Lecture 0: Introduction
Possibly make a mention to security, but there really isn’t any place during this lecture where security concerns should be discussed.

Lecture 1: Simple C Programs, structured programming
Besides possibly mentioning security implications of the errors caused by failing to use cast operator properly there aren’t really any places in this lecture where security would be appropriate.

Lecture 2: Control Structures, booleans, cautions
Introduction to data validation should occur (and does occur) in this lecture.

Lecture 3: Functions
No real opportunities to incorporate “defensive programming” into this lecture besides data validation.

Lecture 4: Tracing Function Calls
Mention the fact that the C compiler WILL LET you do things that are technically insecure and dangerous. Just because it LETS you write bad/insecure code, doesn’t validate the use of that code.

Lecture 5: Purposes of Functions and Separate Compilation
Stress the importance of testing and how modular programming can lead to less testing and more reliable, secure code through code reuse.
Discuss the dangers of ignoring compiler warnings.

Lecture 6: Top-Down Design / Designing A Project
Discuss the importance of keeping security in mind when designing programs.
Mention libraries that might be helpful.

Lecture 7:
Lecture 7 was actually a link to the first project.
This project should grade students’ use of data validation techniques. Design for this project should also include a brief discussion of security implications.

Lecture 8: Arrays
Discuss the dangers of referencing data outside the bounds of an array. Discuss the basic concept of a buffer overflow. Discuss segmentation faults.
In the second project students should also be graded for avoidance of these errors. The design document should discuss the implications of using arrays and the possible dangers.

Lecture 9: Passing Arrays / Sorting & Searching
Carry the discussions from lecture 8 over into this lecture since they are basically dealing with the same concepts.
Lecture 10: Passing Arrays / Sorting & Searching
No real security issues are discussed in this lecture.

Lecture 11: Structures & Arrays of structs
Possibly discuss the fact that structures don’t allow for any protection of data and let the students know that C++ will introduce classes.

Lecture 12: Structures & Arrays of structs
Carry over the discussions from the last lecture and ask students why it could be beneficial to protect data stored within a structure like datatype.

Lecture 13: Pointers
Introduce the concept of pointers, discuss initializing pointers to NULL, and reiterate the dangers of segfaults.

Lecture 14: Pointers Revisited
Show how pointers are just like arrays and therefore subject to the same vulnerabilities. Make sure students discuss any security implications that pointers might cause in their project.

Lecture 15: Pointer Applications
Discuss how malloc works and how any memory allocated by malloc contains whatever data previously existed at that memory address. Discuss the implications of this.

Lecture 16: Pointers to Pointers, Command-Line Args
Stress the importance of data validation when dealing with command line arguments. Explain that pointers to pointers and arrays of pointers can have the same security problems that traditional arrays and pointers can have.

Lecture 17: Characters & Strings
Discuss some of the potential insecurities of the string.h library.

Lecture 18: Streams, I/O and stdio.h
Discuss the dangers of gets() and other insecure functions dealing with strings and characters. Make sure students don’t use insecure functions in project.

Lecture 19: Design - Coupling & Cohesion
There really aren’t any opportunities to discuss defensive programming in this lecture.

Lecture 20: Pointers to Structs and Self-Referencing Structs
This lecture introduces self referencing structures in preparation for teaching linked lists. Make sure to stress the importance of initializing pointers to NULL after declaration.

Lecture 21: Debugging Programs
There aren’t really any opportunities to discuss defensive programming in this lecture.
Lecture 22: Abstract Data Types
There aren’t really any opportunities to discuss defensive programming in this lecture except for possibly mentioning classes and how the students will learn about them in 202.

Lecture 23: Linked Lists
Make sure students understand to initialize pointers to NULL.

Lecture 24: Keeping Lists in Sorted Order
There aren’t really any opportunities to discuss defensive programming in this lecture.

Lecture 25: Stacks & Queues
Possibly bring up the concept of a heap and a stack and let students know that they will learn more about them in future classes.

Lecture 26: Memory Management
Possibly discuss the dangers of printing un-initialized memory space.