Project Set: Lecture 6

Matthew Ellison
SPISE

July 21, 2019

1 Debug It!

Fix the following short programs (available at the github).

1.1 List Sort

```
unsorted_list = [6,2,5,1,3]
sorted_list = unsorted_list.sort()
for i in range(len(unsorted_list)):
    print(sorted_list[i])
```

1.2 Printing Dots

```
smallnumberfromuser = input('Enter a number between 0 and 20: ')
smallnumberfromuser = smallnumberfromuser * 2
for i in range(smallnumberframuser / 2)
    print('.')
```

1.3 Genotype-to-Phenotype Dictionary

```
geno_to_pheno = {['long', 'long']: 'long', ['long', 'short']: 'long',
                 ['short', 'long']: 'long',['short','short']:'short'}
```

2 Testing...

Write a file to comprehensively test the following function (not yet implemented), then implement the function. Make sure it passes all your tests.

```
    def reverse_digits(number):
        '''input is a number (int). The output should be the digits in reverse
        order (e.g. 534 --> 435). In the case of a negative integer, the
        negative sign should remain in the beginning of the reversed number
        (e.g. -32 --> -23).
        '''

```

3 Debug It! v2 (Optional)

Fix the following error-ridden longer program (available at the github).

```
import time

difficulty=0
max_word_size=6
max_paragraph_size=10

lower_letters='abcdefghijklmnopqrstuvwxyz'
upper_letters='ABCDEFGHIJKLMNOPQRSTUVWXYZ'
numerals='1234567890'
symbols='''!@#$%^&*()_+`~-=[]{}\|:;"'/?><.,'''
difficulty_ranking=[lower_letters,upper_letters,numerals,symbols]

def generate_letter():
    level=random.randint(0,difficulty)
    character=random.choice(difficulty_ranking[level])
    return char

def generate_word():
    word_size=random.randint(1,10)
    output=''
    for x in range(word_size):
        output.append(generate_letter)
    return output

def generate_paragraph():
    par_size=random.randint(10,max_par_size)
    output=''
    for x in range(0,par_size):
        output+=generate_word()
        output+=' '
    return output

def words_correct(attempt):
    attempt_words=attempt.split(' ')
    solution_words=solution.split(' ')
    num_words_compare=min(len(attempt_words),len(solution_words))
    words_correct=0
    total_words=len(solution_words)
```

```
39      for word_idx in range(0,num_words_compare):
40          if attempt_words[word_idx]==solution_words[word_idx]:
41              words_correct+=1
42      return len(attempt_words),words_correct,total_words
43
44  def play_game():
45      playing=True
46      print('Get Ready to Type!\n')
47      time.sleep(1)
48      while playing:
49          new_par=generate_paragiph()
50          print(new_par+'\n')
51
52          start=time.time()
53          submission=input('type the above!\n')
54          end=time.time()
55
56          total_time=(end-start)/60
57          words_typed,word_score,total_words=words_correct(submission,new_par)
58          word_accuracy=round(word_score/total_words,3)
59          wpm=round(words_typed/total_time,2)
60
61          print('you typed at',wpm,'wpm with an accuracy of',word_accuracy,'!\n')
62
63          keep_going=input('keep playing? (y/n)\n')
64          if keep_going!='y':
65              playing=True
66
67  play_game()
```