String Overview

String

최성철 교수
Director of TEAMLAB

문자열 (string)

- 시퀀스 자료형으로 문자형 data를 메모리에 저장

- 영문자 한 글자는 1byte의 메모리공간을 사용

Python shell

```
>>> import sys                          # sys 모듈을 호출
>>> print (sys.getsizeof("a"), sys.getsizeof("ab"), sys.getsizeof("abc"))
50 51 52
# “a”, “ab”, “abc”의 각 메모리 사이즈 출력
```

문자열 (string)

- string은 1byte 크기로 한 글자씩 메모리 공간이 할당됨

```
a = “abcde”
```

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

1Byte의 메모리 공간???

- 컴퓨터는 2진수로 데이터를 저장

- 이진수 한 자릿수는 1bit로 저장됨

- 즉 1bit 는 0 또는 1

- 1 byte = 8 bit = 2^8 = 256 까지 저장 가능

Source: http://goo.gl/xr4Az2

1Byte의 메모리 공간???

- 컴퓨터는 문자를 직접적으로 인식 X
  ➔ 모든 데이터는 2진수로 인식

- 이를 위해 2진수를 문자로 변환하는 표준 규칙을 정함

- 이러한 규칙에 따라 문자를 2진수로 변환하여 저장하거나
  저장된 2진수를 숫자로 변환하여 표시함

- 예) 대문자 U는 이진수로 “1000011” 변환됨 (UTF-8기준)

Source: http://goo.gl/xr4Az2

1Byte의 메모리 공간???

000		(nul)	016	►	(dle)	032	sp	048	0	064	@	080	P	096	`	112	p
001	☺	(soh)	017	◄	(dc1)	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	⌂

http://goo.gl/K7Uz38

프로그램 언어에서 데이터 타입

- 각 타입 별로 메모리 공간을 할당 받은 크기가 다름

종류	타입	크기	표현 범위 (32bit)
정수형	int	4바이트	$-2^{31} \sim 2^{31}-1$
	long	무제한	무제한
실수형	float	8바이트	약 $10^{-308} \sim 10^{+308}$

http://goo.gl/JSI4ZP

- 메모리 공간에 따라 표현할 수 있는 숫자범위가 다름

예) 4byte = 32bit = 2^{32} = 4,294M = -2,147M ~ + 2.147M 까지 표시

- 데이터 타입은 메모리의 효율적 활용을 위해 매우 중요

문자열의 특징

인덱싱 (Indexing)

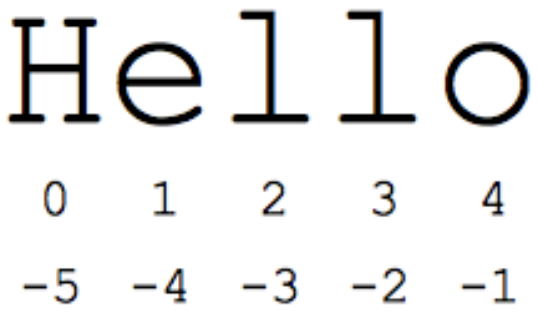
- 문자열의 각 문자는 개별 주소(offset)를 가짐
- 이 주소를 사용해 할당된 값을 가져오는 것이 인덱싱
- List와 같은 형태로 데이터를 처리함

Python shell

```
>>> 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
```

Source: http://goo.gl/XjPSpP

인덱싱 (Indexing)



각 문자의 오프셋은

왼쪽에선 0부터 오른쪽에선 -1부터 시작함

Python shell

```
>>> 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
```

Source: http://goo.gl/XjPSpP

슬라이싱 (Slicing)

- 문자열의 주소값을 기반으로 문자열의 부분값을 반환

Python shell

```
>>> print (a[0:6], " AND ", a[-9:]) # a 변수의 0부터 5까지, -9부터 끝까지
Gachon  AND  University
>>> print (a[:])  # a변수의 처음부터 끝까지
Gachon University
>>> print (a[-50:50])  # 범위를 넘어갈 경우 자동으로 최대 범위를 지정
Gachon University
>>> print (a[::2], " AND ", a[::-1])  # 2칸 단위로, 역으로 슬라이싱
Gco nest  AND  ytisrevnU nohcaG
```

문자열 연산 및 포함여부 검사

- 덧셈과 뺄셈 연산 가능, in 명령으로 포함여부 체크

Python shell

```
>>> a = "TEAM"
>>> b = "LAB"
>>> print (a + " " + b)                # 덧셈으로 a와 b 변수 연결하기
TEAM LAB
>>> print (a * 2 + " " + b * 2)
TEAMTEAM LABLAB          # 곱하기로 반복 연산 가능
>>> if 'A' in a: # 'U'가 a에 포함되었는지 확인
...     print (a)
... else:
...     print (b)
...
TEAM
```

문자열 함수 (1/2)

함수명	기능
len(a)	문자열의 문자 개수를 반환
a.upper()	대문자로 변환
a.lower()	소문자로 변환
a.capitalize()	첫 문자를 대문자로 변환
a.titile()	제목형태로 변환 띄워쓰기 후 첫 글자만 대문자
a.count('abc')	문자열 a에 'abc'가 들어간 횟수 반환
a.find('abc') a.rfind('abc')	문자열 a에 'abc'가 들어간 위치(오프셋) 반환
a.startswith('ab'c)	문자열 a는 'abc'로 시작하는 문자열여부 반환
a.endswith('abc')	문자열 a는 'abc'로 끝나는 문자열여부 반환

문자열 함수 (2/2)

함수명	기능
a.strip()	좌우 공백을 없앰
a.rstrip()	오른쪽 공백을 없앰
a.lstrip()	왼쪽 공백을 없앰
a.split()	공백을 기준으로 나눠 리스트로 반환
a.split('abc')	abc를 기준으로 나눠 리스트로 반환
a.isdigit()	문자열이 숫자인지 여부 반환
a.islower()	문자열이 소문자인지 여부 반환
a.isupper()	문자열이 대문자인지 여부 반환

[예제] 문자열 함수

```
>>> title = "TEAMLAB X Inflearn"
>>> title.upper()
TEAMLAB X INFLEARN'
>>> title.lower()
'teamlab x inflearn'
>>> title.split(" ")
['TEAMLAB', 'X', 'Inflearn']
>>> title.isdigit()
False
>>> title.title()
'Teamlab X Inflearn'
>>> title.startswith("a")
False
>>> title.count("a")
1
>>> title.upper().count("a")
0
```

Python shell

```
>>> "12345".isdigit()
True
>>> title.find("Gachon")
0
>>> title.upper().find("Gachon")
-1
>>> "      Hello      ".strip()
'Hello'
>>> "A-B-C-D-E-F".split("-")
['A', 'B', 'C', 'D', 'E', 'F']
```

다양한 문자열 표현

문자열 선언은 큰따옴표(“”)나 작은 따옴표 (‘’)를 활용

It’s OK 이라는 문자열은 어떻게 표현할까?

① a = ‘It₩’ ok.’

₩’는 문자열 구분자가 아닌 출력 문자로 처리

② a = “It’s ok.” #

큰따옴표로 문자열 선언 후 작은 따옴표는 출력 문자로 사용

다양한 문자열 표현

두줄 이상은 어떻게 저장할까?

① ₩n # ₩n은 줄바꿈을 의미하는 특수 문자

② 큰따옴표 또는 작은 따옴표 세 번 연속 사용

예) a= """ It'Ok

I'm Happy.

See you. """

특수 문자

문자열을 표시할 때 백슬래시 “₩” 를 사용하여
키보드로 표시하기 어려운 문자들을 표현함

문자	설명	문자	설명
₩ [Enter]	다음 줄과 연속임을 표현	₩n	줄 바꾸기
₩₩	₩ 문자 자체	₩t	TAB 키
₩`	` 문자	₩e	ESC 키
₩"	" 문자		
₩b	백 스페이스		

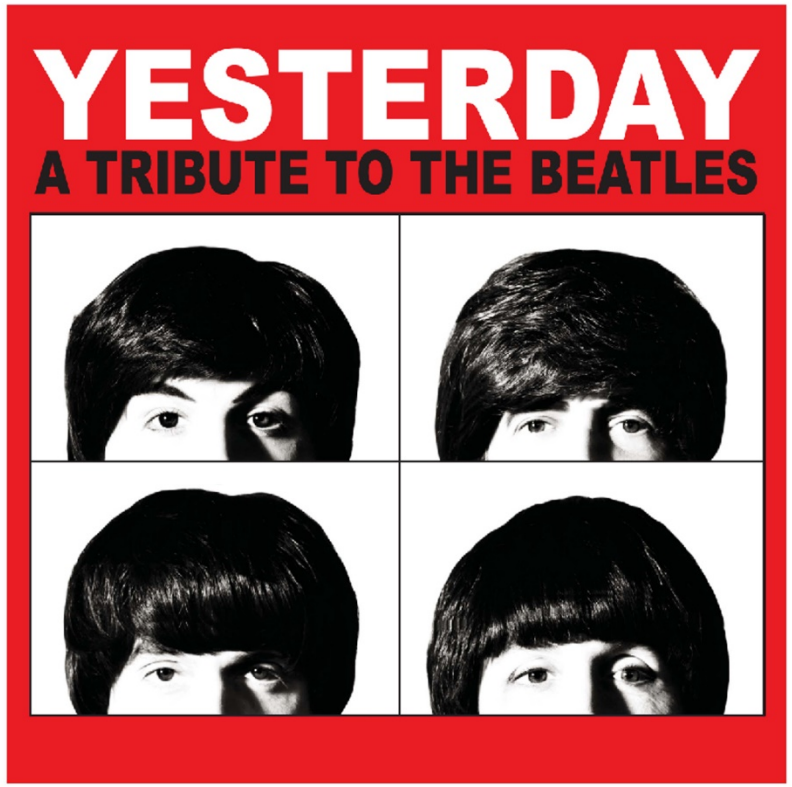
Human knowledge belongs to the world.

Lab: Yesterday Counter

String

최성철 교수
Director of TEAMLAB

Yesterday



Yesterday 노래엔

Yesterday 라는 말이 몇 번 나올까?

http://goo.gl/UqNp1r

Lab: Yesterday Counter

“Yesterday” 노래에 “Yesterday” 단어의 개수?

CMD Window

```
wget https://raw.githubusercontent.com/TeamLab/cs50_example_code/master/12_string/yesterday.txt
```

Python shell

```
f = open("yesterday.txt", 'r')
yesterday_lyric = ""
while 1:
        line = f.readline()
        if not line:
                break
        yesterday_lyric = yesterday_lyric + line.strip() + "\n"
f.close()
n_of_yesterday = yesterday_lyric.upper().count("YESTERDAY")
print ("Number of a Word 'Yesterday'" , n_of_yesterday)
```

yeseterday_counter.py

Lab: Yesterday Counter II

대소문자를 구분하여 “Yesterday”와 “yesterday”의

개수를 나눠서 세는 프로그램을 작성하세요.

Human knowledge belongs to the world.