Midterm Exam Notebook — SOLUTION

Problem 1

7
```
grapheme_list = [
  'a', 'ai', 'ar', 'au', 'aw', 'ay',
  'b', 'bb',
  'c', 'cc', 'ce', 'ch', 'ci', 'ck',
  'd', 'dd', 'dge',
  'e', 'ea', 'ed', 'ee', 'ei', 'ew', 'ey',
  'f',
  'g',
  'ge', 'gg', 'gh', 'gn',
  'h',
  'i', 'ie', 'igh',
  'j',
  'k', 'kn',
  'l', 'll',
  'm', 'mb', 'mm',
  'n', 'ng', 'nn',
  'o', 'oa', 'oi', 'oo', 'ou', 'ough', 'oul', 'ow', 'oy',
  'p', 'ph', 'pp',
  'qu',
  'r', 'rr',
  's', 'sc', 'se', 'sh', 'ss',
  't', 'tch', 'th', 'ti', 'tt',
  'u',
  'v', 've',
  'w', 'wh', 'wr',
  'x',
  'y',
  'z', 'ze', 'zz'
]

test_text1 = 'andiwonderdoessheknowthatmyheartisinthedough'

extra_credit_test_text2 = 'albainiascurrencyistheqindar'

grapheme_list.sort(key=len, reverse=True)
```

NB: My Problem 1 solution is not tail-recursive. It will be educational to look at https://www.programmerinterview.com/recursion/tail-recursion and then figure out how to make my solution tail-recursive. Implementing tail recursion will greatly improve the performance for long strings. In fact, my code will crash (!!) with "stack overflow" or "recursion error" if the string is too long because there is a limit to how many stack frames any program can have, commonly a few hundred. Python's limit is 1000 stack frames, so if the string has close to 1000 graphemes, the code which worked on all shorter strings will mysteriously fail. Before declaring victory with your tail-recursive solution, make sure it still gets 225 and 96 for the the test cases.

My implementation of grapheme_score():

8
```
def grapheme_score(text):
  if len(text) == 0:
    return 0
  for grapheme in grapheme_list:
    if text.startswith(grapheme):
     length_of_grapheme = len(grapheme)
     return pow(3, length_of_grapheme) + grapheme_score(text[length_of_grapheme:])
  # Oh-oh, no match, strip off one character and continue scoring
  return grapheme_score(text[1:])

print(grapheme_score(test_text1))  # this code gets 225
```

```
225
```

9
```
print(grapheme_score(extra_credit_test_text2))  # this code gets 96
```

```
96
```

Problem 2

SOLUTION:

Using the common beginnings, we find that the first message has the following columns:

```
TAHEW
ERPRE
0T6RO
YDE00
NUNSR
    Y
```

Knowing that the message begins with WEATH tells you that the column order is

```
43521
```

Rearranging the columns gives:

```
WEATH
ERREP
ORT06
00DYE
RSUNN
Y
```

and

```
WEATH
ERREP
ORT06
00BIS
HOPPA
RTLYC
LOUDY
```

Pretty typical for early November ;).