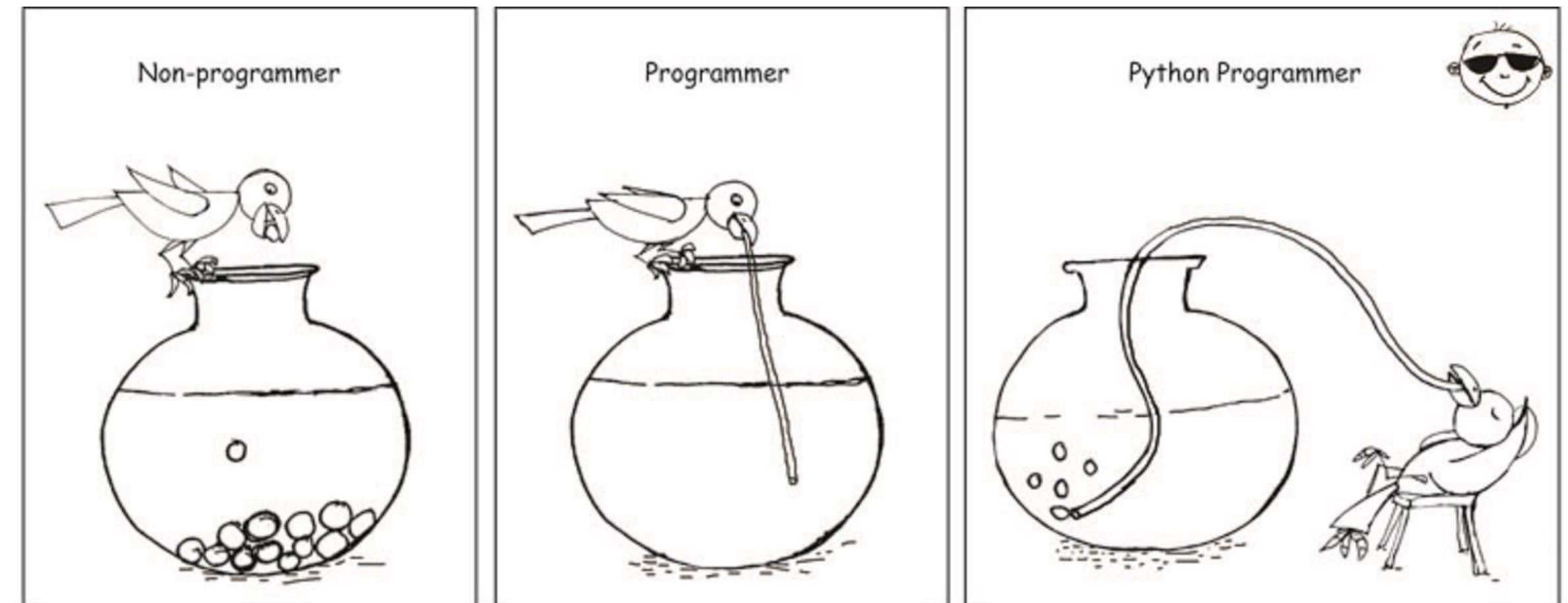
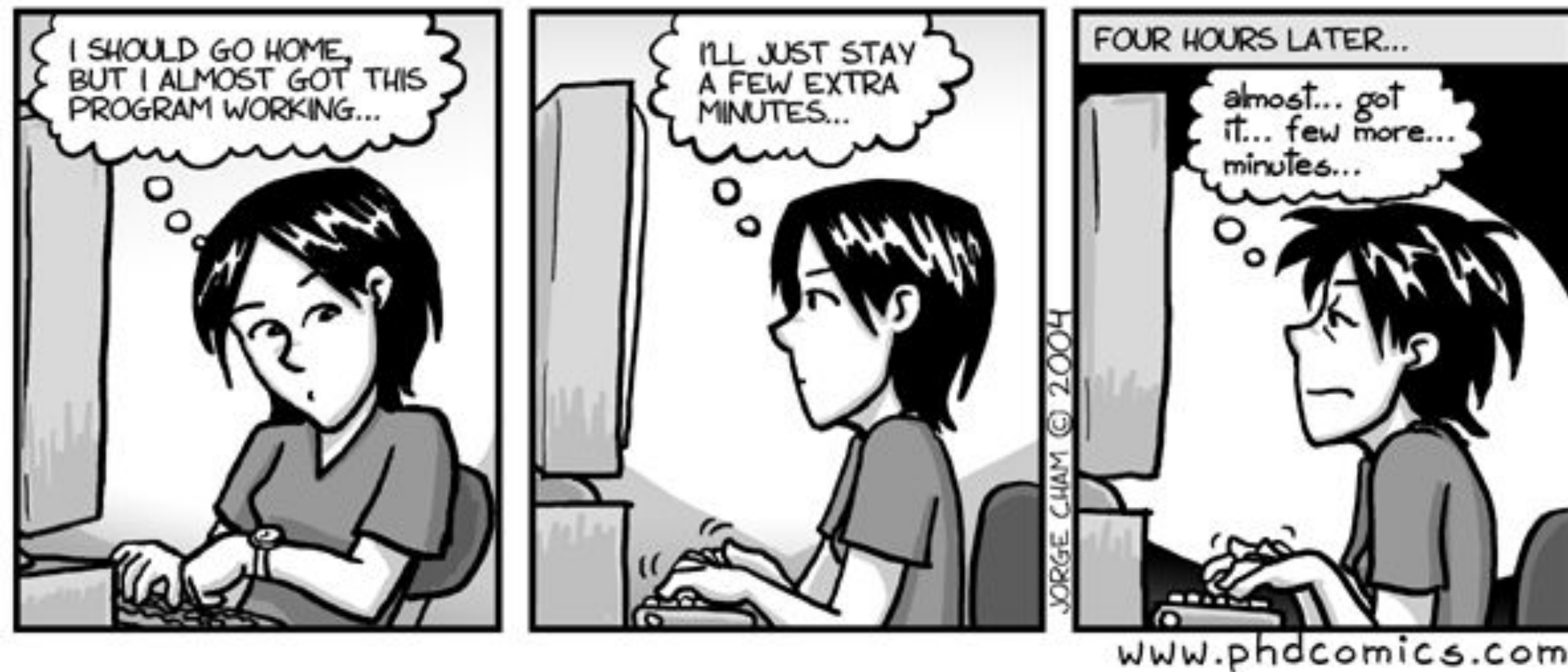


Python Pt2: Decisions & Loops

Spring 2025
PCfB Class 5
February 14, 2025



Check-in

Outline

- Lists & strings revisited
- Decisions (`if`, `elif`, `else`)
- `for` loops
- `while` loops

Lists & strings revisited

Positive indexing

`1 = [1, '1', 'one', [1,2]]`

Negative indexing

`1 = [1, '1', 'one', [1,2]]`

List slicing

1 = [1, '1', 'one', [1,2]]

String indexing & slicing

```
s = 'The dog ran.'
```

split() method

join () method

Decisions

if

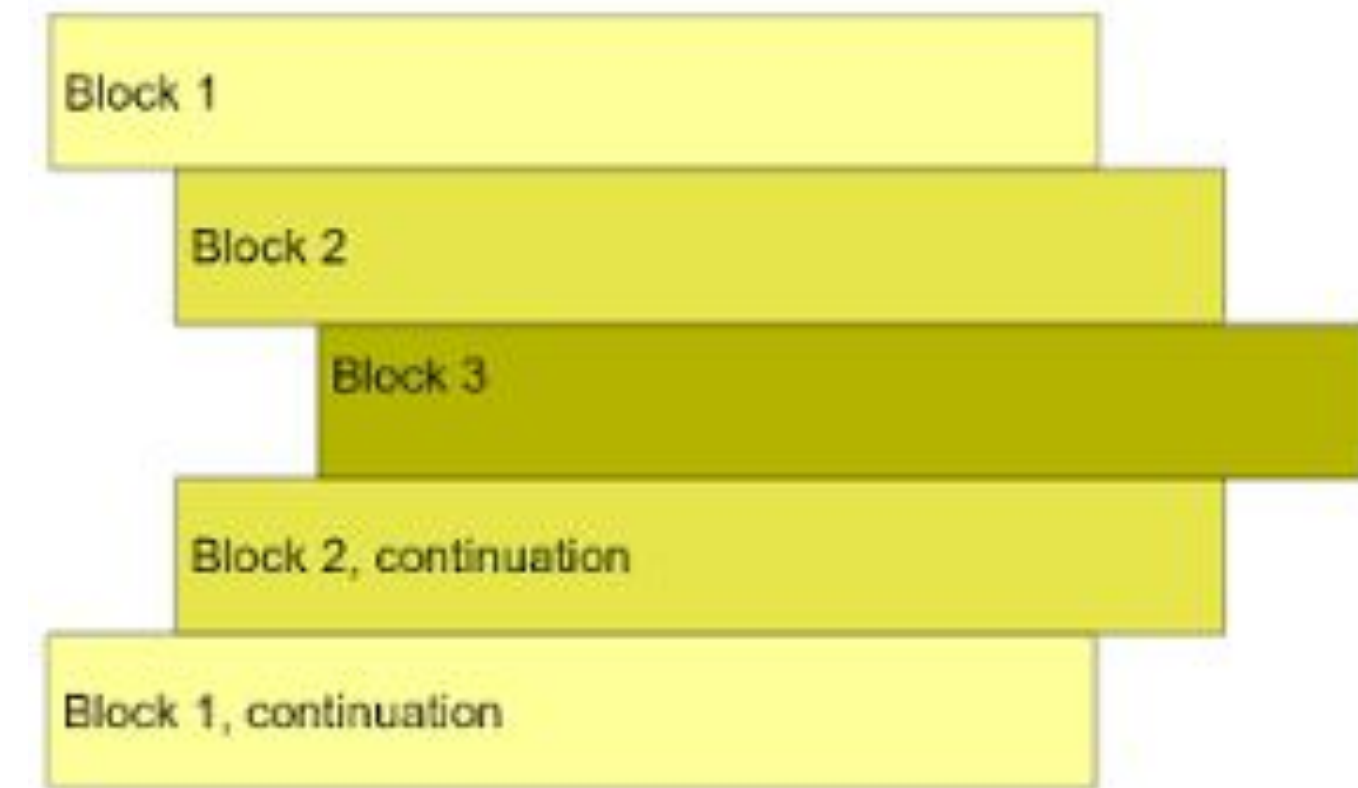
elif

else

```
1 p = 0.0013
2
3 if p > 0.05:
4     print("Insignificant")
5 elif 0.01 < p <= 0.05:
6     print("Significant")
7 elif 0.001 < p <= 0.01:
8     print("Highly Significant")
9 else:
10    print("Holy Cow!")
11
```


Indentation

- White space at the beginning of a line is used to define blocks of code
- Can use tabs, spaces or even a combination
- Best practice is to choose one and stick with it



```
2 for item in range(10):
3     print('I')
4     print('am')
5     print('a')
6     if item % 2 == 0:
7         print('funny')
8         print('and')
9         print('silly')
10    else:
11        print('dull')
12        print('and')
13        print('serious')
14    print('block')
15    print('used')
16    print('as')
17    print('example.')
```

Indentation shortcut

- Many text editors have shortcuts for quickly indenting blocks of code

Comparison operators (Table 9.1)

x == y

x != y

x > y

x < y

x >= y

x <= y

Logical operators (Table 9.2)

A and B

A or B

not B

(not A) or B

not (A or B)

for loops

for loops

```
15 pvalues = [0.67, 0.0003, 0.0013, 0.05, 0.76]
16
17 for p in pvalues:
18     if p > 0.05:
19         print("Insignificant")
20     elif 0.01 < p <= 0.05:
21         print("Significant")
22     elif 0.001 < p <= 0.01:
23         print("Highly Significant")
24     else:
25         print("Holy Cow!")
```

while loops


```
6 int_list = list(range(50)) . . . . . # Creates list with 50 integers
7 num_rmv = 4 . . . . . # # of items to remove from list in each iteration
8 itercount = 0 . . . . . # To keep track of the number of loop iterations
9 while len(int_list) >= num_rmv: . . . . . # Initiate while loop
10     . . . itercount+=1 . . . . . # Increment counter
11     . . . del(int_list[:num_rmv]) . . . . . # Delete items from the int_list variable
12     . . . print(itercount, len(int_list)) . . . # Print current iteration # and length of int_list
```

Initiate variables

While loop

Printed results

```
>>> int_list = list(range(50))
>>> num_rmv = 4
>>> itercount = 0
>>> while len(int_list) >= num_rmv:
...     itercount+=1
...     del(int_list[:num_rmv])
...     print (itercount, len(int_list))
...
1 46
2 42
3 38
4 34
5 30
6 26
7 22
8 18
9 14
10 10
11 6
12 2
```

Control statement within loops

- **break**: internal command to exit loop
- **continue**: end current loop iteration