University of Global Village (UGV), Barishal

Content of the Object-Oriented Programming
University Student (UGV) Format

Program: Bachelor of Science in Computer Science Engineering (CSE)

Prepared By: Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barishal

Course Code	CSE-0613-2203
Name of Course Title	Object-Oriented Programming
Course Type	Core Course
Level	4th Semester
Academic Session	Summer 2024
Name(s) of AcademicCourse teacher(s)	Md. Riadul Islam Assistant Professor, CSE. Mobile: 01842611852 E-mail: riadnwu@gmail.com
Consultation Hours:	

Object-Oriented Programming	
Course Code: CSE-0613-2203	**Credits: 03**
Exam Hours: 03	**CIE Marks: 90**
Course for 4th Semester, **Bachelor of Science in Computer Science Engineering (CSE)**	**SEE Marks: 60**

1. Rationale for the inclusion of the program

Rationale for the inclusion of the course/module in the program	The Object-Oriented Programming (OOP) course serves as a foundational exploration into the principles and practices of designing and implementing software using the object-oriented paradigm. It offers students an opportunity to grasp the essential concepts and methodologies crucial for effective software development and maintenance. This course begins by elucidating the core principles of OOP, including fundamental concepts such as classes, objects, inheritance, polymorphism, encapsulation, and abstraction.
Pre-requisite (if any)	Data Structure
Status	One of the core courses of CSE
Credit Value (hours)	3 (3 hours)
Total Marks	150

2. Course Summary:

Number	Section	Topics
1	Introduction to Object-Oriented Programming	- Definition and history of OOP - Basic concepts: classes, objects, inheritance, polymorphism - Advantages and applications of OOP
2	Classes and Objects	- Definition and creation of classes and objects - Constructors and destructors - Access specifiers: public, private, protected
3	Inheritance and Polymorphism	- Concept of inheritance and types: single, multiple, multilevel, hierarchical, hybrid - Method overriding and method overloading - Runtime polymorphism and dynamic binding
4	Encapsulation and Abstraction	- Definition and importance of encapsulation - Abstract classes and interfaces - Data hiding and

		access control mechanisms
5	Constructors and Destructors	- Types of constructors: default, parameterized, copy constructors - Destructor basics and importance - Constructor overloading
6	Operator Overloading	- Concept and need for operator overloading - Overloading unary and binary operators - Overloading insertion and extraction operators
7	Templates and Generic Programming	- Definition and use of templates - Function templates and class templates - Template specialization and instantiation
8	Exception Handling	- Basics of exception handling: try, catch, throw - Standard exceptions and user-defined exceptions - Exception handling best practices
9	File Handling	- File streams and operations - Reading from and writing to files - File modes and error handling in file operations
10	Standard Template Library (STL)	- Introduction to STL - Components of STL: containers, algorithms, iterators - Using STL for common data structures: vectors, lists, maps, sets
11	Design Patterns	- Introduction to design patterns - Types of design patterns: creational, structural, behavioral - Examples of common design patterns: Singleton, Factory, Observer
12	Hands-on Projects and Practical Implementation	- Implementing OOP concepts in programming languages like C++, Java, Python - Hands-on projects to apply OOP principles to real-world problems

3. Course Objectives:

Upon completing this Object-Oriented Programming course, students will be able to:

i. Understand and explain the fundamental concepts and applications of OOP

- Grasp the core principles of OOP, including its history, development, and key components such as classes, objects, inheritance, polymorphism, encapsulation, and abstraction.
- Explore the various applications of OOP in different branches of computer science and engineering, such as software development, game development, and systems programming.
- Discuss case studies and examples where OOP has been successfully implemented to solve complex problems and improve software design and maintenance.

ii. Apply OOP techniques to solve real-world problems

- Learn about different types of OOP techniques, including design patterns and principles like SOLID.
- Gain hands-on experience with popular OOP languages and environments such as C++, Java, and Python.
- Develop skills to analyze problem statements, design appropriate class structures, implement solutions, and evaluate their performance on real-world datasets.
- Work on practical projects that involve applying OOP techniques to address specific problems, such as developing user interfaces, managing data in applications, and creating reusable software components.

iii. Implement advanced OOP concepts in software development

- Understand the role of advanced OOP concepts in efficient software development.
- Learn about advanced OOP features such as multiple inheritance, interfaces, abstract classes, and generics/templates, and their applications.

- Design and implement complex software systems using OOP to enhance software performance, manage resources efficiently, and solve computational problems.
- Develop software solutions that leverage advanced OOP features for better performance and scalability.

iv. Analyze and interpret OOP performance

- Acquire skills in analyzing the performance of OOP designs using techniques such as design patterns, code refactoring, and software metrics.
- Learn methods for extracting meaningful insights from software performance data, including runtime analysis, memory usage, and efficiency trade-offs.
- Develop the ability to interpret the results of performance analyses and communicate findings effectively to both technical and non-technical audiences.
- Apply statistical and analytical methods to solve performance-related problems, such as identifying bottlenecks, optimizing code, and resource allocation.

v. Evaluate the ethical implications and societal impact of OOP

- Explore the ethical considerations surrounding the use of OOP, including issues related to software reliability, maintainability, and security.
- Discuss the potential societal impacts of OOP, such as software accessibility, impact on employment, and digital divide.
- Examine regulatory frameworks and guidelines for the responsible development and deployment of software.
- Reflect on the role of computer scientists and engineers in ensuring that OOP is used ethically and responsibly for the benefit of society.

4. Course Learning Outcome (CLO) at the end of the course, the students will be able to-

No.	**Course Outcomes**
CO1	Understand and apply object-oriented programming principles and techniques.
CO2	Analyze the design, efficiency, and correctness of object-oriented software.
CO3	Design and implement software solutions using OOP to solve real-world problems.
CO4	Apply OOP principles to optimize and enhance software design and performance.

5. Mapping of Course Learning Outcomes to Program Learning Outcomes

PLOs/CLOs	CLO1	CLO2	CLO3	CLO4
PLO1	3	3	2	3
PLO2		3	3	3
PLO3				3
PLO4				3
PLO5				3

6. Topics to be covered/Content of the course

Sl.no	Date	Week no	Class no	Topics	Specific Outcomes	Teaching Learning Strategy(s)	Assessment Strategy(s)	Alignment to CLO
1	1/7- 7/7	1	1,2,3	Introduction to Object Oriented Programming	Overview of structured programming approach, Object oriented programming approach, Characteristics of object oriented languages	Lecture, multimedia, group discussion, interactive sessions	Feedback, Q&A, assessment of LOs	CLO1
2	8/7 - 14/7	2	4,5&6	Basics of C++ programming	C++ Program Structure, Character Set and Tokens, Data Type, Type Conversion, Preprocessor Directives, Namespace, Input/Output Streams and Manipulators.	Lecture, multimedia, practical examples, simulations	Feedback, Q&A, quizzes	CLO2

3	15/7 - 21/7	3	7,8,&9	Memory Allocation Functions	Dynamic Memory Allocation with new and delete, Control Statements. Functions: Function Overloading, Inline Functions, Default Argument, Pass by Reference, Return by Reference, Scope and Storage Class	Lecture, multimedia, hands-on practice, case studies	Midterm Quiz #1, assessment of LOs	CLO2
4	22/7 - 28/7	4	10,11& 12	Pointers	Pointers: Pointer variables declaration & initialization, Operators in pointers, Pointers and Arrays, Pointer and Function.	Lecture, multimedia, interactive sessions, group work	Feedback, Q&A, assessment of LOs	CLO2
5	29/7 - 4/8	5	13,14& 15	Classes	A Simple Class and Object, Accessing members of class,	Lecture, multimedia, problem-solving sessions, simulations	Midterm Case Study #1, Home Assignment #1	CLO3
6	5/8 - 11/8	6	16,17& 18	Objects	Initialization of class objects:	Lecture, multimedia, interactive sessions, hands-on practice	Feedback, Q&A, quizzes, group discussions	CLO4
7	12/8 - 18/8	7	19,20& 21	Polymorphism, and miscellaneous	Virtual base class, Friend function and Static function, Assignment and copy initialization, Copy constructor, This pointer, Concrete classes, Polymorphism and its roles.	Lecture, multimedia, practical examples, simulations	Feedback, Q&A, group discussions, assessment of LOs	CLO3

8	19/8 - 25/8	8	22,23&24	Function Templates and Exception Handling	Function templates, Function templates with multiple arguments, Class templates, templates and inheritance, Exceptional Handling (Try, throw and catch), Use of exceptional handling.	Lecture, multimedia, practical examples, group work	Feedback, Q&A, quizzes, assessment of LOs	CLO4
9	27/8 – 13/9	9 & 10		Mid-Semester Examination	Mid-Semester Examination	Mid-Semester Examination	Mid-Semester Examination	
10	14/9 - 20/9	11	25,26&27	Constructor and Destructor	(Constructor, Destructor), Default Constructor, Parameterized Constructor, Copy Constructor,	Lecture, multimedia, interactive sessions, hands-on practice	Feedback, Q&A, group discussions, quizzes	CLO3
11	21/9 - 27/9	12	28,29&30	Inheritance	Introduction to inheritance, Derived Class and Base Class, Access Specifiers (private, protected and public) Types of inheritance,	Lecture, multimedia, problem-solving sessions, simulations	Midterm Case Study #2, Home Assignment #2	CLO3
12	28/9 - 4/9	13	31,32&33	Inheritance Continued	Public and Private Inheritance, Constructor and Destructor in derived classes, Aggregation	Lecture, multimedia, interactive sessions, case studies	Feedback, Q&A, group discussions, quizzes	CLO2
13	5/9 -11/9	14	34,35&36	Generalization and Specialization	Generalization: Define generalization as a relationship between classes. Specialization: Discuss specialization and how it relates to generalization.	Lecture, multimedia, interactive sessions, guest lectures	Feedback, Q&A, assessment of LOs	CLO3

14	12/9 - 18/9	15	37,38&39	Association, Composition, and Aggregation	Association: Explain the association between classes and its types. Composition: Define composition and its characteristics. Aggregation: Discuss aggregation and how it differs from composition.	Lecture, multimedia, project presentations, interactive discussions	Feedback, Q&A, project assessments	All CLOs
15	19/9 - 25/9	16	40,41& 42	Course Review and Project Discussions Final Project Presentations	- Reviewing key concepts and challenges faced throughout the course. - Discussing project ideas and approaches with classmates and instructor. - Public presentation of a completed algorithm design project. - Demonstration of problem-solving skills and learned techniques.	Project presentations, group discussions, peer evaluations	Final Project Evaluations, feedback	All CLOs
16	1/10-20/10	17 & 18		Final-Semester Examination	Final-Semester Examination	Final-Semester Examination	Final-Semester Examination	

4. ASSESSMENT PATTERN

Quizzes: Altogether 4 quizzes may be taken during the semester, 2 quizzes will be taken for midterm and 2 quizzes will be taken for final term. Out of these 2 quizzes for each term, average of best 2 quizzes will be counted. No makeup quizzes will be taken. Students are stronglyrecommended not to miss any quizzes

Assignments: Altogether 4 assignments may be taken during the semester, 2 assignments will be taken for midterm and 2 assignments will be taken for final term. Out of these 2 assignments for each term, average of best 2 assignments will be counted. The students will be given assignment during the class which they have to prepare at home and will submit on or before the duedate. No late submission of assignment will be accepted

Presentation: The students will have to form a group of maximum 4 members. The topic of the presentation will be given to each group andstudents will have to do the group presentation on the given topic.

Classroom Participation: All the students are encouraged to come to class prepared, take part in the classroom discussion, make thoughtfulcontribution, and participate actively in all classroom activities

7. CIE- Continuous Internal Evaluation (90 Marks)

Bloom's Category Marks (out of 90)	**Tests (45)**	**Assignments (15)**	**Quizzes (15)**	**Attenda nce (15)**
Remember	**5**	**03**		
Understand	**5**	**04**	**05**	
Apply	**15**	**05**	**05**	
Analyze	**10**			
Evaluate	**5**	**03**	**05**	
Create	5			

8. SEE- Semester End Examination (60 Marks)

Bloom's Category	**Test**
Remember	7
Understand	7
Apply	20
Analyze	15
Evaluate	6
Create	5

9. Textbooks and other resources

Category	Details
Conducted Textbook	• Herbert Schildt, C++ The Complete Reference, FourthEdition, Tata McGraw Hill Publication.
Reference Books	• Robert Lafore, Object Oriented Programming in C++, Fourth Edition, SAMS publications.
Other Resources:	• Online Tutorials, YouTube, Simulation video, Animation video, etc. • https://drive.google.com/drive/folders/1L_PsvRwotmMnFWewQkFn4Devk-J-78x_?usp=sharing

Prepared By: Md. Riadul Islam
Assistant Professor, Dept. Of CSE
University Of Global Village (UGV), Barishal