고급통계프로그래밍 Solution 10

컴퓨터과학부 2017920054 이호준

Ex 01

Solution

```
import random

pc_pick = random.randint(1, 100)
my_pick = 0

while (1):
    my_pick = int(input("Guess What? : "))
    if my_pick > pc_pick:
        print("Your guess is bigger than answer")
    elif my_pick < pc_pick:
        print("Your guess is smaller than answer")
    else:
        print("Correct! Well Done :)")
        break
```

- random 모듈을 호출한다
- `pc_pick` 에는 컴퓨터가 뽑은 숫자를, `my_pick`에는 내가 뽑은 숫자를 입력한다.
- while(1)을 통해 무한반복을 진행:
  - `my_pick`을 입력받는다
  - 만약 `my_pick`보다 `pc_pick`이 더 크거나 작으면 이에 맞는 프롬프트를 출력
  - 만약 `my_pick`과 `pc_pick`이 같으면 프롬프트를 출력 후 반복문 종료

Result

```
# Example

Guess What? : 50
Your guess is smaller than answer
Guess What? : 75
Your guess is bigger than answer
Guess What? : 67
Your guess is bigger than answer
Guess What? : 60
Your guess is bigger than answer
```

```
Guess What? : 55
Your guess is smaller than answer
Guess What? : 57
Correct! Well Done :)
```

Ex 02

Solution

- 주피터 노트북이여서 뒤에 첨부합니다!

Ex 03

Solution

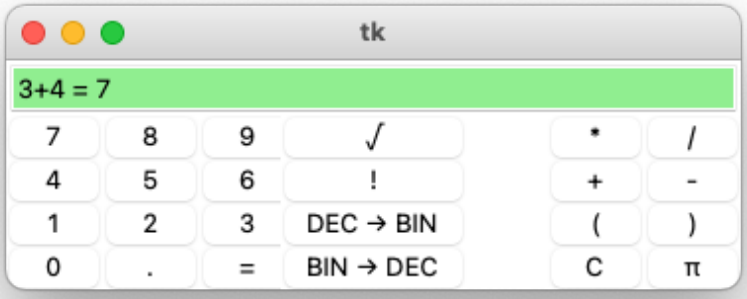
```
from tkinter import *
from decimal import *
import math

cal = Tk()
cal.title = ("Calculator")

display = Entry(cal, width=40, bg="light green")
display.grid()

def factorial(n):
    return math.factorial(int(n))

def dec_to_bin(n):
    return format(int(n), "b")

def bin_to_dec(n):
    return int(n, 2)

def sqrt(n):
    return math.sqrt(float(n))

functions_dict = {
    '√' : sqrt,
    "!" : factorial,
    "DEC → BIN" : dec_to_bin,
    "BIN → DEC" : bin_to_dec
}

def click(key):
    if key == '=':
        try:
            result = str(eval(display.get()))[:10]
            display.insert(END, " = " + result)
        except:
```

```
            display.insert(END, " → ERROR!")
    elif key == "C":
        display.delete(0, END)
    elif key in functions_dict:
        try:
            result = display.get()
            display.delete(0, END)
            display.insert(END, str(functions_dict[key](result)))
        except:
            display.insert(END, " → ERROR!")
    else:
        display.insert(END, key)

top_row = Frame(cal)
top_row.grid(row=0, column=0, sticky=N)

display = Entry(top_row, width=40, bg="light green")
display.grid()

## Number Pad
num_pad = Frame(cal)
num_pad.grid(row=1, column=0, sticky=W)

num_pad_list = [
    '7', '8', '9',
    '4', '5', '6',
    '1', '2', '3',
    '0', '.', '='
]

r = 0
c = 0

for btn_text in num_pad_list:
    def num_cmd(x=btn_text):
        click(x)
    Button(num_pad, text=btn_text, width=5, command=num_cmd).grid(row=r, column=c)
    c = c + 1
    if c > 2:
        c = 0
        r = r + 1

## Function Pad
functions = Frame(cal)
functions.grid(row=1, column=0)

r = 0
c = 0
for btn_text in functions_dict:
    def fn_cmd(x=btn_text):
        click(x)
    Button(functions, text=btn_text, width=10, command=fn_cmd).grid(row=r, column=c)
    r = r + 1

## Operator Pad
operator = Frame(cal)
operator.grid(row=1, column=0, sticky=E)
opr_pad_list = [
```

```
    '*', '/',
    '+', '-',
    '(', ')',
    'C', 'π'
]

r = 0
c = 0

for btn_text in opr_pad_list:
    def opr_cmd(x=btn_text):
        if (x == 'π'):
            click("3.141592")
        else:
            click(x)
    Button(operator, text=btn_text, width=5, command=opr_cmd).grid(row=r, column=c)
    c = c + 1
    if c > 1:
        c = 0
        r = r + 1

cal.mainloop()
```

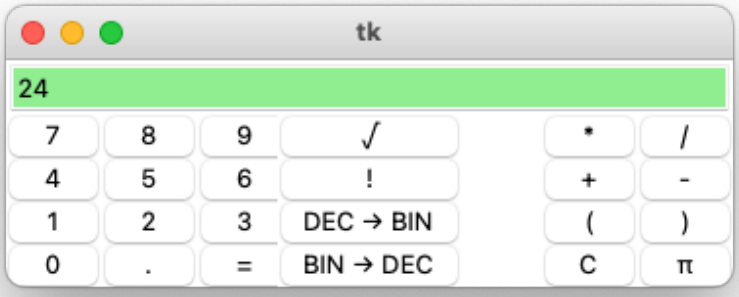
- 교수님이 주신 코드에 함수 구현을 위한 뼈대를 추가
- list 대신 dict를 사용 → key-value 구조를 이용해 바로 호출 가능하다는 장점!
- `factorial` : math.factorial를 사용
- `sqrt` : math.sqrt를 사용
- `dec_to_bin` : format() 함수의 'b' 옵션을 이용 (10진법을 2진법으로 표현)
- `bin_to_dec` : int() 함수의 2 옵션을 이용 (n을 2진법으로 간주하여 10진법으로 변환)
- `functions_dict[key]()` 형태:
  1. key-value를 이용해 key에 해당하는 함수 이름을 가져옴
  2. (result) 인자를 함께 넘겨주면서 해당 함수를 호출
- `pi` 의 경우, 계산기의 심미성을 위해 operator의 빈 자리에 삽입
  - 만약 btn_text가 pi라면 click의 인자를 3.14... 로 변환해서 전달 → 숫자처럼 간주 (eval에서 처리)

Result



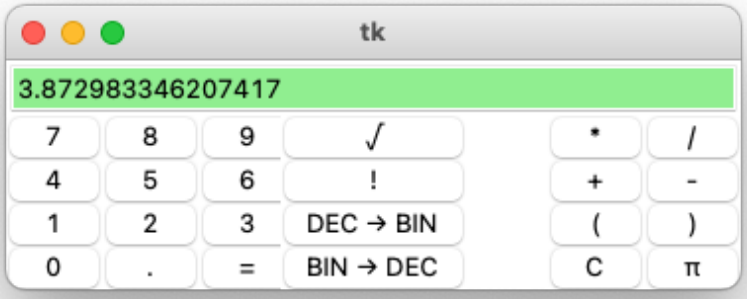
1. 기본 연산

- 3, +, 4, = 입력



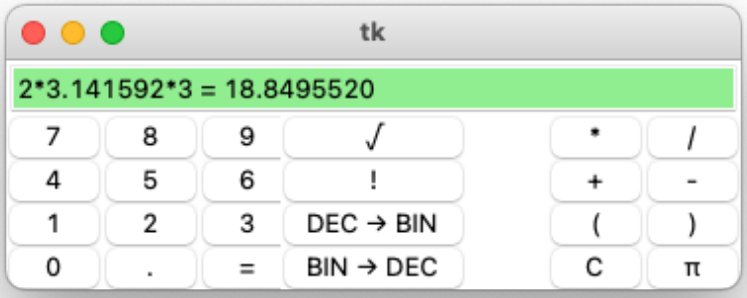
2. bin_to_dec :

- 11000, `BIN → DEC` 입력



3. sqrt :

- 15, `sqrt` 입력



4. pi :

- 2, *, pi, *, 3, = 입력

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
medals = pd.read_csv("./data/Medals.csv")
```

In [3]:
```
medals.head(10)
```

Out[3]:

	Athlete	Age	Year	Closing Ceremony Date	Gold Medals	Silver Medals	Bronze Medals	Total Medals
0	Michael Phelps	23.0	2008	08/24/2008	8	0	0	8.0
1	Michael Phelps	19.0	2004	08/29/2004	6	0	2	8.0
2	Michael Phelps	27.0	2012	08/12/2012	4	2	0	6.0
3	Natalie Coughlin	25.0	2008	08/24/2008	1	2	3	6.0
4	Aleksey Nemov	24.0	2000	10/01/2000	2	1	3	6.0
5	Alicia Coutts	24.0	2012	08/12/2012	1	3	1	5.0
6	Missy Franklin	17.0	2012	08/12/2012	4	0	1	5.0
7	Ryan Lochte	27.0	2012	08/12/2012	2	2	1	5.0
8	Allison Schmitt	22.0	2012	08/12/2012	3	1	1	5.0
9	Natalie Coughlin	21.0	2004	08/29/2004	2	2	1	5.0

In [4]:
```
athlete_country = pd.read_csv("./data/Athelete_Country_Map.csv")

athlete_sports = pd.read_csv("./data/Athelete_Sports_Map.csv")
```

In [5]:
```
athlete_country_dd = athlete_country.drop_duplicates(subset='Athlete')
merged_ac = pd.merge(left=medals, right=athlete_country_dd, left_on='Athlete', right_on='Athlete')

merged_ac
```

Out[5]:

	Athlete	Age	Year	Closing Ceremony Date	Gold Medals	Silver Medals	Bronze Medals	Total Medals	Country
0	Michael Phelps	23.0	2008	08/24/2008	8	0	0	8.0	United States
1	Michael Phelps	19.0	2004	08/29/2004	6	0	2	8.0	United States
2	Michael Phelps	27.0	2012	08/12/2012	4	2	0	6.0	United States
3	Natalie Coughlin	25.0	2008	08/24/2008	1	2	3	6.0	United States
4	Natalie Coughlin	21.0	2004	08/29/2004	2	2	1	5.0	United States
...	...	...	...	...	...	...	...	...	...
8526	Olena Sadovnycha	32.0	2000	10/01/2000	0	1	0	1.0	Ukraine
8527	Kateryna Serdiuk	17.0	2000	10/01/2000	0	1	0	1.0	Ukraine
8528	Wietse van Alten	21.0	2000	10/01/2000	0	0	1	1.0	Netherlands
8529	Sandra Wagner-Sachse	31.0	2000	10/01/2000	0	0	1	1.0	Germany
8530	Rod White	23.0	2000	10/01/2000	0	0	1	1.0	United States

8531 rows × 9 columns

In [6]:
```
len(merged_ac)
```

Out[6]:
```
8531
```

In [7]:
```
athlete_sports_dd = athlete_sports.drop_duplicates(subset='Athlete')

len(athlete_sports_dd)
```

Out[7]:
```
6956
```

In [8]:
```
merged_result = pd.merge(left=merged_ac, right=athlete_sports_dd, left_on='Athlete', right_on='Athlete')

merged_result
```

Out[8]:

	Athlete	Age	Year	Closing Ceremony Date	Gold Medals	Silver Medals	Bronze Medals	Total Medals	Country	Sport
0	Michael Phelps	23.0	2008	08/24/2008	8	0	0	8.0	United States	Swimming
1	Michael Phelps	19.0	2004	08/29/2004	6	0	2	8.0	United States	Swimming
2	Michael Phelps	27.0	2012	08/12/2012	4	2	0	6.0	United States	Swimming
3	Natalie Coughlin	25.0	2008	08/24/2008	1	2	3	6.0	United States	Swimming
4	Natalie Coughlin	21.0	2004	08/29/2004	2	2	1	5.0	United States	Swimming
...	...	...	...	...	...	...	...	...	...	...
8526	Olena Sadovnycha	32.0	2000	10/01/2000	0	1	0	1.0	Ukraine	Archery
8527	Kateryna Serdiuk	17.0	2000	10/01/2000	0	1	0	1.0	Ukraine	Archery
8528	Wietse van Alten	21.0	2000	10/01/2000	0	0	1	1.0	Netherlands	Archery
8529	Sandra Wagner-Sachse	31.0	2000	10/01/2000	0	0	1	1.0	Germany	Archery
8530	Rod White	23.0	2000	10/01/2000	0	0	1	1.0	United States	Archery

8531 rows × 10 columns

In [9]:
```
## Solution 2 Start!

# 한국의 모든 메달리스트 구하기 (중복 포함)
korea_result = merged_result[merged_result['Country'] == 'South Korea']

korea_result
```

Out[9]:

	Athlete	Age	Year	Closing Ceremony Date	Gold Medals	Silver Medals	Bronze Medals	Total Medals	Country	Sport
43	An Hyeon-Su	20.0	2006	02/26/2006	3	0	1	4.0	South Korea	Short-Track Speed Skating
133	Lee Jeong-Su	20.0	2010	02/28/2010	2	1	0	3.0	South Korea	Short-Track Speed Skating
139	Jin Seon-Yu	17.0	2006	02/26/2006	3	0	0	3.0	South Korea	Short-Track Speed Skating
140	Lee Ho-Seok	19.0	2006	02/26/2006	1	2	0	3.0	South Korea	Short-Track Speed Skating
141	Lee Ho-Seok	23.0	2010	02/28/2010	0	2	0	2.0	South Korea	Short-Track Speed Skating
...	...	...	...	...	...	...	...	...	...	...
8497	Lee Chang-Hwan	26.0	2008	08/24/2008	1	0	0	1.0	South Korea	Archery
8508	Jang Yong-Ho	28.0	2004	08/29/2004	1	0	0	1.0	South Korea	Archery
8509	Jang Yong-Ho	24.0	2000	10/01/2000	1	0	0	1.0	South Korea	Archery
8522	Kim Cheong-Tae	20.0	2000	10/01/2000	1	0	0	1.0	South Korea	Archery
8524	O Gyo-Mun	28.0	2000	10/01/2000	1	0	0	1.0	South Korea	Archery

274 rows × 10 columns

In [10]:
```
# 1. 대한민국의 연도별로 메달 집계하기

# 'Year'를 기준으로 Groupby를 진행한다. -> sum() 통계함수로 메달의 합을 구하고 그 중에서 메달에 관련한 Column만 추출한다.
kr_result_by_year = korea_result.groupby('Year').sum()[['Gold Medals', 'Silver Medals', 'Bronze Medals', 'Total Medals']]

# 결과를 출력한다.
kr_result_by_year
```

Out[10]:

Year	Gold Medals	Silver Medals	Bronze Medals	Total Medals
2000	12	26	35	73.0
2002	5	2	0	7.0
2004	14	28	10	52.0
2006	14	3	2	19.0
2008	41	11	26	78.0
2010	6	10	2	18.0
2012	18	13	30	61.0

In [11]:
```
# 2. 대한민국의 선수별 총메달 개수 구하기 (금, 은, 동 순으로 정렬)

# 'Athlete'를 기준으로 Groupby를 진행한다. -> sum() 통계함수로 메달의 합을 구하고 그 중에서 메달에 관련한 Column만 추출한다.
kr_result_by_athlete = korea_result.groupby('Athlete').sum()[['Gold Medals', 'Silver Medals', 'Bronze Medals', 'Total Medals']]

# 금은동을 기준으로(by), 내림차순으로(ascending=False), row를 중심으로(axis=0) sort_values() 를 진행한다.
kr_result_by_athlete.sort_values(by=['Gold Medals', 'Silver Medals', 'Bronze Medals', 'Total Medals'], axis=0, ascending=False)
```

Out[11]:

Athlete	Gold Medals	Silver Medals	Bronze Medals	Total Medals
Jin Jong-O	3	2	0	5.0
Park Seong-Hyeon	3	1	0	4.0
An Hyeon-Su	3	0	1	4.0
Jin Seon-Yu	3	0	0	3.0
Yun Mi-Jin	3	0	0	3.0
...	...	...	...	...
Song Myeong-Seop	0	0	1	1.0
Yang Tae-Yeong	0	0	1	1.0
Yoon Jae-Young	0	0	1	1.0
Yu Ji-Hye	0	0	1	1.0
Yun Seok-Yeong	0	0	1	1.0

232 rows × 4 columns