|  |  |
| --- | --- |
| **SESSION** | **SEPT 2024** |
| **PROGRAM** | **BCA** |
| **SEMESTER** | **II** |
| **COURSE CODE & NAME** | **DCA1203 OBJECT ORIENTED PROGRAMMING – C++** |
|  |  |
|  |  |

**Set-I**

**1. Explain the concept of operator overloading in C++. Why is it important, and how does it enhance the capabilities of the language?**

**Ans 1.**

**Operator Overloading in C++: An Essential Concept**

Operator overloading is a powerful feature in C++ that allows developers to redefine the behavior of standard operators (such as +, -, \*, =, and <<) for user-defined data types. It provides a way to enable operators to work intuitively with objects, enhancing both the readability and functionality of code.

**Understanding Operator Overloading**

Operators in C++ have predefined meanings for fundamental data types. For instance, the + operator can add two integers or concatenate two strings. However, for user-defined types like classes and structs, the compiler does not inherently understand how to apply these operators.

Its Half solved only

Buy Complete assignment from us

**Price – 190/ assignment**

**MUJ Manipal University Complete SolvedAssignments session JULY-AUG 2024**

buy cheap assignment help online from us easily

we are here to help you with the best and cheap help

**Contact No – 8791514139 (WhatsApp)**

**OR**

**Mail us- bestassignment247@gmail.com**

**Our website -** [**www.assignmentsupport.in**](http://www.assignmentsupport.in)

**2. What is the purpose of access specifiers in a class? Name the three access specifiers available in C++.**

**Ans 2.**

**Access Specifiers in C++: Purpose and Types**

Access specifiers in C++ play a fundamental role in object-oriented programming by controlling the accessibility and visibility of members (data members and member functions) within a class. They define how and where the members of a class can be accessed or modified, ensuring a proper encapsulation mechanism. Encapsulation, one of the core principles of object-oriented programming, aims to hide the internal details of a class while exposing only what is necessary to the outside world. This is achieved effectively through the use of access specifiers.

**Purpose of Access Specifiers**

**Data Security and Encapsulation:** Access specifiers protect sensitive data from unauthorized

**3. What do you understand by constant, keyword and identifier?**

**Ans 3.**

**Understanding Constants, Keywords, and Identifiers in C++**

In C++ programming, constants, keywords, and identifiers are fundamental elements that form the building blocks of the language. They each have distinct roles and purposes, contributing to the structure, readability, and functionality of a program. Understanding these elements is essential for writing efficient and error-free code.

**Constants**

A constant in C++ refers to a fixed value that does not change during the execution of a program. Constants are used to define values that remain the same throughout the program, such as

**Set-II**

**4. Explain with the help of an example the method to implement a class template.**

**Ans 4.**

**Class Templates in C++: Explanation and Example**

Class templates in C++ are a powerful feature of the language that allows developers to create generic classes. Instead of writing multiple versions of a class to handle different data types, a single class template can be written and instantiated for any data type. This enhances code reusability and reduces redundancy, making it particularly useful in scenarios where the same logic applies to different types of data.

**Understanding Class Templates**

A class template is defined using the template keyword, followed by a parameter list enclosed in

**5. List and explain the STL components.**

**Ans 5.**

**Components of the Standard Template Library (STL) in C++**

The Standard Template Library (STL) is a powerful library in C++ that provides generic classes and functions to manage data structures and algorithms. STL is designed to promote code reusability and efficiency by offering pre-built implementations of commonly used programming constructs. It is broadly categorized into three main components: **containers, iterators, and algorithms**.

**Containers**

Containers are objects that store collections of data. STL provides a variety of container types,

**6. Describe the different types of inheritance in C++.**

**Ans 6.**

**Types of Inheritance in C++**

Inheritance in C++ is a fundamental concept of object-oriented programming that allows a class (derived class) to inherit properties and behaviors from another class (base class). This mechanism promotes code reusability, scalability, and modularity. C++ supports several types of inheritance, each suited to specific design scenarios.

**Single Inheritance**

In single inheritance, a class inherits from one base class. This is the simplest form of inheritance