LAB 07

2020-01 소프트웨어의이해 01분반 / 조교 이경민

LAB 07

- 리스트
- 리스트 인덱스
- 리스트 생성/항목 추가/항목 삭제
- 리스트 멤버십
- 예제
- 실습 과제

리스트(list)

- 항목 item 들의 모음
  - 대괄호[]를 이용하여 항목들을 정의
  - color = ['blue', 'black', 'green', 'yellow']
  - primes = [2, 3, 5, 7, 11, 13, 17]
  - **자료형이 동일할 필요가 없어, 다양한 자료형의 item을 담을 수 있음**
  - 리스트 자체를 원소로 추가할 수도 있음

cf) array(배열)
: 같은 type만 넣을 수 있음.

```
>>> phoneBook = ['Kim', 'Lee', 1, 2]
>>> print(phoneBook)
['Kim', 'Lee', 1, 2]
>>> numbers = [1, 2, 3]
>>> week = ['sun', 'mon', 'tue']
>>> numbers.append(week)
>>> print(numbers)
[1, 2, 3, ['sun', 'mon', 'tue']]
```

이처럼 리스트

리스트의 생성

- 빈 리스트를 생성한 후 항목을 하나씩 추가 (append)

null list

여러개 extend
인덱스위치에 추가하는거 insert

```
>>> phoneBook = []
>>> phoneBook.append('Lee')
>>> phoneBook.append('Kim')
>>> print(phoneBook)
['Lee', 'Kim']
```

- 리스트 선언 시 항목으로 초기화

```
>>> buylist = ['milk', 'apple', 'banana']
>>> print(buylist)
['milk', 'apple', 'banana']
>>> buylist.append('noodle')
>>> buylist
['milk', 'apple', 'banana', 'noodle']
```

리스트의 인덱스

- 0부터 시작하여 1씩 증가
- list[0] : 리스트의 첫 번째 항목, list[1] : 두 번째 항목
- 2차원 리스트의 경우 : list[][]

```
>>> temp = [2, 3, 4, 5, 7, 11, ['sun', 'mon', 'tue']]
>>> temp[0]
2
>>> temp[5]
11
>>> temp[6]
['sun', 'mon', 'tue']
>>> temp[6][1]
'mon'
```

```
>>> scores = [[50, 60, 70], [75, 85 ,95], [100, 65]]
>>> scores[1]
[75, 85, 95]
>>> scores[1][2]
95
>>> scores[2]
[100, 65]
>>> scores[2][0]
100
```

리스트의 항목 추가

- append
  - 항목 값을 직접 써서 리스트의 오른쪽 끝에 삽입하는 방식
  - 한 번에 하나의 항목만 추가할 수 있음

```
>>> word = []
>>> word.append('I')
>>> word
['I']
```

- extend
  - 리스트에 여러 개 항목을 추가하여 확장

```
>>> word.extend(['love', 'programming'])
>>> word
['I', 'love', 'programming']
```

- insert
  - 항목이 삽입될 위치를 인덱스로 지정하여 새로운 항목을 삽입하고 다음 항목들이 오른쪽으로 이동

```
>>> word.insert(2, 'python')
>>> word
['I', 'love', 'python', 'programming']
```

리스트의 항목 삭제

- 항목의 이름으로 삭제
  - remove(항목 이름)

```
>>> word
['I', 'love', 'python', 'programming']
>>> word.remove('python')
>>> word
['I', 'love', 'programming']
```

- 항목 인덱스(index)로 삭제
  - del 리스트이름[index]

```
>>> del word[0]
>>> word
['love', 'programming']
```

- 가장 나중에 추가된 항목 삭제
  - pop()
  - 특이점은 그냥 삭제가 아니라, 값을 반환하고 삭제됨

```
>>> one = word.pop()
>>> one
'programming'
>>> word
['love']
```

리스트의 멤버십

- 리스트에 특정 항목이 존재하는지 여부를 파악
- **in, not in**

```
>>> week = ['sun', 'mon', 'tue']
>>> 'sun' in week
True
>>> 'wed' in week
False
>>> 'tue' not in week
False
```

리스트 자르기(slicing)

- 하나의 리스트를 인덱스를 사용하여 여러 개의 부분 리스트로 분리
- **1) list[start:end]**
  - start에서 end-1번째까지의 item으로 구성된 부분 리스트

```
>>> week = ['Sun', 'Mon', 'Tue', 'Wed', 'Thur']
>>> days = week[1:4]
>>> days
['Mon', 'Tue', 'Wed']
>>> week[1:2]
['Mon']
>>> week[1:1]
[]
```

리스트 자르기(slicing)

- **2) list[:end]**
  - 맨 앞(= 가장 왼쪽) item 부터 end-1번째 item을 포함하는 리스트

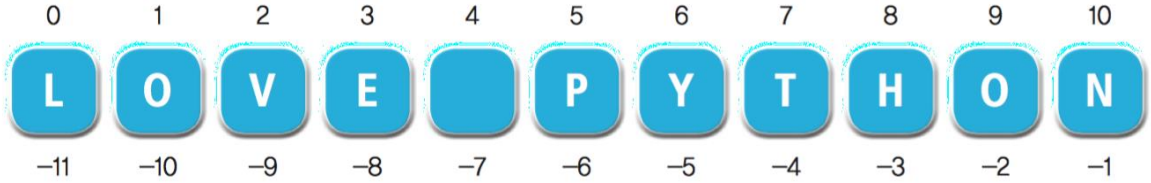
```
>>> week
['Sun', 'Mon', 'Tue', 'Wed', 'Thur']
>>> days = week[:3]
>>> days
['Sun', 'Mon', 'Tue']
```

- **3) list[start:]**
  - start번째 item부터 맨 끝 item까지 포함하는 리스트
  - start와 end가 모두 없으면, 전체 리스트를 반환

```
>>> week[3:]
['Wed', 'Thur']
>>> week[:]
['Sun', 'Mon', 'Tue', 'Wed', 'Thur']
```

리스트 자르기(slicing)

- **음수 인덱스**
  - 오른쪽 끝부터 셀 때 -1부터 1씩 감소




```
>>> week = ['Sun', 'Mon', 'Tue', 'Wed', 'Thur']
>>> week[-4:-1]
['Mon', 'Tue', 'Wed']
>>> week[:-1]
['Sun', 'Mon', 'Tue', 'Wed']
>>> week[-3:]
['Tue', 'Wed', 'Thur']
>>> week[-1:0]
[]
```

리스트 자르기(slicing)

- **list[start:end:step]**
  - step (x) : 1로 간주되어 왼쪽에서 오른쪽 방향으로 순차 탐색
  - step = 2 : 리스트의 항목을 1개씩 건너뛰면서 탐색
  - step = -1 : 오른쪽에서 왼쪽으로 **역방향** 탐색

```
>>> week
['Sun', 'Mon', 'Tue', 'Wed', 'Thur']
>>> week[::2]
['Sun', 'Tue', 'Thur']
>>> week[0:3:3]
['Sun']
>>> week[-1:-4:-1]
['Thur', 'Wed', 'Tue']
>>> week[-1::-2]
['Thur', 'Tue', 'Sun']
>>>
```

문자열 자르기

- 문자열 = 문자들의 리스트
  - 리스트에서 사용하는 연산을 그대로 적용 가능

```
>>> string = 'software'
>>> string[2:3]
'f'
>>> string[3:]
'tware'
>>> string[-4:-1]
'war'
>>> string[-1:-5:-1]
'eraw'
>>> string[:]
'software'
>>> string[::-1]
'erawtfos'
>>>
```

예제) 소수 찾기 1

- **각 숫자(n)를 2부터 n-1까지의 수로 나누어서 떨어지지 않는 수를 소수 리스트에 추가**

- **소수를 탐색하는데 걸리는 시간을 측정**
- **time 함수** : 시간을 측정하는데 사용
  - import time
  - 1970년 1월 1일부터 현재까지의 경과 시간을 초로 환산하여 알려줌
- 프로그램 수행 시간
  - 프로그램 앞부분에서 time 함수 호출
  - 프로그램 완료 후 time 함수 호출
  - 두 값을 빼기

```
import time
t1 = time.time()
 ...
t2 = time.time()
print(t2 - t1)
```

예제) 소수 찾기 1

```
import time

upper = 1000

t1 = time.time()
num=[]

for i in range(2, upper+1) :
    prime = True

    if i == 2 :
        prime = True
    else:
        for j in range(2, i) :
            if i % j == 0 :
                prime = False
                break

    if prime == True :
        num.append(i)

t2 = time.time()
print("₩nElapsed time = ", t2-t1)
print("# of primes = ", len(num))
print("==primes==")
print(num)
```

```
Elapsed time =  0.03358936309814453
# of primes =  168
==primes==
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 10
3, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199,
211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 31
7, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439,
443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569, 57
1, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683,
691, 701, 709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 82
7, 829, 839, 853, 857, 859, 863, 877, 881, 883, 887, 907, 911, 919, 929, 937, 941, 947, 953, 967,
971, 977, 983, 991, 997]
```

예제) 소수 찾기 2

- **2에서 n까지의 수의 배수를 리스트에서 제거하여 탐색**
  - 2부터 n까지의 수를 모두 리스트에 추가
  - 이 리스트의 가장 작은 수 s의 배수 s*i (i=2, 3, 4, …, k)를 계산하여 리스트에서 제거
  - s*i가 n보다 크면 중단하고, 리스트에 남아 있는 s 다음 수에 대하여 이 과정을 반복

예제) 소수 찾기 2

```
import time

upper = 1000

t1 = time.time()
num = []

for i in range(2, upper+1) :
    num.append(i)

for i in num :
    for j in range(2, upper+1) :
        s = i*j
        if s > upper :
            break
        if s in num :
            num.remove(s)

t2 = time.time()
print("₩nElapsed time = ", t2-t1)
print("# of primes = ", len(num))
print("==primes==")
print(num)
```

```
Elapsed time =  0.03179764747619629
# of primes =  168
==primes==
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 10
3, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199,
211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 31
7, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439,
443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569, 57
1, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683,
691, 701, 709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 82
7, 829, 839, 853, 857, 859, 863, 877, 881, 883, 887, 907, 911, 919, 929, 937, 941, 947, 953, 967,
971, 977, 983, 991, 997]
```

LAB07 실습과제 1 (searchList.py)

- 사용자로부터 n개의 정수를 입력 받아 리스트에 저장한 후,
  리스트 내에 값들의 최대값, 최소값, 중간값을 찾아 출력하는 프로그램
- **중간 값 구하는 방법**
  - number_center = int(list전체item수/2)
  - 리스트의 전체 item 수가 홀수인 경우
    - list[number_center]
  - 리스트의 전체 item 수가 짝수인 경우
    - (list[number_center-1] + list[number_center]) / 2
- **최대, 최소 값** : min, max 함수 사용 가능
- **중간 값 출력 시, 소수점 둘째 자리**까지 출력되도록 해야 함

%.4.1f

LAB07 실습과제 1 (searchList.py)

- 참고 코드
  - 사용자로부터 값을 입력 받아, 리스트에 저장하는 부분

```
number = []

while True :
    print("Enter numbers. (To finish press 'Enter' key)")
    n = input()
    if len(n) == 0 :
        break
    number.append(float(n))

number.sort() #자동으로 오름차순 정렬
number_len = len(number) #number 리스트 원소 총 개수
```

LAB07 실습과제 1 (searchList.py)

- 실행화면

```
Enter numbers. (To finish press 'Enter' key)
22
Enter numbers. (To finish press 'Enter' key)
11
Enter numbers. (To finish press 'Enter' key)
44
Enter numbers. (To finish press 'Enter' key)
33
Enter numbers. (To finish press 'Enter' key)
66
Enter numbers. (To finish press 'Enter' key)
77
Enter numbers. (To finish press 'Enter' key)
55
Enter numbers. (To finish press 'Enter' key)
88
Enter numbers. (To finish press 'Enter' key)
99
Enter numbers. (To finish press 'Enter' key)
100
Enter numbers. (To finish press 'Enter' key)

You entered
[11.0, 22.0, 33.0, 44.0, 55.0, 66.0, 77.0, 88.0, 99.0, 100.0]

max : 100.0
min : 11.0
median : 60.50
```

```
Enter numbers. (To finish press 'Enter' key)
22
Enter numbers. (To finish press 'Enter' key)
88
Enter numbers. (To finish press 'Enter' key)
44
Enter numbers. (To finish press 'Enter' key)
100
Enter numbers. (To finish press 'Enter' key)
66
Enter numbers. (To finish press 'Enter' key)

You entered
[22.0, 44.0, 66.0, 88.0, 100.0]

max : 100.0
min : 22.0
median : 66.00
```

LAB07 실습과제 2 (words.py)

- 단어 입력, 단어 삭제, 단어장 전체 출력, 종료 기능으로 구성
  - **1) 1번 메뉴 선택 시, Enter 키를 누를 때까지 단어를 입력 받음**
    - 만약, 이미 저장된 단어인 경우, "Already Exist" 문구가 출력됨
  - **2) 2번 메뉴 선택 시, 단어장에서 해당 단어를 삭제함**
    - 없는 단어가 입력된 경우에는, "No Exist" 문구가 출력되고,
      단어장에 있는 단어가 입력될 때까지 입력 받음
  - **3) 3번 메뉴 선택 시, 단어장에 있는 모든 단어가 출력됨**
  - **4) 4번 메뉴 선택 시, 종료 / 다른 메뉴가 선택될 경우 "You entered wrong menu" 출력**
  - **5) 메뉴 선택 받을 때마다, 위의 메뉴도 출력해주어야 함**

LAB07 실습과제 2 (words.py)

```
**************************************************
*              Sookmyung Dictionary              *
**************************************************
                  1. Save words
                  2. Delete words
                  3. Print all words
                  4. Exit
==================================
Select >> 1
Enter word to save (Press 'Enter' key to finish)

Word : banana
Word : kiwi
Word : apple
Word : kiwi
Already Exist
Word : melon
Word : strawberry
Word :

**************************************************
*              Sookmyung Dictionary              *
**************************************************
                  1. Save words
                  2. Delete words
                  3. Print all words
                  4. Exit
==================================
Select >> 3

banana
kiwi
apple
melon
strawberry
```

```
**************************************************
*              Sookmyung Dictionary              *
**************************************************
                  1. Save words
                  2. Delete words
                  3. Print all words
                  4. Exit
==================================
Select >> 2
Enter word to delete

Word : orange
No Exist
Word : strawberry
Deletion complete

**************************************************
*              Sookmyung Dictionary              *
**************************************************
                  1. Save words
                  2. Delete words
                  3. Print all words
                  4. Exit
==================================
Select >> 3

banana
kiwi
apple
melon
```

```
**************************************************
*              Sookmyung Dictionary              *
**************************************************
                  1. Save words
                  2. Delete words
                  3. Print all words
                  4. Exit
==================================
Select >> 5
You entered wrong menu

**************************************************
*              Sookmyung Dictionary              *
**************************************************
                  1. Save words
                  2. Delete words
                  3. Print all words
                  4. Exit
==================================
Select >> 4
>>>
```

과제 채점 기준·기한

- **과제 제출 기한**
  - **5월 26일 화요일 오후 11시까지 제출**

- **제출 장소**
  - **스노우보드** 해당 주차 과제 제출 페이지에 업로드

- **추가 제출**
  - 제출기한 이후 **24시간 이내 메일로 전송** : 2점 감점
  - **그 이후는 받지 않음**

- **표절X**

과제 제출 형식 & 질문 메일

- **제출물** : 소스파일(.py)과 과제보고서(.docx) 합친 **압축파일 제출**
- **소스파일 이름** : 매 실습 과제 마다 ppt에 제시 예정
- **과제 보고서 양식** : **스노우보드**에서 다운로드
- **(소스파일+과제보고서) 압축 파일 이름 : Lab07_학번_이름**

- **이메일** : newkml22@gmail.com
- **질문 시 주의사항**
  - **과목, 분반, 전공, 이름, 학번** 알려주세요.
  - 몇 번 과제에서, 어떤 부분이 막혔는지 어떤 과정인지 **설명**과 함께 보내주세요.
  - 출석 문의, 과제 늦은 제출도 메일로
  - 답장까지 시간이 걸릴 수도 있으니 제출 과제 질문은 미리 해주세요!