CSE 142
Computer Programming I

UW Catalog Description:
Basic programming-in-the-small abilities and concepts. Highlights include procedural and functional abstraction with simple built-in data type manipulation. Basic abilities of writing, executing and debugging programs.

C Programming Language
The course is taught using the C Programming Language

Emphasis of course on fundamental concepts which are language independent.

C and C++ are widely used but do present hurdles for novice programmers

Are you Ready for this Course?
You should have the equivalent of a U.S. high school diploma
Four years of high school math through Algebra II
Some background in science
Good oral and written English fluency

You do not need any previous background in programming

You should have some basic fluency as a computer user

Is This Course for You?
If you have never studied programming: YES

If you studied programming once but never used it: YES

If you’ve learned a little HTML or Visual Basic or SQL or something else that seems like it might be programming: YES

If you really want to learn C++ or Java eventually but don’t have access to a C++ or Java course: YES
**Is This Course for You?**

If you already know another programming language and just want to learn C: **NO**

If you have studied C before and want to deepen your knowledge: **NO**

If you expect that this one course will make you a programmer so you can get a job: **NO**

If your primary goal is to become a better computer user, not a programmer: Probably **NO**

---

**So What is Programming Like?**

It’s really hard to describe!

Many similarities to solving “word problems” in math

- Translate a problem description into a formal solution
- Symbol manipulation an integral part

Some people describe it as “puzzle solving”

A mix of high-level creativity and low-level picky details

---

**Stages of Problem Solving**

Ultimate goal: use a computer to solve a problem

Typical stages of building a solution:

- Clearly **specify** the problem
- Analyze the problem
- Design an **algorithm** to solve the problem
- **Implement** the algorithm (write the program)
- **Test** and verify the completed program
- Maintain and update the program

---

**Focus**

All stages are important

In this course, we ignore none of them

But we focus on:

- Algorithm development
- Writing a program to implement the algorithm

---

**CSE 142 at the University of Washington**

One quarter (10 week) course

Course Organization (when taught on-campus at UW):

- Lectures 3 times a week
- Quiz section once a week
- 5 Programming projects
  - Done by students outside of class, on their own time
- Two midterm exams
- Final exam

---

**Using the Video Lectures**

Not everything on the slides will be read aloud by the speaker

Not everything the speaker says will be on the slides

You need both! And then some...
Beyond the Video Lectures

You won’t learn programming by watching the lectures as you might watch a TV program.

What you need in addition is:
- Access to the slides (printed or on the web: www.online.washington.edu)
- The ability to ask questions and get them answered
- A textbook you can use for details and examples not in the lectures
- And most important: hands-on practice.

The Importance of Practice

You wouldn’t expect to learn to play the guitar just by watching a TV series on it. There is no substitute for practice. And no one can do it for you.

The same holds for learning to program.

Tips for Success

Take the material in order.
- With rare exceptions: you can’t skip any lecture.
- You can’t take material out of order.

Master each topic before continuing to the next.
- Seek help if you get behind.

Practice, practice, practice!

Homework Can Be Fun

Examples from previous quarters...

Resources

UW CSE 142 Text
“Problem Solving and Program Design in C” - Hanly and Koffman, 3rd Edition, Addison-Wesley

Course Packet
- Slides and reference material

Course Web
- www.online.cs.washington.edu