1. Week 1 - Programming Review

1.1 Programming (general)

1.2 Problem solving

1.3 Why programming

1.4 Computer tour

2. Week 2 - Introduction to C++

2.1 Programming basics

2.2 Comments and whitespace

2.3 Errors and warnings

2.4 Computers and programs (general)

2.5 Language history

2.6 Why whitespace matters

3. Week 3 - Variables / Assignments

3.1 Variables and assignments (general)

3.2 Variables (int)

3.3 Identifiers

3.4 Arithmetic expressions (general)

3.5 Arithmetic expressions (int)

3.6 Example: Health data

3.7 Floating-point numbers (double)

3.8 Scientific notation for floating-point literals

3.9 Using math functions

3.10 Integer division and modulo

3.11 Type conversions

3.12 Integer overflow

3.13 Numeric data types

3.14 Unsigned

3.15 Debugging

4. Week 4 - Variables Continued

4.1 Random numbers

4.2 Auto (since C++11)

4.3 Constant variables

4.4 Binary

4.5 Characters

4.6 Strings

4.7 Style guidelines

5. Week 5 - Branches

5.1 If-else branches (general)

5.2 Detecting equal values with branches

5.3 Detecting ranges with branches (general)

5.4 Detecting ranges with branches

5.5 Detecting ranges using logical operators

5.6 Detecting ranges with gaps

5.7 Detecting multiple features with branches

5.8 Common branching errors

5.9 Example: Toll calculation

5.10 Order of evaluation

5.11 Switch statements

5.12 Boolean data type

5.13 String comparisons

5.14 String access operations

5.15 Character operations

5.16 More string operations

5.17 Conditional expressions

5.18 Floating-point comparison

5.19 Short circuit evaluation

6. Week 6 - Loops

6.1 Loops (general)

6.2 Developing programs incrementally

6.3 Enumerations

7. Week 7 - Loops Continued

7.1 For loops

7.2 More for loop examples

7.3 Loops and strings

7.4 Nested loops

7.5 Variable name scope

7.6 While loops

7.7 More while examples

7.8 Break and continue

8. Week 8 - Arrays

8.1 Array/vector concept (general)

8.2 Array/vector iteration drill

8.3 Swapping two variables (General)

8.4 Debugging example: Reversing a vector

8.5 Two-dimensional arrays

8.6 Char arrays / C strings

8.7 C-String library functions

8.8 Char library functions: ctype

9. Week 9 - Vector

9.1 Vectors

9.2 Iterating through vectors

9.3 Multiple vectors

9.4 Vector resize

9.5 Vector push\_back

9.6 Loop-modifying or copying/comparing vectors

9.7 Arrays vs. vectors

10. Week 10 - User-Defined Functions

10.1 User-defined function basics

10.2 Print functions

10.3 Reasons for defining functions

10.4 Writing mathematical functions

10.5 Functions with branches

10.6 Functions with loops

10.7 How functions work

10.8 Functions: Common errors

10.9 Pass by reference

10.10 Functions with string/vector parameters

10.11 Functions with C string parameters

10.12 Scope of variable/function definitions

11. Week 11 - User Define Functions Continued

11.1 Unit testing (functions)

11.2 Default parameter values

11.3 Function name overloading

11.4 Parameter error checking

11.5 Preprocessor and include

11.6 Separate files

12. Week 12 - Objects and Classes

12.1 Objects: Introduction

12.2 Using a class

12.3 Defining a class

12.4 Inline member functions

12.5 Mutators, accessors, and private helpers

12.6 Initialization and constructors

12.7 Classes and vectors/classes

12.8 Separate files for classes

12.9 Choosing classes to create

12.10 Unit testing (classes)

12.11 Constructor overloading

12.12 Constructor initializer lists

12.13 The 'this' implicit parameter

12.14 Operator overloading

12.15 Overloading comparison operators

12.16 Vector ADT

12.17 Namespaces

12.18 Static data members and functions

12.27 Destructors

13. Week 13 - pointers

13.1 Why pointers?

13.2 Pointer basics

13.3 String functions with pointers

14. Week 14 - Memory, Libraries and Command Line Arguments

14.1 Command-line arguments

14.2 Memory regions: Heap/Stack

14.3 Memory leaks