Maui Community College
Course Outline

ORIGINAL

1.	Alpha and Number	ICS151C
	Course Title	Introduction to C Programming
	Credits	3
	Date of Outline	10/10/2003
2.	Course Description:	Introduces students to the C Programming Language, and an Integrated Development Environment (IDE). Develops structured programs using problem solving, algorithm development and programming concepts using a procedural language.
3.	Contact Hours/Type:	Three (3); Lecture/lab
4.	Prerequisites:	ICS 111 with at least a C; or consent
	Corequisites:	None
	Recommended Preparation:	None

Approved by _____ Date 3/14/07

5. General Course Objectives

   Provide a base understanding of the C Programming Language, an Integrated Development Environment (IDE), structured programming, problem solving, algorithm development and programming concepts. This course will focus on developing conceptual understanding followed by programmatic implementation of those concepts, and will provide students with the opportunity to improve critical thinking and problems solving skills.

6. Student Learning Outcomes

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

Introduction:

1. Explain briefly the history of programming languages.
2. Describe structured programming.
3. Explain the sequence, selection and repetition control structures.
4. Explain the problem solving process used to create a computer program.

Design Tools:

5. Write simple algorithms using sequence, selection and repetition.
6. Write simple algorithms using the following design tools:
   - Report, screen and record layouts
   - IPO charts
   - Hierarchy charts
   - Pseudocode and flowcharts
   - Test data, expected results and desk check tables.

Beginning Programming:

7. Use an Integrated Development Environment (IDE) to create, save, retrieve, edit, print, compile, build and execute a simple program using the C programming language.
8. Implement previously created algorithms in the C IDE.
9. Distinguish between a variable, a named constant and a literal constant.
10. Select an appropriate name, data type and an initial value for a memory location,
11. Type cast data.
12. Use an assignment statement to assign data to a variable.
13. Include arithmetic expressions in an expression.
14. Get input using predefined functions.
15. Create a console application.
16. Use built-in mathematical functions.
17. Format numeric output in a C program.
18. Send program output to a file.
19. Write a function prototype.
20. Create and invoke a programmer-defined function that accepts and returns values.

21. Understand a variable's scope and lifetime.
22. Pass information by value and reference to a programmer defined function.

Intermediate Programming:

23. Use the sequence, selection and repetition control structures in a C program.
24. Write code that uses comparison and logical operators.
25. Write a C program that use nested control structures.
26. Working with characters and strings, be able to:
    Control the case
    Determine the number of characters in a string
    Compare string contents
    Manipulate characters within strings
    Concatenate strings
    Compare strings.

Data Structures:

27. Open, close, read and write to a sequential access data file.
28. Declare, initialize, store and update data in an array.
29. Display the contents of an array.
30. Access an array element.
31. Search an array.
32. Compute the average of an array's contents.
33. Pass an array to a function.
34. Manipulate parallel arrays.
35. Manipulate a two-dimensional array.
36. Declare, initialize, store and update data in a structure.
37. Access structure members.
38. Pass a structure to a function.
39. Create an array of structures.
40. Convert parallel arrays to an array of structures.
41. Add and update data in an array of structures.
42. Search an array of structures.

7. Recommended Course Content and Approximate Time Spent on Each Topic

1-3 Weeks:	Introduction (1-4)
2-5 Weeks:	Design Tools (5-6)
2-5 Weeks:	Beginning Programming (7-22)
2-5 Weeks	Intermediate Programming (23-26)
3-5 Weeks	Data Structures (27-42)

8. Text and Materials, Reference Materials, Auxiliary Materials and Content

   An appropriate text(s) and materials will be chosen at the time the course is to be offered from those currently available in the field. Examples include:

   Texts:
   Computer Science: A Structured Approach Using C++,
   Forouzan and Gilberg, 2nd edition.

   An Introduction to Programming With C++,
   Zak, 2nd edition.

   Materials:
   Text(s) may be supplemented with:
   Articles and/or handouts prepared by the instructor

   Other:
   Appropriate films, videos or internet sites
   Television programs
   Guest speakers
   Other instructional aids

9. Recommended Course Requirements and Evaluation
   Specific course requirements are at the discretion of the instructor at the time the course is being offered. Suggested requirements might include, but are not limited to:

20-50%	Written or oral examinations
0-20%	In-class exercises
30-50%	Programming assignments
0-25%	Homework assignments
0-20%	Quizzes
0-20%	Projects or research (written reports and/or oral class presentations)
0-8%	Attendance and/or class participation

10. Methods of Instruction
    Instructional methods vary considerable with instructors and specific instructional methods will be at the discretion of the instructor teaching the course. Suggested techniques might include, but are not limited to:

    Lecture (PowerPoint or similar)
    Problem solving and design exercises
    Hands-on laboratory exercises
    Group or individual projects
    Class discussions or guest lectures
    Audio, visual or presentations involving the internet

Student class presentations
Field trips
Other contemporary learning techniques (e.g., Service Learning, Co-op, School-to-Work, self-paced, etc.)