

문제해결을 위한 코딩 첫걸음

⋮ 7장 반복문에 대해 알아보시다_break문, continue문

한성대학교 노은희 교수



학습 목표

- 반복문의 필요성에 대해 이해합니다.
- for문과 range문의 사용에 대해 이해합니다.
- while문 사용에 대해 이해합니다.
- break문, continue문에 대해서 이해합니다.



7.3 while문

- while문
 - 사전 조건을 만족 하는 동안 반복을 수행하는 구조

```
while condition :
```

```
    실행문장1
```

```
    ...
```

condition의 결과가 참인 경우 문장을 반복하여 수행

```
while True:
```

```
    print("☆☆☆",end="")
```

Ctrl + C키를 누르면 중지

The screenshot shows a Python 3.8.2 Shell window. The main content is a grid of blue stars (☆☆☆) arranged in 10 rows and 20 columns. Below the grid, a red traceback error is displayed, indicating a KeyboardInterrupt. The error message reads: "Traceback (most recent call last): File 'C:/Users/노은희/Documents/python/source/chapter08/추가파일/while_true.py', line 3, in <module> print('☆☆☆',end='') KeyboardInterrupt >>>". The status bar at the bottom right shows "Ln: 9 Col: 4".



7.4 break문과 continue문

- break문
 - 반복적인 loop를 빠져 나오기 위해 사용
 - “☆☆☆” 100까지 출력하고 빠져나옴

```
cnt = 1  
while True:  
  
    print("☆☆☆",end=" ")  
    print("cnt=",cnt)  
    cnt = cnt + 1  
  
    if cnt > 100 :  
        break
```

```
Python 3.8.2 Shell  
File Edit Shell Debug Options Window Help  
☆☆☆ cnt= 86  
☆☆☆ cnt= 87  
☆☆☆ cnt= 88  
☆☆☆ cnt= 89  
☆☆☆ cnt= 90  
☆☆☆ cnt= 91  
☆☆☆ cnt= 92  
☆☆☆ cnt= 93  
☆☆☆ cnt= 94  
☆☆☆ cnt= 95  
☆☆☆ cnt= 96  
☆☆☆ cnt= 97  
☆☆☆ cnt= 98  
☆☆☆ cnt= 99  
☆☆☆ cnt= 100  
>>>
```




7.4 break문과 continue문

- break문
 - 반복적인 loop를 빠져 나오기 위해 사용
 - 1부터 10까지의 정수 합을 구하는 프로그램

[소스코드] 7-14.py

```
sum = cnt = 0
while True :
    sum += cnt
    cnt += 1
    if cnt > 10 :
        break
print(sum)
```

while문안의 실행문을 무한 반복

cnt > 10 경우 while 반복문을 빠져나옴

[실행결과]

55

7.4 따라 해보기 - 음수값 입력 시 결과 출력

- 키보드로부터 점수를 입력 받아 그 합을 출력하는 프로그램
- 키보드로부터 입력된 값이 음수이면 loop를 빠져 나와 결과 값 출력

[소스코드] 7-15.py

```
sum = score = 0
while True :
    score=int(input("type a score:"))
    if score < 0 :
        break
    sum += score
print("Total score is "+str(sum))
```

[실행결과]

```
type a score: 100
type a score: 200
type a score: 50
type a score: -10
Total score is 350
```



7.4 따라 해보기 - 난수 맞추기

- 컴퓨터가 발생시킨 1부터 12까지의 3번 안에 난수를 맞추는 프로그램
- 사용자가 추정한 값이 맞으면, 'Your guess right!' 메시지 출력
- 틀렸을 경우에는 숫자의 크기를 알려줌

```
import random
```

[소스코드] 7-16.py

```
n = random.randint(1,12)
```

1~12 사이의 난수 발생

```
for i in range(3):
```

```
    guess = int(input('Type an Integer(1~12):'))
```

```
    if guess == n:
```

```
        print('Your guess right!')
```

```
        break
```

```
    elif guess > n:
```

추정값이 틀렸을 경우

```
        print('Your guess is bigger')
```

```
    else:
```

```
        print('Your guess is smaller')
```

```
    if i == 2:
```

기회 3번이 넘을 경우

```
        print('Sorry! No more chance.')
```

[실행결과]

```
type a score: 100
```

```
type a score: 200
```

```
type a score: 50
```

```
type a score: -10
```

```
Total score is 350
```



7.4 break문과 continue문 비교

- continue문
 - loop 안에 존재하는 continue 문 다음의 문장 실행을 생략하고 다음 loop를 계속함

break문

```
for n in range(1,6):  
    if n==4 :  
        break  
    print(n, end=' ')  
print('The End')
```

break

반복문을 벗어남

[실행결과]

1 2 3 The End

continue문

```
for n in range(1,6):  
    if n==4 :  
        continue  
    print(n, end=' ')  
print('The End')
```

continue

반복문의 처음으로 이동함

[실행결과]

1 2 3 5 The End



7.4 따라 해보기 - 369게임

- 369 게임 프로그램
 - 1 ~ 20까지 정수를 출력함
 - 단 3의 배수인 경우 출력하지 않고 다음 숫자로 넘어감

[소스코드] 7-17.py

```
for n in range(1,21):
```

```
    if n%3==0 :  
        continue
```

3의 배수인 경우
다음 숫자로 이동

```
    print(n,end=' ')
```

3의 배수가 아닌 경우 n값 출력

[실행결과]

1 2 4 5 7 8 10 11 13 14 16 17 19 20



마무리 및 다음 차시 예고



이번 수업에서는 `break`문, `continue`문에 대해 알아보았습니다.