4_working with lists

Loop through entire list

For loop

```language-python
magicians = ['alice', 'david', 'carolina']
for idiots in magicians:
        print(idiots.title())
#Alice
#David
#Carolina
# it means" for every x in y, do..."
# repeat the operation for everything in the list
```

Indentation

Use spaces or tabs to make indentation lines. To tell python where to start a for loop and where to stop

Numerical list

Range

1by1

```language-python
# range(a,b) includes all intergers from a to b-1,
for value in range(2,7):
        print(value)
# 2,3,4,5,6
```

However, if we want to make a numerical list directly, just:

```language-python
numbers=list(range(1,7))
print(numbers)
#[1, 2, 3, 4, 5, 6]
```

Even gaps

What if we want to make list that skips numbers

```language-python
coollist=list(range(1,12,2))
print(coollist)
#[1, 3, 5, 7, 9, 11]
# the third figure in the range function, "2", means there is a consecutive gap of two.
```

List with equation

we don't need to remember all the notations for all kinds of lists. Instead, just go through the mathematical operation :

```
```

```language-python
# let's see how to create a list that: x^2+1, when 1<x<9 and x is an interger
#first, create an empty list
list1=[]
#then take every x into calculation, set a variable"newvalue"=x^2+1
for value in range(2,9):
        newvalue=value**2+1
        #add every newvalue into the list
        list1.append(newvalue)
print(list1)
#[5, 10, 17, 26, 37, 50, 65]
```

or

```language-python
list1=[]
for value in range(2,9):
        list1.append(value**2+1)
print(list1)
#[5, 10, 17, 26, 37, 50, 65]
```

Stats

```language-python
numbers=[1,3,4,6,7,8,]
min(numers)
max(numbers)
sum(numbers)
```

List comprehension

a more advanced way to generate way in one step:

```language-python
list_example=[x**2+1for x in range(2,9)]
print(list_example)
#[5, 10, 17, 26, 37, 50, 65]
#requires both the equation and the range for x
# all we need is the range for x and the equation that convert every x within the
range to y
```

Working with part of list

Slicing

language-python

```
family=['mom','dad','xixi','laolao']
print(family[1:3])
#['dad', 'xixi']
#this is like slicing a cake, where we select a starting point and a point to withdraw the knife.
#remember the second boundary means "before", therefore that boundary itself is not included
#[a+1:b]
```

language-python

```
# one can remove the starting or the ending index
print(family[2:])
# ['xixi', 'laolao']
print(family[:3])
# ['mom', 'dad','xixi']
```

Looping through slice

language-python

```
for members in family[:3]:
        print(members)
#mom
#dad
#xixi
```

Copying a list

language-python

```
#sometime we may want to copy a list and modify it
favorite_philosophers=['kant','marx','plato','nietzche']
famous_philosophers=favorite_philosophers[:]
print(famous_philosophers)
#['kant', 'marx', 'plato', 'nietzche']
#in this case, we can modify the new list without changing the previous one
```

Tuples

Tuple is a list that cannot be changed.
One use parentheses and comma instead of [] to denote tuples.

language-python

```
my_friend=('Andrew','zzt','xiaowu')
my_friend[1]='Magggie'
# if one try to change the tuple, errors will be reported
#TypeError: 'tuple' object does not support item assignment
```

language-python

```
# as tuple is denoted by parentheses and comma, to those lists that only have one
or zero item inside, comma is still required.
my_friend_female=('xiaowu',)
```