PAGE No.
DATE / /

Experiment 1

Q)1)

```
#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 <<"nam:" << name;
  }
  };
  int main()
   Student S1;
    S1.accept();
    S1.display();
    return 0;
    }
```

PAGE No.
DATE / /

Q2)

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

(Q7)

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

Q7/17

PAGE No.
DATE / /

Experiment NO 2

Q1)

```
#include <iostream>
using namespace std;
class city
{
int population;
string name;

public
 void accept()
 {
cout << "Enter city name + population";
cin >> name >> population;
 }
void display ()
{
cout<<"name :" << name << ": population:" << population
        << endl;
   }
   int get population ()
   {
   return population;
   }
   };
   int main ()
   {
   city [5];
   for (int = 0 i<5 ; i++)
      {
```

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

PAGE No.
DATE / /

12)

```
#include <iostream>
using namespace std;
class staff
{
string name;
string post;

public
void accept()
{
cout<<"Enter name and Post";
cin>>name>>post;
}
void display()
{
if (post=="HOD" || Post="hod")
{
cout<<"HOP"<<name<<endl;
}
}
};
int main()
{
staff s[5]
for(int i=0; i<5 i++)
{
cout "\n enter details for staff "<<i+1<<
"\n";
```

```
S[i].accept();
}
cout << "\n\tlists of HOD \n";
    for(i=0; i<5; i++)
    {
    S[i].display();
    }
    return 0;
    }
```

PAGE No.
DATE / /

Q3)

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

class account
{
public:
    int acc_no;
    int 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];
int i;
for(i=0;i<10;i++)
{
a[i].accept();
}
```

PAGE No.
DATE / /

```
for (i=0; i<10; i++)
{
    if (a[i].bal > 5000)
    {
        a[i].bal = a[i].bal + a[i].bal * 0.1;
    }
    cout << "\n Customer name : " << a[i].name << endl;
    cout << "Account number : " << a[i].acc_no <<
        endl;
    cout << "Balance : " << a[i].bal << endl;
}
return 0;
}
```

Q
31/7

PAGE No.
DATE / /

Experiment 4

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

```
#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
Numbr n1, n2;
cout << " Enter first number :" << endl;
n1. accept ();
cout << " Enter second number :" << end l;
n2. accept ();
n1 . swap (n2);
```

```
cout << " In after swap :" endl;
cout << " First";
n1.display();
cout << "second";
n2.display();
return 0;
}
```

Output:

enter number : 6
enter number : 9
After swap : 9
6

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

```
#include <iostream>
using namespace std;
class temp {
int x, y, q;
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 = 6
After swap y = 9

5) WAP to swap two numbers from diffent class using concept of friend function

```
#include <iostream>
using namespace std;
class B;
class A {
int num A;
public:
```

PAGE No.
DATE / /

3) 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:
cout << "Enter number A:";
cin >> num A;
}
void display () {
cout << "Number A = " << num A << endl;
}
friend void swap numbers (A&, B&);
};
class B {
int num B;
public:
void accept () {
cout << "Enter number B:";
cin >> num B;
}
void display () {
cout << "number A=" << num B
<< endl;
}
friend void swap numbers
(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 c2;
    c1.accept();
    d1.accept();
    swap numbers (c1, d1);
    cout << "\n after swapping :" << endl;
    c1.display();
    d1.display();
    return 0;
}
```

output :

Enter number A = 67

Enter number B = 89

After swapping =

89

67

4)

```
#include <iostream>
using namespace std;
class result;
class result {
    int a;
public:
void accept () {
```

```
cout << " Enter mark out of 50";
    cin >> a;
}
friend void cal (result r1, result r2);
};
class result {
    int b;
    public:
    void accept() {

cout << " Enter marks out of 50 :";
    cin >> b;
}
friend void cal (result r1, result r2)
& };
void cal (result r1, result 2 r2) {
    float avg = (float) (r1.a + r2.b)/2;
    cout << "\n average : " << avg;
}
int main()
{
    result 1 x;
    result 2 y;
    x.accept();
    y.accept();
    cal (x,y);
}
output:
Enter marks out of 50 : 45
Enter marks out of 50 : 46
        Avg = 45.5
```

PAGE NO.
DATE / /

Experiment 5

Implement Types of constructors

a) ~~Write a program~~

b)
```
#include <iostream>
#include <String>
using namespace std;
class student
{
string name;
double per;
public:
student ()
{ name = "Sanchita Das";
  per = 90; }
  void display ()
  { cout << "Details " << endl;
  cout << " Name : " << name << endl;
  cout << " Percentage : " per << " % " endl;
  } };
  int main ()
  { student s1;
    s1.display ();
    return 0; }
```

Output :
Details
Name : Sanchita Das
Percentage : 90%

c)

```
#include <iostream>
# include <string>
using namespace std;
class college
{ public:
string name, course;
double roll;
college ()
{ name = "Sanchita Das";
roll = 90;
course = "Computer Engineering"; }
College (string n, int r, string c)
{name = n;
roll = r;
course = c; }
void display ()
{ cout << "In Details" << endl;
cout << "Name :" << name << endl;
cout << "Roll no :" roll << endl;
cout << "Course :" course << endl; } };

int main()
{ college c1;
college c2 ("Ria", 80, "Computer engineering");
c1 display ();
c2. display ();
return 0; }
```

output:
Name : Sarchita Das
Roll no. : 90
Course : Computer Engineering

Name : Ria
Roll no. 80
Course : Computer Engineering

Input 2.0

```
#include <iostream>
using namespace std;
class college {
  int roll;
  string name, course;
public:
  college() {
  course = "Computer Engineering"; }

void accept()
{ cout << "Enter details of student :" << endl;
  cout << "Enter roll no: ";
    cin >> roll;
    cout << "Enter name :";
    getline("cin name); }

void display()
cout << "\n student details :" << endl;
cout << "Roll no : << roll << endl;
cout << "Name : "<< name << endl;
cout << "Course : << course << endl; }};
```

```
int main ()
{
 college c1, c2;
 c1.accept();
 c2.accept();
 c1.display();
 c2.display();
 return 0; }
```

Output :

```
Enter details of student
Enter roll no : 60
Enter name : Sanchita
Enter roll no. 80
Enter name : Rita
```

PAGE No.
DATE / /

```
#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; }}
void display () {
cout << "sum of numbs from 1 to " << n << "=" 
<< to << endl; }};
int main ()
{ int n;
cout << "Enter the value of n:";
cin >> n;
sum s(n);
s.display();
return 0; }
```

Output

Enter the value = 10

sum of numbs from 1 to 10 = 55

d)

```
#include <iostream>
using namespace std;
class rectangle
{ int l, b;
public:
  Rectangle () // default
  { l=1;
    b = 6; }
  rectangle (int a, int B)
  { l = a
    b = B; }
  Rectangle (const rectangle &r)
        { l=r.l;
          b = r.b; }
  int area()
    { return l*b; }

int main ()
{ rectangle r;
  rectangle r2 (5,7);
  rectangle r3 (1,2);
  r1.area ();
  r2.area ();
  r3.area ();
```

output:

6

35

35

PAGE No.
DATE / /

Experiment 7

Demonstrate the compile time polymorphism

a)

```
#include <iostream>
using namespace std;
class Area {
public:
int find Area (int length, int breadth)
{ return length * breadth; }
int find Area (int side) {
    return side * side; }};

int main()
{ area a;
  int l, b, s;
  cout << "Enter length & breadth of laboratory:";
  cin >> l >> b;
  cout << "Enter side of classroom;
  cin >> s;
  cout << "In area of laboratory = " << a.find
                                    Area(l,b)
                                    << endl;

  cout << "Area of classroom =" << a.find Area(s) << endl
  ; return 0; }
```

Output:

```
Enter length & breath of laboratory: 3, 4
Enter side of classroom 5
Area of lab = 12
Area of classroom = 25
```

b) sum of

```
#include <iostream>
using namespace std;
class sum {
public
float add (float arr[5]) {
  float total = 0;
for (int i = 0; i < 5; i++) {
  total = arr [i]; }
  return total ; }
  int add (int arr [10])
  { int total = 0;
  for (int &i = 0; i < 10; i++) {
    total += arr[i]; }
    return total; }};
  int main () {
    Sum s;
    float f[5];
    int i[10];
    cout << " enter 5 float value :
    for (int j = 0; j < 5; j++) {
    cin >> f[j]; }
    cout << "sum of 5 float values
    . " << s.add (f) << endl;
    cout << "\n enter 10 integer values:";
    for int (j = 0; j < 10; j++)
    { cin >> i[j]; }
    cout << " sum of 10 integers
    value " << s.add (i) << endl;
    return 0; }.
```

Output :-

Enter 5 values : 3.4, 4.5, 3.6, 3.5, 7.6

Sum = 22.6

Enter 10 int values : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Sum = 55

c)

```
#include <iostream>
using namespace std;
class number {
  int value;
public:
  number (int v=0)
  { value = v; }
  void show () {
  cout << value << endl; }
    void operator - () {
      value = - value; }};
      int main ()
        { int x;
          cout << " enter a number ";
          cin >> n;
        Number n(n);
        cout << "\n Before negation : ";
        n.show();
        -n;
        cout << " After negation : ";
        n : show ();
        return 0; }
```

PAGE NO.
DATE / /

Output

Enter a number : 98

before negation : 98

after negation : -98

Q1

12/11

PAGE No.
DATE / /

Experiment 6

1) Single inheritance

```
#include <iostream>
using namespace std;
class person {
public:
string name;
int age;
void accept person (string n int a)
{ name = n;
age = a; }};
class student : public person {
public
int roll number;
void accept student (string n, int a, int r)
{ accept person (n, a);
roll number = r; }
void display ()
{ cout << "name : " << name << endl;
cout << " Roll no ." roll number << endl; }};
int main ()
{ student s;
s accept student (" Ria, 20, 101);
s.display ();
return 0; }
```

output :

name : Ria

age : 20

Roll no. 101

PAGE NO.
DATE / /

2) Multiple inheritance

```
#include <iostream>
using namespace std;
class academic {
public:
  int marks;
  void set_marks(int m){
    marks = m;
  }
class sports {
public:
  int score;
  void set_score(int s) {
    score = s;
  }
class result : public academic, public
               sports {
public:
  void displayResult
  { int total = marks + score;
  cout << "Academic marks : " <<
    marks << endl;
  cout << "sports & score : " << scor
    << endl;
  cout << "Total >> " << Total << endl;
  }
int main ()
{ result r;
  r.set_marks(85)
  r.set_score(15);
  r.display_result();
```

return 0; }

output :

academic marks : 85

sports score : 15

Total 100

3) Multilevel ~~inheritance~~ inheritance

```
#include <iostream>
using namespace std;
class vehicle {
public:
string brand, model;
void set_vehicle (string b, string m)
{ brand = b;
model = m; } };
class car : public vehicle {
public:
string type;
void setcar (string t) {
type = t; }};
class electric_car : public car {
public:
int battery_capacity;
void set_electric (int bc) {
battery_capacity = bc; }
void display()
{ cout << "Brand : " << brand << endl;
cout << "Model : " << model << endl;
cout << "Battery capacity : "
battery_capacity << "KWh" << endl;
}}
```

PAGE No.
DATE / /

```
int main () {
electric Car ej
  e.set vehicle ("Tesla", model s");
  e set Car ("sedan");
    e.set Electric (100);
    e. display ()
    return 0;
```

output:
Brand: Tesla
model: model s
Type: Sedan
Battery capacity 100 KWh

Hierachal inheritance

```
# include <iostream>
using namespace std;
class employee {
public
int empID;
string name;
void set employee (int id, string n) {
  empI = id
  name = n; };
class manager : public employee {
public;
string department;
void set manager (string dept)
```

PAGE NO.
DATE / /

{ department = dept; }
void display () {
cout << "empID: " << empID << endl;
cout << name; " << name endl; }};

class developer : public employee {

public:
string programming language
void set developer (string lang) {
programming language = lang; }
void display () {
cout << "developer ID " << emp ID << endl;
cout << "name" << name << endl;
cout << "language " << programming language
<< endl; }};

int main()
{ manager m;
m.set employee (" Ravi");
m.set manager ("HR");
m.display();

developer d;
d.set employee ("Anita");
d.set developer (1++);
d.display();
return 0; }

9) Hybrid inheritance

```
#include <iostream>
using namespace std;
class Person
public:
string name;
int age;
void set person (string n, int a){
name = n;
age = a; }};
class student: public person{
public:
int roll no;
void set student (int r){
roll no = r; }};
class academics{
public :
int marks;
void set marks (int m){
marks = m; }};
class sports{
public:
```

PAGE NO.
DATE / /

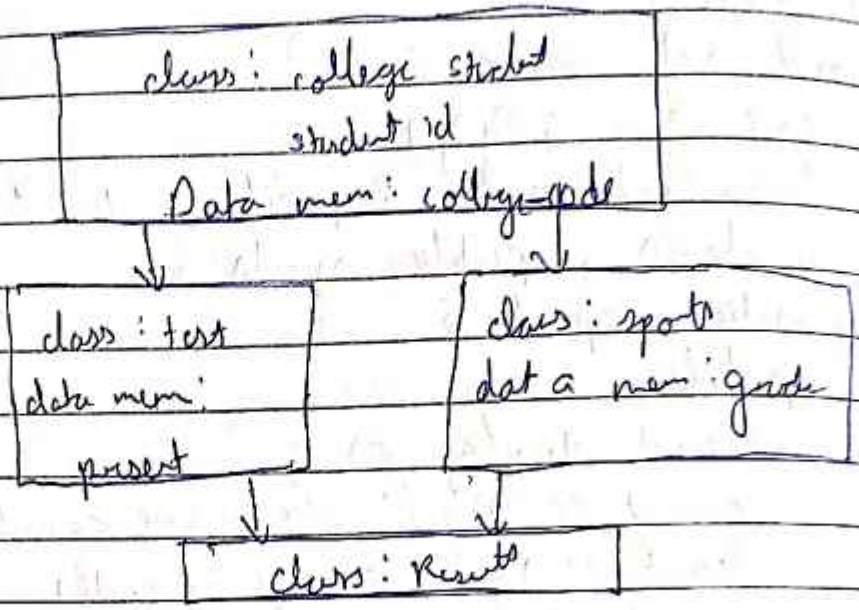
```
int score;
void set score (int s) {
score = s; }};
class result : public student, public
academics, public sports {
public sports {
public:
    void display () {
    cout << "Name": << name << endl;
    cout << "Age" << age << endl;
    cout << "Roll no." << roll no. << endl;
    cout << " sports score :" << score << endl;
    cout << " Total :" << (marks + score) << endl;};

int main
Result
    r. set person ("ria", 19);
    r. set student (101);
    r. set marks (80);
    r. set score (20);
    r. display ();
    return 0; }
```

Output :

Name : ~~re~~ Ria
Age : 19
Roll no. 101
marks = 80
score = 20
Total = 100

PAGE No.
DATE / /

6)




```
#include <iostream>
using namespace std;
class college student
{ protected
  int student_id;
  string college_code;
  public:
  void getstudentData() // void accept();
  { cout << "Enter college code:";
  { cout << "Enter studentID:";
    cin >> student_id;
    cout << "enter college code:";
    cin >> college_code; }
  void show studentData()
  { cout << student ID:" << student_id
    << endl;
    cout << "College code:" << college_
    code << endl; } };
  class Test : Virtual public
    college student { protected:
```

PAGE No.
DATE / /

float percentage;
public:
void getTestData()
{ cout << "Enter Percentage";
cin >> percentage; }
void showTestData()
{ cout << "percentage :" << percentage <<
"%"; }};

class sports : virtual public college Student {
protected:

12/11

PAGE No
DATE / /

Experiment NO 10

Write a C++ code for a simple calculator using class template

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

template <class>
class calculator
{
private:
    num1, num2;

public public:
    Calculator (n1, n2)
    {
    num1 = n1
    num2 = n2;
    }

    void calculator (char op)
    {
    switch (op)
    {
    case '+' :
        cout << " Result : " << num1 + num2 << endl;
        break;
```

```
case '-':
    cout << "Result:" << num1 - num2 << endl;
    break;
case '*':
    cout << "result:" << num1 * num2 << endl;
    break;
case '/': if (num2 != 0)
    cout << "Result:" << num1 / num2 <<
                                    endl;
    else
    cout << "Error: Division by zero!" << endl;
    break;
}
}
};

int main ()
{
double a,b;
  char op;
  cout << "Enter two numbers:";
    cin >> a >> b;
  cout << "Enter operator (+, -, *, /):";
    cin >> op;

  Calculator <double> calc (a, b);
    calc.calculate (op);
    return 0;
}
```

PAGE No.
DATE / /

Experiment No. 8

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

output →

PAGE NO
DATE / /

Enter string : Hello
Enter string : World
Concatenate string : HelloWorld

2) #include <iostream>
using namespace std;
class ILogin {
protected
string name; password;
public;
void accept () {
~~cout~~ cout << "name:"
cin >> name;
cout << "Password:";
cin >> password;
}
}
class Exit login : virtual public ILogin
{
public:
void show Exit () {
cout << "Name <<" " <<

}
}
class reader

PAGE No.
DATE / /

Experiment 11

```
#include <iostream>
#include <vector>
#include <ctype>
using namespace std;
int main (){
  vector <int > vec [5],
  int i;
   cout << "Enter 5 vector elements: ";
   for (i = 0 ; i <5; i++) {
     cout << v << [i] << endl;
   }
   cout << " Modified elements: ";
   for (i=0 ; i<5, i++){
     vec [i] = vec [i] + i * 2;
   }
   for (i=0; i <5 ; i++){
     cout << vec [i] << " ";
   }
   cout << endl;
   int scalar;
     cout << "Enter a scalar value to
     multiply: ";
     cin >> scalar;
     cout << " After multiplying: ";
   for (i=0 ; i<5, i++){
   vec [i] = vec [i] * scalar;
   for (i=0; i<5 ; i++){
     vec[i] = vec [i] * scalar;
   for (i=0;i<5 ; i++) {
     cout << vec [i] << " " ; }
     cout << endl; }
```

Experiment 11 (using iterators)

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

int main () {
vector <int> vec(5);
int scalar;
cout << "Enter 5 vector elements : ";
vector <int> : iterator = vec.begin();
while (itl = vec.end ()) {
    cin >> *it;
    ++ it;
}
cout << "Modified Elements : ";
it = vec begin ();
while (it l = vec.end ()) {
    * it = * it * 2;
    ++ it;
}
it = vec.begin ()
while (it l. = vec end ()) {
it = * it * 2;
++ it;
}
it = vec begin ()
while (it l = vec end () l cout
<< it << " "; ++ it;
cout << "Enter a scalar to multiply.";
cin >> scalar;
cout << " After multiplying : ";
it : vec begin () while (it = vec .end
() {
```

PAGE NO.
DATE / /

```
*it = *it * scalar; ++it;}
    it = vec.begin();
    while (it != vec.end()){
    {cout << *it << " ";
    ++it; } cout << endl;
```

12/11

PAGE No.
DATE / /

Experiment NO.12

```
#include <iostream>
#include <stack>
#include <type>
using namespace std;
 int main () {
  stack <int> v;
   v.push (1),
   v.push (2),
   v.push (3),
   v.push (4),
   v.push (5);
if (v.empty ()) {
    cout << stack is empty >>";
}
else {
    cout << "\n stack is not empty";
}
cout << "\n size "<< v.size ();
cout << "\n      " v.top ();
cout << "\n stack :)
while (!v.empty ())
{
cout << v.top () << " v.pop.
}
cout << \n size after popping
     << v.size ();
}
```

2)
```
#include <iostream>
#include <queue>
#include <type>
using namespace std;
int main () {
    queue <int> v;
    v.push (11)
    v.push (22);
    v.push (33);
    v.push (44);
    v.push (55);
    if (v.empty ()) {
        cout << "\n queue empty";
    }
    else {
        cout << "\n queue is not empty";
    }
    else {
        cout << "\n que is not empty";
    }
    cout << "\n size ; " << v.size();
    cout << "\n front ; " << v.front ();
    cout << "\n queue ;";
    while (!v.empty ()) {
        cout << " v.front () << " " } v.pop ();
    }
    cout << "\n size after
        popping : " << v.size
        ();
}
```

12/11

Experiment 9

1)

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

int main () {
  if stream in file ("First.txt");
  if stream out file ("second.txt");

  if (!infile) {
  cout << "Error opening First.txt" <<
    return 1;
  }
  char ch;
  while (infile.get(ch)) {
      outfile.put(ch);
  }
  cout << "File copied successfully" <<
  infile.close();
    outfile.close();
    return 0;
  }
```

2)

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

int main() {
    ifstream file("First.txt");
    if (file)
        cout << "Error opening File" << endl;
    return 1;
}
char ch;
int digits = 0, spaces = 0;
while (file.get(ch)) {
    if (isdigit(ch))
        digit++
    else if (isspace(ch))
        spaces++;
}
cout << "Digits: " << digits <<
        endl;
cout << "spaces: " << spaces <<
        endl;

file.close();
return 0;
}
```

3)

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

using namespace std;

int main(){
    ifstream file("First.txt");
    if (file) {
        cout << "Error" << endl;
        return 1;
    }

    string word;
    int count = 0;

    while (file >> word)
        count++;

    cout << "Total words: " << count << endl;
    file.close();
    return 0;
```

a)

```
#include <iostream>
#include <fstream>
```

PAGE No.
DATE / /

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

int main () {
if stream file ("First.txt");
if (file);
cout << "Error " << endl;
return 1;

}
string word; target;
int count = 0;

cout << "Enter word to count" endl;
cin >> target;

while (file >> word){
if (word = target)
count ++;
}
cout << "Occurrence of '"
<<target << "' : "
<< count << endl;
file.close();
}
```

Q1

12/11