Course Title: Object Oriented Programming with C++ Program: **BICTE**

Course No. : ICT. Ed. 426 | Nature of course: Theoretical + Practical

Level: Bachelor | Credit Hour: 3 hours (2T+1P)

Semester: Second | Teaching Hour: 80hours (32+48)

---

1. **Course Description**

The aim of the course is to develop the skill on thinking about computation and problem solving in Object Oriented Paradigms. The course helps the students to discover the basic concepts of object-oriented programming concept such as object, class, inheritance, polymorphism, abstraction and encapsulation and apply in C++. Students are more engaged in laboratory work to exaction of programing experiments rather than theoretical concept.

2. **General Objectives**

Following are the general objective of this course:

- To acquaint the student with fundamentals object oriented paradigms and programming style in C++ programming language.
- To develop the skill on apply object oriented programming concept in programming.
- To enable a student in explore the new software development paradigms.

3. **Course Outlines:**

Specific Objectives	Contents
• Compare procedure and object oriented programming concept • Describe the feature of object oriented programming. • Compare C and C++. • Demonstrate the C++ programming styles.	**Unit 1: Concept of C++ with Object Oriented Programming (6T+4P)** 1.1 Structured vs Object Oriented Programming Paradigm 1.2 Features of Object Oriented Programming 1.3 Comparison on C and C++ 1.4 C++ Program Structure 1.5 Data types, variables and constants 1.6 Insertion and Extraction Operators 1.7 Type conversion 1.8 Dynamic memory allocation : new and delete operator **Practical Works:** • Install the compiler of C++. • Use Insertion and Extraction Operator. • Use structure in C++ to bundle data
• Demonstrate inline function, function overloading, default arguments, reference arguments and returning by reference concept	**Unit 2: Function in C++** **(3Th+3Pr)** 2.1 Inline function 2.2 Overloaded function 2.3 Default arguments 2.4 Reference arguments 2.5 Returning by reference **Practical Works:** • Write program to implement inline function, overloaded function, default arguments, reference arguments and returning by reference

Faculty of Education
Office of the Dean
Balkhu, Kathmandu

• Explain the Object and Class • Define Data member and Member function. • Define inline member function. • Use array in member function and objects. • Define static and friend function. • Explain constructor and destructors.	**Unit 3: Object and Class** **(5T+10P)** 3.1 Concept of Object and Class 3.2 Access controls: private, protected and public 3.3 Data Member and Member Function of class 3.4 Constructor and Destructor 3.5 Default, Parameterized and Copy constructor 3.6 Constructor Overloading 3.7 Array in Class and Array of Objects 3.8 Static data member and Static member function 3.9 Friend function and friend class **Practical Works:** • Create class and objects with data member and member function. • Declare and define member function and data member with visibility. • Create static function • Create friend functions. • Create different types of constructors
• Explore the concept of inheritance • Describe the derived class and access specifier • Apply single, multiple, multilevel inheritance. • Identify function overriding and ambiguity in inheritance with possible solution • Demonstrate execution sequence of constructor and destructor in inheritance • Implement containership	**Unit 4: Inheritance** **(3T+9P)** 4.1 Concept of Inheritance 4.2 Deriving class using access specifiers: private, public and protected 4.3 Types of Inheritance 4.4 Function overriding 4.5 Ambiguity due to multiple inheritance 4.6 Execution sequence of constructor and destructor in inheritance 4.7 Containership **Practical Works:** • Create different types of inheritance. • Write program to illustrate execution sequence of constructor and destructor in inheritance • Create containership • Identify and solve the ambiguity due to multiple inheritance
• Describe Virtual	**Unit 5: Polymorphism, Abstraction and Encapsulation** **(3T+3P)**

function. • Describe the Pure virtual function. • Describe the Abstract class • Demonstrate early binding (Compile time polymorphism) and late binding (runtime polymorphism)	5.1 Introduction to Virtual Function 5.2 Pure Virtual function 5.3 Abstract Class 5.4 Runtime Polymorphism and Compile time Polymorphism 5.5 Implementation of encapsulation **Practical Works:** • Create virtual function. • Create pure virtual function. • Create Abstract class • Write program to compare different types of polymorphism
• Apply Binary operator and unary operator overloading. • Describe data conversion methods.	**Unit 6: Operator Overloading** **(4T+6P)** 6.1 Introduction to Operator Overloading 6.2 Unary Operator Overloading: prefix/ postfix ++, -- 6.3 Relational Operator Overloading: >, >=. <, <=, ==, != 6.4 Arithmetic Operator Overloading: +, -, *, /, % 6.5 Type Conversion i. Basic to user-defined types ii. User-defined to basic types iii. User-defined to user-defined types **Practical Works:** • Overload unary operator, relational operator and arithmetic operators • Apply data conversion techniques to convert different types of data
• Explain concept of template. • Demonstrate namespace creation and use of it in programs • Define function template and class template. • Apply the different exception handling methods.	**Unit 7: Namespace, Template and Exception Handling** **(4T+6P)** 7.1 Namespace Concept, creation and usage in programs 7.2 Concept of Template 7.3 Function Template 7.4 Class Template 7.5 Basic of exception handling 7.6 Exception handling mechanism: throw, catch and try 7.7 Exception handling of basic types and user-defined types **Practical Works:** • Create and apply namespace • Create and apply function template. • Create and apply template class.

	• Apply try, catch and throw methods in program to handle exception for user-defined and basic types.
• Explain concept of files • Demonstrate stream classes • Demonstrate sequential and random access to file	**Unit 8: File Handling** **(3T+3P)** 8.1 Introduction to file stream 8.2 fstream, ofstream and ifstream classes 8.3 Using constructor to open file 8.4 File opening modes in C++ 8.5 Using open function to open file 8.6 Sequential and Random access to file **Practical Works:** • Apply fstream, ifstream and ofstream classes to manipulate files • Access files sequentially and randomly
• Create console application using C++ .	**Unit 9: Project** **(5P)** Develop simple Application using C++.

4. Instructional Techniques

The instructional techniques for this course are divided into two groups. First group consists of general instructional techniques applicable to most of the units. The second group consists of specific instructional techniques applicable to particular units.

4.1 General Techniques

Reading materials will be provided to students in each unit. Lecture, Discussion, use of multi-media projector, brain storming are used in all units.

4.2 Specific Instructional Techniques

Demonstration is an essential instructional technique for all units in this course during teaching learning process. Specifically, demonstration with practical works will be specific instructional technique in this course. The details of suggested instructional techniques are presented below:

Units	Activities
Unit 1: Concept of Object Oriented Programming	• Select and Install the different compiler of C++ such as Code Block, Dev C++ etc. • Demonstrate the programming structure of C++. • Compare the other program provide the assignment for understanding of object oriented paradigms. • Monitoring of students' work by reaching each student and providing feedback for improvement • Presentation by students followed by peers' comments and teacher's feedback
Unit 2, 3:Function,	• Demonstrate inline function, function overloading, default arguments, reference arguments and returning by reference.

Object and Class	• Demonstrate class and object creation methods in C++. • Demonstrate the methods and attributes in Class and access from objects. • Demonstrate the different types of methods such as inline, static and friends. • Lab work in pairs in different tasks assigned by the teacher • Monitoring of students' work by reaching each pair and providing feedback for improvement • Presentation by students followed by peers' comments and teacher's feedback
Unit 4: Inheritance	• Demonstrate the single, multiple and multilevel inheritance and applied into C++. • Lab work in pairs in different tasks assigned by the teacher. • Monitoring of students' work by reaching each student and providing feedback for improvement • Presentation by students followed by peers' comments and teacher's feedback
Unit 5: Virtual Function and Polymorphism	• Demonstrate the virtual and pure virtual functions and application. • Demonstrate the abstract and container class. • Lab work in pairs in different tasks assigned by the teacher. • Monitoring of students' work by reaching each student and providing feedback for improvement • Presentation by students followed by peers' comments and teacher's feedback
Unit 6: Operator Overloading	• Demonstrate the unary and binary operator overloading methods. • Lab work in pairs in different tasks assigned by the teacher • Monitoring of students' work by reaching each student and providing feedback for improvement • Presentation by students followed by peers' comments and teacher's feedback
Unit 7: Namespace, Template and Exception Handling	• Demonstrate Namespace creation and usage • Demonstrate the template function and class. • Demonstrate the exception handling concept in OOPs with reference C++. • Monitoring of students' work by reaching each student and providing feedback for improvement • Presentation by students followed by peers' comments and teacher's feedback
Unit 8: File Handling	• Demonstrate stream classes • Demonstrate file opening and closing • Demonstrate random access and sequential access to file
Unit 9: Project	• Develop console application applied with OOPs Concepts including file handling.

5. Evaluation :

Faculty of Education
Office of the Dean
Balkhu, Kathmandu

Evaluation of students' performance is divided into parts: Internal assessment (theory and practical and internal external examinations (theory and practical). The distribution of points is given below:

Internal Assessment Theory	Internal Assessment Practical	Semester Examination (Theoretical exam)	External Practical Exam/Viva	Total Points
25 Points	15 Points	40 Points	20 Points	100 Points

***Note**: Students must pass separately in internal assessment, external practical exam and semester examination.*

5.1 Internal Assessment (25 Points) of theoretical part

Internal assessment will be conducted by subject teacher based on following criteria:

Attendance and learning Activities	5 points
First assignment (Written assignment)	5 points
Second assignment (Project work with presentation)	10 points
Third assignment/written examination	5 point
Total	25 points

5.2 Internal Assessment (15 Points) of practical part

Internal practical assessment will be conducted by subject teacher based on following criteria:

Attendance and learning Activities	5 points
Practical work/project work/lab work	10 points
Total	15 points

5.3 Semester Examination (60 Points)

Examination Division, Dean office will conduct final examination at the end of semester.

Objective question (Multiple choice questions 10 x 1 point)	10 Points
Subjective Questions (6 questions x 5 marks With "OR" two questions)	30 Points
Total	40 points

5.4 **Practical Exam/Viva (20 Points)**

Examination Division, Office of the Dean will appoint an external examiner (ICT teachers working another campus) for conducting practical examination

Items	Points
Evaluation of Record Book	4
Project work/practical work presentation/skill test	10
Viva	6

Faculty of Education
Office of the Dean
Balkhu, Kathmandu

Total	20

6. Recommended books and References materials (including relevant published articles in national and international journals)

Recommended books:

Balagurusamy, E. (2013). *Object oriented programming with C++*. New Delhi: Tata McGraw-Hill (Unit 1-8).

BaralDayasar&BaralDiwakar(2010), Secrete of Object Orientd Programming in C++, Kathmandu, BhundipuranPrakashan (Unit 1-8).

References materials:

Robert Lafore(2003), Object Oriented Programming in Turbo C++, Galgotia Publications Ltd. India, 2003 (Unit 1-8).

Schildt, H. (2003). *C++: the complete reference* (4th ed). New York: McGraw-Hill.

Lippman, S.B., Lajoie. J., C++ Primer, 3rd Ed., Addison Wesley, 1998