

# EASTERN ARIZONA COLLEGE

## C Programming I

Course Design

2010-2011

### Course Information

**Division** Mathematics  
**Course Number** EGR 130  
**Title** C Programming I  
**Credits** 3  
**Developed by** Mike Moore  
**Lecture/Lab Ratio** 2 Lecture/2 Lab

### Transfer Status

ASU	NAU	UA
CSE 100, Computer/Stats (CS)	CS 122 also satisfies: Science/Applied Science [SAS]	ECE 175

**Activity Course** No  
**CIP Code** 14.0101  
**Assessment Mode** Pre/Post Test (50 Questions/50 Points)  
**Semester Taught** Upon Request  
**GE Category** None  
**Separate Lab** No  
**Awareness Course** No  
**Intensive Writing Course** No

### Prerequisites

None

### Educational Value

This purpose of this course is to learn programming principles and techniques using the C language. This course would also be helpful in other programming courses such as Java Programming.

### Description

A comprehensive introduction to the C language; preparation and writing of programs using C, using current programming techniques. A prior programming language is helpful. For engineering majors. Identical to CMP 130.

### Supplies

USB Flash Drive to save programs.

## **Competencies and Performance Standards**

### **1. Define computer hardware and software.**

#### **Learning objectives**

*What you will learn as you master the competency:*

- a. Differentiate among early computers and modern computers.
- b. Understand the functions of computer hardware including memory, central processing units, input devices, and output devices.
- c. Recognize the different types of computer software and the languages.
- d. Explain the Software Development Method.
- e. Apply the Software Development Method to solve programming problems.

#### **Performance Standards**

*Competence will be demonstrated:*

- By successful completion of required program.
- By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- Learner can differentiate among early computers and modern computers.
- Learner understands the functions of computer hardware including memory, central processing units, input devices, and output devices.
- Learner can recognize the different types of computer software and the languages.
- Learner explains the Software Development Method.
- Learner applies the Software Development Method to solve programming problems.

### **2. Understand the element of a C Program and the types of data that can be processed by C.**

#### **Learning objectives**

*What you will learn as you master the competency:*

- a. Evaluate C Programs using directives and identifiers.
- b. Understand the importance of variables and data types.
- c. Use executable statements to code algorithms.
- d. Understand the general form of a C Program.
- e. Use arithmetic expressions to solve programming problems.
- f. Specify the format of numbers in program output.
- g. Differentiate between interactive mode and batch mode.
- h. Locate errors in a C Program.

#### **Performance Standards**

*Competence will be demonstrated:*

- By successful completion of required program.
- By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- Learner evaluates a C Program using directives and identifiers.
- Learner understands the importance of variables and data types.
- Learner uses executable statements to code algorithms.
- Learner can understand the general form of a C Program.

- Learner can use arithmetic expressions to solve programming problems.
- Learner specifies the format of numbers in program output.
- Learner can differentiate between interactive mode and batch mode.
- Learner can locate errors in a C Program.

### 3. Use the Top-Down design techniques to simplify algorithms.

#### **Learning objectives**

*What you will learn as you master the competency:*

- a. Build a program from existing information.
- b. Use library functions to solve mathematical computations.
- c. Use structure charts to track relationships among subproblems.
- d. Define functions that have no arguments.
- e. Write functions containing input arguments.

#### **Performance Standards**

*Competence will be demonstrated:*

- By successful completion of required program.
- By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- Learner can build a program from existing information.
- Learner uses library functions to solve mathematical computations.
- Learner uses structure charts to track relationships among subproblems.
- Learner can define functions that have no arguments.
- Learner can write functions containing input arguments.

### 4. Use selection structures containing the If and Switch statements.

#### **Learning objectives**

*What you will learn as you master the competency:*

- a. Control the flow of execution in a program or function using control structures.
- b. Establish criteria for a group of statements using conditions.
- c. Write IF statements to execute given values.
- d. Write IF statements with compound statements.
- e. Write a program using decision steps.
- f. Modify a program containing function subprograms.
- g. Use nested IF statements to code decisions with multiple alternatives.
- h. Use the Switch statement to execute simple expressions.

#### **Performance Standards**

*Competence will be demonstrated:*

- By successful completion of required program.
- By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- Learner can control the flow of execution in a program or function using control structures.

- Learner can establish the criteria for a group of statements using conditions.
- Learner writes IF statements to execute given values.
- Learner writes IF statements with compound statements.
- Learner writes a program using decision steps.
- Learner modifies a program containing function subprograms.
- Learner uses nested IF statements to code decisions with multiple alternatives.
- Learner uses the Switch statement to execute simple expressions.

## 5. Write and use nested loops in programs.

### **Learning objectives**

*What you will learn as you master the competency:*

- a. Include a repetition structure in a program.
- b. Code a pretest loop using the while statement.
- c. Compute a sum or product in a loop.
- d. Code a loop using the For statement.
- e. Code conditional loops.
- f. Formulate a valid loop structure.
- g. Include nested loops in programs.
- h. Code a loop using the do-while statement.
- i. Use the top-down design to solve programming problems.
- j. Recognize ways to debug and test programs.
- k. Recognize common programming errors.

### **Performance Standards**

*Competence will be demonstrated:*

- By successful completion of required program.
- By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- Learner includes a repetition structure in a program.
- Learner codes a pretest loop using the while statement.
- Learner can compute a sum or product in a loop.
- Learner codes conditional loops.
- Learner formulates a valid loop structure.
- Learner includes a nested loop in programs.
- Learner uses the top-down design to solve programming problems.
- Learner can recognize ways to debug and test programs.
- Learner recognizes common programming errors.

## 6. Define Modular Programming.

### **Learning objectives**

*What you will learn as you master the competency:*

- a. Use output parameters to return multiple results from a function.

- b. Create multiple calls to a function using input/output parameters.
- c. Define and locate scope of names in programming.
- d. Write formal output parameters as actual functions.
- e. Manipulate numeric data to create programs with multiple functions.
- f. Debug and test a program system.
- g. Review common programming errors.

**Performance Standards**

*Competence will be demonstrated:*

- o By successful completion of required program.
- o By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- o Learner uses output parameters to return multiple results from a function.
- o Learner can create multiple calls to a function using input/output parameters.
- o Learner can define and locate scope of names in programming.
- o Learner can write formal output parameters as actual functions.
- o Learner manipulates numeric data to create programs with multiple functions.
- o Learner debugs and tests a program system.
- o Learner reviews common programming errors.

**7. Define simple data types.**

**Learning objectives**

*What you will learn as you master the competency:*

- a. Differentiate the various numeric and data types.
- b. Understand the representation of character values.
- c. Associate numeric codes in categories using enumerated type.
- d. Illustrate methods for iteratively approximating a root of an equation.

**Performance Standards**

*Competence will be demonstrated:*

- o By successful completion of required program.
- o By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- o Learner can differentiate among the various numeric and data types.
- o Learner understands the representation of character values.
- o Learner can associate numeric codes in categories using enumerated type.
- o Learner can illustrate methods for iteratively approximating a root of an equation.

**8. Use arrays to store data.**

**Learning objectives**

*What you will learn as you master the competency:*

- a. Declare and reference arrays.
- b. Understand the distinction between an array subscript value and an array element value.

- c. Use for loops for sequential access.
- d. Use array elements as function arguments.
- e. Write functions that have arrays as arguments.
- f. Perform array searches and sorts.
- g. Use multidimensional arrays to represent multidimensional objects.
- h. Select array elements for processing.
- i. Define subscript range errors.

**Performance Standards**

*Competence will be demonstrated:*

- o By successful completion of required program.
- o By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- o Learner can declare and reference arrays.
- o Learner understands the distinction between an array subscript value and an array element value.
- o Learner uses for loops for sequential access.
- o Learner uses array elements as function arguments.
- o Learner can write functions that have arrays as arguments.
- o Learner can perform array searches and sorts.
- o Learner uses multidimensional arrays to represent multidimensional objects.
- o Learner selects array elements for processing.
- o Learner can define subscript range errors.

**9. Manipulate string data.**

**Learning objectives**

*What you will learn as you master the competency:*

- a. Review string basics.
- b. Create string library functions.
- c. Concatenate strings.
- d. Compare a portion of a string variable's contents to another string.
- e. Modify lists using array of pointers.
- f. Analyze and convert characters.
- g. Convert characters placed in a string.
- h. Create an editor to process text.
- i. Review common string programming errors.

**Performance Standards**

*Competence will be demonstrated:*

- o By successful completion of required program.
- o By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- o Learner reviews string basics.
- o Learner creates string library functions.

- Learner can concatenate strings.
- Learner compares a portion of a string variable's contents to another string.
- Learner can modify lists using array of pointers.
- Learner can analyze and convert characters.
- Learner converts characters placed in a string.
- Learner can create an editor to process text.
- Learner reviews the common string programming errors.

**10. Use Recursion as an alteration to iteration.**

***Learning objectives***

*What you will learn as you master the competency:*

- a. Define recursion.
- b. Trace a recursive function.
- c. Code recursive mathematical functions.
- d. Implement recursive functions with arrays and string parameters.
- e. Apply recursive functions to solve problems.
- f. Compare iterative and recursive functions.
- g. Solve recursive functioning errors.

***Performance Standards***

*Competence will be demonstrated:*

- By successful completion of required program.
- By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- Learner can define recursion.
- Learner can trace a recursive function.
- Learner implements recursive functions with arrays and string parameters.
- Learner can apply recursive functions to solve problems.
- Learner compares iterative and recursive functions.
- Learner solves recursive functioning errors.

**11. Define data types that represent structured collections of data pertaining to particular objects.**

***Learning objectives***

*What you will learn as you master the competency:*

- a. Define the components of a structured data object.
- b. Use structure data types as input and output parameters.
- c. Apply functions to return structured result values.
- d. Manipulate structure types to problem solve.
- e. Organize data using parallel arrays and arrays of structure.
- f. Use union types to interpret a data object in a variety of ways.
- g. Review common programming errors seen when using manipulated structure types.

### **Performance Standards**

*Competence will be demonstrated:*

- By successful completion of required program.
- By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- Learner can define the components of a structured data object.
- Learner uses structure data types as input and output parameters.
- Learner can apply functions to return structured result values.
- Learner manipulates structure types to problem solve.
- Learner organizes data using parallel arrays and arrays of structure.
- Learner uses union types to interpret a data object in a variety of ways.
- Learner reviews common programming errors seen when using manipulated structure types.

## **12. Compare the advantages and disadvantages of text and binary files.**

### **Learning objectives**

*What you will learn as you master the competency:*

- a. Review input/output files.
- b. Use binary files to store information.
- c. Write a program that searches a database.
- d. Review file programming errors.

### **Performance Standards**

*Competence will be demonstrated:*

- By successful completion of required program.
- By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- Learner reviews input/output files.
- Learner uses binary files to store information.
- Learner can write a program that searches a database.
- Learner reviews file programming errors.

## **13. Programming large software systems.**

### **Learning objectives**

*What you will learn as you master the competency:*

- a. Use abstraction to manage complexity.
- b. Create a header file.
- c. Create a library implementation file.
- d. Review the five storage classes of C.
- e. Modify functions for inclusion in a library.
- f. Create conditional compilations.
- g. Code arguments to function main.
- h. Define macros with parameters.

- i. Review common programming errors.

**Performance Standards**

*Competence will be demonstrated:*

- o By successful completion of required program.
- o By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- o Learner uses abstraction to manage complexity.
- o Learner creates a header file.
- o Learner creates a library implementation file.
- o Learner reviews the five storage classes of C.
- o Learner modifies functions for inclusion in a library.
- o Learner creates conditional compilations.
- o Learner can code arguments to function main.
- o Learner defines macros with parameters.
- o Learner reviews common programming errors.

**14. Define dynamic data structures.**

**Learning objectives**

*What you will learn as you master the competency:*

- a. Review pointers and their uses.
- b. Apply dynamic memory allocation.
- c. Define linked lists.
- d. Implement linked list operators using pointer variables.
- e. Represent a stack data structure with a linked list.
- f. Represent a queue with a linked list.
- g. Maintain ordered lists.
- h. Create a binary search tree.
- i. Review common programming errors.

**Performance Standards**

*Competence will be demonstrated:*

- o By successful completion of required program.
- o By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- o Learner reviews pointers and their uses.
- o Learner applies the dynamic memory allocation.
- o Learner defines linked lists.
- o Learner can implement linked list operators using pointer variables.
- o Learner represents a stack data structure with a linked list.
- o Learner represents a queue with a linked list.
- o Learner maintains an ordered list.
- o Learner creates a binary search tree.

- Learner reviews common programming errors.

## 15. Multiprocess using processes and threads.

### **Learning objectives**

*What you will learn as you master the competency:*

- Apply multitasking to operating systems.
- Creates processes.
- Define interprocess communications and pipes.
- Create a thread.
- Use threads to accomplish more than one task at a time.
- Review common programming errors with multithread programs.

### **Performance Standards**

*Competence will be demonstrated:*

- By successful completion of required program.
- By successful completion of final exam.

*Criteria-Performance will be satisfactory when:*

- Learner applies multitasking to operating systems.
- Learner creates processes.
- Learner defines interprocess communications and pipes.
- Learner creates a thread.
- Learner uses threads to accomplish more than one task at a time.
- Learner reviews common programming errors with multithreaded programs.

### **Types of Instruction**

Classroom Presentation

On Campus Laboratory

### **Grading Information**

#### **Grading Rationale**

Post Test 10%

Quizzes 30%

Programming Assignments 60%

#### **Grading Scale**

A	90-100%
B	80-89%
C	70-79%
D	60-69%
F	0-59%