

EXPERIMENT 1

1. Write a program to declare a class book having data members as bid, bname, p - price. Accept data for 2 books and display name of book having greater price.

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

```
class book {
```

```
public :
```

```
int price;
```

```
int pages;
```

```
String name;
```

```
void accept () {
```

```
cout << "Enter book name : ";
```

```
cin >> name;
```

```
cout << "Enter price : ";
```

```
cin >> price;
```

```
cout << "Enter pages : ";
```

```
cin >> pages;
```

```
}
```

```
void display () {
```

```
cout << "Book name is : " << name;
```

```
cout << "Book price is : " << price;
```

```
cout << "Pages of book are : " << pages;
```

```
int get price () {
```

```
return price;
```

```
}
```

```
};
```

```

int main () {
    book b1, b2;
    b1.accept ();
    b2.accept ();
    if (b1.get_price () > b2.get_price ()) {
        cout << "Book1 has higher price \n";
        b1.display ();
    }
    else {
        cout << "Book2 has higher price \n";
        b2.display ();
    }
    return 0;
}

```

Output - Enter book name : Programming in C

Enter price : 120

Enter pages : 230

Enter book name : matilda

Enter price : 230

Enter pages : 120

Book2 has higher price

Book name is matilda

Book price is 230

Book pages of book are 120

Write a program to declare a class student having data members as roll, name. Accept and display data for one object.

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

class student {
public:
    int roll_no;
    string name;
    void display() {
        cout << "name is: " << name << endl;
        cout << "roll no is: " << roll_no << endl;
    }
};

int main() {
    student s1;
    s1.roll_no = 28;
    s1.name = "Anushka";
    s1.display();
    return 0;
}
```

~~Output - name is: Anushka
roll no is : 28~~

Write a program to declare a class time.
Accept time in HH:MM:SS and display total
time in seconds

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

Class Times {

public :

int hour, min, sec, total;

void accept () {

cout << "Enter hours : ";

cin >> hour;

cout << "Enter minutes : ";

cin >> min;

cout << "Enter seconds : ";

cin >> sec;

}

void calculateTotal () {

total = hour * 3600 + min * 60 + sec * 1;

}

void display () {

cout << "Total time in seconds is " <<

total; total << "seconds" << endl;

}

int main () {

Time t;

t.accept ();

t.calculateTotal ();

t.display ();

return 0;

}

Q
78

Output - Enter hours : 2

Enter minutes : 13

Enter seconds : 45

Total time in seconds is 8025 seconds

EXPERIMENT 2

Write a C++ program to demonstrate the use of array of objects

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

```
#include <iostream>
```

```
using namespace std;
```

```
class city {
```

```
private:
```

```
string name;
```

```
public:
```

```
int population;
```

```
void accept () {
```

```
cout << "enter name: ";
```

```
cin >> name;
```

```
cout << "enter population: ";
```

```
cin >> population;
```

```
}
```

```
void display () {
```

```
cout << "name: " << name << "
```

```
population: " << population << endl;
```

```
}
```

```
int main () {
```

~~city c[5];~~

```
int i, max;
```

```
for (i=0; i<5; i++) {
```

```
c[i].accept ();
```

```
c[i].display ();
```

```
}
```

```
max = C[0].population;
for(C i=1; i<5; i++) {
    if(C[i].population > max) {
        max = C[i].population;
    }
}
```

```
cout << "max population is: " << max << endl;
return 0;
}
```

Output - enter name: pune
enter population : 12000

name: pune, population: 1200

enter name: mumbai

enter population : 150000

name: mumbai, population : 150000

enter name: nagpur

enter population : 1100

name: nagpur, population : 1100

not enter name : nashik

enter population : 1

name: nashik, population : 1

enter name : alibaug

enter population : 5000

name: alibaug, population : 5000

max population : 150000

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

```
#include <iostream>
using namespace std;
class Account {
    int a-no, balance;
public:
    void accept() {
        cout << "Enter account number: ";
        cin >> a-no;
        cout << "Enter account balance: ";
        cin >> balance;
    }
    void disp() {
        cout << "The account number is: " << a-no << endl;
        cout << "Your account balance is: " << balance;
    }
};

int main() {
    Account a1;
    if(balance >= 5000) {
        balance = balance * 0.05;
    }
    a1.accept();
    a1.disp();
}
```

Output -

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

```
#include <iostream>
```

```
using namespace std;
```

```
class Staff {
```

```
    string name;
```

```
    string post;
```

```
public:
```

```
    void accept();
```

```
    cout << "Enter name and post: ";
```

```
    cin >> name >> post;
```

```
    void display();
```

```
{
```

```
    if (post == "HOD")
```

```
{
```

```
    cout << "HOD: " << name << endl;
```

```
}
```

```
?;
```

```
int main()
```

```
{
```

```
    Staff s[5];
```

```
    for (i = 0; i < 5; i++)
```

```
{
```

```
    cout << "Enter details for staff: "
```

```
    << i + 1 << endl;
```

```
s[i].accept();
```

3

```
cout << "list of HODs : " << endl;
```

```
for (i=0; i<5; i++)
```

3

```
s[i].display();
```

3

```
return 0;
```

3

Output - Enter details for staff

Enter name and post : Anushka

HOD

Enter name and post : Neel

HOD

Enter name and post : Nakul

Assistant

Enter name and post : Aarya

Lecturer

Enter name and post : Sejal

Clerk

List of HODS

HOD : Anushka

HOD : Neel

Q1

78

EXPERIMENT 3

Pointer to object, 'this' pointer, nested class

1. Write a program to declare a class 'book' containing data members as book - title, author name and price. Accept and display the information for one object using a pointer to that object.

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

```
Class Book {
    int price;
    String book-title, author-name;
public:
    void.getdata () {
        Cout << "Enter book name: ";
        getline (cin, this → book-title);
        cout << "Enter author name: ";
        getline (cin, this → author-name);
        cout << "Enter price: ";
        cin >> price;
    }
};
```

```
Void void disp () {
    this → getdata ();
    Cout << book-title << "by" << author-name
    << "Price: " << price;
}
};
```

```
int main () {  
    book * bp;  
    bp = & b1;  
    bp->disp();  
    return 0;  
}
```

2. Write a program to declare a class 'student' having data members roll-no and percentage. Use 'this' pointer to accept and display data for one object.

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

```
class student {  
    int roll-no;  
    float per;  
public:  
    void accept ();  
    cout << "Enter roll number: ";  
    getline (cin, this->roll-no);  
    cout << "Enter percentage: ";  
    getline (cin) >> per;  
}  
  
void disp () {  
    this->accept ();  
    cout << " Roll no " << roll-no << ". Percentage: " <<  
    per;  
}  
  
}; b1;
```

```
int main () {  
    b1.display ();  
    return 0;  
}
```

3. Write a program to demonstrate the use of nested classes
- ```
#include <iostream>
using namespace std;
```

```
class Enclosing {
 int n = 14;
public:
 void display () {
 cout << "Calling from the enclosing
 class, n = " << n << endl;
 }
}
```

```
class Nested {
 int n = 31;
public:
 void display () {
 cout << "Calling from nested class,
 n = " << n << endl;
 }
};
```

```
int main () {
 Enclosing e;
 Enclosing :: Nested n;
 e.display ();
 n.display ();
}
```

Q

128

# EXPERIMENT 4

Write a C++ program to demonstrate passing of object as function argument.

1. WAP to swap two numbers from the same class using object as function argument. Ensure that swap is member.

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

```
Class Num {
```

```
 int x, y;
```

```
public:
```

```
void accept () {
```

```
 cout << "Enter first number: ";
```

```
 cin >> x;
```

```
 cout << "Enter second number: ";
```

```
 cin >> y;
```

```
}
```

```
void disp () {
```

```
 cout << "First number: " << x << endl;
```

```
 cin << cout << "Second. number: " << y << endl;
```

```
}
```

```
void swap (Num &n) {
```

```
 int temp = n.x;
```

```
 n.x = n.y;
```

```
 n.y = temp;
```

```
}
```

```
g$ n;
```

```
int main () {
```

```
 n.accept ();
```

PAGE

```
n.disp();
cout << "\n Swapping Numbers...\n\n";
n.swap(n);
n.disp();
return 0;
}
```

2. Write a program to swap 2 numbers of the same class using friend function.

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

```
class Num {
 int a, b;
public:
 void accept() {
 cout << "Enter first number: ";
 cin >> a;
 cout << "Enter second number: ";
 cin >> b;
 }
 void disp() {
 cout << "First number: " << a << endl;
 cout << "Second number: " << b << endl;
 }
 friend void swap(Num &n);
}
```

```
void swap(Num &n) {
```

```
int temp = n.a;
n.a = n.b;
n.b = temp;
}
int main() {
 n.accept();
 n.disp();
 cout << "In swapping numbers...\n\n";
 swap(n);
 n.disp();
 return 0;
}
```

3. WAP to swap 2 no.s of different classes using friend function.

```
#include <iostream>
```

```
using namespace std;
```

```
class B;
```

```
class A {
```

```
 int a;
```

```
public:
```

```
void accept() {
```

```
 cout << "Enter first number: ";
```

```
 cin >> a;
```

```
}
```

```
void disp() {
```

```
 cout << "First Number: " << a << endl;
```

```
}
```

```
friend void swap(A &a, B &b);
```

```
} n;
```

```
class B {
 int x;
public:
 void accept () {
 cout << "Enter second number : ";
 cin >> x;
 }
 void disp () {
 cout << "Second number : " << x << endl;
 }
 friend void swap (A &a, B &b);
};
```

```
void swap (A &a, B &b) {
 int temp = a.x;
 a.x = b.x;
 b.x = temp;
}
int main () {
 n1.accept ();
 n2.accept ();
 n1.disp ();
 n2.disp ();
 cout << "In Swapping Numbers... \n";
 n1.disp ();
 n2.disp ();
 return 0;
}
```

Output - Enter first number : 21

Enter second number : 71

First Number : 21

Second Number : 71

Swapping Numbers...

First Number : 71

Second Number : 21

4. Write a program to find the greatest no. from 2 separate classes. Use friend function.

```
#include <iostream>
```

```
using namespace std;
```

```
Class A;
```

```
Class B {
```

```
 int b;
```

```
public:
```

```
void accept () {
```

```
 cout << "Enter second Number: ";
```

```
 cin >> b;
```

```
}
```

```
friend string findgreater (A &a, B &b);
```

```
b;
```

```
Class A {
```

```
 int a;
```

```
public:
```

```
void accept () {
```

```
 cout << "Enter first Number: ";
```

```
 cin >> a;
```

```
3
friend string findgreater(A&a, B&b);
3 a;
```

```
String find greater(A&a, B&b) {
```

```
if (a.b == b.a) {
```

```
 return "Both numbers are equal";
```

```
3
```

```
else if (a.b > b.a) {
```

```
 return "The first number is greater";
```

```
3
```

```
else {
```

```
 return "The second number is greater";
```

```
33
```

```
int main () {
```

```
 a.accept();
```

```
 b.accept();
```

```
String is greater = find greater(a,b);
```

```
cout << is greater;
```

```
return 0;
```

```
3
```

Output - Enter first number : 5

Enter second number : 25

The second number is greater

5. WAP to create 2 classes, Result 1 and Result 2. Which stores the marks of a student. Read the value of marks for both classes & Compute the avg using friend function.

```
#include <iostream>
using namespace std; class Result 2;
class Result 1 {
 float R1;
public:
 void accept () {
 cout << "Enter marks for subject 1: ";
 cin >> R1;
 }
}
```

```
friend float getAvg (Result 1 & a,
 Result 2 & b);
}
```

```
Class Result 2 {
```

```
 float R2;
public:
 void accept () {
 cout << "Enter marks for subject 2: ";
 cin >> R2;
 }
}
```

~~```
friend float getAvg (Result 1 & a, Result 2
                     & b) {
```~~

```
    float avg = (a.R1 + b.R2) / 2;
    return avg;
}
```

int main () {

x1.accept();

x2.accept();

float avg = getAvg(x1, x2);

cout << "Average marks of the Student"

is " << avg;

return 0;

}

Output - Enter marks for subject 1 : 97.5

Enter marks for subject 2 : 98.2

Average marks of the Student is 97.85

Qm

18

EXPERIMENT 4 & 5

Write a C++ program to implement types of constructors.

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.

* Default constructor

```
#include <iostream>
using namespace std;
class Num {
    int n;
    int i;
    int sum = 0;
public:
    Num(int nt) {
        n = nt; n = 5;
    }
    void add() {
        for(i=1; i<=n; i++) {
            sum = sum + i;
        }
    }
    void disp() {
        cout << "sum is : " << sum;
    }
}
```

```
int main () {  
    Num no(5);  
    no.add();  
    no.disp();  
    return 0;  
}
```

* Parameterized constructor

```
#include <iostream>  
using namespace std;  
class Num {  
    int n;  
    int i;  
    int sum = 0;  
public:  
    Num(int n1) {  
        n = n1;  
    }  
    void add () {  
        for (i=1; i<=n; i++) {  
            sum = sum + i;  
        }  
    }  
    void disp () {  
        cout << "Sum is: " << sum;  
    }  
};
```

```
int main () {
    Num no(5);
    no.add ();
    no.disp ();
    return 0;
}
```

* Copy Constructor

```
#include <iostream>
using namespace std;
class Num {
    int n, i;
    int sum = 0;
public:
    Num(int n1) {
        n = n1;
    }
    void add () {
        for (i = 1; i <= n; i++) {
            sum = sum + i;
        }
    }
    void disp () {
        cout << "sum is : " << sum << endl;
    }
};

int main () {
```

```
Num no1(5);  
Num no2(no1);  
no1.add();  
no2.add();  
no1.disp();  
no2.disp();  
return 0;
```

{

Output for default and parameterized
Constructor -
Sum is : 15

Output for copy constructor -
Sum is : 15
Sum is : 15

2. Write a program to declare a class "student" having data members as name and percentage. Write a constructor to initialize these data members. Accept and display data. for one student.

* Using parameterized constructor

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

```
Class Student {  
    int per;  
    String name;
```

```

public :
    Student (int p, string n) {
        per = p;
        name = n;
    }
    void disp() {
        cout << "Name of Student : " <<
            name << endl;
        cout << "Percentage = " << per << endl;
    }
};

int main () {
    Student s1 (93, "Anushka");
    s1.disp();
    return 0;
}

```

Output - Name of student : Anushka
 Percentage : 93

3. Define a class "College" members variables as roll-no, name, course. WAP using constructor with default value as "Computer Engineering" for course. Accept this data for two objects of class and display the data.
- ```

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

```

```
class College {
```

```
 int roll_no;
```

```
 string name;
```

```
 string course;
```

```
public:
```

```
 College(string n, int r = 25, string
```

```
 c = "CSE") {
```

```
 name = n;
```

```
 roll_no = r;
```

```
 course = c;
```

```
 void display() {
```

```
 cout << name << endl;
```

```
 cout << roll_no << endl;
```

```
 cout << course;
```

```
int main() {
```

```
 College c1("Anushka");
```

```
 c1.display();
```

```
 return 0;
```

**Output - Anushka**

25

CSE

4. Write a program to demonstrate  
Constructor overloading.

```

#include <iostream>
using namespace std;
class square {
 int s;
public:
 square() {
 s = 4;
 }
 square(int side) {
 s = side;
 }
 void calculate() {
 int a;
 a = s * s;
 cout << "Area of square = "
 << a << endl;
 }
};

int main() {
 square s1;
 square s2(5);
 square s3(s2);
 s1.calculate();
 s2.calculate();
 s3.calculate();
 return 0;
}

```

Output - Area of square = 16

Area of square = 25

Area of square = 25

# EXPERIMENT 6

Write a C++ program to implement inheritance

1. Write a program to implement single inheritance

```
#include <iostream>
#include <string>
using namespace std;
class Person {
protected:
 string name;
 int age;
};

class Student : protected Person {
public:
 int roll;
 void accept() {
 cout << "Enter name of the person: ";
 cin >> name;
 cout << "Enter age of the person: ";
 cin >> age;
 cout << "Enter roll number of the
person: ";
 cin >> roll;
 }

 void display() {
 cout << endl;
 cout << "Name: " << name << endl;
 cout << "Roll Number: " << roll << endl;
 cout << "Age: " << age << endl;
 }
}
```

Q.

```
int main () {
 Student s;
 s.accept ();
 s.display ();
 return 0;
}
```

Output - Enter name of the person : Anushka  
Enter age of the person : 16  
Enter roll number of the person : 25

Name : Anushka

Roll number : 25

Age : 16

2. Write a program to implement ~~inher~~ multiple inheritance.

Problem : Create ~~a~~ <sup>two</sup> base class called Person with attributes classes Academic and Sports.

- Academic contains marks of a student.
- Sports class contains marks of a student.

Create a derived class Result that inherits from both Academic & Sports. Write a function to calculate the total score and display details.

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
class Academic {
```

protected:

int m;

};

class Sports {

protected:

int score;

};

class Result : protected Academic, protected Sports {

private:

int t;

public:

void accept () {

Cout << "Enter academic marks: ",

Cin >> m;

Cout << "Enter sports marks: ",

Cin >> score;

}

void calculate () {

t = m + score;

Cout << "Total score is: " << t;

}

};

int main () {

Result r;

r.accept ();

r.calculate ();

return 0;

}

Output - Enter Academic marks : 90  
Enter sports marks : 96  
Total score is : 186

3. Create a class Vehicle with attributes like brand and model. Derive a class Car from Vehicle which adds an attribute type (e.g. sedan, SUV etc) Further derive a class Electric car which adds battery capacity.

• Multilevel Inheritance

#include <iostream>

#include <string>

using namespace std;

class Vehicle {

public :

string brand;

string model;

};

class Car : public Vehicle {

protected :

string type;

};

class ElectricCar : protected Car {

private :

int capacity;

public :

void accept () {

cout << "Enter a car brand : ";

cin >> brand;

```
cout << "Enter car model : ";
cin >> model;
cout << "Enter car type : ";
cin >> type;
cout << "Enter battery capacity : ";
cin >> capacity;
}
};

int main () {
 ElectricCar e;
 e.accept ();
 e.display ();
 return 0;
}
```

Output - Enter car brand : tata  
Enter car model : nexon  
Enter car type : SUV  
Enter battery capacity : 2000  
Brand : tata  
Model : nexon  
Type : SUV  
Battery : 2000

4. Create a base class Employee with attributes emp ID and name. Derive two classes Manager and Developer from Employee.  
Manager has an attribute department and developer has an attribute programming language.

## • Hierarchical inheritance

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
class Employee {
```

```
protected:
```

```
 string name;
```

```
 int empid;
```

```
};
```

```
class Manager : public Employee {
```

```
private:
```

```
 string dept;
```

```
public:
```

```
void accept () {
```

```
 cout << "Enter department name: ";
```

```
 cin >> dept;
```

```
}
```

```
void display () {
```

```
 cout << "Department: " << dept << endl;
```

```
}
```

```
};
```

```
class Developer : protected Employee {
```

```
private:
```

```
 string pl;
```

```
public:
```

```
void accept () {
```

```
 cout << "Enter programming
```

```
language: ";
```

```
 cin >> pl;
```

```
PAGE 11
void display() {
 cout << "Programming language: " << endl;
}
int main() {
 Manager m;
 m.accept();
 m.display();
 Developer d;
 d.accept();
 d.display();
 return 0;
}
```

Output - Enter department name : IT  
Department : IT

Enter programming language : C++  
Programming language : C++

5. Combine multilevel and multiple inheritance.

Create a base class Person with attributes name and age.

Derive a class Student from Person.

Create two classes Sports & Academics. Derive class Result from Students & Sports. Write functions to calculate & display total marks & sports score along with student details.

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

```
class Person {
```

```
protected:
```

```
 string name;
```

```
 int age;
```

```
public:
```

```
 void accept();
```

```
 cout << "Enter name: ";
```

```
 cin >> name;
```

```
 cout << "Enter age: ";
```

```
 cin >> age;
```

```
}
```

```
 void display();
```

```
 cout << "Name: " << name << endl;
```

```
 cout << "Age: " << age << endl;
```

```
}
```

~~```
class Student : public person {
```~~~~```
protected:
```~~~~```
    int sid;
```~~~~```
public:
```~~~~```
    void accept();
```~~~~```
 cout << "Enter student ID: " ;
```~~~~```
        cin >> sid;
```~~~~```
}
```~~~~```
    void display();
```~~

```
cout << "Student ID : " << sid << endl;
```

}

};

class

Class Sports {

protected:

int sScore;

public:

void accept()

```
cout << "Enter sports score : ";
```

```
cin >> sScore;
```

}

void display()

```
cout << "Sports Score : " << sScore << endl;
```

}

};

class Academics {

protected:

int marks;

public:

void accept()

```
cout << "Enter academic marks : ";
```

```
cin >> marks;
```

void display()

```
cout << "Academic marks : " << marks << endl;
```

}

class Result : public Student, public Sports {
private :

Academics academic ;

public:

void display () {

```
cout << "Total score: " << (marks +  
sscore) << endl;
```

3

2.

int main() {

Person p

```
p.accept();  
p.display();
```

Student 5:
accept ()

s.display();

spots. Sp
s. accent ()

sp.-display C;

Academics

a. accept();
b. display();

Result :-

g.accept();

g: display(),

Enter student name : Anushka

Entered age: 16

Enter student ID: 1339

Enter sports score: 85
Enter academic marks: 90

Enter academic marks.

80

Output - & Name : Anushka

Age : 16

Age: 16
Student ID: 1339 Sports Score: 85

Academic marks: 90

Total score: 175

Per
16/10

EXPERIMENT 7

(Write a program to demonstrate the compile time polymorphism (function overloading and operators overloading))

WAP using function overloading - to calculate the area of a laboratory (which is rectangular) and area of classroom (which is a square)

~~# 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 the  
    laboratory : " ;  
    cin >> length >> breadth ;  
    cout << "Area of laboratory : " << area  
    ( length, breadth ) << endl ;
```

```
Cout << "Enter side of classroom : " ;
```

```
Cin >> side ;
```

```
Cout << "Area of classroom : " << area ( size ) << endl ;
```

Output - Enter length and breadth of the laboratory : 5 2

Area of laboratory : 10

Enter side of classroom : 5

Area of classroom : 25

2. Write a program using function overloading to calculate the area, sum of 5 float values and sum of 10 integer values.

```
#include <iostream>
```

```
using namespace std;
```

```
class Sum {
```

```
public:
```

```
    float calculate (float a, float b, float c,  
                    float d, float e) {  
        return a + b + c + d + e;
```

```
}
```

```
    int calculate (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 () {
```

```
    Sum s;
```

```
    cout << "sum of 5 float values: " << s.calculate  
         (1.1, 2.2, 3.3, 4.4, 5.5) << endl;
```

```
cout << "Sum of 10 integers: " << s.calculate  
    (1, 2, 3, 4, 5, 6, 7, 8, 9, 10) << endl;  
    return 0;
```

Output - Sum. of 5 floats values : 16.5
Sum of 10 integer values : 55

3. Write a program to implement Char++ operator (for pre-increment and post-Increment) when used with the object so that then the numeric data member of the class is negated.

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

```
class num {
```

```
int i;
```

```
public:
```

```
void accept () {
```

```
cout << "Enter the value: ";
```

```
cin >> i;
```

```
}
```

```
void display () {
```

```
cout << "Value is: " << i;
```

```
}
```

```
void operator++ () {
```

```
i = -i + i;
```

```
}
```

```
g;  
int main () {  
    num n;  
    n.accept ();  
    ++n;  
    n.display ();  
    return 0;  
g}
```

Output - Enter the value : 8
Value is : 9

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

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

```
Class num {
```

```
    int i;
```

```
public:
```

```
void accept () {
```

```
    cout << "Enter the value : "
```

```
    cin >> i;
```

```
}
```

```
void display () {
```

```
    cout << "Value is : " << i << endl;
```

```
}
```

```
void operator++ () {  
    ++i;  
}  
  
void operator-- () {  
    --i;  
}  
  
void operator++ (int) {  
    i++;  
}  
  
void operator-- (int) {  
    i--;  
}  
  
int main () {  
    num n, n1;  
    n.accept ();  
    ++n;  
    cout << "After prefix increment ";  
    n.display ();  
    n1.accept ();  
    n1--;  
    cout << "After postfix decrement ";  
    n1.display ();  
}
```

Enter the value : 10

Output - After prefix increment value is : 11

Enter the value : 10

After postfix increment value is : 9

Q
X 6110

EXPERIMENT 8

Write a program to demonstrate the compile time and run time polymorphism
(operator overloading Binary)

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

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
class join {
```

```
    string a, b;
```

```
public:
```

```
void accept () {
```

```
    cout << "Enter first string: ";
```

```
    cin >> a;
```

```
    cout << "Enter second string: ";
```

```
    cin >> b;
```

```
}
```

```
string concatenated () {
```

```
    return a + b;
```

```
}
```

```
}
```

```
int main () {
```

```
    join j;
```

```
    j.accept ();
```

```
String result = j.concatenate();
cout << "Concatenated string is : "
      << result << endl;
return 0;
```

Output - Enter first string: MTT

Enter second string: WPU

Concatenated string is : MITWPU

2. Write a program to create a base class ILogin having datamembers name and password.

Declare accept () function and derive EmailLogin and MembershipLogin classes from ILogin. Display Email login details and membership login details of the 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;

```
    } display  
virtual void accept () {  
    cout << "Enter name : "; cin << name << endl;  
    cout << "password : "; cin << password << endl;  
}  
  
}; // class EmailLogin  
class EmailLogin : public Ilogin {  
private:  
    string email;  
public:  
    void accept () override {  
        cout << "Email Login : " << endl;  
        ILogin::accept ();  
        cout << "Enter email : ";  
        cin >> email;  
    }  
  
    void display () override {  
        cout << "Email login Details  
are : " << endl;  
        ILogin::display ();  
        cout << "Email : " << email << endl;  
    }  
  
}; // class MembershipLogin  
class MembershipLogin : public ILogin {  
private:  
    int membershipId;  
public:  
    void accept () override {  
        cout << "Membership Login : " << endl;  
    }  
};
```

```
TLogin::accept();  
cout << "Enter membership ID : ";  
cin >> membershipID;
```

y

```
void displayOver () override {
```

```
Cout << "Membership login details  
are : " << endl;
```

```
TLogin::display();
```

```
Cout << "Membership ID : " <<
```

```
membershipID << endl;
```

y

y;

```
int main () {
```

```
EmailLogin emailuser;
```

```
MembershipLogin memberuser;
```

```
Cout << "Enter email login details : "  
<< endl;
```

```
emailuser.accept();
```

```
Cout << "In Enter Membership login  
details : " << endl;
```

```
memberuser.display();
```

```
Cout << "In Displaying email login  
details : " << endl;
```

```
emailuser.display();
```

```
cout << "In Displaying Membership  
login Details : " << endl;  
memberUser.display();  
  
return 0;  
}
```

Output - Enter email login details :

Email login :

Enter name : anushka

Enter password : anu@2008

Enter email : anushka.mohite@gmail.com

Enter membership login details :

Membership Login :

Enter name : Anushka

Enter password : ajjsd

Enter membership ID : 1111

Displaying Email Login details :

Name : anushka

password : anu@2008

email : anushka.mohite@gmail.com

Displaying Membership Login details :

Name : Anushka

password : ajjsd

Qn
16/10

EXPERIMENT 9

Write a program to perform various operations on file.

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

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

```
int main () {
    char ch;
    ifstream fin ("first.txt");
    ofstream fout ("destination.txt");
    if (!fin) {
        cout << "ERROR in opening destination
file" << endl;
        fin.close ();
        return 1;
    }
    while (fin.get (ch)) {
        fout << ch;
    }
    fin.close ();
```

```
fout.close();  
cout << "File is successfully copied.";  
return 0;
```

Output - first.txt
Welcome to MIT WPU Pune

destination.txt
Welcome to MIT WPU Pune

2. Write a C++ program to count words using file handling.

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

```
int main() {  
    string word;  
    string a;  
    int count = 0;  
    int a_count = 0;  
    if stream fin ("source.txt");  
    ofstream fout ("destination.txt");  
  
    cout << "Enter a word: ";  
    cin >> a;
```

```
if (!fout) {
    cout << "ERROR in opening source
file." << endl;
    return 1;
}
```

```
if (!fout) {
    cout << "ERROR in opening destination
file." << endl;
    fin.close();
    return 1;
}
```

```
while (fin >> word) {
    fout << word;
    count++;
    if (word == a) {
        a_count++;
    }
    fout << word;
}
```

```
fin.close();
fout.close();
cout << "Total number of words = " <<
a_count;
return 0;
```

Output - Total number of words = 2

3. Write a C++ program to count occurrence of a given word using file handling.

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

```
int main () {
    string word;
    string a;
    int count = 0;
    int a_count = 0;
    ifstream fin ("source.txt");
    ofstream fout ("destination.txt");
    cout << "Enter a word: ";
    cin >> a;

    if (!fin) {
        cout << "ERROR in opening source
file." << endl;
        return 1;
    }

    if (!fout) {
        cout << "ERROR in opening destination
file." << endl;
        fin.close ();
        return 1;
    }
```

```
while (fin >> word) {  
    fout << word;  
    count++;  
    if (word == a) {  
        a_count++;  
    }  
}
```

```
fin.close();  
fout.close();  
cout << "Total number of words: " repeats  
     << a_count;  
return 0;  
}
```

Output - source.txt
Welcome to mit wpu, wpu pune

Enter a word : wpu
Total number of repeats = 2

Write a C++ program to count digits
and spaces using file handling.

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

```
int main () {  
    char ch;
```

```
int digitcount = 0;  
int spacecount = 0;  
ifstream fin ("source.txt");  
ofstream fout ("destination.txt");  
  
if (!fin) {  
    cout << "Error in opening source  
file" << endl;  
    return 1;  
}  
  
if (!fout) {  
    cout << "ERROR in opening  
destination file" << endl;  
    fin.close();  
    return 1;  
}  
  
while (fin.get(ch)) {  
    fout << ch;  
    if (isdigit(ch)) {  
        digitcount++;  
    }  
    if (!isspace(ch)) {  
        spacecount++;  
    }  
}  
  
fin.close();  
fout.close();
```

```
cout << "File is successfully copied"  
      << endl;
```

```
cout << "Number of digits : "
      << digitcount << endl;
```

```
cout << "Number of Spaces : "  
     << spacecount << endl;
```

between 0;

Output - source.txt
welcome to mit 2025

~~Number of digits : 4
Number of spaces : 3~~

~~Q~~ 16110

EXPERIMENT 10

Write a program to implement a Function template and class template.

1. Write a C++ program to find sum of array elements using function template (e.g. Pass Integer, Float and 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] = { 1.2, 1.2, 2.4 };

```

```
    double a3[3] = { 10.5, 20.5, 30.5 };

```

~~Cout << "Sum of integer array is: "~~

~~<< func(a1, 3) << endl;~~

~~Cout << "Sum of integer or float array is: " << func(a2, 3)~~

~~<< endl;~~

~~Cout << "Sum of double array is: "~~

~~<< func(a3, 3)~~

~~<< endl;~~

Output - sum of integer array : 4

Sum of float array : 4.8

Sum of double array : 61.5

Write a C++ Program of square function using template specialization.

```
#include <iostream>
```

```
using namespace std;
```

```
template <class T> T square(T x) {
```

```
    T sum = 0;
```

```
    sum = x * x;
```

```
    return sum;
```

```
}
```

```
template <> string square <string>
```

```
(string ss) {
```

```
    return (ss + ss);
```

```
}
```

```
int main () {
```

~~int i = 2; ii;~~~~String ww("MIT");~~~~ii = Square <int>(i);~~~~Cout << "Square of 2 is : " << ii <<~~~~endl;~~~~Cout << "Square of given string
is : " << square <<
String(ww) << endl~~

Output - Square of 2 is : 4

Square of given string is : MIT MIT

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

```
#include <iostream>
```

```
#include <math>
```

```
using namespace std;
```

```
template <class T>
```

```
class Calculator {
```

```
private:
```

```
    T 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() {
```

```
        return num1 / num2;
```

```
}
```

```
    T mod() {
```

```
        return fmod(num1, num2);
```

```
}
```

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

```
int main () {  
    double a, b;  
    int ch;  
    cout << "Enter two numbers : ";  
    cin >> a >> b;  
    Calculator<double> calc(a, b);  
    do {
```

```
Cout<<"\n Choose an operation:\n";
```

```
cout << "1. Addition\n";
```

Cout << "2. Subtraction \n".

cout << "3. Multiplication In";

```
cout << "4. Division\n";
```

Cout << "5. Modulus\n".

~~Cout << " / 6 .~~

~~Cout << "6. Exit \n";~~

```
Cout << "Enter your choice : ";
```

$\text{Cin} \gg \text{ch}$;

switch (ch) {

case 1 :

```
cout << "Addition : " << calc.add ()  
      << endl;
```

break;

case 2:

cout << "Subtraction: " << calc.Sub()

<< endl;

break;

case 3:

cout << "Multiplication: " << calc.mul()

<< endl;

break;

case 4:

cout << "Division: " << calc.div()

division() << endl;

break;

case 5:

cout << "Modulus: " << calc.mod()

<< endl;

break;

case 6:

cout << "Exiting..." << endl;

break;

default:

cout << "Invalid choice!!" <<

endl;

}

while (ch != 6);

return 0;

}

Output - Enter two numbers : 2 2

Choose an operation :

1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Modulus
6. Exit

Enter your choice : 4

Division : 1

Choose an operation :

1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Modulus
6. Exit

Enter your choice : 6

Exiting...

Qm
2/11

CAREER IITEN I

Write a C++ program to implement generic vectors. Include the following member functions:

To create the vector.

- (a) To modify the value of a given element.
- (b) To multiply by a scalar value.
- (c) To display the vector in the form (10, 20, 30...)

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std;
```

```
template <typename T>
```

```
class GenericVector {
```

```
    vector<T> data;
```

```
public:
```

```
    void create(int n) {
```

```
        data.resize(n);
```

```
        cout << "Enter " << n << "element:"
```

```
        << endl;
```

```
        for (int i = 0; i < n; i++) {
```

```
            cin >> data[i];
```

```
}
```

```
    void modify(int index, T value) {
```

```
        if (index >= 0 && index < data.size()) {
```

```
            data[index] = value;
```

```
}
```

```
    void multiply(T scalar) {
```

```
for (auto & el : data)
    el * = scalar;
void display () {
    cout << "vector : ";
    for (const auto & el : data)
        cout << el << " ";
    cout << endl;
}
};
```

```
int main () {
    GenericVector < int > vec;
    int n;
    cout << "Enter no. of elements : ";
    cin >> n;
    vec.create (n);
    vec.display ();
    cout << "Modify element at index 1 to
           20 : " << endl;
    vec.modify (1, 20);
    vec.display ();
    cout << "Multiply all elements by 2 : "
        << endl;
    vec.multiply (2);
    vec.display ();
}
return 0;
```

Output - Enter number of elements : 3

En- Enter 3 elements :

10 15 30

vector : 10 15 30

Modify element at index 1 to 20 :

Vector : 10 20 30

Multiply all elements by 2 :

Vector : 20 40 60

* Without Iterator :

```
#include <iostream>
```

```
#include <vector>
```

```
#include <cctype>
```

```
using namespace std;
```

```
int main () {
```

```
    vector <char> v30;
```

```
    unsigned int i;
```

```
    cout << " size = " << v.size()
```

```
        << endl;
```

```
    for (i=0; i<v.size(); i++) {
```

```
        cout << v[i] << " ";
```

```
        cout << "\n\n";
```

```
}
```

```
    cout << "Expand vectors: \n";
```

```
    for (i=0; i<v.size(); i++) {
```

```
        v.push-back (i+40+'a');
```

```
        cout
```

```
        cout << "size now = " << v.size() << endl;
```

```
        cout << "Current contents: \n";
```

```
for(i=0; i<v.size(); i++) {  
    cout << v[i] << " "  
    cout << "\n\n";  
}  
for(i=0; i<v.size(); i++) {  
    v[i] = toupper(v[i]);  
    cout << "Modify Contents : ";  
    for(i=0; i<v.size(); i++) {  
        cout << v[i] << " ";  
    }  
    cout << "\n\n";  
}  
return 0;  
}
```

Output - Initial size = 5

Original contents : a b c d e

Expand vectors

Size now = 10

Current contents = a b c d e f g h i j ;

Modifying contents to uppercase -

Modified contents : A B C D E F G H I J .

Q
plll

EXPERIMENT 12

Write C++ program using STL

- (a) Implement stack (b) Implement Queue
(c) Implement Sorting and Searching with user-defined records such as Person Record (Name, birth date, telephone no), item record (item code, item name, quantity & cost).

Implement Stack :

```
#include <iostream>
#include <stack>
using namespace std;
int main () {
    stack<int> S;
    S.push(10);
    S.push(20);
    S.push(30);
    cout << "Stack top: " << S.top() << endl;
    S.pop();
}
```

~~cout << "Stack top after pop: " << S.top()
 << endl;~~

return 0;

Output - Stack top : 30
Stack top after pop : 10

Implement Queue :

```
#include <iostream>
```

```
#include <queue>
```

Using namespace std;

```
int main()
```

```
{ q.push(100);
```

```
q.push(200);
```

```
q.push(300);
```

```
cout << "Queue front : " << q.front()
```

```
<< endl;
```

```
q.pop();
```

```
cout << "Queue front after pop : " <<
```

```
q.front() << endl;
```

```
return 0;
```

Output - Queue front : 100

Queue front after pop : 200.

Qn

T2/11