Habib University
CS 110 - Computational Thinking I
Fall 2015

Programming Exam II

Practice Questions

- This sheet contains practice questions representative of the questions that you will get in the exam.
- You will have to submit your solutions to the exam via `svn` to your SCM repository. Make sure you know how to do that from your lab.
- Questions will be added continually to this sheet. You will know that this sheet is not yet complete as long as the last question is titled 'More to Come'.
- Keep calm and code!

Problem 1. Random

See the `help` for the `randint` function in the `random` module. Write a function that takes two integer parameters and uses `random.randint` to return a random number in `[min,max]`.

Problem 2. Die Cricket

A die

A batting player

Die Cricket is a fast-paced cricket game played with a single die. Each roll of the die corresponds to a single ball. There are unlimited number of balls. Assuming a batting player, the player's score is increased or the player is out depending on the amount shown on the die. If the die shows anything other than 5, the player's score is increased by that amount. For example, if the die shows 1, the player's score is increased by 1. If the die shows 2, the player's score is increased by 2, and so on. The player is out if the die shows 5.

The game is played between 2 teams, each consisting of 11 players. For each batting team, all 11 players bat and the team's score is the sum of the players' scores. All players bat till they get out. There are an unlimited number of balls.

(a) Write a function that inputs a team. It takes no argument, inputs from the user the name of the team and the names of its 11 players. It returns a list containing 12 items. The first item is the team's name, and the remaining 11 items are the names of the players. For example, see the following.

```
Enter the name of the team: Pakistan
Enter the name of the 1th player: Sana Mir
Enter the name of the 2th player: Bismah Maroof
Enter the name of the 3th player: Bismah Maroof
Enter the name of the 4th player: Batool Fatima
Enter the name of the 5th player: Iram Javed
Enter the name of the 6th player: Javeria Khan
Enter the name of the 7th player: Javeria Rauf
Enter the name of the 8th player: Nahida Khan
Enter the name of the 9th player: Nain Abidi
Enter the name of the 10th player: Nida Dar
Enter the name of the 11th player: Qanita Jalil
```

(b) Write a function corresponding to a player's inning. The function takes the name of the player as argument, outputs the name followed by the player's runs on each ball faced, 'OUT' when the player gets out, and then the total score. All runs are separated by commas. The function returns the total runs scored. For example,

```
Sana Mir: 2, 3, 2, 1, 3, 2, 2, 4, OUT ... total runs: 19
```

The score for each ball is generated as a random value between 1 and 6.

(c) Write a function corresponding to a team's turn to bat. The function takes a team as argument where the team is as returned by `getTeam`. The function prints the team's name followed by the turn of each of its 11 players. The function returns the total runs scored by the team. One possible execution of the function is given below.

```
Pakistan
Sana Mir: 4, 6, OUT ... total runs: 10
Bismah Maroof: 6, OUT ... total runs: 6
Bismah Maroof: 6, 4, 1, 4, 3, 3, OUT ... total runs: 21
Batool Fatima: 2, 3, 4, 3, 6, 1, 1, 4, 4, 1, 2, 2, 2, 3, OUT ... total runs: 38
Iram Javed: 3, 6, 2, 6, 2, OUT ... total runs: 19
Javeria Khan: 3, 1, 6, 6, 2, 6, 1, 1, 6, 1, 2, 4, 2, 1, 1, 6, 1, 3, 3, 3, OUT ...
Javeria Rauf: 3, 6, 4, 2, 2, OUT ... total runs: 17
Nahida Khan: 6, 4, OUT ... total runs: 10
Nain Abidi: 3, 1, 1, 1, 4, 2, 2, 1, 6, OUT ... total runs: 21
Nida Dar: 1, 4, 4, 6, 3, 3, 4, 3, 6, 1, 6, 1, OUT ... total runs: 42
Qanita Jalil: 1, 3, 4, OUT ... total runs: 8
Final score: 251
```

(d) In a game of Die Cricket, all 22 players bat and the team with the higher total score wins. Write a function to play a game of Die Cricket between 2 teams. The function takes no arguments. It first inputs two teams and then executes each team's turn to bat. The team that is input first goes to bat first. In the end, the outcome of the game–`TIE` or the `WINNING TEAM` is announced. Use of previous functions as appropriate is encouraged.

Problem 3. Lists

(a) Write a function called `nested_sum` that takes a nested list of integers and add up the elements from all of the nested lists.

(b) Write a function that takes a list of numbers and returns the cumulative sum; that is, a new list where the i-th element is the sum of the first $i+1$ elements from the original list. For example, the cumulative sum of `[1, 2, 3]` is `[1, 3, 6]`.

(c) Write a function called `middle` that takes a list and returns a new list that contains all but the first and last elements. So `middle([1,2,3,4])` should return `[2,3]`.

(d) Write a function called `chop` that takes a list, modifies it by removing the first and last elements, and returns `None`.

(e) Write a function called `is_sorted` that takes a list of integers as a parameter and returns `True` if the list is sorted in ascending order and `False` otherwise.

(f) Two words are anagrams if you can rearrange the letters from one to spell the other. Write a function called `is_anagram` that takes two strings and returns `True` if they are anagrams.

(g) Write a function called `has_duplicates` that takes a list and returns `True` if there is any element that appears more than once. It should not modify the original list.

(h) Write a function called `remove_duplicates` that takes a list and returns a new list with only the unique elements from the original. Hint: they don't have to be in the same order.

Problem 4. Dictionaries

(a) Write a function that reads a word list from a file and stores them as keys in a dictionary. It doesnt matter what the values are. Then you can use the `in` operator as a fast way to check whether a string is in the dictionary.

(b) Use a dictionary to write a faster, simpler version of `has_duplicates` from Q3g.

(c) Write a function that reads a word list from a file and prints the number of times each letter appears in the file.

(d) Write a function to perform *reverse lookup* in a dictionary. It takes a dictionary and a value as argument and returns the key for that value.

Problem 5. Tuples

(a) Write a function called `most_frequent` that takes a string and prints the letters in decreasing order of frequency.

(b) Write a program that reads a word list from a file and prints all the sets of words that are anagrams.

Here is an example of what the output might look like:

```
['deltas', 'desalt', 'lasted', 'salted', 'slated', 'staled']
['retainers', 'ternaries']
['generating', 'greatening']
['resmelts', 'smelters', 'termless']
```

Hint: you might want to build a dictionary that maps from a set of letters to a list of words that can be spelled with those letters. The question is, how can you represent the set of letters in a way that can be used as a key?

(c) Modify the previous program so that it prints the largest set of anagrams first, followed by the second largest set, and so on.

(d) Two words form a "metathesis pair" if you can transform one into the other by swapping two letters; for example, converse and conserve. Write a program that finds all of the metathesis pairs in the dictionary. Hint: dont test all pairs of words, and dont test all possible swaps.

References

- Allen B. Downey. *Think Python.* Green Tea Press, Needham, MA, USA, 2012.