Many objects in Python are "iterable", meaning we can iterate over every element in the object

Such as every element in a list or every character in a string

We can use for loops to execute a block of code for every iteration

The term iterable means you can "iterate" over the object

For example you can iterate over every character in a string, iterate over every item in a list, iterate over every key in a dictionary.

Syntax of a for loop:

my_iterable = [1,2,3] (this is the iterable name and assignment)
for item_name in my_iterable: (the item_name represents the list of objects previously defined)
print(item_name)

In [1]:
```
mylist = [1,2,3,4,5,6,7,8,9,10]
```

In [2]:
```
for num in mylist:
    print(num)
```

```
1
2
3
4
5
6
7
8
9
10
```

In [3]:
```
for num in mylist:
    print('Hello')
```

```
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
```

Lets see if we can print out only the even numbers in this list.

In [4]:
```
for num in mylist:
    # Check for even
    if num % 2 == 0:
        print(num)
```

```
2
4
6
```

```
8
10
```

Remeber the % (mod) means the remainder So if num % 2 is equal to 0, print num. We see it then prints all even numbers.

In [5]:
```
for num in mylist:
    # Check for even
    if num % 2 == 0:
        print(num)
    else:
        print('Odd Numer')
```

```
Odd Numer
2
Odd Numer
4
Odd Numer
6
Odd Numer
8
Odd Numer
10
```

In [9]:
```
for num in mylist:
    # Check for even
    if num % 2 == 0:
        print(num)
    else:
        print(f'Odd Number: {num}')
```

```
Odd Number: 1
2
Odd Number: 3
4
Odd Number: 5
6
Odd Number: 7
8
Odd Number: 9
10
```

In [10]:
```
list_sum = 0

for num in mylist:
    list_sum = list_sum + num

print(list_sum)
```

```
55
```

list_sum = 0 (This line sets initial sum to 0)
so then we add list_sum (0) to every number in the list, then print it, which is 55

Now lets indent the print(list_sum) to be inside of the for loop

In [12]:
```
list_sum = 0

for num in mylist:
    list_sum = list_sum + num

    um)
```

```
1
3
6
10
15
21
28
36
45
55
```

We just did for loops with lists, now lets do it with string

In [13]:
```
mystring = 'Hello World'

for letter in mystring:
    print(letter)
```

```
H
e
l
l
o

W
o
r
l
d
```

Now lets do it for tuples:

In [15]:
```
tup = (1,2,3)

for item in tup:
    print(item)
```

```
1
2
3
```

In [16]:
```
mylist = [(1,2),(3,4),(5,6),(7,8)]
```

In [17]:
```
len(mylist)
```

Out[17]:
```
4
```

In [18]:
```
for item in mylist:
    print(item)
```

```
(1, 2)
(3, 4)
(5, 6)
(7, 8)
```

In [19]:
```
for (a,b) in mylist:
    print(a)
    print(b)
```

```
1
2
3
4
5
6
7
8
```

In [20]:
```
for (a,b) in mylist:
    print(a)
```

```
1
3
5
7
```

This is known as tuple unpacking

In [21]:
```
for (a,b) in mylist:
    print(b)
```

```
2
4
6
8
```

In [22]:
```
mylist = [(1,2,3),(5,6,7),(8,9,10)]
```

In [23]:
```
for item in mylist:
    print(item)
```

```
(1, 2, 3)
(5, 6, 7)
(8, 9, 10)
```

In [25]:
```
for a,b,c in mylist:
    print(a)
```

```
1
5
8
```

In [26]:
```
for a,b,c in mylist:
    print(b)
```

```
2
6
9
```

In [27]:
```
for a,b,c in mylist:
    print(c)
```

```
3
7
10
```

Now, let us iterate through a dictionary

```
d = {'k1':1, 'k2':2, 'k3':3}

for item in d:
    print(item)
```

```
k1
k2
k3
```

In [2]:

```
d = {'k1':1, 'k2':2, 'k3':3}

for item in d.items():
    print(item)
```

```
('k1', 1)
('k2', 2)
('k3', 3)
```

In [3]:

```
d = {'k1':1, 'k2':2, 'k3':3}

for key,value in d.items():
    print(value)
```

```
1
2
3
```

In [6]:

```
d = {'k1':1, 'k2':2, 'k3':3}

for value in d.values():
    print(value)
```

```
1
2
3
```

In [ ]:

```

```