

## University of Washington Computer Programming I

---

### Lecture 21: Course Wrap-up and Look Ahead

© 2000 UW CSE

## Where Do You Go From Here?

---

At the Seattle campus of UW, the next course is **CSE143**, "Computer Programming II"  
A direct continuation of UW's **CSE142**  
**CSE143** introduces the C++ programming language

At universities on the semester system, the second-semester course is usually called "Data Structures"

## What is CSE143 like?

---

5 credit hours; 4-5 projects  
15-30 files in later projects (you don't write all)  
Students say:  
"intense", "time-consuming"  
Language: C++  
Initially just like C  
Learn programming concepts to support writing larger programs  
Eventually objects + classes ("object-oriented")

## What is Covered in a 2<sup>nd</sup> Course?

---

Lots of programming practice  
but less class discussion of it  
Data structures  
many involving pointers  
abstract data types (ADTs)  
Algorithms  
many recursive  
Problem-solving & design  
Foundation for later CS courses

## Going on in C

---

There is much about the C language we haven't covered  
You have a foundation now to master more advanced C programming features  
Learning more C may or may not be useful  
*The advanced features won't necessarily help learn other languages*

## Going on to C++

---

C++ was developed as an extension to C  
Practically everything we have learned in the course can be used in C++  
The syntax is completely identical in almost all cases  
However, C++ includes some important new concepts which are not part of C  
*These lead to an approach called "object-oriented programming"*

## Going on to Java

---

Java in many ways an improved C++  
Much of Java syntax is very close to C  
*if, while, for*  
*Expressions*

Java is not as close to C as C++ is  
There are also new, object-oriented features

## Programming Concepts

---

We used C, but the concepts go beyond the C language  
Variables; data types; values  
Conditions; conditional execution  
Loops; recursion  
Functions; parameters (including pointers); call/return  
Data structuring: arrays; structs; combinations  
Input/Output

## Beyond Programming...

---

Compiler concepts  
syntax vs semantics; compilation steps; libraries; debugging  
What is a computer?  
Visualize memory, CPU, I/O operation  
Instruction execution, data movement  
Problem solving  
Abstraction  
Functional decomposition, Data decomposition  
Algorithms

## Building and Understanding Software

---

Software give computer its personality  
Computers are proliferating  
I.e., software is.  
Programs are complex artifacts  
Compare to a bridge or a novel  
What's the effect of a small error?  
Taming complexity  
Analysis, design, testing, communicating

## Is Programming **Hard** or is Programming **Fun**?

---

Programming is **hard**...  
NOW YOU KNOW THIS FIRST-HAND!

Programming is also **fun**...  
NOW YOU KNOW THIS FIRST-HAND, TOO!  
(I hope...)