

[Python]



Python으로 배우는 소프트웨어 원리

Chapter 06. 반복

목차

1. 반복 구조
2. 반복문의 종류
3. 반복문의 활용

01

반복 구조

01. 반복 구조

[일상 생활 속 반복]

- 알람 소리에 잠에서 깨어 학교나 직장へ가기
- 자전거를 타거나 운동을 할 때 동작 반복하기

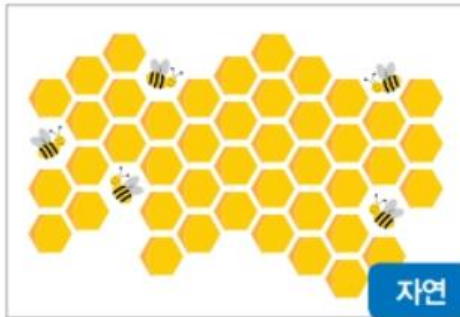


그림 6-1 일상에서의 패턴

01. 반복 구조

I. 일상에서 볼 수 있는 반복

실습 6-1

반복적인 규칙 찾기

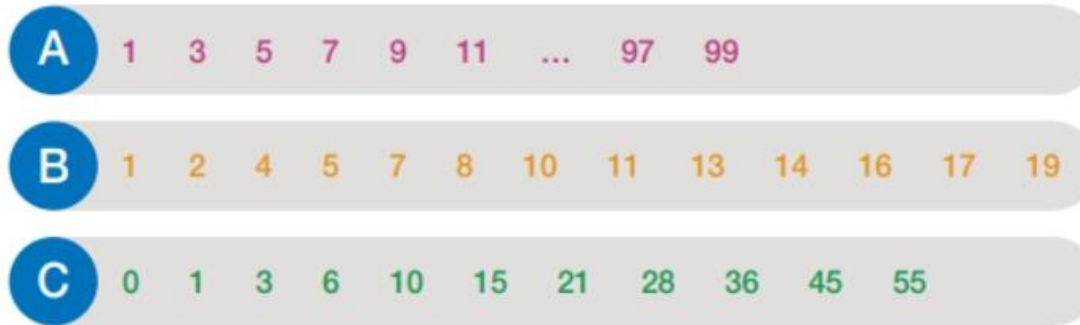


그림 6-2 숫자 속에서의 패턴

❖ 반복 패턴 요소

- 시작 값
- 끝나는 값
- 반복 패턴(증감)

```
for i in range(start, stop, step):  
    print(i)  
for i in range(1, 5, 1):  
    print(i)
```

1
2
3
4

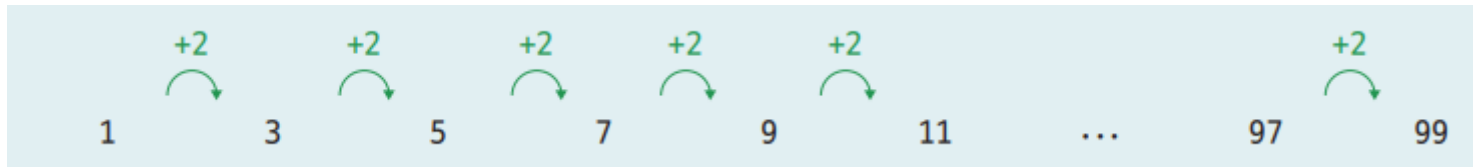
01. 반복 구조

I. 일상에서 볼 수 있는 반복

실습 6-1

반복적인 규칙 찾기

- ① A는 100보다 작은 자연수 중에서 홀수만 출력 for i in range(1, 100, 2):



- ② B는 20보다 작은 자연수 중에서 3의 배수를 빼고 출력



- ③ C는 0에서 10까지 정수를 차례대로 더한 값을 출력



❖ 반복 패턴 요소

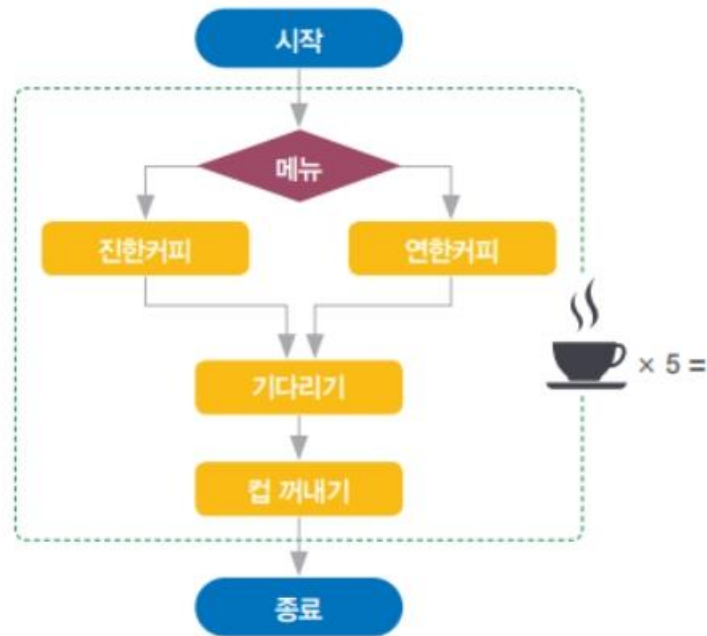
- 시작 값
- 끝나는 값
- 반복 패턴(증감)

01. 반복 구조

II. 반복 구조의 필요성



그림 6-3 커피머신의 반복 동작

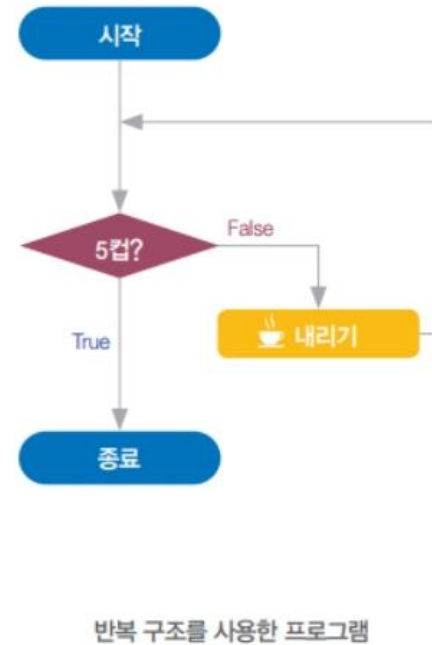


01. 반복 구조

II. 반복 구조의 필요성



그림 6-4 순차 구조와 반복 구조 비교



- 반복되는 구조(위 예에서 내리기)를 여러 번 **재사용** 가능
- 반복되는 횟수를 상태(변수의)에 따라 결정 가능 → 동일 코드로 다른 결과 달성

01. 반복 구조

II. 반복 구조의 필요 요소

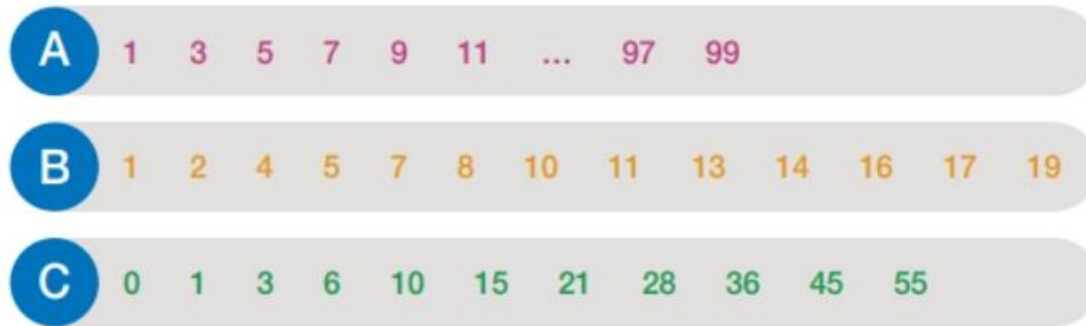


그림 6-2 숫자 속에서의 패턴

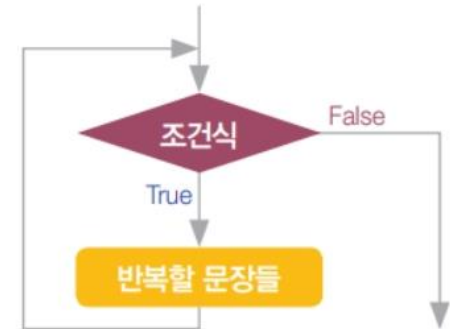


그림 6-6 while 문 구조

- (반복 변수) **시작** 값 :
- 반복 조건(**끝**나는 값까지) :
- (반복 변수 패턴) 증감 :

```
i = 1                #시작 값(반복 변수)
while i < 5:          #반복 조건
    print(i, end=' ') #반복 처리
    i += 1            #값 증감(반복 변수)
```

```
for i in range(1, 5, 1): #반복 조건 처리
    print(i, end=' ')    #반복 처리
```

1
2
3
4

02

반복문의 종류

02. 반복문의 종류

표 6-1 반복문의 종류

	조건 반복	횟수 반복	무한 반복
반복 기준	조건 충족	반복 횟수	무한 실행
사용 예	지칠 때까지 반복하라	10번 반복하라	계속하라
반복문	while	for	while True

I. While 문

- 반복문 시작 전에 **반복 조건식의 진위(true/false)** 결과에 따라 반복 여부를 결정
- 반복문 안에 반복 조건식에 영향을 미치는 변수 값의 변화가 있어야 반복 탈출 가능

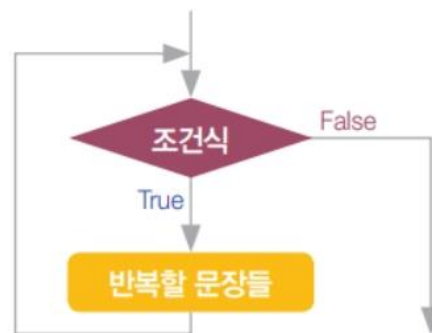


그림 6-6 while 문 구조

while 조건식 :

□ 반복할 문장들

들여쓰기

- (반복 변수) 시작 값 : 반복문 밖(앞)에 필요
- 반복 조건** : while 조건식
- (반복 변수) 증감 : 반복문 안에 필요

02. 반복문의 종류

I. While 문

- 컴퓨터가 랜덤으로 정해 놓은 1부터 20 사이의 정수를 맞추는 게임
- 사용자가 입력한 수가 정해 놓은 값과 같으면 정답이라고 알려주고 프로그램을 종료
- 두 수가 다른 경우에는 숫자를 다시 입력받아 비교해서 더 큰지, 작은지 화살표(↑, ↓)로 표시

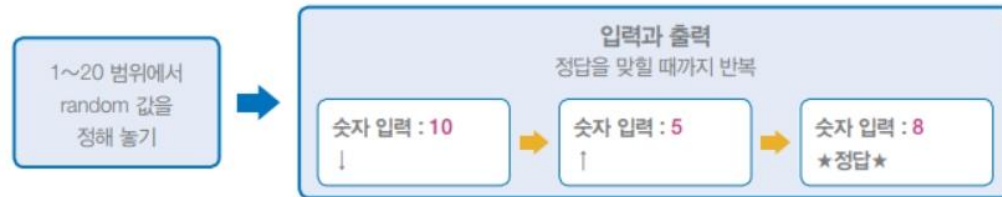


그림 6-7 랜덤 게임 규칙과 실행 순서

- ① random 모듈을 불러오고 random.randint()를 사용해 정수 값 하나를 추출하고 변수 answer에 저장, randint()의 두 인수(1, 20)가 추출할 범위

실습 6-3

랜덤 숫자 맞추기 게임 만들기

code06-03.py

```
01 import random                # random 모듈 불러오기
02 answer = random.randint(1, 20) # 1에서 20 사이의 랜덤 정수 저장
```

02. 반복문의 종류

I. While 문

실습 6-3

랜덤 숫자 맞추기 게임 만들기

code06-03.py

- ② 입력받은 값을 저장하기 위한 변수를 만들고, 정답이 나올 때까지 '입력 → 값의 비교 → 출력' 동작을 반복

```
01 import random                # random 모듈 불러오기
02 answer = random.randint(1, 20) # 1에서 20 사이의 랜덤 정수 저장
03 number = 0                    # 입력하는 값을 저장할 변수
04
05 while number != answer:
06     number = int(input("숫자 입력(1~20) : "))
07
08     if number > answer:        # 입력한 값이 정답보다 큰 경우
09         print("↓")
10     elif number < answer:      # 입력한 값이 정답보다 작은 경우
11         print("↑")
12     else:
13         print("★정답★")
```

- (반복 변수) 시작 값 :
- **반복 조건** : while ~ :
- (반복 변수) 증감 :

02. 반복문의 종류

I. While 문

실습 6-3

랜덤 숫자 맞추기 게임 만들기

code06-03.py

- ③ 작성한 프로그램을 파일로 저장하고 실행 결과를 확인

숫자 입력(1~20) : 10

↑

숫자 입력(1~20) : 13

↑

숫자 입력(1~20) : 16

★정답★

숫자 입력(1~20) : 6

↑

숫자 입력(1~20) : 13

↓

숫자 입력(1~20) : 11

★정답★

02. 반복문의 종류

[실습] 값을 1부터 1씩 증가시켜가면서 값이 10이 될 때까지 출력

◆ 증가 값을 1씩 증가시켜가면서 증가 값이 10이 될 때까지 출력 하기

- 반복 시작 값 : 1
- 증감 값 : +1
- 반복 조건: 현재 값 ≤ 10
- 출력 값: 현재 값



1. (반복 변수) 시작 값: $x = 1$
2. 반복 조건: $x \leq 10$
3. 반복 내용: `print(x)`
4. (반복 변수) 증감: $x = x + 1$

```
x = 1           #시작 값
while x <= 10:  #반복 조건
    print(x, end=' ')
    x = x + 1    #값 증가
```

❖ 검증 : 반복 시작 점과 끝나는 점에서 경계점 시험

02. 반복문의 종류

[실습] 값을 1부터 1씩 증가시켜가면서 값이 10이 될 때까지 출력

◆ 증가 값을 1씩 증가시켜가면서 증가 값이 10이 될 때까지 출력 하기

- 반복 시작 값 : 1
- 증감 값 : +1
- 반복 조건: 현재 값 \leq 10
- 출력 값: 현재 값

1. (반복 변수) 시작 값: $x = 1$
2. 반복 조건: $x \leq 10$
3. 반복 내용: `print(x)`
4. (반복 변수) 증감: $x = x + 1$



```
x = 1           #시작 값(반복 변수)
while x <= 10:  #반복 조건
    print(x, end=' ')
    x += 1      #값 증감(반복 변수)
```

```
x = 0           #시작 값(반복 변수)
while x <= 10:  #반복 조건
    x += 1      #값 증감(반복 변수)
    print(x, end=' ')
```

```
06장#Ch06-00.py
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10 11
>>>
```

Ln: 30 Col: 0

02. 반복문의 종류

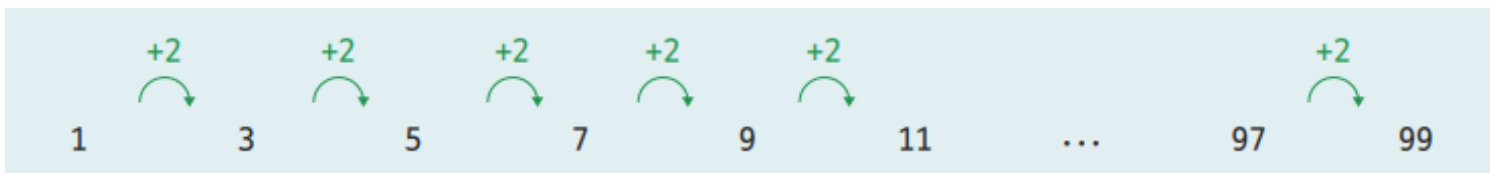
I. While 문

- 반복문 시작 전에 반복 조건식의 진위($x < 100$) 결과에 따라 반복 여부를 결정
- 반복문 안에는 반드시 반복 조건식의 결과 값에 영향을 미치는 변수의 값 변화가 있어야 한다. (05 $x = x + 2$ 에서 x 값이 변하고 있음)

실습 6-2

while 문을 이용하여 규칙을 가진 숫자 출력하기

code06-02.py



①

```
01 x = 1                # 변수 x의 초기화
02
03 while x < 100 :      # x의 값이 100보다 작으면 반복하
04     print(x, end = " ") # x의 값을 출력하고 줄 바꿈 대신
05     x = x + 2         # x를 2 증가시킴
```

1. (반복 변수) 시작 값: $x = 1$
2. 반복 조건: $x < 100$
3. 반복 내용: `print()`
4. (반복 변수) 증감: $x = x + 2$

②

```
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67
69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
```

02. 반복문의 종류

[실습] 숫자 UpDown 맞추기

❖ Ch06-UpDown00.py

실습 6-3

랜덤 숫자 맞추기 게임 만들기

code06-03.py

◆ [실습 6-3] 랜덤 숫자 맞추기 코드 업그레이드 하기

- 매 반복 횟수마다 진행 횟수(반복 횟수)를 출력
- 5회 내에 맞추면 "당신이 이겼습니다."를 그 외는 "당신이 졌습니다."를 출력

```
>>> = RESTART: D:\Documents\강  
의록\Python\컴퓨터사고와파  
이썬-김지연\source\06장\cod  
e06-031.py  
숫자 입력(1~20) : 10  
[01] ↑  
숫자 입력(1~20) : 15  
[02] ↓  
숫자 입력(1~20) : 13  
[03] ↑  
숫자 입력(1~20) : 14  
[04] ★정답★  
당신이 이겼습니다.  
>>>
```

Ln: 76 Col: 0

02. 반복문의 종류

II. 무한 반복과 반복의 제어

■ 무한 반복문

- 반복할 문장들이 계속 실행
- 무한 반복을 끝내려면 키보드에서 Ctrl + C 를 눌러 프로그램을 강제로 종료

```
while True:
```

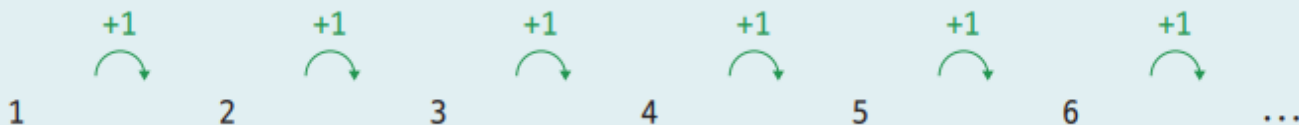
반복할 문장들

조건식을 항상 참으로 기술하는 방법

실습 6-4

무한 반복하는 프로그램 만들기

code06-04.py



```
① 01 x = 0
    02
    03 while True :
    04     x += 1           # x의 값을 하나 증가
    05     print(x, end = ' ') # 한 줄로 나란히 출력
```

02. 반복문의 종류

II. 무한 반복과 반복의 제어

- 무한 반복문

실습 6-4

무한 반복하는 프로그램 만들기

code06-04.py

② 프로그램을 실행하면 1부터 하나씩 증가한 값이 출력

➤ 종료를 위해 단축키 Ctrl + C 입력

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67
68 69 70 71 72 73 74 75 76 77 78 79 80 81
Traceback (most recent call last):
  File "F:\code06-04.py", line 5, in <module>
    print(x, end = ' ')      # 한 줄로 나란히 출력
KeyboardInterrupt
```

02. 반복문의 종류

II. 무한 반복과 반복의 제어

▪ 반복의 제어

- 반복문 안에서 break와 continue를 사용하여 반복문을 끝내거나 반복할 문장을 건너뛰고 조건식 검사 위치로 이동하도록 프로그램 흐름을 제어
- break와 continue는 일반적으로 if 문과 함께 사용하여, 특정 조건이 만족할 때 반복문이 종료되거나 실행 순서의 변경 발생

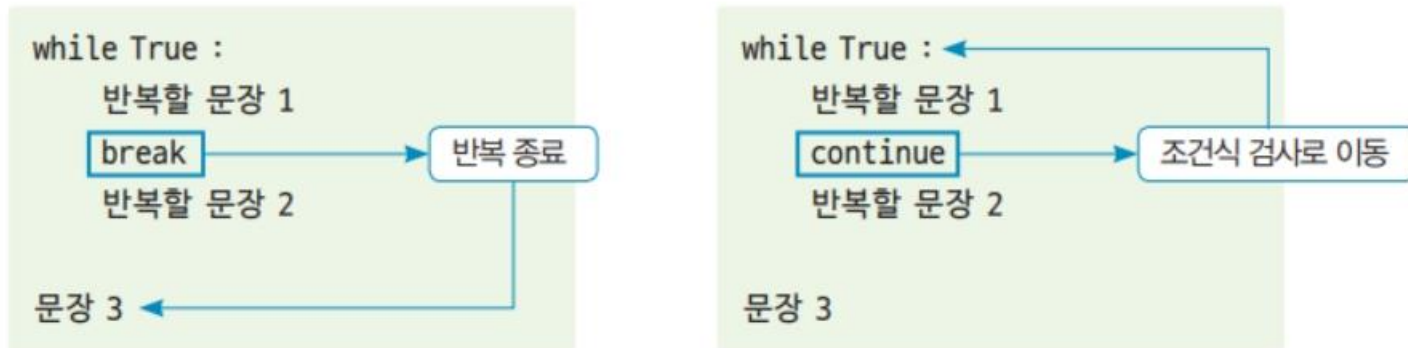


그림 6-8 break와 continue를 사용한 반복문의 제어 흐름

02. 반복문의 종류

[실습] 값을 1부터 1씩 증가시켜가면서 값이 10이 될 때까지 출력

◆ 증가 값을 1씩 증가시켜가면서 증가 값이 10이 될 때까지 출력 하기

- 반복 시작 값 : 1
- 증감 값 : +1
- 반복 조건: 현재 값 \leq 10
- 출력 값: 현재 값



```
x = 1           #시작 값
while x <= 10:  #반복 조건
    print(x, end=' ')
    x = x + 1    #값 증가
```

```
x = 1           #시작 값
while True:     #반복 조건
    if x > 10:   #탈출 조건
        break
    print(x, end=' ')
    x = x + 1    #값 증가
```

```
x = 1           #시작 값
while True:     #반복 조건
    print(x, end=' ')
    x = x + 1    #값 증가
    if x > 10:   #탈출 조건
        break
```

```
IDLE Shell 3.10.1
File Edit Shell Debug Options Window Help
06장\Ch06-00.py
1 2 3 4 5 6 7 8 9 10
```

02. 반복문의 종류

II. 무한 반복과 반복의 제어

■ 반복의 제어

실습 6-5

break와 continue를 사용하여 규칙을 가진 숫자 출력하기

code06-05.py

- 20보다 작은 자연수 중에서 3의 배수만 제외하고 출력
- 반복문의 흐름을 제어하는 break와 continue를 사용하여 출력 패턴에 알맞은 프로그램을 작성

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

```
01 x = 0
02
03 while True:
04     x = x + 1
05     if x >= 20:
06         break
07     if x % 3 == 0:
08         continue
09     print(x, end = ' ')
```

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

❖ break이나 continue를 없앤 코드로 변경하여보시오.

02. 반복문의 종류

[실습] 1에서 20까지 중에 3의 배수만 출력하지 않기

❖ Ch06-05.py

- ◆ [실습 6-5] "1에서 20까지 중에 3의 배수만 출력하지 않기" 코드를 아래 요구대로 수정함.
 - break문과 continue문을 사용하지 마시오.

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

```
01 x = 0
02
03 while True:
04     x = x + 1
05     if x >= 20:
06         break
07     if x % 3 == 0:
08         continue
09     print(x, end = ' ')
```


02. 반복문의 종류

II. 무한 반복과 반복의 제어

▪ 반복의 제어

실습 6-6

영화관 무인단말기로 메뉴 주문 후 종료하기

code06-06.py

- 메뉴를 선택하고 수량을 입력하면 계산한 금액 합계를 출력 하는 프로그램
- 주문을 끝내려면 메뉴를 '0'으로 입력
- 메뉴를 잘못 선택하면 프로그램을 종료하지 말고 다시 입력

```
① 01 menu = 0           # 선택 메뉴
    02 number = 0       # 주문 수량
    03 price = 0        # 상품 단가
    04 total = 0        # 금액 합계
    05
    06 print("*" * 30)   # 메뉴 출력
    07 print("[1]팝콘 [2]나초 [3]핫도그 [4]음료")
    08 print("주문을 끝내려면 [0]을 입력하세요.")
    09 print("-" * 30)
    10
```

02. 반복문의 종류

II. 무한 반복과 반복의 제어

■ 반복의 제어

실습 6-6

영화관 무인단말기로 메뉴 주문 후 종료하기

code06-06.py

```
① 11 while True :  
12     menu = int(input("선택 메뉴 : "))  
13     if menu == 0 :  
14         break  
15     if menu < 1 or menu > 4 :  
16         print("메뉴 선택 오류...다시 선택하세요.\n")  
17         continue  
18  
19     if menu == 1 : price = 5000          # 조건식과 명령문을 한 줄에 적기  
20     elif menu == 2 : price = 4000  
21     elif menu == 3 : price = 3500  
22     else : price = 2000  
23  
24     number = int(input("주문 수량 : "))  
25     total = total + (number * price)  
26     print()          # 다른 메뉴를 선택하기 전에 빈 줄 추가
```

02. 반복문의 종류

II. 무한 반복과 반복의 제어

■ 반복의 제어

실습 6-6

영화관 무인단말기로 메뉴 주문 후 종료하기

code06-06.py

```
27  
① 28 print("-" * 30)  
29 print("금액 합계는", total, "원입니다.")
```

```
② *****  
[1]팝콘 [2]나초 [3]핫도그 [4]음료  
주문을 끝내려면 [0]을 입력하세요.
```

```
-----  
선택 메뉴 : 1  
개수 입력 : 1
```

```
선택 메뉴 : 4  
개수 입력 : 2
```

```
선택 메뉴 : 0  
-----
```

```
금액 합계는 9000 원입니다.
```

```
*****  
[1]팝콘 [2]나초 [3]핫도그 [4]음료  
주문을 끝내려면 [0]을 입력하세요.
```

```
-----  
선택 메뉴 : 5  
메뉴 선택 오류...다시 선택하세요.
```

```
선택 메뉴 : 3  
개수 입력 : 1
```

```
선택 메뉴 : 0  
-----
```

```
금액 합계는 3500 원입니다.
```

02. 반복문의 종류

[과제-1] 숫자 UpDown 맞추기 게임 Upgrade

❖ Ch06-UpDown00.py

◆ [실습 6-3] 랜덤 숫자 맞추기 코드 업그레이드 하기 (**while** 반복문 사용)

- 매 반복 횟수마다 진행 횟수(반복 횟수)를 출력
- 5회 내에 맞추면 "당신이 이겼습니다."를 그 외는 "당신이 졌습니다."를 출력
- **게임을 마치고, 다음 게임을 더 할지 선택**

```
숫자 입력(1~20) : 2
[05] ↓
숫자 입력(1~20) : 1
[06] ★정답★
당신이 졌습니다.
>다시 할까요(y/n)? Y
숫자 입력(1~20) : 5
[01] ↑
숫자 입력(1~20) : 14
[02] ↓
숫자 입력(1~20) : 12
[03] ↓
숫자 입력(1~20) : 10
[04] ↑
숫자 입력(1~20) : 11
[05] ★정답★
당신이 이겼습니다.
>다시 할까요(y/n)?
>>> |
```

Ln: 146 Col: 0

02. 반복문의 종류

III. for 문

- for 문은 반복 조건을 더 다양하게 기술
 - for 반복은 **반복 조건 변수와 반복 방법(시작, 종료, 증감)**을 더 구체적으로 명시
 - while 반복은 반복 조건식만 명시 → 시작과 증감은 별도로 명시

while 조건식 :
 반복할 문장들
 들어쓰기

표 6-1 반복문의 종류

	조건 반복	횟수 반복	무한 반복
반복 기준	조건 충족	반복 횟수	무한 실행
사용 예	지칠 때까지 반복하라	10번 반복하라	계속하라
반복문	while	for	while True

for 변수명 in range(반복 횟수) :
 반복할 문장들
 들어쓰기

for 변수명 in range(**1**, **10**, **2**) :
 반복할 문장들
 들어쓰기

시작값
 증감값
 종료값

02. 반복문의 종류

III. for 문

- for 문은 반복 조건을 더 다양하게 기술
 - for 반복은 **반복 조건 변수**와 **반복 방법(시작, 종료, 증감)**을 더 구체적으로 명시
- 반복 방법은 **range()** 함수와 함께 많이 사용
- range() 함수는 정해진 구간의 정수들(수열)을 생성하는 함수
- range(5)인 경우 5번 반복, 0~4까지 5개의 값을 만들어 사용할 수 있다.
- for i in range(5)인 경우 0~4까지 5개의 반복 값이 i 변수에 할당되면서 반복 실행
- i를 사용하지 않으려면 i 대신 _(언더바) 사용

for 변수명 in range(반복 횟수) :

□ 반복할 문장들

들어쓰기

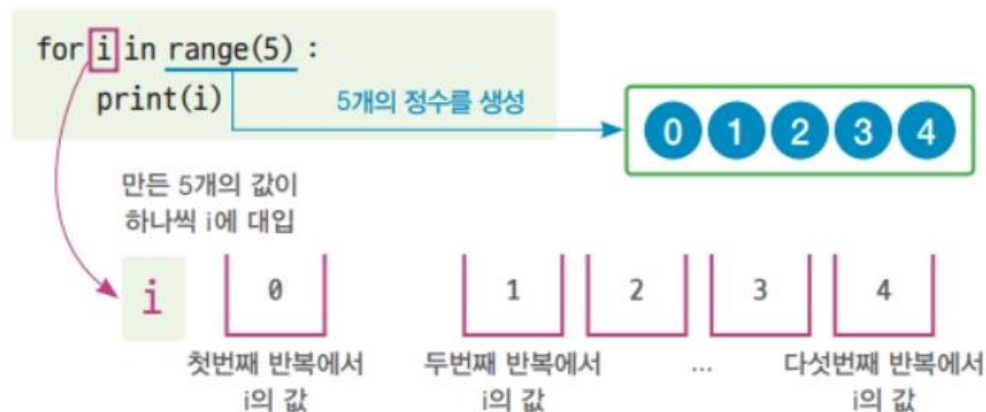
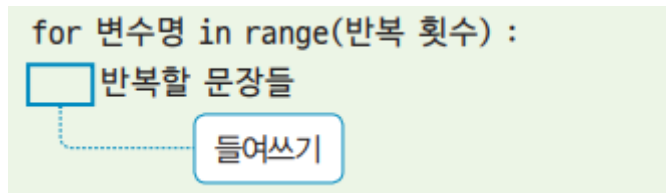


그림 6-10 for 반복문에서 range() 함수의 동작

02. 반복문의 종류

III. for 문

- range() 함수는 정해진 구간의 정수들(수열)을 생성하는 함수
- range(5)인 경우 5번 반복, 0~4까지 5개의 값을 만들어 사용할 수 있다.
- for i in range(5)인 경우 0~4까지 5개의 값이 변수에 할당되면서 반복 실행



```
code06-070.py - D:\Documents\강...
File Edit Format Run Options Window Help

cnt = 0      # 반복횟수를 저장하기 위한 변수

for i in range(5): # 0부터 5번 반복
    cnt += 1
    print("[%02d] i=%d" %(cnt, i))

Ln: 1 Col: 0
```

```
IDLE Shell 3.10.1
File Edit Shell Debug Options Window Help

[01] i=0
[02] i=1
[03] i=2
[04] i=3
[05] i=4
>>> range(5)
range(0, 5)
>>>
```

02. 반복문의 종류

III. for 문

여기서 잠깐

range() 함수

- range() 함수는 다음 그림과 같이 3개의 인수를 가지고, start와 step은 생략할 수 있음

```
range([start,] stop [,step])
```

- start부터 시작해서 step 간격으로 증감하는 수열을 만들다가 stop이 되면 중지
- start를 생략하면 기본값(default)은 0이고, step을 생략하는 경우 기본값은 1
- stop은 반복 전에 탈출 하는 값으로 반드시 필요

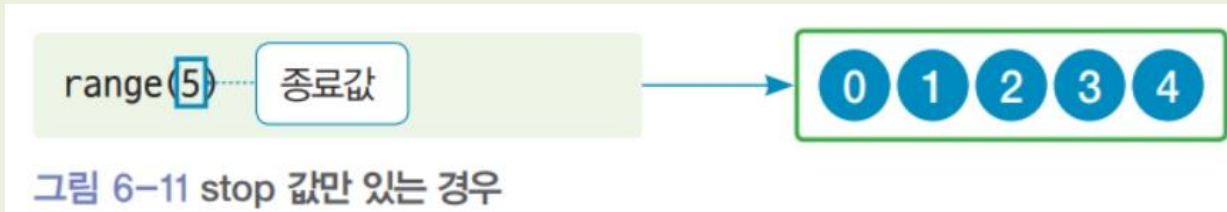
```
s = 0
for x in range(0, 10+1, 1) :
    s = s + x
    print(s, end = " ")
```


02. 반복문의 종류

III. for 문

여기서 잠깐 range() 함수

- stop 값만 있는 경우 : range(5)는 start와 step이 모두 생략된 것으로, 0부터 4까지 하나씩 증가하는 정수들을 생성 → start의 default는 0, step의 default는 1



- start, stop 값만 있는 경우 : range(1, 5)처럼 2개의 인수만 있는 경우, start는 1, stop은 5



02. 반복문의 종류

III. for 문

여기서 잠깐

range() 함수

- start, stop, step 값이 모두 있는 경우 : range(1, 10, 2)는 1부터 9까지 2개씩 증가하는 정수; 10 이상이 되면 반복 종료



그림 6-13 start, step, stop 값이 있는 경우

- 감소하는 수열을 만드는 경우 : 9부터 시작해서 2씩 감소하는 수열을 3까지 만들 수 있음; 1 이하가 되면 반복 종료



그림 6-14 감소하는 수열의 경우

02. 반복문의 종류

[실습] for 반복문 반복 횟수

◆ 아래 반복문의 i 값을 예측해보자.

- for i in range(5) :
- for i in range(0, 5, 1) :
- for i in range(1, 5+1, 1) :

```
cnt = 0      # 반복횟수를 저장하기 위한 변수

for i in range(5): # 0부터 5번 반복
    cnt += 1
    print("[%02d] i=%d" %(cnt, i), end = " ")

print("")
cnt = 0
for i in range(0, 5, 1): # 0부터 5번 반복
    cnt += 1
    print("[%02d] i=%d" %(cnt, i), end = " ")

print("")
cnt = 0
for i in range(1, 5+1, 1): # 1부터 5번 반복
    cnt += 1
    print("[%02d] i=%d" %(cnt, i), end = " ")
```

Ln: 1 Col: 0

02. 반복문의 종류

III. for 문

실습 6-7

for 문과 range() 함수로 규칙을 가진 숫자 출력하기

code06-07.py

- 0부터 10까지 정수를 차례대로 더한 값을 출력하는 프로그램을 for 문을 사용



```
① 01 s = 0                # 합계를 저장하기 위한 변수
    02
    03 for x in range(0, 11): # 0부터 10까지 반복
    04     s += x            # s = s + x와 동일, s가 x만큼 증가
    05     print(s, end = " ")
```

```
② 0 1 3 6 10 15 21 28 36 45 55
```

- ✓ **TIP** 횟수 반복에는 for 문의 문장 구조가 더 간단하다는 장점이 있다. 반면 while 문은 조건 반복이나 무한 반복에 더 편리하게 사용할 수 있다.

02. 반복문의 종류

III. for 문

여기서 잠깐

while 문과 for 문 비교

- **while** 문은 조건식 결과가 참(true)일 동안 반복하는 단순한 **조건 반복** 구조
- **for** 문은 (시작값, 종료값, 증감값)과 함께 변수를 사용하는 **횟수 반복** 구조
- while 문은 조건반복구조이나 for 문은 종료값에 도달하기 전까지 반복

```
01 x = 0
02 s = 0
03
04 while x <= 10 :
05     s = s + x
06     print(s, end = " ")
07     x = x + 1
```

```
s = 0

for x in range(0, 10+1, 1) :
    s = s + x
    print(s, end = " ")
```

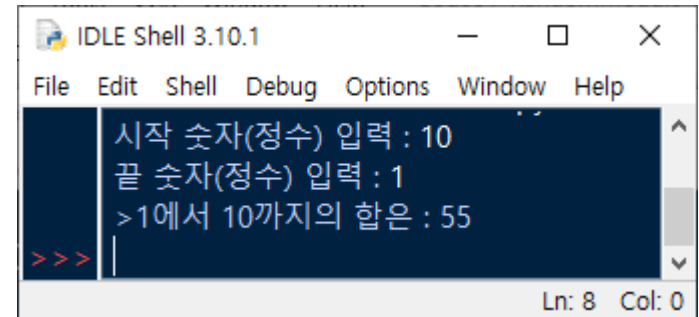
02. 반복문의 종류

[실습] 두 정수 값 범위의 모든 정수 값 더하기 (1)

◆ 입력 받은 두 정수 값의 범위에 있는 모든 값을 더한 결과를 출력하시오.

A. **for** 문을 사용하여 작성하시오.

❖ 시작 수와 종료 수가 뒤바뀌어 입력되어도 동일한 결과가 나오도록 하시오.



```
IDLE Shell 3.10.1
File Edit Shell Debug Options Window Help
시작 숫자(정수) 입력 : 10
끝 숫자(정수) 입력 : 1
>1에서 10까지의 합은 : 55
>>> |
```

Ln: 8 Col: 0

02. 반복문의 종류

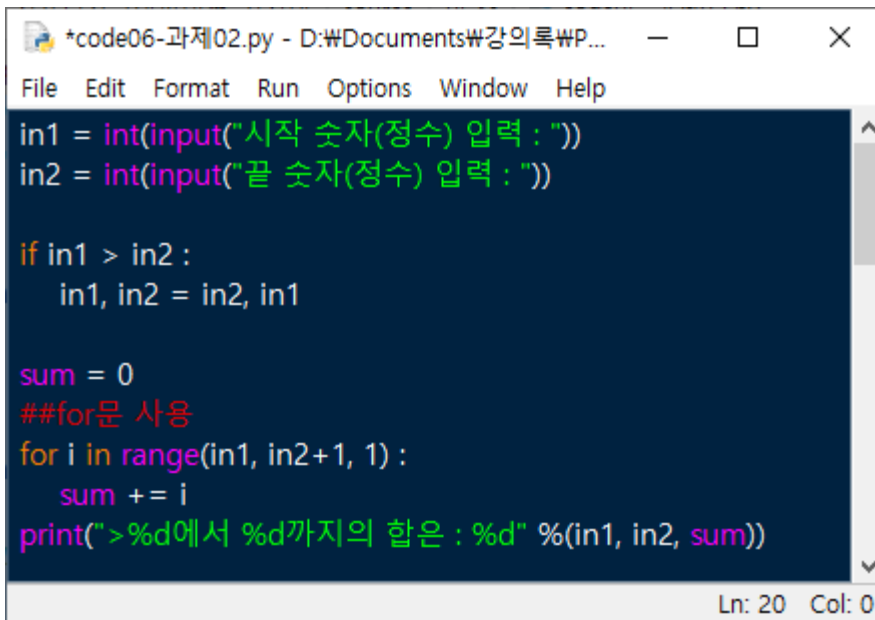
[실습] 두 정수 값 범위의 모든 정수 값 더하기 (2)

❖ Ch06-Sum.py

◆ 입력 받은 두 정수 값의 범위에 있는 모든 값을 더한 결과를 출력하시오.

B. **while** 문을 사용하여 작성하시오.

❖ 시작 수와 종료 수가 뒤바뀌어 입력되어도 동일한 결과가 나오도록 하시오.



```
*code06-과제02.py - D:\Documents\강의록\WP...
File Edit Format Run Options Window Help
in1 = int(input("시작 숫자(정수) 입력 : "))
in2 = int(input("끝 숫자(정수) 입력 : "))

if in1 > in2 :
    in1, in2 = in2, in1

sum = 0
##for문 사용
for i in range(in1, in2+1, 1) :
    sum += i
print(">%d에서 %d까지의 합은 : %d" %(in1, in2, sum))

Ln: 20 Col: 0
```

02. 반복문의 종류

[실습] 두 정수 값 범위의 모든 정수 값 더하기 (2)

❖ Ch06-Sum.py

◆ 입력 받은 두 정수 값의 범위에 있는 모든 값을 더한 결과를 출력하시오.

B. **while** 문을 사용하여 작성하시오.

❖ 시작 수와 종료 수가 뒤바뀌어 입력되어도 동일한 결과가 나오도록 하시오.

```
*code06-과제02.py - D:\Documents\강의록\WP...
File Edit Format Run Options Window Help
in1 = int(input("시작 숫자(정수) 입력 : "))
in2 = int(input("끝 숫자(정수) 입력 : "))

if in1 > in2 :
    in1, in2 = in2, in1

sum = 0
##for문 사용
for i in range(in1, in2+1, 1) :
    sum += i
print(">%d에서 %d까지의 합은 : %d" %(in1, in2, sum))

Ln: 20
```

```
*code06-과제02.py - D:\Documents\강의록\WP...
File Edit Format Run Options Window Help
in1 = int(input("시작 숫자(정수) 입력 : "))
in2 = int(input("끝 숫자(정수) 입력 : "))

if in1 > in2 :
    in1, in2 = in2, in1

sum = 0
##while문 사용
i = in1
while i <= in2 :
    sum += i
    i += 1
print(">%d에서 %d까지의 합은 : %d" %(in1, in2, sum))

Ln: 15 Col: 0
```


02. 반복문의 종류

[문제]

1. 아래 코드의 실행 결과를 _____에 적으시오.

```
for i in range(1, 10, 1):  
    cur_n = i  
    print(cur_n)           # _____  
  
for i in range(10):  
    cur_n = i  
    print(cur_n)           # _____  
  
for i in range(1, 10, 3):  
    cur_n = i  
    print(cur_n)           # _____  
  
for i in range(10, 0, -1):  
    cur_n = i  
    print(cur_n)           # _____
```

02. 반복문의 종류

[문제]

2. 아래 코드의 실행 결과를 _____에 적으시오.

```
sum = 0
seq = 0

while True:
    seq += 1
    if (seq % 2) == 1:
        continue
    else:
        sum += seq
    if sum > 10:
        break

print(seq)          # _____
print(sum)          # _____
```

02. 반복문의 종류

[문제]

3. 아래 코드의 실행 결과를 _____에 적으시오.

```
cnt = 0
for i in range(10):
    for j in range(1, 10):
        cnt = cnt + 1
print(cnt)                                # _____
```

4. 아래 코드의 실행 결과를 적으시오.

```
cnt = 0
for i in range(1, 5):
    for j in range(i, 5):
        cnt += 1
print(cnt)                                # _____
```

03. 반복문의 활용

[과제-2] 토끼와 거북이 경주

- ◆ 토끼와 거북이가 달리기 시합을 한다. 토끼는 1분에 45m를 달리고, 거북이는 1분에 11m를 달린다.
 - 거북이와 토끼의 출발 위치 값(m)는 키보드로 각각 입력받는다.
 - 토끼는 몇 분 후에 거북이를 앞설 수 있을까?
 - 또 그 때까지 토끼가 달린 거리는?
 - 단, 누가 앞서고 있는가는 1분마다 확인한다고 가정한다.

- ✓ 필요한 **변수**를 찾아보시오.
- ✓ 반복을 결정하는 **반복조건**은 무엇인가?
- ✓ while과 for 문 중에 어떤 **반복구문**을 사용하면 좋을까?
- ✓ 검증을 위해 선택해야 할 거북이의 위치 값을 나열해보시오.

거북이 위치(m) 입력 : 100

토끼 위치(m) 입력 : 0

>1분 r: 45m, t: 111m

>2분 r: 90m, t: 122m

>3분 r: 135m, t: 133m

>3분 후에 133m에 거북이, 135m에 토끼가 있음

03. 반복문의 활용

[과제-3] 달팽이 우물에서 탈출하기

- ◆ 우물에 빠진 달팽이가 우물 밖으로 빠져나오는데 걸리는 기간 출력
 - 낮 동안에 55cm를 올라가고, 밤에는 13cm를 미끄러지게 된다.
 - 달팽이가 빠져있는 우물의 깊이(cm)는 키보드로 입력 받음.
 - ">%d일 만에 탈출 성공"로 출력
 - ✓ 검증을 위해 선택해야할 우물 깊이 값 나열해보시오.

```
File Edit Shell Debug Options Window Help
우물 깊이 값 입력(cm, 양수)? 500
1일째 > -445cm -458cm
2일째 > -403cm -416cm
3일째 > -361cm -374cm
4일째 > -319cm -332cm
5일째 > -277cm -290cm
6일째 > -235cm -248cm
7일째 > -193cm -206cm
8일째 > -151cm -164cm
9일째 > -109cm -122cm
10일째 > -67cm -80cm
11일째 > -25cm -38cm
12일째 > 17cm
>>12일만에 탈출 성공
Ln: 37 Col: 0
```

03. 반복문의 활용

◆ for 문의 중첩 사용 : 알고리즘 구성 과정

[연습] 아래 출력 형태를 알고리즘으로 해결

I. 변수 찾기

- ① 행 제어 변수: 열 반복에 대한 횟수 제어: 1행 ~ 5행
- ② 열 제어 변수: 열 값 반복 출력을 제어: 1열 ~ 5열

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

II. 반복 패턴 찾기

- ① 행 패턴: 1행씩 증가, 5개 행 진행
 >> 열 값 출력 개수에 영향을 줌으로 1부터 5까지로 진행
- ① 열 패턴: '*'를 반복 출력
 >> 현재 행 번호 수만큼 반복 출력

```
for i in range(1, 5+1, 1):    #행 제어  
    for j in range(1, i+1, 1): #열 제어  
        print('*', end='')  
    print()
```

03. 반복문의 활용

◆ for 문의 중첩 사용 : 알고리즘 구성 과정

❖ Ch06-TopStar.py

[실습-1] 아래 출력 형태를 알고리즘으로 해결

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

```
for i in range(1, 5+1, 1):    #행 제어  
    for j in range(1, i+1, 1): #열(횟수)  
        print('*', end='')    #열(출력값)  
    print()
```

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

```
for i in range(1, 5+1, 1):    #행(순서)  
    for j in range( , i-1, ): #열 제어  
        print('*', end='')  
    print()
```

```
for i in range(1, 5+1, 1):    #행(순서)  
    for j in range( 6-i, , ): #열 제어  
        print('*', end='')  
    print()
```

03. 반복문의 활용

◆ for 문의 중첩 사용 : 알고리즘 구성 과정

❖ Ch06-TopStar.py

[실습-2] 아래 출력 형태를 알고리즘으로 해결

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

```
for i in range(1, 5+1, 1):    #행 제어  
    for j in range(1, i+1, 1): #열(횟수)  
        print('*', end='')    #열(출력값)  
    print()
```

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

```
for i in range(    ,    ,    ):    #행 제어  
    for j in range(1, i+1, 1):    #열(횟수)  
        print('*', end='')    #열(출력값)  
    print()
```


03. 반복문의 활용

◆ for 문의 중첩 사용 : 알고리즘 구성 과정

[연습] 아래 출력 형태를 알고리즘으로 해결

I. 변수 찾기

- ① 행 제어 변수: 열 반복에 대한 횟수 제어: 1행 ~ 5행
- ② 열 제어 변수: 열 값 반복 출력을 제어: 1열 ~ 5열

II. 반복 패턴 찾기

- ① 행 패턴: 1행씩 증가, 5개 행 진행

열 값 출력에 영향을 줌으로 1부터 5까지로 진행

- ① 열 패턴: 현재 행 번호 수만큼 반복 출력

열값 출력은 행 값부터 시작, 값 변화 없이 계속 행 값 출력

```
1
22
333
4444
55555
```

03. 반복문의 활용

◆ for 문의 중첩 사용 : 알고리즘 구성 과정

[연습] 아래 출력 형태를 알고리즘으로 해결

I. 변수 찾기

- ① 행 제어 변수: i (열 반복에 대한 횟수 제어: 1행 ~ 5행)
- ② 열 제어 변수: j (열 값 반복 출력을 제어: 1열 ~ 5열)

II. 반복 패턴 찾기

- ① 행 패턴: 1행씩 증가, 5개 행 진행

열 값 출력에 영향을 줌으로 1부터 5까지로 진행

- ① 열 패턴: 현재 행 번호 수만큼 반복 출력

열값 출력은 행 값부터 시작, 값 변화 없이 계속 행 값 출력

```
1
22
333
4444
55555
```

```
for i in range(1, 5+1, 1):    #행 제어
    for j in range(1, i+1, 1): #열(횟수)
        print(i, end=' ')    #열(출력값)
    print()
```

03. 반복문의 활용

◆ for 문의 중첩 사용 : 알고리즘 구성 과정

[실습-1] 아래 출력 형태를 알고리즘으로 해결

```
1
22
333
4444
55555
```

```
for i in range(1, 5+1, 1):    #행 제어
    for j in range(1, i+1, 1): #열(횟수)
        print(i, end=' ')    #열(출력값)
    print()
```

```
11111
2222
333
44
5
```

```
for i in range(1, 5+1, 1):    #행 제어
    for j in range(1, i+1, 1): #열(횟수)
        print(i, end=" ")    #열(출력값)
    print()
```

03. 반복문의 활용

◆ for 문의 중첩 사용 : 알고리즘 구성 과정

[실습-2] 아래 출력 형태를 알고리즘으로 해결

```
54321
5432
543
54
5
```

```
for i in range(      ):      #행 제어
    for j in range( 5,      ): #열 제어
        print(      , end='')
    print()
```

03. 반복문의 활용

[과제-4] 숫자 탑 쌓기

◆ 아래 결과와 같이 출력되도록 중첩 for 문을 사용하여 해결하시오.

- for 문을 복수개 사용
- 각 열의 숫자 출력은 해 또는 열 제어 값을 사용

54321

5432

543

54

5

02. 반복문의 종류

[문제]

5. 아래 코드의 실행 결과를 적으시오.

```
for i in range(1, 5):  
    for j in range(1, i+1):  
        print("%d" %j, end=' ')  
    print()
```

6. 아래 코드의 실행 결과를 적으시오.

```
for i in range(4, 0, -1):  
    for j in range(1, i+1):  
        print("%d" %j, end=' ')  
    print()
```

03. 반복문의 활용

III. for 문

실습 6-8

구구단 프로그램 만들기

code06-08.py

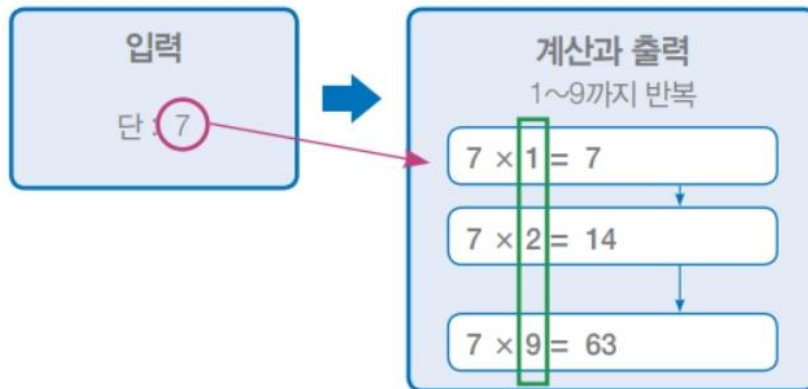


그림 6-15 구구단 출력하기

```
① 01 dan = int(input("몇 단을 출력할까요? : "))
    02
    03 for i in range(1, 10):
    04     print("%d x %d = %2d" %(dan, i, dan * i))
```

② 몇 단을 출력할까요? : 7

```
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
```

03. 반복문의 활용

◆ for 문의 중첩 사용

❖ Ch06-08구구단00.py

실습 6-8

구구단 프로그램 만들기

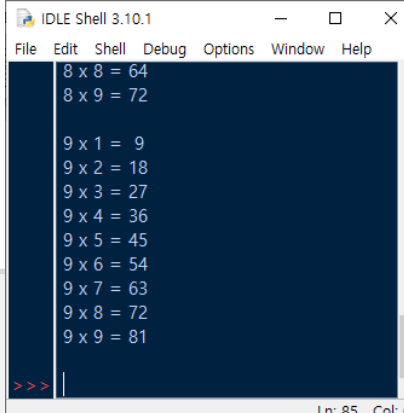
code06-08.py

```
01 dan = int(input("몇 단을 출력할까요? : "))
02
03 for i in range(1, 10):
04     print("%d x %d = %2d" %(dan, i, dan * i))
```

[실습] 구구단표 출력 (1열 출력)

◆ 2단에서 9단까지 모두 출력하는 프로그램을 작성하시오.

- 2~9단까지 1열로 출력



<단 반복: 2~9단>

<곱 반복: 1~9곱>

*총 반복: 8*9=72

```
for dan in range(2, 9+1):
    for i in range(1, 9+1):
        print("%d x %d = %2d" %(dan, i, dan*i))
```

❖ 반복의 1회전이 빠른 반복문이 안으로 들어감

03. 반복문의 활용

[실습] 구구단표 출력 (1열 출력)

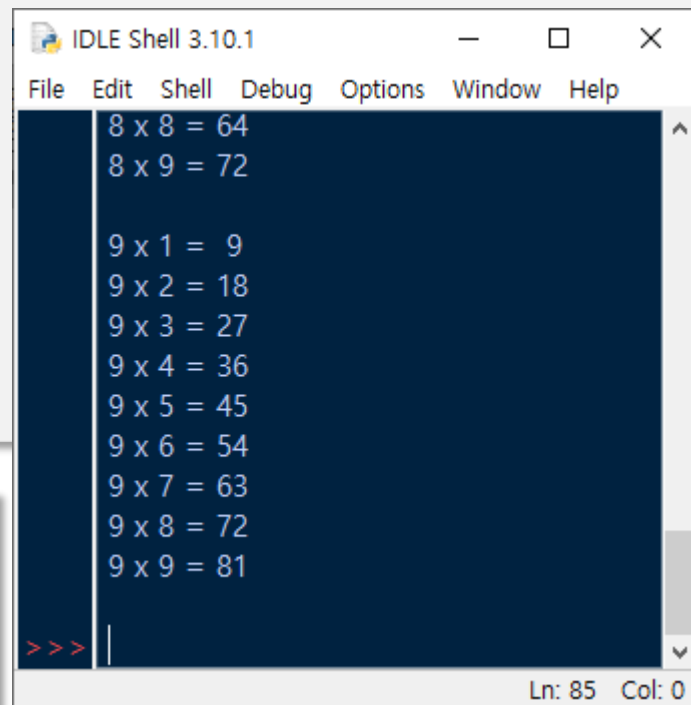
❖ Ch06-08구구단01.py

◆ 2단에서 9단까지 모두 출력하는 프로그램을 작성하시오.

- 2~9단까지 1열로 출력
- 단 제목 붙이기

1. 필요한 변수 찾기 : 단, 곱
2. 반복패턴 찾기: 단 x 곱 = 단*곱
3. 반복패턴 찾기
 - 단 반복: 2~9, +1
 - 곱 반복: 1~9, +1
4. 반복 패턴 배치(반복 변화 우선 순위 → 1회전이 빠른)
 - 늦은 밖 순위: 단(2~9)
 - 빠른 안 순위: 곱(1~9)

```
for dan in range(2, 9+1):  
    for i in range(1, 9+1):  
        print("%d x %d = %2d" %(dan, i, dan*i))
```



```
IDLE Shell 3.10.1  
File Edit Shell Debug Options Window Help  
8 x 8 = 64  
8 x 9 = 72  
  
9 x 1 = 9  
9 x 2 = 18  
9 x 3 = 27  
9 x 4 = 36  
9 x 5 = 45  
9 x 6 = 54  
9 x 7 = 63  
9 x 8 = 72  
9 x 9 = 81  
>>> |  
Ln: 85 Col: 0
```

❖ 반복의 1회전이 빠른 반복문이 안으로 들어감

03. 반복문의 활용

[실습] 구구단표 출력 (8열 출력)

❖ Ch06-08구구단02.py

◆ 2단에서 9단까지 모두 출력하는 프로그램을 작성하시오.

- 2~9단까지 8열 **1행**으로 출력

1. 필요한 **변수** 찾기 : 단, 곱
2. **반복패턴** 찾기: 단 x 곱 = 단*곱
3. 반복패턴 찾기
 - 단 반복: 2~9, +1
 - 곱 반복: 1~9, +1
4. 반복 패턴 배치(반복 변화 우선 순위 → 먼저 변하는 것(안 순위))
 - **늦은 밖 순위:**
 - **빠른 안 순위:**

```
>>> 2 x 1 = 2      3 x 1 = 3      4 x 1 = 4      5 x 1 = 5      6 x 1 = 6      7 x 1 = 7      8 x 1 = 8      9 x 1 = 9
2 x 2 = 4      3 x 2 = 6      4 x 2 = 8      5 x 2 = 10     6 x 2 = 12     7 x 2 = 14     8 x 2 = 16     9 x 2 = 18
2 x 3 = 6      3 x 3 = 9      4 x 3 = 12     5 x 3 = 15     6 x 3 = 18     7 x 3 = 21     8 x 3 = 24     9 x 3 = 27
2 x 4 = 8      3 x 4 = 12     4 x 4 = 16     5 x 4 = 20     6 x 4 = 24     7 x 4 = 28     8 x 4 = 32     9 x 4 = 36
2 x 5 = 10     3 x 5 = 15     4 x 5 = 20     5 x 5 = 25     6 x 5 = 30     7 x 5 = 35     8 x 5 = 40     9 x 5 = 45
2 x 6 = 12     3 x 6 = 18     4 x 6 = 24     5 x 6 = 30     6 x 6 = 36     7 x 6 = 42     8 x 6 = 48     9 x 6 = 54
2 x 7 = 14     3 x 7 = 21     4 x 7 = 28     5 x 7 = 35     6 x 7 = 42     7 x 7 = 49     8 x 7 = 56     9 x 7 = 63
2 x 8 = 16     3 x 8 = 24     4 x 8 = 32     5 x 8 = 40     6 x 8 = 48     7 x 8 = 56     8 x 8 = 64     9 x 8 = 72
2 x 9 = 18     3 x 9 = 27     4 x 9 = 36     5 x 9 = 45     6 x 9 = 54     7 x 9 = 63     8 x 9 = 72     9 x 9 = 81
>>>
```

Ln: 96 Col: 0

03. 반복문의 활용

[과제-5] 구구단표 출력 (n열 출력)

◆ 2단에서 9단까지 모두 출력하는 프로그램을 작성하시오.

- 2~9단까지 **n열**로 출력
- n 값을 키보드로 입력을 받음

1. 필요한 **변수** 찾기 : 단, 곱
2. **반복패턴** 찾기: 단 x 곱 = 단*곱
3. 반복패턴 찾기
 - 단 반복: 2~9, +1
 - 곱 반복: 1~9, +1
4. 반복 패턴 배치(반복 변화 우선 순위 → 1회전이 빠른)
 - **늦은 밖 순위:**
 - **빠른 안 순위:**

몇 열로 출력할까요? 5

2 x 1 = 2	3 x 1 = 3	4 x 1 = 4	5 x 1 = 5	6 x 1 = 6
2 x 2 = 4	3 x 2 = 6	4 x 2 = 8	5 x 2 = 10	6 x 2 = 12
2 x 3 = 6	3 x 3 = 9	4 x 3 = 12	5 x 3 = 15	6 x 3 = 18
2 x 4 = 8	3 x 4 = 12	4 x 4 = 16	5 x 4 = 20	6 x 4 = 24
2 x 5 = 10	3 x 5 = 15	4 x 5 = 20	5 x 5 = 25	6 x 5 = 30
2 x 6 = 12	3 x 6 = 18	4 x 6 = 24	5 x 6 = 30	6 x 6 = 36
2 x 7 = 14	3 x 7 = 21	4 x 7 = 28	5 x 7 = 35	6 x 7 = 42
2 x 8 = 16	3 x 8 = 24	4 x 8 = 32	5 x 8 = 40	6 x 8 = 48
2 x 9 = 18	3 x 9 = 27	4 x 9 = 36	5 x 9 = 45	6 x 9 = 54

7 x 1 = 7	8 x 1 = 8	9 x 1 = 9
7 x 2 = 14	8 x 2 = 16	9 x 2 = 18
7 x 3 = 21	8 x 3 = 24	9 x 3 = 27
7 x 4 = 28	8 x 4 = 32	9 x 4 = 36
7 x 5 = 35	8 x 5 = 40	9 x 5 = 45
7 x 6 = 42	8 x 6 = 48	9 x 6 = 54
7 x 7 = 49	8 x 7 = 56	9 x 7 = 63
7 x 8 = 56	8 x 8 = 64	9 x 8 = 72
7 x 9 = 63	8 x 9 = 72	9 x 9 = 81

Thank You !

[Python]