Compsci 101
Clever Hangman, Problem Solving

Group/Template	Size of Group
_ a _ _ _ _ _ _	587
_ a _ a _ _ _ _	63
_ _ a _ _ _ _ _	498
_ _ _ a _ _ _ _	406
_ _ _ _ _ _ _ _	3,475

Susan Rodger

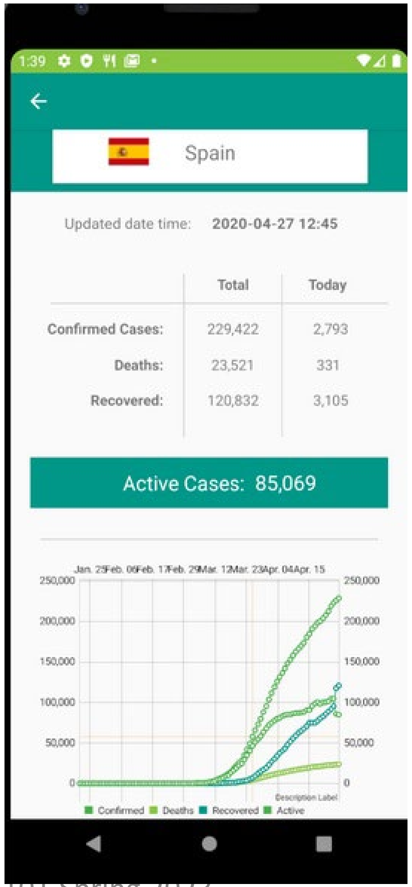
March 28, 2023

`R` is for …

- **Random**
  - `.choice`, `.shuffle`, `.seed`, `.randint`
- **R**
  - Programming language of choice in stats
- **Refactoring**
  - A way to rename your variable, function name

Esther Brown

- **Duke Alum 2020, IDM CS/Cult. Anth.**
- **Harvard MS Data Sci**
- **Now PhD in CS at Harvard!**
- **At Duke, as Senior did I.S. creating five Apps**
  - Covid tracker
  - Movie App



1:39

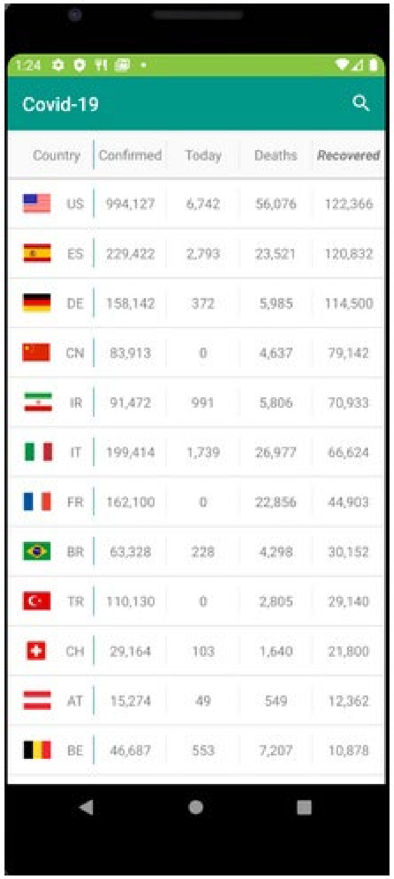
Spain

Updated date time: **2020-04-27 12:45**

	Total	Today
Confirmed Cases:	229,422	2,793
Deaths:	23,521	331
Recovered:	120,832	3,105

Active Cases: 85,069

Jan. 25Feb. 06Feb. 17Feb. 29Mar. 12Mar. 23Apr. 04Apr. 15
250,000
200,000
150,000
100,000
50,000
0
Description Label
🟩 Confirmed 🟩 Deaths 🟦 Recovered 🟩 Active



1:24

Covid-19

Country	Confirmed	Today	Deaths	Recovered
US	994,127	6,742	56,076	122,366
ES	229,422	2,793	23,521	120,832
DE	158,142	372	5,985	114,500
CN	83,913	0	4,637	79,142
IR	91,472	991	5,806	70,933
IT	199,414	1,739	26,977	66,624
FR	162,100	0	22,856	44,903
BR	63,328	228	4,298	30,152
TR	110,130	0	2,805	29,140
CH	29,164	103	1,640	21,800
AT	15,274	49	549	12,362
BE	46,687	553	7,207	10,878

Announcements

- **APT 5 due Thursday!**
- **Assignment 5 due Thursday, April 6**

- **No lab this Friday**

- **Reading and Sakai Quizzes due Thursday**

- **APT Quiz 2 Thursday 1:15pm through 11pm Monday**
  - Must complete by 11pm

PFTD

- **APT Quiz 2**
- **APT Family**
- **Clever Guess Word**
  - Focus on the dictionary
- **Problem solving with lists, sets and dictionaries**

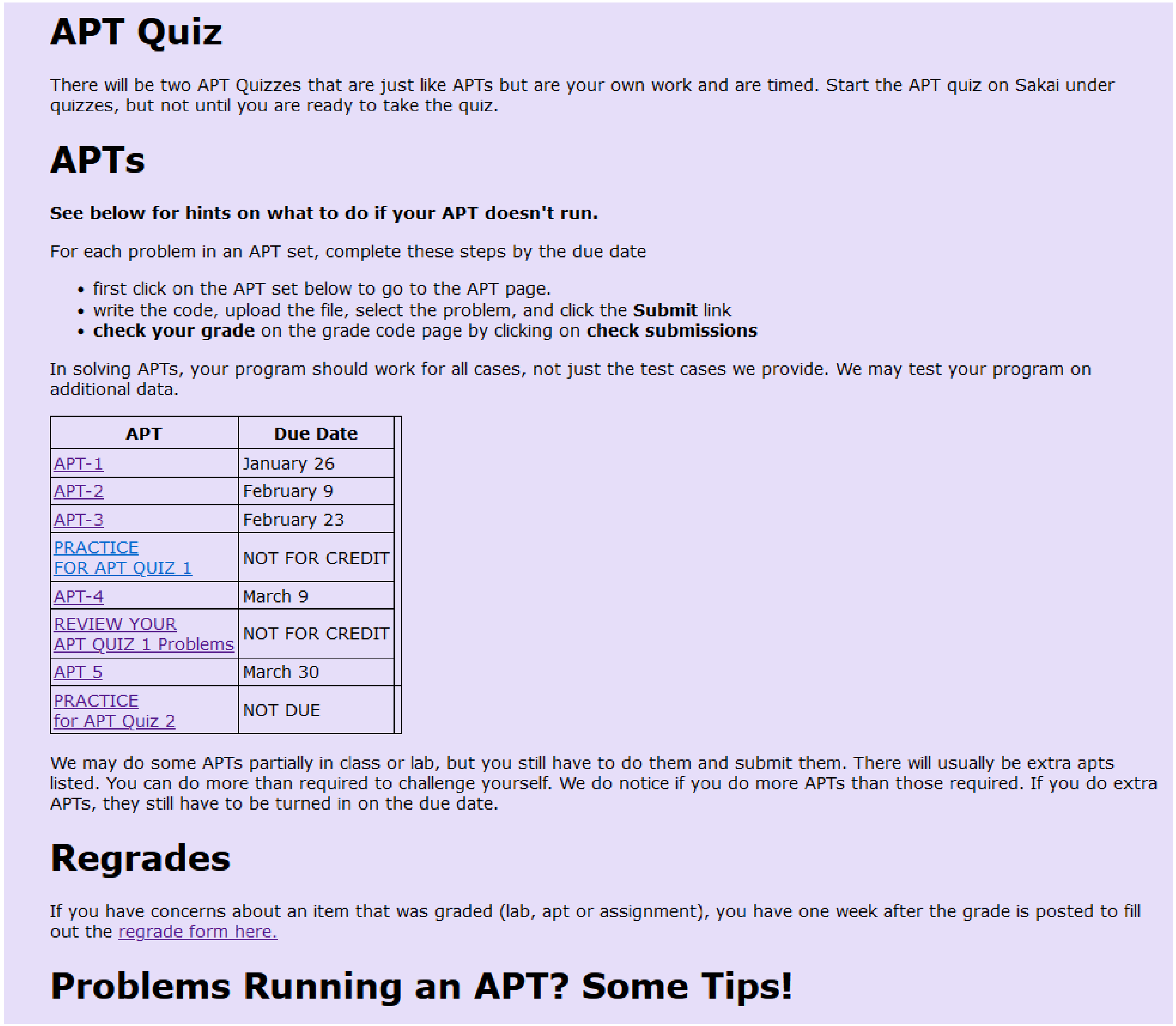
- **Next time: More on Sorting**

APT Quiz 2 March 30-April 3

- **Opens March 30, Thursday, 1:15pm**
- **Closes at 11pm Mon 4/3 – must finish all by this time**
- **There are two parts based on APTs 1-5**
  - Each part has two APT problems
  - Each part is 3 hours – more if you get accommodations
  - Each part starts in Sakai under tests and quizzes
  - Sakai is a starting point with countdown timer that sends you to a new apt page just for each part
  - Could do each part on different day or same days
- **Old APT Quiz so you can practice (not for credit) – on APT Page**

APT Quiz 2

- **Is your own work!**
  - No collaboration with others!
  - Use your notes, lecture notes, your code, textbook
  - DO NOT search for answers!
  - Do not talk to others about the quiz until grades are posted
- **Post private questions on Ed Discussion**
  - We are not on between 9pm and 9am!
  - We are not on all the time, especially weekends
  - Will try to answer questions between 9am – 9pm
    - About typos, cannot help you in solving APTs
- **See 101 APT page for tips on debugging APTs**



APT Quiz

There will be two APT Quizzes that are just like APTs but are your own work and are timed. Start the APT quiz on Sakai under quizzes, but not until you are ready to take the quiz.

APTs

See below for hints on what to do if your APT doesn't run.

For each problem in an APT set, complete these steps by the due date

- first click on the APT set below to go to the APT page.
- write the code, upload the file, select the problem, and click the **Submit** link
- **check your grade** on the grade code page by clicking on **check submissions**

In solving APTs, your program should work for all cases, not just the test cases we provide. We may test your program on additional data.

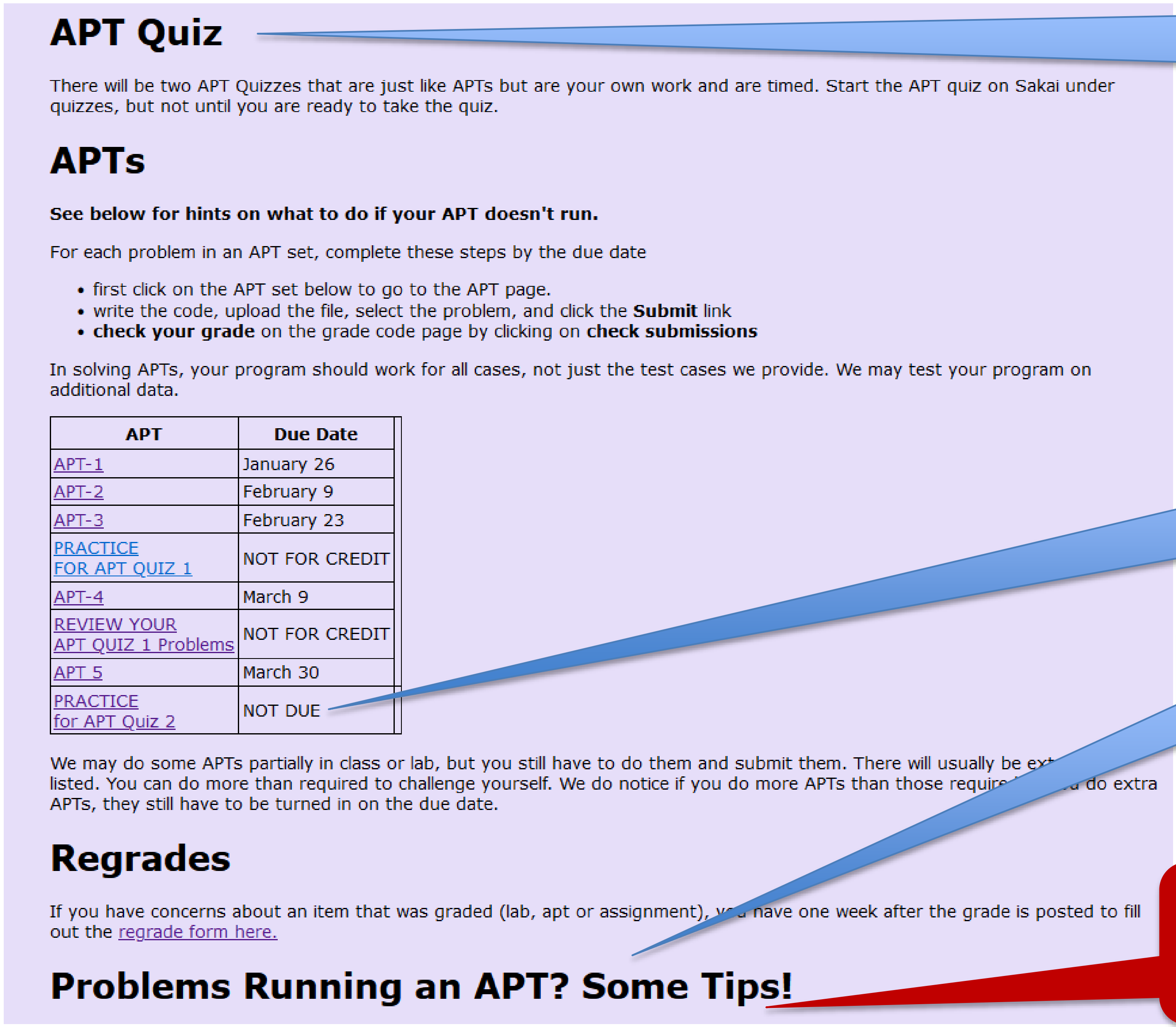
APT	Due Date
APT-1	January 26
APT-2	February 9
APT-3	February 23
PRACTICE FOR APT QUIZ 1	NOT FOR CREDIT
APT-4	March 9
REVIEW YOUR APT QUIZ 1 Problems	NOT FOR CREDIT
APT 5	March 30
PRACTICE for APT Quiz 2	NOT DUE

We may do some APTs partially in class or lab, but you still have to do them and submit them. There will usually be extra apts listed. You can do more than required to challenge yourself. We do notice if you do more APTs than those required. If you do extra APTs, they still have to be turned in on the due date.

Regrades

If you have concerns about an item that was graded (lab, apt or assignment), you have one week after the grade is posted to fill out the regrade form here.

Problems Running an APT? Some Tips!



APT Quiz

There will be two APT Quizzes that are just like APTs but are your own work and are timed. Start the APT quiz on Sakai under quizzes, but not until you are ready to take the quiz.

APTs

See below for hints on what to do if your APT doesn't run.

For each problem in an APT set, complete these steps by the due date

- first click on the APT set below to go to the APT page.
- write the code, upload the file, select the problem, and click the **Submit** link
- **check your grade** on the grade code page by clicking on **check submissions**

In solving APTs, your program should work for all cases, not just the test cases we provide. We may test your program on additional data.

APT	Due Date
APT-1	January 26
APT-2	February 9
APT-3	February 23
PRACTICE FOR APT QUIZ 1	NOT FOR CREDIT
APT-4	March 9
REVIEW YOUR APT QUIZ 1 Problems	NOT FOR CREDIT
APT 5	March 30
PRACTICE for APT Quiz 2	NOT DUE

We may do some APTs partially in class or lab, but you still have to do them and submit them. There will usually be ext [illegible] listed. You can do more than required to challenge yourself. We do notice if you do more APTs than those require [illegible] do extra APTs, they still have to be turned in on the due date.

Regrades

If you have concerns about an item that was graded (lab, apt or assignment), you have one week after the grade is posted to fill out the regrade form here.

Problems Running an APT? Some Tips!

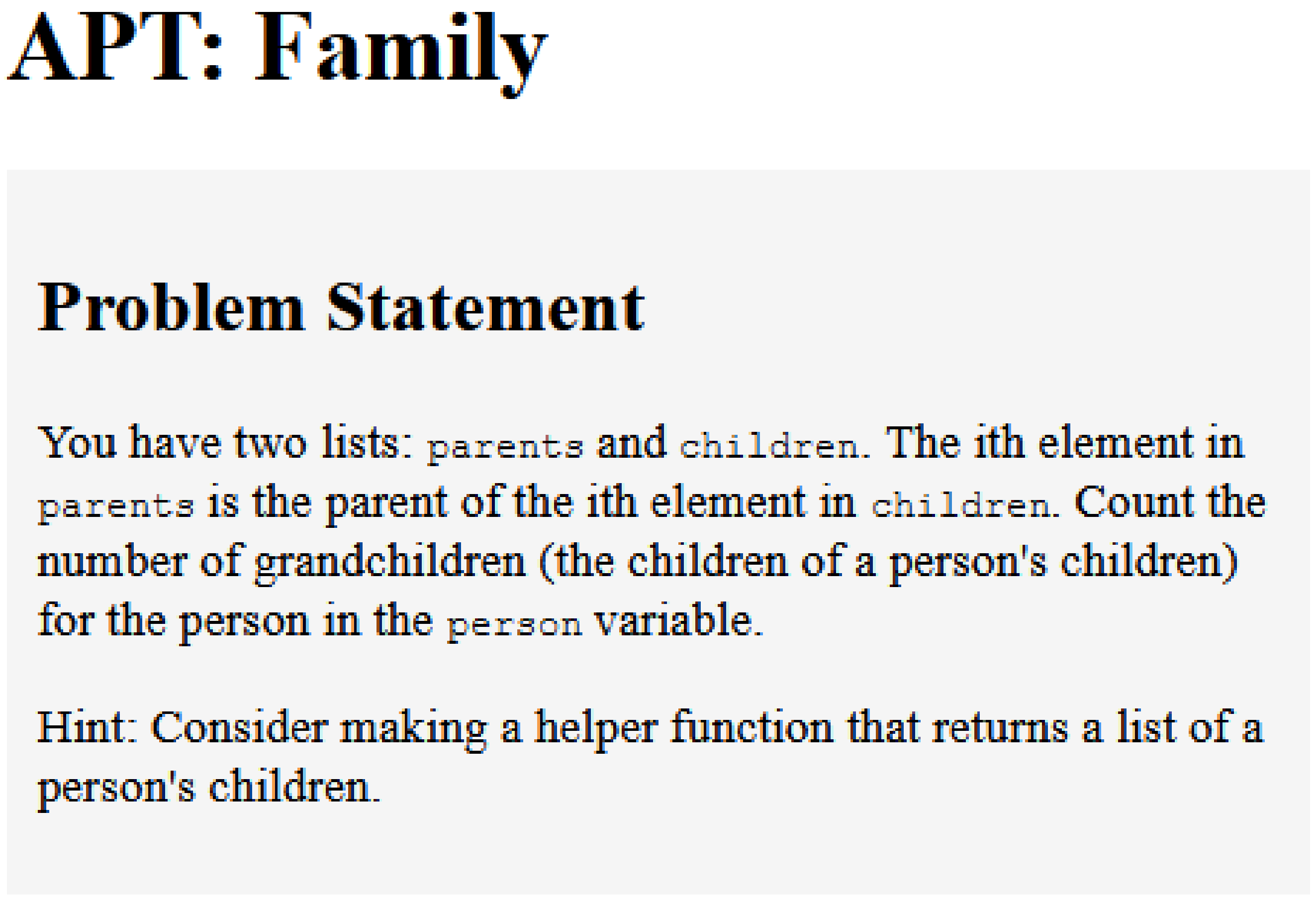
APT Quiz Info

Practice (old APT quiz)

Debugging Tips

Stuck! Use 7 steps!

APT Family



APT: Family

Problem Statement

You have two lists: `parents` and `children`. The ith element in `parents` is the parent of the ith element in `children`. Count the number of grandchildren (the children of a person's children) for the person in the `person` variable.

Hint: Consider making a helper function that returns a list of a person's children.

Step 1: work an example by hand

```
parents = ['Junhua', 'Anshul', 'Junhua', 'Anshul', 'Kerry']
children = ['Anshul', 'Jordan', 'Kerry', 'Paul', 'Kai']
person =  'Junhua'

Returns 3
```

Step 1: work an example by hand

```
parents = ['Junhua', 'Anshul', 'Junhua', 'Anshul', 'Kerry']
children = ['Anshul', 'Jordan', 'Kerry', 'Paul', 'Kai']
person =  'Junhua'

Returns 3
```

Find Junhua's grandchildren

- **First find the children of Junhua**
  - Loop over parents list
    - If name is Junhua add corresponding child to list
      - – How do I do that? I need an index (parallel lists)
    - Kids are [‘Anshul’, ‘Kerry’]
  - For each kid:
    - Loop over parents list:
      - – If name is kid’s name add their child to the list
        - » How do I do that? I need an index (parallel lists)
    - ‘Anshul’s kids -> ‘Jordan’ and ‘Paul’
    - Kerry’s kids -> ‘Kai’
  - Return 3

Step 1: work an example by hand

```
parents = ['Junhua', 'Anshul', 'Junhua', 'Anshul', 'Kerry']
children = ['Anshul', 'Jordan', 'Kerry', 'Paul', 'Kai']
person =  'Junhua'

Returns 3
```

Notice anything?

- **First find the children of Junhua**
  - Loop over parents list
    - If name is Junhua add corresponding child to list
      - How do I do that? I need an index (parallel lists)
    - Kids are ['Anshul', 'Kerry']
  - For each kid:
    - Loop over parents list:
      - If name is kid's name add their child to the list
        - » How do I do that? I need an index (parallel lists)
    - 'Anshul's kids -> 'Jordan' and 'Paul'
    - Kerry's kids -> 'Kai'
  - Return 3

They are the same!

Write a helper function!

Helper function

```
def childrenOf(parents, children, name):
    <missing code to traverse parallel lists>
    return list of name's children
```

How to traverse parallel lists?

```
parents:   ['Junhua', 'Anshul', 'Junhua', 'Anshul', 'Kerry’]
children:  ['Anshul', 'Jordan’,  'Kerry’,    'Paul’,     'Kai’]
                 0           1          2          3          4
```

How to traverse parallel lists?

parents:	['Junhua',	'Anshul',	'Junhua',	'Anshul',	'Kerry’]
children:	['Anshul',	'Jordan’,	'Kerry’,	'Paul’,	'Kai’]
	0	1	2	3	4

Iterate over the list – need a loop!

Need to access same position in each list

- need an index

Use a while loop with an index!

How to traverse parallel lists?

```
parents:   ['Junhua', 'Anshul', 'Junhua', 'Anshul', 'Kerry’]
children:  ['Anshul', 'Jordan’,  'Kerry’,   'Paul’,    'Kai’]
               0          1         2         3         4
```

Build a list of children
Initialize list
Update list

```
index = 0
while index < len(parents):
    <do something>
    index += 1              # update index
```

Assignment 5 - How to play Guess Word Cleverly

- **Make it hard for the player to win!**

- **One way: Try hard words to guess?**
  - "jazziest", "joking", "bowwowing"

- **Another Way: Keep changing the word, sortof**

😉

Clever GuessWord

- **Current GuessWord: Pick random secret word**
  - User starts guessing

- **Can you change secret word?**
  - Yes, but must have letters in same place you have told user
    - Change consistent with all guesses
  - Make the user work harder to guess!

Programming A Clever Game

- **Instead of guessing a word, you're guessing a *group, category*, or *equivalence class* of words**

  Ex: `_ _ _ _ _` and user guesses 'a’

- **`["asked", "adult", "aided", … "axiom"]`**
  - *209 words 'a' as first letter and the only 'a'*
- **`["baked", "cacti", "false", … "walls"]`**
  - *665 words 'a' as second letter and the only 'a'*
- **`["beets", "humor", … "spoof"]`**
  - *2,431 words with no 'a'*
- **What should our secret word be? "asked" ,"baked" or "beets“?**

Programming A Clever Game

- **Instead of guessing a word, you're guessing a *group, category*, or *equivalence class* of words**

  Ex: _ _ _ _ _ and user guesses 'a'

- `["asked", "adult", "aided", … "axiom"]`
  - *209 words 'a' as first letter and the only 'a'*
- `["baked", "cacti", "false", … "walls"]`
  - *665 words 'a' as second letter and the only 'a'*
- `["beets", "humor", … "spoof"]`
  - *2,431 words with no 'a'*

Most words

- **What should our secret word be? "asked" ,"baked" or "beets“?**

Tell user there is no 'a'

Sometimes there will be letters

- **The letter “u” has been guessed and is the 2nd letter**
  Ex: `_ u _ _ _` and user guesses ‘r’

- `["ruddy", "rummy", "rungs", … "rusty"]`
  - *5 words start with “ru” and no other “r” or “u”*
- `["burch", "burly", "burns", … "turns"]`
  - *17 words only ‘u' as second letter and only ‘r’ third letter*
- `["bucks", "bucky", … "tufts"]`
  - *98 words with only “u” second letter and no ‘r'*
- **What should our secret word be? "ruddy" ,"burch" or "bucks“?**

Sometimes there will be letters

- **The letter "u" has been guessed and is the 2nd letter**
  Ex: `_ u _ _ _` and user guesses 'r'

- `["ruddy", "rummy", "rungs", … "rusty"]`
  - *5 words start with "ru" and no other "r" or "u"*
- `["burch", "burly", "burns", … "turns"]`
  - *17 words only 'u' as second letter and only 'r' third letter*
- `["bucks", "bucky", … "tufts"]`
  - *98 words with only "u" second letter and no 'r'*
- **What should our secret word be? "ruddy" ,"burch" or "bucks"?**

Most words

Tell user there is no 'r'

More Details on Game

- **Current secret 8-letter word at random is *catalyst***
  - User guesses 'a', what should computer do?
  - Print `_ a _ a _ _ _ _` and continue?

More Details on Game

- **Current secret 8-letter word at random is *catalyst***
  - User guesses 'a', what should computer do?
  - Print `_ a _ a _ _ _ _` and continue?

No!
Try to change the word!
Best choice may be to tell the user there is no ‘a’

More Details on Game

- **Current secret 8-letter word at random is *catalyst***
  - User guesses 'a', what should computer do?
  - Print `_ a _ a _ _ _ _` and continue?
- **Look at all groups of words and decide on a new word that is more likely to stump player**
  - Why ***"designed"*** better choice than ***"tradeoff"***?
  - 3,475 words with no 'a', 498 with 'a' 3rd letter

Pick category with largest number of words!

Creating Groups/Categories

- **For each of 7,070 words (8 letters), given word and ‘a’, find its group, represented by a template**
- **Use dictionary**
  - Template is KEY, the VALUE is a list of matching words
- **Choose biggest list**
- **Repeat**
- **# words smaller over time**

Group/Template	Size of Group
`_ a _ _ _ _ _ _`	587
`_ a _ a _ _ _ _`	63
`_ _ a _ _ _ _ _`	498
`_ _ _ a _ _ _ _`	406
`_ _ _ _ _ _ _ _`	3,475

Changes to Regular GuessWord

- **List of words from which secret word chosen**
  - Initially this is all words of specified length
    - User will specify the length of the word to guess
  - After each guess, word list is a new subset
- **Keep some functions, modify some, write new ones**
- ***Changes go in another function* to minimize changes to working program**
  - Minimizing changes helps minimize introducing bugs into a working program

Play a game

- `_ _ _ _ _ _`
- Secret word is:
  - **`flamer`**
- User guesses:
  - a
- Possible words:
  - 6166

```
______ : 3441
_____a : 80
____a_ : 233
___a__ : 316
___a_a : 11
__a___ : 549
__a__a : 19
__a_a_ : 10
__aa__ : 1
_a____ : 962
_a___a : 39
_a__a_ : 57
_a_a__ : 40
_a_a_a : 12
_a_aa_ : 3
a_____ : 273
a____a : 21
a___a_ : 30
a__a__ : 32
a__a_a : 3
a_a___ : 26
a_a__a : 7
aa____ : 1
```

Play a game

- _ _ _ _ _ _ _
- Secret word is:
  - **`flamer`**
- User guesses:
  - a
- Possible words:
  - 6166

You build a dictionary for all the possible places an a can be in a word

Keys in dictionary

23 keys

```
______ : 3441
_____a : 80
____a_ : 233
___a__ : 316
___a_a : 11
__a___ : 549
__a__a : 19
__a_a_ : 10
__aa__ : 1
_a____ : 962
_a___a : 39
_a__a_ : 57
_a_a__ : 40
_a_a_a : 12
_a_aa_ : 3
a_____ : 273
a____a : 21
a___a_ : 30
a__a__ : 32
a__a_a : 3
a_a___ : 26
a_a__a : 7
aa____ : 1
```

Play a game

- _ _ _ _ _ _
- Secret word is:
  - **`flamer`**
- User guesses:
  - a
- Possible words:
  - 6166

Each value in dictionary is a list of words

These are the length of each value/list

______ :	3441
_____a :	80
____a_ :	233
___a__ :	316
___a_a :	11
__a___ :	549
__a__a :	19
__a_a_ :	10
__aa__ :	1
[illegible] :	962
[illegible] :	39
_a__a_ :	57
_a_a__ :	40
_a_a_a :	12
_a_aa_ :	3
a_____ :	273
a____a :	21
a___a_ :	30
a__a__ :	32
a__a_a :	3
a_a___ :	26
a_a__a :	7
aa____ :	1

Consider “_ _ _ a _ a” : 11

- **Means “_ _ _ a _ a” is key in dictionary**
- **The value is a list of 11 words**
  - have “a’ in 4th and 6th position

“_ _ _ a _ a”

['cicada', 'errata', 'guiana', 'guyana', 'ithaca’, 'lusaka', 'nevada', 'ottawa', 'sonata', 'tirana', 'urbana']

Consider “_ _ _ a _ a” : 11

- **Means “_ _ _ a _ a” is key in dictionary**
- **The value is a list of 11 words**
    - have “a’ in 4th and 6th position

“_ _ _ a _ a”

key in dictionary

value in dictionary

['cicada', 'errata', 'guiana', 'guyana', 'ithaca’, 'lusaka', 'nevada', 'ottawa', 'sonata', 'tirana', 'urbana']

Play a game

- _ _ _ _ _ _
- Secret word is:
  - **flamer**
- User guesses:
  - a
- Possible words:
  - 6166
- Tell user: NO 'a'

Largest category

```
______ : 3441
_____a : 80
____a_ : 233
___a__ : 316
___a_a : 11
__a___ : 549
__a__a : 19
__a_a_ : 10
__aa__ : 1
_a____ : 962
_a___a : 39
_a__a_ : 57
_a_a__ : 40
_a_a_a : 12
_a_aa_ : 3
a_____ : 273
a____a : 21
a___a_ : 30
a__a__ : 32
a__a_a : 3
a_a___ : 26
a_a__a : 7
aa____ : 1
```

Pick new secret word, any letter without 'a'

Play a game

- `_ _ _ _ _ _`
- Secret word is:
  - **`flamer`**
- User guesses:
  - a
- Possible words:
  - 6166
- Tell user: NO ‘a’

```
______ : 3441
_____a : 30
       : 233
    __ : 316
___a_a : 11
__a___ : 549
__a__a : 19
__a_a_ : 10
__aa__ : 1
_a____ : 962
_a___a : 39
_a__a_ : 57
_a_a__ : 40
_a_a_a : 12
_a_aa_ : 3
a_____ : 273
a____a : 21
a___a_ : 30
a__a__ : 32
a__a_a : 3
a_a___ : 26
a_a__a : 7
aa____ : 1
```

This list of words becomes the "possible words" list.

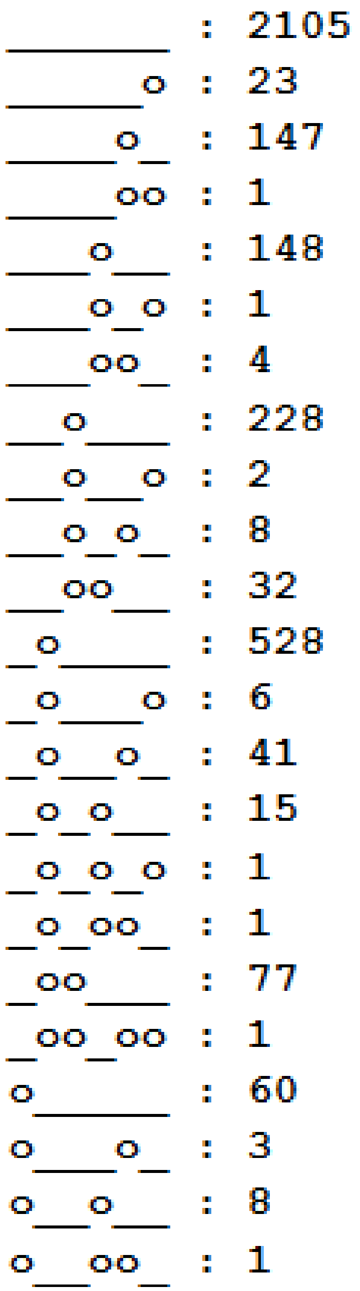
That list is smaller, has 3441 words

List of words has no a's

Pick new secret word, any letter without ‘a’

Play a game

- _ _ _ _ _ _
- Secret word is:
  - `mounds`
- User guesses:
  - o
- Possible words:
  - 3441

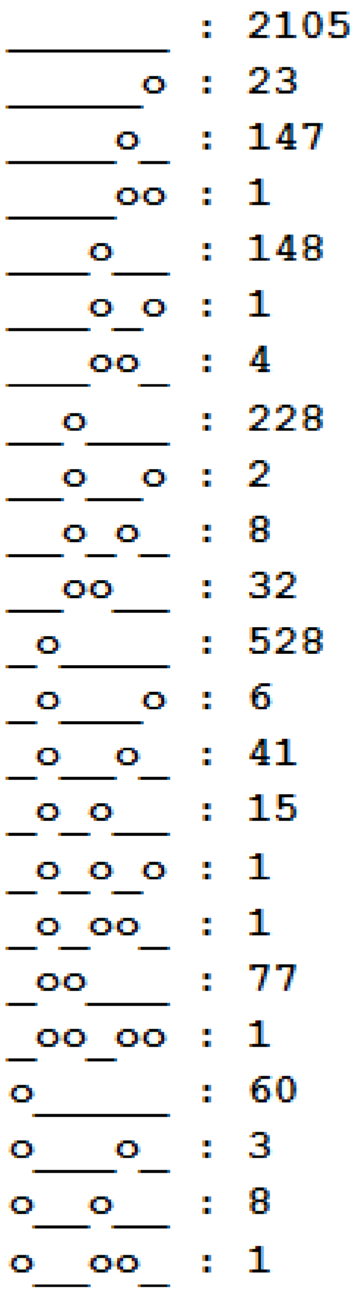


```
______ : 2105
_____o : 23
____o_ : 147
____oo : 1
___o__ : 148
___o_o : 1
___oo_ : 4
__o___ : 228
__o__o : 2
__o_o_ : 8
__oo__ : 32
_o____ : 528
_o___o : 6
_o__o_ : 41
_o_o__ : 15
_o_o_o : 1
_o_oo_ : 1
_oo___ : 77
_oo_oo : 1
o_____ : 60
o___o_ : 3
o__o__ : 8
o__oo_ : 1
```

Play a game

- _ _ _ _ _ _
- Secret word is:
  - **`mounds`**
- User guesses:
  - o
- Possible words:
  - 3441

Note: None of these lists have the letter ‘a’ in them. We removed all words that have ‘a’ from our list of words



```
______ : 2105
_____o : 23
____o_ : 147
____oo : 1
___o__ : 148
___o_o : 1
___oo_ : 4
__o___ : 228
__o__o : 2
__o_o_ : 8
__oo__ : 32
_o____ : 528
_o___o : 6
_o__o_ : 41
_o_o__ : 15
_o_o_o : 1
_o_oo_ : 1
_oo___ : 77
_oo_oo : 1
o_____ : 60
o___o_ : 3
o__o__ : 8
o__oo_ : 1
```

Play a game

Largest category

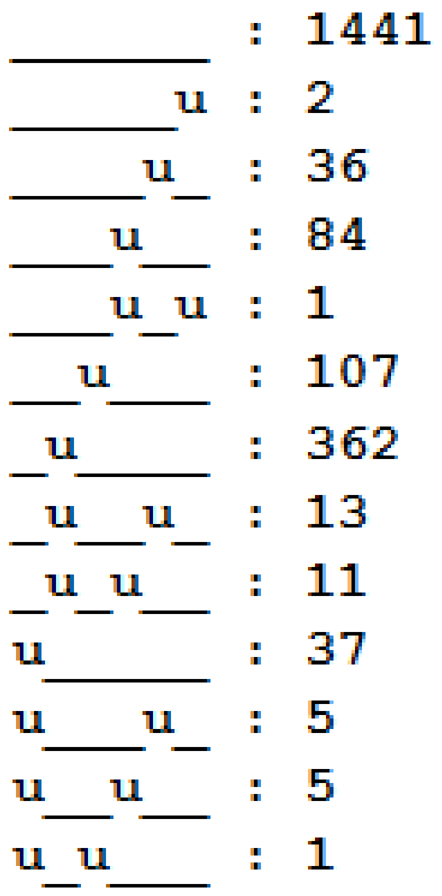
- _ _ _ _ _ _
- Secret word is:
  - `mounds`
- User guesses:
  - o
- Possible words:
  - 3441
- Tell user no 'o'

Pick new secret word, any letter without 'o'

```
______ : 2105
_____o : 23
____o_ : 147
____oo : 1
___o__ : 148
___o_o : 1
___oo_ : 4
__o___ : 228
__o__o : 2
__o_o_ : 8
__oo__ : 32
_o____ : 528
_o___o : 6
_o__o_ : 41
_o_o__ : 15
_o_o_o : 1
_o_oo_ : 1
_oo___ : 77
_oo_oo : 1
o_____ : 60
o___o_ : 3
o__o__ : 8
o__oo_ : 1
```

Play a game

- `_ _ _ _ _ _`
- Secret word is:
  - **`burkes`**
- User guesses:
  - u
- Possible words:
  - 2105



```
______ : 1441
_____u : 2
____u_ : 36
___u__ : 84
___u_u : 1
__u___ : 107
_u____ : 362
_u__u_ : 13
_u_u__ : 11
u_____ : 37
u___u_ : 5
u__u__ : 5
u_u___ : 1
```

Play a game

Largest category

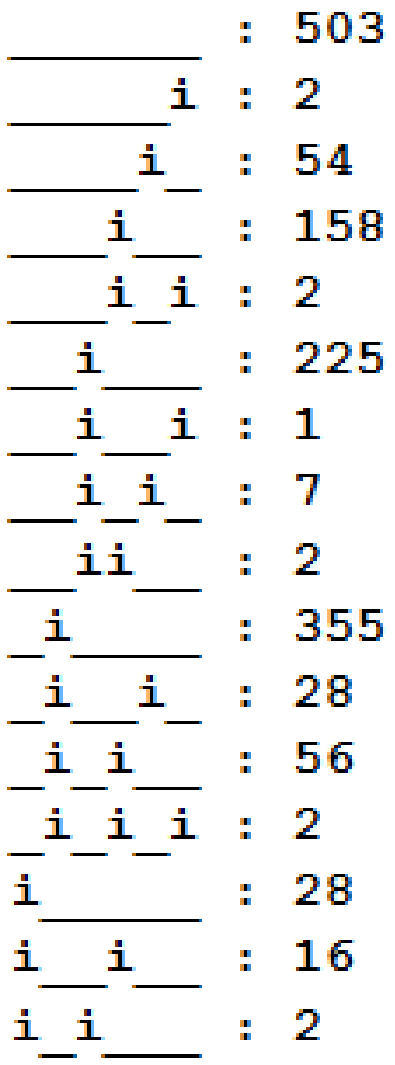
- `------`
- Secret word is:
  - **`burkes`**
- User guesses:
  - u
- Possible words:
  - 2105
- Tell user no ‘u’

```
______ : 1441
_____u : 2
____u_ : 36
___u__ : 84
___u_u : 1
__u___ : 107
_u____ : 362
_u__u_ : 13
_u_u__ : 11
u_____ : 37
u___u_ : 5
u__u__ : 5
u_u___ : 1
```

Pick new secret word, any letter without ‘u’

Play a game

- _ _ _ _ _ _
- Secret word is:
  - **`wilted`**
- User guesses:
  - i
- Possible words:
  - 1441



```
______ : 503
_____i : 2
____i_ : 54
___i__ : 158
___i_i : 2
__i___ : 225
__i__i : 1
__i_i_ : 7
__ii__ : 2
_i____ : 355
_i__i_ : 28
_i_i__ : 56
_i_i_i : 2
i_____ : 28
i__i__ : 16
i_i___ : 2
```

Play a game

Largest category

- _ _ _ _ _ _
- Secret word is:
  - **`wilted`**
- User guesses:
  - i
- Possible words:
  - 1441
- Tell user no 'i'

Pick new secret word, any letter without 'i'

```
______ : 503
_____i : 2
____i_ : 54
___i__ : 158
___i_i : 2
__i___ : 225
__i__i : 1
__i_i_ : 7
__ii__ : 2
_i____ : 355
_i__i_ : 28
_i_i__ : 56
_i_i_i : 2
i_____ : 28
i__i__ : 16
i_i___ : 2
```

Play a game

- `------`
- Secret word is:
  - **`served`**
- User guesses:
  - e
- Possible words:
  - 503

```
______ : 2
_____e : 5
____e_ : 13
___e__ : 9
___e_e : 2
___ee_ : 5
__e___ : 42
__e__e : 12
__e_e_ : 23
__ee__ : 36
__ee_e : 9
_e____ : 13
_e___e : 13
_e__e_ : 160
_e__ee : 2
```

```
_e_e__ : 59
_e_e_e : 7
_e_ee_ : 3
_ee___ : 6
_ee__e : 5
_ee_e_ : 34
e____e : 1
e___e_ : 5
e___ee : 2
e__e__ : 20
e__ee_ : 2
e_e___ : 9
e_e__e : 1
e_e_e_ : 3
```

Play a game

- `_ _ _ _ _ _`
- Secret word is:
  - **`served`**
- User guesses:
  - e
- Possible words:
  - 503
- Tell user 'e' in these two places

```
______ : 2
_____e : 5
____e_ : 13
___e__ : 9
___e_e : 2
___ee_ : 5
__e___ : 42
__e__e : 12
__e_e_ : 23
__ee__ : 36
__ee_e : 9
_e____ : 13
_e___e : 13
_e__e_ : 160
_e__ee : 2
```

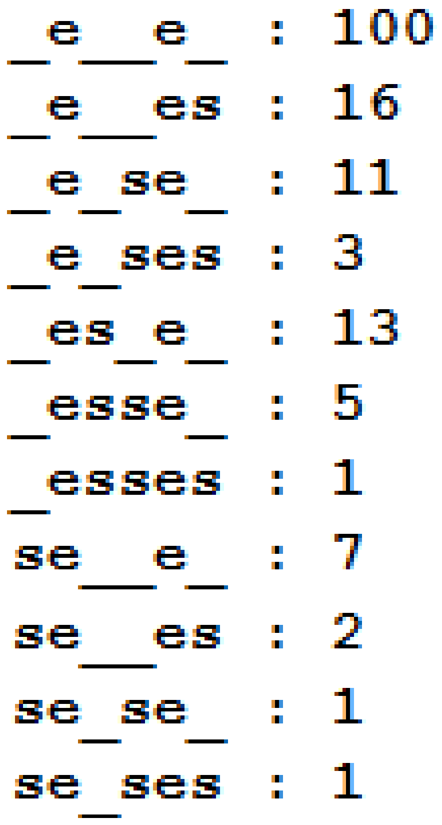
Largest category

```
_e_e__ : 59
_e_e_e : 7
_e_ee_ : 3
_ee___ : 6
_ee__e : 5
_ee_e_ : 34
e____e : 1
e___e_ : 5
e___ee : 2
e__e__ : 20
e__ee_ : 2
e_e___ : 9
e_e__e : 1
e_e_e_ : 3
```

Pick new secret word with 'e' in 2nd and 5th positions

Play a game

- **`_e__e_`**
- Secret word is:
  - **`tested`**
- User guesses:
  - s
- Possible words:
  - 160



```
_e__e_ : 100
_e__es : 16
_e_se_ : 11
_e_ses : 3
_es_e_ : 13
_esse_ : 5
_esses : 1
se__e_ : 7
se__es : 2
se_se_ : 1
se_ses : 1
```

Play a game

Largest category

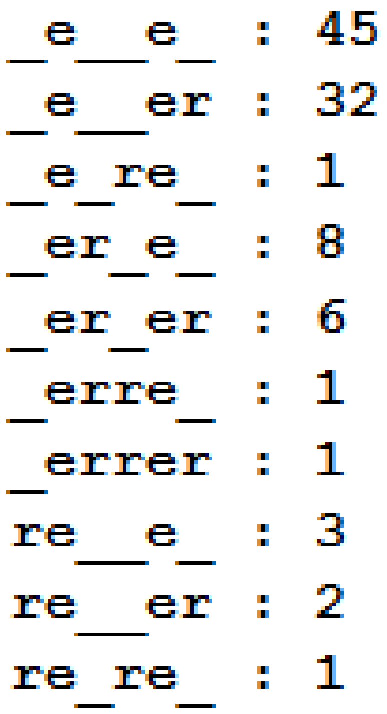
- **_e__e_**
- Secret word is:
  - **tested**
- User guesses:
  - s
- Possible words:
  - 160
- Tell user no ‘s’

```
_e__e_ : 100
_e__es : 16
_e_se_ : 11
_e_ses : 3
_es_e_ : 13
_esse_ : 5
_esses : 1
se__e_ : 7
se__es : 2
se_se_ : 1
se_ses : 1
```

Pick new secret word with no ‘s’ in it

Play a game

- **`_e__e_`**
- Secret word is:
    - **`kepler`**
- User guesses:
    - r
- Possible words:
    - 100



```
_e__e_ : 45
_e__er : 32
_e_re_ : 1
_er_e_ : 8
_er_er : 6
_erre_ : 1
_errer : 1
re__e_ : 3
re__er : 2
re_re_ : 1
```

Play a game

Largest category

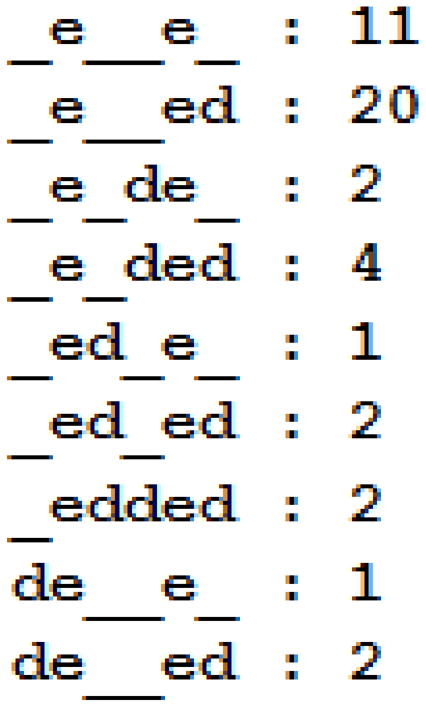
- **`_e__e_`**
- Secret word is:
  - **`kepler`**
- User guesses:
  - r
- Possible words:
  - 100
- Tell user no ‘r’

```
_e__e_ : 45
_e__er : 32
_e_re_ : 1
_er_e_ : 8
_er_er : 6
_erre_ : 1
_errer : 1
re__e_ : 3
re__er : 2
re_re_ : 1
```

Pick new secret word with no ‘r’ in it

Play a game

- **`_e__e_`**
- Secret word is:
  - **`wedded`**
- User guesses:
  - d
- Possible words:
  - 45



```
_e__e_ : 11
_e__ed : 20
_e_de_ : 2
_e_ded : 4
_ed_e_ : 1
_ed_ed : 2
_edded : 2
de__e_ : 1
de__ed : 2
```

Play a game

Largest category

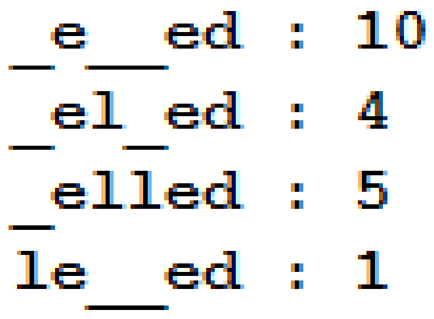
- **_e__e_**
- Secret word is:
  - **wedded**
- User guesses:
  - d
- Possible words:
  - 45
- Tell user last letter is ‘d’

```
_e__e_ : 11
_e__ed : 20
_e_de_ : 2
_e_ded : 4
_ed_e_ : 1
_ed_ed : 2
_edded : 2
de__e_ : 1
de__ed : 2
```

Pick new secret word with ‘d’ as last letter

Play a game

- **_ e _ _ e d**
- Secret word is:
  - **belted**
- User guesses:
  - l
- Possible words:
  - 20



```
_e__ed : 10
_el_ed : 4
_elled : 5
le__ed : 1
```

Play a game

Largest category

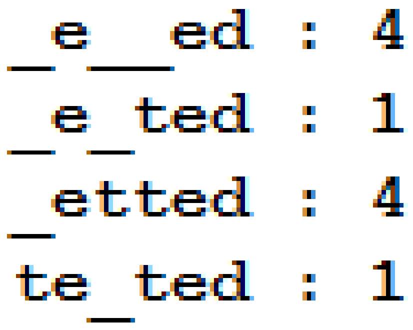
- **_e__ed**
- Secret word is:
  - **belted**
- User guesses:
  - l
- Possible words:
  - 20
  - Tell user no ‘l’

```
_e__ed : 10
_el_ed : 4
_elled : 5
le__ed : 1
```

Pick new secret word with no ‘l’ in it

Play a game

- `_ e _ _ e d`
- Secret word is:
  - **`vented`**
- User guesses:
  - t
- Possible words:
  - 4

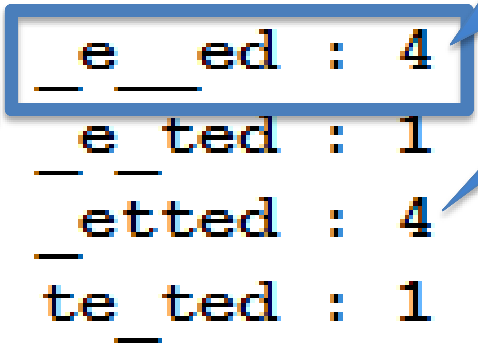


```
_e__ed : 4
_e_ted : 1
_etted : 4
te_ted : 1
```

Play a game

Largest category

- **`_ e _ _ e d`**
- Secret word is:
  - **`vented`**
- User guesses:
  - t
- Possible words:
  - 4
  - Tell user no 't'



```
_e__ed : 4
_e_ted : 1
_etted : 4
te_ted : 1
```

Largest is a tie, randomly pick one of them

It is really hard to win!

That is 10 tries, Game Over!

Greedy Algorithms

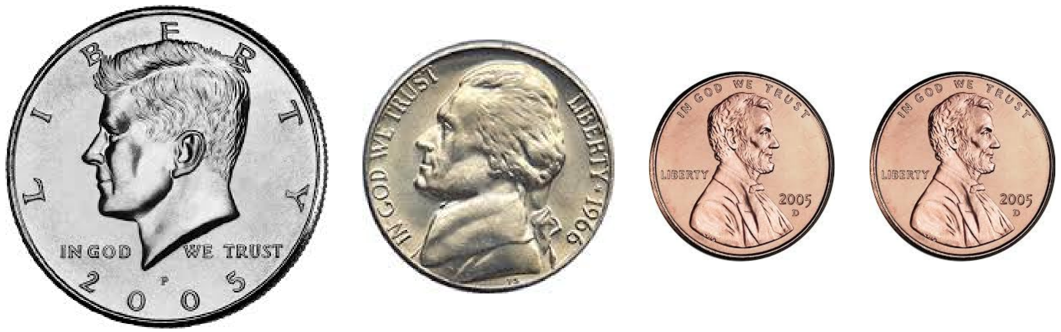
- **“Choosing largest group” -> *greedy algorithm***
  - Make a locally optimal decision that works in the long run
  - Choose largest group to make game last …

- **Greed as in “it chooses the best current choice every time, which results in getting the best overall result”**

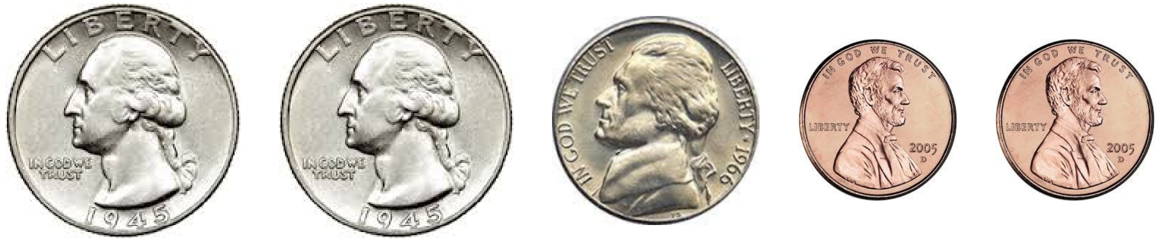
- **Canonical example? Change with coins**
  - Minimize # coins given for change: 57 cents

Making change for 57 cents

- **When choose next coin, always pick biggest**
- **With half-dollar coins**



- **With quarters and no half dollars**



Making change for 57 cents

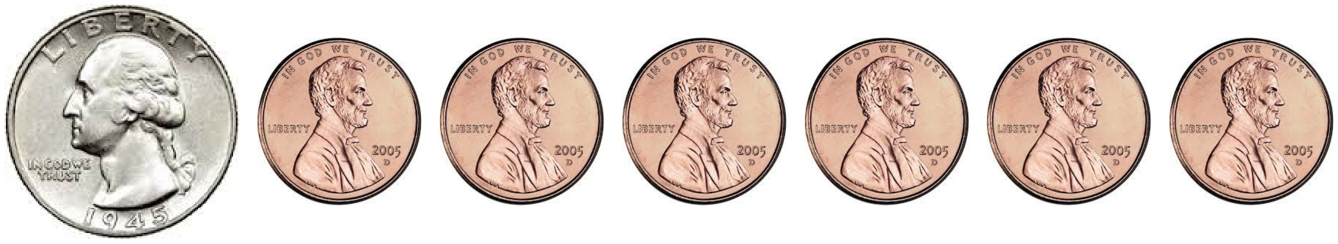
- **When choose next coin, always pick biggest**
- **With half-dollar coins**

- **With quarters and no half dollars**

Always get minimum number of coins

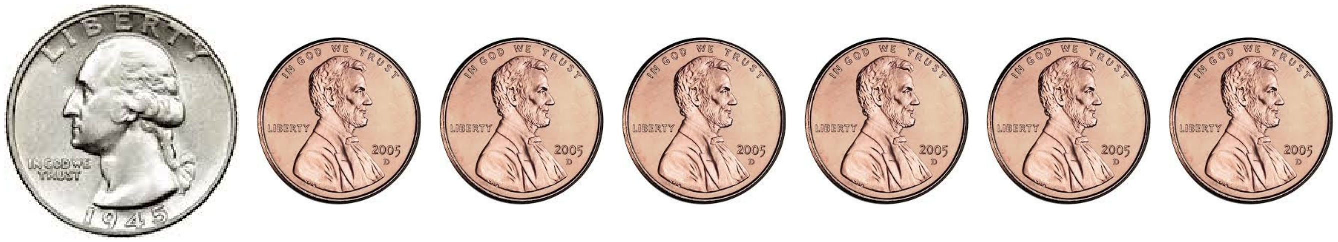
When greedy doesn't work

- **What if no nickels? Making change for 31 cents:**

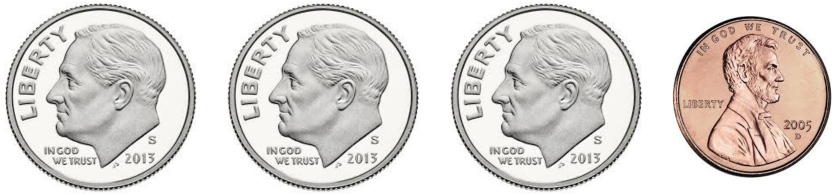


When greedy doesn't work

- **What if no nickels? Making change for 31 cents:**



- **Can we do better? Yes!**



Woto-1 Clever GuessWord
http://bit.ly/101s23-0328-1

More Problem Solving with Dictionaries, Sets and lists

Movie Actors

Each list in datalist has 5 strings:
Movie, Actor, Year of movie, minutes total, minutes Actor in movie

```
datalist = [
[’Saving Mr. Banks’, ’Tom Hanks’, ’2016’, ’125’, ’65’],
[’Saving Mr. Banks’, ’Emma Thompson’, ’2016’, ’125’, ’84’],
[’Enough Said’, ’James Gandolfini’, ’2013’, ’93’, ’52’],
[’Captain Phillips’, ’Catherine Keener’, ’2013’, ’134’, ’22’],
[’The Da Vinci Code’, ’Tom Hanks’, ’2006’, ’149’, ’85’],
[’Saving Mr. Banks’, ’Colin Farrell’, ’2016’, ’125’, ’25’],
[’Forrest Gump’, ’Sally Field’, ’1994’, ’142’, ’56’],
[’Mrs. Doubtfire’, ’Robin Williams’, ’1993’, ’125’, ’94’],
[’Captain Phillips’, ’Tom Hanks’, ’2013’, ’134’, ’110’],
[’Enough Said’, ’Catherine Keener’, ’2013’, ’93’, ’21’],
[’The Da Vinci Code’, ’Ian McKellen’, ’2006’, ’149’, ’60’],
[’Hello, My Name is Doris’, ’Sally Field’, ’2015’, ’95’, ’84’],
[’Alone in Berlin’, ’Emma Thompson’, ’2016’, ’103’, ’70’],
[’Forrest Gump’, ’Tom Hanks’, ’1994’, ’142’, ’110’],
[’Mrs. Doubtfire’, ’Sally Field’, ’1993’, ’125’, ’45’] ]
```

Movie Actors

```
['Saving Mr. Banks', 'Tom Hanks', '2016', '125', '65'],
```

- **For example in first list:**
  - Movie is 'Saving Mr. Banks'
  - Actor is "Tom Hanks"
  - The movie was released in 2016
  - The movie is 125 minutes long
  - Tom Hanks is on screen for 65 minutes

Woto-2 ActorsNotIn http://bit.ly/101s23-0328-2

- **Write**
  - def actors(datalist) – returns a sorted unique list of actors
  - def actorsNotIn(datalist, actorlist)
    - Actorlist is a list of favorite actors
    - Returns a sorted unique list of actors that are in actorlist but not in datalist
    - If favorite is ["Emma Watson”, ”Daniel Radcliffe”, ”Ralph Fiennes”, ”Tom Hanks”] then actorsNotIn

returns:

```
 ['Daniel Radcliffe', 'Ralph Fiennes', 'Emma 
Watson']
```

Woto-2 ActorsNotIn
http://bit.ly/101s23-0328-2

Code for actors

```
def actors(datalist):
    result = set([])
    for item in datalist:
        result.add(item[1])
    return sorted(list(result))
```

item is a list of five things

Or just
return sorted(result)

list comprehension

```
def actors(datalist):
   return sorted(set([item[1] for item in datalist]))
```

Code for actorsNotIn

```
def actorsNotIn(datalist, actorlist):
    result = set(actors(datalist))
    actorset = set(actorlist)
    diff = actorset - result
    return sorted(diff)
```

Call function actors

Put both lists in sets

Set operation difference

Woto-3 dictActorsToMovies http://bit.ly/101s23-0328-3

- **Write**
  - def dictActorsToMovies(datalist) – returns a dictionary of each actor mapped to a list of tuples, each tuple is a movie and the minutes they were in that movie
  - def actorMostMinutes(datalist)
    - Returns the actor from datalist, that was in movies the most minutes, if a tie, return any one of the tie

Woto-3 dictActorsToMovies
http://bit.ly/101s23-0328-3

dictActorsToMovies

```
def dictActorsToMovies(datalist):
    d = {}
    for item in datalist:
        if item[1] not in d:
            d[item[1]] = [(item[0],item[4])]
        else:
            d[item[1]].append((item[0],item[4]))
    return d
```

First time, must create the list

Already there, append tuple to list

actorMostMinutes

```
def actorMostMinutes(datalist):
    d = dictActorsToMovies(datalist)
    totaltime = 0
    totalactor = ""
    for (key,value) in d.items():
        time = sum([int(t[1]) for t in value])
        if time > totaltime:
            totaltime = time
            totalactor = key
    return totalactor
```

Call function for dictionary

Sum all times for this actor

Keep track of largest time

Keep track of actor with largest time