Loop Design and Functional Decomposition:

Days between Dates

CIS 210 week 3 project

Days between Dates

CIS 210 week 3 project

Attacking a more complex problem

Think first. Design on paper. Maybe code a few snippets to experiment, then set them aside.


Days between Dates

Week 3 project:

Given input

Begin date: Month, day, year
End date: Month, day

Determine how many days from begin date to (next occurrence of) end date

ex: How many days until summer break?

From October 6

To June 15

Simplifying assumption: Start month ≠ End month (temporary; we’ll have to relax the assumption later)
**First cut at pseudocode:**

- total days = remainder of start month
- cur month = next month
- while cur month ≠ end month:
  - days += days in cur month
  - cur month = next month
- days += beginning of end month

**Function:**

- days in month

What are the arguments?

(what do you need to calculate it?)

Will it need to call other functions?
First cut at pseudocode:

total days = remainder of start month  
cur month = next month  
while cur month ≠ end month:  
days += days in cur month  
cur month = next month  
days += beginning of end month

Function: next month

next_month( ?? )

What are the arguments?  
What about next_month( December )

Maybe it's really next month and year?

Python tuples (Perkovic pg. 182)

>>> mmyy = (12, 2012)  
>>> mm, yy = mmyy  
>>> print(mm)  
12  
>>> next_mmyy = (1, yy+1)  
>>> print(next_mmyy)  
(1, 2013)

Python tuples (Perkovic pg. 182)

>>> mmyy = (12, 2012)  
>>> mm, yy = mmyy  
>>> print(mm)  
12  
>>> next_mmyy = (1, yy+1)  
>>> print(next_mmyy)  
(1, 2013)

next month function can take a tuple and return a tuple; simpler than using a list
What about that simplifying assumption?

We assumed start month ≠ end month
Time to relax that assumption ... what do we need to do?
Two cases to consider:

Summary: Attacking the date problem

1. Design the logic first, on paper
2. Then write pseudocode, identifying functions to simplify the code
3. Then write code
   Piece by piece. Write, test, debug, repeat.