COURSE OUTLINE: CSD214 - PROG. CONCEPTS II

Prepared: Rodney Martin
Approved: Corey Meunier, Chair, Technology and Skilled Trades

Course Code: Title	CSD214: PROGRAMMING CONCEPTS II
Program Number: Name	2095: COMPUTER PROGRAMMING
Department:	COMPUTER STUDIES
Academic Year:	2023-2024
Course Description:	All programmers must learn to manage complexity in their software. By exploring advanced data structures, design patterns, the S.O.L.I.D. design principles, test-driven development (TDD), Model-View-Controller (MVC) frameworks, and Object-Relational Mappers (ORMs), students in this course learn and practice the high-level design and development techniques that make software systems simpler to test, enhance, and maintain. This course is delivered using the Java programming language.
Total Credits:	4
Hours/Week:	4
Total Hours:	56
Prerequisites:	CSD121, CSD123
Corequisites:	There are no co-requisites for this course.
This course is a pre-requisite for:	CSD223, CSD226, CSD228, CSD230, CSD235
Vocational Learning Outcomes (VLO's) addressed in this course: **Please refer to program web page for a complete listing of program outcomes where applicable.**	**2095 - COMPUTER PROGRAMMING** VLO 2 Contribute to the diagnostics, troubleshooting, documenting and monitoring of technical problems using appropriate methodologies and tools. VLO 4 Implement robust computing system solutions through validation testing that aligns with industry best practices. VLO 5 Communicate and collaborate with team members and stakeholders to ensure effective working relationships. VLO 10 Contribute to the development, documentation, implementation, maintenance and testing of software systems by using industry standard software development methodologies based on defined specifications and existing technologies/frameworks. VLO 11 Apply one or more programming paradigms such as, object-oriented, structured or functional programming, and design principles, as well as documented requirements, to the software development process. VLO 12 Model, design, implement, and maintain basic data storage solutions. VLO 13 Contribute to the integration of network communications into software solutions by adhering to protocol standards.
Essential Employability	

Skills (EES) addressed in this course:	EES 2 Respond to written, spoken, or visual messages in a manner that ensures effective communication. EES 4 Apply a systematic approach to solve problems. EES 5 Use a variety of thinking skills to anticipate and solve problems. EES 6 Locate, select, organize, and document information using appropriate technology and information systems. EES 10 Manage the use of time and other resources to complete projects. EES 11 Take responsibility for ones own actions, decisions, and consequences.
Course Evaluation:	Passing Grade: 50%, D A minimum program GPA of 2.0 or higher where program specific standards exist is required for graduation.
Other Course Evaluation & Assessment Requirements:	Students are expected to be present to write all tests in class, unless otherwise specified. If a student is unable to write a test due to illness or a legitimate emergency, that student must contact the professor prior to class and provide reasoning. Should the student fail to contact the professor, the student shall receive a grade of zero on the test. If a student is not present 10 minutes after the test begins, the student will be considered absent and will not be given the privilege of writing the test. Students exhibiting academic dishonesty during a test will receive an automatic zero. Please refer to the College Academic Dishonesty Policy for further information. In order to qualify to write a missed test, the student shall have: a.) attended at least 75% of the classes to-date. b.) provide the professor an acceptable explanation for his/her absence. c.) be granted permission by the professor. NOTE: The missed test that has met the above criteria will be an end-of-semester test. Labs / assignments are due on the due-date indicated by the professor. Notice by the professor will be written on the labs / assignments and verbally announced in the class. Labs and assignments that are deemed late will have the following penalty: 1 day late - 10% reduction, 2 days late, 20% reduction, 3 days late, 30% reduction. After 3 days, no late assignments and labs will be accepted. It is the responsibility of the student who has missed a class to contact the professor immediately to obtain the lab / assignment. Students are responsible for doing their own work. Labs / assignments that are handed in and are deemed identical or near identical in content may constitute academic dishonesty and result in a zero grade. Students are expected to be present to write in-classroom quizzes. There are no make-up options for missed in-class quizzes. Students have the right to learn in an environment that is distraction-free, therefore, everyone is expected to arrive on-time in class. Should lectures become distracted due to students walking in late, the professor may deny entry until the 1st break period, which is 50 minutes into the class or until that component of the lecture is complete. Grade

	Definition	Grade Point Equivalent
A+	90 - 100%	4.00
A	80 - 89%	
B	70 - 79%	3.00
C	60 - 69%	2.00
D	50 - 59%	1.00
F (Fail)	49% and below	0.00

CR (Credit) Credit for diploma requirements has been awarded.
S Satisfactory achievement in field /clinical placement or non-graded subject area.
U Unsatisfactory achievement in field/clinical placement or non-graded subject area.
X A temporary grade limited to situations with extenuating circumstances giving a student additional time to complete the requirements for a course.
NR Grade not reported to Registrar`s office.
W Student has withdrawn from the course without academic penalty.

Books and Required Resources:

Big Java: Objects First by Cay S. Horstmann
Publisher: Wiley Edition: 7
ISBN: 978-1-119-49909-1

Course Outcomes and Learning Objectives:

Course Outcome 1	**Learning Objectives for Course Outcome 1**
1. Describe and apply high-level software design principles	1.1 Construct and interpret UML diagrams, and discuss how they relate to OOP design 1.2 Explain the importance of loose coupling and strong cohesion in software systems 1.3 Discuss the dis/advantages of composition vs inheritance, and explain when each is most appropriate 1.4 Describe the S.O.L.I.D. design principles of OOP 1.5 Design an OOP software system from a problem description and related information 1.6 Describe the Model-View-Controller architecture 1.7 Explain the advantages of a tiered software architecture 1.8 Create modular software applications 1.9 Use a framework to implement a software design
Course Outcome 2	**Learning Objectives for Course Outcome 2**
2. Create graphical user interfaces	2.1 Create GUIs using common layout managers and controls 2.2 Utilize event-driven programming to respond to user actions and system events 2.3 Use listeners and binding to dynamically update UI components
Course Outcome 3	**Learning Objectives for Course Outcome 3**
3. Describe and employ common programming design patterns	3.1 Describe the purpose and nature of programming design patterns 3.2 Describe the broad design pattern categories: Creational, Structural, and Behavioural 3.3 Describe common individual design patterns, and explain their typical use cases 3.4 Write software that makes appropriate use of design patterns

Course Outcome 4	Learning Objectives for Course Outcome 4
4. Integrate a database with an application using an Object-Relational Mapper (ORM)	4.1 Describe the nature of ORMs 4.2 Explain the importance of software layers to isolate database code from business logic 4.3 Configure an ORM to integrate a database with a working program 4.4 Perform create, read, update, and delete operations using an ORM
Course Outcome 5	**Learning Objectives for Course Outcome 5**
5. Test and validate software functionality	5.1 Use the Test-Driven Development technique to write software 5.2 Describe the difference between unit, integration, and end-to-end testing 5.3 Write unit tests to ensure correct functioning of program sub-components 5.4 Write integration tests to validate the correct functioning of larger program components 5.5 Write end-to-end tests to verify the correct functioning of an application

Evaluation Process and Grading System:

Evaluation Type	Evaluation Weight
Coding Assignments	40%
Test (final)	25%
Tests (mid-term)	35%

Date: June 14, 2023

Addendum: Please refer to the course outline addendum on the Learning Management System for further information.