7/7/25

Write a C program to add 2 nos.

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

output

enter values of a and b

3

15

addition is 18

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 Arithmetic 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 << num1 << "+" << num2 << " = "
        << num1 + num2;
  break;
  case '-':
std: cout << num1 << "-" << num2 << " = "
        << num1 - num2;
  break;
case '*':
std:: cout << num1 << " *" << num 2 << " = "
        << num 1 * num2;
break;
case '/':
if (num2 != 0)
std:: cout << num1 << " / " << num2 << " = "
        << num1 / num2;
else
```

```
std:: cout << "Error! Division by zero ";
break;
default:
std:: cout << " Invalid operator !";
}
return 0;
}
```

output:

```
Enter operator (+, -, *, /): +
 Enter two numbers : 3
  22
  3+22 = 25
Enter operator (+, -, *, /) :-
  Enter two numbers: 45
    18
    45-18 = 27
Enter operator (+, -, *, /): *
    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 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

7/7/25



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

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

output
1 2 3 4 5 6 7 8 9 10

3/7/25

# Experiment No-1

- Write a program to declare class 'student' having data members as roll-no & name & class. Accept & display data for one 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 <<e
}
};
   int main ()
{
 student s1;
 s1.accept ();
 s1. display ();
 return 0;
}
```

output
Enter value of roll & name 58
anushka
Roll: 12 , Name: anushka

- Write a program to declare class "time". accept time in H:M:S format, convert it into total seconds & display them.

```
# include <iostream>
using namespace std;
class time
{
 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 ();
  t1.display ();
}
```

output
Enter time in
H : 2
M : 30
S : 15
Total time in second : 9015 seconds.

Write program to declare class book having data members as book name, price, no. of pages, accept for 2 objects & 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;
   b1.accept();
   b2.accept();
  cout <<"\n book with greater price:\n";
   if (b1.price>b2.price)
   {
      b1.display();
   }
   else
  {
  cout<<"both books have the same price:\n";
   b1.display();
   b2.display();
  }
   return 0;
   }
```

31/7/25

output:

Enter value of name, price, pages: Harry potter
500
300
Enter value of name, price, pages: sudhamurty
650
350

Book with greater price:
Name: sudhamurty, price: 650
pages: 350

# Experiment X10-2

- WAP to declare class "city" having data members of name & population. Accept this data for 5 cities & 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;
}
 void display() {
 cout<<"Name:"<<name<<";Population:"<<
  population << endl;
}
  int getpopulation () {
   return population;
}
};
 int main()
{
   city c[5];
for(int i=0; i<5; i++) {
     c[i].accept();
}
```

```
int maxpopulation = 0;
for(int i=1; i<5; i++) {
 if(c[i].. getpopulation()> c[maxpopulation].get
population()) {
   maxpopulation = i;
}
}
 cout<<"\n city with highest population:\n";
 c[maxpopulation].display();
 return 0;
}
```

output

```
Enter city name & population: Mumbai
20411000
Enter city name & population : Delhi
19814000
Enter city name & population: Bengaluru
12360000
Enter city name & population: Hyderabad
10400000
Enter city name & population: chennai
10000000

city with highest population:
Name: Mumbai, population : 20411000
```

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

```
# include <iostream>
using namespace std;
class staff
{
  string name;
  string post;
  public:
  void accept()
{
 cout << "enter name & post";
 cin >> name >> post;
}
 void display () {
if (post == "HOD")
{
  cout << "HOD" << name << endl;
}
}
};
  int main ()
{
 staff s[5];
for (int i=0; i<5 ; i++)
{
 cout << "\nenter 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
Enter name & post : Alice
HOD
Enter name & post : charlie
HOD
Enter name & post : Mia
Assistant
Enter name & post : David
lecturer
Enter name & post : Eva
clerk

List of HODs
HOD Alice
HOD charlie

3/7/25

Page No.
Date

WAP to declare a class 'Account' having data member 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 bal;
string name;
void accept ()
{
cout<<" enter name of customer : ";
cin>> name;
cout<<" enter account_number: ";
cin>> acc-no;
cout<<" enter balance : ";
cin>> bal;
}
};
int main()
{
account a[10];
for(int i=0; i<10; i++)
{
cout<<"\n enter details for customer "<<i+1<<
\n;
```

```
a[i].accept();
}
  for(int i=0; i<10; i++)
{
  if(a[i].bal>5000)
{
  a[i].bal+=a[i].bal*0.1;
}
  cout<<"customer name:"<<a[i].name<<
        endl;
  cout<<" Account number:"<<a[i].acc_no<<
        endl;
  cout<<" balance after interest:"<<a[i].
        bal<<endl;
}
  return0;
}
```

output

Details for customer 1:
Name: James:
Account No: 126224256 9
balance: 70000

Details for customer 2:
name: charles
Account No: 126224987
balance: 80000

Page No.
Date

Details for customer 3:
name: William
Account no: 987654321
balance: $2000

Details for customer 4:
name: Oliver
Account no: 123456789
balance: 78000

Details for customer 5:
name: Henry
Account no: 895628147
balance: 76000

Details for customer 6:
name: Classen
Account no: 789128456
balance: 98001

Details for customer 7:
name: Commins
Account no: 741852963
balance: 120005

Details for customer 8:
name: Watson
Account no: 963741852
balance: 45621

Page No.
Date

Details for customer 9
name: Ricky
Account no: 456213789
balance: 12009

Details for customer 10:
name: Liam
Account no: 456789123
balance: 12800

Name: James
Account no: 1262242569
After interest: 77000

Name: Charles
Account no: 126224987
After interest: 88000

name: William
Account no: 987654321
After interest: 90200

name: oliver
Account no: 123456789
After interest 85800

Name: Henry
Account no: 895623147
After interest: 23600

Name: Clarren
Account no: 789123456
After interest: 107801

Name: Cummins
Account no: 741852963
After interest: 132006

name: Watson
Account no: 741852963
After interest: ~~132006~~ 50183

name: Ricky
Account no: 456213789
After interest 32001

Name: Liam
Account no: 456789123
After interest: 1230

31/7/25

31/7/25

Page No.
Date

# Experiment - 3

- WAP to declare class 'book' containing book name, author name & price. Accept & display data for one object using pointer to object of that class.

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

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

Output

enter book name harry potter
enter author name jk rowling
~~book~~ enter price 1000

bookname: harry potter name: jk rowling price: 10

31/7/25

Page No.
Date

WAP to declare class student having data member as roll no & percentage. Using this pointer invoke member function to accept the data & display this data for one object of a class.

```
# include <iostream>
using namespace std;
class student
{
int roll;
Float per;
public:
void accept()
{
cout<<"enter student roll & percentage";
cin>> this -> roll >> this -> per;
}
void display()
{
this -> accept();
cout<<"roll = "<<roll;
cout<<" percentage = "<< per;
}
};
int main()
{
student s1;
s1. display()
return 0;
}
```

output
enter student roll & percentage 58
87
roll = 58 percentage = 87

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

```
# include <iostream>
using namespace std;
class student {
int roll;
string name;
public:
void accept () {
cout<<" enter roll & name";
cin>> roll>> name;
}
void display () {
cout<<" Name & roll no is : "<< name<<" roll"
<<endl;
}
class marks {
int m1, m2, avg;
public:
void accept () {
cout<<"enter marks m1 & m2";
cin>>m1>>m2;
}
```

```
void calculate_avg () {
   avg = (m1 + m2) / 2;
}
  void display () {
cout << " Average of marks is : " << avg << endl;
}
};
};
int main ()
   {
student s1;
s1.accept ();
s1.display ();
student :: marks sm;
sm.accept 1 ();
sm.calculate_avg ();
sm.display ();
return 0;
}
```

output

```
Enter roll & name : 58
Anushka
Name & roll no is : 58 Anushka
 Enter marks :
  m1 = 89
  m2 = 79
  Average is : 84
```

14/8

Experiment No. 4

- 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 Number {
 int num;
 public:
void accept()
{
cout<<" enter number:";
cin>> num;
 }
 void display () {
cout<<" Number:"<<num<<endl;
 }
 void swap (Number& obj) {
  int temp = num;
     num = obj.num;
      obj.num = temp;
}
};
 int main () {
Number n1, n2;
cout<<" enter first number:"<<endl;
    n1.accept();
cout<<" enter second number:"<<endl;
    n2.accept();
    n1.swap(n2);
cout<<"\n After swap:"<<endl;
cout<<"first";
    n1.display();
```

```
cout<<"second ";
  n2.display ();
return 0;
}
```

output

Enter First number
enter number: 6
Enter second number
enter number: 9
After swap:
First number: 9
second number: 6.

- WAP to swap two numbers from class using concept of friend function.

```
# include <iostream>
using namespace std;
class temp {
  int x, y, z;
  public:
  void accept() {
cout<<" enter two numbers ";
cin >> x >> y;
  }
  void display ()
  {
cout<<" After swap x is :"<<x;
cout<<" After swap y is :"<<y;
  }
```

```
friend void swap (temp & t);
};
void swap (temp & t)
{
  t.q = t.x;
  t.x = t.y;
  t.y = t.q;
}
int main()
{
temp t1;
t1.accept();
swap (t1);
t1.display();
return 0;
}
```

output

```
Enter two numbers 9
    6
After swap x is : 6
After swap y is 9
```

WAP to swap two numbers from different class using concept of friend function

```
# include <iostream>
using namespace std;
Class B;
class A {
int num A;
Public:
void accept () {
cout<<" enter number A:";
cin>> numA;
}
void display () {
cout<<"number A="<<numA<<endl;
}
friend void swap numbers (A&, B&);
};
Class B {
int numB;
Public:
void accept () {
cout<<" enter number B:";
cin>> numB;
}
void display () {
cout<<"numbers A:"<<num B<<endl;
}
Friend void swapnumbers (A&, B&);
};
```

```
void swap numbers (A &a, B &b) {
  int temp = a.numA;
  a.numA = b.numB;
  b.numB = temp;
}
int main() {
A c1;
B d1;
c1.accept();
d1.accept();
swap numbers (c1, d1);
cout<<"In After swapping !"<<endl;
c1.display();
d1.display();
return 0;
}
```

output

```
enter number A: 67
enter number B: 89
After swapping !
Number A = 89
Number B = 67
```

WAP to create two classes Result 1 & result 2 which stors the marks of students. Read the value of a marks for both class objects & compute the avg of 2 results.

```
#include <iostream>
using namespace std;
 class result 2;
 class result 1; {
   int a;
 Public:
 void accept () {
 cout<<"enter marks out of 50:";
  cin>>a;
 }
 friend void cal(result r1, result r2);
 };
 class result 2 {
  int 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 <<"\n average :"<< avg;
 }
  int main ()
  {
 Result1 n;
```

```
Result 2 y;
x.accept();
y.accept();
cal (x, y);
}
```

Output:

```
enter marks out of 50: 45
enter marks out of 50: 46
Average = 45.5
```

- WAP to find the greatest number among 2 numbers from two different classes using friend function:

```
# include <iostream>
using namespace std;
class B;
class A {
int a;
Public:
void accept (){
cout << " enter a value: ";
cin >> a;
}
friend void greater (Aa1, Bb1);
};
Class B {
int b;
Public:
Void accept (){
```

```
cout<<"enter a value :";
cin>> b;
}
friend void greater (A a1, B b1);
};
void greater (A a1, B b1);
};
void greater (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.accept();
y.accept();
greater (x, y);
}
```

1418

Page No.
Date

# Experiment No 5

- 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()
{
```

Default

```
sum s1;
 s1.displayed();
  return 0;
 }
```

o/p:

```
   sum=10
   object destroyed
```

```
# 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";
    }
    };
    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 & obj) {
      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 & percentage write constructor to initialize these data members:

```
# include <iostream>
 using namespace std;
  class student
   {
    int per;
    string name;
     public:
     student()
     {
     per = 70;
     name = "abc";
     }
```

default

```
void display ()
    {
cout<<" name="<<name<<endl;
cout<<" percentage="<<per<<endl;
    }
~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;
    }
```

Parameterized

```
void display() {
cout << " Percentage = " << per;
cout << " Name = " << name;
}
~student()
{
cout << "object destroyed";
}
};
int main() {
student s1(90, "Anushka");
s1.display();
return 0;
}
```

o/p:

Percentage = 90
Name = Anushka
object destroyed.

```
# include <iostream>
using namespace std;
Class student {
int per;
string name;
public:
student()
{
per = 90;
name = "abc";
}
```

copy

```
student (const student s)
  {
   per = s.per;
   name = s.name;
  }
void display ()
  {
 cout<<" name = "<< name << endl;
 cout<<" percentage = " << per << endl;
  }
 ~student () {
 cout << " object destroyed";
   }
   };
   int main ()
   {
   student s1;
   student s2 = s1;
  cout<<" student 1: "<< endl;
   s1.display ();
  cout << " student 2 : "<< endl;
   s2.display ();
    return 0;
   }
```

o/p:

```
student 1:
name = abc
percentage = 90
student 2:
name = abc
percentage = 90
object destroyed
object destroyed
```

Page No.
Date

- WAP

- ```
# include <iostream>
using namespace std;
class rectangle
{
    int l, w;
    public:
    Rectangle()
    {
      l = 2;
      w = 1;
    }
    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(4,5);
```

→ Rectangle (int a, int b)
{
l = a;
w = b;
}
```

```
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 engineering" for course. Accept & 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, Rollno & course is:" << name << roll no
course;
} };
```

```
int main()
{
college c1(1, "Anushka", "cse");
college c2(1, "Anushka");
c1.display;
c2.display();
}
```

Q

1/9

Page No.
Date

# Experiment No - 6

1) single inheritance:

Create a base class called person with name & age. Derive a class student that adds attribute roll no. write function to display all details of student.

```
# include <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 rollno";
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 aanyushka
age 17
roll no: 58
name = Anushka
age = 17
roll no = 58

2) Multiple inheritance

Create two base classes Academic & sports

```
#include <iostream>
using namespace std;
class Academic
{
protected:
int marks;
};
class sports {
protected:
int score;
};
```

```
class result: protected Academic, protected sports
{
  private:
   int total score;
   public:
void accept () {
cout<<" enter student mark ";
cin>>marks;
cout<<" enter sports score ";
  cin>> score;
  }
  void calculate() {
cout<<"student marks="<< mark;
cout<<" sports score="<< score;
cout<<" Total score="<< total score;
  }
  };
  int main() {
  result r1;
    r1.accept();
    r1.display();
    r1.calculate();
      return 0;
    }
```

o/p:

```
enter student marks: 98
enter sports score: 89
student marks = 98 sports score = 89
      total score = 187.
```

3) multilevel inheritance

create class vehicle with attributes like brand & model.

```
# include <iostream>
using namespace std;
Class vehicle {
 public:
 string brand;
 string model;
 };
class car : Public vehicle
  {
  public:
  string type;
  };
class electricar : 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 = "<<model;
cout<<" car type = "<<type;
cout<<" battery capacity = "<<battery
        capacity;
    } };
    int main() {
    electric car ec1;
     ec1.accept();
     ec1.display();
        return 0;
        }
```

o/p:

```
enter vehicle brand: Teslar
enter vehicle model: 3
enter car type: sedan
enter battery capacity: 57.5
vehicle brand = Teslar vehicle model = 3
car type = sedan  battery capacity = 57.5
```

4) Hierarchial inheritance

Create base class employee with emp ID & name - Derive two classes manager & developer from employee. manager with department

```
#include <iostream>
Using namespace std;
class Employee {
protected:
   int empID;
   string name;
};
 class manager : public Employee {
private:
  string department;
public:
  void accept() {
cout<<" empID, name, department";
cin>> empID >>name>>department;
}
 void display () {
cout<<"empID = " empID;
cout<<"name=" name;
cout<<" department = "department;
}
};
class Developer : public Employee {
private :
   string programminglanguage;
public:
  void accept()
  {
cout<<" programminglanguage ";
cin>> programminglanguage;
}
  void display()
    {
```

```
cout<<"programminglanguage:"<<programminglangua
}
};
int main()
{
Manager m1;
m1.accept();
m1.display();
Developer d1;
d1.accept();
d1.display();
return 0;
}
```

output

empID = 01

name = anushka

department = IT

empID=01 name=anushka department=It.

enter programming language c++

programminglanguage = c++

5) Hybrid Inheritance

```
# include <iostream>
using namespace std;
Class Person {
protected:
    string name;
    int age;
public:
    void ~~accept()~~ getpersondata {
cout<<" Enter name:";
cin>>name;
cout<<" Enter age:";
cin>>age;
}
void showpersonData() {
cout<<"Name:"<<name<<endl;
cout<<" Age:"<<age;
}
};
  Class student : public person {
protected:
    int rollNo;
public:
    void getstudentData() {
cout<<" Enter rollnumber:";
cin>>roll no;
}
```

```
void showstudentData () {
cout<<" RollNO:"<<rollNO<<endl;
}
};
class sports {
protected:
int sportsmarks () {
cout<<" enter sports marks:";
cin>> sportsmarks;
}
void showsports marks () {
cout<<" sports marks :"<<sportsMarks<<endl;
}
};
class Academics {
protected:
int academicmarks;
public:
void getAcademic Marks () {
cout<<" enter academic marks:";
cin>> academicMarks;
}
void showAcademic Marks () {
cout<<" Academic marks :"<<academicmarks<<endl;
}
};
class Result : public student, public sports, public
Academics {
int total;
public:
```

```
void calculateResult() {
  total = sportsMarks + academicMarks;
}
void displayResult() {
  ~~total = sportsMarks + acader~~
  showpersondata();
  showstudentdata();
  showsportsdata();
  show Academicdata();
  cout<<"Total marks:"<<total<<endl;
  ~~return 0;~~ }
    };
    int main() {
  Result r;
  r.getpersondata();
  r.getstudentdata();
  r.getsportsmarks();
  r.getAcademicmarks();
  r.calculateresult();
  cout<<"In Result";
   r.displayresult();
   return 0;
    }
```

output

```
Enter name: Anushka
Enter age: 17
Enter roll no: 58
Enter sports marks: 98
Enter Academic marks: 99
```

Result
Name: Anushka
Age: 17
Roll No: 58
sports marks: 98
Academic marks: 99
Total marks: 197

26/9/25

Page No.
Date

Experiment No - 07

Q1) WAP using function overloading to calculate the area of laboratory (which is rectangular in shape) & area of classroom (which is square in shape).

```
#include <iostream>
using namespace std;
float area( float length, float breadth) {
    return length* breadth;
}
  float area (float side){
      return side*side;
}

 int main() {
float length, breadth, side;
cout<<" enter length & breadth of laboratory
 (rectangle) :";
cin>> length>> breadth;
cout<<" Area of laboratory :"<< area(length,
 breadth )<<endl;
cout<<" enter side of classroom (square): ";
cin>>side;
cout<<" area of classroom :"<<area (side)<<
   endl;
return 0;
}
```

o/p

enter length & breadth of laboratory (rectangle):
354
453

Area of laboratory : 160362

Page No.
Date

enter side of classroom (square): 56
area of classroom : 3136

Q2) Write a program using function overloading to calculate the sum of 5 float values & sum of 10 integer values.

```
# include <iostream>
using namespace std;
Float sum( Float a, Float b, Float c, Float d,
        Float e) {
        return a+b+c+d+e;
  }
int sum (int a1, int a2, int a3, int a4, int a5,
        int a6, int a7, int a8, int a9, int a10)
{ return a1+a2+a3+a4+a5+a6+a7+a8+a9+
    a10;
}
    int main() {
Float f1, f2, f3, f4, f5;
cout<<" enter 5 float values";
cin>> f1>> f2>> f3>> f4>> f5;
cout<<" sum of float values:"; << sum (f1, f2,
        f3, f4, f5) << endl;
int i1, i2, i3, i4, i5, i6, i7, i8, i9, i10
cout<<"enter 10 integer values :";
cin>> i1, i2, i3, i4, i5, i6, i7, i8, i9, i10;
cout<<" sum of integer values = " << sum
(i1, i2, i3, i4, i5, i6, i7, i8, i9, i10)
        << endl;
        return 0;
}
```

Page No.
Date

Q3) Write a program to implement Unary-operator when used with the object so that the numeric data member of class is negated

```
#include <iostream>
using namespace std;
class number {
private:
  int value;
public:
number(int v=0): value(v) {}
number operator -()const {
  return number (-value);
}
void display () const {
cout<<" value: "<< value <<endl;
}
};
int main () {
number num1(10);
number num2 ;
cout<<" original no: "<<endl;
num1.display();
num2 = -num1;
cout<<" After applying unary -operator"<<
endl;
num2.display ();
return 0;
}
```

output
Original number:
Value: 10
After applying unary-operator:
value: -10

(a) WAP to implement unary ++ operator (for pre increment & post increment) when used with object so that numeric data member of class in increment.

```
#include <iostream>
using namespace std;
class Number {
private:
    int value;
public:
Number (int v=0) : value (v) {}
  number & operator ++() {
      ++ value;
    return * this;
  }
Number operator ++(int) {
  Number temp = *this;
    value++;
  return temp;
  }
void display () const {
cout<< "value" << value<< endl;
}
};
```

```
int main() {
number n(5);
cout<<" initial value:"<<endl;
n.display();
cout<<" pre-increment"<<endl;
n.display();
cout<<" post-increment"<<endl;
n.display();
return 0;
}
```

output
initial value:
5

pre-increment
value: 6

post increment
value: 7

16/10

Page No.
Date

Experiment No - 08

Q Write a program to overload the '+' operator so that two strings can be concatenated.

```
#include <iostream>
using namespace std;
class Mystring {
private:
  string str;
  public:
Mystring (string s= " ") : str(s) {}
Mystring operator + (const Mystring &other)
 const {
   return Mystring (str + other.str);
}

  void display () const {
    cout<< str<< endl);
}
};
int main() {
Mystring s1("xyz");
Mystring s2("pqr");
Mystring s3 =s1 +s2;
cout<<" concatenated string ";
s3.display();
return 0;
}
```

output
concatenated string : xyzpqr.

Q WAP to create a base class Login having data members name and password. Declare accept() function virtual. Derive Emaillogin & membership login classes from Ilogin. Display email login details & membership login details of employee.

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

class Ilogin {
protected:
  string name;
  string password;
public:
  virtual void accept(){
  cout<<"Enter name";
  cin>>name;
  cout<<"enter password";
  cin>>password;
  }
  virtual void display() {
  cout<<"Name:"<<name<<endl;
  cout<<"password"<<password<<endl;
  }
  virtual ~Ilogin() {}
};
  class Emaillogin : public Ilogin{
    string emailID;
```

```
public:
    void accept() override {
cout<<" In-- Email login details--"<<endl;
cout<<" Enter name";
cin>> name;
cout<<" Enter password";
cin>> password;
cout<<" Enter email ID";
cin>> email ID
    }
void display () override {
cout<<" In Email login details :"<<endl;
cout<<" Name:"<<name<<endl;
cout<<" Password:" <<password <<endl;
cout<<" Email ID"<< email ID <<endl;
}
};
    class membership login :public I login {
    string memberID;
public:
    void accept() override {
cout<<" Membership login details"<<endl;
cout<<" enter name:";
cin>> name;
cout<<" enter password";
cin>> password;
cout<<" enter membership ID";
cin>> memberID;
    }
```

```
void display() override {
cout<< "membership login details" << endl;
cout << " Name :"<< name << endl;
cout << " password :"<< password << endl;
cout << " membership ID :"<< member ID << endl;
}
};
    Int main() {
Email login email user;
email user.accept();
email user.display();

membership login member user;
member user accept();
member user display();
return 0;
}
```

o/p

```
Enter name: Anushka
Enter password: anu008
Enter Email ID: anush.@gmail.com

Email login details
 Name: Anushka
 password: anu008
 Email ID: anush.@gmail.com

Enter name: anushka
password: anuhhh
Enter member ID: 3456
```

Member login details :
Name: anushka
password: anushh
member ID : 3456

16/10

Experiment No - 9

Q WAP to copy the content of one file into another. open "first.txt" in read (ios::in) mode & "second.text" file in write (ios::out) mode.

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

output
file copied successfully

Q WAP to count digit & spaces using file handling

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

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

① WAP to count words using file handeling

```
#include <iostream>
# include <fstream>
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 && prev_type==0) {
  word count ++;
}
  prev_type = current_type;
}
  cout << "word count:" << word_count <<
  endl;
File.close();
return 0;
}
```

12/11

Page No.
Date

# Experiment -10

Q WAP to find sum of array elements using function template (eg. pass integer, float & double array of 10 elements).

```
#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]= {7, 2, 2.2, 2.4};
double a3[3]= {10.5, 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,
<<endl;
return 0;
}
```

Output

sum of integer array is: 4
sum of float array is: 4.8
sum of double array is: 61.5

b) Write A P of square function using template specialization:

:-

```
# 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 ~~result;~~ (ss + ss);
}
int main () {
int i=2 , ii;
string www("Aanushka");
ii = square <int> (i);
cout << i << " : " << ii << endl;
cout << square <string> (www) << endl;
  return 0;
}
```

output

2 : 4

Anushka Anushka

Q WAP to build simple calculator using class template

```
#include <iostream>
#include <math>
using namespace std;
template <class T>
class calculator {
 private:
   num1, num2;
public:
 calculator (T n1, T n2) {
      num1 = n1;
      num2 = n2;
 }
  T add () {
  return num1 + num2;
  }
  T sub () {
      return num1 - num2;
  }
  T mul () {
      return num1 * num2;
  }
  T divide () {
 if (num2 == 0) {
      cout << " error!" << endl;
    return 0;
  } return num1/num2;
}

  void display Numbers () {
  cout << "Numbers :" << num1 << num2 << endl;
  }};
```

```
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- substraction \n";
cout<< " 3. multiplication \n";
cout<< " 4. Division \n";
cin>>choice;
switch(choice){
case 1;
cout<< "Result :"<< calc.add (a,b)<< endl;
break;
case 2:
cout<< "Result"<< calc.sub (a, b) << endl;
case 3:
cout<< "Result :"<< calc.div (a, b)<< endl;
break;
default:
cout<< " invalid choice "<< endl;
}
return 0; }
```

output

```
enter two number : 2 4
choose an operation : 1
1. Addition
2. substraction
3. Multiplication
4. Division
```

Result : 6

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

Q WAP to implement push & 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 stuck \n";
}
  void pop() {
if (elements.empty()) {
cout<<" stack is empty .cannot pop.\n";
  return;
}
cout<<"element.back()<<" popped \n";
  elements.pop_back();
}
```

```
void display () {
cout<< "stack element ";
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;
}
```

output

10 pushed to stack
20 pushed to stack
30 pushed to stack
stack elements : 10 20 30
30 popped

12/11

Page No.
Date

Experiment -11

Q WAP to implement generic vectors
a) To modify the value of given element

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

output

0 10 20 30 40

after modification: 0 10 99 30 40

Q To multiply by scalar ~~matrix~~ value.

```
#include <iostream>
#include <vector>
using namespace std;
int main() {
vector<int> vec = {1, 2, 3, 4, 5};
int scalar = 3;
for(int &val ~~&scalar~~ vec);
{val = val * scalar};
}

for (int val : vec) {
cout << val << " ";
}
cout << endl;
return 0;
}
```

output

3 6 9 12 15

Page No.
Date

Q To display the vector in 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;
}
```

output

10, 20, 30, 40, 50

Pl

Experiment - 12

a) Write a C++ program using STL
b) 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;
}
```

output

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

Q Implement queue

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

output

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

Page No.
Date

Implement Sorting & searching with user-defined records such as personal record, item record.

```
#include <iostream>
#include <vector>
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> 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 record 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 fou
return 0;
}
```

Output:
sorted records by age:
Bob - 25
David - 28
Alice - 30
Charlie - 35
Person with age 28 found: David:

12/11