2

```
#include <iostream>
using namespace std;
class vect {
    public:
    vector <int> vec = {10, 20, 30, 40, 50, 60};
    void modify() {
       int idx, val;
       cout<<"Enter pos: ";
       cin>> val;
       if (idx<0 || idx >= vec.size()) {
          cout<<"Invalid syntax! ";
}
auto it = vec.begin()
advance (it , idx);
 it = val;
cout <<"Value modified ";
}
void multiply () {
  int scal;
cout<<"Ente value to multiply: ";
cin>>scal;
for(auto it = vect begin(); it! =vec end();i
*it *=scal;

}
cout<<"All elements Multiplied ";

}
void disp() {
   cout<<"Vector elements: ";
  for (auto it =vec begin(); it! =vec end(); }
  cout<<*it<<"  ";

 }
}
};
int main() {
  vect a;
  a.disp();
  cout<<endl;
  a.modify();
  cout<<endl;
   a.multiply();
   cout<<endl;
   a.display();
 }
```

[illegible]

EXPERIMENT-12

1.
```
#include <stack>
#includ <bit/stdc++.h>
using namespace std;
int main () {
 stack<string> cars;
  cars.push("Audi");
  cars.push ("Mercedes");
  cars.push ("Ferrari");
  cout<<"Top element is: "<<cars.top()<<endl;
  cout<<"size of stack is: "<<cars.size();
  cars.pop();
  cars.pop();
  while(!cars.empty()) {
    cout<<"Elements in stack are: "<<cars.top();
    cars.pop;
 }
}
```

2
```
# include <bits/stdc++.h>
# include <Queue>
using namespace std;
int main() {
  queue<int> age;
  ~~age.push(21);~~
  age.push(22);
  age.push(23);
  age.push(24);
  cout<<" Front element is: "<<age.front();
  cout<<" Back element is: "<<age.back();
  age.pop();
  age.pop();
while(!aqe.empty()){
cout<<"Elements in queue are: "<<aqe.front()
aqe.pop();
}
}
```

[illegible]

||||