Exercises of chapter 5

1) Quick Check: Len()

Imagine:
a = [0]
b = []
c = [[1, 3, [4, 5], 6], 7]

Now look at the code below:

```
>>> a = [0]
>>> b = []
>>> c = [[1, 3, [4, 5], 6], 7]
>>> print(len(a))
>>> print(len(b))
>>> print(len(c))
```

So, len(a) would be 1;
And len(b) would be 0;
And len(c) would be 2.

2) Try this: List Slices and Indexes

As the floor division (//) Divides and returns the integer value of the quotient, if we use this operator, we would obtain our result. Look at the code below:

```
>>> list = [20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30]
>>> result = list[len(list) // 2:]
>>> result
```

3) Try This: Modifying Lists

Look at the code below:

```
>>> list = ['pen', 'pencil', 'backpack', 'pencil case', 'ruler', 'color pencils', 'sharpener', 'eraser', 'paper', 'book']

>>> list = list[-5:] + list[:-5]

>>> list

['color pencils', 'sharpener', 'eraser', 'paper', 'book', 'pen', 'pencil', 'backpack', 'pencil case', 'ruler']
```

*4) Try This: Sorting Lists

Look at the code below:

```
>>> list =  [[1, 2, 3], [2, 1, 3], [4, 0, 1]]
>>> list.sort(key = lambda a: a[1])
>>> list
```

Now the result would be:

```
>>> [[4, 0, 1], [2, 1, 3], [1, 2, 3]]
```

5) Try This: List Operations

Index	In
Describes the position	Describes the position by true or false

If an element doesn't exist in the list, it returns an error.

6) Try This: List Operations

I) if item in x:
    x.remove(item)

II) if x.count(item) > 1:
    x.remove(item)

*7) Try This: List Copies

One of the useful methods that can be used in this exercise is deepcopy() method after importing copy module:

```
>>> import copy
>>> x = copy.deepcopy(x)
```

8) Quick Check: Tuples

Once data is assigned to a tuple, the values cannot be changed.

x = sorted(x)

9) Quick Check: Sets

Due to the unrepeatable elements of sets, we would have 0, 1, 2, 3, 5 and the tuple (1, 2, 3).

10) Examining a list:

```
with open('lab_05.txt') as infile:
      for row in infile:
         temperatures.append(float(row.strip()))

min_t = min(temperatures)
max_t = max(temperatures)
mean_t = sum(temperatures) / len(temperatures)
temperatures.sort()
median_t = temperatures[len(temperatures) // 2]

print("min = {}".format(min_t))
print("max = {}".format(max_t))
print("median = {}".format(median_t))
print("mean = {}".format(mean_t))
```

```
min = 0.8
max = 28.2
median = 14.7
mean = 14.848309178743966
```