**List methods**

Let's practice list methods!

Let's imagine a situation: you went to the market and filled your baskets (basket1 and basket2) with fruits. You wanted to have one of each kind but realized that some fruits were put in both baskets.

**Task 1**. Your first task is to remove everything from basket2 that is already present in basket1.

**Task 2**. After the removal it is reasonable to anticipate that one of the baskets might weigh more compared to the another (all fruit kinds weight the same). Therefore, the second task is to transfer some fruits from a heavier basket to the lighter one to get approximately the same weight/amount of fruits.

**Instructions 1/2**

**50 XP**

* [1](javascript:void(0))
* [2](javascript:void(0))
* Remove fruits from basket2 that are already present in basket1.

# Remove fruits from basket2 that are present in basket1

for item in basket1:

if item in basket2 :

basket2.remove(item)

print('Basket 1: ' + str(basket1))

print('Basket 2: ' + str(basket2))

<script.py> output:

Basket 1: ['banana', 'kiwifruits', 'grapefruits', 'apples', 'apricots', 'nectarines', 'oranges', 'peaches', 'pears', 'lemons']

Basket 2: ['grapes', 'dragonfruits', 'limes', 'papaya']

**Instructions 2/2**

**50 XP**

* [2](javascript:void(0))
* Transfer fruits from basket1 to basket2 until the amount in basket2 becomes more or equal to the amount in basket1.

# Remove fruits from basket2 that are present in basket1

for item in basket1:

if item in basket2:

basket2.remove(item)

print('Basket 1: ' + str(basket1))

print('Basket 2: ' + str(basket2))

# Transfer fruits from basket1 to basket2

while len(basket1) != len(basket2):

item\_to\_transfer = basket1.pop()

basket2.append(item\_to\_transfer)

print('Basket 1: ' + str(basket1))

print('Basket 2: ' + str(basket2))

Well done! We practiced some of the list methods but there are many more! Don't hesitate to practice them as well.

# Operations on sets

Putting the information on sets in more mathematical terms, we can define the following operations given two sets XX and YY:

  X∩YX∩Y - the intersection between XX and YY (all elements which are in both XX and YY)

  X∪YX∪Y - the union between XX and YY (all elements which are either in XX or YY)

  X−YX−Y - the difference between XX and YY (all elements which are in XX but not in YY)

You are given 5 sets of integers A, B, C, D,E. What is the result of the following expression?

(A∪(B∩C))−(D∩E)

In [2]: (A.union((B.intersection(C))))-(D.intersection(E))

Out[2]: {1, 2, 3, 4, 5, 6, 7}

##### Possible Answers

* 

{2}

* 

{}

* 

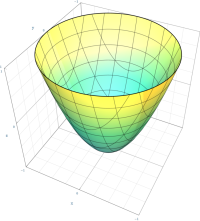
{1, 2}

* 

{1, 2, 3, 4, 5, 6, 7} **(A)**

# Storing data in a dictionary

The surface you see below is called circular paraboloid:



It can be described by the following equation:

x2a2+y2a2=zx2a2+y2a2=z

Let's set the coefficient aa to 1. Therefore, the radius at each cut will be equal to z√z.

Your task is to create a dictionary that stores the mapping from the pair of coordinates (x,y)(x,y) to the zz coordinate (the lists storing considered ranges for xx and yy are given: range\_x and range\_y, respectively).

**Instructions 1/3**

**50 XP**

* [1](javascript:void(0))
* [2](javascript:void(0))
* [3](javascript:void(0))
* Calculate the value for zz coordinate using coordinates xx and yy.
* Create a new key for the dictionary circ\_parab represented as a tuple containing xx and yy.
* Create a new key-value pair for circ\_parab.

range\_x = [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0]

range\_y = [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0]

In [1]: circ\_parab = dict()

for x in range\_x:

for y in range\_y:

# Calculate the value for z

z = x , y

# Create a new key for the dictionary

key = z

# Create a new key-value pair

circ\_parab[key] = z

**O/P :**

range\_x = [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0]

range\_y = [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0]

range\_x = [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0]

range\_y = [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0]

range\_x = [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0]

range\_y = [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0]

range\_x = [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0]

range\_y = [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0]

##### Instructions 2/3

**50 XP**

* [2](javascript:void(0))
* [3](javascript:void(0))

#### Question

What is the value of circ\_parab for the key (1.8, 1.4)?

##### Possible Answers

* 

5.2 **(A)**

* 

10.0

* 

5.96

* 

4.0

In [5]: circ\_parab[(1.8, 1.4)]

Out[5]: 5.2

##### Instructions 3/3

**0 XP**

* [3](javascript:void(0))

#### Question

Is it possible to use a list instead of a tuple for a key in the circ\_parab dictionary?

##### Possible Answers

* 

Yes, there is no substantial difference between two data structures in this regard.

* 

No, because a list is mutable and the operation will result in TypeError. **(A)**

* 

No, because a dictionary can accept only a tuple as a key.

* 

No, because a list is an ordered sequence and the operation will result in TypeError.

**KEYS CAN REPRESENT ONLY IMMUTABLE OBJECTS**