과제 2

ASSIGNMENT 2

391040277 | Nathan Cho | 조나단

Computational Thinking and Problem Solving | 2019-02-18

Problem 1. 자기소개서 이름 추출

소스 코드

```
# problem 1: get the names from formatted sentences

def get_name(sentence):
    # words to replace
    replace = ["저는 ", "이라고 합니다.", "라고 합니다.", "제 이름은 ", "입니다."]

    for word in replace:
        sentence = sentence.replace(word, "")

    return sentence.strip()

while True:
    sentence = input("Input a sentence: ")
    print("Name:", get_name(sentence))
```

결과 화면

```
Input a sentence: 저는 홍길동이라고 합니다.
Name: 홍길동
Input a sentence: 제 이름은 심청입니다.
Name: 심청
Input a sentence: 저는 이순신입니다.
Name: 이순신
```

설명

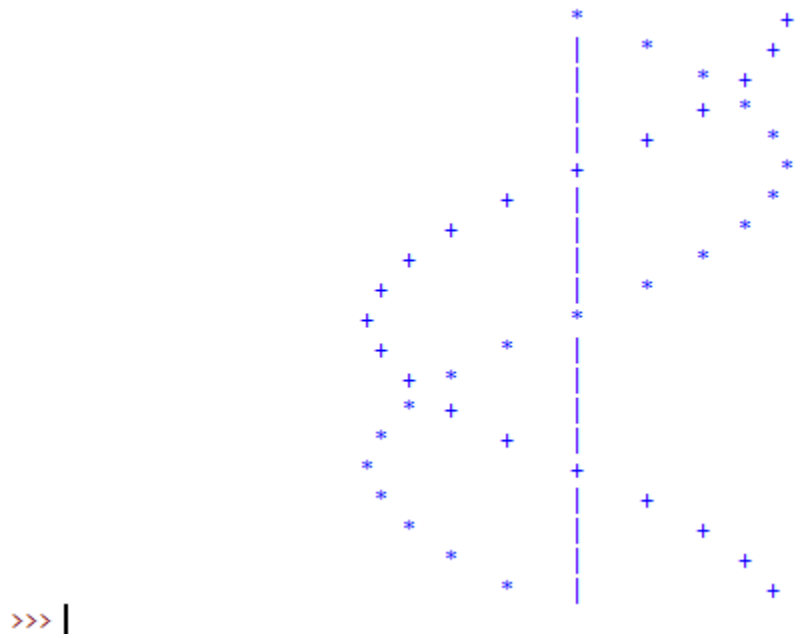
제거할 문자열을 ***replace*** 라는 리스트에 저장해 입력한 문장에서 그 문자열을 모두 제거하는 방식으로 작동하는 알고리즘이다.

Problem 2. 사인 코사인 그래프 그리기

소스 코드

```
1.  # problem 2: sin, cos, axis drawing
2.
3.  from math import sin, cos, pi
4.
5.  window_width = 80
6.  unit = 15
7.  left_margin = " " * int(window_width / 2 - unit)
8.
9.  def get_pos(value):
10.     # position within graph width range (0 is 15)
11.     return int(unit + round(value * unit))
12.
13. def render(x):
14.     row = [' '] * (unit*2 + 1)
15.     row[get_pos(0)] = '|'
16.     row[get_pos(sin(x))] = '*'
17.     row[get_pos(cos(x))] = '+'
18.
19.     print(left_margin + ''.join(row))
20.
21. for i in range(0, 20):
22.     render(2*pi*i/20)
```

결과 화면



설명

row 라는 문자열로 이루어진 리스트를 사용해서 표시하는 알고리즘이다.

1. 리스트를 31 개의 공백 문자로 초기화한 후
2. ***get_pos(value)*** 라는 함수를 사용해 몇 번째에 그 문자가 와야 하는지를 계산해
3. 문자열에서 계산한 값을 인덱스로 그래프를 표시할 문자를 입력한다.
4. ***render(x)*** 함수를 통해 x 값에 대한 그래프를 가진 문자열을 출력한다.

Problem 3. 로또 번호 생성기

소스 코드

```
# problem 3: lottery number generator

import random

freq = [0]*45
recommendation = []

def generate():
    return random.randint(1, 45)

def lotto_generator():
    numbers = [generate()]

    for i in range(1, 6):
        number = generate()

        # remove duplicates
        while number in numbers:
            number = generate()

        numbers.append(number)

    return numbers

for i in range(0, 1000):
    lotto = lotto_generator()
    for j in lotto:
        freq[j - 1] += 1
    # print(lotto)

for i in range(0, 6):
    top = max(freq)
    recommendation.append(str(freq.index(top) + 1)+" (" + str(top) + " times)")
    # resets the top value
    freq[freq.index(top)] = -1

for number in recommendation:
    print(number)
```

결과 화면

```
31 (157 times)
6 (155 times)
8 (148 times)
22 (147 times)
11 (146 times)
12 (145 times)
>>>
```

설명

lotto_generator() 라는 함수를 통해서 무작위의 6 개의 중복되지 않는 숫자를 뽑은 뒤 ***freq*** 라는 빈도를 저장하는 리스트에 반영하고 1000 번의 실행 후에 가장 빈도가 높은 숫자 6 개를 표시하는 알고리즘이다.

Problem 4. 전자사전 만들기

소스 코드

```
# problem 4: dictionary

f = open('dict_test.TXT', 'r', encoding='utf-8')

dictionary = {}

for line in f:
    # split at the first space except line break
    word = line[:-1].split(" : ", 1)
    # add word to dictionary
    dictionary.update({word[0]:word[-1]})

f.close()

while True:
    query = input("Enter word: ")
    result = dictionary.get(query,"Error: the word '"+query+"' doesn't exist")

    if 'Error' in result:
        # error message
        print(result)
    else:
        print(query, '=', result)
```

결과 화면

```
Enter word: apple
apple = n.사과
Enter word: pineapple
pineapple = n.파인애플
Enter word: good
good = a.좋은
Enter word: dictionary
dictionary = n.사전
Enter word: dict
Error: the word 'dict' doesn't exist
Enter word:
```

설명

파일을 읽기 모드로 불러와서 모두 ***dictionary*** 라는 딕셔너리에 *key* 는 단어, *value* 는 뜻으로 저장하는 방식으로 제작했다. 사용자 입력을 딕셔너리의 키로 검색해서 뜻을 출력한다.

Problem 5. 끝말 잇기

소스 코드

```
# problem 5: 5 word shiritori

f = open('dict_test.TXT', 'r', encoding='utf-8')
words = []

for line in f:
    word = line.split(" : ", 1)[0]
    if len(word) == 5:
        if " n." in line:
            words.append(word)

f.close()

previous = "apple"
used_words = [previous]

while True:
    while True:
        word = input(previous + ", next word: ").lower()

        if len(word) is 5:
            if word in words:
                if word not in used_words:
                    if word[0] == previous[-1]:
                        used_words.append(word)
                        previous = word
                        break
                    else:
                        print("Error: the word '" + word + "' doesn't match")
                else:
                    print("Error: the word '" + word + "' was already used")
            else:
                print("Error: the word '" + word + "' doesn't exist")
        elif len(word) > 5:
            print("Error: the word is longer than 5 characters")
        else:
            print("Error: the word is shorter than 5 characters")
```

결과 화면

```
apple, next word: eagle
eagle, next word: error
error, next word: orange
Error: the word is longer than 5 characters
error, next word: rose
Error: the word is shorter than 5 characters
error, next word: eagle
Error: the word 'eagle' was already used
error, next word: eeeee
Error: the word 'eeeee' doesn't exist
error, next word: peach
Error: the word 'peach' doesn't match
error, next word:
```

설명

사전 파일에서 5 글자이고 명사인 단어들을 ***words*** 리스트에 저장하고 그것을 사용자의 입력과 비교하며 적절성을 판단한다. 입력한 단어의 길이, 단어의 존재 유무, 이미 사용된 단어인지의 여부, 단어의 끝말이 이어지는 지의 여부에 따라 오류 메시지를 출력하거나 다음 단어로 진행하는 알고리즘이다.

Problem 6. 성적 처리하기

소스 코드

```
1.  # problem 6: score processing
2.
3.  f = open('score.csv', 'r')
4.
5.  score = []
6.
7.  for line in f:
8.      column = line.replace('\n', '').split(",")
9.      score.append([column[0], int(column[1]), int(column[2]), int(column[3]),
10.                   int(column[1]) + int(column[2]) + int(column[3])])
11.
12. for person in score:
13.     rank = 1
14.     for compare in score:
15.         if compare[4] > person[4]:
16.             rank += 1
17.     person.append(rank)
18.
19. print("%-
    8s | %4s | %4s | %4s | %6s | %4s" % ("ID", "Kor", "Eng", "Math", "Total", "Rank"
    ))
20. print("%-
    8s ┼ %4s ┼ %4s ┼ %4s ┼ %6s ┼ %4s" % ("-"*8, "-"*4, "-"*4, "-"*4, "-"*6, "-"*4))
21.
22. for person in score:
23.       print("%8s | %4d | %4d | %4d | %6d | %4d" %
24.             (person[0], person[1], person[2], person[3], person[4], person[5]))
```

결과 화면

```
ID       |  Kor |  Eng | Math |  Total | Rank
---------┼------┼------┼------┼--------┼-----
20170101 |   90 |   80 |   70 |    240 |    4
20168372 |   90 |   95 |   90 |    275 |    2
20180392 |  100 |  100 |   90 |    290 |    1
20173882 |   95 |   80 |   75 |    250 |    3
>>>
```

설명

파일에서 점수를 불러와 학번, 과목별 점수, 총합이 있는 ***score*** 리스트를 만든 후에 석차를 계산해서 리스트에 다시 넣고 출력하는 알고리즘이다.

Problem 7. 수식 계산기

소스 코드

```
1.  # problem 7: expression calculator
2.
3.  # change minus to negative value with the integer
4.  exp = input("Enter an expression: ").replace(" ", '').replace("-", "+-
    ").split('+')
5.
6.  result = 0
7.
8.  for number in exp:
9.      result += int(number)
10.
11. print("=", result)
```

결과 화면

```
Enter an expression: 12+345+6789
= 7146
>>>

Enter an expression: 12+34-5
= 41
>>>

Enter an expression: 6-14
= -8
>>>
```

설명

더하기와 빼기로만 구성되어있기 때문에 빼기를 음수를 더한다고 생각하고 계산하는 알고리즘이다. 빼기 (‘-‘)를 ‘+-‘ 로 변환한 다음 식 전체를 ‘+’로 *split* 해 모두 더하는 방식으로 작동한다.