University of Gujrat

Faculty of CS & IT

Department of Computer Sconce

Title	**Object Oriented Programming** BS MSc BS. LAB
Code	**CS – 106** CS-103(3+1)=4, CS106(3), CS107(1)
Credit hours	3+1
Prerequisite	CS102-Programming Fundamentals
Course Description	This course provides in-depth coverage of object-oriented programming principles and techniques using C++. Topics include classes, overloading, data abstraction, information hiding, encapsulation, inheritance, polymorphism, file processing, templates, exceptions, container classes. The course briefly covers the mapping of UML design to C++ implementation and object-oriented considerations for software design and reuse. The course has a strong practical emphasis, and students will be required to implement OO concepts in C++ during supervised laboratory sessions and in unsupervised assignment work. In general, each class will consist of a one and a half hour lecture, and a one and a half hour laboratory session, which will be held weekly.
Course Goal:	This course provides in-depth coverage of OOP using the C++ programming language. At the completion of this course the student should be comfortable coding a program using C++. He or she will know the strengths and weaknesses of the language. Students will then be exposed to OO analysis and design. C++ syntax and its idioms will be covered, with particular emphasis on how to program in C++ with an OO mindset.
Course objective	• By the end of the course, students should be able to: 1. create class hierarchies using OOP design 2. understand and apply inheritance techniques to their programs 3. overload and override methods and understand the difference 4. create modular programs using accepted structured programming • Create and use UML diagrams • Understand the strengths and weaknesses of OOP programming. • Use files, both binary and text. • Multifile Programming
Evaluation System	a) The teacher is responsible for the evaluation of work of the students of his/her class and for the grades on the basis of such evaluation. b) The number and nature of tests and assignments depends on the nature of the course. However, there will be at least two tests, mid semester and final examination in addition to class work. c) Each course will follow the weight age as under: Mid term 25% Sessional work 25%(Quiz=8, Assignments=7, Project=10) Final term 50% d) To pass a course, student must obtain 'D' grade (50% marks) with at least 20% marks separately in (i) mid term + sessional work and (ii) final term. e) The final term examination will cover the entire course.
Grading System	

Marks in Percentage	Letter Grade	Numeric Value of Grade	Description
85 and above	A+	4.00	Exceptional
80-84	A	3.70	Outstanding
75-79	B+	3.40	Excellent
70-74	B	3.00	Very Good
65-69	B-	2.50	Good
60-64	C+	2.00	Average
55-59	C	1.50	Satisfactory
50-54	D	1.00	Pass
49 and below	F	0.0	Fail
	W		Withdrawal
	I		Incomplete

QUALIFYING ATTENDANCE	You must attend every class for your own personal benefit. Please refer to university policy of minimum attendance requirement. Failing to conform qualifying attendance threshold, the student will stand debarred from sitting in the examination and assigned with "F" Grade.
Academic and Moral Integrity:	1. All assignments should be your own work (or your group's when approved). PLAGIARISM will be awarded with "F" grade and/or reported to the University for academic and moral misconduct. To protect yourself, ALWAYS PROVIDE REFERENCES! 2. Missed quizzes/presentations/assignments will not be rescheduled. 3. Late/Copied assignments shall not be accepted and will result in deduction of marks already scored.

Instructions / Suggestions for STUDENTS for satisfactory progress in this course:

- ✓ On average, most students find at least three hours outside of class for each class hour necessary for satisfactory learning.
- ✓ Chapters should be read and homework should be attempted before class.
- ✓ You may contact me through email on email-id: to you within 24 hours.
- ✓ The homework assigned is a minimum. You should always work extra hours on your own.
- ✓ Use the few minutes you usually have before the start of each class to review the prior meetings' notes and homework. This will save us valuable in-class time to work on new material.
- ✓ Develop a learning habit rather than memorizing; work in groups, whenever appropriate.
- ✓ Apply the learned principles and gained knowledge; be creative in thinking.
- ✓ **Assignments/ Activities:** They are not meant simply for grades, but to reinforce your learning. Assignments are due on time. Each day late will lower your assignment grade by 10%. Apart from value of content, spelling, grammar, punctuation, and good presentation (printing and paper quality) will figure into your assignment grade. To guard against errors, please keep copies of the papers you turn in and retain all graded assignments for your reference.
- ✓ Your Assignments must include all the References. For this course you are highly encouraged to follow the Harvard style of referencing (if you use anything outside of the recommended/referred material)

Recommended Books	Object Oriented Programming in C++ by Robert Lafore 4th Edition
Reference Books & Materials:	C++ Programming: From Problem Analysis to Program Design by D.S. Malik, 5th Edition C++ How to Program by Deitel & Deitel, (www.deitel.com) Let us C++ by Yashavant Kanetkar

COURSE OUTLINE

Week	Lecture	Topic	Assessment
1	1	**The Big Picture** • Course Intro, Class policies, learning from previous semester failures • Beginning of programming • Structured programming • Why Do We Need Object-Oriented Programming? • Object oriented programming • Difference between procedural and OOP • Sturcutre basics	Assignment 1
	2	**Structure** • Structure within structure • Structures and classes • Nesting of structs arrays, structs of array • Enumerations, • Software Engineering Case Study: examining the ATM Requirements Document	
2	3	• Characteristics of Object-Oriented Languages o Objects o Classes o Inheritance o Reusability o Data Abstraction o Data Encapsulation o Creating new data types o Polymorphism and overloading • Software Engineering Case Study: introduction to Object Technology and the UML	Quiz 1
	4	**Objects and classes** o Basics of class(struct vs. classes) and objects with real world example o Basics of class and objects with programming example o Data member and member function o Access specifier	Project Mile Stone – I
3	5	• C++ objects as data types • Constructors • Destructors • Constructor Overloading • Copy Constructor	Assignment 2
	6	• Object as function argument o Overloaded constructor o Member functions defined outside the class o Objects as arguments	
4	7	• The default copy constructor • Returning objects from function • Class, object and memory • Static class data	Quiz 2
	8	• Const and classes o Const member functions o Const objects • Software Engineering Case Study: identifying the classes in the ATM Requirements Document, • Identifying the class Attributes • Objects states and activates	
5	9	**Functions and functions overloading** • Functions • Functions Basics • Overloaded functions/ Function Over-riding o Different numbers of arguments o Different kinds of arguments	

	10	• Inline functions • Default arguments • Variables and storage classes o Automatic variable o External variables o Static variables o Storage • Const function arguments • Software Engineering Case Study: class operation in the ATM system.	Project Milestone- II
6	11	o **Composition, Association and Aggregation**	
	12	Implementation of the case study.	
7	13	**Inheritance:** • Inheritance basics in real world and programming • Derived class and base class o public, private & protected, Abstract Classes o Specifying the derived class o Accessing base class members o The protected access specifier	Assignment 3
	14	• Derived class constructors • Overriding member functions • Class hierarchies o Abstract base class Constructor and member functions	Quiz 3
8	15	• Scope resolution with overridden functions • Public and private inheritance o Access combinations o Access specifiers: when to use what Level of inheritance	
	16	• Multiple inheritance • Ambiguity in multiple inheritance • Containership: classes within class o Composition and aggregation • Inheritance and program development	Project Mile Stone – III
		Mid Term	
9	17	**Operator overloading** • Overloading unary operator • Overloading binary operator • Data conversion o Conversion between basic types o Conversion between objects and basic types o Conversion between objects of different classes	
	18	• Conversion: when to use what. pitfall of operator overloading and c o use similar meanings o use similar syntax o show restraint o avoid ambiguity not all operator can be overloaded	
10	19	**Pointers** • Pointer basics concepts • Addresses and pointers • The address of operator • Pointer and arrays • Pointers and functions • Pointers and ctype string • Memory management: new and delete o The new opearaotr o The delete operator o A string class using new	Project Milestone – IV
	20	• Pointer to objects • Pointers to pointers	

11	21	**Virtual functions** • Virtual functions o Normal member function accessed with pointer o Normal member function accessed without pointer o virtual member function accessed with pointer o Virtual member functions accesses without pointer o Late binding	Assignment4
	22	o Abstract classes and pure virtual functions o Virtual destructors o Virtual base classes	
12	23	• Friend functions • Friend classes • Static functions • The this pointer	
	24	**Polymorphism,** • Type of Polymorphism – o Compile time and runtime, o Function Overloading, o Operator Overloading (Unary and Binary) Polymorphism by parameter,	Quiz 4
13	25	o Pointer to objects, o this pointer, o Virtual Functions, o Pure virtual functions	
	26	**Streams and files** • Stream classes o Advantages of streams o The stream class hierarchy o The ios class o The isteam class o The ostram class	Assignment 5
14	27	• Disk file I/O with streams • File pointers	Quiz 5
	28	• Error handling in file I/O • File I/O with member functions	
15	29	**Multifile Programs** • Reason for multifile program • Creating a multifile program	
	30	o Header file o Directory o Projects • Case study	
16	31	**Templates and exceptions** • Functions templates o A simple functions template o Functions templates with multiple arguments • Class templates	Project Evaluations (Project Milestone-IV)
	32	• Exception o Why do we need exception o Exception syntax o A simple exception example o Multiple exceptions with arguments	

Project should be executed by lab-instructor under course-instructor supervision.