AI 프로그래밍 7

융합학과 권오영
oykwon@koreatech.ac.kr

PYTHON 프로그래밍 연습(풀이)

연습

❖ 1960년대에 Frank Drake가 인간과 교신할 수 있는 지적인 외계 문명(생명체)의 수 N을 추정하는 Drake 방정식을 만들었다.

N = R * p * n * f * i * c * L, 여기서

R: 우리 은하 안에서 1년동안 탄생하는 항성의 수 (7로 가정)

p: 항성이 행성을 갖고 있을 비율(percent)

n: 행성들 중에서 생명체가 살 수 있는 조건을 갖춘 행성의 수

f: 조건을 갖춘 행성에서 실제로 생명체가 탄생할 행성의 비율(percent)

i: 탄생한 생명체가 지적 생명체로 진화할 비율(percent)

c: 지적 생명체가 외부와 교신할 기술을 갖고 있을 비율(percent)

L: 통신 기술을 가진 생명체가 존속할 수 있는 기간(단위: 년)

❖ 위 입력을 받아서 N 값을 구하는 코드를 작성하세요.

연습

```
# https://ko.wikipedia.org/wiki/드레이크_방정식 참조
# R = 7
# p = 0.5
# n = 2
# f = 1
# i = 0.01
# c = 0.01
# L = 10000

R = int(input('우리 은하 안에서 1년동안 탄생하는 항성의 수 ( 7로 가정) : '))
p = float(input('항성이 행성을 갖고 있을 비율 (0에서 1 사이) : '))
n = int(input('행성들 중에서 생명체가 살 수 있는 조건을 갖춘 행성의 수 : '))
f = float(input('조건을 갖춘 행성에서 실제로 생명체가 탄생할 행성의 비율 (0에서 1 사이) : '))
i = float(input('탄생한 생명체가 지적 생명체로 진화할 비율 (0에서 1 사이) : '))
c = float(input('지적 생명체가 외부와 교신할 기술을 갖고 있을 비율 (0에서 1 사이) : '))
L = int(input('통신 기술을 가진 생명체가 존속할 수 있는 기간(단위: 년) : '))

N = R * p * n * f * i * c * L

print('\n\n인간과 교신할 수 있는 지적인 외계 문명(생명체)의 수 : ',N)
```

연습

❖ 파이 값을 구하는 방법중에 Leibniz 수열을 이용하는 방법이 있다.

$$4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \cdots = \pi$$

위 식을 이용하여 파이 값을 구하는 코드를 작성하시오.

초기값 $\pi = 4$

첫번째 $\pi = 4 - \frac{4}{3}$

두번째 $\pi = 4 - \frac{4}{3} + \frac{4}{5}$

세번째 $\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7}$

$\cdots$

연습

```python
p = 4
l = 1
n = 1
m = 1
k = int(input('반복횟수 ='))

while k >= l :
    print('p:', p,'n: ', n, 'm: ', m, 'l: ', l)
    n = n + 2
    p = p - m*(4/(n))
    l = l + 1
    m = m *(-1)
```

```python
# Leibniz수열로 파이 값 구하기

n=int(input('몇 번째까지 연산하시겠습니까?'))

i = 1
S = 0
Pi = 4
while i <= n :
    S = ((-1)**i) * (4/(2*i + 1))
    i = i + 1
    Pi = Pi + S

print(Pi)
```

```python
# Leibniz 수열을 이용하여 파이 값을 구하는 방법

pi = 4
deno = 1

for i in range(1, 10000001):
    if i % 2 == 1:
        deno = deno + 2
        pi = pi - (4/deno) # 분자는 4로 일정하여 상수 사용
#         print('aa =', pi)
    else:
        deno = deno + 2
        pi = pi + (4/deno)
#         print('bb =', pi)

print('파이 값 = ', pi)
```

연습

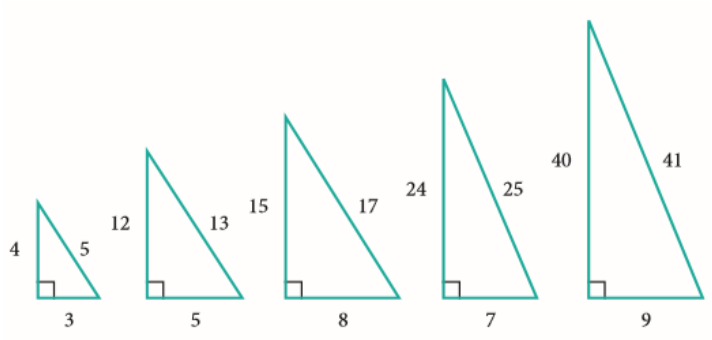
```
'''
파이 값을 구하는 방법중에 Leibniz 수열을 이용하는 방법이 있다. 위
4 - 4/3 + 4/5 - 4/7 + 4/9 - 4/11 + ... = pi
초기값 pi = 4
첫번째 pi = 4 - 4/3
두번째 pi = 4 - 4/3 + 4/5
세번재 pi = 4 - 4/3 + 4/5 - 4/7
.....
'''
to_divide = 4
divide_value = 1
pi = to_divide

for divide_value in range(3, 1000000, 2):
    if divide_value % to_divide == 3:  # 3, 7, 11, 15
        print('- ', divide_value)
        pi -= to_divide / divide_value
    else :
        print('+ ', divide_value)
        pi += to_divide / divide_value

print('Leibniz 수열을 이용한 원주율 구하기 : ', pi)
```

```
pi=0

for a in range(1, 10000, 4):
    b = ((1/a)-(1/(a+2)))
    pi = pi + b

print("Leibniz 수열을 이용한 파이 값은? ", 4*pi)
```

연습

❖ 피타고라스의 정리를 만족하는 삼각형들을 모두 찾아보자. 삼각형 한 변의 길이는 1 부터 30 이하이다.

x2+y**2==z**2**



[(3, 4, 5), (5, 12, 13), (6, 8, 10), (7, 24, 25), (8, 15, 17), (9, 12, 15), (10, 24, 26), (12, 16, 20), (15, 20, 25), (20, 21, 29)]

Solution

```
new_list = []
for x in range(1, 30):
        for y in range(x, 30):
                for z in range(y, 30):
                    if x**2+y**2==z**2:
                        new_list.append((x, y, z))
print(new_list)
```

```
2
3  for x in range(1,31):
4      for y in range(1,31):
5          for z in range(1,31):
6              if x + y > z and x < z and y < z:
7                  if (x**2 + y**2) == z**2 and x < y:
8                      print('(', x, ' ,', y, ' ,', z, ')', end = ' ')
9
```

연습

❖ 0과 1미만 사이의 실수를 돌려주는 함수 random.random() 있다. 이 함수를 이용하여 coin 함수를 만들었다.

```
import random

def coin():
    if random.random() < 0.5:
        return 1      # head
    return 0          # tail
```

❖ 위 함수를 이용하여 동전을 n번 던졌을때 헤드가 나오는 평균값을 구하는 함수를 작성하고, 100번, 1000번, 10000번 던졌을 때 나오는 평균값을 출력하는 코드를 작성하세요.

연습

```python
#코인던지기
import random

def coin():
    if random.random() < 0.5:
        return 1 # head
    else:
        return 0 # tail

def headCount(n):
    a = 0
    headTotal = 0

    while  a < n:
        headTotal =  headTotal + coin()
        a = a+1

    print(n, "번 던졌을 때  : headTotal :" , float(headTotal))
    print(n, "번 던졌을 때  : avg :" , float(headTotal/n))

count = int(input("동전던지기 횟수 : "))

headCount(count)
headCount(100)
headCount(1000)
headCount(10000)
```

```python
'''
0과 1미만 사이의 실수를 돌려주는 함수 random.random() 있다. 이 함수를
함수를 만들었다.
import random
def coin():
if random.random() < 0.5:
return 1 # head
return 0 # tail
v 위 함수를 이용하여 동전을 n번 던졌을때 헤드가 나오는 평균값을 구하는
고, 100번, 1000번, 10000번 던졌을 때 나오는 평균값을 출력하는 코드를
'''
import random

def coin():
    if random.random() < 0.5:
        return 1 # head
    return 0 # tail

# Average value of coin head function
def avg_coin_head(n):
    sum = 0;
    for k in range(1, n + 1):
        result = coin()
        if result == 1:
            sum += result
    return sum / n

print("Average value of coin head : ", avg_coin_head(100))
print("Average value of coin head : ", avg_coin_head(1000))
print("Average value of coin head : ", avg_coin_head(10000))
```