I

Name _____

Roll No. _____ Year 20____20____

Exam Seat No. _____

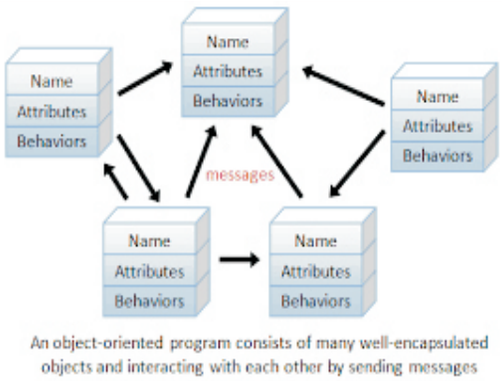
COMPUTER GROUP | SEMESTER - III | DIPLOMA IN ENGINEERING AND TECHNOLOGY

A LABORATORY MANUAL

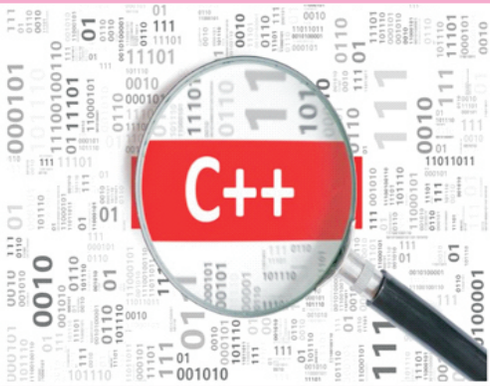
FOR

OBJEECT ORIENTED PROGRAMMING USING C++

(22316)



An object-oriented program consists of many well-encapsulated objects and interacting with each other by sending messages



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION, MUMBAI
(Autonomous) (ISO 9001 : 2015) (ISO / IEC 27001 : 2013)

VISION

To ensure that the Diploma level Technical Education constantly matches the latest requirements of technology and industry and includes the all-round personal development of students including social concerns and to become globally competitive, technology led organization.

MISSION

To provide high quality technical and managerial manpower, information and consultancy services to the industry and community to enable the industry and community to face the changing technological and environmental challenges.

QUALITY POLICY

We, at MSBTE are committed to offer the best in class academic services to the students and institutes to enhance the delight of industry and society. This will be achieved through continual improvement in management practices adopted in the process of curriculum design, development, implementation, evaluation and monitoring system along with adequate faculty development programmes.

CORE VALUES

MSBTE believes in the followings:

- Education industry produces live products.
- Market requirements do not wait for curriculum changes.
- Question paper is the reflector of academic standards of educational organization.
- Well designed curriculum needs effective implementation too.
- Competency based curriculum is the backbone of need based program.
- Technical skills do need support of life skills.
- Best teachers are the national assets.
- Effective teaching learning process is impossible without learning resources.

A Laboratory Manual

for

Object Oriented Programming Using C++

(22316)

Semester-III

(IF/CO/CM/CW)

Maharashtra State
Board of Technical Education, Mumbai
(Autonomous) (ISO:9001:2015) (ISO/IEC 27001:2013)

Maharashtra State Board of Technical Education,
(Autonomous) (ISO : 9001 : 2015) (ISO/IEC 27001 : 2013)
4th Floor, Government Polytechnic Building, 49, Kherwadi,
Bandra (East), Mumbai - 400051.
(Printed on June, 2018)

MAHARASHTRA STATE

BOARD OF TECHNICAL EDUCATION

Certificate

This is to certify that Mr. / Ms. ……..…… Roll No. ……..…., of Third Semester of Diploma in ……..…… of Institute, ……..…….……..… (Code: ……...) has completed the term work satisfactorily in course **Object Oriented Programming Using C++ (22316)** for the academic year 20…….. to 20…….. as prescribed in the curriculum.

Place: …… Enrollment No:.……

Date: …… Exam. Seat No: ……

Subject Teacher **Head of the Department** **Principal**

Seal of Institution

Preface

The primary focus of any engineering laboratory/ field work in the technical education system is to develop the much needed industry relevant competencies and skills. With this in view, MSBTE embarked on this innovative 'I' Scheme curricula for engineering diploma programmes with outcome-base education as the focus and accordingly, relatively large amount of time is allotted for the practical work. This displays the great importance of laboratory work making each teacher; instructor and student to realize that every minute of the laboratory time need to be effectively utilized to develop these outcomes, rather than doing other mundane activities. Therefore, for the successful implementation of this outcome-based curriculum, every practical has been designed to serve as a ***'vehicle'*** to develop this industry identified competency in every student. The practical skills are difficult to develop through 'chalk and duster' activity in the classroom situation. Accordingly, the 'I' scheme laboratory manual development team designed the practicals to ***focus*** on the ***outcomes***, rather than the traditional age old practice of conducting practicals to 'verify the theory' (which may become a byproduct along the way).

This laboratory manual is designed to help all stakeholders, especially the students, teachers and instructors to develop in the student the pre-determined outcomes. It is expected from each student that at least a day in advance, they have to thoroughly read through the concerned practical procedure that they will do the next day and understand the minimum theoretical background associated with the practical. Every practical in this manual begins by identifying the competency, industry relevant skills, course outcomes and practical outcomes which serve as a key focal point for doing the practical. The students will then become aware about the skills they will achieve through procedure shown there and necessary precautions to be taken, which will help them to apply in solving real-world problems in their professional life.

This manual also provides guidelines to teachers and instructors to effectively facilitate student- centered lab activities through each practical exercise by arranging and managing necessary resources in order that the students follow the procedures and precautions systematically ensuring the achievement of outcomes in the students.

In the modern world of Information technology, the Object Oriented Programming has become the most preferred approach for software development. It offers a powerful way to cope up with complexity of real world problems. Among the OOP languages available, C++ is the primitive language which develops fundamental understanding of Object Oriented Concepts. This course enables students to develop programs in 'C++' using Object Oriented Programming approach.

Although best possible care has been taken to check for errors (if any) in this laboratory manual, perfection may elude us as this is the first edition of this manual. Any errors and suggestions for improvement are solicited and highly welcome

Programme Outcomes (POs) to be achieved through Practical of this Course:

PO1. **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.

PO2. **Discipline knowledge:** Apply Information Technology knowledge to solve broad-based Information Technology related problems.

PO3. **Experiments and practice:** Plan to perform experiments, practices and to use the results to solve Information Technology related problems.

PO4. **Engineering tools:** Apply appropriate Information Technology related techniques/ tools with an understanding of the limitations.

PO7. **Ethics:** Apply ethical principles for commitment to professional ethics, responsibilities and norms of the practice also in the field of Computer engineering.

PO8. **Individual and team work:** Function effectively as a leader and team member in diverse/ multidisciplinary teams.

PO9. **Communication:**. Communicate effectively in oral and written form.

PO10. **Life-long learning:** Engage in independent and life-long learning activities in the context of technological changes in the Computer engineering field and allied industry.

Practical- Course Outcome matrix

Course Outcomes (COs)

a. Develop C++ programs to solve problems using Procedure Oriented Approach.
b. Develop C++ programs using classes and objects.
c. Implement Inheritance in C++ program.
d. Use Polymorphism in C++ program.
e. Develop C++ programs to perform file operations

S. No.	Practical Outcome	CO a.	CO b.	CO c.	CO d.	CO e.
1.	Develop minimum 2 programs using constants, variables, arithmetic expression, operators, exhibiting data type conversion.	√	-	-	-	-
2.	Develop a program to implement decision making statements (If-else, switch).	√	-	-	-	-
3.	Develop a program to demonstrate control structures (for, while, do-while).	√	-	-	-	-
4.	Develop a program to implement 1-dimension array.	√	-	-	-	-
5.	Develop a program to perform matrix operations using multi-dimensional array.	√	-	-	-	-
6.	Develop programs that implements a class and use it with objects.	-	√	-	-	-
7.	Develop programs that implements a class and create array of objects.	-	√	-	-	-
8.	Write a program to implement friend function.	-	√	-	-	-
9.	Write a program to implement inline function.	-	√	-	-	-
10.	Write a program to implement all types of constructors (constructor overloading) with destructor.	-	√	-	-	-
11.	Write a program for implementing single inheritance	-	-	√	-	-
12.	Write a program for implementing multi-level inheritance.	-	-	√	-	-

13.	Write a program for implementing multiple inheritance.	-	-	√	-	-
14.	Develop minimum 1 program to demonstrate Pointer to object.	-	-	√	√	-
15.	Develop minimum 1 program to demonstrate Pointer to derived class	-	-	√	√	-
16.	Write a program to demonstrate operator overloading for Unary operator.	-	-	-	√	-
17.	Write a program to demonstrate operator overloading for Binary operator	-	-	-	√	-
18.	Write a program to demonstrate function overloading	-	-	-	√	-
19.	Write a program to read and write data to and from a file.	-	-	-		√

List of Industry Relevant Skills

The following industry relevant skills of the competency to develop applications using OOPs concepts in C++ are expected to be developed in you by undertaking the practical's of this laboratory manual.

1. Develop an application by implementing Inheritance.
2. Develop an application by using Polymorphism.
3. Use appropriate File handling operations for developing applications.

Guidelines to Teachers

1. There will be two sheets of blank pages after every practical for the student to report other matters (if any), which is not mentioned in the printed practicals.
2. For difficult practicals if required, teacher could provide the demonstration of the practical emphasizing of the skills which the student should achieve.
3. Teachers should give opportunity to students for hands-on after the demonstration.
4. Assess the skill achievement of the students and COs of each unit.
5. One or two questions ought to be added in each practical for different batches. For this teachers can maintain various practical related question bank for each course.
6. For effective implementation and attainment of practical outcomes, teacher ought to ensure that in the beginning itself of each practical, students must read through the complete write- up of that practical sheet.
7. During practical, ensure that each student gets chance and takes active part in taking observations/ readings and performing practical.
8. Teacher ought to assess the performance of students continuously according to the MSBTE guidelines.

Instructions for Students

Note: Kindly do add specific instructions for students for effective implementation of practical's depending upon your course, if needed.

1. For incidental writing on the day of each practical session every student should maintain a ***dated log book*** for the whole semester, apart from this laboratory manual which has to ***submit for assessment to the teacher*** in the next practical session.
2. For effective implementation and attainment of practical outcomes, in the beginning itself of each practical, students need to read through the complete write-up including the practical related questions and assessment scheme of that practical sheet.
3. Student ought to refer the reference books, lab manuals, etc.
4. Student should not hesitate to ask any difficulties they face during the conduct of practical's.

Content Page

List of Practical's and Progressive Assessment Sheet

Sr. No	Practical Outcome	Page No.	Date of perfor mance	Date of submi t- ssion	Assess- ment marks(25)	Dated sign. of teacher	Rema rks (if any)
13.	Write a program for implementing multiple inheritance.	82					
14.	Develop minimum 1 program to demonstrate Pointer to object.	92					
15.	Develop minimum 1 program to demonstrate Pointer to derived class	99					
16.	Write a program to demonstrate operator overloading for Unary operator.	106					
17.	Write a program to demonstrate operator overloading for Binary operator	115					
18.	Write a program to demonstrate function overloading	125					
19.	Write a program to read and write data to and from a file.	132					
Total							

To be transferred to Proforma of CIAAN-2017.

Practical No. 1: Develop minimum 2 programs using constants, variables, arithmetic expression, operators, exhibiting data type conversion.

I Practical Significance:

1. The declaration of constants and variables facilitates the evaluation of arithmetic expressions in real life problems.
2. The data type conversion helps in real life problems to get the accurate results of evaluation of arithmetic expressions.

II Relevant Program Outcomes (POs)

1. **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
2. **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
3. **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
4. **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
5. **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define and declare constants and variables.
2. Use data type conversion.
3. Compile the program.
4. Debug and execute the program.

IV Relevant Course Outcome(s)

Develop C++ programs to solve problems using Procedure Oriented Approach..

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using constants, variables, arithmetic expressions and data type conversion.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow ethical practices.

VII Minimum Theoretical Background

C++ Basic Input/Output:

C++ I/O operation is using the stream concept. Stream is the sequence of bytes or flow of data. It makes the performance fast. If bytes flow from main memory to device like printer, display screen, or a network connection, etc, this is called as output operation. If bytes flow from device like printer, display screen, or a network connection, etc to main memory, this is called as input operation.

Header File	Function and Description
<iostream>	It is used to define the cout, cin and cerr objects, which correspond to standard output stream, standard input stream and standard error stream, respectively.
<iomanip>	It is used to declare services useful for performing formatted I/O, such as setprecision and setw.
<fstream>	It is used to declare services for user-controlled file processing.

Standard output stream (cout)

The cout is a predefined object of ostream class.

It is connected with the standard output device, which is usually a display screen.

The cout is used in conjunction with stream insertion operator (<<) to display the output on a console.

Standard input stream (cin)

The cin is a predefined object of istream class.

It is connected with the standard input device, which is usually a keyboard.

The cin is used in conjunction with stream extraction operator (>>) to read the input from a console.

Standard end line (endl)

The endl is a predefined object of ostream class.

It is used to insert a new line characters and flushes the stream.

Constants: A variable which does not change its value during execution of a program is known as a constant variable. Any attempt to change the value of a constant will result in an error message. A constant in C++ can be of any of the basic data types, const qualifier can be used to declare constant as shown below:

```
const float PI = 3.1415;
```

The above declaration means that PI is a constant of float types having a value 3.1415.

Examples of valid constant declarations are:

```
const int RATE = 50;
const float PI = 3.1415;
const char CH = 'A';
```

Literal constants: can be classified into: integer, floating-point, characters, strings, Boolean, pointers, and user-defined literals.

Other literals: Three keyword literals exist in C++: true, false and nullptr: true and false are the two possible values for variables of type bool. nullptr is the null pointer value.

```
bool foo = true;
bool bar = false;
int* p = nullptr;
```

Preprocessor definitions (#define): Another mechanism to name constant values is the use of preprocessor definitions. They have the following form:

Syntax: #define identifier replacement

Example : **#define PI 3.14159**

#define NEWLINE '\n'

Basic Data Types: C++ supports a large number of data types. The built in or basic data types supported by C++ are integer, floating point and character. C++ also provides the data type bool for variables that can hold only the values True and false. Some commonly used data types are summarized in table along with description.

Type	Description	Size in bytes
Int	Small integer number	2
long int	Large integer number	4
float	Small real number	4
double	Double precision real number	8
long double	Long double precision real number	8
char	A Single Character	1

Variable: Variable is a location in the computer memory which can store data and is given a symbolic name for easy reference. The variables can be used to hold different values at different times during the execution of a program.

Declaration of a variable and Initialization:

Example:
```
float total;
int x, y;
int a = 20;
```

Type Conversion: The process in which one pre-defined type of expression is converted into another type is called conversion. There are two types of conversion in C++.

1. Implicit conversion
2. Explicit conversion

Implicit conversion: Data type can be mixed in the expression. For

Example double a;

int b = 5;

float c = 8.5;

a = b * c;

When two operands of different type are encountered in the same expression,

Explicit conversion: It is also called type casting. It temporarily changes a variable data type from its declared data type to a new one.

Syntax: (type) expression;

Example: (float) (x + y/2);

VIII Resources required :

Sr. No	Name of Resource	Specification	Quantity	Remarks
1.	Hardware : Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2.	Operating system	Windows /LINUX		
3.	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specificati
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Constant variables can be created in C++ by using _____ .
   a. const b. #define d. Both a& b d. None of these

2. State output of the following code:

```
#include <iostream.h>
 void  main()
 {
     typedef int num;
     num a = 10, b = 15;
```

```
        num c = a + b + a - b;
        cout << c;

    }
```

3. Regarding following statement which of the statements is true? const int a = 100;
   a) Declares a variable a with 100 as its initial value
   b) Declares a construction a with 100 as its initial value c) Declares a constant a whose value will be 100
   d) Constructs an integer type variable with a as identifier and 100 as value

4. Which of the following statement is not true about preprocessor directives?
   a) These are lines read and processed by the preprocessor b) They do not produce any code by themselves
   c) These must be written on their own line d) They end with a semicolon

XIII Exercise

Attempt Q1 or Q2 and Q.3 a or b from the following:
(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to evaluate the following expressions:

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}, \quad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

2. Write a C++ program to demonstrate the use of operator precedence.

3. Complete the given table:

Program Code	Write & justify Output

a) `#include<iostream.h>`
`#define PI 3.14159`
`int main ()`
`{`
`float r = 2;`
`float circle;`
`circle = 2 * PI * r;`
`cout << circle;`
`return 0;`
`}`	

a) `#include<iostream.h>`
`#include<conio.h>`
`void main()`
`{`
`clrscr();`
`float res;`
`float f1=15.5, f2=2;`
`res = (int)f1/(int)f2;`
`cout<<res<<endl;`
`res = (int)(f1/f2);`
`cout<<res<<endl;`
`res = f1/f2;`
`cout<<res;`
`getch();`
`}`	

(Space for Answers)

XIV References / Suggestions for further Reading

1. https://www.sanfoundry.com/c-plus-plus-interview-questions-experienced-references/
2. http://people.scs.carleton.ca/~dehne/projects/cpp-doc/tutorial/tut1-2.html
3. http://www.cplusplus.com/doc/tutorial/constants/
4. http://home.pacifier.com/~mmead/cs161/lesson03_07.html

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2.	Appropriate use of Variables, constants, arithmetic expressions and data type conversions.	20%
3.	Debugging ability	20%
4.	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 2: Program to implement decision making statements (If-Else, Switch)

I Practical Significance:

The decision making statements like if-else and switch-case helps to decide the order of execution of statements based on certain conditions or repeat a group of statements until certain specified conditions are met.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you:

Develop C++ programs to solve broad-based problems

1. Define decision making statements applicable to the particular problem.
2. Compile the program.
3. Debug and execute the program.

IV Relevant Course Outcome(s)

Develop C++ programs using decision making statements.

V Practical Outcome (POs)

a)Write/ Compile/ debug / Execute simple C++ program using decision making statements.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow ethical practices.

VII Minimum Theoretical Background

Decision Making Statement

1. Decision making statement is depending on the condition block need to be executed or not which is decided by condition.
2. If the condition is "true" statement block will be executed, if condition is "false" then statement block will not be executed.
3. In this section we are discuss about if-then (if), if-then-else (if else), and switch statement. In C++ language there are three types of decision making statement.

if
if-else
switch

if-else Statement :- In general it can be used to execute one block of statement among two blocks, if and else are the keyword in C++.

Syntax

```
if(condition)
{
.....
statements
.....
}
else
{
.....
statements
.....
}
```

Switch Statement:-

A switch statement work with byte, short, char and int primitive data type, it also works with enumerated types and string.

Syntax

```
switch(expression/variable)
{
case  value:
//statements
// any number of case statements
break;  //optional
default: //optional
//statements
}
```

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Observe the following block of code and determine what happens when x=2?

```
switch (x){
case 1:
case 2:
case 3:cout<< "x is 3, so jumping to third
branch"; goto thirdBranch;
default:
cout<<"x is not within the range, so need to say
Thank You!";
        }
```

2. Conditional operator in C++ is a
   a) Unary operator b) Binary operator c)Ternary operator d) None of them

3. Which of the following is selection statement in C++?
   a) break b) goto c) exit d) switch

[Space for Answers]

.....

.....

.....

.....

.....

XIII Exercise:

Attempt Q1 or Q2 or Q3 and Q.4 a ,b or a, c from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program check leap year.
2. Write a C++ program armstrong numbers between two integers.
3. Write a C++ program to check whether a number is palindrome or not.
4. Complete the given table:

Program Code	Write & justify Output

a) `#include <iostream.h>`
`void main()`
`{`
`int a = 10;`
`if (a < 15)`
`{`
`time:`
`cout << a;`
`goto time;`
`}`
`break;`
`}`	

b)
`#include <iostream.h>`
`long factorial (long a)`
`{`
`if (a > 1)`
`return (a * factorial (a + 1));`
`else`
`return (1);`
`}`
`int main ()`
`{`
`long num = 3;`
`cout << num << "! = " << factorial ( num );`
`}`	

c) `#include <iostream.h>`
`void main()`
`{`
`int percentage;`
`cout << "Enter the percentage : ";`
`cin >> percentage;`
`switch (percentage / 10)`
`{`
`case 10: case 9:`
`cout << "You have got grade A+" << endl;`
`break;`
`case 8:`
`cout << "You have got grade A" << endl;`
`break;`
`case 7:`	

```
        cout << "You have got grade B+" << endl;
        break;
    case 6:
        cout << "You have got grade B" << endl;
        break;
    case 5:
        cout << "You have got grade C" << endl;
        break;
    default:
        cout << "You have got grade D" << endl;
        break;
    }
}
```

(Space for Answers)

XIII References / Suggestions for further Reading

1. https://www.programiz.com/c-programming/examples/leap-year
2. https://www.cpp.thiyagaraaj.com/c-programs/c-basic-example- programs/if-else-example-program-in-c
3. https://www.sanfoundry.com/cpp-program-illustrate-switch-statement/

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate use of Decision making statements (if-else , switch)	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 3: Program to Implement Control Structure (For, While, Do-While)

I Practical Significance:

Loops are used to execute a set of statements repeatedly until a particular condition is satisfied.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Use appropriate looping construct(for, while, do-while) in the given problem.
2. Compile the program.
3. Debug and execute the program.

IV Relevant Course Outcome(s)

Develop C++ programs to solve problems using Procedure Oriented Approach.

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using for loop, while loop and do-while loop .

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow ethical practices.

VII Minimum Theoretical Background

Control Structure

Loops are used in programming to repeat a specific block until some end condition is met. There are three types of loops in C++ programming:

for loop
while loop
do...while loop

for loop:-

- **Syntax**

```
for(initializationStatement; testExpression; updateStatement)
 {
```

```
        // codes
}
```

How for loop works?

- The initialization statement is executed only once at the beginning. Then, the test expression is evaluated.
- If the test expression is false, for loop is terminated.
- But if the test expression is true, codes inside body of for loop is executed and update expression is updated.
- Again, the test expression is evaluated and this process repeats until the test expression is false.

while Loop:-

The syntax of a while loop is:

```
while (testExpression)
{
    // codes
}
```

where, test Expression is checked on each entry of the while loop.

How while loop works?

The while loop evaluates the test expression.
If the test expression is true, codes inside the body of while loop is evaluated.
Then, the test expression is evaluated again. This process goes on until the test expression is false.
When the test expression is false, while loop is terminated.

do-while Loop:-

The do...while loop is a variant of the while loop with one important difference. The body of do...while loop is executed once before the test expression is checked.

The syntax of do..while loop is:

```
do {

  // codes;

}
while (testExpression);
```

How do...while loop works?

- The codes inside the body of loop is executed at least once. Then, only the test expression is checked.
- If the test expression is true, the body of loop is executed. This process continues until the test expression becomes false.
- When the test expression is false, do...while loop is terminated.

VIII Resources required

Sr.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Observe the following block of code and determine what happens when x=2?

```
#include<iostream.h>
void main ()
{
int a,d,n,sum,term=0;

cout<<"Enter the first term, common difference,"
```

```
<<"and the number of terms to be summed"
<<"respectively:\n";
Cin>>a>>d>>n;
sum=0;

int i=1;
cout<<"\nThe terms are ";
do
{
  term= a+ (i-1)*d;
  sum+=term;
  cout<<term<<" ";
  ++i;
}
while (i<=n) ;
cout<<"\nThe sum of A.P. is "<<sum;
}
}
```

2. Which loops needs a semi colon after?
   a) for b) do c) while

3. In the while statement, while(x<100)..., when does the statement controlled by the condition execute?
a. When x is less than one hundred
b. When x is greater than one hundred
c. When x is equal to one hundred
d. While it wishes

(Space for Answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIII Exercise

Attempt any 2 from 1-4 and Q.5 a or b from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to print a multiplication table of 7
2. Write a C++ program to print 100 to 1.
3. Write a C++ program which will print (half pyramid) pattern of natural numbers.

```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

4. Write a C++ program; ask the user to enter the number of rows to print the pyramid of stars (*)

```
    *
   * *
  * * *
 * * * * *
* * * * * * *
```

5. Complete the given table:

Program Code	Write & justify Output

a) #include <iostream.h>
int main()
{
int a, d, n;
 	
std::cout << "Enter a, d, n of AP: ";
std::cin >> a >> d >> n;
std::cout << std::endl << "Terms of AP : a = " << a
<< ", d = " << d << ", n = " << n << std::endl;
do {
std::cout << a << "\t";
a = a + d;
}
while (n-- > 1);
std::cout << std::endl;
}	

b)
#include <iostream.h>
#include <stack.h>
int main()
{
std::stack< char > s;
 	
// Pushing elements into stack
for (char c = 'a'; c <= 'f'; c++)
{
s.push(c);
}
while (!s.empty())
{
std::cout << "Top element of stack \'" << s.top()
<< "\'" << std::endl;
s.pop();
}
std::cout << "Stack is empty!" << std::endl;
}	

(Space for Answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIV References / Suggestions for further Reading

1. https://www.programiz.com/cpp-programming/for-loop
2. http://ecomputernotes.com/cpp/control-structures/iteration-statements
3. https://www.sanfoundry.com/cpp-program-demonstrate-do-while-loop/
4. https://codescracker.com/cpp/program/cpp-program-print-star-pyramid-patterns.htm

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate use of control structure (for, while,do-while)	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1 ……..

2 ……..

3 ……..

4 ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 4: Program to Implement One Dimension Array

I Practical Significance:

The arrays helps to represent collection of similar type of data under common name.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define one dimensional array.
2. Compile the program.
3. Debug and execute the program.

IV Relevant Course Outcome(s)

Develop C++ programs to solve problems using Procedure Oriented Approach..

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using one dimensional arrays.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow ethical practices.

VII Minimum Theoretical Background

Arrays in C++

Array is a collection of data of same types stored in sequential memory location. It is a linear data structure, where data is stored sequentially one after the other. The elements in an array is accessed using an index.

For example, In an array of n elements, the first element has index *zero* and the last element has index *(n-1)*.

Elements with consecutive index (i.e. I and i+1) are stored in consecutive memory location in the system.

Array can be divided into following types:

One Dimensional Array

Multi-Dimensional Array

One Dimensional Array Syntax:

type arrayName [arraySize];

Examples:

```
double salary[15000];
int age[5]={22,25,30,32,35};
```

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. State output of the following code:

```
#include <iostream.h>
int main()
{
    int n, i;
```

```
    float num[100], sum=0.0, average;

    cout << "Enter the numbers of data: ";
    cin >> n;

    while (n > 100 || n <= 0)
    {
cout << "Error! number should in range of (1 to 100)." <<
     endl;
        cout << "Enter the number again: ";
        cin >> n;
    }
    for(i = 0; i < n; ++i)
    {
        cout << i + 1 << ". Enter number: ";
        cin >> num[i];
        sum += num[i];
    }
    average = sum / n;
    cout << "Average = " << average;
    return 0;
}
```

2. Which of the following correctly declares an array?
   a) int array[10]; b) int array;
   c) array{10}; d) array array[10];
3. Can we change the size of an array at run time?
4. What is the default value of Array?
5. Can we declare array size as a negative number?
6. What is the meaning of anonymous array? Explain with an example?
7. Is there any difference between int[] a and int a[]?
8. How to copy an array into another array?
9. Can we change the size of an array at run time?
10. Can you declare an array without assigning the size of an array?

(Space for Answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIII Exercise

Attempt any 2 programs from 1-4 and Q.5 attempt any 2 from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to find median of two sorted arrays of same size.
2. Write a C++ program to find the two repeating elements in a given array.
3. Find the smallest and second smallest elements in an array.
4. Write a C++ program to find the Missing Number.
5. Complete the given table:

Program Code	Write & justify Output

a) `#include <iostream.h>`
`int array1[] = {1200, 200, 2300, 1230, 1543};`
`int array2[] = {12, 14, 16, 18, 20};`
`int temp, result = 0;`
`void main()`
`{`
`for (temp = 0; temp < 5; temp++)`
`{`
`result += array1[temp];`
`}`
`for (temp = 0; temp < 4; temp++)`
`{`
`result += array2[temp];`
`}`
`cout << result;`
`}`	

b) `#include <iostream.h>`
`void main ()`
`{`
`int array[] = {0, 2, 4, 6, 7, 5, 3};`
`int n, result = 0;`
`for (n = 0; n < 8; n++) {`
`result += array[n];`
`}`
`cout << result;`
`}`	

c) `#include <iostream.h>`
`void main()`
`{`
`char str[5] = "ABC";`
`cout << str[3];`
`cout << str;`
`}`	

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIV References / Suggestions for further Reading

1 https://www.w3schools.in/cplusplus-tutorial/arrays/
2. https://www.geeksforgeeks.org/c-programs-gq/array-programs-gq/
3. https://www.sanfoundry.com/c-programming-examples-arrays/
4. https://www.programtopia.net/cplusplus/docs/arrays

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate one dimensional array declaration	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4 ..……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 5: Program to Perform Matrix Operation Using Multi-Dimension Array.

I Practical Significance:

The Two Dimensional Array helps to represent the real life data in tabular format with different operations on it.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1 Define real life problem and solve it using multidimensional array if applicable.

2. Compile the program.

3. Debug and execute the program.

IV Relevant Course Outcome(s)

Develop C++ programs to solve problems using Procedure Oriented Approach.

V Practical Outcome (POs)

a)Write/ Compile/ debug / Execute simple C++ program using multidimensional array.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

VII Minimum Theoretical Background

Multi-dimensional array :-

A multi-dimensional **array** is an array of arrays. 2-dimensional arrays are the most commonly used. They are used to store data in a tabular manner.

- **Syntax:** type arr[row_size][column_size];
- **Example:** int arr[3][5];
- **2D array initialization**: An array can either be initialized during or after declaration. The format of initializing an array during declaration is as follows:
- **Syntax** :type arr[row_size][column_size] = {{elements}, {elements} ... };
- **Example** :

```
int arr[3][5] = {{5, 12, 17, 9, 3}, {13, 4, 8, 14, 1}, {9, 6, 3, 7, 21}};
```

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. What is the error in this C++ code?

```
#include<iostream.>
void main()
{
    int ary[2][3];
    ary[][] = {{1, 2, 3}, {4, 5, 6}};
            cout<<ary[1][0];
```

```
    }
```

2. State output of the following code:

```
#include<iostream.h>
void main()
{
    // initializing the 3-dimensional
    array int x[2][3][2] =
    {
        { {0,1}, {2,3}, {4,5} },
        { {6,7}, {8,9}, {10,11} }
    };

    // output each element's value for
    (int i = 0; i < 2;++i)
    {
        for (int j = 0; j < 3; ++j)
        {
            for (int k = 0; k < 2; ++k)
            {
                cout << "Element at x[" << i << "][" << j
                     << "][" << k << "] = " << x[i][j][k]
                       <<
                      endl;
            }

        }
    }
}
```

3. How many kinds of elements an array can have?
   a) Char and int type b) Only char type
   c) Only int type d) All of them have same type

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIII Exercise

Attempt Q1 or Q2 and Q.3 a or b from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to Multiply Two 3*3 Matrix Using Multi-dimensional Arrays.
2. Write a C++ program to find Transpose of a 2*2 Matrix
3. Complete the given table:

Program Code	Write & justify Output

a)
`#include <iostream.h>`
`void main()`
`{`
`int a[2][3] = {1, 2, 3, , 4, 5};`
`int i = 0, j = 0;`
`for (i = 0; i < 2; i++)`
`for (j = 0; j < 3; j++)`
`cout<<a[i][j];`
`}`	

b)
`#include <iostream.h>`
`void main()`
`{`
`int a[2][3] = {1, 2, 3, 4, 5};`
`int i = 0, j = 0;`
`for (i = 0; i < 2; i++)`
`for (j = 0; j < 3; j++)`
`cout<<a[i][j];`
`}`	

(Space for answers)

.....

.....

.....

.....

XIII References / Suggestions for further Reading

1. https://www.sanfoundry.com/multiple-choice-questions-c-multidimensional-arrays/
2. https://www.hackerearth.com/practice/data-structures/arrays/multi- dimensional/tutorial/
3. https://www.programiz.com/cpp-programming/examples/matrix-transpose

XIV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate multidimensional array definition	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 6: Program to Implement Classes and Objects

I Practical Significance:

The classes and objects help to represent real life entity with different attributes and related member functions.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define real life entity into classes and objects.
2. Define and use classes and objects.
3. Compile the program.
4. Debug and execute the program.

IV Relevant Course Outcome(s)

Develop C++ programs using classes and objects.

V Practical Outcome (POs)

a)Write/ Compile/ debug / Execute simple C++ program using classes and objects.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow ethical practices.

VII Minimum Theoretical Background

- **Class:**. Class is a user defined data type, which holds its own data members and associated member functions, which can be accessed and used by creating an instance of that class.

- **Object:** An **Object** is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated.

```
class ClassName
        {
                Access specifier:       //can be private, public or protected
                Data members;           // Variables to be used
                Member functions()      //Methods to access data members
                {
                        //body of the function
                }
        };              //Class ends with semicolon
```

Declaring Objects: When a class is defined, only the specification for the object is defined; no memory or storage is allocated. To use the data and access functions defined in the class, you need to create objects.

Syntax:

className ObjectName;

eg.

Student std;

Accessing data members and member functions: The data members and member functions of class can be accessed using the dot('.') operator with the object.

For example if the name of object is std and you want to access the member function with the name getData() then you will have to write std.getData() .

Member Functions in Classes: There are 2 ways to define a member function:

Inside class definition:

Syntax:

```
class className
{
        //other members declaration
        return type functionName(list of parameters1)
        {
        // body of function
        }
        //other members declaration
};
```

Outside class definition:

Syntax:

```
return type className :: functionName(list of parameters)
{
        //body of function

}
```

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. What is the difference between struct and class in C++?
2. State output of the following code:

```
#include<iostream.h>
class Empty {};
void main()
{
  cout << sizeof(Empty);
}
```

a) A non-zero value b) 0 c) Compile time error d) Runtime error

3. State True or False: Data items in C++ class are by default public.

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIII Exercise

Attempt Q1 or Q2 and Q.3 a or b from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to declare a class "Book" having data members book_name, author and price. Accept and display data for book having maximum price.

2. Write a C++ program to declare a class "staff" having data members name, basic salary, DA, HRA and calculate gross salary. Accept and display data for one staff.
   a. Where DA=74.5% of basic
      i. HRA= 30% of basic.
      ii. Gross_salary=basic+HRA+DA

3. Complete the given table:

Program Code	Write & justify Output

a) `#include<iostream.h>`
`class Empty { };`
 	
`void main()`
`{`
`  Empty a, b;`
 	
`  if (a == b)`
`  cout << "impossible " <<`
`  endl;`
`  else`
`  cout << "Fine " << endl;`
`}`	

b) #include <iostream.h>
class Rectangle {
int width, height;
public:
void set_values (int,int);
int area();
{return width*height;}
};
void Rectangle::set_values (int x, int y) {
width = x;
height = y;
}
void main () {
rectangle rect;
rect.set_values (3,4);
cout << "area: " <<
rect.area();
 	
}	

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIV References / Suggestions for further Reading

1. https://www.geeksforgeeks.org/c-classes-and-objects/
2. https://www.tutorialspoint.com/cplusplus/cpp_classes_objects.htm

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate class and object definition	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 7: Program to Implement Array of Objects

I Practical Significance:

The classes and objects help to represent real life entity with different attributes and related member functions.

Array of objects are used to represent the data of similar type.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1.Define real life entity into classes and objects.
2..Define and use classes and array of objects.
3..Compile the program.
4..Debug and execute the program.

IV Relevant Course Outcome(s)

Develop C++ programs using classes and objects.

V Practical Outcome (POs)

a)Write/ Compile/ debug / Execute simple C++ program using classes and array of objects.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

VII Minimum Theoretical Background

Array of objects: Arrays of variables of type "class" is known as "Array of objects".

Declaring Array of Objects:

Syntax:

```
className ObjectArrayName[size];
eg.
Student std[5];
```

Accessing data members and member functions: The data members and member functions of class can be accessed using the dot('.') operator with the object.
For example if the name of object is *std[2]* and you want to access the member function with the name *getData()* then you will have to write *std[2].getData()* .

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. State the output of the following program:

```
#include<iostream.h>
 class books
 {
```

```
    char title [30];
    float price ;
    public:
    void getdata ();
    void putdata ();
} ;
void books :: getdata ()
{
    cout<<"Title:”;
    Cin>>title;
    cout<<"Price:”;
    cin>>price;
}

void books :: putdata ()
{
    cout<<"Title:"<<title<< "\n";
    cout<<"Price:"<<price<< "\n”;
    const int size=3 ;
}
void main ()
{
    books book[size] ;
    for(int i=0;i<size;i++)
    {
    cout<<"Enter details o£ book "<<(i+1)<<"\n";
    book[i].getdata();
    }
    for(int i=0;i<size;i++)
    {
    cout<<"\nBook "<<(i+l)<<"\n";
    book[i].putdata() ;
}
```

(Space for answers)

XIII. Exercise

Attempt Q1 or Q2 and Q3 a or b from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to define a class “Employee” having data members emp_id, emp_name and emp_salary. Accept and display data for employees having salary greater than 25,000/-.

2. Write a C++ program to define a class “City” having data members name, population. Accept and display data for 10 cities.

3. Complete the given table:

Program Code	**Write & justify Output**

a)
`#include <iostream.h>`
 	
`class Student`
`{`
`char name[20]; int marks; public:`
`void getName()`
`{`
`cin>>name;`
`}`
`void getMarks()`
`{`
`cin >> marks;`
`}`
`void displayInfo()`
`{`
`cout << "Name : " << name << endl;`
`cout << "Marks : " << marks << endl;`
`}`
`};`
 	
`void main()`
`{`
`Student st[5];`
`for( int i=0; i<5; i++ )`
`{`
`cout << "Student " << i + 1 <<`	

```
          endl;
          cout  <<  "Enter  name"  <<  endl;
          st[i].getName();
          cout  <<  "Enter  marks"  <<  endl;
          st[i].getMarks();
          }

          for(  int  i=0;  i<5;  i++  )
          {
          cout  <<  "Student  "  <<  i  +  1  <<
          endl;
          st[i].displayInfo();
          }
          }
```

b)
```
#include<iostream.h>
#include<conio.h>
    class Employee
    {
        int Id;
        char Name[25];
        int Age;
        long Salary;
        public:


    void GetData()
    {
     cout<<"\n\tEnter Employee Id : ";
     cin>>Id;
     cout<<"\n\tEnter Employee Name : ";
     cin>>Name;
     cout<<"\n\tEnter Employee Age : ";
     cin>>Age;
     cout<<"\n\tEnter Employee Salary : ";
     cin>>Salary;
    }
     void PutData()
    {

cout<<"\n"<<Id<<"\t"<<Name<<"\t"<<Age<<"\t"<<Salary;
    }
};
void main()
{
    int i;
    Employee E[3];
    for(i=0;i<3;i++)
    {
     cout<<"\nEnter details of "<<i+1<<"
           Employee";
     E[i].GetData();
    }

    cout<<"\nDetails of Employees";
    for(i=0;i<3;i++)
        E[i].PutData();
}
```

(Space for answers)

XIII References / Suggestions for further Reading

1. https://www.geeksforgeeks.org/c-classes-and-objects/
2. https://www.tutorialspoint.com/cplusplus/cpp_classes_objects.htm

XIV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate class and array of objects definition	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 8: Program to Implement Friend Function

I Practical Significance:

The classes and objects help to represent real life entity with different attributes and related member functions.

The use of friend function allows non-member functions to access the private and protected data of the class.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define real life entity into classes and objects.
2. Define and use of friend function.
3. Compile the program.
4. Debug and execute the program.

IV Relevant Course Outcome(s)

Develop C++ programs using classes and objects.

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using classes, objects and friend function.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

VII Minimum Theoretical Background

- **Friend Function:** A friend function of a class is defined outside that class' scope but it has the right to access all private and protected members of the class. Even though the prototypes for friend functions appear in the class definition, friends are not member functions. **friend** keyword is used to declare the function as friend of class.
- **Declaring friend function**

  Syntax:

```
class className
{
        //other members declaration
```

```
        public:
            friend return type friendFunctionName(list of parameters);
        //other members declaration
};
return type friendFunctionName(list of parameters)
{
        //body of friend function
}
```

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Can member functions of one class be friend functions of another class?
   a) Yes b) No
2. A function can be declared as friend maximum only in two classes.
   a) True b) False
3. Which of the following rules will not affect the friend function?
   a) private and protected members of a class cannot be accessed from outside
   b) private and protected member can be accessed anywhere
   c) protected member can be accessed anywhere
   d) none of the mentioned
4. Where does keyword 'friend' should be placed?
   a) function declaration
   b) function definition
   c) main function
   d) none of the mentioned

(Space for answers)

XIII Exercise

Attempt Q1 or Q2 and Q.3 a or b from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to swap the values of two variables using friend function.
2. Write a C++ program to define a class "Distance" having data member as meters.

Display the addition of two Distance objects using friend function.

3. Complete the given table:

Program Code	Write & justify Output

a)#include <iostream.h>
class A
{
int a;
public:
A() {a = 10;}
friend void showA(A&);
};
void showA(A& x) {
cout << "a=" << x.a;
}
void main()
{
A a;
showA(a);
}	

#include <iostream.h>
class Box
{
double width;
public:
friend void printWidth(Box box);
void setWidth(double wid);
};
void Box::setWidth(double wid)
{
width = wid;
}
void printWidth(Box box)
{
box.width = box.width * 2;
cout << "Width of box : " << box.width << endl;
}
void main( )
{
Box box;
box.setWidth(10.0);
printWidth(box);
}	

(Space for answers)

XIV References / Suggestions for further Reading

1 https://www.geeksforgeeks.org/c-classes-and-objects/
2 https://www.tutorialspoint.com/cplusplus/cpp_classes_objects.htm
3 https://www.tutorialspoint.com/cplusplus/cpp_friend_functions.htm
4 http://www.sanfoundry.com/multiple-choice-questions-c-plus-plus- friends/

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate friend function declaration and definition	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1 ……..
2 ……..
3 ……..
4 ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 9: Program to Implement Inline Function

I Practical Significance:

The classes and objects help to represent real life entity with different attributes and related member functions.

The use of inline functions facilitates faster execution of the program.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define real life entity into classes and objects.
2. Define and use of inline function.
3. Compile the program.
4. Debug and execute the program.

IV Relevant Course Outcome(s)

Develop C++ programs using classes and objects.

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using classes, objects and inline functions.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

VII Minimum Theoretical Background

Inline Function: C++ inline function is concept that is commonly used with classes. If a function is inline, the compiler places a copy of the code of that function at each point where the function is called at compile time.

A function definition in a class definition is an inline function definition, even without the use of the **inline** keyword.

To inline a function, place the keyword **inline** before the function name and define the function before any calls are made to the function. The compiler can ignore the inline qualifier in case defined function is more than a line.

Declaring inline function:

Syntax:

```
class className
{
        //other members declaration public:
                inline return type FunctionName(list of parameters)
                {
                    //body of inline function
                }
        //other members declaration
};
```

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1 Handle computer system and peripherals with care.
2 Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	

XI Result (Output of the Program)

.....
.....
.....
.....
.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. What does the inline keyword do?
A. Indicates a function declaration
B. Tells the compiler to use the function only within the same source code file
C. Causes all function calls to be replaced by the code from the function
D. Allows one-line function declarations

2. Why would you want to use inline functions?
A. To decrease the size of the resulting program
B. To increase the speed of the resulting program
C. To simplify the source code file
D. To remove unnecessary functions

3. Which of the following is a limit on inline functions?
A. Inline functions cannot return a value.
B. Inline functions must return a value.
C. Inline functions must be less than ten lines.
D. The compiler may choose to ignore an inline directive

4. Which of the following is a valid inline for function foo?
A. inline void foo() {}
B. void foo() inline {}
C. inline:void foo() {}
D. None of the above

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIII Exercise

Attempt Q1 and Q.2 from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to create a class "Number" having data members x and y and perform mathematical operations like addition, subtraction, multiplication and division on two numbers using inline functions.

2. Complete the given table:

Program Code	Write & justify Output

a)#include <iostream.h>
 	
class InlineDemo
{
public:
int square(int s); // declare the function
};
 	
//use inline prefix
 	
inline int InlineDemo::square(int s)
{
return s*s;
}
 	
void main()
{
InlineDemo s;
cout<<"Square of a No is :"<<s.square(10);
}	

(Space for answers)

.....

.....

XIV References / Suggestions for further Reading

1. https://www.geeksforgeeks.org/c-classes-and-objects/
2. https://www.tutorialspoint.com/cplusplus/cpp_classes_objects.htm
3. https://www.geeksforgeeks.org/inline-functions-cpp/

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate inline function declaration and definition	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1 ……..

2 ……..

3 ……..

4 ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 10: Program to Implement Constructors and Destructors

I Practical Significance:

The classes and objects help to represent real life entity with different attributes and related member functions.
The use of constructor functions facilitates initialization of instance variables.
The use of destructor functions facilitates deallocation of object memory.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define real life entity into classes and objects.
2. Define and use of constructor function.
3. Define and use of destructor function.
4. Implement constructor overloading.
5. Compile the program.
6. Debug and execute the program.

IV Relevant Course Outcome(s)

Develop C++ programs using classes and objects.

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using constructors and destructors.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

VII Minimum Theoretical Background

- **Constructor:** Constructors are special member functions which performs initialization of every object. The Compiler calls the Constructor whenever an object is created. Constructors initialize values to instance variables after storage is allocated to the object.

  Constructors are of three types :

  1. Default Constructor
  2. Parameterized Constructor
  3. Copy Constructor

- **Declaring constructor function:**
  Syntax:

```
class className
{
        //other members declaration
        public:
                className(list of parameters)
                {
                    //body of constructor function
                }
        //other members declaration
};
```

- **Constructor Overloading:**

  Just like other member functions, constructors can also be overloaded. In fact when you have both default and parameterized (more than one) constructors defined in your class you are having **Overloaded Constructors**, one with no parameter and other with parameter.

- **Declaring overloaded constructor functions:**
  Syntax:

```
class className
{
        //other members declaration
        public:
                className()   //Default constructor
                {
                    //body of default constructor function
                }
                className(list of parameters)  //Parameterized constructor
                {
                    //body of parameterized constructor function
                }
                className(className &obj)  //copy constructor
                {
                    //body of copy constructor function
                }

        //other members declaration
};
```

Destructor: Destructor is a special member function which destroys the object as soon as the scope of object ends. The destructor is called automatically by the compiler when the object goes out of scope.
The syntax for destructor is same as that for the constructor, the class name is used for the name of destructor, with a **tilde (~)** sign as prefix to it.

Declaring destructor function:

Syntax:

```
class className
{
        //other members
        declaration public:
        //other members declaration

                ~className()
                {
                        //body of destructor
                                function
                }
};
```

VIII Resourcesrequired

Sr. No.	Name of	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. What happens when a class with parameterized constructors and having no default constructor is used in a program and we create an object that needs a zero-argument constructor?
   a. Compile-time error. b. Preprocessing error. c. Runtime error. d. Runtime exception.
2. For automatic objects, constructors and destructors are called each time the objects a. enter and leave scope
   b. inherit parent class c. are constructed
   d. are destroyed

3. Which of the following statement is correct about the program given below?

```
#include<iostream.h>
class abc
{
   public:
   abc()
   {
      cout<< "Welcome";
   }
   ~abc()
   {
      cout<< "India";
   }
};
void main()
{
  abc obj;

}
```

a. The program will print the output Welcome.
b. The program will print the output India.
c. The program will print the output Welcome India.
d. The program will report compile time error.

(Space for answers)

XIII Exercise

Attempt Q1 and Q.2 from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to define a class “Number” having data members x and y and perform mathematical operations like addition, subtraction, multiplication and division on two numbers using constructor and destructor functions.

2. Write a C++ program to define a class “Product” having data members Prod_id, Prod_name, Prod_price. Accept and display data for 3 products using constructor overloading.

3. Complete the given table:

Program Code	Write & justify Output

a)
`#include<iostream.h>`
`int val = 0;`
`class Test`
`{`
`public:`
`Test()`
`{`
`cout<< ++val;`
`}`
`~Test()`
`{`
`cout<< val--;`
`}`
`};`
 	
`void main()`
`{`
`Test obj1, obj2, obj3;`
`{`
`Test obj4;`
`}`
`}`	

b)
`#include <iostream.h>`
`class construct`
`{`
`public:`
`float area;`
`// Constructor with no parameters`
`construct()`
`{`
`area = 0;`
`}`
`construct(int a, int b)`
`{`
`area = a * b;`
`}`
`void disp()`
`{`
`cout<< area<< endl;`
`}`
`};`
`void main()`
`{`
`// Constructor Overloading`
`// with two different constructors`
`// of class name`
`construct o;`
`construct o2( 10, 20);`
`o.disp();`
`o2.disp();`
`}`	

(Space for answers)

XIV References / Suggestions for further Reading

1. https://www.geeksforgeeks.org/c-classes-and-objects/
2. https://www.tutorialspoint.com/cplusplus/cpp_classes_objects.htm
3. http://www.studytonight.com/cpp/constructors-and-destructors-in-cpp
4. https://www.geeksforgeeks.org/constructor-overloading-c/

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate constructor, destructor function declaration and definition	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1……..

2……..

3……..

4……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 11: Program to Implement Single Inheritance

I Practical Significance:

The use of inheritance shows the reusability of the existing class properties and deriving a new class with additional properties.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define real life example class and its derived class with additional properties.
2. Compile the program.
3. Debug and execute the program.

IV Relevant Course Outcome(s)

Implement Inheritance in C++ program.

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using single inheritance.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

VII Minimum Theoretical Background

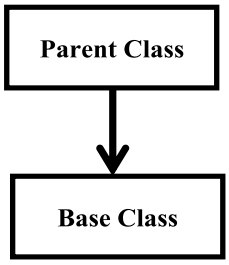
Inheritance: It is the process of inheriting properties of objects of one class by objects of another class. The class which inherits the properties of another class is called Derived or Child or Sub class and the class whose properties are inherited is called Base or Parent or Super class. When a single class is derived from a single parent class, it is called Single inheritance. It is the simplest of all inheritance

Types of Inheritance:

1. Single Inheritance
2. Multiple Inheritance
3. Multilevel Inheritance
4. Hierarchical Inheritance
5. Hybrid Inheritance

Syntax of Single Inheritance:

```
class base_classname
{
   properties;
   member functions;
};
class derived_classname : visibility_mode base_classname
{
   properties;
member functions;
};
```



Single Inheritance

Visibility Modes of Inheritance:

	Derived Class	**Derived Class**	**Derived Class**
Base Class	**public Mode**	**private Mode**	**protected Mode**
Private	Not inherited	Not inherited	Not inherited
Protected	protected	Private	Protected
Public	Public	Private	Protected

VIII Resources required

Sr. No.	**Name of Resource**	**Specification**	**Quantity**	**Remarks**
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. A derived class with only one base class is called …… inheritance.
   a) single
   b) multiple
   c) multilevel
   d) hierarchical
2. When a base class is privately inherited by the derived class, then _____ .
   a. protected members of the base class become private members of derived class
   b. public members of the base class become private members of derived class
   c. both a and b
   d. only b
3. The derivation of Child class from Base class is indicated by ____ symbol.
   a. ::
   b. :
   c. ;
   d. |

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIII Exercise

Attempt Q1 and Q.2 from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to define a class “Student” having data members roll_no, name . Derive a class “Marks” from “Student” having data members m1,m2,m3, total and percentage. Accept and display data for one student.

2. Write a C++ program to define a class “Employee” having data members emp_no, emp_name and emp_designation. Derive a class “Salary” from “Employee” having data members basic, hra, da, gross_sal. Accept and display data for one employee.

3. Complete the given table:

Program Code	Write & justify Output

a)
#include <iostream.h>
 	
class Base
{
public:
Base(){}
~Base(){}
protected:
private:
};
class Derived:public Base
{
public:
Derived(){}
Derived(){}
private:
protected:
};
void main()
{
cout << "The program exceuted" <<
endl;
}	

b) #include<iostream.h>
 	
class Test
{
int value;
 	
public:
Test(int v = 0) { value = v; }
int getValue() { return value;
}
};
 	
void main()
{
const Test t;
cout << t.getValue();
 	
}	

(Space for answers)

XIV. References / Suggestions for further Reading

1. https://www.programtopia.net/cplusplus/docs/single-inheritance/
2. https://www.geeksforgeeks.org/inheritance-in-c/
3. http://www.siteforinfotech.com/2017/05/top-20-mcq-questions-inheritance-in- cpp.html
4. https://www.careerride.com/mcq/inheritance-c-mcq-questions-and-answers-114.aspx
5. https://www.geeksforgeeks.org/output-of-c-program-set/
6. https://ide.geeksforgeeks.org/index.php

XV Assessment Scheme

Performance indicators		Weightage
Process related (35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate base class and derived class declaration using single inheritance	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 12: Program to Implement Multi Level Inheritance

I Practical Significance:

The use of inheritance shows the reusability of the existing class properties and deriving a new class with additional properties.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define real life example class and its multilevel derived class hierarchy with additional properties.
2. Compile the program.
3. Debug and execute the program.

IV Relevant Course Outcome(s)

Implement Inheritance in C++ program.

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using multilevel inheritance.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

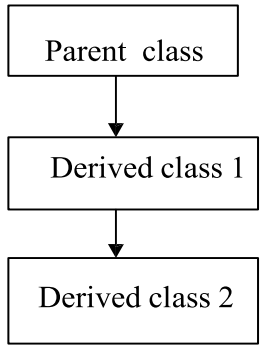
VII Minimum Theoretical Background

Multilevel Inheritance: In C++ programming, not only you can derive a class from the base class but you can also derive a class from the derived class. This form of inheritance is known as multilevel inheritance.

Syntax of multilevel Inheritance:

```
class base_classname
{
   properties;
   member functions;
};
class derived_classname1 : visibility_mode base_classname
{
   properties;
   member functions;
};
class derived_classname2 : visibility_mode derived_classname1
{
   properties;
   member functions;
};
```

Pictorial Representation:



VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1 Handle computer system and peripherals with care.
2 Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....
.....
.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1.

```
#include <iostream.h>
class A
{
public:
void print() { cout << "print() in A"; }
};

class B : private A
{
public:
void print() { cout << "print() in B"; }
};

class C : public B
{
public:
void print() {
cout << "print() in
C"; A::print(); }
};
void main()
{


C b;
b.print();
}
```

a) print() in A b) print() in B

c) Compile time error d) None of the above

2. A class can be derived from another derived class which is known as ……. inheritance.
A) single B) multiple C) multilevel D) hierarchical

3. In ……. inheritance, the constructors are executed in the order of inheritance.
A) multipath B) multiple C) multilevel D) hierarchical

4. base class and derived class relationship comes under
A) Inheritance B) Polymorphism C) encapsulation D)None

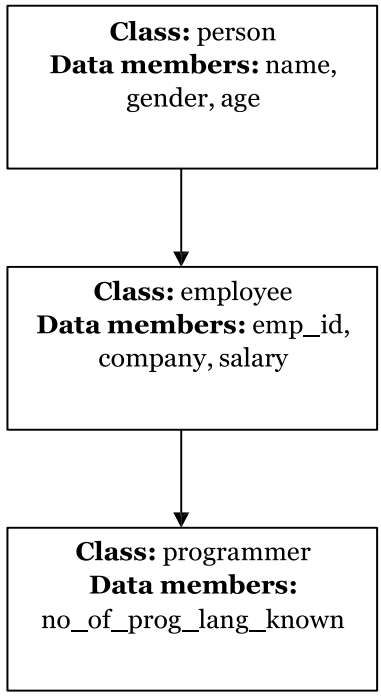
(Space for answers)

XIII Exercise

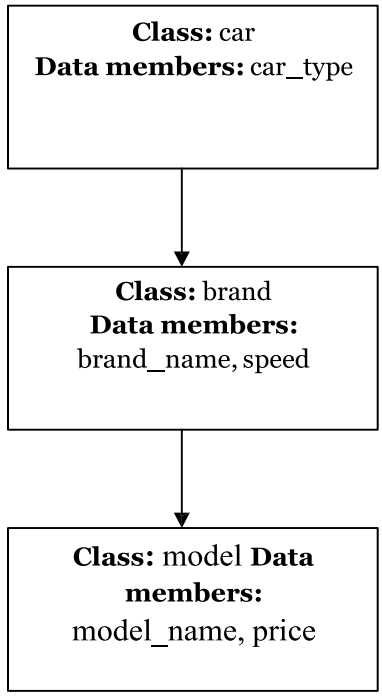
Attempt Q1 and Q.2 from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1 Write a C++ program to implement following Multilevel Inheritance



2 Write a C++ program to implement following Multilevel Inheritance



(Space for answers)

XIV References / Suggestions for further Reading

1. https://www.programtopia.net/cplusplus/docs/single-inheritance/
2. https://www.geeksforgeeks.org/inheritance-in-c/
3. http://www.siteforinfotech.com/2017/05/top-20-mcq-questions-inheritance- in-cpp.html
4. https://www.careerride.com/mcq/inheritance-c-mcq-questions-and-answers-114.aspx

.

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate base class and derived classes declaration using multilevel inheritance	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 13: Program to Implement Concept Multiple Inheritance

I Practical Significance:

Multiple Inheritances is a feature of C++ where a class can inherit from more than one classes.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Apply multiple inheritances to real life problems.
2. Compile the program.
3. Debug and execute the program.

IV Relevant Course Outcome(s)

Implement Inheritance in C++ program.

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using multiple inheritance.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

VII Minimum Theoretical Background

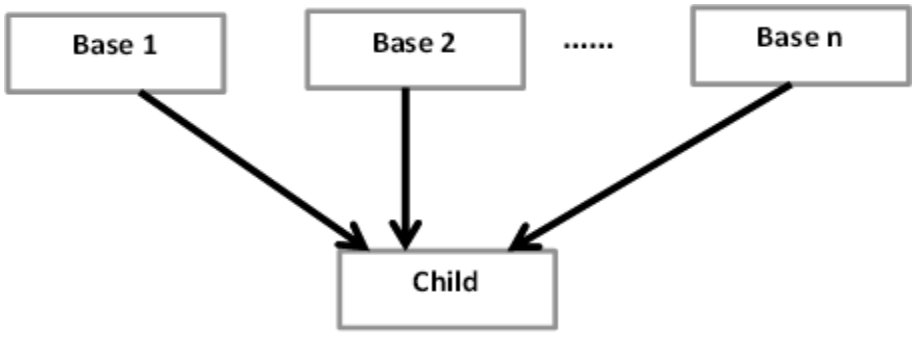
1. **Inheritance** :-It is the process of inheriting properties of objects of one class by objects of another class. The class which inherits the properties of another class is called Derived or Child or Sub class and The class whose properties are inherited is called Base or Parent or Super class.
2. **Multiple Inheritance** :-When a class is derived from two or more base classes, such inheritance is called Multiple Inheritance. It allow us to combine the features of several existing classes into a single class.

Syntax of Multiple Inheritance

```
class base_class1
        {
           properties;
           member functions;
        };

        class base_class2
        {
           properties;
           member functions;
        };
        ... ... ...
        ... ... ...
        class base_classN
        {
           properties;
           member functions;
        };

        class derived_classname : visibility_mode base_class1,
        visibility_mode base_class2,... ,visibility_mode base_classN
        {
           properties;
           member functions;
        };
```

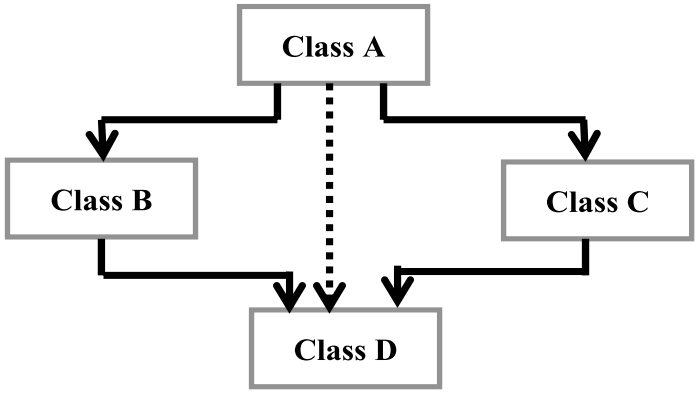


The different access specifier for inheritance concept−

Access	public	protected	Private
Same class	yes	Yes	yes
Derived classes	yes	Yes	no
Outside classes	yes	No	no

Virtual Base Class:

1. Virtual Base class: An ambiguity can arise when several paths exist to a class from the same base class. This means that a child class could have duplicate sets of members inherited from a single base class.
C++ solves this issue by introducing a virtual base class. When a class is made virtual, necessary care is taken so that the duplication is avoided regardless of the number of paths that exist to the child class.
When two or more objects are derived from a common base class, we can prevent multiple copies of the base class being present in an object derived from those objects by declaring the base class as virtual when it is being inherited. Such a base class is known as virtual base class. This can be achieved by preceding the base class's name with the word **virtual**.



Syntax:

```
class derived_class_name: virtual visibility_mode base_class
{
  ----------

  ----------//members of derived class
};
```

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. What is the difference between multiple and multilevel inheritance in C++?
2. State output of the following code:

```
#include <iostream.h>
#include <conio.h>
class liquid
{
    float specific_gravity;
    public:
        void input()
        {
            cout<<"Specific gravity: ";
            cin>>specific_gravity;
        }
        void output()
        {
            cout<<"Specific gravity: "<<specific_gravity<<endl;
        }
};

class fuel
{
    float rate;
    public:
        void input()
        {
            cout<<"Rate(per liter): $";
            cin>>rate;
        }
        void output()
        {
            cout<<"Rate(per liter): $"<<rate<<endl;
```

```
        }
};

class petrol: public liquid, public fuel
{
    public:
        void input()
        {
            liquid::input();
            fuel::input();
        }
        void output()
        {
            liquid::output();
            fuel::output();
        }
};

int main()
{
    petrol p;
    cout<<"Enter data"<<endl;
    p.input();
    cout<<endl<<"Displaying data"<<endl;
    p.output();
    getch();
    return 0;
}
```

3.

```
#include<iostream.h>
#include<conio.h>

class ClassA


{
       public:
       int a;
};

class ClassB : virtual public ClassA
{
       public:
       int b;
};
class ClassC : virtual public ClassA
{
       public:
       int c;
};

class ClassD : public ClassB, public ClassC
{
       public:
       int d;
};

void main()
{

  Class D obj;
  obj.a =10;
  obj.a= 100;

  obj.b = 20;
```

```
        obj.c = 30;
        obj.d= 40;
        cout<< "\n A : "<< obj.a;
        cout<< "\n B : "<< obj.b;
        cout<< "\n C : "<< obj.c;
        cout<< "\n D : "<< obj.d;

}
```

4.Which of the following advantages we lose by using multiple inheritance?

a) Dynamic binding b) Polymorphism

c) Both Dynamic binding & Polymorphism d) None of the mentioned

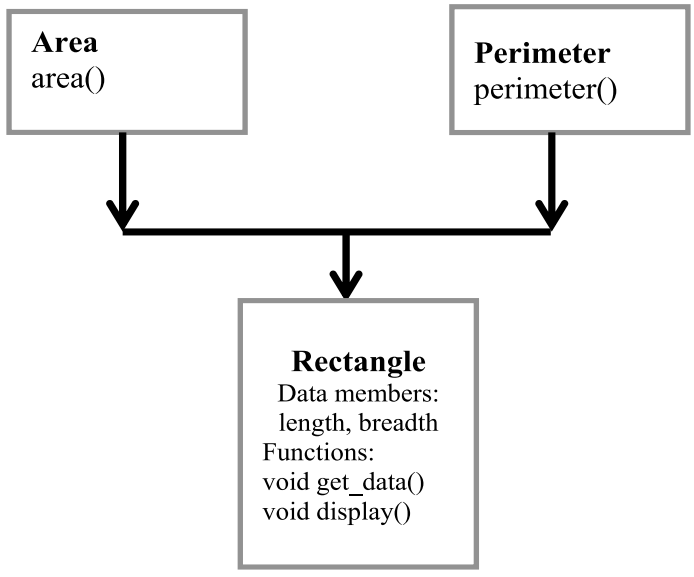
(Space for answers)

XIII Exercise

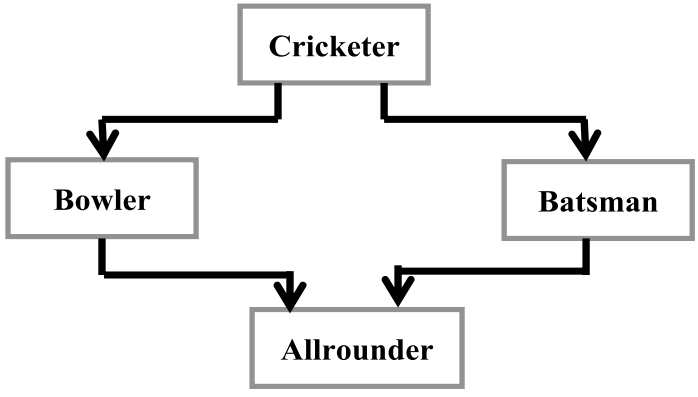
Attempt Q1 or Q2 and Q.3 a or b from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to calculate the area and perimeter of rectangles using concept of inheritance.



2. Write a C++ program for representation of class hierarchy as below .Assume suitable data and function members.



3 Complete the given table:

Program Code	Write & justify Output

a)
`#include <iostream.h>`
`class Base`
`{`
`public:`
`virtual void print() const = 0;`
`};`
`class DerivedOne : public Base`
`{`
`public:`
`void print() const`
`{`
`cout << "DerivedOne\n";`
`}`
`};`
`class DerivedTwo : public Base`
`{`
`public:`
`void print() const`
`{`
`cout << "DerivedTwo\n";`
`}`
`};`
`class Multiple : public DerivedOne, public DerivedTwo`
`{`
`public:`
`void print() const`
`{`
`DerivedTwo :: print();`
`}`
`};`
`int main()`
`{`
`int i;`
`Multiple both;`
`DerivedOne one;`
`DerivedTwo two;`
`Base *array[ 3 ];`
`array[ 0 ] = &both;`
`array[ 1 ] = &one;`
`array[ 2 ] = &two;`
`array[ i ] -> print();`
`return 0;`
`}`	

b)	#include <iostream.h>
	struct a
	{
	int count;
	};
	struct b {
	int* value;
	};
	struct c : public a, public b
	{
	};
	int main()
	{
	c* p = new c;
	p->value = 0;
	cout << "Inherited";
	return 0;
	}	

(Space for answers)

XIV References / Suggestions for further Reading

1. https://www.geeksforgeeks.org/multiple-inheritance-in-c/
2. https://www.programiz.com/cpp-programming/multiple-inheritance

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate definition of multiple inheritance.	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 14: Program to Implement Pointer to Object

I Practical Significance:

The pointer to objects helps to improve the efficiency of the program and access the objects dynamically.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define real life problems using pointer to objects.
2. Compile the program.
3. Debug and execute the program.

IV Relevant Course Outcome(s)

Implement Inheritance in C++ program.
Use Polymorphism in C++ program.

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using pointer to object.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

VII Minimum Theoretical Background

Pointers to objects:-

C++ allows you to have pointers to objects.
The pointers pointing to objects are referred to as Object Pointers.
Syntax:- class-name * object-pointer ;

where class-name is the name of an already defined class and object-pointer is the pointer to an object of this class type.
For example, to declare optr as an object pointer of Sample class type, we shall write Sample *optr ;where Sample is already defined class.
When accessing members of a class using an object pointer, the arrow operator (->) is used instead of dot operator.

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. State output of the following code:

```
#include <iostream.h>
#include<conio.h>
class myclass
{
int i;
public:
        void read(int j)
{
        i= j;
  }
        int getint()
{
                        return i;
```

```
                    }
};
        void main()
    {
         clrscr();
         myclass ob, *objectPointer;
         objectPointer = &ob;
         objectPointer->read(10);
         cout<<objectPointer->getint();
         getch();
    }
```

2. Which is the pointer which denotes the object calling the member function?
   a) Variable pointer b) This pointer
   c) Null pointer d) Zero pointer

3. A pointer can be initialized with _____.
   a) null b) zero
   c) Address of an object of same type d) All of them

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIII Exercise:

Attempt Q1 or Q2 or Q3 and Q.4 a or b from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1 Write a C++ program to declare a class "Book" containing data members book_name, auther_name, and price .Accept this information for one object of the class using pointer to that object.

2 Write a C++ program to declare a class "Box" having data members height, width and breadth. Accept this information for one object using pointer to that object .Display the area and volume of that object.

3 Write a C++ program to declare a class birthday having data members day, month, year. Accept this information for five objects using pointer to the array of objects.

4. Complete the given table:

Program Code	Write & justify Output

a)
#include<iostream.h>
#include<conio.h>
class Time
{
short int hh, mm, ss;
public:
Time()
{
hh = mm = ss = 0;
}
void getdata(int i, int j, int k)
{
hh = i;
mm = j;
ss = k;
}
void prndata(void)
{
cout<<"\nTime is "<<hh<<":"<<mm<<":"<<ss<<"\n";
}
};
void main()
{
clrscr();
Time T1, *tptr;
cout<<"Initializing data members using the object, with values 12, 22, 11\n";
T1.getdata(12,22,11);
cout<<"Printing members using the object ";
T1.prndata();
tptr = &T1;
cout<<"Printing members using the object pointer ";
tptr->prndata();
cout<<"\nInitializing data members using the object pointer, with values 15, 10, 16\n";
tptr->getdata(15, 10, 16);
cout<<"printing members using the object ";
T1.prndata();
cout<<"Printing members using the object
pointer "; tptr->prndata();
getch();
}	

b)
```
#include <iostream>
#include <string>
using namespace std;
class student
{
private:
            int rollno;
            string name;
public:
            student():rollno(0),name("")
            {}
            student(int r, string n):
rollno(r),name (n)
            {}
            void get()
            {
                        cout<<"enter roll no";
                        cin>>rollno;
                        cout<<"enter  name";
                        cin>>name;
            }
            void print()
            {
                        cout<<"roll no is
"<<rollno;
                        cout<<"name is "<<name;
            }
};
void main ()
{
            student *ps=new student;
            (*ps).get();
            (*ps).print();
            delete ps;
}
```

(Space for answers)

.....

.....

.....

.....

.....

XIV References / Suggestions for further Reading

1. https://gradestack.com/Programming-in-C-/Pointers/Pointer-To-Object/
2. https://stackoverflow.com/questions/653423/pointer-to-objects

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate use of pointer to object	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 15: Program to Implement Pointer Derived Class

I Practical Significance:

The key feature of class inheritance is that a pointer to a derived class is type-compatible with a pointer to its base class.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define pointer to derived class.
2. Compile the program.
3. Debug and execute the program.

IV Relevant Course Outcome(s)

Implement Inheritance in C++ program.
Use Polymorphism in C++ program.

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using pointers to derived class.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

VII Minimum Theoretical Background

Pointers to derived class

1. C++ allows base class pointers to point to derived class objects.
2. Let we have –

```
class base { … };
class derived : public base { … };
```

Then we can write –

```
base *p1;
derived
```

d_obj; p1 =
&d_obj;
base *p2 = new derived;

3. Using a base class pointer (pointing to a derived class object) we can access only those members of the derived object **that were inherited from the base**.
   a. It is different from the behavior that Java shows.
   b. We can get Java-like behavior using virtual functions.
4. This is because the **base pointer** has knowledge only of the base class.
5. It knows nothing about the members added by the derived class.

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....
.....
.....
.....
.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. State output of the following code:

```
#include<iostream.h>
class base {
public:
   void show() {
      cout << "base\n";
   }
};
class derived : public base {
public:
   void show() {
      cout << "derived\n";
   }
};
void main() {
   base b1;
   b1.show();
   derived d1;
   d1.show();
   base *pb = &b1;
   pb->show();
   pb = &d1;
  pb->show();
}
```

2. A pointer to the base class can hold address of
   A) only base class object B) only derived class object
   C) base class object as well as derived class object D) None of the above

3. Which variable stores the memory address of another variable?
   A) Reference B) Pointer
   C) Array D) None of the above

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIII Exercise

Attempt Q1 and Q.3 a or b from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program declare a class “polygon” having data members width and height. Derive classes “rectangle” and “triangle” from “polygon” having area() as a member function. Calculate area of triangle and rectangle using pointer to derived class object.

2. Complete the given table:

Program Code	Write & justify Output

a)
#include<iostream.h>
class base
{
public:
int n1;
void show()
{
cout<<"\nn1 = "<<n1;
}
};
class derive : public base
{
public:
int n2;
void show()
{
cout<<"\nn1 = "<<n1;
cout<<"\nn2 = "<<n2;
}
};
int main()
{
base b;
base *bptr;
cout<<"Pointer of base class points to it";
bptr=&b;
bptr->n1=44;
bptr->show();
derive d;
cout<<"\n";
bptr=&d;
bptr->n1=66;
bptr->show();
return 0;
}	

b) #include <iostream.h>
class BaseClass {
int x;
public:
void setx(int i) {
x = i;
}
int getx() {
return x;
}
};
 	
class DerivedClass : public BaseClass {
int y;
public:
void sety(int i) {
y = i;	

```
  }
  int gety() {
     return y;
  }
};

int main()
{
  BaseClass *p; BaseClass
  baseObject; DerivedClass
  derivedObject;


  p = &baseObject;
  p->setx(10);
  cout << "Base object x: " << p->getx() <<
'\n';
  p = &derivedObject;
  p->setx(99);

  derivedObject.sety(88);
  cout << "Derived object x: " << p->getx() <<
'\n';
  cout << "Derived object y: " <<
derivedObject.gety() << '\n';

  return 0;
}
```

(Space for answers)

XIV References / Suggestions for further Reading

1. http://www.learntoprogramming.com/content/pointer-derived-class
2. http://www.java2s.com/Code/Cpp/Class/Demonstratepointertoderived class.htm
3. http://www.siteforinfotech.com/2014/08/solved-mcq-on-cpp-programming-language.html

XV Assessment Scheme

Performance indicators		Weightage
Process related (35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate use of pointer to derived class	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
	TOTAL	**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 16: Program to Implement Operator Overloading Using Unary Operator

I Practical Significance:

The concept of operator overloading helps to assign the new meaning to the existing operator and helps to extend the concept of polymorphism.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define and use the overloaded operator function in the class.
2. Compile the program.
3. Debug and execute the program.

IV Relevant Course Outcome(s)

Use Polymorphism in C++ program.

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using unary operator overloading.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

VII Minimum Theoretical Background

Operator overloading

Operator overloading is a type of polymorphism in which an operator is overloaded to give user defined meaning to it. The main purpose of operator overloading is to perform operation on user defined data type. For eg. The '+' operator can be overloaded to perform addition on various data types. Operator overloading is used by the programmer to make a program clearer. It is an important concept in C++.

Syntax:

```
Return_type classname :: operator OperatorSymbol (Argument_List)
{
        //Statements;
}
```

The operator keyword is used for overloading the operators.

There are a few operators which cannot be overloaded are follows,

1. Scope resolution operator (::)
2. sizeof
3. member selector (.)
4. member pointer selector (*)
5. ternary operator (? :)

There are some restrictions considered while implementing the operator overloading,

1. The number of operands cannot be changed. Unary operator remains unary, binary remains binary etc.
2. Only existing operators can be overloaded.
3. The precedence and associativity of an operator cannot be changed.
4. Cannot redefine the meaning of a procedure.

The Unary Operators

I. The unary operators require only one operand.
II. They perform various operations such as incrementing/decrementing a value by one, negating an expression, or inverting the value of a boolean.

Operator	Description
+	Unary plus operator; indicates positive value (numbers are positive without this, however)
-	Unary minus operator; negates an expression
++	Increment operator; increments a value by 1
--	Decrement operator; decrements a value by 1
!	Logical complement operator; inverts the value of a boolean

.

Unary Operator Overloading Algorithm/Steps:

Step 1: Start the program.
Step 2: Declare the class.
Step 3: Declare the variables and its member function.
Step 4: Using the function getvalue() to get the two numbers.
Step 5: Define the function operator ++ to increment the values
Step 6: Define the function operator - -to decrement the values.
Step 7: Define the display function.
Step 8: Declare the class object.
Step 9: Call the function getvalue()
Step 10: Call the function operator ++() by incrementing the class object and call the function display.
Step 11: Call the function operator - -() by decrementing the class object and call the function display.
Step 12: Stop the program.

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

(1) What is function overloading and operator overloading?

(2) State output of the following code:

```
#include <iostream.h>
class Distance {
   private:
      int feet;
      int inches;
```

```
   public:
      // required constructors
      Distance() {
         feet = 0;
         inches = 0;
      }
      Distance(int f, int i) {
         feet = f;
         inches = i;
      }

      void displayDistance() {
         cout << "F: " << feet << " I:" << inches <<endl;
      }

      // overloaded minus (-) operator
      Distance operator- () {
         feet = -feet;
         inches = -inches;
         return Distance(feet, inches);
      }
};
int main() {
   Distance D1(11, 10), D2(-5, 11);
    -D1;
   D1.displayDistance();
   -D2;
   D2.displayDistance();
   return 0;
 }
```

(3) Write a C++ program using operator overloading for the following:

1. >> : To accept the time.
2. << : To display the time.

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIII Exercise

Attempt Q1 and Q.3 a or b from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to overload unary operators (++) increment and (--)decrement.

2. Complete the given table:

Program Code	Write & justify Output

```
a)
class 3D
{
   int x, y, z;
   public:
       3D (int a=0, int b=0, int c=0)
       {
          x = a;
          y = b;
          z = c;
       }

   3D operator ++()
   {
       x = x + 1;
       y = y + 1;
       z = z + 1;
       return *this;
   }
   3D operator ++(int)
   {
       3D t = *this;
       x = x + 1;
       y = y + 1;
       z = z + 1;
       return t;
   }

   3D show()
   {
       cout<<"The elements are:\n"
       cout<<"x:"<<this→x<<", y:<<this→y <<",
z:"<<this→z;
   }
};
int main()
{
   3D pt1(2,4,5), pt2(7,1,3);
   cout<<"Point one's dimensions before increment
are:"<< pt1.show();
   ++pt1;
   cout<<"Point one's dimensions after increment
are:"<< pt1.show();
   cout<<"Point two's dimensions before increment
are:"<< pt2.show();
   pt2++;
   cout<<"Point two's dimensions after increment
are:"<< pt2.show();
   return 0;
}
```

b)
```
#include <iostream.h>
 class Point
{
private:
    double m_x, m_y, m_z;
 public:
    Point(double x=0.0, double y=0.0, double z=0.0):
        m_x(x), m_y(y), m_z(z)
    {
    }
    Point operator- () const;
    bool operator! () const;
    double getX() { return m_x; }
    double getY() { return m_y; }
    double getZ() { return m_z; }
};
Point Point::operator- () const
{
    return Point(-m_x, -m_y, -m_z);
}
bool Point::operator! () const
{
    return (m_x == 0.0 && m_y == 0.0 && m_z == 0.0);
}
int main()
{
    Point point; // use default constructor to set
to (0.0, 0.0, 0.0)

    if (!point)
        std::cout << "point is set at the
origin.\n";
    else
        std::cout << "point is not set at the
origin.\n";
    return 0;
}
```

(Space for answers)

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

XIV References / Suggestions for further Reading

1. http://www.careerride.com/C++-what-is-overloading-unary-operator.aspx
2. http://www.learncpp.com/cpp-tutorial/95-overloading-unary-operators/
3. http://web.itu.edu.tr/bkurt/Courses/blg252e/blg252e_mod05.pdf
4. https://www.tutorialride.com/cpp-operator-overloading-programs/unary-operator-overloading-c-program.htm
5. https://www.tutorialride.com/cpp-operator-overloading- programs/accept-display-compare-time-with-operator-overloading.htm
6. https://www.tutorialride.com/cpp-operator-overloading-programs/overload-unary-minus-operator-c-program.htm

XV Assessment Scheme

Performance indicators		Weightage
Process related (35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate definition of unary operator overloading function.	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 17: Program to Implement Operator Overloading Using Binary Operator

I Practical Significance:

The concept of operator overloading helps to assign the new meaning to the existing operator and helps to extend the concept of polymorphism.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define real life entity pointer to objects.
2. Compile the program.
3. Debug and execute the program.

IV Relevant Course Outcome(s)

Use Polymorphism in C++ program.

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using binary operator overloading.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

VII Minimum Theoretical Background

The Binary Operators

Binary operator works with two operands.
The first operand becomes the operator overloaded function caller and the second is passed as an argument.

Binary Operator Overloading Algorithm/Steps:

Step 1: Start the program. Step 2:
Declare the class.
Step 3: Declare the variables and its member function.

Step 4: Using the function get value() to get the two numbers.
Step 5: Define the function operator +() to add two complex numbers.
Step 6: Define the function operator –()to subtract two complex numbers. Step 7: Define the display function.
Step 8: Declare the class objects obj1,obj2 and result. Step 9: Call the function get value using obj1 and obj2
Step 10: Calculate the value for the object result by calling the function operator + and operator -.
Step 11: Call the display function using obj1 and obj2 and result. Step 12: Return the values.
Step 13: Stop the program.

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....
.....
.....
.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. What is the difference between unary operator overloading and binary operator overloading in C++?
2. State output of the following code:

```
#include<iostream.h>
#include<conio.h>

class overloading {
    int value;
public:

    void setValue(int temp) {
        value = temp;
    }

    overloading operator+(overloading ob) {
        overloading t;
        t.value = value + ob.value;
        return (t);
    }

    void display() {
        cout << value << endl;
    }
};

void main() {
    overloading obj1, obj2, result;
    int a, b;

    cout << "Enter the value of Complex Numbers a,b:";
    cin >> a>>b;
    obj1.setValue(a);
    obj2.setValue(b);

    result = obj1 + obj2;

    cout << "Input Values:\n";
    obj1.display();
    obj2.display();

    cout << "Result:";
    result.display();

    getch();

}
```

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIII Exercise

Attempt Q1 or Q2 or Q3 and Q.4 a or b from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program to add two complex numbers using operator overloaded by a friend function.
2. Write a C++ program to create a class Binary that contains one float data member. Overload the 4 arithmetic operators'.

3. Write a C++ program to compare two strings using '= =' operator overloading.

4. Complete the given table:

Program Code	**Write & justify Output**

a)
#include<iostream>
#include<cstdlib>
using namespace std;
 	
class Fraction
{
public:
int num, deno;
public:
Fraction()
{
num = 1;
deno = 1;
}
Fraction(int n, int d)
{
num = n;
if (d==0)
{
cout << "Error: Attempting to Divide by Zero" << endl;
exit(0); // it will terminate the program if division by 0 is attempted
}
else
deno = d;
}
Fraction operator +(Fraction f)
{
int n = num*f.deno+f.num*deno;
int d = deno*f.deno;
return
Fraction(n/gcd(n,d),d/gcd(n,d));
}
Fraction operator -(Fraction f)
{
int n = num*f.deno-f.num*deno;
int d = deno*f.deno;
return
Fraction(n/gcd(n,d),d/gcd(n,d));
}
Fraction operator *(Fraction f)
{
int n = num*f.num; int d = deno*f.deno; return
Fraction(n/gcd(n,d),d/gcd(n,d));
}
Fraction operator /(Fraction f)	

```
{
int n = num*f.deno; int d = deno*f.num; return
Fraction(n/gcd(n,d),d/gcd(n,d));
}
bool operator == (Fraction &f)
{
{
int n = num*f.deno; int d = deno*f.num; return
Fraction(n/gcd(n,d),d/gcd(n,d));
}
bool operator == (Fraction &f)
{

(deno==f.deno);
}

return (num==f.num) &&
int gcd(int n, int d)
{
int rem;
while (d != 0)
{
rem = n % d;
n = d;
d = rem;
}
return n;
}
void accept()
{
cout<<"\n Enter Numerator : ";
cin>>num;
cout<<"\n Enter Denominator : ";
cin>>deno;
}
};
void main()
{
Fraction f1; Fraction f2; Fraction f3;

cout<<"\n Enter 1st Fraction Value "; cout<<"\n -------
---------- \n"; f1.accept();

cout<<"\n Enter 2nd Fraction Value "; cout<<"\n ------
---------- \n"; f2.accept();

f3=f1+f2;
cout<<"\n ------------------------------
----";
cout<<"\n Sum of Two Numbers :
"<<f3.num<<"/"<<f3.deno<<endl;
```

```
f3=f1-f2;
cout<<"\n Difference of Two Numbers :
"<<f3.num<<"/"<<f3.deno<<endl;

f3=f1*f2;
cout<<"\n Product of Two Numbers :
"<<f3.num<<"/"<<f3.deno<<endl;
f3=f1/f2;
cout<<"\n Division of Two Numbers :
"<<f3.num<<"/"<<f3.deno;
cout<<"\n ------------------------------
----";

if(f1 == f2)
cout<<"\n Fraction 1 is Equal to
Fraction 2"<<endl;
else
cout<<"\n Fraction 1 is Not Equal to
Fraction 2"<<endl;

}
```

b)

```
#include <iostream.h>
class Box {
double length;
double breadth;
double height;

public:

double getVolume(void) {
return length * breadth * height;
}

void setLength( double len ) {
length = len;
}

void setBreadth( double bre ) {
breadth = bre;
}

void setHeight( double hei ) {
height = hei;
}

objects.
Box operator+(const Box& b) {
Box box;
box.length = this->length + b.length;
```

```
box.breadth = this->breadth + b.breadth; box.height =
this->height + b.height; return box;
}
};

int main() { Box Box1; Box Box2; Box Box3;
double volume = 0.0;

Box1.setLength(6.0); Box1.setBreadth(7.0);
Box1.setHeight(5.0);

Box2.setLength(12.0); Box2.setBreadth(13.0);
Box2.setHeight(10.0);


volume = Box1.getVolume();
cout << "Volume of Box1 : " << volume <<endl;

volume = Box2.getVolume();
cout << "Volume of Box2 : " << volume <<endl;

Box3 = Box1 + Box2;

volume = Box3.getVolume();
cout << "Volume of Box3 : " << volume <<endl;
return 0;
}
```

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIV References / Suggestions for further Reading

1. https://www.tutorialride.com/cpp-operator-overloading-programs/demonstrating-operator-overloading-by-using-friend-function.htm
2. https://www.tutorialride.com/cpp-operator-overloading-programs/overload-arithmetic-insertion-and-extraction-operators.htm
3. https://www.tutorialride.com/cpp-operator-overloading-programs/compare-perform-arithmetic-operations-on-two-fractions.htm
4. http://www.includehelp.com/cpp-programs/cpp-program-to-add-two-objects-using-binary-plus-operator-overloading.aspx

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate definition of binary operator overloading function.	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 18: Program to Implement Functional Overloading

I Practical Significance:

The concept of function overloading helps to use more than one definitions for a function name in the same scope and helps to extend the concept of polymorphism.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Define more than one definitions for a function name in the same scope to implement function overloading.
2. Compile the program.
3. Debug and execute the program.

IV Relevant Course Outcome(s)

Use Polymorphism in C++ program

V Practical Outcome (POs)

Write/ Compile/ debug / Execute simple C++ program using function overloading.

VI Relevant Affective domain related Outcome(s)

1 Select proper programming environment in C++.
2 Follow safety measures
3 Follow ethical practices.

VII Minimum Theoretical Background

Function overloading.

1) Function overloading is a feature in C++ where two or more functions can have the same name but different parameters.
2) The C++ compiler selects the proper function by examining the number, types and order of the arguments in the call. Function overloading is commonly used to create several functions of the same name that perform similar tasks but on different data types.
3) Function overloading can be considered as an example of polymorphism feature in C++ .
4) There are two ways to overload the method in C++
   a) By changing number of arguments or parameters
   b) By changing the data type

Syntax

```
class class_Name
{
returntype method()
{
.....
.....
}
returntype method(datatype1 variable1)
{
.....
.....
}
returntype method(datatype1 variable1, datatype2 variable2)
{
.....
.....
}
};
```

Examples

```
void display();              //function with no arguments
void display( int );         //function with one integer type arguments
void display( float );       //function with one floating point arguments
void display( int, float );  //function with one floating and one integer type argument
```

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Does constructor overloading is implementation of function overloading?
2. State output of the following code:

```
#include <iostream.h>
#include<conio.h>

        int operate (int a, int b)
        {
            return (a * b);
        }
        float operate (float a, float b)
        {
            return (a / b);
        }
        int main()
        {
            int x = 5, y = 2;
            float n = 5.0, m = 2.0;
            cout << operate(x, y) <<"\t";
            cout << operate (n, m);
            return 0;
        }
```

3. Which of the following in Object Oriented Programming is supported by Function overloading and default arguments features of C++?
   a) Inheritance b) Polymorphism
   c) Encapsulation d) None of these

4. Overloaded functions are
   a) Very long functions that can hardly run
   b) One function containing another one or more functions inside it.
   c) Two or more functions with the same name but different number parameters or type.
   d) None of these

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIII Exercise

Attempt Q1 or Q2 or Q3 and Q.4 a or b from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ Program to interchange the values of two int , float and char using function overloading .

2. Write a C++ Program that find the distance between two points in 2D and 3D space using function overloading.

3. Write C++ program to find the area of various geometrical shapes by function overloading.

4. Complete the given table:

Program Code	Write & justify Output

a)
#include<iostream.h>
int absolute(int);
float absolute(float);
 	
int main() {
int a = -5;
float b = 5.5;
cout << "Absolute value of " << a << " = " <<
absolute(a) << endl;
cout << "Absolute value of " << b << " = " <<
absolute(b);
return 0;
}
 	
int absolute(int var) {
if (var < 0)
var = -var;
return var;
}
 	
float absolute(float var){
if (var < 0.0)
var = -var;
return var;
 	
}	

b)

```
#include <iostream.h>
class Test
{
public:
    int main(int s)
    {
        cout << s << "\n";
        return 0;
    }
    int main(char *s)
    {
        cout << s << endl;
        return 0;
    }
    int main(int s ,int m)
    {
        cout << s << " " << m;
        return 0;
    }
};
int main()
{
    Test obj;
    obj.main(3);
    obj.main("I like C++");
    obj.main(9, 6);
    return 0;
}
```

(Space for answers)

XIV References / Suggestions for further Reading

1. https://www.geeksforgeeks.org/can-main-overloaded-
2. http://www.trytoprogram.com/cplusplus-programming/function- overloading/
3. http://www.includehelp.com/cpp-tutorial/function-overloading-cpp-programming-tutorial.aspx
4. https://resume.mcalglobal.com/socialresume/user/take-quiz.htm?catid=ccpp&nameid=dc180db6-4964-4663-b4ae-5abc5bd6bd4b

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate definition of overloaded functions	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

Practical No. 19: Program to Read and Write Data From a File.

I Practical Significance:

The File I/O streams helps to store data permanently on storage and handle it using different file operations.

II Relevant Program Outcomes (POs)

- **Basic knowledge:** Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Computer engineering problem.
- **Discipline knowledge:** Apply Computer engineering discipline - specific knowledge to solve core computer engineering related problems.
- **Experiments and practice:** Plan to perform experiments and practices to use the results to solve broad-based Computer engineering problems.
- **Engineering tools:** Apply relevant Computer technologies and tools with an understanding of the limitations.
- **Communication:** Communicate effectively in oral and written form.

III Competency and Practical skills

This practical is expected to develop the following skills in you :

Develop C++ programs to solve broad-based problems

1. Use Read & Write File Operation.
2. Compile the program.
3. Debug and execute the program.

IV Relevant Course Outcome(s)

Develop C++ programs to perform file operations.

V Practical Outcome (POs)

a) Write/ Compile/ debug / Execute simple C++ program using read and write data from a file.

VI Relevant Affective domain related Outcome(s)

1. Select proper programming environment in C++.
2. Follow safety measures
3. Follow ethical practices.

VII Minimum Theoretical Background

File Handling

1. This concept in C++ language is used for store a data permanently in computer.
2. Using file handling we can store our data in Secondary memory (Hard disk).

Standard File handling Classes

1. **Ofstream:** This file handling class in C++ signifies the output file stream and is applied to create files for writing information to files.
2. **Ifstream:** This file handling class in C++ signifies the input file stream and is applied for reading information from files.
3. **Fstream:** This file handling class in C++ signifies the file stream generally, and has the capabilities for representing both ofstream and ifstream.

Opening a File

1. The first operation generally performed on an object of one of these classes to use a file is the procedure known as to opening a file.
2. An open file is represented within a program by a stream and any input or output task performed on this stream will be applied to the physical file associated with it.
3. The syntax of opening a file in C++ is:

   open (filename, mode);
4. There are some mode flags used for file opening.

   ios::app: append mode
   ios::ate: open a file in this mode for output and read/write controlling to the end of the file
   ios::in: open file in this mode for reading
   ios::out: open file in this mode for writing
   ios::trunk: when any file already exists, its contents will be truncated before file opening

Closing a File

1. When any C++ program terminates, it automatically flushes out all the streams releases all the allocated memory and closes all the opened files.
2. But it is good to use the close() function to close the file related streams and it is a member of ifsream, ofstream and fstream objects.
3. The structure of using this function is:

void close();

General functions used for file handling

1. open(): To create a file
2. close(): To close an existing file
3. get(): to read a single character from the file
4. put(): to write a single character in the file
5. read(): to read data from a file
6. write(): to write data into a file

VIII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards, HDD 40GB and above	As per batch size	For all Experiments
2	Operating system	Windows /LINUX		
3	Software	Turbo C++ Version 3.0 or any other		

IX Precautions

1. Handle computer system and peripherals with care.
2. Follow safety practices.

X Resources used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XI Result (Output of the Program)

.....

.....

.....

XII Practical Related Questions

Note: Below given are few sample questions for reference. Teacher must design more such questions so as to ensure the achievement of identified CO.

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. What are file handling classes in C++?
2. State output of the following code:

```
#include<iostream.h>
#include<fstream.h>
#include<string.h>
#include<cctype.h>

int main()
{

    ifstream ifile;
    ifile.open ("text.txt");
    cout << "Reading data from a file :-" << endl ;
    int c = ifile.peek();
    if ( c == EOF ) return 1;
    if ( isdigit(c) )
    {
    }
    int n;
    ifile >> n;
    cout << "Data in the file: " << n << '\n';
    }
    else
    {
    string str;
    ifile >> str;
    cout << "Data in the file: " << str << '\n';
```

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

3. State output of the following code:

```
#include<iostream.h>
#include<fstream.h>
#include<cctype.h>
int main ()
{
    ifstream ifile;
    ifile.open ("text.txt");
       char last;
    ifile.ignore (256, ' ');
    last = ifile.get();

    cout << "Your initial is " << last << '\n';
    ifile.close();
    return 0;
}
```

4. Which operator is used to insert the data into file?
   a) >> **b) <<**
   c) < d) none of the mentioned

5. Which function is used to position back from the end of file object?
   a) seekg () b) seekp()
   c) both seekg() & seekp() d) none of the mentioned

(Space for answers)

.....

.....

.....

.....

.....

.....

XIII Exercise

Attempt any 2 from (1-4) and Q.5 a or b from the following:

(Note: Use Point VIII to X and XIII to XV for all relevant programming exercise use blank pages provided or attach more pages if needed.)

1. Write a C++ program ask to the user to enter file name to encrypt its content.
2. Write a C++ program to merge two files.
3. Write a C++ program to Read and Display File's Content.
4. Write a C++ program to List Files in Current Directory.
5. Complete the given table:

Program Code	**Write & justify Output**

a) #include<iostream.h>
#include<fstream.h>
int main()
{
ofstream ofile;
ofile.open ("text.txt");
ofile << "geeksforgeeks" << endl;
cout << "Data written to file" << endl;
ofile.close();
return 0;
}	

b)#include <iostream.h>
#include<fstream.h>
int main()
{
char data[100];
ifstream ifile;
 	
//create a text file before executing.
ifile.open ("text.txt");
while (!ifile.eof())
{
ifile.getline (data, 100);
cout << data << endl;
}
ifile.close();
return 0;
}	

(Space for answers)

XIV References / Suggestions for further Reading

1. https://codescracker.com/cpp/program/cpp-program-encrypt-file.htm
2. https://www.geeksforgeeks.org/output-c-programs-set-34-file- handling/
3. https://www.sanfoundry.com/interview-questions-cplusplus-file- string-streams/
4. http://www.includehelp.com/code-snippets/cpp-program-to-write- and-read-time-in-from-binary-file-using-fstream.aspx

XV Assessment Scheme

Performance indicators		Weightage
Process related(35 Marks)		**70%**
1	Logic formation	20%
2	Appropriate class and object definition	20%
3	Debugging ability	20%
4	Follow ethical practices.	10%
Product related (15 Marks)		**30%**
5	Expected Output	10%
6	Timely Submission of report	10%
7	Answer to sample questions	10%
Total (50 Marks)		**100%**

List of Students /Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total(50)**	

List Of Laboratory Manuals Developed by MSBTE

First Semester:

1	Fundamentals of ICT	22001
2	English	22101
3	English Work Book	22101
4	Basic Science (Chemistry)	22102
5	Basic Science (Physics)	22102

Second Semester:

1	Bussiness Communication Using Computers	22009
2	Computer Peripherals & Hardware Maintenace	22013
3	Web Page Design with HTML	22014
4	Applied Science (Chemistry)	22202
5	Applied Science (Physics)	22202
6	Applied Machines	22203
7	Basic Surveying	22205
8	Applied Science (Chemistry)	22211
9	Applied Science (Physics)	22211
10	Fundamental of Electrical Engineering	22212
11	Elements of Electronics	22213
12	Elements of Electrical Engineering	22215
13	Basic Electronics	22216
14	'C' programming Language	22218
15	Basic Electronics	22225
16	Programming in "C"	22226
17	Fundamentals of Chemical Engineering	22231

Third Semester:

1	Applied Multimedia Techniques	22024
2	Advanced Serveying	22301
3	Highway Engineering	22302
4	Mechanics of Structures	22303
5	Building Construction	22304
6	Concrete Technology	22305
7	Strength Of Materials	22306
8	Automobile Engines	22308
9	Automobile Transmission System	22309
10	Mechanical Operations	22313
11	Technology Of Inorganic Chemicals	22314
12	Object Oriented Programming Using C++	22316
13	Data Structure Using 'C'	22317
14	Computer Graphics	22318
15	Database Management System	22319
16	Digital Techniques	22320
17	Principles Of Database	22321
18	Digital Techniques & Microprocessor	22323
19	Electrical Circuits	22324
20	Electrical & Electronic Measurment	22325
21	Fundamental Of Power Electronics	22326
22	Electrical Materials & Wiring Practice	22328
23	Applied Electronics	22329
24	Electrical Circuits & Networks	22330
25	Electronic Measurments & Instrumentation	22333
26	Principles Of Electronics Communication	22334
27	Thermal Engineering	22337
28	Engineering Matrology	22342
29	Mechanical Engineering Materials	22343
30	Theory Of Machines	22344

Fourth Semester:

1	Hydraulics	22401
2	Geo Technical Engineering	22404
3	Chemical Process Instrumentation & Control	22407
4	Fluid Flow Operation	22409
5	Technology Of Organic Chemicals	22410
6	Java Programming	22412
7	GUI Application Development Using VB.net	22034
8	Microprocessor	22415
9	Database Managment	22416
10	Electric Motors And Transformers	22418
11	Industrial Measurements	22420
12	Digital Electronics And Microcontroller Applications	22421
13	Linear Integrated Circuits	22423
14	Microcontroller & Applications	22426
15	Basic Power Electronics	22427
16	Digital Communication Systems	22428
17	Mechanical Engineering Measurments	22443
18	Fluid Mechanics and Machinery	22445
19	Fundamentals Of Mechatronics	22048

Fifth Semester:

1	Design of Steel and RCC Structures	22502
2	Public Health Engineering	22504
3	Heat Transfer Operation	22510
4	Environmental Technology	22511
5	Operating Systems	22516
6	Advanced Java Programming	22517
7	Software Testing	22518
8	Control Systems and PLC's	22531
9	Embedded Systems	22532
10	Mobile and Wireless Communication	22533
11	Industrial Machines	22523
12	Switchgear and Protection	22524
13	Energy Conservation and Audit	22525
14	Power Engineering and Refrigeration	22562
15	Solid Modeling and Additive Manufacturing	22053
16	Guidelines & Assessment Manual for Micro Projects & Industrial Training	22057

Sixth Semester:

1	Solid Modeling	17063
2	Highway Engineering	17602
3	Contracts & Accounts	17603
4	Design of R.C.C. Structures	17604
5	Industrial Fluid Power	17608
6	Design of Machine Elements	17610
7	Automotive Electrical and Electronic Systems	17617
8	Vehicle Systems Maintenance	17618
9	Software Testing	17624
10	Advanced Java Programming	17625
11	Mobile Computing	17632
12	System Programing	17634
13	Testing & Maintenance of Electrical Equipments	17637
14	Power Electronics	17638
15	Illumination Engineering	17639
16	Power System Operation & Control	17643
17	Environmental Technology	17646
18	Mass Transfer Operation	17648
19	Advanced Communication System	17656
20	Mobile Communication	17657
21	Embedded System	17658
22	Process Control System	17663
23	Industrial Automation	17664
24	Industrial Drives	17667
25	Video Engineering	17668
26	Optical Fiber & Mobile Communication	17669
27	Therapeutic Equipment	17671
28	Intensive Care Equipment	17672
29	Medical Imaging Equipment	17673

Pharmacy Lab Manual

First Year:

1	Pharmaceutics - I	0805
2	Pharmaceutical Chemistry - I	0806
3	Pharmacognosy	0807
4	Biochemistry and Clinical Pathology	0808
5	Human Anatomy and Physiology	0809

Second Year:

1	Pharmaceutics - II	0811
2	Pharmaceutical Chemistry - II	0812
3	Pharmacology & Toxicology	0813
4	Hospital and Clinical Pharmacy	0816

HEAD OFFICE

Secretary,
Maharashtra State Board of Technical Education
49, Kherwadi, Bandra (East), Mumbai - 400 051
Maharashtra (INDIA)
Tel: (022)26471255 (5 -lines)
Fax: 022 - 26473980
Email: -secretary@msbte.com
Web -www.msbte.org.in

REGIONAL OFFICES:

MUMBAI

Deputy Secretary (T),
Mumbai Sub-region,
2nd Floor, Govt. Polytechnic Building,
49, Kherwadi, Bandra (East)
Mumbai - 400 051
Phone: 022-26473253 / 54
Fax: 022-26478795
Email: rbtemumbai@msbte.com

PUNE

Deputy Secretary (T),
M.S. Board of Technical Education,
Regional Office,
412-E, Bahirat Patil Chowk,
Shivaji Nagar, Pune
Phone: 020-25656994 / 25660319
Fax: 020-25656994
Email: rbtepn@msbte.com

NAGPUR

Deputy Secretary (T),
M.S. Board of Technical Education
Regional Office,
Mangalwari Bazar, Sadar, Nagpur - 440 001
Phone: 0712-2564836 / 2562223
Fax: 0712-2560350
Email: rbteng@msbte.com

AURANGABAD

Deputy Secretary (T),
M.S. Board of Technical Education,
Regional Office,
Osmanpura, Aurangabad -431 001.
Phone: 0240-2334025 / 2331273
Fax: 0240-2349669
Email: rbteau@msbte.com