Course Outline

Course: SWE121 Structure Programming

Teaching Staff:

Category	Description
Corse Teacher with Section and Semester	Md. Shohel Arman, Summer 2020
Course Code	SWE121
Title	Structure Programming
Credit	3
Total Cr. Hour	40
Class room	
Class time	
Office Hour	
Consultancy Hour	
Email	arman.sew@diu.edu.bd
Google Classroom Code	

Course Objectives (CO's):

1. The students should learn the differences between different types of data storage and why each is need. How data is transferred from RAM to permanent storage including the use of arrays.
2. The students will learn how the control structures of sequence, repetition and condition can be used to process data.
3. The students should learn the various parts of a C program and how modular structures are introduced. They would learn all these as part of a class project and should complete the course with a small project.

Prerequisite: SWE112 Computer Fundamentals

Learning Outcomes (LO’s):

No.	Outcomes (LO’s)
Knowledge	
1	Describe programming methodologies
2	Explain the basic concepts of programming principles, including programming style, developing approach, implementation, testing and maintenance
3	Identify programming data types, control structure
Skill	
4	To be able to load and save data from a file in order to test small programs
5	To be able to explain the parts of a C program and the need for structure
6	To be able to identify the logical errors in the sequence, condition and repetition
Self Development	
7	To be able to demonstrate learned concepts by developing a simple project in a structured language.
Role in Context	
Attitude	
8	To be able to develop confidence for braking down and solving problems.

Course Description:

Structured programming sets the foundation for understanding the mechanics of modern structured languages for machines. This course introduces a methodical approach to program development, starting from problem formulation and requirement specification, through design of the logical solution, implementation error finding and evaluation of the solution with documentation. The course matter is taught through a high-level structured programming language.

Course Methodology:

1. Class size of 30-35 Students
2. Classroom with a projector, a computer and a white marker board
3. Industrial visit may be required for the course at the end of the semester
4. Students may need to attend seminar and workshop during the turner of the course

Assessment:

No.	Assessment Methods	Weighing			Remarks
1	Continuous Assessment	50%	5%	Attendance	To measure how well students have learned throughout the semester.
			15%	Quiz (Min 3 Quizzes)	
			5%	Assignment	
			25%	Lab	
2	Examinations	50%	20%	MID term exam	

			30%	Final Exam	To measure how far students have achieved the learning outcomes.

Mapping of Assessment with Learning Outcomes (LO’s):

No.	Learning Outcome	Course Assessment Methods					
(LO’S)		Attendance	Quiz	Assignment	MID	LAB	FINAL
1	Describe programming methodologies	x	x		x		
2	Explain the basic concepts of programming principles, including programming style, developing approach, implementation, testing and maintenance	x	x	x	x	x	
3	Identify programming data types, control structure	x	x	x	x	x	x
4	To be able to load and save data from a file in order to test small programs	x	x			x	x
5	To be able to explain the parts of a C program and the need for structure	x	x		x	x	x
6	To be able to identify the logical errors in the sequence, condition and repetition	x	x			x	x
7	To be able to demonstrate learned concepts by developing a simple project in a structured language	x		x	x	x	x
8	To be able to develop confidence for braking down and solving problems.			x	x	x	x

Rubrics:

No.	Weighing	Letter Grade	Category	Description
1	80%	A+	Outstanding	Very Strong evidence of having achieved all the LO's and demonstration of exceptional mastery of programming knowledge and skills. Able to develop correct programs to solve problems Demonstration of exceptional mastery of program design, testing and debugging.
2	75%	A	Excellent	Strong evidence of having achieved all the LO's and demonstration of mastery of programming knowledge and skills. Able to develop correct programs to solve problems Demonstration of mastery of program design, testing and debugging.
3	70%	A-	Very Good	Evidence of having achieved 90% of the LO's with good understanding of programming knowledge and skills. Able to develop correct programs to solve problems Demonstrate a complete level of program design, testing and debugging.
4	65%	B+	Good	Evidence of having achieved 80% of the LO's with understanding of programming knowledge and skills. Able to develop correct programs to solve problems Demonstrate a complete level of program design, testing and debugging.
5	60%	B	Satisfactory	Evidence of having achieved 70% of the LO's with basic understanding of programming knowledge and skills. Able to develop acceptable solution to solve problems Demonstrate a adequate level of program design, testing and debugging
6	55%	B-	Above Average	Evidence of having achieved 60% of the LO's with minimal understanding of programming knowledge and skills.

				Able to provide solution to simple problems Demonstrate a basic level of program design, testing and debugging
7	50%	C+	Average	Evidence of having achieved 50% of the LO’s with minimal understanding of programming knowledge and skills. Able to provide solution to simple problems. Demonstrate a basic level of program design, testing and debugging.
8	45%	C	Below Average	Evidence of having achieved 40% of the LO’s with minimal understanding of programming knowledge and skills. Able to provide solution to very simple problems. Demonstrate a low level of program design, testing and debugging.
9	40%	D	Pass	Evidence of having achieved 30% of the LO’s with little understanding of programming knowledge and skills. Able to provide solution to very simple problems. Demonstrate a very lower level of program design, testing and debugging.
10	<40	F	Fail	Evidence of having achieved below 30% of the LO’s with very little understanding of programming knowledge and skills. Unable to provide solution to very simple problems. Programming knowledge and skills falling below the basic minimum level.

Teaching Method (TM):

No.	**Method Name**	**Description**	**Medium Used**
1.	**Authority, or lecture style**	This traditional, formal approach to teaching is sometimes referred to as “the sage on the stage.”	**white board, marker**
2.	**Demonstrator, or coach style**	This style retains the formal authority role while allowing teachers to demonstrate their expertise by showing students what they need to learn	**use projector, PC, presenter**

3.	**Facilitator, or activity style**	This approach encourages teachers to function as advisors who help students learn by doing.	**Use VIP card, Marker, provide sheet**
4.	**Developer, or group style**	This style allows teachers to guide students in a group setting to accomplish tasks and learn what works or doesn't.	**use card and poster and presentation in a group**
5.	**Hybrid, or blended style**	This approach incorporates different aspects of the various styles and gives teachers flexibility to tailor a personal style that's right for their coursework and students	**use multiple approaches together**
6.	**Virtual Learning**	Use Google classroom or Learning feedback system to provide distant learning to students.	**Use internet, computer network, web site**

Mapping of Lesson plan to the Learning Outcomes and Teaching Methods:

No.	Session	Topic/ Content	Teaching Method (TM)	Learning Outcomes (LO's)
1	Session 1	Programming Methodology Class 1: Brief History, Programming Basics Class 2: Structured programming, Programming Environment	1, 2	1, 2
2	Session 2	Structured programming: Class 1: character set, reserved words, identifiers Class 2: data types, data type modifiers, constants, escape sequences	1,2,3	1,2,3
3	Session 3	Structured programming: Class 1: variables, expressions Class 2: statements, operators	1,2,3	2,3
4	Session 4	Control structures: Class 1: IF, IF-ELSE, IF-ELSE-IF Class 2: Problem solve using IF, IF-ELSE, IF-ELSE-IF **Assignment 1 through google classroom**	1,2,4,6	3,4,5
5	Session 5	Control structures:	1,2,4	3,4,5

		Class 1: SWITCH CASE as an alternative approach of IF-ELSE-IF Class 2: **QUIZ 1**, Review the IF, IF-ELSE, IF-ELSE-IF, SWITCH CASE		
6	Session 6	Control structures: Class 1: WHILE and DO-WHILE loop Class 2: Problem solve using WHILE and DO-WHILE loop	1,2,4	3,4,5,6
7	Session 7	Control structures: Class 1: FOR loop and problem solve using FOR loop Class 2: QUIZ 2, Review the WHILE , DO-WHILE and FOR loop **Solving problem 1 through google classroom**	1,2,4,6	3,4,5,6
8	Session 8	Data storage: Class 1: ARRAY and POINTER Class 2: Problem solve using ARRAY and POINTER	1,2,3	3,4,5
9	Session 9	Functions and program structure: Class 1: function basics Class 2: Problem solve using basics function	1,2,4	3,4,5,6
10	Session 10	Functions and program structure: Class 1: parameter passing conventions Class 2: Problem solve using parameter passing conventions **Solving problem 2 through google classroom**	1,2,4,6	3,4,5,6,7,8
11	Session 11	Content title Class 1: recursive function and problem solve using recursive function Class 2: **QUIZ 2**, and review the Functions	1,2,4	3,4,5,6,7,8
12	Session 12	User defined data types: Class 1: structures, unions, enumerations Class 2: Problem solve using structures, unions, enumerations	1,2,4,6	3,4,5,6,7,8

13		FILE access: Class 1: FILE access process Class 2: Problem solve using FILE access **QUIZ 3 through google classroom**	1,2,4,6	3,4,5,6,7,8

Reference:

1. Programming in ANSI C Seventh Edition by E. Balagurusamy
2. Teach yourself C Third Edition Herbert Schildt

Disclaimer:

While every effort has been made to ensure that the information contained in this document is accurate, the information is subject to change. Changes will be notified in class and/or tutorials, via Google Classroom or email. Students are encouraged to check Google Classroom or email for any changes. It is your responsibility to be informed.