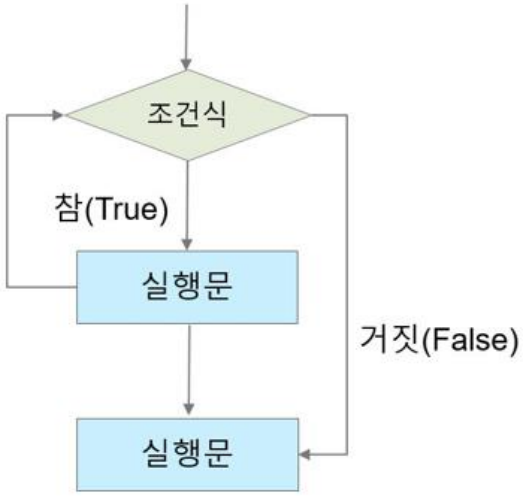
UNISTUDY

파이썬

9강. 제어문(반복문1)

1. 반복문

- 반복문이란 특정 부분을 반복해서 실행하는 명령문을 의미한다.
- 즉 조건식이 참인 동안 특정 실행문이 반복적으로 수행되는 명령문으로 파이썬에서는 while과 for 명령어로 제공된다.
- 한편 반복문은 조건부터 검사하기 때문에 처음부터 조건이 거짓이면 반복문을 한번도 수행하지 않는 경우도 있다.
- 반복문 수행 과정을 나타내는 순서도이다.



2. while문

- while은 조건식과 반복 실행문(loop)으로 블록을 구성한다.
- 반복문을 수행하기 위해 조건부터 먼저 검사하기 때문에 반복의 대상인 실행문이 한번도 수행하지 않는다.
- 이러한 while문의 블록 구조를 사전 시험 루프(pre-test loop)5)라고 부른다.

```
while 조건식 :
        실행문1
          :
        실행문n
```

2. while문

chapter03.lecture.step02_while.py

```
# (1) 카운터와 누적변수

cnt = tot = 0 # 변수 초기화

while cnt < 5 : # True : loop 수행
    cnt += 1 # 카운터 변수(cnt = cnt + 1)
    tot += cnt # 누적변수 : tot = tot + cnt
    print(cnt, tot)


# [실습] 1 ~ 100 사이 3의 배수 합과 원소 추출하기
cnt = tot = 0

dataset = []  # 빈 list


while cnt < 100:  # 100회 반복

    cnt += 1  # 카운터
    if cnt % 3 == 0:
        tot += cnt  # 누적변수
        dataset.append(cnt)  # cnt 추가

print('1 ~ 100 사이 3의 배수 합 = %d' % tot)
print('dataset =', dataset)
```

Python Console

```
1 1
2 3
3 6
4 10
5 15


1 ~ 100 사이 3의 배수 합 = 1683
dataset = [3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48, 51, 54, 57, 60, 63, 66, 69, 72, 75, 78, 81, 84, 87, 90, 93, 96, 99]
```

2. while문

- (1) 카운터와 누적변수
- 반복문에서 카운터 변수와 누적 변수는 자주 언급되는 용어이다.
- 반복을 수행하는 과정에서 카운터 변수(cnt)는 정해진 상수값에 의해서 일정하게 값이 증가하는 변수를 말하고, 누적 변수(tot)는 카운터 변수의 값을 차곡차곡 저장하는 변수를 의미한다.
- 예문에서 cnt는 0을 초기값으로 조건식에 사용되는 카운터 변수이다.
- 즉 cnt가 5보다 작으면 루프(loop)가 실행된다.
- cnt는 1씩 증가하고, tot은 cnt 값을 누적한다. 여기서 카운터 변수인 cnt는 무한정 1씩 증가하지 않는다.
- 왜냐하면 cnt 변수는 조건식으로 사용되기 때문에 cnt값이 5가 되는 순간 조건은 False가 되므로 loop를 탈출(exit)하게 된다.

2. while문

- [실습] 1 ~ 100 사이 3의 배수 합과 원소 추출하기
- 카운터 변수(cnt)에 의해서 while 반복문은 100회 반복 수행된다.
- 이 과정에서 cnt 변수를 3으로 나누어서 나머지 값이 0이면 현재 cnt가 3의 배수이므로 빈 list로 선언한 dataset에 cnt값을 추가(append)한다.
- 100회 반복이 모두 완료되면 3의 배수의 합을 출력하고, dataset 변수를 출력하 여 3의 배수에 대한 원소를 출력한다.

2. while문

chapter03.lecture.step02_while.py

```
# 무한 루프(loop)
numData = [] # 빈 list

while True :
    num = int(input("숫자 입력 : "))

    if num % 10 == 0 : # exit 조건식
        print("프로그램 종료")
        break
    else :
        print(num)
        numData.append(num) # list 추가
```

Python Console

```
숫자 입력 : >? 1
1
숫자 입력 : >? 3
3
숫자 입력 : >? 5
5
숫자 입력 : >? 10
프로그램 종료
```

2. while문

- 무한 루프(loop) 예
- 무한 루프(loop)는 반복문에서 사용되는 조건식이 True인 경우를 말한다. 조건식이 True이기 때문 반드시 루프 내에 탈출(exit) 조건을 지정해야한다.
- 만약 탈출 조건이 없는 경우에는 무한 루프에 빠져서 반복이 멈추지 않는다.
- 예문에서는 키보드로 임의의 숫자를 입력받아서 홀수이면 빈 list로 선언된 numData에 입력한 값이 원소로 추가되고, 루프가 반복된다.
- 한편 짝수가 입력되면 “프로그램 종료”라는 문자열이 출력되고, 루프에서 탈출된다.

3. random 모듈

- random 모듈은 컴퓨터에 의해서 임의 난수를 발생시키는 함수들을 제공한다. 활용분야는 모집단으로 부터 표본(sample)을 추출하거나 동전을 던져서 앞면이 나올 확률을 구하기 위해 컴퓨터로 시뮬레이션 하는 용도 등으로 사용된다.
- random 모듈에서 제공되는 주요 함수와 기능은 다음과 같다.

3. random 모듈

함수명	기능
choice(seq)	비어 있지 않은 sequence에서 임의의 요소를 선택한다.
choices(population, k)	모집단으로부터 k의 크기를 갖는 목록을 선택한다.
randint(a, b)	[a, b] 범위에서 난수 정수를 반환한다.
random(...)	0~1 사이의 난수 실수를 반환한다.
sample(population, k)	모집단(Population)으로 부터 k개를 임의 추출한다.
seed()	종자(seed)값을 지정하여 동일한 난수가 선택되도록 한다.
uniform(a, b)	[a, b] 범위의 난수 실수를 균등하게 반환한다.
normalvariate(mu, sigma)[6]	정규 분포(Normal distribution)를 따르는 난수를 반환한다.(mu는 평균이고 sigma는 표준 편차를 의미한다)

UNISTUDY

3. random 모듈

chapter03.lecture.step02_while.py

```
# (1) random module 추가
import random
help(random) # 모듈 도움말

# (2) random모듈의 함수 도움말
help(random.random)

# (3) 0~1 사이 난수 실수
r = random.random()
print('r=', r)  # r= 0.3940

# [실습] 난수 0.01 미만이면 종료 후 난수 개수 출력
cnt = 0
while True:
    r = random.random()
    print(random.random())
    if r < 0.01:
        break # loop exit
    else:
        cnt += 1

print('난수 개수 = ', cnt)
```

Python Console

```
random(...) method of rando
m.Random instance
random() -> x in the interva
l [0, 1).


r= 0.08630661167968179

0.14640918537283643
0.9369600355686324
0.059625469568146516
0.46133459132683863
0.668349609085757
0.9560769260825098
0.8169072288210812
0.4313008512665134
0.08940306809006371
0.32738193623091594

난수 개수 =  9
```

3. random 모듈

- (1) random module 추가
- random 관련 함수를 사용하기 위해서 import 명령으로 random 모듈(module)을 가져온다. 파이썬에 서 모듈은 다수의 함수와 클래스를 포함하는 *.py 파일 형식으로 제공된다. 모듈의 도움말은 help(모 듈명) 형식으로 참고할 수 있다.
- (2) random모듈 관련 함수 도움말
- random 모듈에서 제공되는 함수는 help(모듈명.함수명) 형식으로 도움말을 참고할 수 있다.
- (3) 0~1 사이 난수 실수
- random 모듈에서 제공되는 random() 함수를 이용하여 0과1 사이의 임의의 난수 실수를 반환한 결과이다.
- [실습] 난수 0.01 미만이면 종료 후 난수 개수 출력
- 난수가 0.01 미만이면 반복을 탈출(exit)하고, 그렇지 않으면 임의의 난수를 반복적으로 발생시켜서 출력하고 카운트한 결과를 출력하는 예문이다.

3. random 모듈

chapter03.lecture.step02_while.py

```
# (1) random모듈 관련 함수 도움말
help(random.randint)
help(random.choices) # 모집단에서 k 크기 목록 반환

# (2) 이름 list에 전체 이름, 특정 이름 출력
names = ['홍길동', '이순신', '유관순']
print(names) # 전체 이름
print(names[2]) # , 특정 이름 출력

# (3) list에서 자료 유무 확인하기
if '유관순' in names:
    print('유관순 있음')
else:
    print('유관순 없음')

# (4) 난수 정수로 이름 선택하기
idx = random.randint(0, 2)
print(names[idx])
```

Python Console

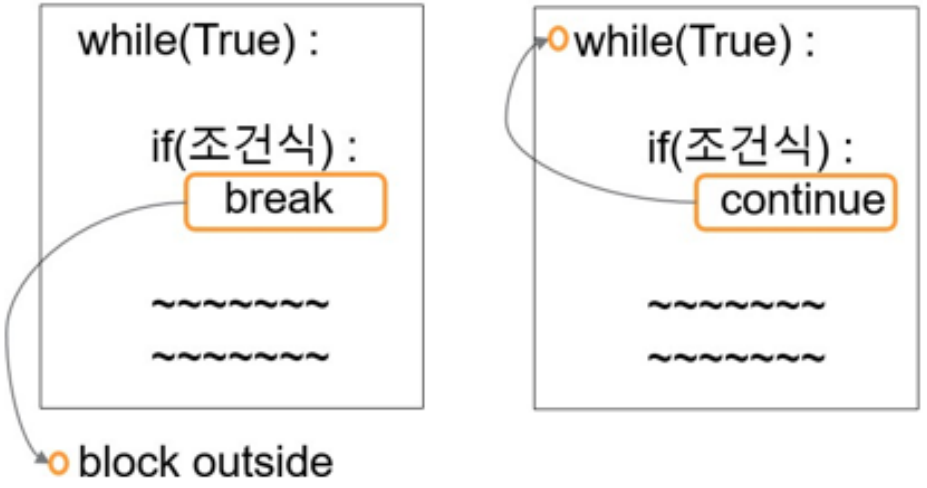
```
randint(a, b) method of random.Random instance
    Return random integer in range [a, b], including both end points.

Return a k sized list of population elements chosen with replacement.

['홍길동', '이순신', '유관순']
유관순

유관순 있음


random.randint
홍길동
```

3. random 모듈

- random 관련 함수2 예
- (1) random모듈 관련 함수 도움말
- randint와 choices 함수의 도움말을 참고한다.
- (2) 이름 list에 전체 이름, 특정 이름 출력
- 여러 명의 이름을 갖는 names는 list형 자료구조를 갖는 변수이다. list 자료구조는 여러 개의 원소를 1차원의 배열 형식으로 저장한다. names 변수를 출력하면 전체 원소가 출력되고, names[2]와 같이 색인(index)을 이용하면 3번째 원소의 이름이 출력된다. list 자료구조에 대해서는 제4장 자료구조 편에서 자세히 알아본다.
- (3) list에서 자료 유무 확인하기
- 에서도 in을 사용할 수 있다. 특히 여러 개의 원소를 갖는 list형 변수에서 해당 원소의 유무를 판단할 경우 이용된다. 현재 names에는 '유관순' 원소를 포함하고 있기 때문에 조건식의 결과
- (4) 난수 정수로 이름 선택하기
- randint(0, 2) 함수에 의해서 0~2사이의 임의의 난수 정수가 반환된다. 이렇게 반환된 값을 색인으 로 하여 names에서 해당 이름을 추출하는 예문이다.

4. break, continue

- break와 continue는 반복문에서 이용되는 명령어 이다.
- break 명령어는 반복을 탈출(exit)하는 역할을 하고, continue는 다음에 오는 실행문을 실행하지 않고(skip) 반복을 지속하는 역할을 한다.
- while문에서 break와 continue 명령어의 수행과정을 나타내고 있다.



4. break, continue

chapter03.lecture.step02_while.py

```
i = 0
while i < 10:
    i += 1  # 카운터
    if i == 3:
        continue  # 다음 문장 skip
    if i == 6:  # 탈출 조건
        break  # exit
    print(i, end=' ')
```

Python Console

```
1 2 4 5
```

4. break, continue

- break, continue 예
- 카운트 변수 i를 이용하여 조건식을 갖는 while 반복문은 정상적으로 10회 반복된다. 하지만 출력 결과를 보면 5까지 출력되고, 6이 될 때 반복이 종료되는 것을 볼 수 있다. 이유는 'i == 6' 조건식에 의해서 break 명령어가 실행되기 때문이다.
- 한편 3이 출력되지 않은 이유는 'i==3'의 조건식에 의해서 continue 명령어가 실행되기 때문에 continue 다음에 오는 모든 실행문이 되지 않으므로 3은 출력되지 않고 반복이 지속된다.

THANK YOU