**PYTHON**

**Unit test: python3 -m unittest test\_structures.py**

**Code examples:** [**https://www.programiz.com/python-programming/examples**](https://www.programiz.com/python-programming/examples)

**Help:**

* [**https://www.w3schools.com/python/python\_strings.asp**](https://www.w3schools.com/python/python_strings.asp)
* [**https://stackoverflow.com/questions/tagged/python**](https://stackoverflow.com/questions/tagged/python)
* [**https://www.geeksforgeeks.org**](https://www.geeksforgeeks.org)
* Linting: pylint checks for errors

**GIT:** Pull-> create/change file -> stage (add) files to be committed -> write commit message -> commit -> push (copy over all changes to all other computers with the same repo). **Forking** is creating a copy of a repo within your own user account where you can make changes that don’t mess with the original repo.

Open python

* Python3
* Open program: import filename (without the .py). Ex: toys
* Open function: filename.function. Ex: toys.addOne

Structure: **“”” docstring “””** (first string after defining a function - can be called by writing help(greg)) and **#comments** (explain in plain language what is happening in your code)

|  |  |  |
| --- | --- | --- |
| Functions   * print(): * input(): user types * len(): * str() * int() * float() * == equal to * != not equal to * <= less than or equal to * >= greater than or equal to * +=1, -=1, \*=1, /=1, %=1   Boolean operators:   * and (both true), or (one true)   *x = "Hello"*  *y = 15*  *print(bool(x))*  *print(bool(y))*   * If, elif, else   Loops:   * While: keeps repeating   + Break   + Continue: goes to the start   + Avoid infinite loop: CTRL-C * For: repeats for a set nº of times * Range: third value is the amount that the variable is increased by. Range(5, -1, -1) should print from five down to zero   Errors   * Try and except   *Def div42by(divideBy):*  *Try:*  *Return 42/divideBy*  *Except ZeroDivisionError:*  *print(‘Error: can’t do it’)*  *Print(div42By(2))*    *print (‘How many cats?’)*  *numCats = input()*  *Try:*  *If int(numCats) <= 4:*  *Print(‘that’s a lot’)*  *Else:*  *Print(‘That’s not a lot’)*  *Except ValueError:*  *print (‘You didn’t enter a nº’)*  Types: AssertionError, ImportError, NameError, SyntaxError, TypeError, ValueError, ZeroDivisionError | Strings   * Import:   *Import random*  *Random.randit(1,10)*  *Import sys*  *sys.exit()*   * Slicing:   *b = "Hello, World!"*  *print(b[2:5])*   * In or not in * Escape:   *txt = "I am \"Okay\" now.“*   * Caseless comparison   *my\_str = my\_str.casefold()*   * To reverse the string   *rev\_str = reversed(my\_str)*  String methods   * .strip(): removes white spaces from the beginning or the end * .lower(): * .upper(): * .replace(‘H’, ‘J’) * .format()   *myorder= ‘I want {}’*  *print(myorder.format(you))*  Lists: first item has index 0  *list = ["apple", "pear", "kiwi"]*  *print(list[1])*  *print(list[2:5])*  *thislist[1] = "blackcurrant"*  List methods:   * .append() : adds element to end * .insert(1, ‘something’) * .remove() : * .clear(): removes all elements * .copy(): returns a copy of the list * .count(): Returns nº of elements * .extend(): joins 2 lists * .index(): gives index of the first element with the specified value * .pop (): removes the element at the specified position * .remove(): Removes the item with the specified value * .reverse() * .sort()   + List.sort(reverse=True)   + alphabetical order = list.sort(key=str.lower)   Tuples; unchangeable, immutable  Set: unordered   * .add() * .update(): add more than 1 item | Dictionaries: {}   * Key, variable [] * Prints all keys:   *for x in thisdict:*  *print(x)*  *for x in thisdict.keys():*  *print(x)*   * Prints all values:   *for x in thisdict:*  *print(thisdict[x])*  *for x in thisdict.values():*  *print(x)*   * Prints everything:   *for x, y in thisdict.items():*  *print(x, y)*   * Adding values:   *thisdict = {‘a’=’1’, ‘b’=’2’}*  *thisdict["c"] = "3"*  *print(thisdict)*   * Removing values:   *thisdict = {‘a’=’1’, ‘b’=’2’} thisdict.pop("a")*  *print(thisdict)*  Dictionary methods   * .clear(): removes everything * .copy() * .get(): gives value of specific key * .items(): list key value pair * .keys(): only keys * .pop(): removes specific element * .popitem(): removes the last pair * .setdefault(): returns the value of the specified key. If the key does not exist: insert the key, with the specified value * .update(): * .values(): list of all values   Default:   * Create optional arguments   *def fun(country = "Spain"):*  *print("I am from " + country)* |

Remove any duplicates from a List: dictionaries cannot have duplicate keys

mylist = ["a", "b", "a", "c", "c"]

mylist = list(dict.fromkeys(mylist))

print(mylist)

* Sep=’,’: deletes the default separating space between strings

>>> **print('cats', 'dogs', 'mice')**

cats dogs mice

>>> print('cats', 'dogs', 'mice', sep=',')

cats,dogs,mice

Converting between lists and tuples: Tuple([]), List(())

if name in birthdays:

print(birthdays[name] + ' is the birthday of ' + name)

else:

print('I do not have birthday information for ' + name)

print('What is their birthday?')

bday = input()

birthdays[name] = bday

print('Birthday database updated.')

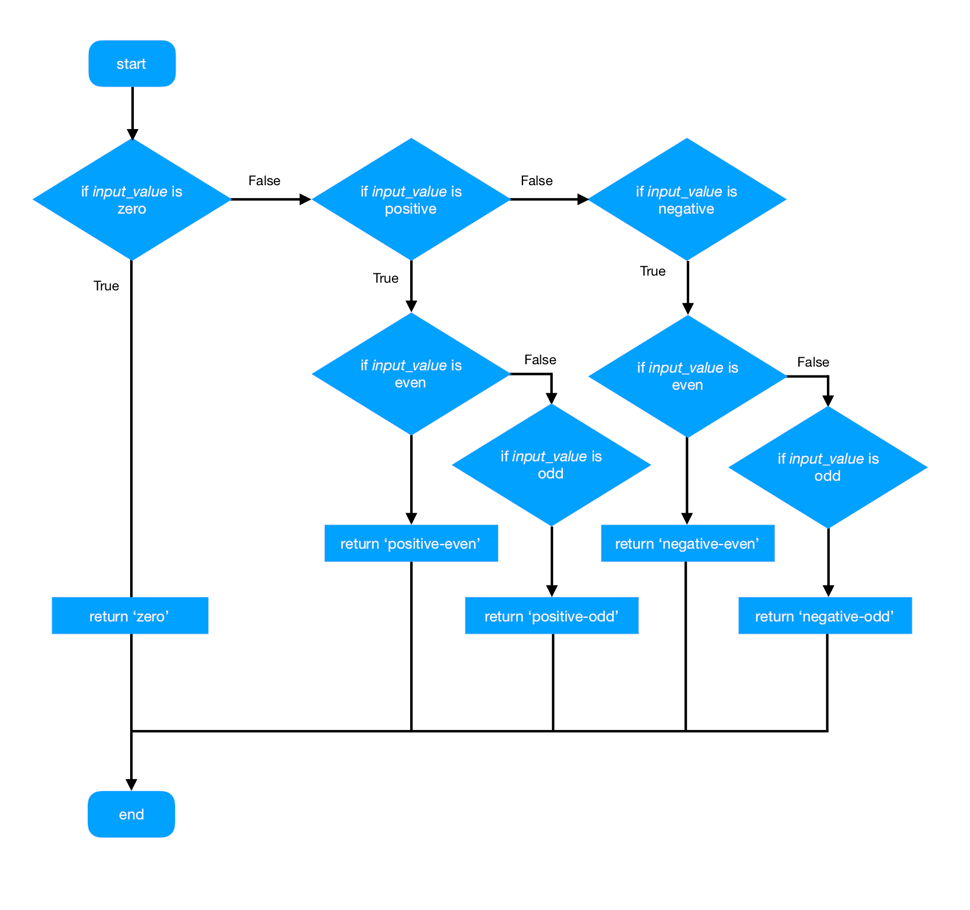
>>> **picnicItems = {'apples': 5, 'cups': 2}**

>>> **'I am bringing ' + str(picnicItems.get('cups', 0)) + ' cups.'**

'I am bringing 2 cups.'

>>> **'I am bringing ' + str(picnicItems.get('eggs', 0)) + ' eggs.'**

'I am bringing 0 eggs.'

**ASSESEMENT PRACTICE:**

def flowchart(input\_value):

    if input\_value == 0:

        return 'zero'

    elif input\_value > 0:

        if input\_value%2:

            return 'positive-even'

        else:

            return 'positive-odd'

    else:

        if input\_value%2:

            return 'negative-even'

        else:

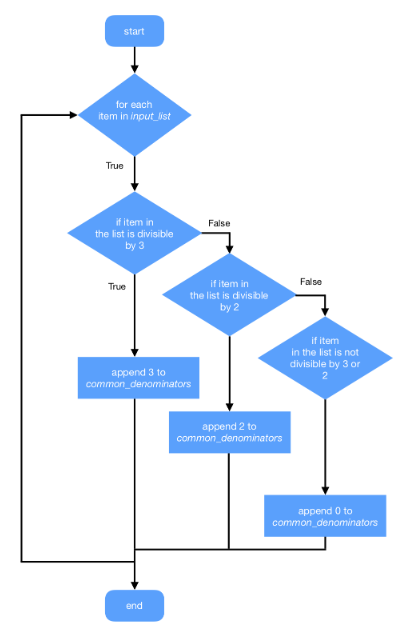
            return 'negative-odd'

def flowchart(input\_list):

    '''Takes in list and returns a new list of the numbers 3, 2, or 0

    corresponding with what number the original list was divisble by

    (3, 2, or neither).'''

****

    # create an empty list to fill and return

    common\_denominators = []

    for i in input\_list:

        # if the item is divisble by three

        if not i%3:

            # add the value 3 to the list

            common\_denominators.append(3)

        # if the item is divisible by two

        elif not i%2:

            # add the value 2 to the list

            common\_denominators.append(2)

        # otherwise

        else:

            # append the value 0

            common\_denominators.append(0)

    return common\_denominators

def count\_words(input\_string):

    '''Counts the number of words between spaces'''

    # create a list of each word

    words = input\_string.split()

    # return the length of the list

    return len(words)

def main():

    '''Main function to be run first.'''

    my\_string = 'Imperial College London'

    # count number of words in string

    num\_words = count\_words(my\_string)

    # print how many words were counted

    print('There are {} words in the string:'.format(num\_words))

    print(my\_string)

if \_\_name\_\_ == '\_\_main\_\_':

    main()

def reverse\_words(input\_string):

    '''Reverse the order of the words between spaces'''

    # create a list of each word

    words = input\_string.split()

    # reverse the list

    words = words[::-1]

    # put back together in a string

    rev = ''

    for i in words:

        rev += i + ' '

    # return the string

    return rev

def main():

    '''Main function to be run first.'''

    my\_string = 'Imperial College London'

    # count number of words in string

    reverse\_string = reverse\_words(my\_string)

    # print how many words were counted

    print('The original string is: {}'.format(my\_string))

    print('and the words reversed is: {}'.format(reverse\_string))

if \_\_name\_\_ == '\_\_main\_\_':

    main()

# Lists

# Write a function that merges two lists together only if the length of both

# lists is greater than 4. Otherwise return an empty list.

def merge\_list(list1, list2):

    if len(list1) > 4 and len(list2) > 4:

        return list1 + list2

    return []

# Write a function that returns a list containing unique values from the input\_list.

# In other words, any repeated values are removed.

def unique\_values(input\_list):

    return list(set(input\_list))

# Strings

# Write a function that returns a string where first "x" characters are lower

# case. "x" is passed as the second parameter.

def start\_to\_lower(string, x):

    return string[:x].lower() + string[x:]

# Write a function that returns the character which first occurs for the second

# time in the string. For example in string "abcdefcab" the character "c"

# occurs twice first. If no character occurs more than once, an empty string

# should be returned.

def first\_double\_char(string):

    chars = []

    for c in string:

        if c in chars:

            return c

        else:

            chars.append(c)

    return ""

# Dictionaries

# Write function that returns a sum of all values stored in the dictionary.

def sum\_of\_dict(dictionary):

    return sum(dictionary.values())

# Write a function that returns a new dictionary where each value is multiplied

# by 3. Example: for input dictionary {'a': 1, 'b': 2, 'c': 3} the returned

# dictionary will be {'a': 3, 'b': 6, 'c': 9}

def multiply\_dict(dictionary):

    new\_dict = {}

    for key, value in dictionary.items():

        new\_dict[key] = value \* 3

    return new\_dict

# Arguments

# Write a function with the first argument mandatory and the second optional.

# The first argument accepts a list as an argument, the second accepts two

# strings "max" or "min". The function returns either the maximum or the

# minimum value from the list. The default operation is "max". If the operation

# is neither "max" nor "min" the function should return "None" (the value, not

# the string).

def max\_min(input\_list, operation = 'max'):

    if operation == 'max':

        return max(input\_list)

    elif operation == 'min':

        return min(input\_list)

    else:

        return None