CIS 161 - Introduction to C Programming – SPRING 2015

INSTRUCTOR: Ron Mummaw OFFICE: BE 210
PHONE: 661-722-6379 E-MAIL: rmummaw@avc.edu
OFFICE HOURS: MTWR 7:00am-8:00am; MW 5:30-6:00pm
WEB SITE: http://bit.ly/ronmum
ADVISORY: **Completion of CIS 111**, and Eligibility for ENGL 099, READ 099 and MATH 130.
TEXTBOOK: Programming in C - zybooks.zyante.com Instructions for accessing the textbook:

1. Sign up at zybooks.zyante.com
2. Enter zyBook code: AVCCIS161MummawSpring2015
3. Click 'Subscribe'

DISCUSSION FORUM: This term we will be using **Piazza** for class discussion. The system is highly catered to getting you help fast and efficiently from classmates and your instructor. Rather than emailing questions you are encouraged to post your questions on Piazza. https://piazza.com/avc/fall2014/cis161/home

DESCRIPTION: This course is designed to give the student an introduction to the C programming language. Students will learn the basic elements of the C language, and a disciplined approach to program development using structured programming techniques for readability, maintainability, and defensive programming. Problem solving through stepwise development of algorithms and the mechanics of running, testing, and correcting programs is presented. Students will plan, create, test, and run their own programs to solve typical problems. BEFORE ENROLLING students should have basic computer experience and be able to save and retrieve files, run applications, print documents, and have sufficient aptitude with mathematics to solve simple algebraic equations and to appreciate the use of mathematical notation and formalism**.** ***You should also have completed with a passing grade, one semester of a programming class.***

ASSIGNMENTS: During the course of the semester, you will be assigned one programming assignment, textbook activities and textbook homework assignments for each chapter. Obviously copied work (from ANY other source) will be given no points. Be prepared to explain every line of code in your program, why it is there and what it does. Chapter activities and homework problems must be completed in one week. Programming assignments are due one week after the chapter discussion, however, they will be accepted up to two weeks after the chapter is discussed. A programming assignment not turned in after by 6:00pm two weeks after the chapter discussion will be given **NO** credit, ***REGARDLESS*** of the reason. No time will be given at the beginning of class for printing on the night of the deadline. Assignments may not be emailed. Failure to complete any combination of more than 10 programming/activity/homework assignments will result in you being dropped from the class.

IMPORTANT STUFF TO REMEMBER IF YOU WANT 5 POINTS ON A PROGRAMMING ASSIGNMENT!!!

1. The first thing I want to see on the top of your paper is your name on one line all by itself. On the second line, I want to see the chapter number.
2. Assignments must be turned in at the BEGINNING of class on ***or before*** the deadline to receive credit. NO CREDIT WILL BE GIVEN TO LATE WORK (no exceptions). You will not be given time to print out a program on the deadline day.
3. All identifiers (variable/function names) must be a minimum of three characters
4. Code must be properly formatted as per examples given in class.

EXAMS: Three exams will be given. Two exams during the semester and one final exam. All three are equal credit. You will not be allowed to make up a missed exam unless I receive prior notice that you are going to be absent. You will be expected to be prepared to take the exam on your return.

GRADING:

chapter activities/homework	10 @ 5 points each	50 points	
in-class quiz	10 @ 5 points each	50 points	
programming assignments	10 @ 5 points each	50 points	
tests	3 @ 50 points each	150 points	
total		300 points	
		270 - 300	A
		240 - 269	B
		etc.	

ATTENDANCE: More than 2 absences or 4 tardies (1 absence = 2 tardies) will result in you being dropped from the class. (I may make exceptions to this if I am given advance notice and there is a legitimate reason.) **Missing the first or second class meeting without notifying the instructor prior to the class meeting time will result in you being dropped from the class if there is a waiting list.**

Schedule (week beginning):

02/09/15	Intro
02/16/15	HOLIDAY
02/23/15	Chapter 1
03/02/15	Chapter 2
03/09/15	Chapter 3
03/16/15	Chapter 4
03/23/15	TEST chapters 1-3
03/30/15	Chapter 5
04/06/15	SPRING BREAK
04/13/15	Chapter 6
04/20/15	Chapter 7
04/27/15	TEST chapters 4-6
05/04/15	Chapter 8
05/11/15	Chapter 9
05/18/15	Chapter 10
05/25/15	HOLIDAY
06/01/15	TEST Chapters 7-10

LAB RULES:

1. Food and drinks are not allowed on the third floor of this building with the exception of bottled water. There will be no exceptions to this rule.
2. Computers must be turned off during the lecture unless otherwise specified.

Student Learning Outcomes (to be assessed each semester):

Learning Outcome	Achievement Method
Evaluate and describe the functions of C program code written by others.	Short answer questions assessed with a rubric developed by faculty.
Describe and explain the syntax and elements of the C programming language.	Short answer questions assessed with a rubric developed by faculty.
Plan, code, run, and correct required problems using C.	Based on a set of requirements, students will create and test simple programs written in C containing basic programming structures to be assessed using a rubric developed by faculty.

COURSE OBJECTIVES:

Upon completion of course, the successful student will be able to:

1. Describe and explain the essence of the C programming language and characteristics specifically related to:
   a. declarations of variables of various data types
   b. arithmetic and logical operators
   c. control structures for selection and iteration
   d. function definitions
   e. passing arguments to functions and receiving its returned value
   f. declaring and using arrays, strings, and structures
   g. console input and output
   h. reading and writing data files
   i. bit operations
   j. pointers
   k. standard C library routines
2. Solve problems through stepwise development of algorithms.
3. Use structured programming techniques to plan, code, run, test, and debug C code to solve the problem.
4. Create well documented, highly modular and easily maintainable programs.

COURSE OUTLINE:

- I. Introduction
  - a. Operating systems
  - b. Overview of programming languages
  - c. Compiling, testing, and debugging
  - d. Integrated development environment
  - e. Structure of a C program
  - f. Header files
  - g. Preprocessor commands and macros
  - h. Program documentation
- II. Syntax and Semantics of the C Language
  - a. Data types
  - b. Naming of identifiers
  - c. Variable declarations and definitions
  - d. Arithmetic, logical, and bit-wise operators
  - e. Assignment operator
  - f. Expressions
  - g. Operator precedence and associativity
  - h. Variable scope, storage class, and lifetime
  - i. Conditional and iterative control structures
  - j. Function definitions and declarations
  - k. Passing values and addresses to functions
  - l. Complex data types
    - 1. Arrays
    - 2. Addresses and pointers
    - 3. Character strings
    - 4. Structures
  - m. Console input and output
  - n. Data files
  - o. Casts
  - p. Dynamic memory allocation
  - q. Command line arguments
- III. Standard C Library Functions
  - a. Input and output functions
  - b. Character functions
  - c. Mathematical functions
  - d. Utility functions
  - e. String functions
- IV. Machine Level Representation of Data
  - a. Bits, bytes, and words
  - b. Numeric data representation
  - c. Number bases: dec, oct, hex
  - d. Representation of character data
- V. Algorithm and Problem Solving
  - a. Problem solving strategies
  - b. Implementation strategies for algorithms
  - c. Debugging strategies
- VI. Software Development Methodology
  - a. Fundamental design concepts
  - b. Functional decomposition
  - c. Human-computer interaction