Page No.
Date

Experiment - 1

1.
```
#include <iostream>
#include <string>
using namespace std;
class student
{
    int r;
    string n;
public:
    void accept (){
        cout << "Enter name & roll number :-";
        cin >> n >> r;
    }
    void display ()
    {
    void display ()
    {
        cout << " Name = " << n << endl;
        cout << " Roll.no = " << r << endl;
    }
};
int main()
{
    Student S1;

    S1. accept();
    S1. display();
    return 0;
}
```

Output -
Enter name & roll no.
38

2.
```
# include <iostream>
# include < string >
using namespace std;
class Book
{
public:
    int bp, bpgs
    string n;
    void accept()
    {
        cout << " Enter name & price: ";
        cin >> n >> bp;
        cout << " Enter pages: ";
        cin >> bpgs.
    }
    void display() {
        cout << " Book name = " << n << endl;
        cout << " Book price = " << bp << endl;
        cout << " Book page = " << bpages << endl;
    }
};
int main(){
    book b1, b2;
    b1. accept();
    b2. accept();
    if (b1.bp > b2.bp){
        cout << Book with higher price: ";
        b1.display();
}
```

```
cout << " city " << c[max].name;
c[max].diplay ();
return 0;
```

Output:

enter city name & population :

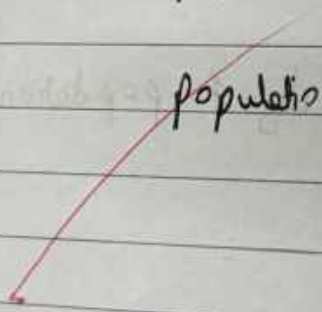
enter city name & population :

enter city name & population :

enter city name & population :

enter city name & population

city: population:



Page No.
Date

Exp 2:

1.

```
#include <iostream>
#include <string.h>
using namespace std;
class city
{
    public:
    int p;
    string name;
    void accept()
    {
        cout << "enter city name & population:";
        cin >> name >> p;
    }
    void display()
    {
        cout << "city & population " << name << p;
    }
};

int main()
{                    0  1  2    3  4
    city c[5];
    for(int i=0; i<5; i++)
    {
        c[i].accept();
    }
    int max=0;
    for(int i=0; i<5; i++)
    {
        if(c[i].p > c[max].p)
            max=i;
    }
```

3) #include <iostream>
using namespace std;
class time {
int H, M, S;
public:
void accept();
void display seconds();
};
Void time :: accept(){
cout << "\n Enter hours : ";
cin >> H;
cout << "Enter minutes:";
cin >> M;
cout << "Enter seconds : ";
cin >> S;
}
void display inSeconds();
return 0;
}

Enter Hours : 1
Enter minutes : 30
seconds : 15
Total time in seconds = 5415.

2.

```
#include <iostream>
using namespace std;
class student{
  public:
  float p;
  int r;
  void acc(){
    cout<<"enter rollno & percentage ";
    cin>>this->r>>this->p;
  }
  void disp(){
      this->accept();
      cout<<" rollno:"<<r<<endl;
      cin<<" percentage "<<p<<
  }
};
int main(){
    student S;
    S.disp();
  return 0;
}
```

3.

#in

Page No.
Date

Assignment 3

1.

```
#include <iostream>
using namespace std;
class book
{
Public:
float pr;
string n;
string b;
void accept(){
        cout<< "enter title, author name, price";
        cin >> b>> n >>pr;
}
void disp(){
    cout << "book title "<< b<<endl;
    cout<<" Author name "<<n<<endl;
    Cout << " book price "<<pr<<endl;
  }
};
  int main(){
      book b1;
      book* p;
      p= &b;
      p -> accept();
      p -> display();
        return 0;
    }
```

Page No.
Date

2

```
#include <iostream>
using namespace std;
class account {
    Public :
    float B ;
    int n ;
void accept(){
    cout << "Enter account number & balance" ;
    cin >> n >> B ;
}
};
int main() {
 Account. A[10]
 for (int i = 0; i<10 ; i++){
     A[i]. accept () ;
    }
 for (int i=0 ; i<10 ; i++ ){
   if (a[i] . b >= 5000 ){
     a[i] . b = a[i] . b * 0.1 + a[i] . b ;
   }
}
cout << " account balance " << a[i] . b ;
return 0 ;
```

```
cout<<"Sum: "<<sum(obj1,obj2)<<endl;
    return 0;
}
```

output:

Sum : 12

Exp 4

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

class B;
class A{
private:
    int x;

public:
    A(int val){
        x = val;
    }
    friend int sum(A,B);
};

class B{
private:
    int y;
public:
    B(int val){
        y = val;
    }
    friend int sum(A,B);
};
int sum(A a, B b){
    return a.x+b.y;
}
int main()
A obj1(5);
B obj2(2);
```

Page No.
Date

3.

```
#include <iostream>
using namespace std;
class student {
public:
    int roll;
    string n;
    void accept(){
        cout<<"enter the name & rollno ";
        cin >> n >> roll;
    }
    void disp(){
        cout<<"name"<<n<<endl;
        cout<<"roll no. "<<roll<<endl;
    }
    class marks{
    int m1, m2;
    public:
    void accept(){
    cout<<"enter marks"<<endl;
    cin >> m1 >> m2;
    }
    void display(){
        float p1 = ((m1+m2) / 100.0) * 100;
        cout<<"precentage: "<<p1<<endl;
    }
};
};  int main(){
    student s1;
    student :: marks m;
    s1.accept();
    s2.disp();
    m.accept();
    m.disp(); return 0
```

```
int main(){
  Box box1(100);
  Cube box2(80);

  find greater (box1, cube1);

  return 0;
}
```

Output:
Box has greater volume : 100

3.
```
#include <iostream>
using namespace std;

class Cube;

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

public:
    Cube(int v) {
        volume = v;
    }
    friend void findGreater(Box, Cube);
};
void findGreater(Box b, Cube c) {
    if (b.volume > c.volume) {
        cout << "Box has greater volume: " <<
    } else if (c.volume > b.volume) { b.volume << endl;
        cout << "Cube has greater volume: " << c.volume << endl;
    } else {
        cout << "Both have equal volume: " << b.volume << endl;
    }
}
```

2.
```
#include <iostream>
using namespace std;
class Number {
private:
    int value;
public:
    Number(int val){
        value = val;
    }
    void show(){
        cout << "value: " << 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 a(10);
    Number b(20);
    cout << "After swap:" << endl;
    a.show();
    b.show();
    return 0;
}
```

Output:

before swap: value: 10 value: 20

After swap: value: 20 value: 10

```
void findGreater(A x, B y){
    if(x.numA > y.numB)
        cout<<"Greatest:"<<x.numA<<endl;
    else if(y.numB > x.numB<<endl;
        cout<<"Greatest:"<<y.numB<<endl;
    else
        cout<<" Equal"<<endl;
}
int main(){
    A obj1;
    B obj2;

    obj1.read();
    obj2.read();

    findGreatest(obj1, obj2);
    return 0;
}
```

Output:

Enter number for class A: 17

Enter number for class B: 23

Greater : 23

5.
```
#include <iostream>
using namespace std;
class B;
class A {
    int numA;
public:
    void read(){
        cout << "Enter number for class A: ";
        cin >> numA;
    }
    friend void find Greatest (A,B)
};
class B {
    int numB;
public:
    void read(){
        cout << "Enter number for class B: ";
        cin >> numA;
    }
    friend void find Greatest (A, B)
};
class B{
    int numB;
public:
    void read(){
        cout << "Enter number for class B: ";
        cin >> numB;
    }
    friend void find Greatest (A,B);
```

4. #include <iostream>
using namespace std;

Class Complex {
private:
    int real;
    int img;

Public:
    Complex (int r=0; int i=0){
        real = r;
        imag = i;
    }
    void show (){
        cout << real << " + " << imag << "i" << endl;
    }
    friend Complex add (Complex, Complex);
};
Complex add (Complex 1, Complex 2) {
    Complex result;
}

int main (){
    Complex a (2,3);
    Complex b (4, 5);
    Complex c = add(a, b)
    cout << "Sum of complex no: ";
    c show ();
    return 0;
}

Output:
Sum of complex numbers: 6+8i

7.
```
#include <iostream>
#include <cmath>
using namespace std;

class Point {
private
    int x, y;

public:
    Point (int xVal, int yVal) {
        x = xval;
        y = yval;
    }

    friend double Calculate Distance (Point, Point);
};
double Calculate Distance (Point p1, Point p2) {
        int dx = p2.x - p1.x;
        int dy = p2.y - p1.y;
        return sqrt (dx*dx + dy*dy);
}
int main() {
    Point p1(3,4)
    Point p2(7,1);

    double distance = Calculate Distance (p1, p2);

    Cout << "Distance between points: " << dist
    return 0;
}
```

8.

```
friend void total sum (Alpha, Beta, Gema);
};

void total sum (Alpha x, Betay, Gemma z){
        int sum = x.a+y.b+z.c;
        cout<< "Sum of all values: " << sum << endl;
}
int main(){
   Alpha A1(10)
   Beta B1(20);
   Gema C1(30);

   total Sum( A1, B1, C1);

   return 0;
}
```

Output:
Sum of all values: 60

6. #include <iostream>
using namespace std;

class Beta;
class Gamma;

class Alpha {
private:
    int a;
public:
    Alpha (int val){
        a = val;
    }
    friend void total Sum (Alpha, Beta, Gamma);
};
class Beta {
  private:
      int b;

  public:
    Beta(int val) {
      b = val;
    }
    friend void total sum (Alpha, Beta, Gamma)
};
class Gama {
private:
    int c;
public:
    Gamma (int val){
        c=val;
}

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

class B;
class A {
    int numA;
public:
    A(int n) { numA = n; }
    friend void swapNumbers(A &x, B &y);
    void show() { cout << numA << " "; }
};

class B {
    int numB;
public:
    B(int n) { numB = n; }
    friend void swapNumbers(A &x, B &y);
    void show() { cout << numB << " "; }
};

void swapNumbers(A &x, B &y) {
    int t = x.numA;
    x.numA = y.numB;
    y.numB = t;
}

int main() {
    A obj1(10);
    B obj2(20);
    obj1.show(); obj2.show(); cout << endl;
    swapNumbers(obj1, obj2);
    obj1.show(); obj2.show();
    return 0;
}
```

10

```
#include <iostream>
using namespace std;
class Number {
    int number;
public:
    Number(int n) { num = n; }

    friend void swapNumbers(Number &n, Number
    void show() {
    cout << " " << num ; }
};

void swapNumber(Number &x, Number &y) {
        int t = x.num
        a.num = y.num
        y.num = t;
}
int main() {
    Numb a(10), b(20);
    a.show(); b.show(); cout << endl;
    swapNumber(a,b);
    a.show(); b.show();
}
```

Output

10 20

20 10

Output:
Distance between points

8.
```
#include <iostream>
using namespace std;

class Audit;

class BankAccount {
private:
    int balance;
public:
    BankAccount(int b) {
        balance = b;
    }
    friend void auditBalance(BankAccount, Audit);
};
class Audit {
public:
    void startAudit(BankAccount acc) {
        auditBalance(acc, *this);
    }
    friend void auditBalance(BankAccount, Audit);
};
void auditBalance(BankAccount acc, Audit) {
    cout << "Balance: " << acc.balance << endl;
}
int main() {
    BankAccount myAcc(50000);
    Audit auditor;
    auditor.startAudit(myAcc);
    return 0;
}
```

5 Parameterized

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

class calculator{
    int sum;
public:
    calculator (int n) {
        sum = 0
        for(int i=1; i<=n; i++){
            sum += i;
        }
    }
    void display(){
        cout << "Sum is " << sum << endl;
    }
};

int main() {
    calculator c1(10);
    c1.display();
    return 0;
}
```

Output:
Sum is: 55

```
in main()
{
    Result obj1;
    Result obj2;

    obj1.read();
    obj2.read();

    average(obj1,obj2);

    return 0;
}
```

Enter marks for Result 1: 100
Enter marks for Result 2: 99
Avg marks = 99.5

Q.
21

12 #include <iostream>
using namespace std;

class Result2;

class Result1 {
    float marks;
public:
    void accept() {
    cout << "Enter marks for Result1: ";
    cin >> marks;
    }
    friend void average (Result1, Result2);
};

class Result2 {
    float marks;
public:
    void read () {
        cout << "Enter marks for Result2:"
        cin >> marks;
    }
    friend void average (Result1, Result2);
}

void average (Result1 r1, Result2 r2) {
    float avg = (r1.marks + r2.marks) /2
    cout << "Avg marks = " << avg << endl
    cout << "A
}

2. Default :

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

class student {
    string name;
    float percent;

public:
    student(){
        name = "arhon";
        percent = "100.0";

        cout << "name" << name << "\n Percentage: " << Percent << "%"
            << endl;
    }
};

int main() {
    student s1;
    return 0;
}
```

Name: arhon
percentage: 100 %.

Copy:

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

class calcu {
  int sum;
  Public:
   calcu ( ){
      sum =
      int n = 10;
      sum = 0;
      for (i=0; i<=n; i++) {
        sum += i;
      }
   }
   calcu ( ) {
    calcu ( calcu & x ){
      sum = x. sum ;
    cout << " Sum " << sum ;
  }
};
int main (){
    calcu c1 ;
    calcu c2 (c1);
return 0;
}
```

Page No.
Date

Default:

```
#. ~~iostre~~. include < iostream>
using namespace std;

class calculator {
    int sum;
public:
    calculator (){
        int n = 10;
        sum = 0;
        for (int i = 1; i <= n; i++){
            sum += i;
        }
        cout << "Sum is: " << sum << endl;
    }
};

int main (){
    calculator c;
    return 0;
}
```

2 Constructor overload

```
#include<iostream>
using namespace std;
class student{
    string name;
    float perc;
Public:
    student(){
    name = "arhan";
    float = 90.2;  Cout<< name<< perc;
    }
    student(string n, float f){
      name = n;
      perc = f;
    } cout<< name<<perc;
    }
};
int main(){
    student s();
    student s1("Arhan", 93);
return 0;
}
```

O/P: arhan 90.2
Arhan 93

3 Defaul

```
#in
usin
clas
```

Page No.
Date

2 copy

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

class student {
    public:
    string name;
    float perc;

    student (string h, float n){
        name = h;
        perc = n;
    }
    student (student&x){
    name = name.x;
    perc = x.perce;
    }
};
int main(){
    student s1("Arhon", 100);
    student s2 (s1);
Cout << "name " << name;
Cout << "percentage " << perc;
}
return 0;
},
```

2 Parameterized.

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

class student {
    string name;
    float percent;

public:
  student (string n, float p){
        name = n;
        percent = p;

    cout << "Name: " << name << "\n Percent: " << p
        << endl;
  }
};
int main(){
    student S1("arhan", 100);
    return 0;
}
```

Name: arhan
Percent: 100 %.

Page No.
Date

4. Write a program for constructor overloading.

```
#include <iostream>                    // We have to use 2 or
using namespace std; //more constructures for constructor
class student {  //overload.
    int rollno;
    string name;
  Public:
    student ( ){
      rollno: 28 ;
      name : "archan";
    cout << "name:" << name << "rollno:" << rollno << endl;
    }
      student ( int r, string n){
        rollno = r ;
        name = n ;
      cout << "name:" << name << "rollno:" << rollno << endl;
      }
};
int main(){
    student s;
    student s. ( 28, "archan");
return 0;
}
```

Qn

14/10

3 Parameterized & Copy & Constructor overload

```
#include <iostream>
using namespace std;
class college {
    string name;
    string course;
    int rollno;
Public:
    college(string n, string c, int r){

        name = n;
        r = roll
        rollno = r;
    cout << " name" << name << "rollno: " << rollno << course
    }
    college (college &x){

        name = x.name ;
        rollno = x rollno;
        course = x. course ;
    cout << name << rollno << course;
    }
};
int main(){
    college c ("Arhan", 28);
    college c1(c);
return 0;
}
```

3 Default:

```
# include <iostream>
using namespace std;
class college {
    int rollno;
    string name;
    string course;
    Public:
     college(){
      name = "arhan";
      Rollno = 10;
      course = "Computer engineering";
     cout << " name " << name << "Rollno: " << Rollno << endl;
     cout << "course: " << course;
     }
};
int main(){
  college C, C1;
Return 0;
}
```

```
int main(){
    manager m;
    m.calw();
    developer d;
    d.call();
return 0;
}
```

b Hierarchial.

```
#include <iostream>
using namespace std;
class employee {
    protected:
        int empid;
        string name;
};
class manager : Protected employee {
    Private:
        string dname;
    Public: void calcu(){
        cout<<" enter employee name & id";
        cin>> ename >> empid;
        cout <<" enter department name:";
        cin>> dname;
        cout<< name << empid << dname;
    }
};
class developer : Protected employee {
    Privat:
        string program_lang;
    Public:
        void call(){
        cout <<" enter emp name & id ");
        cin >> name >> id;
        cout << " enter program language: ";
        cin >> program_lang;
        cout<< name<< id << program_lang;
}
};
```

Exp 6:

a. multilevel inheritance

```
# include <iostream>
using namespace std;
class depart {
Protected: string dname;
};
class student : Protected depart {
Protected int rollno;
        string studentname;
};
class marks : Protected student {
Private:  int m1, m2
          int total;
Public:
        void calculate (){
        { cout<<" enter department name, student name & rollno
          cin >> depart dname >> name >> rollno;
          cout<<" enter marks m1 & m2 ";
          cin >> m1 >> m2;
          cout<<" Total: " << m1+m2 <<endl;
        }
};
int main () {
    marks m;
    m. Calculate ();
return 0;
}
```

Page No.
Date

```
int main(){
    student s;
    s.getdata();
    staff s1;
    s1.accept();
return 0;
}
```

Q.

14/10

d Hybrid:

```
#include <iostream>
#include < string>
using namespace std;
class college {
str. dname
Protected:
        string cname;
};
class student: Protected college {
    int rollno;
    string name;
Public: void setdata(){
    cout << " enter college name, studentname & rollno :
    cin >> cname >> rollno >> name;
    cout << cname << rollno << name;
    }
};
class employee: Protected college {
Protected: string ename;
        int empid;
};
class staff: Protected employee {
        string sname;
Public:
        void accept(){
        cout << " enter college name, employee name & id &
        cin >> cname >> ename >> empid >> sname;
        cout << cname << ename << empid << sname;
        }
};
```

Page No.
Date

C. Multiple inheritance

```
#include <iostream>
using namespace std;
class department{
protected string dname;
};
class student {
Protected: int rollno;
    string name;
};
class marks : Protected department, Procted student {
Private: int m1, m2;
Public:
    void getdata(){
    cout<<" enter dname, rollno, name:";
    cin>> dname >> rollno>> name;
    cout<<" enter marks";};
    cin >> m1 >> m2;
    cout << m1+m2;
    }
int main(){
    marks m;
    m.getdata();
return 0;
}
```

b.

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

int sum(int a[], int n){
  int s=0;
  for(int i=0;i<n;i++)
    s+=a[i];
  return s;}

float sum(float a[], int n){
  float s=0.0f;
  for(int i=0;i<n;i++)
  s+=a[i];
  return s;

int main(){
  int ai[10];
  float af[5];

  cout<<"enter 10 integers:";
  for(int i=0;i<10;i++)
  cin>>ai[i];

  cout<<"enter 5 floats:\n";
  for(int i=0;i<5;i++)
  cin>>af[i];

cout<<"sum of int:"<<sum(ai,10)<<endl;
  cout<<"sum of float:"<<sum(af,5)<<endl;
    return 0;
}
```

Page No.
Date

Exp 7.

a.
```
#include <iostream>
using namespace std;
class Area{
public:

    double calculate(double length, double breadth){
        return length * breadth;
    }

    double calculate(double side){
        return side * side;
    }
};
int main(){
    area a;
    cout<<"Area of rect: "<<a.calculate(10,5)<<endl;
    cout<<"Area of class: "<<a.calculate(7)<<endl;
return 0;
}
```

e virtual

```
#include <iostream>
using namespace std;
class department {
Protected: string dname;
};
class employee : Protected virtual department {
Protected: int no_of_emp;
};
class student : Protected virtual department {
Protected: int no_of_student;
};
class Total : Protected employee, Protected student {
    int total;
  Public:
      void call() {
      cout << " enter dname, no_of_emp, no_of_student ";
      cin >> dname >> no_of_emp >> no_of_student;
Total = no_of_emp + no_of_student;
      cout << total;
}
};
int main() {
   Total t;
   t.call();
return 0;
}
```

14/10

Experiment 6

a.

```
#include <iostream>
#include <string>

class Mystring {
    string s;
public:
    Mystring() {}
    Mystring(string x): s(x) {}

    Mystring operator+(Mystring &other) {
        return Mystring(s + other.s);
    }
    void read(char *label) {
        cout << "Enter " << label << ": ";
        cin >> s;
    }
    void show() {
        cout << s;
    }
};
int main() {
    Mystring a, b, c;
    a.read(a);
    b.read(b);
    c = a + b;
    cout << "concatenated: ";
    c.show();
    return 0;
}
```

d. #include <iostream>
using namespace std;

class Number {
    int n;
public:
    void getdata() {
    cout << "Enter number: ";
    cin >> n;
    }
    void showdata () {
    cout << "value of n = " << n << endl;
    }

    void operator ++() {
    ++n;
    }

    void operator ++(int) {
    n++;
    }
};
int main () {
Number n;
n.getdata();
Cout << "original: ";
n. Show data();
++num;
Cout << "After pre increment";
n. showdata();
n++;

4/11

Cout << "After
n. showdata();
return 0;
}

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

class Num {
    int x;
public:
    void read () {
    cout << " enter number : ";
    cin >> x;
    }
    void show () {
    cout << " x = " << x << endl;
    }
    void operator - () { x = -x; }
```

9:

```
int main () {
    Num n;
    n.read
    -n;
    n.show ();
    return 0;
}
```

Page No.
Date

Exp 9.

a

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

int main(){
    ifstream fin("First.txt", ios::in);
    ofstream fout("Second.txt", ios::out);

if(!fin || !fout){
cout<<"file open error";
return 0;
}

fout<<fin.rdbuf();
cout<<"Copied";
return 0;
}
```

b.

```
class Membershiplogin : public Ilogin {
    int member id;
public:
    void accept () {
        cout << "membership Login" ;
        cout << " enter member id " ;
        cin >> membe id;
        Ilogin :: accept();
    }

    void display() {
        cout << membership Login details;
        cout << " member id : " << member id << endl;
        Ilogin :: display();
    }
};
int main() {
    Ilogin* ptr;
    Emaillogin e;
    Membershiplogin m;

    ptr = &e;

    ptr -> accept();
    ptr -> display();

    ptr = & m;

    ptr -> accept();
    ptr -> display();
    return 0;
}
```

4/11

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

class Login {
protected:
    string name, password;
public:
    virtual void accept() {
        cout << "Enter Name : ";
        cin >> name;
        cout << "Enter Password : ";
        cin >> password;
    }
virtual void display() {
    cout << "Name : " << name << "Password : " << password;
    }
};
class EmailLogin : public Login {
    String email;
public:
    void accept() {
        cout << "emaillogin";
        cout << "enter email login : ";
        cin >> email;
        Login :: accept();
    }
    void display() {
    cout << "Login details ";
    cout << "email : " << email << endl;
    Login :: display();
    }
};
```

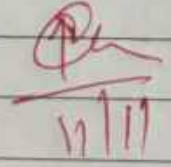
Page No.
Date

d

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

int main(){
    ifstream fin("first.txt");
    if(!fin){
        cout << "file not found";
        return 0;
    }
    string word, target;
    int count = 0;

    cout << "Enter word to search: ";
    cin >> target;
    while (fin >> word){
        if (word == target)
            count++;
    }
Cout << "The word: " << target << " appears " << count << " times " << endl;
fin.close();
return 0;
}
```

c.
```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;

int main() {
    fstream fin("first.txt");
    if(!fin){
    cout << "file not found!";
    return 0;
}
    string word;
    int word count = 0 ;

    while (fin >> words){
        count++;
}

cout << "Total words= " << count << endl;
fin.close();
return 0;
}
```

b.
```
#include<iostream>
#include<fstream>
#include<cctype>
using namespace std;

int main(){
    ifstream fin("first.txt");
    if(!fin){
    cout<<"File open error!";
    return 0;
}

long digits=0, spaces=0
char ch;
while (fin.get(ch)){
    if(isdigit(static_cast<unsigned>(ch))) digits++;
    if(ch==' ') spaces++;
}

cout<<"Digits:"<<digits<<"spaces:"<<spaces<<endl;
return 0;
}
```



3. WAP to design simple calculator using class template

```
# include <iostream
using namespace std;

template < class T, T2> class A
{
  Public :

  T num1 ;
  T num2 ;

  ~~void add()~~
  cout << " enter number " ;
  cin >> num1 >> num2 ;

  void add() {
    cout << " add: " << num1+num2 ;
  }
  void sub(){
    cout << " subs: " << num1 - num2 ;
  }
  void multiple(){
    cout << " multiply: " << num1 * num2 ;
  }
  void divide (){
    cout << " divide: " << num1 / num2 ;
  }
int main(){
  A < int > d ;
  d. add ();
  d. sub ();
  d. multiple ();
  d divide ();
  return 0
  }
```

Page No.
Date

6.
```
#include <iostream>
#include <string>
using namespace std;
template <class T>
T square (T x){
    return x*x;
}
template <>
string square (string s){
    return s+s;
}

int main(){
    cout << square(4) << endl;
    cout << square(string("Hi"));
}
```

Page No.
Date

Experiment 10.

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

template <class T>
T sum (T a[], int n){
  T s=0;
  for (int i=0; i<n; i++)
     s+= a[i];
   return s;
}
int main(){
  int a[5]
  for (int i=0; i<5; i++)
     cin>> a[i];
   cout << sum (a,5);
}
```

Page No.
Date

Exp 11:

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

template <class T>
class vector {
    T v[10];
    int n;
public:
    void create() {
        cin >> n;
        for (int i=0; i<n; i++)
            cin >> v[i];
    }
    void modify (int index, T value) {
        if (index >= 0 && index < n)
            v[index] = value;
    }
    void multiply (T scalar) {
        for (int i=0; i<n; i++)
            v[i] *= scalar;
    }
    void display() {
        cout << "(";
        for (int i=0; i<n; i++) {
            cout << v[i];
            if (i < n-1) cout << ",";
        }
        cout << ")";
    }
};

int main() {
    vector <int> vec;
    vec.create();
    int index, val, scalar;
    cin >> index >> val >> scalar;
    vec.modify(index, val);
    vec.display();
}
```

```
cin >> ch;

Calc<double> c;
c.get();
switch (ch) {
    case 1: cout << c.add(); break;
    case 2: cout << c.sub(); break;
    case 3: cout << c.multi(); break;
    case 4: cout << c.divide(); break;
    case 5: cout << c.modv(); break;
    case 6: cout << c.power(); break;
    case 7: cout << c.avg(); break;
    case 8: cout << c.maxv(); break;
    case 9: cout << c.minv(); break;
    case 10: c.trig(); break;
    default: cout << "Invalid choice";
}
}
```

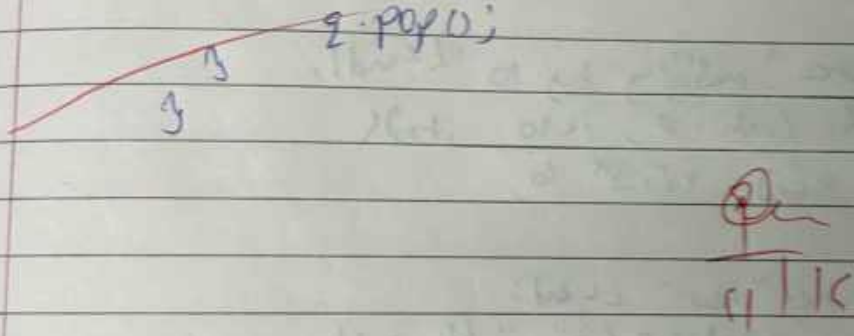
Page No.
Date

4. Oper Modulus, sin, cos, etc latest 10 with switch case

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

template <class T>
class Calc {
  T a, b;
Public:
    void get () { cin >> a >> b; }
    T add () { return a+b; }
    T sub () { return a-b; }
    T mul () { return a*b; }
    T divid () { return a/b; }
    T modu () { return fmod(a,b); }
    T power () { return (a+b)/2; }
    T avg () { return (a+b)/2; }
    T maxx () { return a>b ? a:b; }
    T minu () { return a<b ? a:b; }
    void trig () {
    cout << "Sin(a) = " << sin(a) << "Cos(a)=" << cos(a) << "tan(a)=" <<
    tan(a) << endl;
    cout << "Sin(b) = " << sin(b) << "Cos(b)" << cos(b) << "tan(b)
    << tan(b) << endl;
    }
};
int main() {
int ch;
cout << "1. Add 2. Substract 3. Multiply 4. Divide 5 Modulus 6 Power
    7. Avg 8. Maximum 9. Minimum 10. Trigo";
```

Page No.
Date

6.
```
#include <iostream>
#include <queue>
using namespace std;

int main (){
    queue <int> q;
    q.push(10);
    q.push(20);
    q.push(30);
    while (!q.empty()){
        cout << q.front() << " ";
        q.pop();
    }
}
```



Exp 12:

A.
```
#include <iostream>
#include <stack>
using namespace std;

int main () {
    stack<int> s;
    s.push(10);
    s.push(20);
    s.push(30);
    while (!s.empty()) {
        cout << s.top() << " ";
        s.pop();
    }
}
```

Page No.
Date

Exp 11.

B.

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

int main(){
  vector<int>v= {1,2,3,4,5,6,7,8,9,10}

  cout<<"Initial vector :"<<endl;
  for (int i=0; i<10; i++){
      cout<<v[i]<<" "<<endl;
  }
  cout<<"multiply by 10 "<<endl;
    for (int i=0; i<10; i++){
    v[i]= v[i]* 10;
  }
  cout<<"New:"<<endl;
    cout<<v[i]<<" "<<endl;
    }
    return 0;
}
```

Arhan Yaganegi

@4Oarhan

Personal Information

4Oarhan@gmail.com

+91-7559226500

India

My Resume

+ Add Resume

Add your resume here

EEO settings

Complete your profile

Add your missing details →

This data will be helpful to auto-fill your job applications

20%

My Badges

My Certifications

Work Experience

+ Add Work Experience

Add your work experience. Don't forget to add those internships as well.