

Compsci 101, Spring 2018

Owen Astrachan and Kristin Stephens-Martinez

February 13, 2018

Midterm 1 Review

<http://bit.ly/101spring18-feb13>

Announcements

- Midterm details
 - Help sheet we provide, help sheet that you provide
 - Timing for all students
 - Taking the exam: writing utensil ✓✓electronics 💣 💀
- What's coming next?
 - Next set of APTs
 - Next Assignment



Are you interested in technology, design, hackathons, or social issues? If so, come to HackDuke's recruiting sessions on Wednesday and Thursday this week to learn more about joining our awesome team! Working with us allows you to get directly involved in Duke's CS community and have a hand in shaping its future. Our annual hackathon has been described as one of the nation's best beginner hackathons, our spring design conference has featured professionals from Spotify, Facebook, and more, and our focus on tech for social good sets us apart from any other campus organization. Check us out at hackduke.org and we hope to see you at our info sessions!

Info session details:

Info session #1:

Wednesday, Feb 14th

7:15 - 8:00 PM

West Union room 248

Facebook event: <https://tinyurl.com/HDinfosession1>

Info session #2:

Thursday, Feb 15th

7:15 - 8:00 PM

West Union room 248

Facebook event: <https://tinyurl.com/HDinfosession2>



Midterm Review

- We will answer questions from you: practice midterm
 - You'll be able to ask questions anonymously too
- We will have some new questions for you as well based on labs, assignments, APTs, lectures
 - You'll work on these we answer questions

Name, Type, Value

```
words = ["red", "green", "tree", "play"]
```

```
str = "Oh! no! not now!"
```

<code>b = str[2]</code>		
<code>c = words[-1]</code>		
<code>d = 13/4</code>		
<code>e = 5 + 2.0*4</code>		
<code>f = len(words)</code>		
<code>g = words[2][1]</code>		

Name, Type, Value

```
words = ["red", "green", "tree", "play"]
```

```
str = "Oh! no! not now!"
```

<code>h = words[1].upper()</code>		
<code>i = str.split()[1]</code>		
<code>j = str[5:6] + str[:2]</code>		
<code>k = str.find("not") > str.find("how")</code>		
<code>m = "y".join(["mo", "bo"])</code>		
<code>n = 24 % 7</code>		

There is more than one way ...

```
6 def dinnerPrice(item1, item2, day, month):
7     low = min(item1, item2)
8     high = max(item1, item2)
9
10    if day == "Monday":
11        low = low * 0.5
12    if day == "Tuesday":
13        low = low * 0.90
14
15    total = low + high
16    if month == "April" or month == "May":
17        total = total - 5
18    if total < 0:
19        total = 0
20    return total
```

What's the bug? How to Fix?

```
1 def countWordInPhrases(listPhrases, word):  
2     count = 0  
3     for phrase in listPhrases:  
4         if word in phrase:  
5             count += 1  
6     return count
```

```
listPhrases = ["great Tuesday", "one sunny day", "day in and out"]
```

Here are two calls, one with a wrong answer and one with a correct answer

CALL	returns	correct answer
<code>countWordInPhrases(listPhrases, "day")</code>	3	2
<code>countWordInPhrases(listPhrases, "and")</code>	1	1

Stretch it out

```
24 def stretchOut(word):
25     word = word.lower()
26     if len(word) < 3:
27         return word
28     if isVowel(word[0]):
29         return word
30     twos = word[0]+word[0] + word[1] + word[1]
31     return twos + word[2:] + "eh"
```

Categorizing Food

```
33 def category(fname, food):
34     f = open(fname)
35     ret = []
36     for line in f:
37         line = line.strip()
38         data = line.split(",")
39         if data[0] == food:
40             ret.append(data[1])
41     f.close()
42     return ret
43 ..
```

How many calories?

```
45 def biggest(fname):  
46     f = open(fname)  
47     caloric = ""  
48     cmax = 0  
49     for line in f:  
50         data = line.strip().split(",")  
51         cal = int(data[2])  
52         food = data[1]  
53         if cal > cmax:  
54             cmax = cal  
55             caloric = food  
56     f.close()  
57     return caloric
```

WOTO

<http://bit.ly/101spring18-feb13-woto>

WordGame

In a word game a player's words earn points for each word that is a valid word, e.g., is in a dictionary. The score for each valid word is the square of the word's length. Write the function **score** to determine the total score earned. The example call below return 58: 9 + 49 for "cat" and "doormat". First parameter is wordsPlayed

```
score(["at", "bat", "cat", "doormat"],  
      ["cat", "dog", "scarf", "doormat"])
```

WordGame

- wordsPlayed ["dog", "lettuce", "cookie"]
- validWords ["ant", "cat", "dog", "element", "lettuce", "tuna"]
- Function **score** returns the total score for words played during game that are valid: length^2 for each such word
 - Score for example? 3 for dog, 7 for lettuce $9+49=58$

```
def score(wordsPlayed, validWords):
```

APT Score

- Text file in format shown: netid,apt-name,score (float)
 - Given filename, netid and APT-name find maximal score for that APT
 - Call below returns 9.5

```
maxScore("data.txt",  
         "epa7", "tetra")
```

```
epa7,tetra,8  
epa7,tetra,9  
fpo9,baker,2  
epa7,tetra,9.5  
fpo9,baker,4.5  
epa7,tetra,8
```