**5. LISTS**

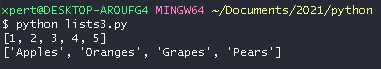
A list is a collection which is ordered and changeable. It allows duplicate members. This is basically an array in javascript, php etc. List is always in a square bracket [ ]

numbers = [1,2,3,4,5]

fruits = ['Apples', 'Oranges', 'Grapes', 'Pears']

print(numbers)

print(fruits)



**Get Value**

*#get value*

print(fruits[2])

The above code will print Grapes as the value

**Get Length**

*#get length*

print(len(fruits))

The above code result will be 4

**Append to list**

*#Append to list*

fruits.append('Oranges')

print(len(fruits))

This will add oranges to the list of fruits

**Remove from list**

*# Remove from list*

fruits.remove('Pears')

print(fruits)

This will remove pears from the fruit list

**Inserting into position:** This takes in two parameters, the position and item you want to add

*# Insert into a position*

fruits.insert(2,'Banana')

print(fruits)

**Remove from a position:**

*# Remove from a postion*

fruits.pop(3)

print(fruits)

**Reverse list**

*# Reverse list*

fruits.reverse()

print(fruits)

**Sort list**

*# sort list*

fruits.sort()

print(fruits)

There are more methods you can do on a list just like the string method, but these are the most commonly used.

**Classwork**

1. create a list of students
2. check the length, remove from a list and with position
3. add a student to a list and from a position
4. print out a particular student

**6. TUPLES AND SET**

**Tuples**

This is a collection which is ordered and unchangeable. It allows duplicate members. And it always in a bracket ( )

*# Tuples*

fruit\_turple =('Apple', 'Orange', 'Mango')

print(fruit\_turple)



**Get Value**

*#Get Value*

print(fruit\_turple[2])

Tuples with one value should have trailing comma, else the result will be a string instead of tuple.

*# Trailing comma*

fruit\_turple = ('Apple',)

print(fruit\_turple)

**Get Length**

*#Get Length*

print(len(fruit\_turple))

**Delete**

*# Delete a tuple*

del fruit\_turple

**SET**

This is a collection which is unordered and unindexed. And no duplicate members. And it always in a curly braces { }

fruit\_set = {'Apple', 'Orange', 'Mango', 'Grape'}

print(fruit\_set)



**Check if in set:** This brings out a True or False result

*# Check if in set*

print('Apple' in fruit\_set)

**Add to set**

*# Add to set*

fruit\_set.add('Banana')

**Remove to set**

*# Remove from set*

fruit\_set.remove('Orange')

**Delete to set**

*# Delete set*

del fruit\_set

**7. DICTIONARIES**

A dictionary is a collection which is unordered, changeable and indexed. No duplicate members.

students = {

  'first\_name': 'Anchorsoft',

  'last\_name': 'Academy',

  'number': 23,

}

print(students)



**Access value**

*# Access value*

print(students['first\_name'])

**Add key/value**

*# Add key/value*

students['phone'] = '08029501995'

print(students)

**Get keys**

*# Get key*

print(students.keys())

**Get Items**

*# Get items*

print(students.items())

**Making Copy**

*# Make Copy*

student1 = students.copy()

student1['address'] = 'Yinusa Adeniji'

print(student1)

**Get Length**

*# Get lenght*

print(len(students))

**List of dictionary**

*# List of dict*

student = [

  {

    'name': 'Anchorsoft',

    'number': 12,

  },

  {

    'name': 'Academy',

    'number': 1,

  },

  ]

**Classwork**

1. create a dictionary of a person
2. print the age of the person and add hobbies to the dictionary
3. make a copy of the person to another person2 adding height
4. create a list of 4 dictionary and access the information in the 3rd dictionary

**9. ARITHEMETIC, COMPARISON AND LOGICAL OPERATORS**

**Arithmetic Operators**

*# Arithemetic operators*

names            signs

----------------------

addition          +

subtraction       -

division          /

multiplication    \*

module %

**Comparison Operators**

*# Comparison Operators*

names                      signs

---------------------------------

equal to                      ==

not eqaul to                  !=

greater than                  >

less than                     <

greater than and equal to     >=

less than and equal to        <=

**Logical Operators**

*# Logical Operators*

AND: True if both the operands are true

OR: True if either of the operands is true

NOT: True if operand is false

**9. FUNCTIONS**

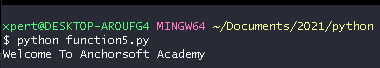
Function is a block of code which only runs when it is called. In python, we do not use parentheses and curly brackets; we use indentation with tabs or spaces.

*# create function*

def sayHello():

  print('Welcome To Anchorsoft Academy')

sayHello()  *#we have to call the function*



**Function with arguments**

*# Parameters or arguments*

def sayHello(name):

  print('Welcome ' + name)

sayHello('Anchorsoft')

**Default parameter/ arguments**

*# Default parameters or arguments*

def sayHello(name = 'Anchorsoft'):

  print('Welcome ' + name)

sayHello()

However, passing a default parameter or argument and also passing a parameter to the call function at the same time is possible, but the call function will override the default parameter.

def sayHello(name = 'Anchorsoft'):

  print('Welcome ' + name)

sayHello('Academy')

The code above, the Academy will override the Anchorsoft.

**Return Value**

Usually we don’t print out from a function we return.

*# Return Value*

def sumNum(num1, num2):

  total = num1 + num2

  return total

print(sumNum(3,5))

Note variable define in a function cannot be accessible outside the function.

**Lambda Function**

This is a small anonymous function and can take any number of arguments, but can only have one expression.

*# Lambda*

getSum = lambda num1, num2 : num1 + num2

print(getSum(3, 5))

**Classwork**

1. Create a simple calculator to add, subtract, divide, multiply
2. Adding two unknown number i.e user must input any number

**Assignment**

1. Adding two unknown number but one should have a default parameter
2. A simple calculator to add three numbers, add two of the three numbers and subtract one.

**10. CONDITIONALS**

This is use to decide an action base on certain condition; if (condition) then do this.

**If Statement**

*# if statement*

x = 10

y = 5

if x == y:

  print(f'{x} is equal to {y}')

**If/else Statement**

*# if/else statement*

x = 10

y = 5

if x == y:

  print(f'{x} is equal to {y}')

else:

  print(f'{x} is not equal to {y}')

**elif Statement**

*# elif statement*

x = 10

y = 5

if x > y:

  print(f'{x} is greater than {y}')

elif x < y:

  print(f'{x} is less than {y}')

else:

  print(f'{x} is not greater or less than {y}')

**Nested if Statement**

*# Nested if*

x = 6

if x > 2:

  if x <= 10:

    print(f'{x} is greater than 2 and less than 10')

**Logical Operator**

The code above using nested if should be written using logical operator

*# Logical Operator*

x = 6

if x > 2 and x <= 10:

  print(f'{x} is greater than 2 and less than 10')

#or

x = 6

if x > 2 or x <= 10:

  print(f'{x} is greater than 2 and less than 10')

*#not*

x =3

y =4

if not(x==y):

  print(f'{x} is not equal to {y}')

**Membership Operators**

These are used to test if a sequence is presented in an object and they are (not, not in) and returns True or False as the output.

*#Membership Operators*

number = [1,2,3,4,5,6]

x= 6

if x in number:

  print(x in number)

*# not in*

number = [1,2,3,4,5,6]

x= 6

if x not in number:

  print(x in number)

**Identity Operators**

These are used to compare the objects, not if they are equal, but if they are actually the same object, with the same memory location. And they are (is, is not) and returns True or False as the output.

*# Identity Operators*

*# is*

x = 10

y = 10

if x is y:

  print(x is y)

*#is not*

x = 10

y = 5

if x is not y:

  print(x is not y)