학습목표

- 파이썬 프로그램을 통하여 range 명령을 사용하여 변화 주면서 반복하기, 거북이 그래픽으로 그림 그리기, 정보입력하기, True/False 판단하기 등을 통해 프로그램을 구현 하여 원하는 결과를 얻을 수 있어야 한다.

Unit1. range 명령을 사용하여 변화를 주면서 반복하기

01 range란?

02 for 명령어로 숫자를 반복하여 출력하는 프로그램

03 1부터 10까지 숫자의 합계를 구하는 프로그램

01

슬라이드 1. range란?

- range : 파이썬에서의 range는 반복할 '범위'를 지정하는 명령어
- 대화형 셸에서 range를 확인하는 예제

```
>>> list(range(5))
[0, 1, 2, 3, 4]
>>> list(range(0, 5))
[0, 1, 2, 3, 4]
>>> list(range(1, 11))
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
>>> list(range(1, 4))
[1, 2, 3]
>>>
```

range(5) : 0,1, 2, 3, 4로 값을 다섯 개 가짐.

for x in range(5) : '변수 x의 값을 차례대로 0, 1, 2, 3, 4로 바꾸면서 반복 블록을 실행'.

range(0, 5) :
range(a, b)의 값은 a에서 시작해서b 바로 앞의 값까지 1씩 늘리면서 반복.

range(0,5) :
0부터 시작해서 5 바로 앞의 값까지 반복
0, 1, 2, 3, 4를 출력.

range(1, 11) :
1에서 시작해서 11 바로 앞(10)까지를 반복
1, 2, 3, 4, 5, 6, 7, 8, 9, 10을 출력.

실습

range(1, 4) : 1에서 4 바로 앞(3)까지 반복
1, 2, 3을 출력.

02

슬라이드2. for 명령어로 숫자를 반복하여 출력하는 프로그램

```
print("[0-4]")
for x in range(5):          # range(5)로 0, 1, 2, 3, 4까지 다섯 번 반복
    print(x)                # 변수 x 값을 출력

print("[1-10]")
for x in range(1, 11):      # 1, 2, ..., 10까지 열 번 반복(11은 제외).
    print(x)                # 변수 x 값을 출력
```

실습

- range 명령어를 쓸 때 두 가지는 꼭 기억하세요!

① range(5) : 0부터 시작해서 4까지 다섯 번 반복한다(5는 제외한다).
② range(1, 11) : 1부터 시작해서 10까지 열 번 반복한다(11은 제외한다).

03

슬라이드3. 1부터 10까지 숫자의 합계를 구하는 프로그램

```
s = 0                          # 합계를 구하는 변수 s, 처음 값은 0을 입력
for x in range(1, 10+1):.      # 1, 2, ..., 10으로 열 번 반복(11은 제외).
    s = s+x                    # 현재의 s 값에 x 값을 더함
    print("x:", x, " sum:", s) # 현재의 x 값과 s 값을 출력
```

실행결과

```
x: 1 sum: 1
x: 2 sum: 3
x: 3 sum: 6
x: 4 sum: 10
x: 5 sum: 15
x: 6 sum: 21
x: 7 sum: 28
x: 8 sum: 36
x: 9 sum: 45
x: 10 sum: 55
```

실습

Unit 2. 거북이 그래픽으로 그림 그리기

01 거북이 그래픽 사용하기

02 거북이 그래픽의 동작 방식

01

슬라이드1. 거북이 그래픽 사용하기

as t를 붙이지 않았을 때	as t를 붙였을 때
import turtle turtle.forward(100) turtle.right(100) turtle.forward(100)	import turtle as t t.forward(100) t.right(100) t.forward(100)

02

슬라이드 2. 거북이 그래픽의 동작 방식

» 자주 사용하는 거북이 그래픽 명령어 1

함수	설명	사용 예
forward(거리)/ fd(거리)	거북이가 앞으로 이동합니다.	t.forward(100) # 거북이가 100만큼 앞으로 이동합니다.
backward(거리) / back(거리)	거북이가 뒤로 이동합니다.	t.back(50) # 거북이가 50만큼 뒤로 이동합니다.
left(각도) / lt(각도)	거북이가 왼쪽으로 회전합니다.	t.left(45) # 거북이가 45도 왼쪽으로 회전합니다.
right(각도) / rt(각도)	거북이가 오른쪽으로 회전합니다.	t.right(45) # 거북이가 45도 오른쪽으로 회전합니다.
circle(반지름)	현재 위치에서 원을 그립니다.	t.circle(50) # 반지름이 50인 원을 그립니다.
down( ) / pendown( )	펜(잉크 묻힌 꼬리)을 내립니다.	t.down() # 이제 움직이면 그림이 그려집니다.
up( ) / penup( )	펜(잉크 묻힌 꼬리)을 올립니다.	t.up() # 거북이가 움직여도 선이 그려지지 않습니다.
shape("모양")	거북이 모양을 바꿉니다.	t.shape("turtle") # 진짜 거북이 모양으로 지정합니다. t.shape("arrow") # 화살표 모양의 거북이로 지정합니다. ※ 거북이 모양으로 "circle", "square", "triangle"을 사용할 수 있습니다.
speed(속도)	거북이 속도를 바꿉니다.	t.speed(1) # 가장 느린 속도 t.speed(10) # 빠른 속도 t.speed(0) # 최고 속도

02

2. 거북이 그래픽의 동작 방식

» 자주 사용하는 거북이 그래픽 명령어 1

pensize(굵기) / width	펜 굵기를 바꿉니다.	t.pensize(3) # 굵은 선으로 선을 그립니다.
color("색 이름")	펜 색을 바꿉니다.	t.color("red") # 빨간색으로 선을 그립니다.
bgcolor("색 이름")	화면의 배경색을 바꿉니다.	t.bgcolor("black") # 배경색을 흰색에서 검은색으로 바꿉니다.
fillcolor("색 이름")	도형 내부를 칠하는 색을 바꿉니다.	t.fillcolor("green") # 녹색으로 도형 내부를 칠합니다. ※ 색상을 따로 지정하지 않으면 현재 색으로 칠합니다.
begin_fill( )	도형 내부를 색칠할 준비를 합니다.	t.begin_fill() # 거북이 움직임을 색칠할 준비를 합니다.
end_fill( )	도형 내부를 색칠합니다.	t.end_fill() # begin_fill( ) 이후부터 지금까지 그린 그림에 맞춰 내부를 색칠합니다.
showturtle( ) / st( )	거북이를 화면에 표시합니다.	t.st() # 거북이를 화면에 표시합니다(기본 상태).
hideturtle( ) / ht( )	거북이를 화면에서 가립니다.	t.ht() # 거북이를 숨깁니다.
clear( )	거북이를 그대로 둔 채 화면을 지웁니다.	t.clear()
reset( )	화면을 지우고 거북이도 원래 자리와 상태로 되돌립니다.	t.reset()

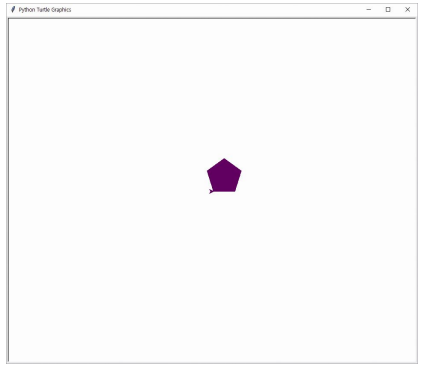
02

2. 정오각형을 그리는 프로그램

» import turtle as t

```
n = 5                   # 오각형을 그림(다른 값을 입력하면 다른 도형을 그림)
t.color("purple")
t.begin_fill()          # 색칠할 영역을 시작
for x in range(n):      # n번 반복
    t.forward(50)       # 거북이가 50만큼 앞으로 이동
    t.left(360/n)       # 거북이가 360/n만큼 왼쪽으로 회전
t.end_fill()            # 색칠할 영역을 마무리
```

실습



02

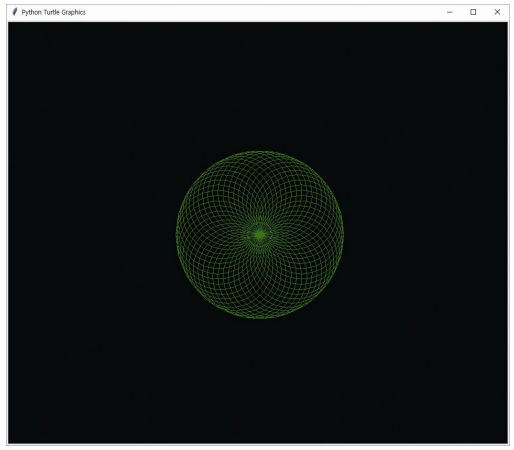
2. 거북이 그래픽의 동작 방식

```
import turtle as t

n = 50
t.bgcolor("black")
t.color("green")
t.speed(0)
for x in range(n):
  t.circle(80)
  t.left(360/n)
```

원을 50개 그립니다.
배경색을 검은색으로 지정합니다.
펜색을 녹색으로 지정합니다.
거북이 속도를 가장 빠르게 지정합니다.
n번 반복합니다.
현재 위치에서 반지름이 80인 원을 그립니다.
거북이가 360/만큼 왼쪽으로 회전합니다.

실습



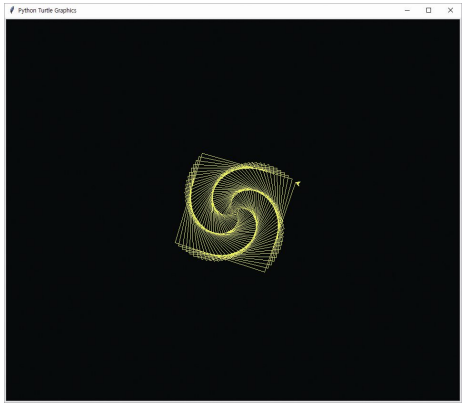
02

2. 선을 반복해서 그리는 프로그램

```
import turtle as t

angle = 89                  # 거북이가 왼쪽으로 회전할 각도를 지정(값을 바꿀 수 있음).
t.bgcolor("black")          # 배경색을 검은색으로 지정
t.color("yellow")           # 펜 색을 노란색으로 지정
t.speed(0)                  # 거북이 속도를 가장 빠르게 지정
for x in range(200):        # x 값을 0에서 199까지 바꾸면서 200번 실행
    t.forward(x)            # x만큼 앞으로 이동(실행을 반복하면서 선이 길어짐)
    t.left(angle)           # 거북이가 왼쪽으로 89도 회전
```

실습



Unit 3. 정보 입력하기

01 컴퓨터의 입출력 장치

02 파이썬의 입력 처리

03 파이썬의 자료형

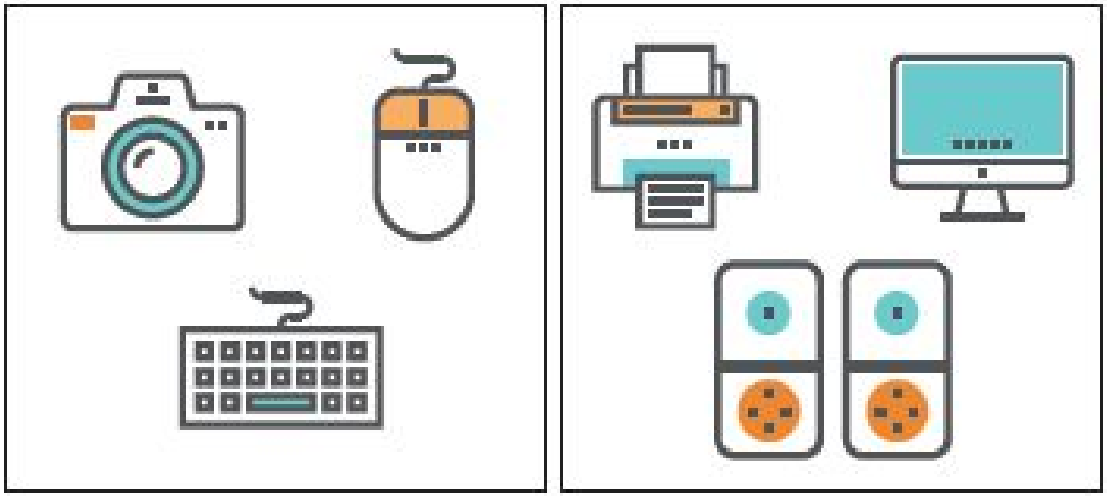
04 문자열이란?

01

슬라이드 1. 컴퓨터의 입출력 장치

» '입력 →처리 → 출력' 과정을 효과적으로 처리하기 위해 다양한 종류의 입력 장치와 출력 장치가 있음

- 입력 장치 : 키보드, 마우스, 터치스크린, 마이크, 카메라, 스캐너
- 출력 장치 : 모니터, 프린터, 스피커



02

슬라이드 2. 파이썬의 입력 처리

» 이름을 입력받아 Hello와 함께 보여 주는 프로그램

```
name = input("Your name? ")    # 이름을 입력받아 name 변수에 저장
print("Hello", name)           # Hello와 함께 name을 출력
```

» 실행결과

```
Your name? 김길벗
Hello 김길벗
```

» 파이썬과 한글 입력

영어를 사용하는 사람들이 컴퓨터를 발명했기 때문에 대부분의 컴퓨터 시스템은 영어를 기준으로 설계되었습니다. 컴퓨터가 전 세계로 보급되면서 영어가 아닌 언어도 사용할 수 있게 되었지만, 여전히 특정 컴퓨터나 환경에서는 한국어나 다른 언어를 사용 하기 어렵습니다. 파이썬 역시 윈도(Windows)에서는 한글을 사용하는 데 큰 문제가 없지만, 애플(Apple)의 맥(Mac) 운영체제인 OS X에서는 한글이 제대로 입력되지 않는 현상이 나타납니다. 안타깝게도 이럴 때는 예제에 나오는 한글 문자열을 영어로 바꿔서 입력해야 합니다. input( )을 이용해서 문자열을 입력 받을 때도 한글을 입력하면 제대로 입력되지 않습니다.

03

슬라이드 3. 파이썬의 자료형

» 파이썬에서 자주 사용하는 자료형

자료형	영어 이름	파이썬 표기	설명	예
정수	Integer	int	소수점이 없는 수	-2, -1, 0, 1, 2, 3
소수	Floating-point number	float	소수점(.)이 있는 수, 부동소수점수라고도 불린다.	-3.5, 0.0, 1.25, 5.0
문자열	String	str	알파벳 혹은 다른 문자로 이루어진 문장	“a”, “abc”, Hello?”, “3 people”, “비”, “여름”

04

슬라이드 4. 문자열이란?

- 문자열 : '문자의 나열'
  - 한 글자로 된 문자열 : "a", "가"
  - 단어로 된 문자열 : "boy", "소년"
  - 문장으로 된 문자열 : "It rains.", "비가 옵니다."

실습

04

4. 문자열이란?

» 숫자 두 개를 입력받아 곱하는 프로그램

```
x = input("?")                  # 변수 x에 첫 번째 입력을 받습니다. x = 문자열
a = int(x)                      # 문자열 x의 값을 정수(int)로 바꿔서 a에 넣음

x = input("?")                  # 변수 x에 두 번째 입력을 받습니다. x = 문자열
b = int(x)                      # 문자열 x의 값을 정수(int)로 바꿔서 b에 넣음

print(a * b)                    # a와 b를 곱한 결과를 출력
```

» 실행결과

```
? 3
? 7
21
```

실습

04

4. 문자열이란?

» 속으로 20초를 세어 맞히는 프로그램

```
import time

input("엔터를 누르고 20초를 셉니다.")
start = time.time()

input("20초 후에 다시 엔터를 누릅니다.")
end = time.time()

et = end - start          # end 시간에서 start 시간을 빼면 실제 걸린 시간을 계산할 수 있음
print("실제 시간 :", et, "초")
print("차이 :", abs(et - 20), "초")
```

» 실행결과

```
엔터를 누르고 20초를 셈
20초 후에 다시 엔터를 누름
실제 시간 : 20.608863830566406 초
차이 : 0.6088638305664062 초
```

실습

Unit 4. True/False 판단하기

01 True/False와 비교 연산자

02 판단 명령어 if

01

슬라이드 1. True/False와 비교 연산자

연산자	설명	예
==	양쪽이 같다(같으면 True, 다르면 False).	3 == 3 → True 1 == 7 → False
!=	양쪽이 다르다(다르면 True, 같으면 False).	3 != 3 → False 1 != 7 → True
<	왼쪽이 오른쪽보다 작다.	3 < 7 → True 3 < 3 → False
>	왼쪽이 오른쪽보다 크다.	7 > 3 → True 7 > 7 → False
<=	왼쪽이 오른쪽보다 작거나 같다.	3 <= 7 → True 3 <= 3 → True
>=	왼쪽이 오른쪽보다 크거나 같다.	7 >= 3 → True 7 >= 7 → True

01

1. True/False와 비교 연산자

» 비교 연산자(==)와 대입 연산자(=)

연산자	설명
3==7	3과 7이 같은지 판단(False)
a=3	변수 a에 3을 저장
A==3	변수 a에 저장된 값이 3과 같은지 판단 ※현재 a에 저장된 값에 따라 True인지 False인지 결정

02

슬라이드 2. 판단 명령어 if

- if 비교할 문장:
    True일 때 실행할 문장(들여쓰기)

- if 비교할 문장:
    True일 때 실행할 문장(들여쓰기)
  else:
    False일 때 실행할 문장(들여쓰기)

02

2. 판단 명령어 if

» **if 판단문과 if-else로 판단하는 프로그램**

```
a = 3                   # 변수 a에 3을 저장

if a == 2:              # a가 2와 같은지 비교함
    print("A")          # False이므로 이 부분은 실행되지 않음

if a == 3:              # a가 3과 같은지 비교
    print("B")          # True이므로 이 부분이 실행됨.

if a == 4:              # a가 4와 같은지 비교
    print("C")          # False이므로 이 부분은 실행되지 않음

else:                   # print(“C”) 대신 이 부분이 실행됨
    print("D)
```

» 실행결과

B
D

실습

02

2. 판단 명령어 if

» **덧셈 문제를 맞히는 프로그램**

```
x = input("12+23 = ")        # 문제를 보여 주고 답을 입력받아 x에 저장(문자열임).
a = int(x)                   # 숫자를 비교할 수 있게 x에 저장된 문자열을 정수로 바꿈

if a == 12+23:
    print("천재!")
else:
    print("바보?")
```

» 실행결과

```
12+23 = 33
바보?
```

실습

» 학습정리

» 변수 이름 정하기!

변수는 변할 수 있는 값, 즉 컴퓨터가 정보를 보관하는 곳입니다. 컴퓨터는 수많은 정보를 보관하고, 이러한 정보를 구분하려면 변수 이름이 필요하다고 배웠습니다. 우리는 지금까지 a, b, c, d, x, s처럼 매우 단순한 이름으로 변수 이름을 지었습니다. 변수 이름은 프로그램을 만드는 사람이 자유롭게 정할 수 있지만, 모든 이름을 변수 이름으로 쓸 수 있는 것은 아닙니다. 변수 이름을 정하려면 다음과 같은 규칙을 따라야 합니다.

① 변수 이름은 영문 대/소문자, 숫자, 밑줄(_)로만 만들 수 있습니다. 변수 이름에는 공백을 사용할 수 없습니다.
② 변수 이름은 숫자로 시작할 수 없습니다. 즉, 3name이나 150m은 변수로 사용할 수 없습니다.
③ 영문 대/소문자를 구분합니다. 즉, A와 a는 다른 변수입니다.
④ 파이썬이 문법으로 사용하는 단어는 헷갈릴 수 있으므로 사용할 수 없습니다. 예를 들면, False, None, True, and, as, assert, break, class, continue, def, del, elif, else, except, finally, for, from, global, if, import, in, is, lambda, nonlocal, not, or, pass, raise, return, try, while, with, yield 등입니다.

이러한 규칙을 지키면서 자유롭게 변수 이름을 정할 수 있습니다. 너무 길지 않은 것이 좋고, 변수 이름을 봤을 때 변수에 저장된 정보가 무엇인지 예상할 수 있는 이름이라면 더욱 좋습니다.

학습평가

1. 다음 예제들에 대한 실행 결과는?

>>> 1+1 == 2 정답 : True

>>> 3-1 == 1 정답 : False

>>> 3 == 3 정답 : True

>>> 3 != 3 정답 : False

>>> 7 >= 3 정답 : True

>>> "abc" == "abc" 정답 : True

>>> "abc" == "ABC" 정답 : False