****

| **L** | **T** | **P/S** | **SW/FW** | **TOTAL CREDIT UNITS** |
| --- | --- | --- | --- | --- |
| 3 | 0 | 2 | 0 | 4 |

**Course Title: Object Oriented Programming Using C++**

**Credit Units:**

**Course Level: UG**

**Course Code: ES203**

**Course Objectives:**

1. Get introduction of object-oriented programming.
2. Explore and implement the various features of OOP such as inheritance, polymorphism, Exceptional handling etc. using programming language C++. To easily identify the basic difference between the programming approaches like procedural and object oriented

**Pre-requisites:** Knowledge of Programming logic and techniques

**Course Contents/Syllabus:**

|  | **Weightage (%)** |
| --- | --- |
| **Module I Introduction** | **10** |
| **Descriptors/Topics**   * Difference between C and C++ * Procedure Oriented and Object-Oriented Approach * Basic Concepts: Objects, classes, Principals like Abstraction, Encapsulation, Inheritance and Polymorphism. Dynamic Binding, Message Passing * Characteristics of Object-Oriented Languages |
| **Module II Classes & Objects** | **25** |
| **Descriptors/Topics**   * Abstract data types * Object & classes, attributes, methods * C++ class declaration * Local Class and Global Class * State identity and behaviour of an object * Local Object and Global Object * Scope resolution operator * Friend Functions * Inline functions * Constructors and destructors, instantiation of objects * Types of Constructors * Static Class Data * Array of Objects * Constant member functions and Objects * Memory management Operators |
| **Module III Inheritance** | **20** |
| **Descriptors/Topics**   * Inheritance * Types of Inheritance * access modes – public, private & protected * Abstract Classes * Ambiguity resolution using scope resolution operator and Virtual base class * Aggregation, composition vs classification hierarchies * Overriding inheritance methods, * Constructors in derived classes, * Nesting of Classes. |
| **Module IV Polymorphism** | **20** |
| **Descriptors/Topics**   * Polymorphism, Type of Polymorphism – Compile time and runtime, * Function Overloading, * Operator Overloading (Unary and Binary) * Polymorphism by parameter, * Pointer to objects, * this pointer, * Virtual Functions, * Pure virtual functions. |
| **Module V Strings, Files and Exception Handling** | **25** |
| **Descriptors/Topics**   * Manipulating strings, * Streams and files handling, * Formatted and Unformatted Input output. * Exception handling * Generic Programming – function template, class Template * Standard Template Library: Standard Template Library, Overview of Standard Template Library, * Containers, Algorithms, Iterators, Other STL Elements, the Container Classes. |

**Course Learning Outcomes:**

After completing the Course, students will learn:

• Articulate the principles of object-oriented problem solving and programming.

• Outline the essential features and elements of the C++ programming language.

• Explain programming fundamentals, including statement and control flow and recursion.

• Apply the concepts of data abstraction, function abstraction, inheritance, overriding, overloading, and polymorphism.

• Apply the concepts using objects and data abstraction, class, and methods in function abstraction.

• Analyze, write, debug, and test basic C++ codes using the approaches introduced in the course.

• Analyze problems and implement simple C++ applications using an object-oriented software engineering approach.

**Pedagogy for Course Delivery:**

The class will be taught using remote teaching methodology. Students’ learning and assessment will be on the basis of four quadrants and flipped class method. E-content will be also provided to the students for better learning. The class will be taught using theory, practical and case-based method.

**Lab/ Practical’s details, if applicable:**

* Creation of objects in programs and solving problems through them
* Different use of private, public member variables and functions and friend functions.
* Use of constructors and destructors.
* Operator overloading
* Use of inheritance in and accessing objects of different derived classes.
* Polymorphism and virtual functions (using pointers).
* File handling​

**Assessment/ Examination Scheme:**

| **Theory L/T (%)** | **Lab/Practical/Studio (%)** | **Total** |
| --- | --- | --- |
| 75% | 25% | 100% |

**Theory**

| **Components (Drop down)** | Mid Term Exam | Home Assignment | Attendance | **Quiz** | **End Term Examination** |
| --- | --- | --- | --- | --- | --- |
| **Weightage (%)** | 15% | 10% | 5% | 10% | 60% |

**Lab/ Practical/ Studio Assessment:**

|  |  | **Continuous Assessment/Internal Assessment** | | | | | **End Term Examination** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Components (Drop down** | **Lab record** | | **Performance** | **Viva** | **Open Ended Experiment** | **Attendance** | 60 |
| **Weightage (%)** | 10 | | 10 | 5 | 10 | 5 |

**Text:**

1. A.R. Venugopal, Rajkumar, T. Ravishanker “Mastering C++”, TMH, 1997
2. R. Lafore, “Object Oriented Programming using C++”, BPB Publications, 2004.
3. “Object Oriented Programming with C++” By E. Balagurusamy.
4. Schildt Herbert, “C++: The Complete Reference”, Wiley DreamTech, 2005

**LIST OF PRACTICALS**

1. Write a program to check number palindrome and **string** palindrome.
2. Write a program to show the effect of **call by value** and **call by reference** in functions.
3. Write a program to perform following operations on **matrix** using functions and **switch case**:

(a) Addition (b) subtraction (c) multiplication (d) transpose

1. Define a **class** Shape whose attributes are radius, length and width calculate the perimeter of the rectangle and circle. Use **constructors and destructors**.
2. Create a class Person which includes: character array name of size 64, age in numeric, character array address of size 64, and total salary in real numbers (divide salary in different components, if required). Make an **array of objects** of class Person of size 10.
   1. Write **inline** function that obtains the youngest and eldest age of a person from a class person using arrays.
   2. Write a program to develop the salary slip and display result by using constructors.
3. Create a class called Complex for performing following operations:
   1. **Overload** increment and decrement operators for increasing and decreasing complex number values (Unary operator overload).
   2. **Overload** ‘+’ op and ‘-‘op for complex numbers (Binary operator overloading).
4. Write a program to find the area (function name AREA) of circle, rectangle and triangle by **Function overloading** concept.
5. Design three classes: Student, Exam and Result. The student class has data members such as roll no, name etc. Create a class Exam by inheriting the Student class. The Exam class adds data members representing the marks scored in six subjects. Derive the Result from class Exam and it has its own members such as total marks. Write an interactive program to model this relationship. What type of **inheritance** this model belongs to?
6. Write a program to swap two numbers (create two classes) by using **Friend function.**
7. Consider an example of book shop which sells books and video tapes. These two classes are inherited from base class called media. The media class has command data members such as title and publication. The book class has data members for storing number of pages in a book and tape class has playing time in a tape. Each class will have member functions such as read () and show (). In the base class, these members have to be defined as **virtual functions**. Write a program to models the class hierarchy for book shop and processes objects of these classes using pointers to base class. Write the rules of virtual functions.
8. Write a program to calculate the total mark of a student using the concept of virtual base class.
9. Write a program to show the use of **this pointer.** Write the application of this pointer.
10. Write a program to implement stack functions using **templates**.
11. Write a program to demonstrate **exception handling**
12. Write a program that input a file, which determines its length. Also count the number of word occurrence. For example:” that person is going to town to meet other person”. Here “to” and “person”-2times.