Using Pseudocode

You mean you can program in English?!

Dr. Mattox Beckman

University of Illinois at Urbana-Champaign
Department of Computer Science

Spring 2024

Objectives

- Explain why we should use pseudocode even though you can't run it
- Read a pseudocode algorithm and translate it into Python
- Give a pseudocode description of an algorithm

Consider this…

Scenario: Suppose for some reason you forgot how to take the average of a list of elements. You ask your CS major friend for help.
There's a problem though… your friend happens not to know Python.
(That actually does happen…..)

In Haskell

So maybe they give you this:

```haskell
average :: Fractional a => [a] -> a
average xs = sum xs / fromIntegral (length xs)

main :: IO ()
main = do
    let xx = [1, 2, 3, 4, 5]
    putStrLn $ "Average: " ++ show (average xx)
```

In Emacs Lisp

```
(defun average (lst)
  (/ (apply #'+ lst) (length lst)))

(let ((xx '(1 2 3 4 5)))
  (message "Average: %s" (average xx)))
```

In MatLab

```
xx = [1, 2, 3, 4, 5];
avg = sum(xx) / numel(xx);
disp(['Average: ', num2str(avg)]);
```

In Ruby

```ruby
xx = [1, 2, 3, 4, 5]
average = xx.reduce(:+) / xx.size.to_f
puts "Average: #{average}"
```

In Python

```
xx = [1, 2, 3, 4, 5]
avg = sum(xx) / len(xx)
print("Average:", avg)
```

The point

- We often need to communicate algorithms to each other.
- But we don't always speak the same computer languages.
- Solution: use Pseudocode! (Pseudo = "False" in Greek)
- The purpose is not to run, it is to communicate to fellow humans!

In Pseudocode

Compared to Python, what is different / the same?

```
total := 0
count := 0

for each element num in xx:
    total := total + num
    count := count + 1

if count <> 0:
    avg := total / count
else:
    avg := NaN  // handle case when list is empty

print "Average:", avg
```

Comparison

- Assignment uses :=
  - Sometimes you will see ← instead.
  - Other math symbols like ≠ will also show up.
- We use indentation and colons like in Python
- We do not use special keywords or operators.
- Comments usually start with //

```
total := 0
count := 0

for each element num in xx:
    total := total + num
    count := count + 1

if count <> 0:
    avg := total / count
else:
    // etc...
```

Counting Occurrences

Here is pseudocode for counting the number of occurrences of a character in a string.

```
function countOccurrences(string, target):
    count := 0
    for each character in string:
        if character equals target:
            count := count + 1
    return count
```

- Can you convert this to Python?

Python Version

```
def count_occurrences(string, target):
    count = 0
    for char in string:
        if char == target:
            count += 1
    return count

# Example usage:
my_string = "banana"
target_char = "a"
result = count_occurrences(my_string, target_char)
print(f"The character '{target_char}' occurs {result} times
```

Guess what this does

```
function guess(num_points):
    points_inside_circle := 0
    total_points := 0

    for i from 1 to num_points:
        generate random point (x, y) in the unit square
                  [0, 1) × [0, 1)
        // check if point is inside the unit circle
        if x^2 + y^2 <= 1:
            points_inside_circle := points_inside_circle + 1
        total_points := total_points + 1

    ratio := points_inside_circle / total_points
    something := 4 * ratio
    return something
```

Python Equivalent

```
import random

def estimate_pi(num_points):
    points_inside_circle = 0
    total_points = 0

    for _ in range(num_points):
        x = random.random()  # Generate x in [0, 1)
        y = random.random()  # Generate y in [0, 1)
        if x**2 + y**2 <= 1:  # Is point inside the unit c
            points_inside_circle += 1
        total_points += 1

    ratio = points_inside_circle / total_points
    estimated_pi = 4 * ratio
    return estimated_pi
```

Your turn!

- Remember the standard deviation question from the homework?
  - Try to write a pseudocode version. After a few minutes compare with your friend to see if they are similar.
  - Assume you have a function to compute the mean.

My version

```
function compute_standard_deviation(xx):
    mean := compute_mean(xx)
    sum_squared_difference := 0

    for each element in xx:
        difference := element - mean
        sum_squared_difference :=
            sum_squared_difference + (difference^2)

    variance := sum_squared_difference / length(xx)
    standard_deviation := square_root(variance)

    return standard_deviation
```

Other Uses

- Pseudocode is great for communicating to other people.
- It's also great for trying to develop an actual program!