

CS 1110

Prelim 2 Review Spring 2021

Announcements

- Prelim 2 Thurs Apr 22 at 6:30 - 8pm (university-scheduled)
 - Your seat or Zoom link will be assigned this afternoon via CMS
 - In-person: Bring pens/pencils/erasers (bring several). Bring a watch or even an actual clock if you have one. No smart watches/phones! You may not be able to see the wall clock in Barton from your seat. Bring Cornell ID.
 - Online: *Different this time: log on to Zoom proctor session on both devices.* Students who have not done a mock exam (for Prelim 1) will be contacted to do one.
- Labs this week: Prelim 2 review, focus on class methods
- Thurs Apr 22 lecture time → office hours

Studying for the Exam

- Read study guide. Notes differences among the semesters
- Review all labs and assignments
 - You should be able to do all problems now
- Look at exams from past years
 - Exams with solutions on course web page
 - Refer to info in study guide regarding differences among the semesters

Prelim 2 Review

4

Prelim 2 Topics

- Topics after prelim 1:
 - Recursion now
 - Classes lab
- Topics before but not on prelim 1:
 - Nested lists now
 - Iteration with nested loops now
 - Dictionaries and tuples now

While-loop *not* on Prelim 2

Prelim 2 Review

5

Recursion: Before You Begin

- Plan out how you will approach the task before writing code
- Consider the following:
 - How can you “divide and conquer” the task?
 - Do you understand the spec?
 - How would you describe the implementation of the function using words?

Prelim 2 Review

6

Recursion

1. Base case
2. Recursive case
3. Ensure the recursive case makes progress towards the base case

Prelim 2 Review

7

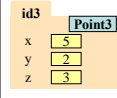
Base Case

- Create cases to handle smallest units of data
- Depends on what type of data is being processed and what the function must do to that data

Prelim 2 Review

8

Base Case Examples

| | Strings | Lists | Point3 objects |
|------------|---------|-------|---|
| 1 Element | "5" | [5] |  |
| 0 Elements | "" | [] | None |

Prelim 2 Review

9

Recursive Case

- Divide and conquer: how to divide the input so that we can call the function recursively on smaller input
- When calling the function recursively, assume that it works exactly as the specification states it does -- don't worry about the specifics of your implementation here
- Use this recursive call to handle the rest of the data, besides the small unit being handled

Prelim 2 Review

10

Make Progress

- Recursive calls must always make some sort of "progress" towards the base cases
- This is the only way to ensure the function terminates properly
- Risk having infinite recursion otherwise

Prelim 2 Review

11

Recursive Function (Fall 2017)

```
def filter(nlist):
```

```
    """Return: a copy of nlist with all negative numbers removed.
```

```
    The order of the original list is preserved
```

```
    Example: filter([1,-1,2,-3,-4,0]) returns [1,2,0]
```

```
    Precondition: nlist is a (possibly empty) list of numbers."""
```

Prelim 2 Review

12

Prelim 2 Review

14

Recursive Function (Fall 2014)

def histogram(s):

"""Return: a histogram (dictionary) of the # of letters in string s.

The letters in s are keys, and the count of each letter is the value. If the letter is not in s, then there is NO KEY for it in the histogram.

Example: histogram("") returns {},

histogram('abracadabra') returns {'a':5, 'b':2, 'c':1, 'd':1, 'r':2}

Precondition: s is a string (possibly empty) of just letters."""

Prelim 2 Review

17

Dictionaries (Type dict)

```
>>> d = {'ec1':'Ezra', 'ec2':'Ezra', 'tm55':'Toni'}
>>> d['ec1']
'Ezra'
>>> d[0]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 0
>>> d[:1]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unhashable type: 'slice'
>>>
```

Global Space

d id8

Heap Space

id8

dict

'ec1' 'Ezra'

'ec2' 'Ezra'

'tm55' 'Toni'

- Can access elements like a list
- Must use the key, not an index
- Cannot slice ranges

19

Iteration with For-Loops

Two ways to implement the for-loop

for x in alist:

- x is each value inside the list
- Modifying x does not modify the list

for x in range(len(alist)):

- x represents each *index* of the list
- Modifying alist[x] modifies the list

Prelim 2 Review

20

Prelim 2 Review

22

Example with 2D Lists

def max_cols(table):

"""Returns: List storing max value of each column

We assume that table is a 2D list of floats (so it is a list of rows and each row has the same number of columns. This function returns a new list that stores the maximum value of each column.)

Examples:

max_cols([[1,2,3], [2,0,4], [0,5,2]]) is [2,5,4]

max_cols([[1,2,3]]) is [1,2,3]

Precondition: table is a NONEMPTY 2D list of floats

Built-in function max not allowed. """

Prelim 2 Review

23

Prelim 2 Review

24

Questions? Next up: Office Hours



Prelim 2 Review

26

Recursion with Objects

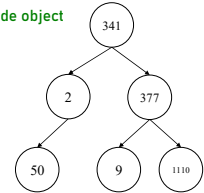
class TreeNode (object):

"""Attributes:

value: An int, the "value" of this TreeNode object

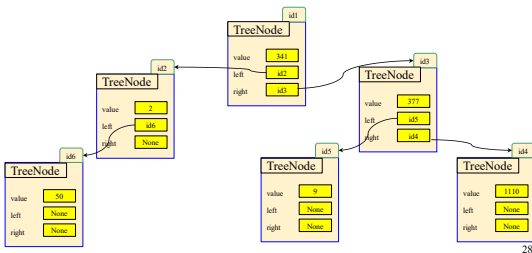
left: A TreeNode object, or None

right: A TreeNode object, or None"""



27

Understanding the Object's Structure



28

Recursion with Objects

def contains (t, v):

"""

Return: True if any of the TreeNode objects in the entire "tree" have the value v

Define the "tree" as the TreeNode t, as well as the TreeNodes accessible through the left and right attributes of t (if not None)

Preconditions: t is a TreeNode, or None. v is an int.

"""

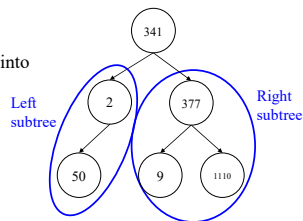
29

Divide and Conquer on Trees

Recall the tree structure...

They can be easily divided into left and right subtrees!

Recursion on left
 Recursion on right
 Put result back together



31