Draft Copy

Pokhara University
Faculty of Science and Technology

Course No.: CMP 162 (3 Credits)	Full Marks: 100
Course Title: Object Oriented Programming in C++ (3-1-3)	Pass Marks: 45
Nature of the Course: Theory and Practical	Total Lectures: 45 hrs
Level: Bachelor	Program: BE (Computer / Software/ Information Technology)

1. Course Description

This course is designed to encompass the concept of Object Oriented Programming to implement its important features such as data abstraction, encapsulation, inheritance, polymorphism, generic programming, exception handling and file handling using the object-oriented programming language called the C++ language.

2. General Objectives

- To familiarize the students with the basic concepts of Object Oriented Programming.
- To acquaint the students with the knowledge of features of C++.
- To develop the skills in students to solve the problems using Object Oriented concepts using C++.

3. Methods of Instruction

Lectures, Discussions, Readings, Practical works, and Project works.

4. Contents in Detail

Specific Objectives	Contents
• Understand the basic concepts of Object Oriented Analysis and Design.	**Unit 1: Object Oriented Concepts (8hrs)** 1. Object Oriented Programming Paradigm 2. A way of viewing World Agent 3. Procedure Oriented vs Object-Oriented Programming 4. Features of Object Oriented Programming: Class and Object, Data Abstraction, Encapsulation, Inheritance, Polymorphism, Message passing 5. Computation as Simulation, Coping with Complexity and Abstraction Mechanisms 6. Object Oriented Analysis and Design: Introduction, Responsibility Driven Design (RDD), Component Responsibility and Collaborator (CRC) Cards, Responsibility Implies Non-Interference, Programming in Small and Programming in Large
• Implement the use of class, object, and method, Data Abstraction,	**Unit 2: Classes and Objects (8hrs)** 1. Introduction to C++: Origin of C++, Basic C++ Program Structure, Console Input/output Streams and Manipulators 2. Structure in C and C++

Encapsulation, message passing, data hiding in C++. • Understand and implement the concept of constructor, destructor, memory allocation and advanced functions in C++.	3. Classes and Objects 4. Array of Objects 5. Class Diagram and Object Diagram 6. Access Specifiers and Visibility Mode 7. State and Behavior, Methods and Responsibilities 8. Implementation of Data Abstraction, Encapsulation, Message Passing and Data Hiding 9. Memory Allocation for Objects 10. Constructor: Default Constructor, Parameterized Constructor, Copy Constructor 11. Constructor Overloading 12. Destructors 13. Dynamic Memory Allocation: new and delete. 14. Dynamic Constructor 15. Functions: Inline function, Default argument, Passing and Returning by Value, Pointer and Reference, Static Data Member and Static Member Function 16. Friend Function and Friend Class
• Understand and implement the concept of software reusability using inheritance in C++.	**Unit 3: Inheritance (8hrs)** 1. Introduction to Inheritance 2. Inheritance Relationship Diagram 3. Inheritance Mode: Public, Private & Protected 4. Types of Inheritance: Single, Multilevel, Hierarchical, Multiple and Hybrid 5. Ambiguity Resolution 6. Multipath Inheritance and Virtual Base Class 7. Constructor and Destructor in Derived Class 8. Subclass, Subtype and Principle of Substitutability 9. Composition and its Implementation 10. Composition Relationship Diagram 11. Software Reusability
• Understand and implement the concept of polymorphism in C++.	**Unit 4: Polymorphism (9 hrs)** 1. Introduction to Polymorphism 2. Types of Polymorphism: Compile Time Polymorphism: Function Overloading, Operator Overloading Runtime Polymorphism: Virtual Function 3. Overloading Unary and Binary Operators 4. Function Overriding 5. this Pointer and Object Pointer 6. Pure Virtual Function, Abstract Class 7. Virtual Destructor 8. Type Conversion: Basic to User-Defined, User-Defined to Basic, User-defined to User-Defined
• Understand and implement the basic concept of generic programming and Standard Template Library (STL) in C++.	**Unit 5: Templates (5 hrs)** 1. Generic Programming 2. Class Template and Function Template 3. Standard Template Library (STL): Container, Algorithm, Iterator

• Understand and implement file stream operations and exception handling in C++.	**Unit 6: Exception Handling and Stream I/O (7 hrs)** 1. Exception Handling: Error and Exception, Exception Handling Mechanism (try, throw, and catch), Multiple Exception Handling 2. File Handling: Stream Class Hierarchy, Opening and Closing a File, Reading and Writing Object

5. Practical Works

Laboratory works of 45 hours per group of maximum 24 students should cover all the concepts of Object Oriented Programming in C++ language studied in the lectures. Students should submit a final project that uses all the constructs and features of Object Oriented Programming language using C++. The evaluation of the practical works should also be based on project work.

6. List of Tutorials

The various tutorial activities that suit this course should cover all the contents of this course to give students a space to engage more actively with the course content in the presence of the instructor. Students should submit tutorials as assignments to the instructor for evaluation. The following tutorial activities of 15 hours per group of maximum 24 students should be conducted to cover the content of this course.

A. Discussion based tutorials (3hrs)
   1. Evolution of Object Oriented Programming Paradigm.
   2. The way of viewing the world in Object Oriented Programming.
   3. The features of Object Oriented Programming.
   4. Object Oriented Analysis and Design.

B. Problem solving base tutorials (8 hrs)
   1. Write a program to illustrate class, access specifiers, objects, encapsulation, data hiding and data abstraction in C++.
   2. Write a C++ program to illustrate the array of objects.
   3. Write a C++ program to illustrate dynamic memory allocation.
   4. Write a C++ program to illustrate the use of static data members and static member functions.
   5. Write a C++ program to illustrate the use of friend function and friend class.
   6. Write a C++ program to illustrate the use of virtual base class.
   7. Write a C++ program to illustrate the overloading of unary and binary operators using and without using friend function.
   8. Write a C++ program to illustrate the use of STL-Vector and its operations.
   9. Write a C++ program to illustrate how to catch all the exceptions.
   10. Write a C++ program to illustrate reading and writing into multiple files.

C. Review and Question/Answer-based Tutorials (4hrs)

   1. Case Study on origin of C++ languages.
   2. Case study on “Responsibility Driven Design- Interactive Intelligent Kitchen Helper”.
   3. Students ask questions within the course content, assignments and review key course content in preparation for tests or exams.

7. Evaluation system and Students' Responsibilities

Evaluation System

The internal evaluation of a student may consist of assignments, attendance, internal assessment, lab reports, project works etc. The internal evaluation scheme for this course is as follows:

Internal Evaluation	**Weight**	**Marks**	**External Evaluation**	**Marks**
Theory		30	Semester-End examination	50
Attendance & Class Participation	10%			
Assignments	20%			
Presentations/Quizzes	10%			
Internal Assessment	60%			
Practical		20		
Attendance & Class Participation	10%			
Lab Report/Project Report	20%			
Practical Exam/Project Work	40%			
Viva	30%			
Total Internal		50		
Full Marks: 50 + 50 = 100				

Student Responsibilities

Each student must secure at least 45% marks separately in internal assessment and practical evaluation with 80% attendance in the class in order to appear in the Semester End Examination. Failing to get such a score will be given NOT QUALIFIED (NQ) to appear for the Semester-End Examinations. Students are advised to attend all the classes, formal exam, test, etc. and complete all the assignments within the specified time period. Students are required to complete all the requirements defined for the completion of the course.

Text Books:

1. Lafore, R. (2001). *Object-oriented programming in Turbo C++*. Galgotia publications.
2. Budd, T. (2008). *Introduction to object-oriented programming*. Pearson Education India.
3. Balagurusamy, E., Balagurusamy, E., & Balagurusamy, E. (2008). *Object oriented programming with C++* (Vol. 4). Tata McGraw-Hill.

References:

1. Parsons, D. (2001). *Object-oriented Programming with C++*. Cengage Learning EMEA.
2. Schildt, H. (2003). *C++: The complete reference*. McGraw-Hill.