

Copyright

ND Barnette, WD McQuain,

© 1995-1999

MA Keenan



All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, transmitted in any form by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the authors.

Information in this document is subject to change without notice and does not represent a commitment on the part of the authors. No warranties, either express or implied, regarding the use of this document are made by the authors. The authors shall not be liable in any event for incidental or consequential damages, losses, costs, charges, claims, demands or claim for lost profits or grades, fees, or expenses of any nature or kind, in connection with, arising out of, the furnishing, performance, or use of this document. The software systems described in this document are copyrighted by their authors or companies and are subject to their own license or nondisclosure agreements.

Computer Science Dept. Compiler Policy

The only supported compiler for this course is Microsoft Visual C/C++ (version 6.0 or higher) compiler. A student may choose to use a different ANSI standard compiler if they wish. However, it is the student's responsibility to ensure that his/her programs compile and run under the Microsoft Visual C/C++ environment under MS Windows NT. GTAs will only be supporting the Microsoft Visual C/C++ compiler. This means that students who choose to use other compilers cannot expect the GTAs to help them with specific compiler problems, (e.g, interface questions, compiler messages, warnings or errors).

- 1 Introduction
 - The Programming Process
 - Polya's 4 Step Process
 - Understand the problem
 - Devise a plan
 - Implement the plan
 - Test the plan
 - Algorithm
 - Language Levels
 - Program Translation
- 2 Program Design
 - Top Down Design
 - Design Specification
 - Outline Design Example
 - Pseudocode Design Example
 - Top-Down Design Example
 - Program Proofs
 - Program Documentation
 - Modular Decomposition
 - Structure Chart
 - Procedural Abstraction
- 3 Fundamentals
 - Programming Errors
 - Program Development Process
 - Background
 - Skeleton Program
 - Syntax and Semantics
 - Identifiers
 - Directives
 - The Simple Data Types
 - Constants
 - Standard Constants
 - Statements
 - Variables
 - Assignment
 - Increment/Decrement

- 3 Fundamentals (continued)
 - Arithmetic Expressions
 - Operator Hierarchy
 - Assignment and Expressions
 - Compound Statement
 - Standard Functions
 - Function Invocation
 - Function Communication
- 4 Input / Output
 - Output
 - Output Examples
 - Input
 - Input Examples
 - Extraction Operator
 - C++ Streams are Classes
 - Whitespace Input
 - get() Member Function
 - ignore() Member Function
 - Insertion Operator
 - Interactive I/O
 - Streams for File I/O
 - open() Member Function
 - File Names and Paths
 - close() Member Function
 - Formatting Output
 - Numeric Output Manipulators
 - Padding and Justification Manipulators
 - Manipulators Example
 - Reading to Input Failure
 - Failure-Controlled Input Example
 - Incorrect Failure-Controlled Input
 - eof() Member Function
 - peek() Member Function
 - putback() Member Function
 - fail() Member Function
 - Operating System I/O Redirection
 - File Output Redirection

- 5 Booleans and Selection
 - Conditions & Boolean Expressions
 - Boolean Variables in C++
 - C++ versus C Booleans
 - Forms of Boolean Expressions
 - Boolean Expression Examples
 - Truth Tables
 - Short Circuiting
 - C++ Operator Hierarchy
 - C-style Boolean Examples
 - Control Flow
 - Selection: if...else
 - Nested if Statements
 - Nested if Examples
 - Dangling else
 - Execution Trace Program
 - Execution Trace
 - Switch Statement
 - Switch Example
 - Switch Limitations
 - C Language assert() Function
 - Assert Debugging Control
- 6 Iteration
 - While Loop
 - Count Controlled Loop
 - Event Controlled Loop
 - EOF Controlled Loop
 - End-of-line Controlled Loop
 - Sentinel-Controlled Loop
 - Loop Design Considerations
 - For Loop
 - Do-While Loop
 - While vs Do-While
 - Break Statement
 - Continue Statement

7 Functions

- Functions
- Function Structure
- Identifier Scope
- Scope Example
- Function Prototypes
- Parameter Communication
- C++ Parameter Passing Mechanisms
- Choosing a Parameter Passing Mechanism
- Parameter Lists
- Parameter Example 1
- Parameter Example 2
- Function Names
- The return Statement
- void versus return
- Use of void
- Use of return
- Multiple Reference Communication
- Function Prototype Scope
- Prototype Scope Example
- File Scope Prototypes
- Prototype Scope Problems

8 Arrays

- Structured Types
- Definitions
- Array Declaration
- Indices
- Accessing
- Aggregate Operations
- Implementing Aggregate Operations
- Array Usage
- More on Array Indices
- Array Initialization at Declaration
- Array Initialization via a Loop
- Array Elements as Parameters
- Entire Arrays as Parameters
- Arrays as Parameters

8 Arrays

Cautionary Note

C-style Character Strings

Character Strings: Input

Character Strings: Output

strcat() function

strlen() function

strcpy() function

strcmp() function

Character Strings: input (get)

Character Strings: input (getline)

Parallel Arrays

Array of Arrays

Multi-Dimensional Arrays

Multi-Dimensional Arrays as Parameters

9 Types

Data Types

Enumerated Types

Enumerated Types Example

Type Coercion

Compound Operators

Binary Shift Operators

Bitwise Operators

Cast Operators

Other Operators

Typedef

10 Structures

Structures

Referencing Structure Fields

Aggregate Operations

Structure Field Access

Structure Manipulation

Structure Comparison

Structure Initialization

Array of Structures

Hierarchical Structures

Unions

11 Searching and Sorting

- Simple Searching
- Linear Searching
- Binary Search
- Binary Search Code
- Binary Search Trace
- Cost of Searching
- Sorting
- Decision Trees
- Bubble Sort Algorithm
- Bubble Sort Code
- Bubble Sort Trace
- Selection Sort Algorithm
- Straight Selection Sort
- Selection Sort Trace
- Cost of Sorting

12 C++ Strings

- String Variables
- String Initialization
- String Output
- String Input: extraction
- String Input: getline()
- String Input: Example
- Strings are C++ Classes
- Length of a String
- Testing if a String is Empty
- String Concatenation
- Comparing Strings for Equality
- Lexicographic Comparison
- Lexicographic Comparison Example
- Accessing String Elements
- Inserting one String into Another
- Inserting a Part of one String into Another
- Extracting a Substring
- Erasing a Substring
- Replacing a Substring
- Searching for a Substring

- 12 C++ Strings (continued)
 - Just for Fun
 - And That's Just the Beginning
- 13 Computer Arithmetic
 - Independent Representation
 - Arithmetic Errors
 - Considerations
 - Floats as LCVs
 - Float Underflow
 - Machine Epsilon
- 14 Recursion
 - Simple Recursion
 - Recursive Array Summation
 - Recursive Array Summation Trace
 - Recursive Selection Sort
 - Recursive Selection Sort Trace
- 15 Binary Numbers (*supplemental material*)

Appendices

- 1 MS Visual C++ Introduction
- 2 MS Visual C++ Debugger
- 3 PseudoCode
- 4 Simple Payroll Design Example
- 5 Payroll Program Examples
- 6 Friends
- 7 Programming Correctness and Sins
- 8 Elements of Programming Style
- 9 Standard C Input/Output
- 10 Course Policies
- 11 Trademarks
- 12 Submitting to the Automated Grader