Q: What is the initialization? (careful!)

1. Recognize that a range of integers b..c has to be processed
2. Write the command and equivalent postcondition
3. Write the basic part of the while-loop
4. Write loop invariant
5. Figure out any initialization

A: k = 0; big = int_list[0]
B: k = 1; big = int_list[0]
C: k = 1; big = int_list[1]
D: k = 0; big = int_list[1]
E: None of the above

```
# set big to largest element in int_list, a list

# Invariant: big is largest int in int_list[0...k-1]
while  k < len(int_list):

    k = k + 1
# Postcondition: big = largest int in int_list[0..len(int_list)−1]
```

34

Q: What range of s has been processed?

2. Write the command and equivalent postcondition
3. Write the basic part of the while-loop

A: 0..k
B: 1..k
C: 0..k–1
D: 1..k–1
E: I don't know

```
# set n_pair to number of adjacent equal pairs in s

while  k < len(s):
    k = k + 1
# POST: n_pair = # adjacent equal pairs in s[0..len(s)-1]
```

k: next integer to process.
What range of s has been processed?

22

Q: What is the loop invariant?

2. Write the command and equivalent postcondition
3. Write the basic part of the while-loop
4. Write loop invariant

```
# set n_pair to number of adjacent equal pairs in s

# INVARIANT:
while  k < len(s):

    k = k + 1
# POST: n_pair = # adjacent equal pairs in s[0..len(s)-1]
```

A: n_pair = num adj. equal pairs in s[1..k]
B: n_pair = num adj. equal pairs in s[0..k]
C: n_pair = num adj. equal pairs in s[1..k–1]
D: n_pair = num adj. equal pairs in s[0..k–1]
E: I don't know

24

Q: how to initialize k?

2. Write the command and equivalent postcondition
3. Write the basic part of the while-loop
4. Write loop invariant
5. Figure out any initialization

A: k = 0
B: k = 1
C: k = –1
D: I don't know

```
# set n_pair to # adjacent equal pairs in s
n_pair = 0; k = ?
# INV: n_pair = # adjacent equal pairs in s[0..k-1]
while  k < len(s):

    k = k + 1
# POST: n_pair = # adjacent equal pairs in s[0..len(s)-1]
```

26

Q: What do we compare to "process k"?

2. Write the command and equivalent postcondition
3. Write the basic part of the while-loop
4. Write loop invariant
5. Figure out any initialization
6. Implement the body (aka repetend) (# Process k)

```
# set n_pair to # adjacent equal pairs in s
n_pair = 0; k = 1
# INV: n_pair = # adjacent equal pairs in s[0..k-1]
while  k < len(s):

    k = k + 1
# POST: n_pair = # adjacent equal pairs in s[0..len(s)-1]
```

A: s[k] and s[k+1]
B: s[k-1] and s[k]
C: s[k-1] and s[k+1]
D: s[k] and s[n] E: I don't know

28

Q: What is the initialization? (careful!)

1. Recognize that a range of integers b..c has to be processed
2. Write the command and equivalent postcondition
3. Write the basic part of the while-loop
4. Write loop invariant
5. Figure out any initialization

A: k = 0; big = int_list[0]
B: k = 1; big = int_list[0]
C: k = 1; big = int_list[1]
D: k = 0; big = int_list[1]
E: None of the above

```
# set big to largest element in int_list, a list

# Invariant: big is largest int in int_list[0...k-1]
while  k < len(int_list):

    k = k + 1
# Postcondition: big = largest int in int_list[0..len(int_list)−1]
```

34

http://www.cs.cornell.edu/courses/cs1110/2018sp

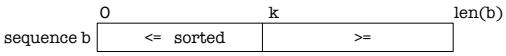
Lecture 25: Sequence Algorithms

CS 1110

Introduction to Computing Using Python

[E. Andersen, A. Bracy, D. Gries, L. Lee, S. Marschner, C. Van Loan, W. White]

Box Notation for Sequences



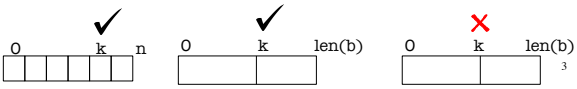
Graphical assertion about sequence b. It asserts that:

1. b[0..k–1] is sorted (values are in ascending order)
2. all of b[0..k–1] is ≤ all of b[k..len(b)–1]

Pro Tip #1:

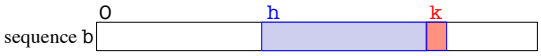
index always goes *above a box*, never above a line

(just like house numbers go on a house not between the houses)



3

Q: Indices for Box Notation



Given:

- index **h** of the first element of a segment
- index **k** of the element that follows that segment,

Questions:

1. How many values are in segment b[h .. k – 1]
2. How many values are in b[h .. h – 1] ?
3. How many values are in b[h .. h + 1] ?

A: 0
B: 1
C: 2
D: k - h
E: k + h

Pro Tip #2:

Size is "Follower minus First"

Follower: next thing outside the specified range

4

count num adjacent equal pairs (1)

Approach #1: compare s[k] to the character in front of it (s[k-1])

```
# set n_pair to # adjacent equal pairs in s


n_pair = 0
k = 1

while k < len(s):
    if s[k-1] == s[k]:
        n_pair += 1
    k = k + 1
```

6

find the max of a seq (1)

Task: find the maximum of a sequence s

```
k = 1
big = s[0]


while k < len(s):
    big = max(big, s[k])
    k = k + 1
```

8

Task: separate + from – in a list

Task: Put negative values before nonnegative ones and return the split index

11