CONESTOGA

COURSE OUTLINE

Course	C Programming (2014-2015)
Code / Version	PROG1347 (100)
Total Hours	75
Credits	5

PreRequisite(s)

CoRequisite(s)

Course Description

This course will introduce software design and implementation using the C language. Topics include: algorithm design, modular code design, programming style, functions, arrays, pointers, strings, data structures, file I/O and operating system function calls. An emphasis will be placed on proper design to produce maintainable software.

PLAR Eligible: Yes

Course Outcomes

Successful completion of this course will enable the student to:

1. Design computer programs from a written problem description.
2. Use variables using built-in data types in the C language.
3. Use operators to manipulate variables.
4. Use flow of control programming language constructs, including decisions, loops and switches.
5. Use existing functions.
6. Create programmer-defined functions.
7. Use arrays, both single and multi-dimensional.
8. Use pointers (addresses) and explain their impact on the efficiency of a C program.
9. Use programmer-defined data structures, enumerations and type definitions.
10. Use dynamic memory capabilities in C.
11. Use file manipulation, both general and operating system specific.
12. Use command-line arguments.
13. Identify program maintenance issues.
14. Use proper C programming style.
15. Debug programs.

Unit Outcomes

Successful completion of the following units will enable the student to:

1.0 Program Design

- 1.1 Analyze problems.
- 1.2 Design computer code from written problems.
- 1.3 Troubleshoot and debug problems.

2.0 Introduction to C

- 2.1 Describe the development cycle for C programmers.
- 2.2 Use proper program structure.

2.3 Use header files and main () function.

2.4 Use proper program style.

2.5 Use comments appropriately, according to industry-accepted criteria.

3.0 Data Types and Variables

3.1 Use C built in data types.

3.2 Declare and initialize variables.

3.3 Use formatting codes to output data types.

3.4 Define constants.

3.5 Use type conversion.

3.6 Use storage classes and scope properly.

3.7 Use void.

4.0 Operators

4.1 Use arithmetic operators.

4.2 Use logical operators.

4.3 Use other operators.

5.0 Flow of Control

5.1 Create conditional statements.

5.2 Use if statements and if/else statements.

5.3 Use while, do-while and for statements.

5.4 Use the switch statement.

5.5 Use break and continue statements.

6.0 Functions

6.1 Use pre-existing functions.

6.2 Create new functions.

6.3 Use parameters and return values.

6.4 Pass parameters by reference.

6.5 Describe and handle local variable issues.

6.6 Use recursion.

6.7 Use prototypes.

6.8 Use input and output functions.

7.0 Arrays

7.1 Explain the justification for arrays.

7.2 Declare and use arrays.

7.3 Explain the base address of arrays and array offsets.

7.4 Use multi-dimensional arrays.

7.5 Initialize array data.

7.6 Pass arrays as parameters.

8.0 Strings

8.1 Define strings as arrays of characters.

8.2 Use string functions.

8.3 Explain common string problems.

9.0 Pointers

- 9.1 Explain pointers and addresses.
- 9.2 Compare arrays and pointers.
- 9.3 Use indirection and address-of operators.
- 9.4 Explain and use pointer arithmetic.
- 9.5 Use pointers to obtain multiple return values from a function.
- 9.6 Do string handling using pointers.
- 9.7 Explain and use pointers to void.

10.0 Command-line Arguments

- 10.1 Use argc and argv.

11.0 Programmer Defined Data Types

- 11.1 Use structs.
- 11.2 Declare enums.
- 11.3 Use typedef.
- 11.4 Pass aggregate data types to functions.

12.0 Dynamic Memory

- 12.1 Compare and explain static versus dynamic allocation.
- 12.2 Use standard memory allocation functions: malloc, realloc, free.
- 12.3 Explain uses for dynamic memory allocation.
- 12.4 Use NULL pointers.
- 12.5 Use error checking when using dynamic memory allocation.

13.0 File Handling

- 13.1 Use file I/O (text mode).
- 13.2 Use file I/O (binary mode).
- 13.3 Use operating system level file handling.
- 13.4 Use operating system file manipulation (rename, delete, etc.).

14.0 Application Frameworks and Program Maintenance

- 14.1 Use multiple file C projects.
- 14.2 Explain the extern keyword.
- 14.3 Create programmer-defined header files.
- 14.4 Debug code using a source-level debugger.
- 14.5 Explain the difference between developing with and without an IDE.

Required Student Resources

Kelley and Pohl. A Book On C. Addison-Wesley Professional.
C Programming (Software Engineering Technician/Technology) Course Notes.

Optional Student Resources

Farrell, J. Just Enough Programming Logic and Design (2009). Boston: Course Technology.
Gookin, D. C All-in-One Desk Reference for Dummies (2004). Toronto: John Wiley & Sons.

CONESTOGA

COURSE OUTLINE

Course	C Programming (2014-2015)
Code / Version	PROG1347 (100)

Robertson, L.A. Simple Program Design (any). Toronto: Thomson Learning.

Evaluation

The minimum passing grade for this course is 55 (D).

In order to successfully complete this course, the student is required to meet the following evaluation criteria:

Early-term Exam	10.00
Midterm Exam	20.00
Final Exam	30.00
Assignments	15.00
Course Project	15.00
Quizzes	3.00
Discussions	3.00
Mini-Assignments	4.00
	100.00 %

Other

Conestoga College is committed to providing academic accommodations for students with documented disabilities. Please contact the Accessibility Services Office.

Prepared By	Carlo Sgro
School	Information Technology
Date	2014-08-26