CS 1110
Lecture 18: While loops

Announcements

Prelim 2 conflicts
If you have a conflict you need to submit the information in CMS. We need a little more information than for Prelim 1—please see the Exams page of the CS110 website.

Recall: For Loops

# Print contents of seq
x = seq[0]
print x
x = seq[1]
print x
...
x = seq[len(seq)-1]
print x

The for-loop:
for x in seq:
|     print x

• Key Concepts
  • loop sequence: seq
  • loop variable: x
  • body: print x
  • Also called repetend

Iteration: Doing things repeatedly

1. Process each item in a sequence
   • Compute aggregate statistics for a dataset, such as the mean, median, standard deviation, etc.
   • Send everyone in a Facebook group an appointment time
2. Perform n trials or get n samples
   • Draw n cards to make a poker hand
   • Run a protein-folding simulation for $10^6$ time steps
3. Do something an unknown number of times
   • CUAUV team, vehicle keeps moving until reached its goal

Beyond Sequences: The while-loop

while <condition>:
  statement 1
  ...
  statement n

• Relationship to for loop
  • Broader notion of “still stuff to do”
  • Must ensure condition eventually becomes false
  • You explicitly manage what changes per iteration

while Versus for

# process range b..c-1
for k in range(b,c):
  process k

Must remember to increment

# process range b..c+1
for k in range(b,c+1):
  process k

Note on Ranges

• m..n is a range containing n+1-m values
  • 2..5 contains 2, 3, 4, 5.
  • 2..4 contains 2, 3, 4.
  • 2..3 contains 2, 3.
  • 2..2 contains 2.
  • 2..1 contains ???

What does 2..1 contain?

A: nothing
B: 2, 1
C: 1
D: 2
E: something else

# process range b..c-1
for k in range(b,c):
  k = b
while k < c:
  process k
  k = k+1

# process range b..c
for k in range(b,c+1):
  k = b
while k <= c:
  process k
  k = k+1
Note on Ranges

- \( m..n \) is a range containing \( n+1-m \) values
  - \( 2..5 \) contains \( 2, 3, 4, 5 \). Contains \( 5+1 - 2 = 4 \) values
  - \( 2..4 \) contains \( 2, 3, 4 \). Contains \( 4+1 - 2 = 3 \) values
  - \( 2..3 \) contains \( 2, 3 \). Contains \( 3+1 - 2 = 2 \) values
  - \( 2..2 \) contains \( 2 \). Contains \( 2+1 - 2 = 1 \) values
  - \( 2..1 \) contains ???
- The notation \( m..n \), always implies that \( m \leq n+1 \)
  - So you can assume that even if we do not say it
  - If \( m = n+1 \), the range has 0 values

while Versus for

**Have to know in advance where to stop**

### # Table of squares to \( N \)

```python
n = floor(sqrt(N)) + 1
for k in range(n):
    seq[k] = k*k
```

### # table of squares to \( N \)

```python
n = floor(sqrt(N)) + 1
for k in range(n):
    seq[k] = k*k
```

- \( k = 0 \)
- \( k \times k < N \):
  - \( seq[k] = k*k \)
  - \( k = k+1 \)

**while** is more flexible, but
is **tricker** to use

Patterns for Processing Integers

**while** Versus **for**

### While-Loops and Flow

**Output:**

```
Before while
count = 0
i = 0
while i < 3:
    print 'Start loop '+'\'+str(1)
    count = count + 1
    i = i + 1
print 'End loop'
print 'After while'
```

```
Before while
count = 0
i = 0
while i < 3:
    print 'Start loop '+'\'+str(1)
    count = count + 1
    i = i + 1
print 'End loop'
print 'After while'
```