MTA 98-381 LESSON 6 LISTS

LISTS

- A list is a collection of data, similar to vector in C++ or ArrayList in C#/Java.
- Example:

```
list1 = []              # Create empty list.
list2 = [3, 5, 9, 6]
print (list2)           # [3, 5, 9, 6]
print (len(list2))      # length of list2 = 4.
print (max(list2))      # 9
print (min(list2))      # 3
print (sum(list2))      # 23
```

LIST INDEX

- Use [*n*] to access the element at index *n*.
- *n* starts with 0 if count from left side.
- *n* starts with -1 if count from right side.
- Example:

```
list3 = ['FCI', 'FOE', 'FOM', 'FET', 'FIST', 'FOE']
print (list3[0])      # FCI
print (list3[2])      # FOM
print (list3[-1])     # FOE
print (list3[-3])     # FET
```

+ AND * OPERATORS

- + – create a new list by combining 2 lists.
- *n – create a new list by duplicating a list *n* times.
- Examples:

```
list1 = [3, 2]
list2 = [4, 5]
list3 = list1 + list2  # [3, 2, 4, 5]
list4 = list3 * 2      # [3, 2, 4, 5, 3, 2, 4, 5]
```

[:] OPERATORS

- `[ ]` – access an element in the list
- `[ : ]` – create a new list by slicing a list.
- `[start:stop:step]` – similar to `range(start,stop,step)`.
- Examples:

```
list4 = [3, 2, 4, 5, 1]
list4[2:4]              # [4, 5]
list4[:3]               # [3, 2, 4]
list4[1:]               # [2, 4, 5, 1]
list4[::2]              # [3, 4, 1]
list4[::-1]             # [1, 5, 4, 2, 3]
```

IN OPERATOR

- `in` – check whether an element exists in a list or not.
- `for..in` – iterate a loop.

```
list3 = [3, 2, 4, 5]
print (0 in list3)   # False
print (4 in list3)   # True

for item in list3:          # iterate using for/in.
    print (item)
for i in range(len(list3)): # iterate using range.
    print (list3[i])
```

MODIFYING ELEMENT USING FOR LOOP

- Note that `for/in` loop cannot be used to modify element in list.

```
list3 = [3, 2, 4, 5]

for item in list3:          # iterate using for/in.
   item *= 2
print (list3)           # [3, 2, 4, 5]. No changes.

for i in range(len(list3)): # iterate using range.
    list3[i] *= 10
print (list3)           # [30, 20, 40, 50]. Changed.
```

LIST EXERCISE 1:

```
list1 = [9, 6, 3, 2, 4, 5, 7]
```

- What is the result?

1. `list1[0]`
2. `list1[4]`
3. `list1[-1]`
4. `list1[-5]`
5. `list1[:4]`
6. `list1[:4:2]`
7. `list1[:4:-2]`
8. `list1[-1:-5:-2]`
9. `list1[::-2]`
10. `list1[::3]`

COMMON LIST OPERATIONS

- Examples.

```
list4 = []
# append (data) – add data to the back of list.
list4.append (10)      # [10]
list4.append (20)      # [10, 20]
# insert (index, data) – insert data at index.
list4.insert (0, 30)   # [30, 10, 20]
list4.insert (1, 40)   # [30, 40, 10, 20]
# remove (data)
list4.remove (30)      # [40, 10, 20]
# pop (index) – remove element at index.
list4.pop (1)           # [40, 20]
```

LIST EXERCISE 2

- Write a program that gets user inputs for exam marks, then calculate the following:
  1. Total number of marks
  2. Total number of passes and failures
  3. Highest mark and lowest mark
  4. Average mark
- Sample output

```
Enter mark (-1 to end): 80
Enter mark (-1 to end): 40
Enter mark (-1 to end): 55
Enter mark (-1 to end): -1
```

LIST EXERCISE 2 CONT.

- Sample output cont.

```
No   Mark  Result
 1   40.0  Fail
 2   80.0  Pass
 3   55.0  Pass
Total Pass   = 2
Total Fail   = 1
Highest      = 100.0
Lowest       = 5.0
Average Mark = 51.67
```