Top Down Design

Methodology for writing larger programs

Step 1: Divide problem into smaller, easy-to-solve subproblems

   Compile list of features

   Sketch high-level algorithm on paper

   Sketch program (in code) using stubs (e.g. placeholder functions)

Step 2: Implement program bottom-up

   Use incremental development

   Refine design in step 1 and re-stub as necessary
Goals of TDD

Good design makes your program

- easy to build and test incrementally
- easy to debug
- well-organized
- easy for a human to read
- resistant to bugs

Rules of thumb:

NO CUT AND PASTED CODE BLOCKS!

Functions should do a single, clearly defined task

Algorithms should be clear from function and variable names
Top-down design - Analogies

Approach is the same as any you would take with a large project:

- Applying to schools
- Organizing an event
- Building a piece of furniture
- Writing a paper
TDD Example - checkbook

Step 1: List features

- Keep track of current balance
- Allow user to make deposits
- Allow user to make withdrawals
- Prevent user from withdrawing more than they have
- Print summary with current balance
- Press ‘q’ to quit
TDD Example - checkbook

Step 2: Sketch high-level algorithm on paper

Goal: subdivide program into small, easy steps

Get the starting balance from the user
while not timeToQuit:
   Ask the user what they want to do (withdraw, deposit, quit)
   if withdraw:
      withdraw
   elif deposit:
      deposit
   elif quit:
      timeToQuit = True
   else:
      Report an unrecognized command

This is a small example, so we have only two functions we need to stub: withdraw() and deposit()
TDD Example - checkbook

Step 3: Sketch program using stubs

NOTE: There are multiple good potential designs (but watch out because also many bad designs that will make your life miserable!)

NOTE: This program runs!
TDD Example - checkbook

Step 3: Sketch program using stubs

NOTE: There are multiple good potential designs (but watch out because also many bad designs that will make your life miserable!)

NOTE: This program runs!
Code stubs - best practices

Your program with stubs should still run

Your stubs should have comments describing their function

Your stubs should have the arguments and return type that you expect it to use

All the stubs you define should be used somewhere in your program
TDD Example - checkbook

Bottom-up Implementation

Implement and test each stub one at a time! In class, we started with deposit and then implemented withdrawal