CS101 – 자료 구조: 문자열과 집합

Lecture 17

School of Computing
KAIST

학습 목표:

- 문자열을 다양한 방법으로 활용할 수 있다.
- 집합 (set) 자료구조를 활용할 수 있다.

문자열 서식화

문자열을 출력할 때 여러 변수의 값을 이용해야 하는 경우가 있습니다.

```
print("Max between " + str(x0) + " and " +
      str(x1) + " is " + str(val))
```

문자열 포맷 연산자 %를 사용하면 더 쉽고 간단하게 문자열을 출력할 수 있습니다.

```
print("Max between %d and %d is %g" % (x0, x1, val))
```

포맷 연산자는 다음처럼 사용합니다.

```
format_string % (arg0, arg1, .... )
```

포맷 연산자가 사용하는 튜플의 원소들이 `format_string` 안의 문자열 포맷코드들의 값에 일대일로 지정됩니다. 자료형에 따라 다른 문자열 포맷코드를 사용합니다.

- `%d`: 10진법 정수
- `%g`: 실수
- `%.2f`: 소수점 자리수가 설정된 실수 (이 경우는 소숫점 둘째 자리까지)
- `%s`: 모든 자료형 (문자열 등)

문자열 서식화

하나의 문자열 포맷 코드만 사용하는 데는 튜플을 쓰지 않아도 됩니다.

```
print("Maximum is %g" % val)
```

각 문자열 포맷 코드가 나타낼 문자열이 차지하는 영역의 크기를 설정할 수 있습니다.

```
>>>(x0,x1,x2) = (1,2,3)
>>> print("%3d~%3d:%10g" % (x0, x1, x2))
  1~  2:         3
```

다음처럼 문자열 포맷 코드가 나타낼 문자열을 왼쪽으로 정렬시킬 수도 있습니다.

```
>>> print("%-3d~%-3d:%-12g" % (x0, x1, x2))
1  ~2  :3
```

문자열

문자열은 시퀀스입니다.

```
def is_palindrome(s):
  for i in range(len(s) // 2):
    if s[i] != s[len(s) - i - 1]:
      return False
  return True
```

문자열은 불변 객체입니다.

문자열에는 `in` 연산자를 사용할 수 있습니다.

```
>>> "abc" in "01234abcdefg"
True
>>> "abce" in "01234abcdefg"
False
```

리스트와 튜플에서는 `in` 연산자가 리스트/튜플의 원소를 대상으로 하지만,
문자열에서는 부분 문자열을 대상으로 합니다.

문자열의 멤버 함수

문자열 객체에서는 다음과 같은 멤버 함수를 사용할 수 있습니다.

- `upper(), lower(), capitalize()`
- `isalpha(), isdigit()`
- `startswith(prefix), endswith(suffix)`
- `find(str1), find(str1, start),`
  `find(str1, start, end)`
- `replace(str1, str2)`
- `rstrip(), lstrip(), strip()`
- `split(), split(sep)`
- `join(list1)`

모든 멤버 함수는 Python에서 제공하는 문서에도 설명되어 있습니다.

집합

집합(Set)은 수학의 집합과 관련된 자료들을 쉽게 처리하기 위한 자료 구조입니다.
집합은 서로 다른 여러 객체들로 이루어져 있습니다 (중복된 자료가 없습니다).
집합은 중괄호 { } 나 `set()` 함수를 이용해서 만들 수 있습니다.

```
>>> odds = {1, 3, 5, 7, 9}
>>> evens = {2, 4, 6, 8, 10}
>>> emptyset = set() # {} creates an empty dictionary
>>> randomset = {4, 6, 2, 7, 5, 2, 3} # Duplicated ele.

>>> odds
{9, 3, 5, 1, 7}
>>> evens
{8, 10, 2, 4, 6}
>>> emptyset
set()
>>> randomset
{2, 3, 4, 5, 6, 7}
```

집합

리스트는 집합으로 변환할 수 있습니다.

```
>>> gold = [0, 4, 5, 10, 3, 0, 2, 1, 4, 8, 1, 0, 1,
    0, 0, 8, 11, 4, 13, 1, 2, 3, 2, 6, 1, 9]
>>> gold
[0, 4, 5, 10, 3, 0, 2, 1, 4, 8, 1, 0, 1,
 0, 0, 8, 11, 4, 13, 1, 2, 3, 2, 6, 1, 9]
>>> goldset = set(gold)
>>> goldset
{0, 1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 13}
>>> type(goldset)
<class 'set'>
```

문자열 또한 집합으로 변환할 수 있습니다.

```
>>> set("Good morning!")
{'G', 'm', 'i', 'd', 'o', '!', 'g', 'n', 'r', ' '}
```

집합

집합의 원소에는 순서가 없기 때문에 원소의 위치를 특정할 수 없습니다.

```
>>> odds[1]
TypeError: 'set' object does not support indexing
```

하지만, `in` 연산자는 사용할 수 있습니다.

```
>>> 3 in odds
True
>>> 2 in odds
False
>>> for num in odds:
...     print(num)
9
3
5
1
7
```

집합의 멤버 함수

집합 객체 s에는 다음과 같은 멤버 함수들을 사용할 수 있습니다.

- `s.add(v)`: 원소 v를 집합 s에 추가한다.
- `s.remove(v)`: 원소 v를 집합 s에서 제거한다.
- `s.pop()`: 무작위 원소를 집합 s에서 제거하고 그 원소를 반환한다.
- `s.intersection(k)`: 집합 s, k의 공통 원소를 반환한다. ($s \cap k$)
- `s.union(k)`: 집합 s, k의 합집합을 반환한다. ($s \cup k$)
- `s.difference(k)`: 집합 k에 있는 원소들을 s에서 제거한다. ($s \cap k^c$)

집합의 멤버 함수

다음은 멤버 함수들의 사용 예시입니다.

```
>>> randomset
{2, 3, 4, 5, 6, 7}
>>> randomset.add(9)
>>> randomset
{2, 3, 4, 5, 6, 7, 9}
>>> randomset.remove(7)
>>> randomset
{2, 3, 4, 5, 6, 9}
>>> randomset.pop()
2
>>> randomset
{3, 4, 5, 6, 9}
```

집합의 멤버 함수

다음은 멤버 함수들의 사용 예시입니다.

```
>>> randomset
{3, 4, 5, 6, 9}
>>> randomset.intersection(odds)
{9, 3, 5}
>>> randomset.union(evens)
{2, 3, 4, 5, 6, 8, 9, 10}
>>> randomset.difference(odds)
{4, 6}
>>> odds.difference(randomset)
{1, 7}
>>> randomset.difference(odds, evens)
set()
```

정리 및 예습

본 강의 학습 목표:

- 문자열을 다양한 방법으로 활용할 수 있다.
- 집합 (set) 자료구조를 활용할 수 있다.

다음 강의 학습 목표:

- 사전 (dictionary) 자료구조를 활용할 수 있다.
- 리스트, 집합, 사전의 장단점을 이해하여
  목적에 적합한 자료구조를 사용할 수 있다.