

08

CHAPTER

문자열

학습목표

- 문자열을 익힌다.
- 문자열을 조작하는 함수를 익힌다.
- 문자열의 활용법을 익힌다.

SECTION 01 이장에서 만들 프로그램

SECTION 02 문자열 기본

SECTION 03 문자열 함수

요약

연습문제

응용예제



Section01 이 장에서 만들 프로그램

■ [프로그램 1] 입력된 문자열 거꾸로 출력

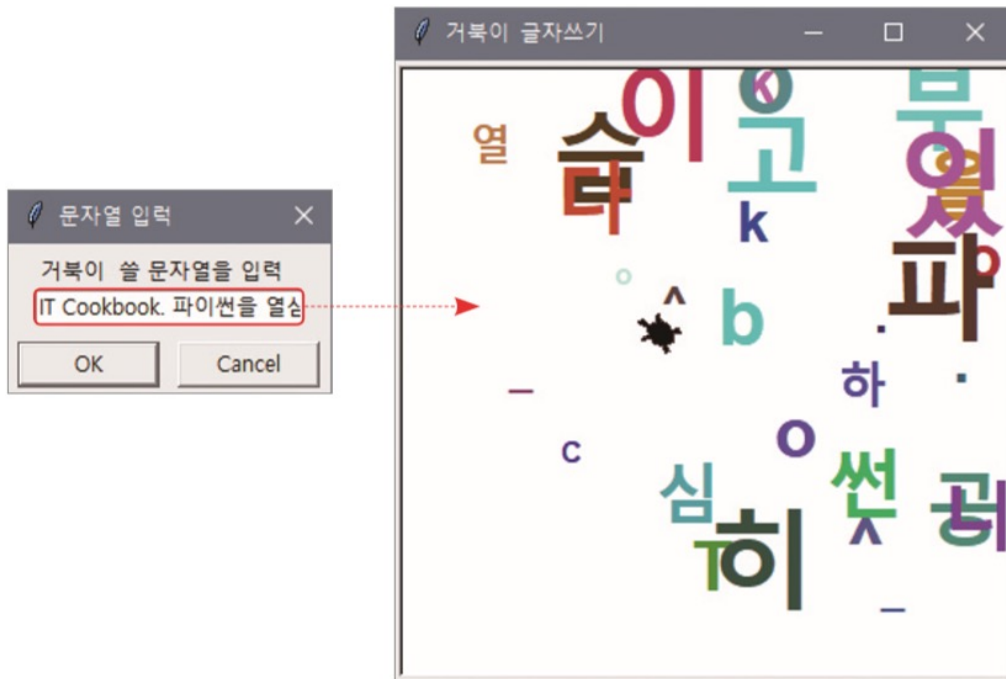


The image shows a Python 3.6.0 Shell window. The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The text area contains the following code and output:

```
===== RESTART: C:\CookPython\Code08-02.py =====  
문자열을 입력하세요 : 즐거운 Python 프로그래밍~~~  
내용을 거꾸로 출력 --> ~~~밍레그로프 nohtyP 윤거즐  
>>>
```

A red dashed arrow points from the text "사용자가 입력한 값" (Value entered by the user) to the input string "즐거운 Python 프로그래밍~~~". The status bar at the bottom right shows "Ln: 7 Col: 4".

■ [프로그램 2] 임의의 위치에 글자를 쓰는 거북이



tkinter.simpdialog

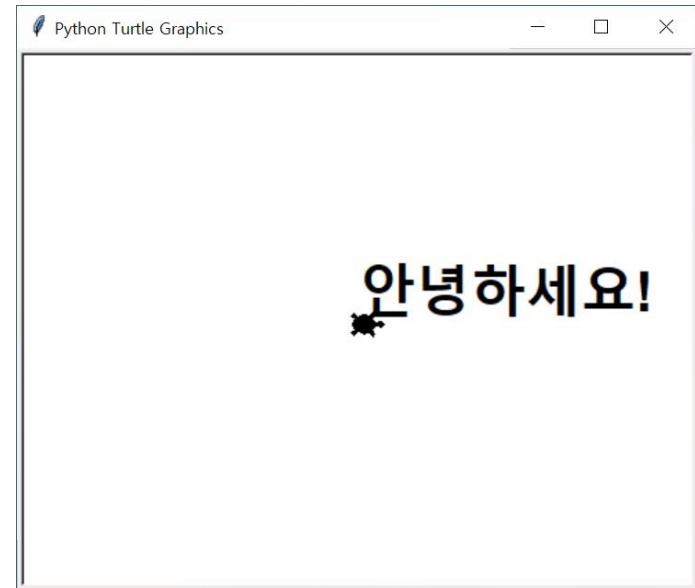
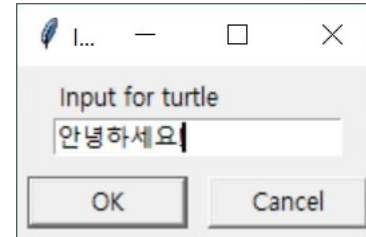
```
import tkinter
# import tkinter as tk
# from tkinter.simpdialog import *

import turtle as t

swidth, sheight = 300, 300
txtSize = 30
t.shape('turtle')
t.setup(width = swidth + 50, height = sheight + 50)
t.screensize(swidth, sheight)

retStr = tkinter.simpdialog.askstring('Input char', 'Input for turtle')
t.write(retStr, font=('맑은고딕', txtSize, 'bold'))

t.done()
```



Section02 문자열 기본

■ 문자열의 개념

- 리스트 코드와 비교 : 리스트는 대괄호 []로 묶고 문자열은 작은따옴표로 묶어 출력
- 문자열은 immutable 자료형, 리스트는 mutable 자료형

```
aa = [10, 20, 30, 40, 50]  
aa[0]  
aa[1:3]  
aa[3:]
```

변경 X

변경 O

출력 결과

```
10  
[20, 30]  
[40, 50]
```

```
ss = "파이썬최고"  
ss[0]  
ss[1:3]  
ss[3:]
```

출력 결과

```
'파'  
'이썬'  
'최고'
```

Section02 문자열 기본

- 더하기(+) 기호 사용해 연결. 또 곱하기(*) 기호 사용 문자열 반복

```
ss = '파이썬' + '최고'
```

```
ss
```

```
ss = '파이썬' * 3
```

```
ss
```

출력 결과

```
'파이썬최고'
```

```
'파이썬파이썬파이썬'
```

Section02 문자열 기본

- len() 함수 : 리스트나 문자열의 개수를 셀 때 사용

```
ss = '파이썬abcd'  
len(ss)
```

출력 결과

7

Section02 문자열 기본

- 문자열의 모든 글자 뒤에 \$를 붙여서 출력하는 코드

Code08-01.py

```
1 ss = '파이썬짱!'
2
3 sslen = len(ss)
4 for i in range(0, sslen) :
5     print(ss[i] + '$', end = '')
```

3행 : 문자열의 길이를 sslen 변수에 저장
4행 : 문자열의 길이만큼 반복
5행 : 글자 하나와 \$를 붙여서 출력

출력 결과

파\$이\$썬\$짱\$!\$

SELF STUDY 8-1

Code08-01.py를 수정해서 '파이썬은완전재미있어요'에서 '파#썬#완#재#있#요'를 출력해 보자. 즉 0부터 시작한다고 가정하면 짝수 번째 글자는 그대로 출력되고, 홀수 번째는 글자 대신 #이 표시되도록 하면 된다.

힌트 if 문을 사용해 i가 짝수일 때와 홀수일 때를 다르게 처리한다. 짝수는 2로 나누어서 나머지가 0이면 짝수이다.

Section02 문자열 기본

■ [프로그램 1]의 완성

- 문자열을 입력받아 반대로 출력

Code08-02.py

```
1  ## 변수 선언 부분 ##      2행 : inStr은 문자열 입력받을 변수, outStr은 문자열을 거꾸로 저장하는 변수
2  inStr, outStr = "", ""    3행 : count는 문자열의 개수를 저장 i는 0, 1, 2, ...로 변함
3  count, i = 0, 0          6행 : 문자열 입력
4                          7행 : 입력받은 문자열의 개수 계산
5                          9행 : 문자열의 개수만큼 반복.
5  ## 메인 코드 부분 ##
6  inStr = input("문자열을 입력하세요 : ")
7  count = len(inStr)
8
9  for i in range(0, count) :
10     outStr += inStr[count - (i + 1)]
11
12  print("내용을 거꾸로 출력 --> %s" % outStr)
```

10행 : 입력한 문자열이 3글자라면 inStr[0], inStr[1], inStr[2] 3개가 있는 것이므로 첫 번째(i가 0)에는 inStr[2]가 추가, 두 번째(i가 1)에는 inStr[1]이 추가, 세 번째(i가 2)에는 inStr[0]이 추가
12행 : 거꾸로 돌린 문자열 출력

```
Python 3.6.0 Shell
File Edit Shell Debug Options Window Help
===== RESTART: C:\CookPython\Code08-02.py =====
문자열을 입력하세요 : 즐거운 Python 프로그래밍~~~
내용을 거꾸로 출력 --> ~~~밍래그로프 nohtyP 윤거즐
>>>
```

사용자가 입력한 값

Ln: 7 Col: 4

Section03 문자열 함수로그램

■ 문자열 함수의 사용

- 대문자와 소문자 변환하기 : upper(), lower(), swapcase(), title()

```
ss = 'Python is Easy. 그래서 programming이 재미있습니다. ^^'  
ss.upper()  
ss.lower()  
ss.swapcase()  
ss.title()
```

출력 결과

```
'PYTHON IS EASY. 그래서 PROGRAMMING이 재미있습니다. ^^'  
'python is easy. 그래서 programming이 재미있습니다. ^^'  
'pYTHON IS eASY. 그래서 PROGRAMMING이 재미있습니다. ^^'  
'Python Is Easy. 그래서 Programming이 재미있습니다. ^^'
```

Section03 문자열 함수로그람

여기서 잠깐



함수(Function)와
메서드(Method)

함수와 메서드는 상당히 비슷하지만 차이점이 약간 있다. 우선 함수는 단독으로 사용된다. 예로 리스트나 문자열의 길이를 알아내는 len() 함수는 다음과 같이 사용된다.

```
ss = "abcd"  
len(ss)
```

하지만 메서드는 문자열 자료형에 그 기능이 들어 있기 때문에 '변수명.메서드()' 형식으로 사용된다. 예로 문자열을 대문자로 바꾸는 upper() 메서드는 다음과 같이 사용된다.

```
ss = "abcd"  
ss.upper()
```

※ 메소드

객체.메소드()

12장에서 배울 객체지향에서는 함수와 메서드를 정확히 구분해야 하지만 지금은 '둘 다 뒤에 괄호가 붙는다' 정도만 알면 되므로 당분간은 모두 함수라고 칭한다.

Section03 문자열 함수로그람

- 문자 찾기 : count(), find(), rfind(), index(), rindex(), startswith(), endswith()

```
ss = '파이썬 공부는 즐겁습니다. 물론 모든 공부가 다 재미있지는 않죠. ^^'
ss.count('공부')
print(ss.find('공부'), ss.rfind('공부'), ss.find('공부', 5), ss.find('없다'))
print(ss.index('공부'), ss.rindex('공부'), ss.index('공부', 5))
print(ss.startswith('파이썬'), ss.startswith('파이썬', 10), ss.endswith('^^'))
```

출력 결과

```
2
4 21 21 -1
4 21 21
True False True
```

- 실습 : 결과 값은?

```
a = 'happy python'
print(a.find('a'))
print(a.find('h'))
print(a.rfind('h'))
print(a.find('h', 5))
print(a.find('x'))
print(a.startswith('p'))
print(a.endswith('n'))
```

Section03 문자열 함수로그래밍

- 문자열이 괄호로 감싸 있지 않으면 괄호로 감싸 주는 프로그램

Code08-03.py

```
1 ss = input("입력 문자열 ==> ")
2 print("출력 문자열 ==> ", end = '')
3
4 if ss.startswith('(') == False :
5     print("(", end = '')
6
7 print(ss, end = '')
8
9 if ss.endswith(')') == False :
10     print(")", end = '')
```

1행 : 문자열 입력

4행 : 문자열의 시작이 (가 아니면 (를 우선 출력

7행 : 입력한 문자열을 그대로 출력

9행 : 문자열의 끝이)가 아니면)를 우선 출력

출력 결과

입력 문자열 ==> 파이썬 열공 중~~

출력 문자열 ==> (파이썬 열공 중~~)

Section03 문자열 함수로그람

- 문자열 공백 삭제·변경하기 : strip(), rstrip(), lstrip(), replace()

```
ss = ' 파 이 션 '  
ss.strip()  
ss.rstrip()  
ss.lstrip()
```

출력 결과

```
'파 이 션'  
' 파 이 션'  
'파 이 션 '
```

Section03 문자열 함수로그람

- 앞뒤의 특정 문자 삭제

```
ss = '----파---이---썬----'  
print(ss.strip('-'))  
ss = '<<<파 << 이 >> 썬>>>'  
print(ss.strip('◇'))
```

출력 결과

```
파---이---썬  
파 << 이 >> 썬
```

Section03 문자열 함수로그래밍

- 문자열中间的 공백까지 삭제해 주는 코드

Code08-04.py

```
1 inStr = " 한글 Python 프로그래밍 "
```

```
2 outStr = ""
```

```
3
```

```
4 for i in range(0, len(inStr)) :
```

```
5     if inStr[i] != ' ' :
```

```
6         outStr += inStr[i]
```

```
7
```

```
8 print("원래 문자열 ==> " + '[' + inStr + ']')
```

```
9 print("공백 삭제 문자열 ==> " + '[' + outStr + ']')
```

출력 결과

원래 문자열 ==> [한글 Python 프로그래밍]

공백 삭제 문자열 ==> [한글Python프로그래밍]

SELF STUDY 8-2

Code08-04.py를 수정해서 '《《파《이》썬》》'이 '파이썬'으로 출력되도록 해 보자.

힌트 if문이 '《'이 아닐 때와 '》'이 아닐 때를 and로 연결해야 한다.

Section03 문자열 함수로그람

- 문자열 변경

```
ss = '열심히 파이썬 공부 중~'  
ss.replace('파이썬', 'Python')
```

출력 결과

```
'열심히 Python 공부 중~'
```


Section03 문자열 함수로그래밍

- 문자열을 입력받아 그중 o를 \$로 변경하는 문자열 변경을 응용

Code08-05.py

```
1 ss = input("입력 문자열 ==> ")
2
3 print("출력 문자열 ==> ", end = '')
4 for i in range(0, len(ss)) :
5     if ss[i] != 'o' :
6         print(ss[i], end = '')
7     else :
8         print('$', end = '')
```

1행 : 문자열을 입력
4행 : 입력된 문자열의 개수만큼 반복
5~8행 : 문자가 o 라면 \$ 대신 출력

출력 결과

입력 문자열 ==> IT CookBook for Python

출력 문자열 ==> IT C\$\$kB\$\$k f\$r Pyth\$n

- 4~8행을 한줄로

```
print(ss.replace('o', '$'))
```

Section03 문자열 함수로그람

- 문자열 분리·결합하기 : split(), splitlines(), join()

```
ss = 'Python을 열심히 공부 중'
ss.split()
ss = '하나:둘:셋'
ss.split(':')
ss = '하나\n둘\n셋'
ss.splitlines()
ss = '%'
ss.join('파이썬')
```

출력 결과

```
['Python을', '열심히', '공부', '중']
['하나', '둘', '셋']
['하나', '둘', '셋']
'파%이%썬'
```

- 리스트를 문자열로 출력

```
a = ['a', 'b', 'c']
result1 = '%'.join(a)
result2 = '\n'.join(a)
print(result1)
print(result2)
```

출력 결과

```
a%b%c
a
b
c
```

Section03 문자열 함수로그래밍

- 연/월/일 형식으로 문자열을 입력받아 10년 후 날짜를 출력하는 코드

Code08-06.py

```
1 ss = input("날짜(연/월/일) 입력 ==> ") 3행 : 입력한 문자열을 /로 분리
2                                     따라서 ssList에 ['2019', '12', '31'] 형식으로 분리
3 ssList = ss.split('/')               6행 : 연도를 가리키는 문자열인 ssList[0](이 코드에서는
4                                     '2019')을 먼저 int() 함수를 사용해 정수로 변환한 후
5                                     10을 더함. 그리고 다시 str() 함수로 문자열로 변경한
6 print("입력한 날짜의 10년 후 ==> ", end = '') '2029'를 '년' 글자와 연결
7 print(str(int(ssList[0]) + 10) + "년", end = '')
8 print(ssList[1] + "월", end = '')
9 print(ssList[2] + "일")
```

출력 결과

날짜(연/월/일) 입력 ==> 2019/12/31

입력한 날짜의 10년 후 ==> 2029년12월31년

Section03 문자열 함수로그람


- 함수명에 대입하기 : map() 함수

```
before = ['2019', '12', '31']  
after = list(map(int, before))  
after
```

출력 결과

```
[2019, 12, 31]
```

함수 자료(list, tuple 등)



리스트에 값을 하나씩 더해서 새로운 리스트를 만드는 작업

```
myList = [1, 2, 3, 4, 5]  
result1 = []  
for val in myList:  
    result1.append(val + 1)  
print(f'result1 : {result1}')
```

map 함수

```
def add_one(n):  
    return n + 1
```

map반환을 list로 변환

```
result2 = list(map(add_one, myList))  
print(f'result2 : {result2}')
```

Section03 문자열 함수로그람

- 문자열 정렬하기, 채우기 : center(), ljust(), rjust(), zfill()

```
ss = '파이썬'  
ss.center(10)  
ss.center(10, '-')  
ss.ljust(10)  
ss.rjust(10)  
ss.zfill(10)
```

출력 결과

```
'  파이썬  '  
'---파이썬----'  
'파이썬      '  
'      파이썬'  
'0000000파이썬'
```

Section03 문자열 함수로그룹

- 문자열 구성 파악하기 : isdigit(), isalpha(), isalnum(), islower(), isupper(), isspace()

```
'1234'.isdigit()
'abcd'.isalpha()
'abc123'.isalnum()
'abcd'.islower()
'ABCD'.isupper()
'   '.isspace()
```

SELF STUDY 8-3

입력한 값이 영어나 한글이면 '글자입니다', 숫자이면 '숫자입니다', 섞여 있으면 '글자+숫자입니다', 특수문자 등이면 '모르겠습니다'가 출력되는 프로그램을 작성해 보자.

출력 결과

문자열 입력 : **abcd123**
글자+숫자입니다.

Section03 문자열 함수로그래밍

■ [프로그램 2]의 완성

- 터틀그래픽에서 문자열 입력, 입력받은 문자열을 한 글자씩 임의의 크기와 색상으로 임의의 위치에 거북이가 쓰는 프로그램

Code08-07.py

```
1 import turtle
2 import random
3 from tkinter.simpledialog import *
4
5 ## 전역 변수 선언 부분 ##
6 inStr = ''
7 swidth, sheight = 300, 300
8 tX, tY, txtSize = [0] * 3
9
10 ## 메인 코드 부분 ##
11 turtle.title('거북이 글자쓰기')
12 turtle.shape('turtle')
13 turtle.setup(width = swidth + 50, height = sheight + 50)
14 turtle.screenize(swidth, sheight)
15 turtle.penup()
16
17 inStr = askstring('문자열 입력', '거북이 쓸 문자열을 입력')
18
19 for ch in inStr :
```

6~8행 : 입력받을 문자열 inStr 및 각 글자의 위치 tX, tY를 준비

15행 : 거북이가 이동할 때 선을 긋지 않도록 함

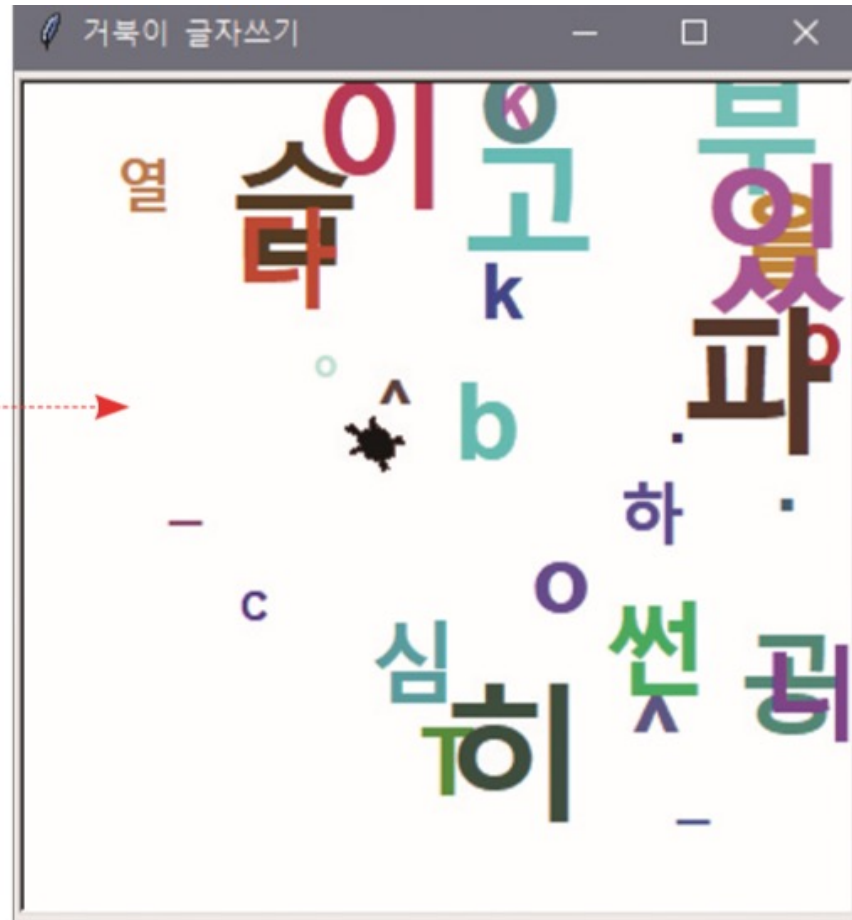
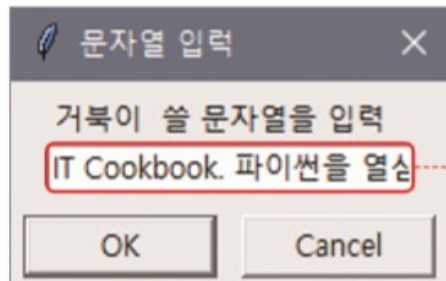
17행 : 문자열을 입력, 17행은 3행에서 tkinter. simpledialog를 임포트 했다면 사용 가능

19행 : 입력받은 문자열에서 한 글자씩 꺼내 ch에 넣고 29행까지 반복

Section03 문자열 함수로그래밍

```
20
21     tX = random.randrange(-swidth / 2, swidth / 2)
22     tY = random.randrange(-sheight / 2, sheight / 2)
23     r = random.random(); g = random.random(); b = random.random()
24     txtSize = random.randrange(10, 50)
25                                     23행 : 글자의 위치 및 색상 크기 랜덤 추출
26     turtle.goto(tX, tY)           26행 : 랜덤한 위치로 거북이가 이동
27                                     28행 : 펜 색상을 지정
28     turtle.pencolor((r, g, b))    29행 : 한 글자를 설정된 크기로 화면에 씀
29     turtle.write(ch, font=('맑은고딕', txtSize, 'bold'))
30
31 turtle.done()
```


Section03 문자열 함수로그그램





Thank You
