Syllabus

CSC 214 – Structured Programming (Credit Units: 3)

Department of Information Communication Technology
Faculty of Computing
University of Delta, Agbor, Nigeria

Lecturer: Dr. A.E.Okpako
Instructor: Miss Snow
Office Location: Room 2
Lecture Hall: Hall 5
Email:
Phone: +234 7088991282
Office Hours: Monday , Tuesday , Wednesday, Thursday & Friday
.

Meeting Time and Place: Monday, 1:00 pm to 3:00 pm,

Attendance

Every class attendance is compulsory. If you must miss a class, it is your responsibility to make up for the work that you missed. If you are going to be absent from any class, you must please notify the instructor in advance.

Methods of Instruction

This syllabus contains an overview of what will be covered in class; for specific information, students are referred to the class web page maintained on the University website. Assignments will be posted on University of Delta LMS or given in the class and should be submitted through University of Delta LMS. Class attendance, doing all your practical and homework will help the borderline cases.

Overview

Structured Programming cannot be separated from application development. A structured programmer needs knowledge on how a program works to effectively do his/her job. Programmers using structured programs to do their jobs must have knowledge of the structured control flow constructs, block structures and subroutines. The course presents an overview of structured programming with main focus on C programming Language and Linux Environment. The objective of this class is to emphasize the fundamentals of the structured programming. Students will learn the different elements of structured programming, structured design principles and structured design techniques which will enable them solve problems using C programming language. The course focuses mainly on abstraction modularity, stepwise refinement, variables and data types, input and output, logical expressions and control flows, loops and arrays.

Objectives

The objectives of this course are to: (i) give an introduction on Structured programming; (ii) use C programming language to demonstrate the elements of structured programming; (iii) demonstrate abstraction and modularity by writing functions; (iv) discuss stepwise refinement; (v) discuss structured design principles and techniques; (vi) give an introduction on Linux Environment; (vii) give an introduction on C programming language; (viii) describe variables and types using illustrations; (ix) discuss Input and Output operations; (x) illustrate logical expressions and control flow; (xi) demonstrate loops; and (xii) demonstrate arrays

Learning outcomes

Upon completion of this course, the students should be able to: (i) explain sequence, selection and iteration as structured programming elements and demonstrate each using C programming language ; (ii) explain modularity, encapsulation, cohesion, loose coupling, stepwise refinement and structured control flow as structured design principles, thus create designs that are modular; (iii) describe the major structured design techniques; (iv) explain Linus Environment, its architecture, advantages and disadvantages; (v) explain C programming language, its features and application areas; (vi) describe variables and data types and demonstrate the format for declaring variables and data types; (vii) use C programming language to demonstrate basic functions for Input and Output; (viii) explain logical expressions and control flow with illustrations using C programming language; (ix) demonstrate different loop structures using C programming language; (x) explain array with illustrations using C programming language

Course Contents

Structured programming introduced. Structured programming elements. Structured design principles. Abstraction and Modularity. Stepwise refinement. Structured design techniques. Introduction to Linux. Introduction to C programming. Variables and data types. Input and Output. Logical expressions and Control flow. Loops. Arrays

Lecture Schedules

Week	Content	Lecture notes/slides
1.	Introduction to Structured Programming	
2.	Structured programming elements	
3.	Structured design principles	
4.	Abstraction and Modularity & Stepwise refinement	
5.	Structured design techniques	
6.	Introduction to Linux Environment	
7.	Introduction to C programming	
8.	Variables and Data Types	
9.	Test	
10.	Input and Output	
11.	Logical Expressions and Control flow	
12.	Loops	
13.	Arrays	
14.	Revisions	
15.	Final Exam	

Examination schedule

- Attendance
- Homework
- Class Test
- Practical exercises
- End of Semester Exam

Practical Exercises

1: loop structure for negative and positive step size using flowchart
2: C program to add n numbers and print the n numbers
3: C program to perform division of two numbers
4: C program to print message if negation number is entered
5: C program to Sum two numbers, find the Quotient and print if Average is less than Sum
6: C program to check for the relation between two numbers
7: C program for a University's admission system to accept students who passed Mathematics and English and either Chemistry or Physics
8: C program to add all the numbers entered by user until user enters 0
9: C program to print the integers from 1 to 6 and their cubes
10: C program to print Sum of N natural numbers
11: C program to print series from 20 to 1 using nested loop
12: C program to print EVEN or ODD depending on the integer number entered at the prompt and Simulate the output
13: C program to increment every element of a given array by 1 and print the incremented array
14: C program to find Sum of two matrices using an array function

Grading

- Homework: 10% of grade
- Practical: 10% of grade
- Midterm Exam: 10% of grade
- Final Exam: 70% of grade

Text & References

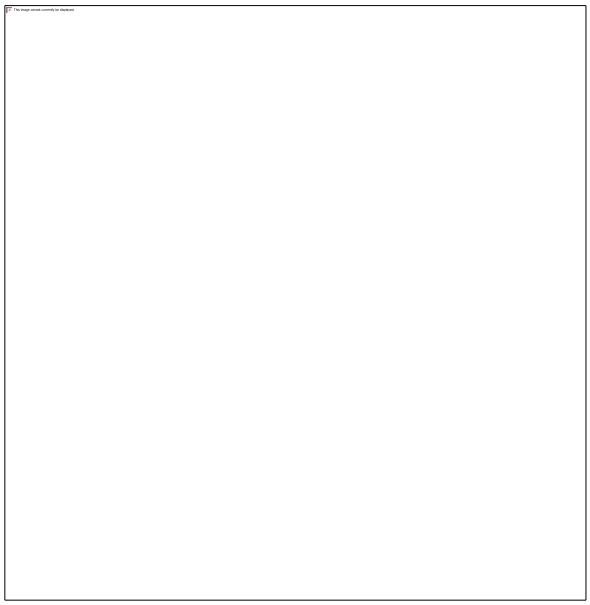
Lubia V. (2016), Fundamentals of Structured Programming Language
Guru C. (2016), C Programming for Beginners
Tim B. (2005), An Introduction to C programming Language and Software Design

Student Conduct

All students enrolled at the University shall follow the tenets of common decency and acceptable behaviour conducive to a positive learning environment. The code of student conduct is described in detail in the student handbook or University website.

Academic Honesty

"All students enrolled at the University shall follow the tenets of common decency and acceptable behaviour conducive to a positive learning environment." It is the policy of the University, that no form of plagiarism or cheating will be tolerated. Plagiarism is defined as the deliberate use of another's work and claiming it as one's own. This means ideas as well as text or code, whether paraphrased or presented verbatim (word-for-word). Cheating is defined as obtaining unauthorised assistance on any assignment. Proper citation of sources must always be utilised thoroughly and accurately. If you are caught sharing or using other people's work in this class, you will receive a 0 grade and a warning on the first instance. A subsequent instance will result in receiving an F grade for the course, and possible disciplinary proceedings. If you are unclear about what constitutes academic dishonesty, ask.


This image cannot currently be displayed.