

파이썬프로그래밍 2분반

<10주차 과제>

정보보안공학과

202121556

곽지현

2023-05-15

In []: # 1. while문의 구조 이해를 위해 다음 코드를 실행하여 결과를 확인해보세요.

```
while False: # False라서 반복문이 실행되지 않는다.
    print("조건이 False라서 반복 실행되지 않음")
```

```
while True: # True라서 반복문이 계속 실행된다.
```

```
    print("조건이 True라서 반복 실행 됨. Wn 계속 True라서 무한반복 상태. ctrl+c를 눌러 종료")
```

조건이 True라서 반복 실행 됨.

계속 True라서 무한반복 상태. ctrl+c를 눌러 프로그램 강제 종료 필요

조건이 True라서 반복 실행 됨.

계속 True라서 무한반복 상태. ctrl+c를 눌러 프로그램 강제 종료 필요

조건이 True라서 반복 실행 됨.

계속 True라서 무한반복 상태. ctrl+c를 눌러 프로그램 강제 종료 필요

조건이 True라서 반복 실행 됨.

계속 True라서 무한반복 상태. ctrl+c를 눌러 프로그램 강제 종료 필요

조건이 True라서 반복 실행 됨.

계속 True라서 무한반복 상태. ctrl+c를 눌러 프로그램 강제 종료 필요

조건이 True라서 반복 실행 됨.

계속 True라서 무한반복 상태. ctrl+c를 눌러 프로그램 강제 종료 필요

조건이 True라서 반복 실행 됨.

계속 True라서 무한반복 상태. ctrl+c를 눌러 프로그램 강제 종료 필요

조건이 True라서 반복 실행 됨.

계속 True라서 무한반복 상태. ctrl+c를 눌러 프로그램 강제 종료 필요

조건이 True라서 반복 실행 됨.

계속 True라서 무한반복 상태. ctrl+c를 눌러 프로그램 강제 종료 필요

조건이 True라서 반복 실행 됨.

계속 True라서 무한반복 상태. ctrl+c를 눌러 프로그램 강제 종료 필요

In [4]: # 2. for문 처럼 반복변수를 사용하는 while문의 이해를 위해 다음 코드를 실행하여 결과

```
cnt = 0 # cnt에 0을 대입
```

```
while cnt < 5: # cnt가 5가 되기전 까지 반복,
                # 5가 되면 False라서 반복문이 실행되지 않는다.
```

```
    print(cnt, "번째 반복") # cnt와 "번째 반복"을 출력
```

```
    cnt += 1 # cnt에 1을 증가하여 대입
```

0 번째 반복

1 번째 반복

2 번째 반복

3 번째 반복

4 번째 반복

In [5]: # 3. 상태를 기반으로 반복하는 while문의 이해를 위해 다음 코드를 실행하여 결과를 확인

```
list2 = ["a", "b", "c", "d"] # 리스트 선언
```

```
while len(list2) != 0: # 리스트의 길이가 0이 아닐때 까지 반복,
                        # 0이 되면 False이기 때문에 반복문 실행 안됨
```

```
    value = list2[0] # value에 리스트의 0번째 인덱스를 대입
```

```
    list2.remove(value) # 리스트에 value의 값 제거
```

```
    print(value, end=", ") # value 출력
```

```
print(list2) # list2 출력
```

a, b, c, d, []

In [3]: # 4. 시간을 기반으로 반복하는 while문의 이해를 위해 다음 코드를 실행하여 결과를 확인

```
import time # 시간과 관련된 기능을 가져옴

cnt = 0 # cnt에 0을 대입
start_time = time.time() # start_time에 현재 시,분,초를 초단위로 바꾼값을 대입
while time.time() <= start_time + 5:
    # time.time()이 start_time + 5 보다 값이 크면 반복 종료
    cnt += 1 # cnt에 1을 증가하여 대입
end_time = time.time() # end_time에 현재 시,분,초를 초단위로 바꾼값을 대입
print("{}초 동안 {}번 반복했습니다.".format(end_time-start_time, cnt))
# end_time에서 start_time를 뺀 값과, cnt의 값을 출력
```

5.004804372787476초 동안 30527049번 반복했습니다.

In [2]: # 5. break 키워드의 이해를 위해 다음 코드를 실행하여 결과를 확인해보세요.

```
list5 = [1, "2", 3, "4"] # 리스트 선언
idx = 0 # idx에 0을 대입
search_target = 3 # search_target에 3을 대입

print("start scan list to find value({})".format(search_target))
# search_target의 값 출력
print("scanning", end="") # "scanning"을 출력
while True: # True라서 반복문이 계속 실행
    print(".", end="") # "." 출력

    if list5[idx] == search_target: # list5의 [idx]번째 인덱스 값이 3이라면
        print("! value({}) is list[{}]" .format(search_target, idx))
        # search_target의 값과 idx의 값 출력
        break # 반복 종료

    if idx+1 == len(list5): # idx+1의 값과 list5의 길이와 같으면
        print("⌘ Not Found..") # "⌘ Not Found.."를 출력
        break # 반복 종료

    idx += 1 # idx에 1을 증가하여 대입
```

start scan list to find value(3)
scanning...! value(3) is list[2]

In [9]: # 6. continue 키워드의 이해를 위해 다음 코드를 실행하여 결과를 확인해보세요.

```
list6 = [1, "2", "3", "4", 5] # 리스트 선언
string_cnt = 0 # string_cnt에 0을 대입

while True: # True라서 반복문이 계속 실행
    if len(list6) == string_cnt: # list6의 길이와 string_cnt의 값과 같으면
        break # 반복 종료

    value = list6[string_cnt] # value에 list6[string_cnt]를 대입
    if type(value) != str: # value의 타입이 str이 아니라면
        list6.remove(value) # value 제거
        continue # 반복문 처음으로 돌아감
    else: # value의 타입이 str이라면
        string_cnt += 1 # string_cnt에 1을 증가하여 대입

print("문자열 요소 개수: {}, 결과: {}".format(string_cnt, list6))
# string_cnt의 값과 list6를 출력
```

문자열 요소 개수: 3, 결과: ['2', '3', '4']

In [7]: # 7. 적절한 입력을 받도록 작성된 계산기 프로그램입니다. 다음과 같은 결과가 나오도록

```
while True: # True라서 반복문이 계속 실행
    num1 = input("첫번째 10진 정수 입력 > ").strip() # 문자 입력받기
    if num1.isdecimal(): # num1이 정수 형태인지 확인
        num1 = int(num1) # num1에 입력받은 값을 정수로 대입
        break # 반복 종료
    else: # 정수가 아니라면
        print("잘못된 입력을 하셨습니다.") # "잘못된 입력을 하셨습니다."를 출력

while True: # True라서 반복문이 계속 실행
    num2 = input("두번째 10진 정수 입력 > ").strip() # 문자 입력받기
    if num2.isdecimal(): # num2가 정수 형태인지 확인
        num2 = int(num2) # num2에 입력받은 값을 정수로 대입
        break # 반복 종료
    else: # 정수가 아니라면
        print("잘못된 입력을 하셨습니다.") # "잘못된 입력을 하셨습니다."를 출력

while True: # True라서 반복문이 계속 실행
    op = input("+, -, *, / 중 하나 입력 > ").strip() # 문자 입력받기
    if len(op) >= 2: # op의 길이가 2 이상이면
        print("잘못된 입력을 하셨습니다. 한 글자만 입력해주세요.")
        # "잘못된 입력을 하셨습니다. 한 글자만 입력해주세요."를 출력
    elif op not in "+-*/": # "+-*/"중에 op가 없다면
        print("잘못된 입력을 하셨습니다. +, -, *, / 중 하나를 입력해주세요.")
        # "잘못된 입력을 하셨습니다. +, -, *, / 중 하나를 입력해주세요."를 출력
    else: # 입력이 잘 되면
        break # 반복 종료

result = 0 # result에 0을 대입
if op == "+": # op가 "+"와 같다면
    result = num1 + num2 # result에 num1 + num2의 값 대입
if op == "-": # op가 "-"와 같다면
    result = num1 - num2 # result에 num1 - num2의 값 대입
if op == "*": # op가 "*"와 같다면
    result = num1 * num2 # result에 num1 * num2의 값 대입
if op == "/": # op가 "/"와 같다면
    result = num1 / num2 # result에 num1 / num2의 값 대입

print("{} {} {} = {}".format(num1, op, num2, result))
# num1, op, num2, result의 값 출력
```

첫번째 10진 정수 입력 > 1a

잘못된 입력을 하셨습니다.

첫번째 10진 정수 입력 > 12.1

잘못된 입력을 하셨습니다.

첫번째 10진 정수 입력 > 10

두번째 10진 정수 입력 > 2

+, -, *, / 중 하나 입력 > +-

잘못된 입력을 하셨습니다. 한 글자만 입력해주세요.

+, -, *, / 중 하나 입력 > ?

잘못된 입력을 하셨습니다. +, -, *, / 중 하나를 입력해주세요.

+, -, *, / 중 하나 입력 > *

10 * 2 = 20

In [48]: # 8. 입력한 값의 누적합을 구하는 프로그램입니다. 다음과 같은 결과가 나오도록 빈칸을

```
print("==== 누적합 더하기 프로그램 =====")
# "==== 누적합 더하기 프로그램 ====="을 출력
result = 0 # result에 0을 대입
string = "" # string에 "" 대입

print("계속 숫자 입력, 0 입력 시 마침")
# "계속 숫자 입력, 0 입력 시 마침"을 출력
while True: # True라서 반복문이 계속 실행

    if string: # string라면
        print(string, end=" ") # string 출력

    num = input().strip() # 문자 입력받기

    if float(num) == 0: # num이 0과 같다면
        string = "{}= {}".format(string[:-2], result)
        # string[:-2]와 result의 값 출력
        print(string) # string을 출력
        break; # 반복 종료

    if float(num): # num이 실수라면
        result += float(num) # result에 float(num)값을 더하여 대입
    else: # 정수라면
        result += int(num) # result에 int(num)값을 더하여 대입
    string += "{} + ".format(num) # num의 값 출력
```

==== 누적합 더하기 프로그램 =====

계속 숫자 입력, 0 입력 시 마침

1

1 + 2

1 + 2 + 4.2

1 + 2 + 4.2 + -1

1 + 2 + 4.2 + -1 + 5.2

1 + 2 + 4.2 + -1 + 5.2 + 0

1 + 2 + 4.2 + -1 + 5.2 = 11.4