Level 6: Python Lists

The most basic data structure in Python is the **sequence**. Each element of a sequence is assigned a number - its position or index. The first index is zero, the second index is one, and so forth.

Python has six built-in types of sequences, but the most common ones are lists and tuples.

The list is a most versatile datatype available in Python which can be written as a list of comma-separated values (items) between square brackets. Important thing about a list is that items in a list need not be of the same type.

Ex: mylist = [1, 2, 3, 4, 5]

Accessing Values in Lists

To access values in lists, use the square brackets for slicing along with the index or indices to obtain value available at that index. Ex: mylist[2] -> 3

Basic List Operations

lists respond to all the general sequence operations we use on strings:

Python Expression	Results	Description
len([1, 2, 3])	3	Length
[1, 2, 3] + [4, 5, 6]	[1, 2, 3, 4, 5, 6]	Concatenation
['Hi!'] * 4	['Hi!', 'Hi!', 'Hi!', 'Hi!']	Repetition
3 in [1, 2, 3]	True	Membership
for x in [1, 2, 3]: print x,	1 2 3	Iteration

Indexing, Slicing, and Matrixes

Assuming following input –

```
L = ['spam', 'Spam', 'SPAM!']
```

Python Expression	Results	Description
L[2]	SPAM!	Offsets start at zero
L[-2]	Spam	Negative: count from the right
L[1:]	['Spam', 'SPAM!']	Slicing fetches sections

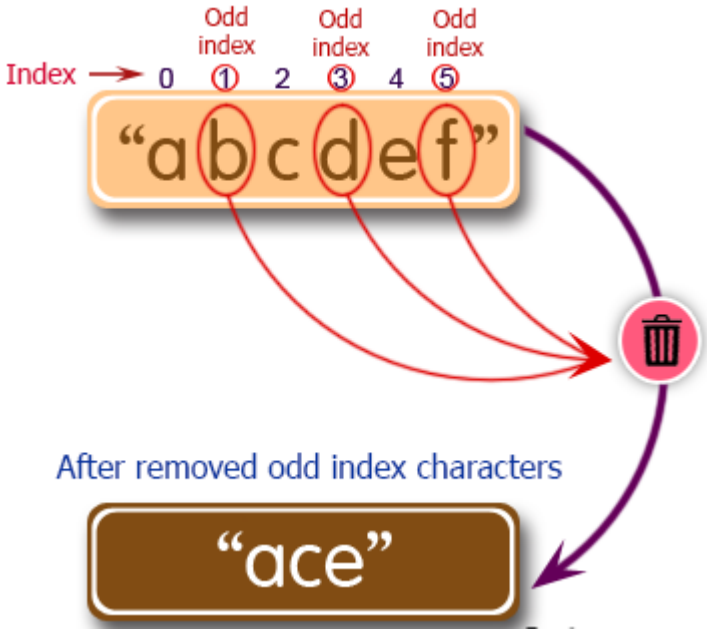
Built-in List Functions & Methods

Sr.No.	Function with Description
1	cmp(list1, list2) Compares elements of both lists.
2	len(list) Gives the total length of the list.
3	max(list) Returns item from the list with max value.
4	min(list) Returns item from the list with min value.
5	list(seq) Converts a tuple into list.

Sr.No.	Methods with Description
1	list.append(obj) Appends object obj to list
2	list.count(obj) Returns count of how many times obj occurs in list
3	list.extend(seq) Appends the contents of seq to list
4	list.index(obj) Returns the lowest index in list that obj appears
5	list.insert(index, obj) Inserts object obj into list at offset index
6	list.pop(obj=list[-1]) Removes and returns last object or obj from list
7	list.remove(obj) Removes object obj from list
8	list.reverse() Reverses objects of list in place
9	list.sort([func]) Sorts objects of list, use compare func if given

Problem to solve:

Write a Python program to remove the characters which have odd index values of a given string.



Example: Input: “HDEOLKLJOL”

Output: “HELLO”

Try your program on this sentence:

```
Tdhsi3sv aegxdekrsctihsder hsjeicyrredts ewtoyrhdf eissa wh4urnygureyw dkg
eheypi yi7ty rfeowrf gljaytterre
```

Level 7: Python Lists2

Problem to solve:

Show 2 ways of finding the length of a list.

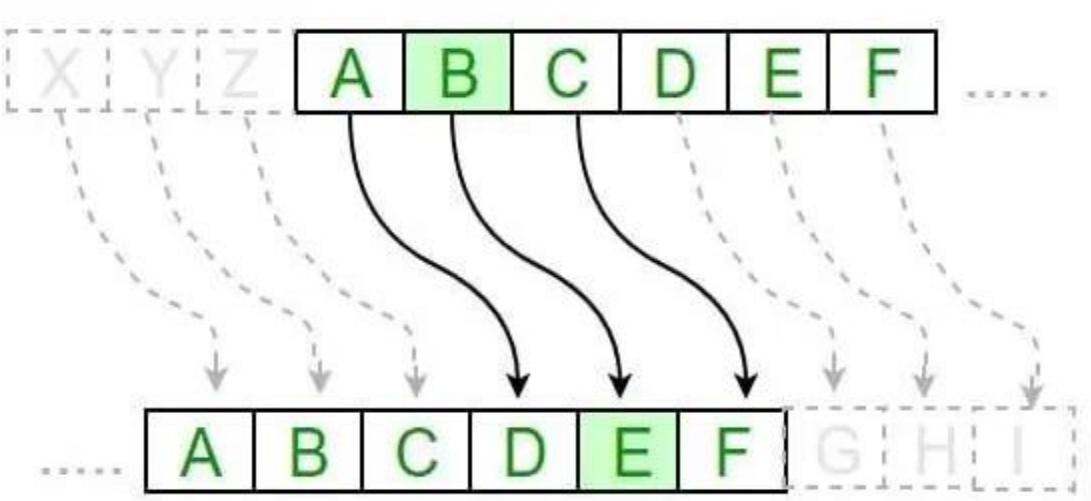
Level 8: Python Lists3

Problem to solve:

Write a function that can encrypt and decrypt a text.

You should use the "**Caesar Cipher"** method to encrypt and decrypt your text**.** The Caesar Cipher is an encryption algorithm that takes in a *key* (integer) and text (string). It encrypts the text by moving every letter of the text "forward" in the alphabet a total of *key* places. This *key* acts as the password that will be required to decrypt the encrypted text.

Example: here is an implementation of Caesar Cipher using key = 3.



In the example, A → D, B → E, C → F etc. The letters move three places "forward." Notice how this "wraps around" when you get to X? X → A, Y → B and Z → C. You can think of the alphabet as a clock that starts at A and goes clockwise up to Z and then back to A again. *What do you think happens if **key** = -3?*

Use the function to decrypt the following message (the message was encrypted using with key=4):

```
csyv wigsrh wigvix asvh mw fyvkiv oiit mx amxl csy hs rsx xlvsa mx eaec
```

Level 9: 2d lists

2-dimensional lists are an extremely important data structure in Python programming, with many applications.

Understanding 2d Lists in Python

To not get lost when using 2D arrays in Python, you need to fix your orientation, much like when reading a map. You should always be clear about whether a particular index is referring to a row or a column.

You can think of a 2D list as a list of lists. When arranged into a grid, each row represents a single list.

Accessing elements in a 2D List in Python

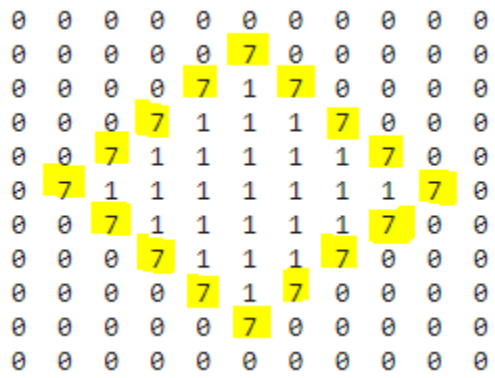
To access an element from the 2d list, you need a double index, which looks like this:

print(list2d[3][1])

Note that print(list2d[3,1]) will lead to a TypeError, even though it might seem to make sense

Problem to solve (level 2 squid game):

Create a 2d array that contains a shape Ex:



7 represents the borders of the shape, 1s are the content inside the shape, and 0s are the outside of your shape.

The purpose of this game is to eliminate all 7s

Create a program that will keep asking the user to enter dimensions Ex : 1 5 (my2dlist[1][5])

Check the 2d list at these indeces. If, the number is 0 or 7 -> replace it by 2.

If the number is 1 print “you lost the game, bye!” and finish the program.

When all 7s are eliminated from the list, return “you win! To the next game!” and finish the program.

you can use this list:

```
level2sg =[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 7, 0, 0, 0,
0, 0], [0, 0, 0, 0, 7, 1, 7, 0, 0, 0, 0], [0, 0, 0, 7, 1, 1, 1, 7, 0, 0, 0
], [0, 0, 7, 1, 1, 1, 1, 1, 7, 0, 0], [0, 7, 1, 1, 1, 1, 1, 1, 1, 7, 0], [
0, 0, 7, 1, 1, 1, 1, 1, 7, 0, 0], [0, 0, 0, 7, 1, 1, 1, 7, 0, 0, 0], [0, 0
, 0, 0, 7, 1, 7, 0, 0, 0, 0], [0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0], [0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0]]
```

Level 9: dictionary

Each key is separated from its value by a colon (:), the items are separated by commas, and the whole thing is enclosed in curly braces. An empty dictionary without any items is written with just two curly braces, like this: {}.

Keys are unique within a dictionary while values may not be. The values of a dictionary can be of any type, but the keys must be of an immutable data type such as strings, numbers, or tuples.

Accessing Values in Dictionary

To access dictionary elements, you can use the familiar square brackets along with the key to obtain its value.

Ex: dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}

print(dict['Name']) -> Zara

print(dict['Age']) -> 7

If we attempt to access a data item with a key, which is not part of the dictionary, we get an error

Properties of Dictionary Keys

Dictionary values have no restrictions. They can be any arbitrary Python object, either standard objects or user-defined objects. However, same is not true for the keys.

There are two important points to remember about dictionary keys:

(a) More than one entry per key not allowed. Which means no duplicate key is allowed. When duplicate keys encountered during assignment, the last assignment wins.

(b) Keys must be immutable. Which means you can use strings, numbers or tuples as dictionary keys but something like ['key'] is not allowed.

Built-in Dictionary Functions & Methods

Sr.No.	Function with Description
1	cmp(dict1, dict2) Compares elements of both dict.
2	len(dict) Gives the total length of the dictionary. This would be equal to the number of items in the dictionary.
3	str(dict) Produces a printable string representation of a dictionary
4	type(variable) Returns the type of the passed variable. If passed variable is dictionary, then it would return a dictionary type.

Sr.No.	Methods with Description
1	dict.clear() Removes all elements of dictionary *dict*
2	dict.copy() Returns a shallow copy of dictionary *dict*
3	dict.fromkeys() Create a new dictionary with keys from seq and values *set* to *value*.
4	dict.get(key, default=None) For *key* key, returns value or default if key not in dictionary
5	dict.has_key(key) Returns *true* if key in dictionary *dict*, *false* otherwise
6	dict.items() Returns a list of *dict*'s (key, value) tuple pairs
7	dict.keys() Returns list of dictionary dict's keys
8	dict.setdefault(key, default=None) Similar to get(), but will set dict[key]=default if *key* is not already in dict
9	dict.update(dict2) Adds dictionary *dict2*'s key-values pairs to *dict*
10	dict.values() Returns list of dictionary *dict*'s values

Problem to solve:

Your grandma/grandpa is new to the internet, and there are some slang words that s/he does not understand. Create a dictionary that contains the below slang words and create a program that will take a slang from the user and give the explanation to this word.

Acronym	Full Form
BRB	Be Right Back
DP	Display Picture
LMAO	Laughing My A** Off
LOL	Laughing Out Loud
OK	Olla Kalla or Oll Korrect
PFA	Please Find Attachment, Predictive Failure Analysis
ROFL	Rolling On Floor Laughing
TBH	To Be Honest
TTYL	Talk To You Later
ETA	Estimated Time of Arrival
FYI	For Your Information

Next Exercises:

To open the next set of exercises you need the password to this link:

https://lauedu74602-my.sharepoint.com/:f:/g/personal/nour_kfoury_lau_edu/EppZhHWIgdRIkQ-N1eS6QrkBUKmLivnXgVMQWWQfOfw7QQ?e=FZsiPh

To get the password you need combine the secret word you got from level 6 and the secret word you got from level 8 + “123”. Ex:
Secret word from level 6: “incorrect”
Secret word from level 8: “password”

Then the password is : incorrectpassword123

References:

https://www.tutorialspoint.com/python/python_lists.htm

https://compucademy.net/python-2d-lists/

https://www.tutorialspoint.com/python/python_dictionary.htm