

2주. Python Basics

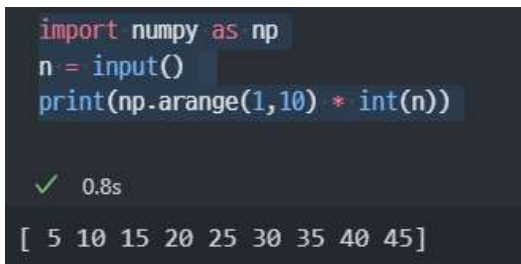
학번	32170578	이름	김산
----	----------	----	----

Q1. 화면에서 구구단의 단(2~9)를 입력받아 해당 단을 출력하는 프로그램을 작성하시오

Source code :

```
import numpy as np
n = input()
print(np.arange(1,10) * int(n))
```

실행화면 캡처:



```
import numpy as np
n = input()
print(np.arange(1,10) * int(n))
```

✓ 0.8s

[5 10 15 20 25 30 35 40 45]

Q2. 비만도 체질량지수 (BMI)는 체중(kg단위) ÷ (키(m단위))² 로 계산을 한다.

(예: 키170cm, 몸무게 85kg 이면 BMI = 85 ÷ (1.7)² 이다.

BMI 값에 따라 다음과 같이 비만도를 판단한다.

18.5 이하: 저체중, 18.5~23: 정상, 23~25: 과체중, 25~30: 비만, 30이상: 고도비만

화면에서 키(kg)와 몸무게(cm)를 입력받아 BMI 값과 비만도를 출력하는 프로그램을 작성하시오.

Source code :

```

h = input()
w = input()

bmi = float(w) / ((float(h) / 100.0) ** 2)

result = ""

if bmi <= 18.5:
    result = "저체중"
elif 18.5 < bmi and bmi <= 23:
    result = "정상"
elif 23 < bmi and bmi <= 25:
    result = "과체중"
elif 25 < bmi and bmi <= 30:
    result = "비만"
else:
    result = "고도비만"

print('BMI : {}'.format(bmi))
print("비만도 : {}".format(result))

```

실행화면 캡처: 입력 173, 60

```

BMI : 20.04744562130375
비만도 : 정상

```

Q3. 다음과 같이 높이를 입력하면 트리를 그리는 프로그램을 작성하시오

높이를 입력하시오: 5

```

*
**
***
****
*****

```

Source code :

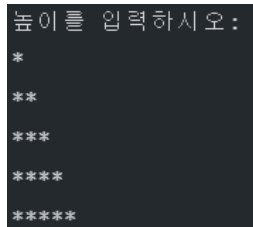
```

print("높이를 입력하시오: ")
n = input()

for i in range(int(n)):
    print("*" * (i + 1))

```

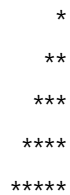
실행화면 캡처:



```
높이를 입력하시오:
*
**
***
****
*****
```

Q4. 다음과 같이 높이를 입력하면 트리를 그리는 프로그램을 작성하시오. 단, 트리를 그리는 부분을 함수로 작성하여 이용하시오.

높이를 입력하시오: 5



```
*
**
***
****
*****
```

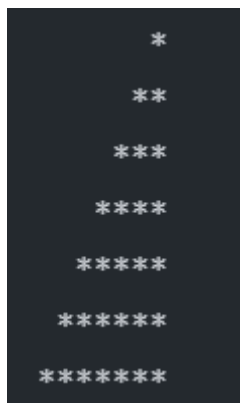
Source code :

```
print("높이를 입력하시오: ")
n = int(input())

for i in range(n):
    tmp = "*" * (i + 1)
    print("{:>{n}}".format(tmp, n = n))
```

실행화면 캡처:

입력 7



```
*
**
***
****
*****
******
*****
```

Q5. 배열 `before_arr` 에는 다음과 같은 값이 저장되어 있다.

{ 7, 1, 10, 4, 6, 9, 2, 8, 15, 12, 17, 19, 18 }

`before_arr` 에 있는 값들을 오름차순으로 정렬하여 배열 `after_arr` 에 저장하는 프로그램을 작성하시오. 단, 알고리즘은 다음과 같다.

- 1) `before_arr`에서 가장 작은 값을 찾는다.
- 2) 그 값을 `after_arr` 에 앞쪽에 넣는다.
- 3) `before_arr`에서 찾은 가장 작은 값은 999 로 변경한다.
- 4) 1)~3) 과정을 배열의 길이만큼 반복한다.

Source code :

```
import sys
before_arr = [7, 1, 10, 4, 6, 9, 2, 8, 15, 12, 17, 19, 18]
after_arr = []

# 4. 배열의 길이만큼 반복한다
for idx in range(len(before_arr)):
    minVal = 999
    minIdx = 0

    # 1. before_arr에서 가장 작은 값을 찾는다
    for i in range(len(before_arr)):
        if (minVal > before_arr[i]):
            minVal = before_arr[i]
            minIdx = i

    # 2. 그 값을 after_arr의 앞쪽에 넣는다.
    after_arr.insert(idx, minVal)

    # 3. before_arr에서 찾은 가장 작은 값은 999로 변경한다
    before_arr[minIdx] = 999

print(before_arr)
print(after_arr)
```

실행화면 캡처:

```
[999, 999, 999, 999, 999, 999, 999, 999, 999, 999, 999, 999, 999]
[1, 2, 4, 6, 7, 8, 9, 10, 12, 15, 17, 18, 19]
```

Q6. 길이가 같은 두 정수 배열에서 동일 인덱스 위치의 값을 곱하여 새로운 배열에 저장하여 반환하는 함수 `mul_arr()`을 작성하고 제대로 작동하는지 테스트 하시오.

예) $A = \{ 1, 2, 3 \}$, $B = \{ 4, 5, 6 \}$ 일 때 `mul_arr(A,B)` 는 $\{ 4, 10, 18 \}$ 를 return 한다.

Source code :

```
def mul_arr(A, B):
    return (np.array(A) * np.array(B)).tolist()

a = [1,2,3,4]
b = [4,5,6,7]

print(mul_arr(a,b))
```

실행하면 캡처:

`[4, 10, 18, 28]`

Q7. 다음과 같은 구조의 numpy array 인 `my_arr` 를 생성하여 내용을 보이시오.

- 10행 5열의 구조
- array의 내용은 1~50 사이의 정수.
- 1~50의 정수로 1차원 array를 만든 뒤 `reshape` 으로 2차원 배열로 변형

Source code :

```
# 1 ~ 50의 정수로 1차원 array를 만듦
my_arr = np.random.rand(1,50)
my_arr = np.round(my_arr * 50).astype(int) + 1

print(my_arr)

# reshape로 1차원 배열을 2차원 배열(10행 5열)로 변형
my_arr = my_arr.reshape(10,5)

print(my_arr)
```

실행하면 캡처:

```
[[31 32 26 17 34 11  1 38 23 36 10  4 20 36  6 11 24 32 39  6 32 20 35 11
 27 10  2 25 10 33 28 40 21 50 27 25 38 30 19 25 10 21  2 24 15 27  2 29
 16 30]]
[[31 32 26 17 34]
 [11  1 38 23 36]
 [10  4 20 36  6]
 [11 24 32 39  6]
 [32 20 35 11 27]
 [10  2 25 10 33]
 [28 40 21 50 27]
 [25 38 30 19 25]
 [10 21  2 24 15]
 [27  2 29 16 30]]
```

Q8. Q7에서 생성한 `my_arr` 에 대해 다음의 문제를 해결하는 코드를 작성하고 실행 결과를 보이시오

- (1) `my_arr` 의 값들에 각각 2를 곱한 결과를 보이시오
- (2) `my_arr` 의 값들중 20 이하의 값들에 대해서만 100을 더한 후에 `my_arr` 에 저장하시오. `my_arr` 의 내용을 보이시오
- (3) `my_arr` 에서 2,3열의 데이터만 잘라서 보이시오.
- (4) `my_arr` 에서 5~8행의 데이터만 잘라서 보이시오.

Source code :

```
# (1) my_arr 의 값들에 각각 2를 곱한 결과를 보이시오
print(my_arr * 2)

# (2) my_arr 의 값들중 20 이하의 값들에 대해서만 100을 더한 후에 my_arr 에 저장하시오. my_arr 의 내용을 보이시오
my_arr[my_arr <= 20] += 100
print(my_arr)

# (3) my_arr 에서 2,3열의 데이터만 잘라서 보이시오.
print(my_arr[:,1:3])

# (4) my_arr 에서 5~8행의 데이터만 잘라서 보이시오.
print(my_arr[4:8, :])
```

실행화면 캡처:

```
# (1) my_arr 의 값들에 각각 2를 곱한 결과를 보이시오
print(my_arr * 2)
```

✓ 0.7s

```
[[78 30 38 32 44]
 [24 34 14 98 52]
 [94 76 48 78 64]
 [32 54 74 32  4]
 [28 82 94 64 16]
 [78 54 78 88 96]
 [68 88 76 74 88]
 [94 92 42 10 84]
 [56 56 84  6 90]
 [ 6 72 60 66 40]]
```



```
# (2) my_arr 의 값들중 20 이하의 값들에 대해서만 100을 더함
my_arr[my_arr <= 20] += 100
print(my_arr)
```

✓ 0.8s

```
[[ 39 115 119 116  22]
 [112 117 107  49  26]
 [ 47  38  24  39  32]
 [116  27  37 116 102]
 [114  41  47  32 108]
 [ 39  27  39  44  48]
 [ 34  44  38  37  44]
 [ 47  46  21 105  42]
 [ 28  28  42 103  45]
 [103  36  30  33 120]]
```



```
# (3) my_arr 에서 2,3열의 데이터만 잘라서 보이시오.
print(my_arr[:,1:3])
```

✓ 0.8s

```
[[115 119]
 [117 107]
 [ 38  24]
 [ 27  37]
 [ 41  47]
 [ 27  39]
 [ 44  38]
 [ 46  21]
 [ 28  42]
 [ 36  30]]
```



```
# (4) my_arr 에서 5~8행의 데이터만 잘라서 보이시오.
print(my_arr[4:8, :])
```

✓ 0.1s

```
[[114  41  47  32 108]
 [ 39  27  39  44  48]
 [ 34  44  38  37  44]
 [ 47  46  21 105  42]]
```

Q9. slide 46의 그래프 작성 코드를 참조하여 다음의 월별 강수량을 그래프로 나타내시오

1월	2월	3월	4월	5월	6월	7월	8월	9월	10월	11월	12월
20	22	37	79	90	109	288	277	140	50	48	19

Source code :

```
import matplotlib.pyplot as plt

x = np.arange(1,13)
y = np.array([20, 22, 37, 79, 90, 109, 288, 277, 140, 50, 48, 19])
plt.plot(x,y)

plt.title("monthly precipitation")

plt.xlabel("month")
plt.ylabel("precipitation(mm)")

plt.xlim(1,12)
plt.grid(True)
plt.show()
```

실행화면 캡처:

