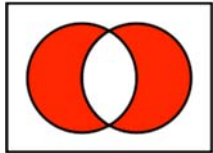


# CompSci 101

## Introduction to Computer Science



Oct 17, 2017  
Prof. Rodger

cps101 fall2017

1

# Enjoy a taste of the NC State Fair!



Fudge

Salt Water Taffy

cps101 fall2017

2

## Exam 1... on Gradescope

- Use **YOURNETID@duke.edu** for email
- Solutions posted – request regrades til Oct 24
  - Ask for regrade on gradescope
- Try working problem you missed first
  - Then look at solution
- Once you think you understand
  - Get blank sheet of paper – try again
- Understand all solutions

cps101 fall2017

3

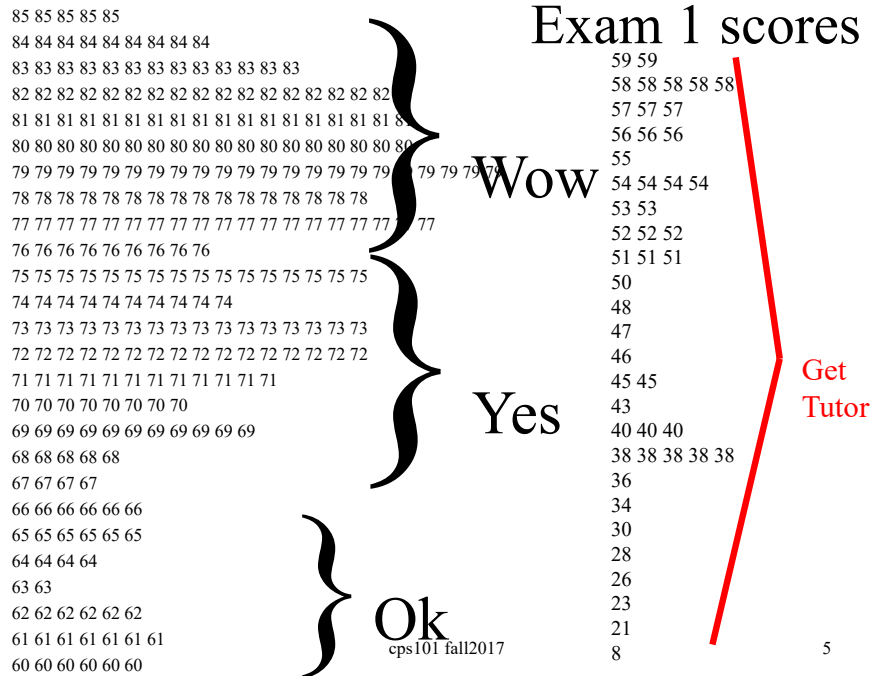
## Exam 1 scores

85 85 85 85 85  
84 84 84 84 84 84 84 84  
83 83 83 83 83 83 83 83 83 83  
82 82 82 82 82 82 82 82 82 82 82 82  
81 81 81 81 81 81 81 81 81 81 81 81 81 81  
80 80 80 80 80 80 80 80 80 80 80 80 80 80  
79 79 79 79 79 79 79 79 79 79 79 79 79 79  
78 78 78 78 78 78 78 78 78 78 78 78 78 78  
77 77 77 77 77 77 77 77 77 77 77 77 77 77  
76 76 76 76 76 76 76 76 76 76 76 76 76 76  
75 75 75 75 75 75 75 75 75 75 75 75 75 75  
74 74 74 74 74 74 74 74 74 74 74 74 74 74  
73 73 73 73 73 73 73 73 73 73 73 73 73 73  
72 72 72 72 72 72 72 72 72 72 72 72 72 72  
71 71 71 71 71 71 71 71 71 71 71 71 71 71  
70 70 70 70 70 70 70 70 70 70 70 70 70 70  
69 69 69 69 69 69 69 69 69 69 69 69 69 69  
68 68 68 68 68 68 68 68 68 68 68 68 68 68  
67 67 67 67 67 67 67 67 67 67 67 67 67 67  
66 66 66 66 66 66 66 66 66 66 66 66 66 66  
65 65 65 65 65 65 65 65 65 65 65 65 65 65  
64 64 64 64 64 64 64 64 64 64 64 64 64 64  
63 63 63 63 63 63 63 63 63 63 63 63 63 63  
62 62 62 62 62 62 62 62 62 62 62 62 62 62  
61 61 61 61 61 61 61 61 61 61 61 61 61 61  
60 60 60 60 60 60 60 60 60 60 60 60 60 60

cps101 fall2017

4

59 59  
58 58 58 58 58  
57 57 57  
56 56 56  
55  
54 54 54 54  
53 53  
52 52 52  
51 51 51  
50  
48  
47  
46  
45 45  
43  
40 40 40  
38 38 38 38 38  
36  
34  
30  
28  
26  
23  
21  
8



## Exam 1 stats

- Average: 70.9/86
- Median: 74.5/86
- See your email about small group tutoring or private tutors

cps101 fall2017

6

## Announcements

- Reading and RQ due next time
- Assignment 5 out today
- APT 4 due Thursday, APT 5 out partially
- Lab 6 this week
  - Read APT Anagramfree and Assignment 5 before going to lab!
- Today:
  - list comprehension – shortcut for building a list
  - Sets – new way to organize data

cps101 fall2017

7

## APT MorseLikeCode

```
1. library = ["O ---",
             "S ..."]
message = "... --- ..."
Returns: "SOS"
```

The example from the problem statement.

```
2. library = ["O ---"]
message = "... --- ..."
Returns: "?O?"
```

This time we don't have the S, so we replace the three dashes with question marks.

```
3. library = ["H -", "E .", "L --", "L ..", "O -."]
message = "-. -- .. -." Returns: "HELLO"
```

compsci101 fall 2017

8

## Solving APT MorseLikeCode

- Put library in a different format?
  - [“H -”, “E .”, “L -”, “O ..”]

## Solving APT MorseLikeCode

- Put library in a different format?
  - [“H -”, “E .”, “L -”, “O ..”]
- 1) list of lists?
  - lib = [ [“H”, “-”], [“E”, “.”], [“L”, “-.”], [“O”, “..”] ]
- 2) list of strings?
  - lib = [“H”, “-”, “E”, “.”, “L”, “-.”, “O”, “..”]
  - code in pos *i* corresponds to letter in pos *i-1*
- 3) 2 parallel lists?
  - letters = [“H”, “E”, “L”, “O”]
  - codes = [“-”, “.”, “-.”, “..”]
  - code in pos *i* corresponds to letter in pos *i*

## Solving problems – APT MorseLikeCode

- Compare find vs index
  - find with string – returns -1 when not found
  - index with list – CRASHES if not there!
  - You can’t say: pos = alist.index(“...”)
  - Instead: if “...” in alist:
    - pos = alist.index(“...”)

## MorseLikeCode cont

- Write helper function – for a code, determine the letter for that code using the library
- Send library in new format

```
def codeToSymbol(library, code)
```

```
    return letter
```

## Back to Lists ...

### Build a list from another list

- Given a list of numbers, create a second list of every number squared.

```
nums = [8, 3, 5, 4, 1]
```

```
sqnums = []
```

```
for v in nums:
```

```
    sqnums.append(v*v)
```

```
print sqnums
```

```
[64, 9, 25, 16, 1]
```

cps101 fall2017

13

## List Comprehension -

### Short cut way to build a list

- Take advantage of patterns, make a new list based on per element calculations of another list

- Format:

```
[<expression with variable> for <variable> in  
<old list>]
```

- Example:

```
nums = [8, 3, 5, 4, 1]
```

```
sqnums = [v*v for v in nums]
```

14

### These result in the same list!

```
nums = [8, 3, 5, 4, 1]
```

```
1) sqnums = []  
   for v in nums:  
       sqnums.append(v*v)
```

```
2) sqnums = [v*v for v in nums]
```

cps101 fall2017

15

### Examples of List Comprehensions

[bit.ly/101f17-1017-1](http://bit.ly/101f17-1017-1)

```
nums = [4, 3, 8]
```

```
x = [v for v in nums]
```

```
x = [2 for v in nums]
```

```
x = sum([v*2 for v in nums])
```

```
x = [v+5 for v in nums][1]
```

```
x = [ nums[len(nums)-i -1] for i  
      in range(len(nums)) ]
```

cps101 fall2017

16

## Creating a list with just the even numbers

```
nums = [8, 3, 5, 4, 1]
evennums = []
for v in nums:
    if v % 2 == 0:
        evennums.append(v)
print evennums
```

[8, 4]

cps101 fall2017

17

## List Comprehension with Filtering

- Create list and use “if” to filter out elements to the list
- Format:
- [`<expression with variable> for <variable> in <old list> if <filter with variable>` ]

• Example: `nums = [8, 3, 5, 4, 1]`  
`evennums =`  
`[v for v in nums if v%2==0]`

cps101 fall2017

18

## More on List Comprehensions

[www.bit.ly/101f17-1017-2](http://www.bit.ly/101f17-1017-2)

```
names = ["Bo", "Moe", "Mary", "Aaron", "Joe"]
```

- What is the list for the following:
  - 1) `[w for w in names if w.endswith("e")]`
  - 2) `[w for w in names if w.lower()[0] > 'c']`
  - 3) `[j+1 for j in range(20) if (j%3) == 0]`
  - 4) `[i*2 for i in [j+1 for j in range(20) if (j%3) == 0] if i*i > 19]`

cps101 fall2017

19

## More on List Comprehensions

[bit.ly/101f17-1017-3](http://bit.ly/101f17-1017-3)

- Problem: Given a list of strings, return the longest string. If there are more than one of that length, return the first such one.  
`fruit = ['kiwi', 'plum', 'orange', 'lemon', 'banana']`  
Use a list comprehension for this problem

cps101 fall2017

20

## Richard Stallman

- MacArthur Fellowship (Genious grant)
- ACM Grace Murray Hopper award
- Started GNU – Free Software Foundation (1983)
  - GNU Compiler Collection
  - GNU Emacs



cps101 fall2017

21

## Python Sets

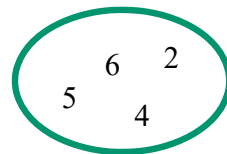
- Set – unordered collection of distinct items
  - Unordered – can look at them one at a time, but cannot count on any order
  - Distinct - one copy of each
- Operations on sets:
  - Modify: add, clear, remove
  - Create a new set: difference(-), intersection(&), union (|), symmetric\_difference(^)
  - Boolean: issubset <=, issuperset >=
- Can convert list to set, set to list
  - Great to get rid of duplicates in a list

cps101 fall2017

22

## List vs Set

- List
  - Ordered, 3<sup>rd</sup> item, can have duplicates
  - Example: `x = [4, 6, 2, 4, 5, 2, 4]`
- Set
  - No duplicates, no ordering
  - Example: `y = set(x)`
- Both
  - Add, remove elements
  - Iterate over all elements



cps101 fall2017

23

## Summary (from wikibooks)

```
• set1 = set() # A new empty set
• set1.add("cat") # Add a single member
• set1.update(["dog", "mouse"]) # Add several members
• set1.remove("cat") # Remove a member - error if not there

• print set1
• for item in set1: # Iteration or "for each element"
    print item
• print "Item count:", len(set1) # Length, size, item count
• isempty = len(set1) == 0 # Test for emptiness
• set1 = set(["cat", "dog"]) # Initialize set from a list
• set3 = set1 & set2 # Intersection
• set4 = set1 | set2 # Union
• set5 = set1 - set3 # Set difference
• set6 = set1 ^ set2 # Symmetric difference (elements in either set but not both)

• issubset = set1 <= set2 # Subset test
• issuperset = set1 >= set2 # Superset test
• set7 = set1.copy() # A shallow copy (copies the set, not the elements)
• set8.clear() # Clear, empty, erase
```

cps101 fall2017

24

## Creating and changing a set

```
colorList = ['red', 'blue', 'red', 'red', 'green']
colorSet = set(colorList)
smallList = list(colorSet)
colorSet.clear()
colorSet.add("yellow")
colorSet.add("red")
colorSet.add("blue")
colorSet.add("yellow")
colorSet.add("purple")
colorSet.remove("yellow")
```

What is the value of smallList and colorSet after this code executes?

cps101 fall2017

25

## Creating and changing a set

```
colorList = ['red', 'blue', 'red', 'red', 'green']
colorSet = set(colorList)
smallList = list(colorSet)
colorSet.clear()
colorSet.add("yellow")
colorSet.add("red")
colorSet.add("blue")
colorSet.add("yellow")
colorSet.add("purple")
colorSet.remove("yellow")
```

smallList = ['red', 'green', 'blue'] **order?**  
colorSet = set(["purple", "red", "blue"]) **order?**

26

## Set Operations

```
UScolors = set(['red', 'white', 'blue'])
dukeColors = set(['blue', 'white', 'black'])
print dukeColors.union(UScolors)
print dukeColors | UScolors
print dukeColors.intersection(UScolors)
print dukeColors & UScolors
```

cps101 fall2017

27

## Set Operations

```
UScolors = set(['red', 'white', 'blue'])
dukeColors = set(['blue', 'white', 'black'])
print dukeColors.union(UScolors)
print dukeColors | UScolors
print dukeColors.intersection(UScolors)
print dukeColors & UScolors
```

```
set(['blue', 'black', 'white', 'red'])
set(['blue', 'black', 'white', 'red'])
set(['blue', 'white'])
set(['blue', 'white'])
```

cps101 fall2017

28

## Set Operations

```
UScolors = set(['red', 'white', 'blue'])
dukeColors = set(['blue', 'white', 'black'])
print dukeColors.difference(UScolors)
print dukeColors - UScolors
print UScolors - dukeColors
```

## Set Operations

```
UScolors = set(['red', 'white', 'blue'])
dukeColors = set(['blue', 'white', 'black'])
print dukeColors.difference(UScolors)
print dukeColors - UScolors
print UScolors - dukeColors
```

```
set(['black'])
```

```
set(['black'])
```

```
set(['red'])
```

## Set Operations

```
UScolors = set(['red', 'white', 'blue'])
dukeColors = set(['blue', 'white', 'black'])
print dukeColors ^ UScolors
print UScolors ^ dukeColors
```

## Set Operations

```
UScolors = set(['red', 'white', 'blue'])
dukeColors = set(['blue', 'white', 'black'])
print dukeColors ^ UScolors
print UScolors ^ dukeColors
```

```
set(['black', 'red'])
```

```
set(['black', 'red'])
```

## Set Examples

bit.ly/101f17-1017-4

```
poloClub = set(['Mary', 'Laura', 'Dell'])
```

```
rugbyClub = set(['Fred', 'Sue', 'Mary'])
```

Questions:

```
print [w for w in poloClub.intersection(rugbyClub)]
```

```
print poloClub.intersection(rugbyClub)
```

```
print [w for w in poloClub.union(rugbyClub)]
```

```
print poloClub.union(rugbyClub)
```

## Set Examples (cont)

```
lista = ['apple', 'pear', 'fig', 'orange', 'strawberry']
```

```
listb = ['pear', 'lemon', 'grapefruit', 'orange']
```

```
listc = [x for x in lista if x in listb]
```

```
listd = list(set(lista)|set(listb))
```

## Assignment 5 - Hangman

- Guess a word given the number of letters.
  - Guess a letter
  - see if it is in the word and where.
- Demo
- Will start in lab

## APT AnagramFree

```
words = ["creation", "sentence", "reaction", "sneak", "star", "rats", "snake"]
```

Returns: 4

```
"star" "rats"  
"snake" "sneak"  
"creation" "reaction"  
"sentence"
```

→ both have letters: a r t s