문자열 함수

- 문자열 처리를 위한 다양한 함수 제공

함수호출 : 문자열.함수명()

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

문자열 함수

- 문자열 처리를 위한 다양한 함수 제공

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

문자열 함수

- len() : 문자열 길이, str() : 문자열 변환
- split() : 문자열 분리, join() : 문자열 결합

```
s = 'i have a dream'
n=100
s100 = s + str(n)  #100을 문자열로 변환하여 결합
print('s=', s)
print('s100=', s100)
print('len(s)=', len(s))      #s문자열의 길이
print('len(s100)=',len(s100))
c1 = s.count(' ')  #s의 빈공백 개수 반환
c2 = s.count('e')  #s의 'e'문자 개수 반환
print('s.count(' ')=', c1)
print('s.count('e')=', c2)
slist = s.split(' ') #s문자열을 빈공백으로 분리하여 리스트에 대입
print(slist)
sjoin = ",".join(slist) #slist를 콤마로 결합
print(sjoin)
slist2 = sjoin.split(',')  #s를 콤마로 분리
print(slist2)
```

```
s= i have a dream
s100= i have a dream100
len(s)= 14
len(s100)= 17
s.count(' ')= 3
s.count('e')= 2
['i', 'have', 'a', 'dream']
i,have,a,dream
['i', 'have', 'a', 'dream']
```

문자열 함수

- find() 문자의 위치를 찾아서 반환, 찾는 문자나 문자열이 존재하지 않는다면 -1을 반환
- rfind() 문자의 위치를 오른쪽에서부터 찾아서 반환
- index() 문자의 위치를 찾아서 반환, 찾는 문자나 문자열이 존재하지 않는다면 오류발생
- upper() 대문자로 변환, title() 단어 첫 글자만 대문자로 변환
- startswith() 특정 단어로 시작하는지 확인, endswith() 특정 단어로 끝나는지 확인

```
s = 'i have a dream'
f1 = s.find('a')        #'a'문자의 위치를 찾아서 반환
f2 = s.rfind('a')       #'a'문자의 위치를 오른쪽에서부터 찾아서 반환
i = s.index('a')
print("s.find('a')=", f1)
print("s.rfind('a')=", f2)
print("s.index('a')=", i)
print("s.upper()=", s.upper())    #대문자로 변환
print("s.title()=", s.title())    #단어 첫들자만 대문자로 변환
print("s.startswith(₩"dream₩")=", s.startswith("dream"))    #'dream'으로 시작하는지 확인
print("s.endswith(₩"dream₩")=", s.endswith("dream"))        #'dream'으로 끝나는지 확인
```

```
s.find('a')= 3
s.rfind('a')= 12
s.index('a')= 3
s.upper()= I HAVE A DREAM
s.title()= I Have A Dream
s.startswith("dream")= False
s.endswith("dream")= True
```

문자열 함수

예제2. 문자를 반복 입력하여 숫자끼리, 소문자 끼리 결합

```
#문자를 입력받아 숫자이면 숫자변수에 결합 소문자이면 문자변수에 결합하여 출력
#반복입력 q이면 종료
n = ""
c =""
while True:
    ch = input("문자 입력 ? ")
    if (ch == 'q'):
        print("종료합니다.")
        break
    if (ch.isdigit()):      #숫자인지 확인
        n += ch
    elif (ch.islower()):    #소문자인지 확인
        c += ch

print("n=", n)
print("c=", c)
```

```
문자 입력 ? a
문자 입력 ? b
문자 입력 ? c
문자 입력 ? D
문자 입력 ? E
문자 입력 ? 1
문자 입력 ? 2
문자 입력 ? 3
문자 입력 ? 4
문자 입력 ? 5
문자 입력 ? q
종료합니다.
n= 12345
c= abc
```

문자열 함수

예제3. '홍'을 찾아서 '김'으로 변환한 문자열

```
#문자열 에서 문자를 찾아서 해당위치를 기준으로 분리하여 새로운 글자를 결합
name = "화이팅 홍길동"
i = name.find('홍')
name = name[:i] + '김' + name[i+1:]
print(name)

#replace()함수를 사용하여 문자열 변환
name = name.replace('김', '홍')
print(name)

#문자 패턴을 생성하여 패턴에 따라 변환,
#1은 a로, 2는 b로 ... 5는 e로 변환하도록 패턴생성
table = str.maketrans('12345', 'abcde')
s = '15apple'
s = s.translate(table)   #s문자열에서 table 패턴에 속하는 문자를 변환
print(s)
```

```
화이팅 김길동
화이팅 홍길동
aeapple
```

문자열 함수

- strip() 좌우 공백 제거
- rstrip() 우 공백 제거
- lstrip() 좌 공백 제거

```
name = "  홍길동  "
print("(", name, ")")
print ("strip() =", "(", name.strip() , ")")
print ("rstrip()=","(",  name.rstrip() , ")")
print ("lstrip()=", "(", name.lstrip(), ")")
```

```
(   홍길동   )
strip() = ( 홍길동 )
rstrip()= (   홍길동 )
lstrip()= ( 홍길동   )
```

문자열 함수

예제4. 문장을 입력하고, 새로운 단어를 입력하여 문장내 검색, 문장을 대문자로 변환하여 단어 리스트에 저장, 단어의 수를 출력하는 프로그램

```
# 문장 입력
s = input("문장 입력 ? ")
# 찾는 단어
while True:
    w = input("단어 입력 ? ")
    if (w == 'q'):
        print("종료합니다.")
        break
    if (w in s):   #입력 단어가 문장에 포함여부 확인
        print("단어 " + w + "는 문장에 포함되어 있습니다.")
    else:
        print("단어 " + w + "는 문장에 포함되어 있지 않습니다.")

#대문자로 변환
s = s.upper()

#문장에서 단어를 분리하여 리스트에 저장
words = s.split(" ")
print("words =", words)
print("단어 개수  =", len(words))
```

```
문장 입력 ? i have a dream
단어 입력 ? dream
단어 dream는 문장에 포함되어 있습니다.
단어 입력 ? have
단어 have는 문장에 포함되어 있습니다.
단어 입력 ? sky
단어 sky는 문장에 포함되어 있지 않습니다.
단어 입력 ? q
종료합니다.
words = ['I', 'HAVE', 'A', 'DREAM']
단어 개수  = 4
```

문자열 함수

예제 5. 문자 패턴을 생성하여 암호화 하는 프로그램 작성

<암호화 처리 조건>
(1)'abcdef'는 '!@#$%&"패턴으로 변환
(2)문자를 뒤집기

```
#문자 패턴을 생성하여 암호화 하는 프로그램 작성
x = 'abcdef'
y = '!@#$%&'
code = str.maketrans(x, y)      #암호화 패턴
decode = str.maketrans(y, x)    #복호화 패턴

#문장 입력
s = 'i have a dream'
print("s=", s)

#(1)암호화 처리
cod = s.translate(code)    #code 패턴으로 암호화
cod = cod[::-1]            #문자열 뒤집기
print("code = ", cod)
#(2)복호화 처리
cod = cod[::-1]            #문자열 뒤집기
dcod = cod.translate(decode)    #decode 패턴으로 복호화
print("decode = ", dcod)
```

```
s= i have a dream
code =  m!%r$ ! %v!h i
decode =  i have a dream
```

중간시험 일정 안내

과목명 : 데이터처리프로그래밍

시험일시 : 2021년 4월 21일 수요일 10:00 ~ 12:00

시험장소 : 한빛관 311

시험범위 및 문제 유형 :

1. 이론 문제
컴퓨터동작원리, 수의체계
변수, 자료형, 자료형 변환
리스트
조건문
반복문
함수
문자열

2. 프로그래밍 문제
예제, report 유사문제