Course Title: Object Oriented Programming with C++ **Credit: 3**
Course No: CSIT.214 **Number of period per week: 3+3**
Nature of the Course: Theory + Lab **Total hours: 45+45**
Year: Second, Semester: Third
Level: B. Sc. CSIT

1. Course Introduction

This course describes basic features of C++ that are different from C programming language. It also covers principles of object oriented programming like polymorphism, class, object, encapsulation, inheritance etc. Besides this, the course describes features like exception handling, templates and File handling using C++.

2. Objectives

At the end of this course the students should be able to:

- Differentiate structured programming from object oriented programming.
- Understood principles of object oriented programming
- Write programs using OOP principles
- Use concepts like exception handling and generics in programming
- Apply C++ in solving scientific problems and simulation

3. Specific Objectives and Contents

Specific Objectives	Contents
• Understand programming language paradigms and History. • Use cin and cout objects along with insertion and extraction operators. • Enable to manage memory dynamically by using New and Delete operators. • Describe reference variables, Scope resolution operator, and Enumerations.	**Unit I: C++ Basics(4)** 1.1. Programming Language Paradigms: Unstructured Programming, Procedural Programming, Modular, Programming, Object Oriented Programming. History of C++. 1.2. Input and Output in C++, Manipulators, Reference variable, Comments, Type Conversion. 1.3. put() and get() Functions, getline() Function. 1.4. New and Delete Operators, Scope Resolution Operators, Enumerations.
• Understand difference between Functions, Macros, and Inline Functions • Use concept of default arguments and method overloading • Enable to pass arguments and	**Unit II: Functions (4)** 2.1. Drawbacks of Functions, Macros, Macro vs Functions, Inline Functions, Macros vs Inline Functions. 2.2. Default Arguments, Overloaded Functions: With Different Number of Arguments, with Different Type of Arguments. 2.3. Passing Arguments to Functions: Pass by Value, Pass

get output from function in different ways.	by Reference, Pass by Pointer 2.4. Returning from Functions: Returning by Value, Return by Reference, Return by Pointer. 2.5. Constant Arguments
• Understand class, object, encapsulation and data hiding. • Explain memory allocation strategy data members and member functions. • Use arrays of objects, pointer objects, and object as argument. • Understand the concept of friend function, friend class and this pointer • Apply the concept of construction and destructors in writing programs.	**Unit III: Class and Objects(10)** 3.1. C++ Structures vs C Structures, Class and Objects, Defining Member Functions, Memory Allocation for Objects and methods. 3.2. Array of Objects, Pointer Objects, Access Specifiers, Passing Objects as Arguments, Returning Objects. 3.3. Static Data Members, Static Methods, Nested Class. 3.4. Friend Functions, Friend Class, This Pointer 3.5. Constructors, Types of Constructors, Constructor Overloading, Copy Initialization, Destructors
• Understand importance and need of operator overloading. • Enable to overload different operators. • Enable to write programs that converts data of one type into another type. • Use nameless temporary objects.	**Unit IV:** Operator Overloading **(7)** 4.1. Introduction, Operators that cannot be overloaded, Rules for Operator Overloading. 4.2. Overloading Unary Operators: Pre-increment operator, Post-increment operator, Negation Operator. 4.3. Overloading Binary Operators: Plus/Minus Operator, Comparison Operators, String Concatenations, Overloading using friend Functions. 4.4. Nameless Temporary Objects 4.5. Type Conversion: Basic to Object, Object to basic, Object to Object.
• Describe need and importance of inheritance • Use inheritance in writing programs • Understand and program different forms of inheritance. • Understand ambiguities in inheritance and handle them. • Use containership and differentiate it from inheritance.	**Unit V: Inheritance & Aggregation (6)** 5.1. Introduction, Benefits, Forms of Inheritance, Protected Access Specifier. 5.2. Public, private, and Protected Derivation. 5.3. Constructor and Inheritance, Destructor and Inheritance 5.4. Method Overriding, Ambiguities in Inheritence: Multiple Inheritance, Multipath Inheritance, Virtual Base Class. 5.5. Containership, Inheritance vs Containership.
• Differentiate static and dynamic polymorphism • Enable to program dynamic polymorphism • Understand importance of pure virtual functions and abstract classes.	**Unit VI: Dynamic Polymorphism(4)** 6.1. Static vs Dynamic Polymorphism, Pointers to base Classes, Virtual Functions 6.2. Implementing Dynamic Polymorphism, Pure Virtual Functions. Abstract Classes 6.3. Virtual Destructors

• Understand exceptions and differentiate it from errors. • Enable to catch and handle exception in programs. • Program own exceptions	**Unit VII: Exception Handling (3)** 7.1. Exception vs Error, Exception Handling mechanism. 7.2. Throw Statement, Try and Catch Statements, Multiple Catch Statements, Catching All Exceptions. 7.3. Nested try-catch, User Defined Exception
• Describe importance of generic programming • Use function and class templates • Understand template specialization and program it..	**Unit VIII: Generic Programming (3)** 8.1. Introduction and Concept, Function Templates, Class Templates. 8.2. Template Specialization. Rules for Using templates.
• Understand concept of streams. • Enable to read/write text and binary files • Use random file access in file handling	**Unit IX: Input/output with Files (4)** 9.1. Streams, Opening and Closing Files, Reading and Writing Text Files. 9.2. Detecting End of File, Reading and Writing Binary Files, Random File Access.

Evaluation System

<table>
<tr><th colspan="8">Undergraduate Programs</th></tr>
<tr><th>External Evaluation</th><th>Marks</th><th>Internal Evaluation</th><th>Weightage</th><th>Marks</th><th>Practical</th><th>Weightage</th><th>Mark</th></tr>
<tr><td>End semester examination</td><td rowspan="4">60</td><td>Assignments</td><td>20%</td><td rowspan="4">20</td><td>Practical Report copy</td><td>25%</td><td rowspan="4">20</td></tr>
<tr><td>(Details are given in the separate table at the end)</td><td>Quizzes</td><td>10%</td><td>Viva</td><td>25%</td></tr>
<tr><td></td><td>Attendance</td><td>20%</td><td>Practical Exam</td><td>50%</td></tr>
<tr><td></td><td>Internal Exams</td><td>50%</td><td></td><td></td></tr>
<tr><td>Total External</td><td>60</td><td>Total Internal</td><td>100%</td><td>20</td><td></td><td>100%</td><td>20</td></tr>
<tr><td colspan="8">Full Marks 60+20+20 = 100</td></tr>
</table>

External evaluation

1. **End semester examination:**
   It is a written examination at the end of the semester. The questions will be asked covering all the units of the course. The question model, full marks, time and others will be as per the following grid.

2. **External Practical Evaluation:**
   After completing the end semester theoretical examination, practical examination will be held. External examiner will conduct the practical examination according to the above

mentioned evaluation. There will be an internal examiner to assist the external examiner. Three hours time will be given for the practical examination. In this examination Students must demonstrate the knowledge of the subject matter.

Full Marks: 100, Pass Marks: 45, Time: 3 Hrs

Nature of question	Total questions to be asked	Total questions to be answered	Total marks	Weightage
Group A: multiple choice*	20	20	20×1 = 20	60%
Group B: Short answer type questions	8	6	6×8 = 48	60%
Group C: Long answer type question/long menu driven programs	3	2	2×16 =32	60%
			100	100%

Each student must secure at least 50% marks in internal evaluation in order to appear in the end semester examination. Failed student will not be eligible to appear in the end semester examinations.

Internal evaluation

Assignment: Each student must submit the assignment individually. The stipulated time for submission of the assignment will be seriously taken.

Quizzes: Unannounced and announced quizzes/tests will be taken by the respective subject teachers. Such quizzes/tests will be conducted twice per semester. The students will be evaluated accordingly.

Attendance in class: Students should regularly attend and participate in class discussion. Eighty percent class attendance is mandatory for the students to enable them to appear in the end semester examination. Below 80% attendance in the class will signify NOT QUALIFIED (NQ) to attend the end semester examination.

Presentation: Students will be divided into groups and each group will be provided with a topic for presentation. It will be evaluated individually as well as group-wise. Individual students have to make presentations on the given topics.

Mid-term examination: It is a written examination and the questions will be asked covering all the topics in the session of the course.

Discussion and participation: Students will be evaluated on the basis of their active participation in the classroom discussions.

Instructional Techniques: All topics are discussed with emphasis on real-world application. List of instructional techniques is as follows:

- Lecture and Discussion
- Group work and Individual work
- Assignments
- Presentation by Students
- Quizzes
- Guest Lecture

Students are advised to attend all the classes and complete all the assignments within the specified time period. If a student does not attend the class(es), it is his/her sole responsibility to cover the topic(s) taught during that period. If a student fails to attend a formal exam/quiz/test, there won't be any provision for re-exam.

Laboratory Work

Student should write programs, prepare lab sheet for each of the topics discussed in classes. Minimum 3 lab hour per week in required. Nature of programming problem can be decided by instructor. Lab sheet of around 50 programming problems is recommended.

Prescribed Text

- *Object-Oriented Programming in C++*: Robert Lafore, Sams Publishing, 4th edition, 2002

Reference

- *C++ Programming with Object Oriented Approach*, Arjun Singh Saud, KEC Publication, Kathmandu, First Edition 2012.
- *C++ How To Program,* Paul J. Ditel & Dr. Harvey M. Ditel, Prentice Hall, 9th Edition, 2013