edX (https://www.edx.org)

MITx: 6.00.1x Introduction to Computer Science and Programming Using Python

zhushun0008 (/dashboard) ▼

Courseware (/courses/MITx/6.00.1_3x/2T2014/courseware)   Updates & News (/courses/MITx/6.00.1_3x/2T2014/info)

Calendar (/courses/MITx/6.00.1_3x/2T2014/89309559b0414f6d8cbef9e48ca19f4b/)   Wiki (/courses/MITx/6.00.1_3x/2T2014/course_wiki)

iscussion (/courses/MITx/6.00.1_3x/2T2014/discussion/forum)   Progress (/courses/MITx/6.00.1_3x/2T2014/progress)

Help

INTRODUCTION - A WORD GAME

In this problem set, you'll implement *two* versions of the 6.00 wordgame!

Don't be intimidated by the length of this problem set. There is a lot of reading, but it can be done with a reasonable amount of thinking and coding. It'll be helpful if you start this problem set a few days before it is due!

Let's begin by describing the 6.00 wordgame: This game is a lot like Scrabble or Words With Friends, if you've played those. Letters are dealt to players, who then construct one or more words out of their letters. Each **valid** word receives a score, based on the length of the word and the letters in that word.

The rules of the game are as follows:

Dealing

- A player is dealt a hand of *n* letters chosen at random (assume *n=7* for now).

- The player arranges the hand into as many words as they want out of the letters, using each letter at most once.

- Some letters may remain unused (these won't be scored).

Scoring

- The score for the hand is the sum of the scores for each word formed.

- The score for a word is the sum of the points for letters in the word, multiplied by the length of the word, plus 50 points if all *n* letters are used on the first word created.

- Letters are scored as in Scrabble; A is worth 1, B is worth 3, C is worth 3, D is worth 2, E is worth 1, and so on. We have defined the dictionary `SCRABBLE_LETTER_VALUES` that maps each lowercase letter to its Scrabble letter value.

- For example, 'weed' would be worth 32 points ((4+1+1+2) for the four letters, then multiply by `len('weed')` to get (4+1+1+2)*4 = 32). Be sure to check that the hand actually has 1 'w', 2 'e's, and 1 'd' before scoring the word!

- As another example, if *n=7* and you make the word 'waybill' on the first try, it would be worth 155 points (the base score for 'waybill' is (4+1+4+3+1+1+1)*7=105, plus an additional 50 point bonus for using all *n* letters).

Sample Output

Here is how the game output will look!

Show Discussion   🖉 New Post

ed**X** (https://www.edx.org)

MITx: 6.00.1x Introduction to Computer Science and Programming Using Python

| zhushun0008 (/dashboard) | ▼ |

Courseware (/courses/MITx/6.00.1_3x/2T2014/courseware)   Updates & News (/courses/MITx/6.00.1_3x/2T2014/info)

Calendar (/courses/MITx/6.00.1_3x/2T2014/89309559b0414f6d8cbef9e48ca19f4b/)   Wiki (/courses/MITx/6.00.1_3x/2T2014/course_wiki)

iscussion (/courses/MITx/6.00.1_3x/2T2014/discussion/forum)   Progress (/courses/MITx/6.00.1_3x/2T2014/progress)

Help

GETTING STARTED

1. Download and save Problem Set 4 (/c4x/MITx/6.00.1_3x/asset/ProblemSet4.zip), a zip file of all the skeleton code you'll be filling in. Extract the files from the zip folder and make sure to save *all* the files - `ps4a.py`, `ps4b.py`, `test_ps4a.py` and `words.txt` - in the **same folder**. We recommend creating a folder in your Documents folder called 6001x, and inside the 6001x folder, creating a separate folder for each problem set. If you don't follow this instruction, you may end up with issues because the files for this problem set depend on one another.

2. Run the file `ps4a.py`, without making any modifications to it, in order to ensure that everything is set up correctly (this means, open the file in IDLE, and use the Run command to load the file into the interpreter). The code we have given you loads a list of valid words from a file and then calls the `playGame` function. You will implement the functions it needs in order to work. If everything is okay, after a small delay, you should see the following printed out:

```
Loading word list from file...
     83667 words loaded.
playGame not yet implemented.
```

If you see an `IOError` instead (e.g., "No such file or directory"), you should change the value of the `WORDLIST_FILENAME` constant (defined near the top of the file) to the **complete pathname** for the file `words.txt` (This will vary based on where you saved the files).

For example, if you saved all the files including this `words.txt` in the directory "C:/Users/Ana/6001x/PS4" change the line:

`WORDLIST_FILENAME = "words.txt"` to something like

`WORDLIST_FILENAME = "C:/Users/Ana/6001x/PS4/words.txt"`

Windows users, if you are copying the file path from Windows Explorer, you will have to change the backslashes to forward slashes.

3. The file `ps4a.py` has a number of already implemented functions you can use while writing up your solution. You can ignore the code between the following comments, though you should read and understand how to use each helper function by reading the docstrings:

```
# ----------------------------------
# Helper code
# You don't need to understand this helper code,
# but you will have to know how to use the functions
# (so be sure to read the docstrings!)
    .
    .
    .
# (end of helper code)
# ----------------------------------
```

Canopy specific instructions: Every time you modify code in `ps4a.py` go to

```
Run -> Restart Kernel (or hit the CTRL with the dot on your keyboard)
```

before running `test_ps4a.py`. **You have to do this every time you modify the file** `ps4a.py` **and want to run the file** `test_ps4a.py`, otherwise changes to the former will not be incorporated in the latter.

4. This problem set is structured so that you will write a number of modular functions and then glue them together to form the complete word playing game. Instead of waiting until the entire game is *ready*, you should test each function you write, individually, before moving on. This approach is known as *unit testing*, and it will help you debug your code.

We have provided several test functions to get you started. After you've written each new function, unit test by running the file `test_ps4a.py` to check your work.

If your code passes the unit tests you will see a `SUCCESS` message; otherwise you will see a `FAILURE` message. These tests aren't exhaustive. You will want to test your code in other ways too.

Try running `test_ps4a.py` now (before you modify the `ps4a.py` skeleton). You should see that all the tests fail, because nothing has been implemented yet.

These are the provided test functions:

`test_getWordScore()`
    Test the `getWordScore()` implementation.

`test_updateHand()`
    Test the `updateHand()` implementation.

`test_isValidWord()`
    Test the `isValidWord()` implementation.

Show Discussion   ☑ New Post

edX (https://www.edx.org)

MITx: 6.00.1x Introduction to Computer Science and Programming Using Python
zhushun0008 (/dashboard) ▼

Courseware (/courses/MITx/6.00.1_3x/2T2014/courseware)   Updates & News (/courses/MITx/6.00.1_3x/2T2014/info)

Calendar (/courses/MITx/6.00.1_3x/2T2014/89309559b0414f6d8cbef9e48ca19f4b/)   Wiki (/courses/MITx/6.00.1_3x/2T2014/course_wiki)

iscussion (/courses/MITx/6.00.1_3x/2T2014/discussion/forum)   Progress (/courses/MITx/6.00.1_3x/2T2014/progress)

Help

WORD SCORES (10 points possible)

The first step is to implement some code that allows us to calculate the score for a single word. The function `getWordScore` should accept as input a string of lowercase letters (a *word*) and return the integer score for that word, using the game's scoring rules.

> A Reminder of the Scoring Rules

HINTS

- You may assume that the input `word` is always either a string of lowercase letters, or the empty string `""` .

- You will want to use the `SCRABBLE_LETTER_VALUES` dictionary defined at the top of `ps4a.py` . You should not change its value.

- Do **not** assume that there are always 7 letters in a hand! The parameter `n` is the number of letters required for a bonus score (the maximum number of letters in the hand). Our goal is to keep the code modular - if you want to try playing your word game with *n=10* or *n=4*, you will be able to do it by simply changing the value of `HAND_SIZE` !

- **Testing:** If this function is implemented properly, and you run `test_ps4a.py` , you should see that the `test_getWordScore()` tests pass. Also test your implementation of `getWordScore` , using some reasonable English words.

Fill in the code for `getWordScore` in `ps4a.py` and be sure you've passed the appropriate tests in `test_ps4a.py` before pasting your function definition here.

Canopy specific instructions: If you modify code in `ps4a.py` go to

```
Run -> Restart Kernel (or hit the CTRL with the dot on your keyboard)
```

before running `test_ps4a.py` . **You have to do this every time you modify the file** `ps4a.py` **and want to run the file** `test_ps4a.py` , otherwise changes to the former will not be incorporated in the latter.

```
 1 def getWordScore(word, n):
 2     """
 3     Returns the score for a word. Assumes the word is a valid word.
 4
 5     The score for a word is the sum of the points for letters in the
 6     word, multiplied by the length of the word, PLUS 50 points if all n
 7     letters are used on the first turn.
 8
 9     Letters are scored as in Scrabble; A is worth 1, B is worth 3, C is
10     worth 3, D is worth 2, E is worth 1, and so on (see SCRABBLE_LETTER_VALUES)
11
12     word: string (lowercase letters)
13     n: integer (HAND_SIZE; i.e., hand size required for additional points)
14     returns: int >= 0
15     """
16     # TO DO     <-- Remove this comment when you code this function
```

Unanswered

Check   Save   *You have used 0 of 30 submissions*

Show Discussion

About (https://www.edx.org/about-us)   Jobs (https://www.edx.org/jobs)
Press (https://www.edx.org/press)   FAQ (https://www.edx.org/student-faq)
Contact (https://www.edx.org/contact)

EdX is a non-profit created by founding partners Harvard and MIT whose mission is to bring the best of higher education to students of all ages anywhere in the world, wherever there is Internet access. EdX's free online MOOCs are interactive and subjects include computer science, public health, and artificial intelligence.

(http://www.meetup.com/edX-Global-Community/)

(http://www.facebook.com/EdxOnline)

(https://twitter.com/edXOnline)

(https://plus.google.com/108235383044095082

(http://youtube.com/user/edxonline)

Terms of Service and Honor Code -
Privacy Policy (https://www.edx.org/edx-privacy-policy)

edX (https://www.edx.org)

MITx: 6.00.1x Introduction to Computer Science and Programming Using Python

zhushun0008 (/dashboard) ▼

Courseware (/courses/MITx/6.00.1_3x/2T2014/courseware)   Updates & News (/courses/MITx/6.00.1_3x/2T2014/info)

Calendar (/courses/MITx/6.00.1_3x/2T2014/89309559b0414f6d8cbef9e48ca19f4b/)   Wiki (/courses/MITx/6.00.1_3x/2T2014/course_wiki)

iscussion (/courses/MITx/6.00.1_3x/2T2014/discussion/forum)   Progress (/courses/MITx/6.00.1_3x/2T2014/progress)

Help

DEALING WITH HANDS (10 points possible)

Please read this problem entirely!! The majority of this problem consists of learning how to read code, which is an incredibly useful and important skill. At the end, you will implement a short function. Be sure to take your time on this problem - it may seem easy, but reading someone else's code can be challenging and this is an important exercise.

REPRESENTING HANDS

A **hand** is the set of letters held by a player during the game. The player is initially dealt a set of random letters. For example, the player could start out with the following hand: `a, q, l, m, u, i, l`. In our program, a hand will be represented as a dictionary: the keys are (lowercase) letters and the values are the number of times the particular letter is repeated in that hand. For example, the above hand would be represented as:

```
hand = {'a':1, 'q':1, 'l':2, 'm':1, 'u':1, 'i':1}
```

Notice how the repeated letter `l` is represented. Remember that with a dictionary, the usual way to access a value is `hand['a']`, where `'a'` is the key we want to find. However, this only works if the key is in the dictionary; otherwise, we get a `KeyError`. To avoid this, we can use the call `hand.get('a',0)`. This is the "safe" way to access a value if we are not sure the key is in the dictionary. `d.get(key,default)` returns the value for `key` if `key` is in the dictionary `d`, else `default`. If `default` is not given, it returns `None`, so that this method never raises a `KeyError`. For example:

```
>>> hand['e']
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'e'
>>> hand.get('e', 0)
0
```

CONVERTING WORDS INTO DICTIONARY REPRESENTATION

One useful function we've defined for you is `getFrequencyDict`, defined near the top of `ps4a.py`. When given a string of letters as an input, it returns a dictionary where the keys are letters and the values are the number of times that letter is represented in the input string. For example:

```
>>> getFrequencyDict("hello")
{'h': 1, 'e': 1, 'l': 2, 'o': 1}
```

As you can see, this is the same kind of dictionary we use to represent hands.

DISPLAYING A HAND

Given a hand represented as a dictionary, we want to display it in a user-friendly way. We have provided the implementation for this in the `displayHand` function. Take a few minutes right now to read through this function carefully and understand what it does and how it works.

GENERATING A RANDOM HAND

The hand a player is dealt is a set of letters chosen at random. We provide you with the implementation of a function that generates this random hand, `dealHand`. The function takes as input a positive integer `n`, and returns a new object, a hand containing `n` lowercase letters. Again, take a few minutes (right now!) to read through this function carefully and understand what it does and how it works.

REMOVING LETTERS FROM A HAND (YOU IMPLEMENT THIS)

The player starts with a hand, a set of letters. As the player spells out words, letters from this set are used up. For example, the player could start out with the following hand: `a, q, l, m, u, i, l`. The player could choose to spell the word `quail`. This would leave the following letters in the player's hand: `l, m`. Your task is to implement the function `updateHand`, which takes in two inputs - a `hand` and a `word` (string). `updateHand` uses letters from the hand to spell the word, and then returns a copy of the `hand`, containing only the letters remaining. For example:

```
>>> hand = {'a':1, 'q':1, 'l':2, 'm':1, 'u':1, 'i':1}
>>> displayHand(hand) # Implemented for you
a q l l m u i
>>> hand = updateHand(hand, 'quail') # You implement this function!
>>> hand
{'a':0, 'q':0, 'l':1, 'm':1, 'u':0, 'i':0}
>>> displayHand(hand)
l m
```

Implement the `updateHand` function. Make sure this function has no side effects: i.e., it must not mutate the hand passed in. Before pasting your function definition here, be sure you've passed the appropriate tests in `test_ps4a.py`.

Hints
Testing
Copying Dictionaries

Your implementation of updateHand should be short (ours is 4 lines of code). It does not need to call any helper functions.

Canopy specific instructions: If you modify code in `ps4a.py` go to

```
Run -> Restart Kernel (or hit the CTRL with the dot on your keyboard)
```

before running `test_ps4a.py`. **You have to do this every time you modify the file `ps4a.py` and want to run the file `test_ps4a.py`**, otherwise changes to the former will not be incorporated in the latter.

```
 1 def updateHand(hand, word):
 2     """
 3     Assumes that 'hand' has all the letters in word.
 4     In other words, this assumes that however many times
 5     a letter appears in 'word', 'hand' has at least as
 6     many of that letter in it.
 7
 8     Updates the hand: uses up the letters in the given word
 9     and returns the new hand, without those letters in it.
10
11     Has no side effects: does not modify hand.
12
13     word: string
14     hand: dictionary (string -> int)
15     returns: dictionary (string -> int)
16     """
```

Unanswered

Check   Save   *You have used 0 of 30 submissions*

Show Discussion

New Post

edX (https://www.edx.org)

MITx: 6.00.1x Introduction to Computer Science and Programming Using Python

zhushun0008 (/dashboard) ▾

Courseware (/courses/MITx/6.00.1_3x/2T2014/courseware)   Updates & News (/courses/MITx/6.00.1_3x/2T2014/info)

Calendar (/courses/MITx/6.00.1_3x/2T2014/89309559b0414f6d8cbef9e48ca19f4b/)   Wiki (/courses/MITx/6.00.1_3x/2T2014/course_wiki)

iscussion (/courses/MITx/6.00.1_3x/2T2014/discussion/forum)   Progress (/courses/MITx/6.00.1_3x/2T2014/progress)

Help

VALID WORDS (10 points possible)

At this point, we have written code to generate a random hand and display that hand to the user. We can also ask the user for a word (Python's `raw_input`) and score the word (using your `getWordScore`). However, at this point we have not written any code to verify that a word given by a player obeys the rules of the game. A *valid* word is in the word list; **and** it is composed entirely of letters from the current hand. Implement the `isValidWord` function.

Testing: Make sure the `test_isValidWord` tests pass. In addition, you will want to test your implementation by calling it multiple times on the same hand - what should the correct behavior be? Additionally, the empty string (`''`) is not a valid word - if you code this function correctly, you shouldn't need an additional check for this condition.

Fill in the code for `isValidWord` in `ps4a.py` and be sure you've passed the appropriate tests in `test_ps4a.py` before pasting your function definition here.

Canopy specific instructions: If you modify code in `ps4a.py` go to

```
Run -> Restart Kernel (or hit the CTRL with the dot on your keyboard)
```

before running `test_ps4a.py` . **You have to do this every time you modify the file** `ps4a.py` **and want to run the file** `test_ps4a.py` , otherwise changes to the former will not be incorporated in the latter.

```
1  def isValidWord(word, hand, wordList):
2      """
3      Returns True if word is in the wordList and is entirely
4      composed of letters in the hand. Otherwise, returns False.
5
6      Does not mutate hand or wordList.
7
8      word: string
9      hand: dictionary (string -> int)
10     wordList: list of lowercase strings
11     """
12     # TO DO ... <-- Remove this comment when you code this function
13
```

Unanswered

| Check | Save | *You have used 0 of 30 submissions*

Show Discussion   ✎ New Post

Courseware (/courses/MITx/6.00.1_3x/2T2014/courseware)   Updates & News (/courses/MITx/6.00.1_3x/2T2014/info)

Calendar (/courses/MITx/6.00.1_3x/2T2014/89309559b0414f6d8cbef9e48ca19f4b/)   Wiki (/courses/MITx/6.00.1_3x/2T2014/course_wiki)

iscussion (/courses/MITx/6.00.1_3x/2T2014/discussion/forum)   Progress (/courses/MITx/6.00.1_3x/2T2014/progress)

Help

HAND LENGTH (5 points possible)

We are now ready to begin writing the code that interacts with the player. We'll be implementing the `playHand` function. This function allows the user to play out a single hand. First, though, you'll need to implement the helper `calculateHandlen` function, which can be done in under five lines of code.

```
1  def calculateHandlen(hand):
2      """
3      Returns the length (number of letters) in the current hand.
4
5      hand: dictionary (string int)
6      returns: integer
7      """
8      # TO DO... <-- Remove this comment when you code this function
9
```

Unanswered

| Check | Save |

You have used 0 of 30 submissions

Show Discussion

📝 New Post

About (https://www.edx.org/about-us)   Jobs (https://www.edx.org/jobs)
Press (https://www.edx.org/press)   FAQ (https://www.edx.org/student-faq)
Contact (https://www.edx.org/contact)

edX is a non-profit created by founding partners Harvard and MIT whose mission is to bring the best of higher education to students of all ages anywhere in the world, wherever there is Internet access. EdX's free online MOOCs are interactive and subjects include computer science, public health, and artificial intelligence.

(http://www.meetup.com/edX-Global-Community/)

(http://www.facebook.com/EdxOnline)

(https://twitter.com/edXOnline)

(https://plus.google.com/10823538304409508271

PLAYING A HAND   (15 points possible)

In `ps4a.py`, note that in the function `playHand`, there is a bunch of *pseudocode*. This pseudocode is provided to help guide you in writing your function. Check out the Why Pseudocode? (/c4x/MITx/6.00.1_3x/asset/files_ps04_files_WhyPseudocode.pdf) resource to learn more about the What and Why of Pseudocode before you start coding your solution.

Note: Do **not** assume that there will always be 7 letters in a hand! The parameter `n` represents the size of the hand.

Testing: Before testing your code in the answer box, try out your implementation as if you were playing the game. Here is some example output of `playHand` :

Test Cases
Case #1
Case #2
Case #3
Additional Testing

```
1  def playHand(hand, wordList, n):
2      """
3      Allows the user to play the given hand, as follows:
4
5      * The hand is displayed.
6      * The user may input a word or a single period (the string ".")
7        to indicate they're done playing
8      * Invalid words are rejected, and a message is displayed asking
9        the user to choose another word until they enter a valid word or "."
10     * When a valid word is entered, it uses up letters from the hand.
11     * After every valid word: the score for that word is displayed,
12       the remaining letters in the hand are displayed, and the user
13       is asked to input another word.
14     * The sum of the word scores is displayed when the hand finishes.
15     * The hand finishes when there are no more unused letters or the user
16       inputs a "."
```

Unanswered

| Check | Save | *You have used 0 of 30 submissions* |

Show Discussion   New Post

PLAYING A GAME (10 points possible)

A game consists of playing multiple hands. We need to implement one final function to complete our word-game program. Write the code that implements the `playGame` function. You should remove the code that is currently uncommented in the `playGame` body. Read through the specification and make sure you understand what this function accomplishes. For the game, you should use the `HAND_SIZE` constant to determine the number of cards in a hand.

Testing: Try out this implementation as if you were playing the game. Try out different values for `HAND_SIZE` with your program, and be sure that you can play the wordgame with different hand sizes by modifying *only* the variable `HAND_SIZE` .

Sample Output
Here is how the game output should look…
Hints about the output
Entering Your Code
A Cool Trick about 'print'

```
1  def playGame(wordList):
2      """
3      Allow the user to play an arbitrary number of hands.
4
5      1) Asks the user to input 'n' or 'r' or 'e'.
6        * If the user inputs 'n', let the user play a new (random) hand.
7        * If the user inputs 'r', let the user play the last hand again.
8        * If the user inputs 'e', exit the game.
9        * If the user inputs anything else, tell them their input was invalid.
10
11     2) When done playing the hand, repeat from step 1
12     """
13     # TO DO ... <-- Remove this comment when you code this function
14
```

Unanswered

Check   Save   *You have used 0 of 30 submissions*

Show Discussion   ✎ New Post

Courseware (/courses/MITx/6.00.1_3x/2T2014/courseware)   Updates & News (/courses/MITx/6.00.1_3x/2T2014/info)

Calendar (/courses/MITx/6.00.1_3x/2T2014/89309559b0414f6d8cbef9e48ca19f4b/)   Wiki (/courses/MITx/6.00.1_3x/2T2014/course_wiki)

iscussion (/courses/MITx/6.00.1_3x/2T2014/discussion/forum)   Progress (/courses/MITx/6.00.1_3x/2T2014/progress)

Help

COMPUTER CHOOSES A WORD (15 points possible)

**Part B is dependent on your functions from `ps4a.py`, so be sure to complete `ps4a.py` before working on `ps4b.py` **

Now that you have completed your word game code, you decide that you would like to enable your computer (SkyNet) to play the game (your hidden agenda is to prove once and for all that computers are inferior to human intellect!) In Part B you will make a modification to the `playHand` function from part A that will enable this to happen. The idea is that you will be able to compare how you as a user succeed in the game compared to the computer's performance.

It is your responsibility to create the function `compChooseWord(hand, wordList, n)`. Pseudocode is provided in the file `ps4b.py`.

If you follow the pseudocode, you'll create a computer player that is legal, but not always the best. Once you've implemented it following our approach, feel free to try your own approach! As much as we'd love to give you credit for making an improved `compChooseWord` function, we hope you can understand our automatic grading facilities are limited in their ability to accept differing solutions.

Hints and Sample Output
A Note On Runtime
Test Cases

```
1  def compChooseWord(hand, wordList, n):
2      """
3      Given a hand and a wordList, find the word that gives
4      the maximum value score, and return it.
5
6      This word should be calculated by considering all the words
7      in the wordList.
8
9      If no words in the wordList can be made from the hand, return None.
10
11     hand: dictionary (string -> int)
12     wordList: list (string)
13     returns: string or None
14     """
15     # BEGIN PSEUDOCODE (available within ps4b.py)
16
```

Unanswered

[Check] [Save] *You have used 0 of 30 submissions*

Show Discussion   [✎ New Post]

About (https://www.edx.org/about-us)   Jobs (https://www.edx.org/jobs)
Press (https://www.edx.org/press)   FAQ (https://www.edx.org/student-faq)
Contact (https://www.edx.org/contact)

EdX is a non-profit created by founding partners Harvard and MIT whose
mission is to bring the best of higher education to students of all ages
anywhere in the world, wherever there is Internet access. EdX's free online
MOOCs are interactive and subjects include computer science, public health,
and artificial intelligence.

(http://www.meetup.com/edX-Global-
Community/)

(http://www.facebook.com/EdxOnline)

(https://twitter.com/edXOnline)

(https://plus.google.com/108235383044095082

(http://youtube.com/user/edxonline)

Terms of Service and Honor Code -
Privacy Policy (https://www.edx.org/edx-privacy-policy)

Help

COMPUTER PLAYS A HAND (10 points possible)

Now that we have the ability to let the computer choose a word, we need to set up a function to allow the computer to play a hand - in a manner very similar to Part A's `playHand` function (get the hint?).

Implement the `compPlayHand` function. This function should allow the computer to play a given hand, using the procedure you just wrote in the previous part. This should be very similar to the earlier version in which a user selected the word, although deciding when it is done playing a particular hand will be different.

Be sure to test your function on some randomly generated hands using `dealHand` .

Test Cases
Test Cases

Paste your definition of `compChooseWord` , in addition to your definition of `compPlayHand` , in the box below.

```
 1 def compPlayHand(hand, wordList, n):
 2     """
 3     Allows the computer to play the given hand, following the same procedure
 4     as playHand, except instead of the user choosing a word, the computer
 5     chooses it.
 6
 7     1) The hand is displayed.
 8     2) The computer chooses a word.
 9     3) After every valid word: the word and the score for that word is
10     displayed, the remaining letters in the hand are displayed, and the
11     computer chooses another word.
12     4)  The sum of the word scores is displayed when the hand finishes.
13     5)  The hand finishes when the computer has exhausted its possible
14     choices (i.e. compChooseWord returns None).
15
16     hand: dictionary (string -> int)
```

Unanswered

| Check | Save | *You have used 0 of 30 submissions*

Show Discussion   ✎ New Post

EdX is a non-profit created by founding partners Harvard and MIT whose mission is to bring the best of higher education to students of all ages anywhere in the world, wherever there is Internet access. EdX's free online MOOCs are interactive and subjects include computer science, public health, and artificial intelligence.

(http://www.meetup.com/edX-Global-Community/)

(http://www.facebook.com/EdxOnline)

(https://twitter.com/edXOnline)

(https://plus.google.com/10823538304409508271

(http://youtube.com/user/edxonline)

Terms of Service and Honor Code -
Privacy Policy (https://www.edx.org/edx-privacy-policy)

edX (https://www.edx.org)

MITx: 6.00.1x Introduction to Computer Science and Programming Using Python

zhushun0008 (/dashboard)

Courseware (/courses/MITx/6.00.1_3x/2T2014/courseware)   Updates & News (/courses/MITx/6.00.1_3x/2T2014/info)

Calendar (/courses/MITx/6.00.1_3x/2T2014/89309559b0414f6d8cbef9e48ca19f4b/)   Wiki (/courses/MITx/6.00.1_3x/2T2014/course_wiki)

iscussion (/courses/MITx/6.00.1_3x/2T2014/discussion/forum)   Progress (/courses/MITx/6.00.1_3x/2T2014/progress)

Help

YOU AND YOUR COMPUTER (15 points possible)

Now that your computer can choose a word, you need to give the computer the option to play. Write the code that re-implements the `playGame` function. You will modify the function to behave as described below in the function's comments. As before, you should use the `HAND_SIZE` constant to determine the number of cards in a hand. Be sure to try out different values for `HAND_SIZE` with your program.

Sample Output and Hints
Here is how the game output should look...
Hints about the output
A Note On Runtime
Entering Your Code

```
 1 def playGame(wordList):
 2     """
 3     Allow the user to play an arbitrary number of hands.
 4
 5     1) Asks the user to input 'n' or 'r' or 'e'.
 6         * If the user inputs 'e', immediately exit the game.
 7         * If the user inputs anything that's not 'n', 'r', or 'e', keep asking them again.
 8
 9     2) Asks the user to input a 'u' or a 'c'.
10         * If the user inputs anything that's not 'c' or 'u', keep asking them again.
11
12     3) Switch functionality based on the above choices:
13         * If the user inputted 'n', play a new (random) hand.
14         * Else, if the user inputted 'r', play the last hand again.
15
16         * If the user inputted 'u'  let the user play the game
```

Unanswered

Check   Save   *You have used 0 of 30 submissions*

Show Discussion

New Post

EdX is a non-profit created by founding partners Harvard and MIT whose mission is to bring the best of higher education to students of all ages anywhere in the world, wherever there is Internet access. EdX's free online MOOCs are interactive and subjects include computer science, public health, and artificial intelligence.

(http://www.meetup.com/edX-Global-Community/)

(http://www.facebook.com/EdxOnline)

(https://twitter.com/edXOnline)

(https://plus.google.com/108235383044095082

(http://youtube.com/user/edxonline)