

Section01 문자열의 이해

■ 문자열의 개념

- 문자열은 시퀀스 자료형(sequence data type)이다.

a	0100 1001
b	0100 1010
c	0100 1011
d	0100 1100
e	0100 1101

[시퀀스 자료형]

Section01 문자열의 이해

■ 문자열과 메모리 공간

- 일반적으로 문자열을 저장하기 위해서는 영문자 한 글자당 1바이트의 메모리 공간을 사용한다.
- 다음과 같은 코드로 문자열이 저장된 공간의 크기를 확인할 수 있다.

```
>>> import sys # sys 모듈을 호출
>>> print(sys.getsizeof("a"), sys.getsizeof("ab"), sys.getsizeof("abc"))
50 51 52 # "a", "ab", "abc" 각각의 메모리 크기 출력
```

- 컴퓨터에 a라고 알려 줘도 컴퓨터는 정확히 a라는 텍스트를 인식하는 것이 아니다. 대신 컴퓨터는 이 정보를 이진수로 변환하여 저장한다.

Section01 문자열의 이해

■ 문자열과 메모리 공간

- 컴퓨터 공학자들은 이러한 문자를 처리하기 위해 이진수로 변환되는 표준 규칙을 만들었다. ASCII, CP949, MS949, UTF-8 등 이러한 규칙을 인코딩(encoding)이라고 한다.

- ① 컴퓨터는 문자를 직접 인식하지 못한다.
- ② 컴퓨터는 문자를 숫자로 변환하여 인식한다.
- ③ 사람들은 문자를 숫자로 변환하기 위한 규칙을 만들었다.
- ④ 일반적으로 이 규칙은 1개의 영문자를 1바이트, 즉 2의 8승(28) 정도의 공간에 저장될 수 있도록 정하였다.

Section01 문자열의 이해

■ 문자열과 메모리 공간

000 (nul)	016 ▶ (dle)	032 sp	048 0	064 @	080 P	096 `	112 p
001 ☐ (soh)	017 ◀ (dcl)	033 !	049 1	065 A	081 Q	097 a	113 q
002 ● (stx)	018 ∴ (dc2)	034 "	050 2	066 B	082 R	098 b	114 r
003 ▼ (etx)	019 ‡ (dc3)	035 #	051 3	067 C	083 S	099 c	115 s
004 ✦ (eot)	020 ¶ (dc4)	036 \$	052 4	068 D	084 T	100 d	116 t
005 ⬤ (enq)	021 ⌘ (nak)	037 %	053 5	069 E	085 U	101 e	117 u
006 ⬤ (ack)	022 − (syn)	038 &	054 6	070 F	086 V	102 f	118 v
007 ✦ (bel)	023 ‡ (etb)	039 *	055 7	071 G	087 W	103 g	119 w
008 ▣ (bs)	024 † (can)	040 (	056 8	072 H	088 X	104 h	120 x
009 (tab)	025 ⚡ (em)	041 )	057 9	073 I	089 Y	105 i	121 y
010 (lf)	026 eof	042 +	058 :	074 J	090 Z	106 j	122 z
011 ␣ (vt)	027 ⚡ (esc)	043 +	059 ;	075 K	091 [	107 k	123 {
012 ✦ (np)	028 ⚡ (fs)	044 ,	060 <	076 L	092 \	108 l	124
013 (cr)	029 ⇄ (gs)	045 -	061 =	077 M	093 ]	109 m	125 }
014 ⬤ (so)	030 ⬤ (rs)	046 .	062 >	078 N	094 ^	110 n	126 ~
015 ☐ (si)	031 ▼ (us)	047 /	063 ?	079 O	095 _	111 o	127 o

[UTF-8의 유니코드(출처: Nicolas Bouliaue)]

Section01 문자열의 이해

- 문자열의 인덱싱
  - 리스트처럼 글자 하나하나가 상대적인 주소(offset)를 가지는데, 이 주소를 사용해 할당된 값을 가져오는 인덱싱을 사용할 수 있다.

Hello

0 1 2 3 4

-5 -4 -3 -2 -1

[문자열의 처리]

```
>>> a = "abcde"
>>> print(a[0], a[4])      # a 변수의 0번째, 4번째 주소에 있는 값
a e
>>> print(a[-1], a[-5])   # a 변수의 오른쪽에서 0번째, 4번째 주소에 있는 값
e a
```

Section02 문자열 기본

- 문자열의 개념
  - 리스트 코드와 비교 : 리스트는 대괄호 []로 묶고 문자열은 작은따옴표로 묶어 출력

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

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                                     3행 : 문자열의 길이를 sslen 변수에 저장
3 sslen = len(ss)                   4행 : 문자열의 길이만큼 반복
4 for i in range(0, sslen) :       5행 : 글자 하나와 $를 붙여서 출력
5     print(ss[i] + '$', end = '')
```

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.

Section03 문자열 함수

- 문자열의 연산

함수명	기능
len()	문자열의 문자 개수를 반환
upper()	대문자로 변환
lower()	소문자로 변환
title()	각 단어의 앞글자만 대문자로 변환
capitalize()	첫 문자를 대문자로 변환
count('찾을 문자열')	'찾을 문자열'이 몇 개 들어 있는지 개수 반환
find('찾을 문자열')	'찾을 문자열'이 왼쪽 끝부터 시작하여 몇 번째에 있는지 반환
rfind('찾을 문자열')	find() 함수와 반대로 '찾을 문자열'이 오른쪽 끝부터 시작하여 몇 번째에 있는지 반환
startswith('찾을 문자열')	'찾을 문자열'로 시작하는지 여부 반환
endswith('찾을 문자열')	'찾을 문자열'로 끝나는지 여부 반환

Section03 문자열 함수

- 문자열의 연산

함수명	기능
strip()	좌우 공백 삭제
rstrip()	오른쪽 공백 삭제
lstrip()	왼쪽 공백 삭제
split()	문자열을 공백이나 다른 문자로 나누어 리스트로 반환
isdigit()	문자열이 숫자인지 여부 반환
islower()	문자열이 소문자인지 여부 반환
isupper()	문자열이 대문자인지 여부 반환

Section03 문자열 함수

- 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 함수

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()
```

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

Section03 문자열 함수

- 문자열 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
```

Section03 문자열 함수

- 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 함수

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

```
ss = ' 파 이 썬 '
```

```
ss.strip()
ss.rstrip()
ss.lstrip()
```

출력 결과

```
'파 이 썬'
' 파 이 썬'
'파 이 썬 '
```

Section03 문자열 함수

- 앞뒤의 특정 문자 삭제

```
ss = '----파---0|---썩----'
print(ss.strip('-'))
ss = '<<<파 << 이 >> 썩>>>'
print(ss.strip('<'))
```

출력 결과

파---0|---썩

파 << 이 >> 썩

Section03 문자열 함수

- 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<<0>>>썩>>>)'이 파이썬으로 출력되도록 해 보자.

**힌트** if 문이 '<'이 아닐 때와 '>'이 아닐 때를 and로 연결해야 한다.

Section03 문자열 함수

- 문자열 변경

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

출력 결과

'열심히 Python 공부 중~'

Section03 문자열 함수

- 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 = '')
```

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 함수

- 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']
'파이썬'
```

Section03 문자열 함수

- 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'를 먼저 int() 함수를 사용해 정수로 변환한 후
5 print("입력한 날짜의 10년 후 ==> ", end = '') '2029'를 '년' 글자와 연결
6 print(str(int(ssList[0]) + 10) + "년", end = '')
7 print(ssList[1] + "월", end = '')
8 print(ssList[2] + "일")
```

출력 결과

```
날짜(연/월/일) 입력 ==> 2019/12/31
입력한 날짜의 10년 후 ==> 2029년12월31년
```

Section03 문자열 함수

- 함수명에 대입하기 : map() 함수

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

출력 결과

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

Section03 문자열 함수

- 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'
'00000000파이썬'
```

Section03 문자열 함수

- 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.