WEEK 1. Basic python 1

Python 출력 함수 및 주석

- ctrl + alt + n : 터미널 실행
- 프린트를 꼭 써줘야 출력이 된다.

1. 문자열 - 홑따옴표, 쌍따옴표

- print('Hello Python') / print(“Hello Python”)

2. 정수 및 실수 출력

- 사칙연산 결과 출력 : (+ - * /) 사용 / , 사용하여 여러 개의 결과 산출 가능
- print(10) / print(10 + 11, 12 - 13) / print(4 * 5, 6 / 3)

3. 혼합 출력

- print('10 + 11 =', 10 + 11)
- print('12 / 6 =', 12 / 6)

4. 출력 함수 option

(1) Sep

- sep : 쉼표로 구분하여 나타나는 결과에 대해서 sep 로 설정한 문자열이 쉼표를 대신하게 된다.
- sep 는 기본값이 원래 공백이다. sep 설정 없이 입력하면 공백이 추가된다.
- sep은 마지막에 한 번만 적용한다.
- print('010', '1234', '5678', sep='-')
- print('나이', 30, sep=':')

(2) end

- end ' ' : 다음줄로 넘어가지 않게 해준다. 이 역시 기본값은 end='₩n'
- print(10, end='%')
- print(30, end='$')

(3) 이스케이프 문자

이스케이프 문자	기능
\n	다음 줄로 이동 (개행)
\r	해당 줄의 처음으로 이동
\t	8 칸 공백
\'	' 문자
\"	" 문자
\\	\ 문자

이스케이프 문자는 문자열을 출력하기 위해서 사용 되는 기능 외의 부가적인 기능을 사용하기 위해서 쓰인다.

- ₩n : 개행(다음 줄로 이동)
- ₩r : carriage return(해당줄의 처음으로 이동) - 수정 모드로 출력된다. 해당 줄의 처음으로 이동 -> 이동 줄의 처음으로
- ₩t : tab 공백(8 칸 공백) - 앞 두칸 포함해서 8 칸, 두번 쓰면 16 칸
- ₩' : 홀 따옴표 문자
- ₩" : 쌍 따옴표 문자 - 꼭 이스케이프 문자로서 쓰일 필요는 없다.
- ₩₩ : 역슬래쉬 문자
- ₩b : 앞으로 한칸 이동하는데 수정 모드로 출력된다.

(4) Comment(주석)

- # 주석 : 코드를 실행하지 않게 한다. 불필요한 코드를 실행하지 않게 하거나 특정 코드에 대한 설명이 필요한 경우
- ' ' ' 주석 블럭: 설명해놓는것.

문제 앞에서 학습한 내용을 바탕으로 다음을 출력하시오.

1.

이름:최수지

전화번호:010-1234-5678

주소:서울시 종로구 종로3가

2.

"C:₩Program Files₩Python35₩Scripts₩"

"C:₩Users₩admin₩temp₩"

3.

```
3.
                #### 회비 정보 ####
=============================================
이름          나이     전화번호          회비
=============================================
김동완        38       010-1111-1111     ₩20,000
서지수        24       010-1234-5678     ₩30,000
이지은        25       010-2525-2345     ₩50,000
---------------------------------------------
총합계                                   ₩100,000
=============================================
```

Python 내장함수

1. 크기 비교 함수

(1) Max, min

- Max(3,7,-1,5)
- Min(3,7,-1,5)

2. 연산 함수

- Sum
- pow : 거듭제곱
- divmod = 몫, 나머지

3. 진법 변환 함수

- 수를 표현하기 위한 방식

(1) 진법 표현 방법

진법	표현 문자	표현 식
2	0, 1	0b
8	0, 1, 2, 3, 4, 5, 6, 7	0o
10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9	
16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F	0x

- 0b100 / 0o100 / 100 / 0x100

(2) 진법 변환

- bin() - 2 진수 값으로 변환
- oct() - 8 진수
- hex() - 16 진수

4. 그 외 함수

(1) Round

- round(123.567,2) 반올림, 소수점 자리
- 분수도 가능하다.
- round(11.56,1) / round(5/3,3)

(2) abs : 절대값

Python 서식 문자

1. 서식 문자

(1) 기본 서식 문자

C 스타일	Python 3	설명
%s	{ }	문자열 출력
%d	{ }	정수 출력
	{:b}	표현식 없는 2진수 값 출력
%o	{:o}	표현식 없는 8진수 값 출력
%x	{:x}	표현식 없는 16진수 값 출력
%f	{:f}	실수 출력
%.2f	{:.2f}	소수점 2자리 까지의 실수 출력
%6d	{:6}	6자리 고정 출력

(2) 문자열과 정수 출력

- print('{}: {}'.format('나이', 30))

(3) 실수 출력

- print('%f,%.2f' % (1.123, 1.123))
- print('{:f}, {:.2f}'.format(1.123, 1.123))

(4) 진법 출력

- print('%o, %x, %X' % (10, 10, 10))
- print('{:b}, {:o}'.format(10, 10))

- print('{:x}, {:X}'.format(10, 10))

(5) 고정 길이 출력

- print('|{:5}|'.format(123))
- print('|{:5}|'.format('abc'))

(6) 고정 길이 정렬

- print('|{:<5}|'.format(123))
- print('|{:>5}|'.format('abc'))
- print('|{:^5}|'.format('abc'))

(7) 여백 채우기

- print('|{:05}|'.format(123))
- print('|{:_>5}|'.format('abc'))
- print('|{:-^5}|'.format('abc'))

(8) 정수, 실수 단위 구분

- print('{:,}'.format(1000000))
- print('{:,.2f}'.format(1000000))

Python 변수

1. 변수

(1) 작명 규칙

1) 알파벳, 숫자, 언더스코어(_)로 구성

2) 알파벳은 대/소문자 구분

3) 한글 사용 가능
4) 변수명의 시작은 숫자로 할 수 없음
5) 공백이나 특수 기호는 포함 할 수 없음

- 공백은 불가능하기 때문에 user name 대신 user_name 을 쓴다.

6) Python 예약어는 사용하면 안된다.
- if, while, true, false, break, continue, pass, return 등 변수로서 사용 불가능하다.
- 내장함수도 역시 사용하지 않는 것이 좋다.

(2) 자료형 종류

1) 부울형 : True, False 만을 가지는 값 (bool)

2) 정수 : 0 과 음수, 양수 값을 포함하는 숫자 값 (int)

3) 실수 : 소수점을 사용하는 숫자 값 (float)

4) 문자열 : 따옴표로 묶여 있는 값 (str)

5) 리스트 : 정수, 실수 및 문자열 등 자료들의 집합 (값의 집합) (list) - [ ] 사용
6) 튜플 : 정수, 실수 및 문자열 등 자료들의 집합 (값의 집합) (tuple) - ( ) 사용
7) 사전 : 정수, 실수, 및 문자열 등 자료들의 집합 (키와 값이 쌍으로 존재) (dict) - { }사용

(3) 변수 정의

- 변수명 = 값
- 변수명 1, 변수명 2 = 값 1, 값 2
- 값(오른쪽)을 변수명(왼쪽)에 저장시키겠다는 것을 할당이라고 한다. =을 사용하여 할당한다.

2. 자료형

(1) 자료형 확인

- Type(변수명)

(2) 자료형 변환

함수	설명
bool()	부울형 자료로 변환
int()	정수형 자료로 변환
str()	문자열 자료로 변환
float()	실수형 자료로 변환

문제

다음의 변수에 저장되어 있는 값을 활용하여 동일한 결과가 나오도록 하시오.

x, y, z = '100', 10.5, 20

1. 110.5
2. 10020
3. 10.520.0
4. 110.520

Python 연산자

1. 연산자

(1) 산술 연산자

연산자	예제	설명
+	3 + 2	두 값을 더한 결과를 반환
-	3 – 2	두 값을 뺀 결과를 반환
*	3 * 2	두 값을 곱한 결과를 반환
/	3 / 2	두 값을 나눈 결과를 반환(실수 값)
//	3 // 2	두 값을 나눈 결과의 몫 반환(정수 값)
%	3 % 2	두 값은 나눈 결과의 나머지 반환
**	3 ** 2	거듭 제곱의 결과 반환

(2) 비교 연산자

연산자	예제	설명
==	3 == 3	두 피 연산자 값을 비교하여 동일하면 True, 동일하지 않으면 False
!=	3 != 2	두 피 연산자 값을 비교하여 동일하면 False, 동일하지 않으면 True
>	3 > 2	두 피 연산자 값을 비교하여 왼쪽의 값이 크면 True, 그렇지 않으면 False
<	2 < 3	두 피 연산자 값을 비교하여 왼쪽의 값이 작으면 True, 그렇지 않으면 False
>=	3 >= 2	두 피 연산자 값을 비교하여 왼쪽의 값이 크거나 같으면 True, 그렇지 않으면 False
<=	3 <= 3	두 피 연산자 값을 비교하여 왼쪽의 값이 작거나 같으면 True, 그렇지 않으면 False

(3) 논리 연산자

연산자	예제	설명
and	True and True True and False	두 피 연산자가 전부 True인 경우에만 True (논리곱)
or	True or True True or False	두 피 연산자가 전부 False인 경우에만 False (논리합)
not	not True not False	오른쪽 피 연산자에 대한 부정

(4) 멤버 연산자

연산자	예제	설명
in	1 in (1, 2, 3)	왼쪽 피 연산자의 값이 오른쪽 피 연산자 멤버 중 일치하는 값이 존재 하면 True
not in	1 not in (1, 2, 3)	왼쪽 피 연산자의 값이 오른쪽 피 연산자 멤버 중 일치하는 값이 존재 하지 않으면 True

(5) 식별 연산자

연산자	예제	설명
is	type(1) is int	두 피 연산자의 식별 값을 비교하였을 때 동일한 객체이면 True
is not	type('1') is not int	두 피 연산자의 식별 값을 비교하였을 때 동일한 객체이면 False

(6) 비트 연산자

연산자	예제	설명
&	10 & 5	두 피 연산자의 and 비트 연산을 수행 한다.
|	10 | 5	두 피 연산자의 or 비트 연산을 수행 한다.
^	10 ^ 5	두 피 연산자의 xor 비트 연산을 수행 한다.
<<	10 << 2	왼쪽 피 연산자의 비트를 왼쪽으로 2개 비트 이동
>>	10 >> 2	왼쪽 피 연산자의 비트를 오른쪽으로 2개 비트 이동

Python 랜덤 함수

1. **random**
   - from random import random
   - 내장함수가 아니지만 내장시키기 위한 작업이 필요하다.
   - from (file) import (function)

- print(random()) - 호출한다(call)
- from random import random 통해서 인터프리터 상에 임의의 값을 호출한다.
- ex) print(random( )) : 0.0~1.0 미만의 임의의 값 생성
- ex) print(random( )*10) : 0.0~10.0 미만의 임의의 값 생성
- ex) print(int(random( )*10)) : 0~10 미만의 임의의 값 생성
- ex) print(int(random( )*10) + 1) : 1~10 까지의 임의의 값 생성

2. 그 외 랜덤 함수

(1) randint

- from random import randint - 내가 설정한 값에서 난수를 추출
- ex) randint(1,10) 1~10 까지의 임의 값 생성

(2) randrange

- from random import randrange
- ex) randrange(1,10) – 1~ 10 미만의 임의 값 생성
- ex) randrange(2,10,2) - 2~10 미만에서 2 부터 2 씩 증가된 값에 대해 10 미만의 임의 값 생성

3. 임의의 문자 생성

(1) Chr

- chr() - 아스키 코드로 변환

```
from random import randint
print(chr(randint(65, 90))) # 'A', 'Z' 임의의 대문자 나옴
print(chr(randint(97, 122))) # 'a', 'z'임의의 소문자 나옴
```

문제

1. 랜덤 함수를 사용하여 생성된 값이 짝수 또는 홀수 인지를 구분하는 코드를 작성하시오.
2. 랜덤 함수를 사용하여 생성된 2 개의 값을 빼기 계산 할 때 항상 양의 정수가 나올 수 있도록 하시오.
3. 랜덤 함수를 사용하여 생성된 2 개의 값을 홀/짝 비교하였을 때 2 개의 값이 전부 홀수 또는 짝수이면, 2 개의 정수 값을 더하고 2 개의 값이 홀-짝 또는 짝-홀이면, 2 개의 정수 값을 곱하시오.

Python 조건문

1. 조건문

(1) 기본 사용법

```
if 조건식 :
<-----> 수행코드
indent - 들여쓰기
```

- 바로 수행코드 - 들여쓰기가 안 되어 있음. 개별 코드
- tap -> 들여쓰기
- 들여써진 코드에 대해서 들여쓰기 싫을 때 shift + tap -> 내어쓰기
  - x = 15

    ```
    if x > 10:
        print('x 는 10 보다크다.')
    ```

(2) If ...else 문

```
number = int(input('정수값입력: '))
if x % 3 == 0:
    print('3의배수이다.')
else :
    print('3 의배수가아니다.')
```

(3) If...elif...else 조건문

```
score = int(input('점수입력: '))
if 90 <= score <= 100:
    print('수입니다.')
elif 80 <= score < 90:
    print('우입니다.')
elif 70 <= score < 80:
    print('미입니다.')
elif 60 <= score < 70:
    print('양입니다.')
else:
    print('가입니다.')
```

- 중첩 if 조건문도 가능하다. If 아래에 다른 if 조건식을 사용할 수 있다.

문제

사용자로부터 이름, 키, 체중 값을 입력 받아 비만도를 구하고 결과를 출력 할 때 비만도 값 100 을 기준으로 100 미만이면 저체중, 100 이상 110 미만은 정상, 110 이상 120 미만 과제중,120 이상 130 미만 비만, 130 이상은 고도비만으로 출력 하시오.

비만도 계산 식 : 비만도(%) = 현재 체중 / 표준 체중 * 100
표준 체중 계산 식 : 표준 체중 = (현재 키 – 100) * 0.9

```
name = input('이름 입력 : ')
tall = float(input('키 입력(cm) : '))
weight = float(input('체중 입력(kg) : '))

std_weight = (tall - 100) * 0.9
fat_rate = weight / std_weight * 100

if fat_rate < 100:
    fat_str = '저체중'
elif 100 <= fat_rate < 110:
    fat_str = '정상'
elif 110 <= fat_rate < 120:
    fat_str = '과체중'
elif 120 <= fat_rate < 130:
    fat_str = '비만'
elif 130 <= fat_rate:
    fat_str = '고도비만'

print('{}님의 비만도는 {:.2f}% 로 {} 입니다.'₩
.format(name, fat_rate, fat_str))
```

Python 반복문

1. **For 반복문**
   (1) 기본 사용법

```
for 변수명 in range(반복횟수) :
    수행 코드
```

   (2) Range 함수 응용

```
for x in range(10):
    print(x)
for x in range(5, 10):
    print(x)
```

```
for x in range(1, 10, 2):
    print(x)
```

(3) Range 함수 대신 사용하는 방법

```
For char in 'abcde':
    print(char)
for tup in(1, 2, 3, 4, 5):
    print(tup)
```

(4) 중첩 반복

- 메인 반복 * 서브 반복

```
for x in range(1,10):
    for y in range(1,10):
        print(x*y)
```

문제

1. a ~ z 까지 임의의 문자를 생성하여 출력하는 코드를 작성, 임의의 문자는 총 16 자리 생성을 한다.

```
from random import randint
for x in range(16):
    print(chr(randint(97, 122)), end='')
print()
```

2. 1 ~ 20 까지의 누적 합을 구하는 코드를 작성하시오

ex) 1 + 2 + 3 + 4 + 5 + ... + 20 = ???

```
tot = 0
# tot = tot + 1    # tot = 0 + 1
# tot = tot + 2    # tot = 1 + 2
# tot = tot + 3    # tot = 3 + 3
# tot = tot + 4    # tot = 6 + 4
for x in range(1, 21):
    tot = tot + x
print(tot)
```

3. 구구단을 출력하시오.

2. while 반복문

(1) 기본 사용법

```
While 조건문 :
    수행 코드
```

(2) 비교 연산자 사용

```
x = 0
while x < 3 :
    수행 코드
    x = x + 1
```

(3) 멤버 연산자 사용

```
x = 0
while x in (0, 1, 2):
    수행 코드
    x = x + 1
```

(4) 무한 반복

```
while True :
    print(x)
    x = x + 1
```

- Ctrl + C 단축키로 강제 종료

3. break, continue 문

(1) 반복 종료

```
x = 0
while True :
    if x == 5 :
        break
    print(x)
    x = x + 1
```

(2) 반복의 처음으로 이동

```
x = 0
while True :
```

```
if x % 2 == 0 :
        continue - 반복의 처음으로 돌아간다.
print(x)
x = x + 1
```

- line 을 맞춰야 한다. continue, break 의 위치에 따라서 main 인지 sub 인지 구분된다.
- continue 의 경우 main 줄과 맞춰져 있다면 main 반복의 처음으로 돌아가는 것이다.

문제

1. 1 ~ 20 까지의 정수 값을 출력하는 코드를 작성 하시오.(while 문으로만 작성)

2. 1 ~ 100 까지의 누적 합을 구하는 코드를 작성 하시오.(while 문으로만 작성)

3. 사용자가 입력한 값을 초과하지 않는 한도에서의 누적 합을 구하는 코드를 작성 하시오