How to develop a program

Michael Ernst

CSE 140

University of Washington

Program development methodology: English first, then Python

1. Define the problem
2. Decide upon an algorithm
3. Translate it into code

Try to do these steps in order

Program development methodology: English first, then Python

1. **Define the problem**
    1. Write an English description of the input and output
    2. Do not give details about how you will compute the input and output
    3. Create test cases
        - Input *and* expected output
2. Decide upon an algorithm
3. Translate it into code

Try to do these steps in order

Program development methodology: English first, then Python

1. Define the problem
2. **Decide upon an algorithm**
    1. Implement it in English
        - Write the recipe or step-by-step instructions
    2. Test it using paper and pencil
        - Use small but not trivial test cases
        - Play computer, animating the algorithm
        - Be introspective
            - Notice what you really do
            - May be more or less than what you wrote down
            - Make the algorithm more precise
3. Translate it into code

Try to do these steps in order

Program development methodology: English first, then Python

1. Define the problem
2. Decide upon an algorithm
3. **Translate it into code**
    1. Implement it in Python
        - Decompose it into logical units (functions)
        - For each function:
            - Name it (important and difficult!)
            - Write its documentation string (its specification)
            - Write tests
            - Write its code
            - Test it
    2. Run the system test

Try to do these steps in order

Program development methodology: English first, then Python

1. Define the problem
2. Decide upon an algorithm
3. Translate it into code

Try to do these steps in order

- It's OK (even common) to back up to a previous step when you notice a problem
- You are incrementally learning about the problem, the algorithm, and the code
- "Iterative development"

The *Wishful Thinking* approach to implementing a function

- If you are not sure how to implement one part of your function, define a helper function that does that task
  - "I wish I knew how to do task X"
  - Give it a name and assume that it works
  - Complete the implementation of your function
  - Later, implement the helper function
  - The helper function should have a simpler/smaller task
- Can you test the original function?
  - Yes, by using a stub for the helper function
  - Often a lookup table: works for only 5 inputs, crashes otherwise