Course Specification
(Bachelor)

Course Title:	**Object-Oriented Programming**
Course Code:	**CS1241**
Program:	**Bachelor of Computer Science**
Department:	**Computer Science**
College:	**College of Computer and Information Sciences**
Institution:	**Imam Mohammad Ibn Saud Islamic University**
Version:	1.0
Last Revision Date:	27 April 2025 *Pick Revision Date.*

Table of Contents

A. General information about the course:

1. Course Identification

1. Credit hours: (4 Hours)					
2. Course type					
A.	☐University	☐College	☒ Department	☐Track	☐Others
B.	☒ Required		☐Elective		
3. Level/year at which this course is offered: (Level 3/Year 2)					
4. Course General Description:					
This course will introduce the student to the concepts of object-oriented programming. Programming topics include data hiding/encapsulation and abstraction using classes and objects, inheritance, polymorphism, generic programming using generic classes and methods, and exception handling.					
5. Pre-requirements for this course (if any)**:**					
CS1140 - Introduction to Computer Programming					
6. Co-requisites for this course (if any)**:**					
7. Course Main Objective(s):					
The main purpose of the course is to build a strong programming background for future study in computer science. Furthermore, it helps students to develop their problem-solving skills by using the proper logical thinking.					

2. Teaching mode (mark all that apply)

No	Mode of Instruction	Contact Hours	Percentage
1	Traditional classroom	90	100%
2	E-learning	-	-
3	Hybrid • Traditional classroom • E-learning	-	-
4	Distance learning	-	-

3. Contact Hours (based on the academic semester)

No	Activity	Contact Hours
1.	**Lectures**	30
2.	**Laboratory/Studio**	30
3.	**Field**	0
4.	**Tutorial**	30
5.	**Others (specify)**	0
Total		90

B. Course Learning Outcomes (CLOs), Teaching Strategies and Assessment Methods

Code	Course Learning Outcomes	Code of PLOs aligned with the program	Teaching Strategies	Assessment Methods
1.0	**Knowledge and understanding**			
1.1	Identify pitfalls related to inheritance side effects [Usage]	K1	1. Class lectures Class exercises and discussions.	Written exams (midterm and final)
1.2	Identify the relative strengths and weaknesses among multiple object-oriented designs or implementations for a problem. [Assessment]	K1, K2, K3	1. Class lectures Class exercises and discussions.	Written exams (midterm and final)
1.3	Describe the process of analyzing and implementing changes to a large existing code base. [Usage]	K2, K3	1. Class lectures 2. Class exercises and discussions. Lab programming exercises	1. Written exams (midterm and final) Lab exams
1.4	Recall common coding errors that lead to insecure programs (e.g., buffer overflows, memory leaks, malicious code, integer overflows, race conditions) and rewrite a simple program to remove such errors. [Usage]	K2, K3	1. Class lectures 2. Class exercises and discussions. Lab programming exercises	1. Written exams (midterm and final) Lab exams

Code	Course Learning Outcomes	Code of PLOs aligned with the program	Teaching Strategies	Assessment Methods
2.0	**Skills**			
2.1	Design and implement an appropriate object-oriented code for a problem by combining different object-oriented concepts (e.g., encapsulation, polymorphism, and inheritance). [Usage]	S1, S2, S3	1. Class lectures 2. Class exercises and discussions. Lab programming exercises	1. Written exams (midterm and final) Lab exams
2.2	Exploit object-oriented inheritance concept to design inheritance hierarchies and reuse code in subclasses. [Usage]	S1, S2	1. Class lectures 2. Class exercises and discussions. Lab programming exercises	1. Written exams (midterm and final) Lab exams
2.3	Use object-oriented polymorphism to make systems extensible and maintainable. [Usage]	S1	1. Class lectures 2. Class exercises and discussions. Lab programming exercises	1. Written exams (midterm and final) Lab exams
2.4	Create and use generic methods and classes that perform identical tasks on arguments or data fields of different types. [Usage]	S1, S4	1. Class lectures 2. Class exercises and discussions. Lab programming exercises	1. Written exams (midterm and final) Lab exams
2.5	Build GUIs and write event handlers to handle events that are generated by user interactions with GUIs. [Usage]	S1	1. Class lectures 2. Class exercises and discussions. Lab programming exercises	1. Written exams (midterm and final) Lab exams
2.6	Apply the principles of least privilege, fail-safe defaults, and isolation as applied to system design. [Familiarity]	S1, S4	1. Class lectures 2. Class exercises and discussions. Lab programming exercises	1. Written exams (midterm and final) Lab exams
2.7	Build robust code using exception handling mechanisms. [Usage]	S1	1. Class lectures 2. Class exercises and discussions. Lab programming exercises	1. Written exams (midterm and final) Lab exams
2.8	Apply techniques, coding idioms and mechanisms for	S1, S2	1. Class lectures 2. Class exercises and discussions.	1. Written exams

Code	Course Learning Outcomes	Code of PLOs aligned with the program	Teaching Strategies	Assessment Methods
	creating correct program components that meet programming style standards and achieve desired properties such as reliability, efficiency, and robustness. [Assessment]		Lab programming exercises	(midterm and final) Lab exams
3.0	**Values, autonomy, and responsibility**			
3.1				
3.2				
...				

C. Course Content

No	List of Topics	Contact Hours

1.	• Object-oriented design
	  ○ Decomposition into objects carrying state and having behavior
	  ○ Class-hierarchy design for modeling
	• Definition of classes: fields, methods, and constructors
	• Subclasses, inheritance, and method overriding
	[CLOs 2.1, 2.2, 2.3]
		18

2.	• Object-oriented idioms for encapsulation
	  ○ Privacy and visibility of class members
	  ○ Interfaces revealing only method signatures
	  ○ Abstract base classes
	• Using collection classes, iterators, and other common library components
	[CLOs 2.2]
		18

3.	• Fundamental design concepts and principles
	  ○ Abstraction
	  ○ Program decomposition
	  ○ Encapsulation and information hiding
	  ○ Separation of behavior and implementation
	[CLOs 1.2, 2.1, 2.3]
		18

4.	• References and aliasing [CLOs 2.3]	2
5.	• Events and event handlers • Canonical uses such as GUIs, mobile devices, robots, servers [CLOs 2.5]	10

6.	• Generic types (parametric polymorphism)
	  ○ Definition
	  ○ Use for generic libraries such as collections
	[CLOs 2.4]
		10

7.	• Program correctness
	  ○ Types of errors (syntax, logic, run-time)
	  ○ The concept of a specification
	  ○ Defensive programming (e.g. secure coding, exception handling)
	  ○ Code reviews
	  ○ Testing fundamentals and test-case generation
	  ○ Unit testing
	• Simple refactoring
	• Modern programming environments
	  ○ Code search
	  ○ Programming using library components and their APIs
	• Debugging strategies
	• Documentation and program style
	[CLOs 1.3, 1.4, 2.8]
		8

8.	• Principle of Least Privilege (PoLP) in OO context [CLOs 2.6]	2

9.	• Principles of secure design and coding (cross-reference IAS/Principles of Secure Design)
	  ○ Principle of least privilege
	  ○ Principle of fail-safe defaults
	  ○ Principle of psychological acceptability
	[CLOs 2.6]
		1

10.	• Coding practices: techniques, idioms/patterns, mechanisms for building quality programs
	  ○ Defensive coding practices
	  ○ Secure coding practices
	  ○ Using exception handling mechanisms to make programs more robust, fault-tolerant
	• Coding standards
	• Development context: “green field” vs. existing code base
	  ○ Change impact analysis
	  ○ Change actualization
	[CLOs 2.7, 1.3, 2.8]
		1

11.	• Potential security problems in programs
	  ○ Buffer and other types of overflows
	  ○ Race conditions
	  ○ Improper initialization, including choice of privileges
	  ○ Checking input
	  ○ Assuming success and correctness
	  ○ Validating assumptions
	[CLOs 1.4]
		1

12.	• Attribute and method inheritance & extension examples that may lead to security issues in some OO languages (C++ or JAVA) [CLOs 1.1, 2.2]	1
Total		**90**

D. Students Assessment Activities

No	Assessment Activities *	Assessment timing (in week no)	Percentage of Total Assessment Score
1.	1st Quiz	4th	7.5%
2.	1st Lab Exam	5th	10%
3.	Midterm Exam	8th	20%
4.	2nd Lab Exam	10th	10%
5.	2nd Quiz	9th	7.5%
6.	Lab Evaluations	regularly	5%
7.	Final Exam	Final Week	40%

*Assessment Activities (i.e., Written test, oral test, oral presentation, group project, essay, etc.).

E. Learning Resources and Facilities

1. References and Learning Resources

Essential References	Java: How to Program (Early Objects Version), Global Edition, 10/E, Paul Deitel and Harvey Deitel, Pearson, 2015, ISBN-10: 0133807800• ISBN-13: 9780133807806
Supportive References	
Electronic Materials	
Other Learning Materials	

2. Required Facilities and equipment

Items	Resources
facilities (Classrooms, laboratories, exhibition rooms, simulation rooms, etc.)	Classroom with: 1. At least 35 seats 2. A projector connected to a PC preferably with Internet connection Sliding board
Technology equipment (projector, smart board, software)	
Other equipment (depending on the nature of the specialty)	

F. Assessment of Course Quality

Assessment Areas/Issues	Assessor	Assessment Methods
Effectiveness of teaching	Students	1. Students feedback (collected through surveys) as per

Assessment Areas/Issues	Assessor	Assessment Methods
		university policy/procedure Teacher's Course report
Effectiveness of Students assessment	Faculty	1. 1. Review of Course Reports 2. Review of Student feedback
Quality of learning resources	Program Leaders	Continuous review of the course contents and teaching strategies, and utilizing the best practices
The extent to which CLOs have been achieved	Peer Reviewer	Upon student request, his/her work might be rechecked by another faculty member within the department
Other	Program Leaders	Reviewing Course Specification and reports at the end of each academic year/semester, which is a part of the continuous improvement process in the department

Assessors (Students, Faculty, Program Leaders, Peer Reviewers, Others (specify)

Assessment Methods (Direct, Indirect)

G. Specification Approval

COUNCIL /COMMITTEE	**COMPUTER SCIENCE DEPARTMENT COUNCIL**
REFERENCE NO.	**1446/3101**
DATE	**29-4-2025**