RECAP

Defining Variables

>>> Item\_name = "Computer" #A String

>>> Item\_qty = 10 #An Integer

>>> Item\_value = 1000.23 #A floating point

Check the Type of variable

>>>Type(Item\_name)

Find the last character in the string

>>> Item\_name[-1]

Find the first character in the string

>>> Item\_name[0]

Check the length of string

>>> len(Item\_name)

Find the last character in the string counting forward

>>> Item\_name(len(Item\_name) - 1)

Calculation

>>> x = 2

>>> y = 3

>>> z = 5

>>> x \* y

6

>>> x + y

5

>>> x \* y + z

11

>>> (x + y) \* z

25

>>> 2\*\*8

256

>>> 5/2

2.5

Remainder

>>> 10%7

3

Question:

If today is Saturday, what day it will be after 100 days?

100%7

How to print hello 3 times?

Print(“hello” \* 3)

How to convert a int to string?

Str(3)

How to convert a int to float?

float(3)

List is a collection which is ordered and changeable. Allows duplicate members.

thislist = ["apple", "banana", "cherry"]  
print(thislist)

**Access Items**

You access the list items by referring to the index number:

thislist = ["apple", "banana", "cherry"]  
print(thislist[1])

**Change Item Value**

To change the value of a specific item, refer to the index number:

thislist = ["apple", "banana", "cherry"]  
thislist[1] = "blackcurrant"  
print(thislist)

**List slicing**

>>>a=[0,11,22,33,44,55,66,77,88,99]

Syntax of slicing: a[x:y:z]

>>> a[0:7:1] *#elements from 0 (included) to 7 (excluded)*

[0, 11, 22, 33, 44, 55, 66]

>>> a[0:7:2]

[0, 22, 44, 66]

*#If z is not given, z is taken as 1*

>>> a[0:7]

[0, 11, 22, 33, 44, 55, 66]

*#If z is positive , slicing will be from left to right , so x should be less than y.*

>>> a[2:7:1]

[22, 33, 44, 55, 66]

>>> a[7:2:1] *#wrong indexes for slicing*

[]

*#If z is negative ,slicing will be from right to left, so x should be greater than y*

>>> a[7:2:-1]

[77, 66, 55, 44, 33]

'''

if z is positive and

if x is not given , x is taken as 0

if y is not given , y is taken as the length of the list

'''

>>> a[:7:1]

[0, 11, 22, 33, 44, 55, 66]

>>> a[5::1]

[55, 66, 77, 88, 99]

'''

if z is negative and

if x is not given , x is taken as -1 (index of the last element)

if y is not given , y is taken as negative of length of the list

'''

>>> a[:-5:-1]

[99, 88, 77, 66]

>>> a[-5::-1]

[55, 44, 33, 22, 11, 0]

>>> a[5:-1:1]

[55, 66, 77, 88]

>>> a[-1:5:-1]

[99, 88, 77, 66]

**List inside list**

>>> a = [10,20,[30,40,50],60]

>>> print (a)

[10, 20, [30, 40, 50], 60]

>>> print (a[2])

[30, 40, 50]

>>> print (a[2][0])

30

>>> print (a[2][1])

40

**List Length**

To determine how many items a list have, use the len() method:

thislist = ["apple", "banana", "cherry"]  
print(len(thislist))

**max**

a=[0,11,22,33,44,55,66,77,88,99]

>>> max(a)

99

**min**

a=[0,11,22,33,44,55,66,77,88,99]

>>> min(a)

0

**Add Items**

**To add an item to the end of the list, use the append() method:**

thislist = ["apple", "banana", "cherry"]  
thislist.append("orange")  
print(thislist)

To add an item at the specified index, use the insert() method:

thislist = ["apple", "banana", "cherry"]  
thislist.insert(1, "orange")  
print(thislist)

**Remove Item**

There are several methods to remove items from a list:

The remove() method removes the specified item:

thislist = ["apple", "banana", "cherry"]  
thislist.remove("banana")  
print(thislist)

The pop() method removes the specified index, (or the last item if index is not specified):

thislist = ["apple", "banana", "cherry"]  
thislist.pop()  
print(thislist)

The del keyword removes the specified index:

thislist = ["apple", "banana", "cherry"]  
del thislist[0]  
print(thislist)

The clear() method empties the list:

thislist = ["apple", "banana", "cherry"]  
thislist.clear()  
print(thislist)

**Python List** reverse()**Method**

Reverse the order of the fruit list:

fruits = ['apple', 'banana', 'cherry']  
  
fruits.reverse()

**Python List** sort()**Method**

cars = ['Ford', 'BMW', 'Volvo']  
  
cars.sort()

**Tuple**

A tuple is a collection which is ordered and **unchangeable**. In Python tuples are written with round brackets.

**Create a Tuple:**

thistuple = ("apple", "banana", "cherry")  
print(thistuple)

**Access Tuple Items**

You can access tuple items by referring to the index number:

thistuple = ("apple", "banana", "cherry")  
print(thistuple[1])

**Change Tuple Values**

Once a tuple is created, you cannot change its values. Tuples are **unchangeable**.

thistuple = ("apple", "banana", "cherry")  
thistuple[1] = "blackcurrant"  
# The values will remain the same:  
print(thistuple)

**Add Items**

Once a tuple is created, you cannot add items to it. Tuples are **unchangeable**.

thistuple = ("apple", "banana", "cherry")  
thistuple[3] = "orange" # This will raise an error  
print(thistuple)

**Remove Items**

**Note:** You cannot remove items in a tuple.

 Tuples are **unchangeable**, so you cannot remove items from it, but you can delete the tuple completely:

The del keyword can delete the tuple completely:

thistuple = ("apple", "banana", "cherry")  
del thistuple  
print(thistuple) #this will raise an error because the tuple no longer exists

**Tuple Vs lists:**

Tuple is faster than lists.

When you define something that is fixed and you don’t want to change anymore, use tuple. Otherwise use list.

**Dictionary**

A dictionary is a collection which is unordered, changeable and indexed. In Python dictionaries are written with curly brackets, and they have keys and values.

Create and print a dictionary:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
print(thisdict)

**Accessing Items**

You can access the items of a dictionary by referring to its key name:

Get the value of the "model" key:

x = thisdict["model"]

There is also a method called get() that will give you the same result:

Get the value of the "model" key:

x = thisdict.get("model")

**Change Values**

You can change the value of a specific item by referring to its key name:

Change the "year" to 2018:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
thisdict["year"] = 2018

**Dictionary Length**

To determine how many items (key-value pairs) a dictionary have, use the len() method.

Print the number of items in the dictionary:

print(len(thisdict))

**Adding Items**

Adding an item to the dictionary is done by using a new index key and assigning a value to it:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
thisdict["color"] = "red"  
print(thisdict)

**Removing Items**

There are several methods to remove items from a dictionary:

The del keyword removes the item with the specified key name:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
del thisdict["model"]  
print(thisdict)

The pop() method removes the item with the specified key name:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
thisdict.pop("model")  
print(thisdict)

The clear() keyword empties the dictionary:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
thisdict.clear()  
print(thisdict)

**The dict() Constructor**

It is also possible to use the dict() constructor to make a dictionary:

thisdict = dict(brand="Ford", model="Mustang", year=1964)  
# note that keywords are not string literals  
# note the use of equals rather than colon for the assignment  
print(thisdict)

**Lists from Dictionaries**

>>> w={"house":"Haus","cat":"Katze","red":"rot"}

>>> w.keys()

['house', 'red', 'cat']

>>> w.values()

['Haus', 'rot', 'Katze']

**Dictionaries from Lists**

>>> dishes = ["pizza", "roasted duck", "taco", "Hamburger"]

>>> countries = ["Italy", "China", "Mexico", "USA"]

Now we will create a dictionary, which assigns a dish to a country, of course according to the common prejudices. For this purpose we need the function zip(). The name zip was well chosen, because the two lists get combined like a zipper. 

>>> country\_specialities = zip(countries, dishes)

>>> print country\_specialities

[('Italy', 'pizza'), ('China', 'roasted duck'), ('Mexico', 'taco'), ('USA', 'Hamburger')]

>>>

The variable country\_specialities contains now the "dictionary" in the 2-tuple list form. This form can be easily transformed into a real dictionary with the function dict().

>>> country\_specialities\_dict = dict(country\_specialities)

>>> print country\_specialities\_dict

{'China': 'roasted duck', 'Mexico': 'taco', 'Italy': 'pizza', 'USA': 'Hamburger'}

>>>

There is still one question concerning the function zip(). What happens, if one of the two argument lists contains more elements than the other one? It's easy: The superfluous elements will not be used:

>>> countries = ["Italy", "China", "Mexico", "USA", "Korea"]

>>> dishes = ["pizza", "roasted duck", "taco", "Hamburger"]

>>> country\_specialities = zip(countries,dishes)

>>> print country\_specialities

[('Italy', 'pizza'), ('China', 'roasted duck'), ('Mexico', 'taco'), ('USA', 'Hamburger')]

Exercise:

1. Given a = [12,23,6,123,99,108], find the largest number in a
2. With the same list, find the smallest number using sort(). The largest?
3. Given b = [“apple”, “banana”, “pear”, “orange”, “pomelo”, “lime”], create a dictionary using a and b(hint: use function zip() first)
4. Given c = (12, 6, 1, 8, 10, 17), find the numbers between second and fifth, inclusive.
5. Delete the smallest number in a. In c?