Colab Link - https://colab.research.google.com/drive/1bCHzL1U8EetaXcp96Z5rkX-5YFJA_UZk?usp=sharing

Universal Functions (Ufuncs) - vectorised functions

- Aggregate Functions/Reduction Functions
- Logical Functions

+ Code + Text

```
import numpy as np
```

```
a = np.array([1, 2, 3, 4])
b = np.array([4, 5, 6, 7])
a + b
```

```
array([ 5,  7,  9, 11])
```

```
np.add(a, b)
```

```
array([ 5,  7,  9, 11])
```

```
# sum
a = np.arange(12).reshape(3, 4)
```

```
a
```

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

Double-click (or enter) to edit

```
np.sum(a)
```

```
66
```

Saving...

```
a.shape
```

```
(3, 4)
```

```
np.sum(a, axis=0)
```

```
array([12, 15, 18, 21])
```

```
np.sum(a, axis=1)
```

```
array([ 6, 22, 38])
```

```
a
```

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

```
np.mean(a)
```

```
5.5
```

```
np.mean(a, axis=1)
```

```
array([1.5, 5.5, 9.5])
```

```
np.mean(a, axis=0)
```

```
array([4., 5., 6., 7.])
```

```
np.min(a)
```

```
0
```

```
 # calculate minimum value in every column
 np.min(a, axis=0)
```

```
array([0, 1, 2, 3])
```

```
# Logical Functions
```

```
a = np.array([1, 2, 3, 4])
```

```
# if there is any element in the array which is non-zero
np.any(a)
```

```
True
```

Saving...

```
np.any(a)
```

```
True
```

```
a = np.array([0, 0, 0, 0])
np.any(a)
```

```
False
```

```
# if there is any element pair where certain condition is triu
a = np.array([1,2,3,4])
```

```
b = np.array([4,3,2,1])
print(a, b)
```

```
[1 2 3 4] [4 3 2 1]
```

```
a < b
```

```
array([ True,  True, False, False])
```

```
np.any(a < b)
```

```
True
```

```
a = np.array([1,2,3,4])
b = np.array([4,3,2,1])
```

```
# if this condition is true for ALL the cases
a < b
```

```
array([ True,  True, False, False])
```

```
np.all(a < b)
```

```
False
```

```
a = np.array([1, 2, 3, 2])
b = np.array([2, 2, 3, 2])
c = np.array([6, 4, 4, 5])
np.all((a <= b) & (b <= c))
```

```
True
```

```
# Sorting
```

```
a = np.array([2,30,41,7,17,52])
```

Saving...

```
array([ 2, 30, 41,  7, 17, 52])
```

```
np.sort(a)
```

```
array([ 2,  7, 17, 30, 41, 52])
```

```
np.argsort(a)
```

```
array([0, 3, 4, 1, 2, 5])
```

```
a
```

```
array([ 2, 30, 41,  7, 17, 52])
```

```
# sorting? if axis is not provided, byt default, sort, argsort, sorts the last dime
```

```
a = np.array([[23,4,43],
              [12,89,3],
              [69,420,0]])
```

```
b = np.sort(a)
```

```
b
```

```
array([[  4,  23,  43],
       [  3,  12,  89],
       [  0,  69, 420]])
```

```
# fitbit case
```

```
!gdown 1kXqcJo4YzmwF1G2BPoA17CI49TZVHANF
```

```
Downloading...
From: https://drive.google.com/uc?id=1kXqcJo4YzmwF1G2BPoA17CI49TZVHANF
To: /content/fitness.txt
100% 3.14k/3.14k [00:00<00:00, 4.62MB/s]
```

```
data = np.loadtxt("fitness.txt", dtype="str")
```

```
data.ndim
```

```
2
```

```
# no of rows and no of columns
```

```
data.shape
```

Saving...

```
data
```

```
array([['06-10-2017', '5464', '200', '181', '5', '0', '66'],
       ['07-10-2017', '6041', '100', '197', '8', '0', '66'],
       ['08-10-2017', '25', '100', '0', '5', '0', '66'],
       ['09-10-2017', '5461', '100', '174', '4', '0', '66'],
       ['10-10-2017', '6915', '200', '223', '5', '500', '66'],
       ['11-10-2017', '4545', '100', '149', '6', '0', '66'],
       ['12-10-2017', '4340', '100', '140', '6', '0', '66'],
       ['13-10-2017', '1230', '100', '38', '7', '0', '66'],
       ['14-10-2017', '61', '100', '1', '5', '0', '66'],
       ['15-10-2017', '1258', '100', '40', '6', '0', '65'],
```

```
       ['16-10-2017', '3148', '100', '101', '8', '0', '65'],
       ['17-10-2017', '4687', '100', '152', '5', '0', '65'],
       ['18-10-2017', '4732', '300', '150', '6', '500', '65'],
       ['19-10-2017', '3519', '100', '113', '7', '0', '65'],
       ['20-10-2017', '1580', '100', '49', '5', '0', '65'],
       ['21-10-2017', '2822', '100', '86', '6', '0', '65'],
       ['22-10-2017', '181', '100', '6', '8', '0', '65'],
       ['23-10-2017', '3158', '200', '99', '5', '0', '65'],
       ['24-10-2017', '4383', '200', '143', '4', '0', '64'],
       ['25-10-2017', '3881', '200', '125', '5', '0', '64'],
       ['26-10-2017', '4037', '200', '129', '6', '0', '64'],
       ['27-10-2017', '202', '200', '6', '8', '0', '64'],
       ['28-10-2017', '292', '200', '9', '5', '0', '64'],
       ['29-10-2017', '330', '300', '10', '6', '0', '64'],
       ['30-10-2017', '2209', '200', '72', '5', '0', '64'],
       ['31-10-2017', '4550', '300', '150', '8', '500', '64'],
       ['01-11-2017', '4435', '300', '141', '5', '0', '64'],
       ['02-11-2017', '4779', '300', '156', '4', '0', '64'],
       ['03-11-2017', '1831', '300', '57', '5', '0', '64'],
       ['04-11-2017', '2255', '300', '72', '4', '0', '64'],
       ['05-11-2017', '539', '300', '17', '5', '500', '64'],
       ['06-11-2017', '5464', '300', '181', '4', '0', '64'],
       ['07-11-2017', '6041', '200', '197', '3', '0', '64'],
       ['08-11-2017', '4068', '300', '131', '2', '0', '64'],
       ['09-11-2017', '4683', '300', '154', '9', '0', '64'],
       ['10-11-2017', '4033', '300', '137', '5', '0', '64'],
       ['11-11-2017', '6314', '300', '193', '6', '500', '64'],
       ['12-11-2017', '614', '300', '19', '4', '500', '64'],
       ['13-11-2017', '3149', '300', '101', '5', '500', '64'],
       ['14-11-2017', '4005', '300', '139', '8', '500', '64'],
       ['15-11-2017', '4880', '300', '164', '4', '500', '64'],
       ['16-11-2017', '4136', '300', '137', '5', '500', '64'],
       ['17-11-2017', '705', '300', '22', '6', '500', '64'],
       ['18-11-2017', '570', '200', '17', '5', '500', '64'],
       ['19-11-2017', '269', '300', '9', '6', '500', '64'],
       ['20-11-2017', '4275', '300', '145', '5', '0', '64'],
       ['21-11-2017', '5999', '300', '192', '6', '0', '64'],
       ['22-11-2017', '4421', '300', '146', '5', '0', '64'],
       ['23-11-2017', '6930', '300', '234', '6', '0', '64'],
       ['24-11-2017', '5195', '300', '167', '5', '0', '64'],
       ['25-11-2017', '546', '300', '16', '6', '0', '64'],
       ['26-11-2017', '493', '300', '17', '7', '500', '64'],
       ['27-11-2017', '995', '300', '32', '6', '500', '64'],
       ['28-11-2017', '1163', '200', '35', '7', '500', '64'],
       [                  , '100', '220', '6', '500', '64'],
       [                  , '300', '116', '5', '500', '64'],
       [                  , '300', '23', '6', '500', '64'],
       ['02-12-2017', '1421', '300', '44', '7', '500', '64'],
```

Saving...

```
data.T[0]
```

```
array(['06-10-2017', '07-10-2017', '08-10-2017', '09-10-2017',
       '10-10-2017', '11-10-2017', '12-10-2017', '13-10-2017',
       '14-10-2017', '15-10-2017', '16-10-2017', '17-10-2017',
       '18-10-2017', '19-10-2017', '20-10-2017', '21-10-2017',
       '22-10-2017', '23-10-2017', '24-10-2017', '25-10-2017',
       '26-10-2017', '27-10-2017', '28-10-2017', '29-10-2017',
       '30-10-2017', '31-10-2017', '01-11-2017', '02-11-2017',
       '03-11-2017', '04-11-2017', '05-11-2017', '06-11-2017',
       '07-11-2017', '08-11-2017', '09-11-2017', '10-11-2017',
```

```
       '11-11-2017', '12-11-2017', '13-11-2017', '14-11-2017',
       '15-11-2017', '16-11-2017', '17-11-2017', '18-11-2017',
       '19-11-2017', '20-11-2017', '21-11-2017', '22-11-2017',
       '23-11-2017', '24-11-2017', '25-11-2017', '26-11-2017',
       '27-11-2017', '28-11-2017', '29-11-2017', '30-11-2017',
       '01-12-2017', '02-12-2017', '03-12-2017', '04-12-2017',
       '05-12-2017', '06-12-2017', '07-12-2017', '08-12-2017',
       '09-12-2017', '10-12-2017', '11-12-2017', '12-12-2017',
       '13-12-2017', '14-12-2017', '15-12-2017', '16-12-2017',
       '17-12-2017', '18-12-2017', '19-12-2017', '20-12-2017',
       '21-12-2017', '22-12-2017', '23-12-2017', '24-12-2017',
       '25-12-2017', '26-12-2017', '27-12-2017', '28-12-2017',
       '29-12-2017', '30-12-2017', '31-12-2017', '01-01-2018',
       '02-01-2018', '03-01-2018', '04-01-2018', '05-01-2018',
       '06-01-2018', '07-01-2018', '08-01-2018', '09-01-2018'],
      dtype='<U10')
```

```
date, step_count, mood, calories_burned, hours_of_sleep, activity_status, weight =
```

```
a, b, c = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

```
c
```

```
[7, 8, 9]
```

```
date
```

```
array(['06-10-2017', '07-10-2017', '08-10-2017', '09-10-2017',
       '10-10-2017', '11-10-2017', '12-10-2017', '13-10-2017',
       '14-10-2017', '15-10-2017', '16-10-2017', '17-10-2017',
       '18-10-2017', '19-10-2017', '20-10-2017', '21-10-2017',
       '22-10-2017', '23-10-2017', '24-10-2017', '25-10-2017',
       '26-10-2017', '27-10-2017', '28-10-2017', '29-10-2017',
       '30-10-2017', '31-10-2017', '01-11-2017', '02-11-2017',
       '03-11-2017', '04-11-2017', '05-11-2017', '06-11-2017',
       '07-11-2017', '08-11-2017', '09-11-2017', '10-11-2017',
       '11-11-2017', '12-11-2017', '13-11-2017', '14-11-2017',
       '15-11-2017', '16-11-2017', '17-11-2017', '18-11-2017',
       '19-11-2017', '20-11-2017', '21-11-2017', '22-11-2017',
       '23-11-2017', '24-11-2017', '25-11-2017', '26-11-2017',
                                   2017', '29-11-2017', '30-11-2017',
                                   2017', '03-12-2017', '04-12-2017',
                                   2017', '07-12-2017', '08-12-2017',
       '09-12-2017', '10-12-2017', '11-12-2017', '12-12-2017',
       '13-12-2017', '14-12-2017', '15-12-2017', '16-12-2017',
       '17-12-2017', '18-12-2017', '19-12-2017', '20-12-2017',
       '21-12-2017', '22-12-2017', '23-12-2017', '24-12-2017',
       '25-12-2017', '26-12-2017', '27-12-2017', '28-12-2017',
       '29-12-2017', '30-12-2017', '31-12-2017', '01-01-2018',
       '02-01-2018', '03-01-2018', '04-01-2018', '05-01-2018',
       '06-01-2018', '07-01-2018', '08-01-2018', '09-01-2018'],
      dtype='<U10')
```

Saving...

```
step_count
```

```
array(['5464', '6041', '25', '5461', '6915', '4545', '4340', '1230', '61',
       '1258', '3148', '4687', '4732', '3519', '1580', '2822', '181',
       '3158', '4383', '3881', '4037', '202', '292', '330', '2209',
       '4550', '4435', '4779', '1831', '2255', '539', '5464', '6041',
       '4068', '4683', '4033', '6314', '614', '3149', '4005', '4880',
       '4136', '705', '570', '269', '4275', '5999', '4421', '6930',
       '5195', '546', '493', '995', '1163', '6676', '3608', '774', '1421',
       '4064', '2725', '5934', '1867', '3721', '2374', '2909', '1648',
       '799', '7102', '3941', '7422', '437', '1231', '1696', '4921',
       '221', '6500', '3575', '4061', '651', '753', '518', '5537', '4108',
       '5376', '3066', '177', '36', '299', '1447', '2599', '702', '133',
       '153', '500', '2127', '2203'], dtype='<U10')
```

```
step_count.dtype
```

```
dtype('<U10')
```

```
step_count = np.array(step_count, dtype="int")
step_count.dtype
```

```
dtype('int64')
```

```
step_count
```

```
array([5464, 6041,   25, 5461, 6915, 4545, 4340, 1230,   61, 1258, 3148,
       4687, 4732, 3519, 1580, 2822,  181, 3158, 4383, 3881, 4037,  202,
        292,  330, 2209, 4550, 4435, 4779, 1831, 2255,  539, 5464, 6041,
       4068, 4683, 4033, 6314,  614, 3149, 4005, 4880, 4136,  705,  570,
        269, 4275, 5999, 4421, 6930, 5195,  546,  493,  995, 1163, 6676,
       3608,  774, 1421, 4064, 2725, 5934, 1867, 3721, 2374, 2909, 1648,
        799, 7102, 3941, 7422,  437, 1231, 1696, 4921,  221, 6500, 3575,
       4061,  651,  753,  518, 5537, 4108, 5376, 3066,  177,   36,  299,
       1447, 2599,  702,  133,  153,  500, 2127, 2203])
```

```
calories_burned = np.array(calories_burned, dtype = 'int')

hours_of_sleep = np.array(hours_of_sleep, dtype = 'int')

weight = np.array(weight, dtype = 'int')
```

Saving...

```
array(['200', '100', '100', '100', '200', '100', '100', '100', '100',
       '100', '100', '100', '300', '100', '100', '100', '100', '200',
       '200', '200', '200', '200', '200', '300', '200', '300', '300',
       '300', '300', '300', '300', '300', '200', '300', '300', '300',
       '300', '300', '300', '300', '300', '300', '300', '200', '300',
       '300', '300', '300', '300', '300', '300', '300', '300', '200',
       '100', '300', '300', '300', '300', '300', '300', '300', '100',
       '200', '200', '100', '100', '200', '200', '300', '200', '200',
       '100', '200', '100', '200', '200', '100', '100', '100', '100',
       '300', '200', '300', '200', '100', '100', '100', '200', '200',
       '100', '100', '300', '200', '200', '300'], dtype='<U10')
```

```
np.unique(mood)
```

```
array(['100', '200', '300'], dtype='<U10')
```

```
# 100->Sad, 200->Neurtal, 300-->Happy
```

```
mood[mood == "300"] = "Happy"
```

```
mood[mood == '200'] = 'Neutral'
mood[mood == '100'] = 'Sad'
```

```
mood # data-munging, data-manipulation, data-cleaning
```

```
array(['Neutral', 'Sad', 'Sad', 'Sad', 'Neutral', 'Sad', 'Sad', 'Sad',
       'Sad', 'Sad', 'Sad', 'Sad', 'Happy', 'Sad', 'Sad', 'Sad', 'Sad',
       'Neutral', 'Neutral', 'Neutral', 'Neutral', 'Neutral', 'Neutral',
       'Happy', 'Neutral', 'Happy', 'Happy', 'Happy', 'Happy', 'Happy',
       'Happy', 'Happy', 'Neutral', 'Happy', 'Happy', 'Happy', 'Happy',
       'Happy', 'Happy', 'Happy', 'Happy', 'Happy', 'Happy', 'Neutral',
       'Happy', 'Happy', 'Happy', 'Happy', 'Happy', 'Happy', 'Happy',
       'Happy', 'Happy', 'Neutral', 'Sad', 'Happy', 'Happy', 'Happy',
       'Happy', 'Happy', 'Happy', 'Happy', 'Sad', 'Neutral', 'Neutral',
       'Sad', 'Sad', 'Neutral', 'Neutral', 'Happy', 'Neutral', 'Neutral',
       'Sad', 'Neutral', 'Sad', 'Neutral', 'Neutral', 'Sad', 'Sad', 'Sad',
       'Sad', 'Happy', 'Neutral', 'Happy', 'Neutral', 'Sad', 'Sad', 'Sad',
       'Neutral', 'Neutral', 'Sad', 'Sad', 'Happy', 'Neutral', 'Neutral',
       'Happy'], dtype='<U10')
```

```
activity_status
```

```
array(['0', '0', '0', '0', '500', '0', '0', '0', '0', '0', '0', '0',
       '500', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0',
       '500', '0', '0', '0', '0', '500', '0', '0', '0', '0', '0', '500',
       '500', '500', '500', '500', '500', '500', '500', '500', '0', '0',
       '0', '0', '0', '0', '500', '500', '500', '500', '500', '500',
       '500', '500', '500', '500', '500', '500', '0', '500', '500', '0',
       '500', '500', '500', '500', '500', '0', '500', '500', '500', '500',
       '0', '0', '0', '0', '500', '500', '500', '500', '0', '0', '0', '0',
       '0', '0', '0', '0', '500', '0', '500'], dtype='<U10')
```

Saving...

```
= '500'] = 'Active'
activity_status[activity_status == '0'] = 'Inactive'
```

```
activity_status
```

```
array(['Inactive', 'Inactive', 'Inactive', 'Inactive', 'Active',
       'Inactive', 'Inactive', 'Inactive', 'Inactive', 'Inactive',
       'Inactive', 'Inactive', 'Active', 'Inactive', 'Inactive',
       'Inactive', 'Inactive', 'Inactive', 'Inactive', 'Inactive',
       'Inactive', 'Inactive', 'Inactive', 'Inactive', 'Inactive',
       'Active', 'Inactive', 'Inactive', 'Inactive', 'Inactive', 'Active',
       'Inactive', 'Inactive', 'Inactive', 'Inactive', 'Inactive',
       'Active', 'Active', 'Active', 'Active', 'Active', 'Active',
```

```
       'Active', 'Active', 'Active', 'Inactive', 'Inactive', 'Inactive',
       'Inactive', 'Inactive', 'Inactive', 'Active', 'Active', 'Active',
       'Active', 'Active', 'Active', 'Active', 'Active', 'Active',
       'Active', 'Active', 'Active', 'Inactive', 'Active', 'Active',
       'Inactive', 'Active', 'Active', 'Active', 'Active', 'Active',
       'Inactive', 'Active', 'Active', 'Active', 'Active', 'Inactive',
       'Inactive', 'Inactive', 'Inactive', 'Active', 'Active', 'Active',
       'Active', 'Inactive', 'Inactive', 'Inactive', 'Inactive',
       'Inactive', 'Inactive', 'Inactive', 'Inactive', 'Active',
       'Inactive', 'Active'], dtype='<U10')
```

```
# insights from the data
```

```
# what is the average step count - step_count
```

```
np.mean(step_count)
```

```
2935.9375
```

```
# On which date the step count was highest? step_count, date
```

```
date[np.argmax(step_count)]
```

```
'14-12-2017'
```

```
# What is the most frequent mood? ---> if given counts of all the mood, calculate m
```

```
len(mood[mood == "Sad"])
```

```
29
```

```
len(mood[mood == "Happy"])
```

```
40
```

Saving...

```
np.unique(mood)
```

```
array(['Happy', 'Neutral', 'Sad'], dtype='<U10')
```

```
np.unique(mood, return_counts=True)
```

```
(array(['Happy', 'Neutral', 'Sad'], dtype='<U10'), array([40, 27, 29]))
```

```
# Does the step_count changes with mood?
```

```
np.mean(step_count[mood == "Happy"])
```

```
3392.725
```

```
np.mean(step_count[mood == "Neutral"])
```

```
3153.777777777778
```

```
np.mean(step_count[mood == "Sad"])
```

```
2103.0689655172414
```

```
np.unique(mood[step_count > 4000], return_counts = True)
```

```
(array(['Happy', 'Neutral', 'Sad'], dtype='<U10'), array([22,  9,  7]))
```

```
np.unique(mood[step_count < 2000], return_counts = True)
```

```
(array(['Happy', 'Neutral', 'Sad'], dtype='<U10'), array([13,  8, 18]))
```

```
# Operations on arrays in Numpy
```

```
m1 = np.arange(12).reshape(3, 4)
m1
```

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

```
m1 + 2
```

```
array([[ 2,  3,  4,  5],
       [ 6,  7,  8,  9],
       [10, 11, 12, 13]])
```

Saving...

```
a + b
```

```
array([3, 4, 6])
```

```
a * b
```

```
array([2, 4, 9])
```

```
a = np.array([0,2,3])
b = np.array([1,3,5])
a >= b
```

```
array([False, False, False])
```

```
A = np.arange(12).reshape(3,4)
B = np.arange(12).reshape(3,4)
```

```
A * B # not matrix muliplication
```

```
array([[  0,   1,   4,   9],
       [ 16,  25,  36,  49],
       [ 64,  81, 100, 121]])
```

```
#(3, 4) (4, 3)
```

```
#(5, 6)X(6, 8)
```

```
B = B.reshape(4, 3)
```

```
B
```

```
array([[ 0,  1,  2],
       [ 3,  4,  5],
       [ 6,  7,  8],
       [ 9, 10, 11]])
```

```
A
```

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

```
A * B
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-129-a4cedde81ed0> in <module>()
```

Saving...

```
ot be broadcast together with shapes (3,4) (4,3)
```

SEARCH STACK OVERFLOW

```
np.matmul(A, B)
```

```
array([[ 42,  48,  54],
       [114, 136, 158],
       [186, 224, 262]])
```

```
A @ B
```

```
array([[ 42,  48,  54],
       [114, 136, 158],
       [186, 224, 262]])
```

```
B @ A
```

```
array([[ 20,  23,  26,  29],
       [ 56,  68,  80,  92],
       [ 92, 113, 134, 155],
       [128, 158, 188, 218]])
```

```
np.dot(A, B)
```

```
array([[ 42,  48,  54],
       [114, 136, 158],
       [186, 224, 262]])
```

```
a= np.array([1,2,3])
b = np.array([1,1,1])
np.dot(a, b)
```

```
6
```

```
A = np.arange(12).reshape(3, 4)
A
```

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

```
a = np.array([1, 2, 3])
```

```
A @ a
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-139-08cade3665aa> in <module>()
```

Saving...

```
ValueError: matmul: Input operand 1 has a mismatch in its core dimension 0,
with gufunc signature (n?,k),(k,m?)->(n?,m?) (size 3 is different from 4)
```

SEARCH STACK OVERFLOW

```
a @ A
```

```
array([32, 38, 44, 50])
```

```
A * a
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-141-920aa4e58700> in <module>()
----> 1 A * a

ValueError: operands could not be broadcast together with shapes (3,4) (3,)
```

SEARCH STACK OVERFLOW

```
a * A
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-142-d157f8e14faf> in <module>()
----> 1 a * A

ValueError: operands could not be broadcast together with shapes (3,) (3,4)
```

SEARCH STACK OVERFLOW

```
A * 2 # Vectorisation
```

```
array([[ 0,  2,  4,  6],
       [ 8, 10, 12, 14],
       [16, 18, 20, 22]])
```

```
# unfuncs provided by numpy
```

```
np.log(np.arange(1, 100))
```

```
array([0.        , 0.69314718, 1.09861229, 1.38629436, 1.60943791,
       1.79175947, 1.94591015, 2.07944154, 2.19722458, 2.30258509,
       2.39789527, 2.48490665, 2.56494936, 2.63905733, 2.7080502 ,
       2.77258872, 2.83321334, 2.89037176, 2.94443898, 2.99573227,
       3.04452244, 3.09104245, 3.13549422, 3.17805383, 3.21887582,
       3.25809654, 3.29583687, 3.33220451, 3.36729583, 3.40119738,
       3.4339872 , 3.4657359 , 3.49650756, 3.52636052, 3.55534806,
       3.58351894, 3.61091791, 3.63758616, 3.66356165, 3.68887945,
       3.71357207, 3.73766962, 3.76120012, 3.78418963, 3.80666249,
       3.8286414 , 3.8501476 , 3.87120101, 3.8918203 , 3.91202301,
                                2, 3.97029191, 3.98898405, 4.00733319,
                                7, 4.06044301, 4.07753744, 4.09434456,
                                9, 4.14313473, 4.15888308, 4.17438727,
       4.18965474, 4.20469262, 4.21950771, 4.2341065 , 4.24849524,
       4.26267988, 4.27666612, 4.29045944, 4.30406509, 4.31748811,
       4.33073334, 4.34380542, 4.35670883, 4.36944785, 4.38202663,
       4.39444915, 4.40671925, 4.41884061, 4.4308168 , 4.44265126,
       4.4543473 , 4.46590812, 4.47733681, 4.48863637, 4.49980967,
       4.51085951, 4.52178858, 4.53259949, 4.54329478, 4.55387689,
       4.56434819, 4.57471098, 4.58496748, 4.59511985])
```

Saving...

```
import math
```

```
math.log(10)
```

```
2.302585092994046
```

```
math.log([10, 100, 1000])
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-151-b8a53373c6ea> in <module>()
----> 1 math.log([10, 100, 1000])

TypeError: must be real number, not list
```

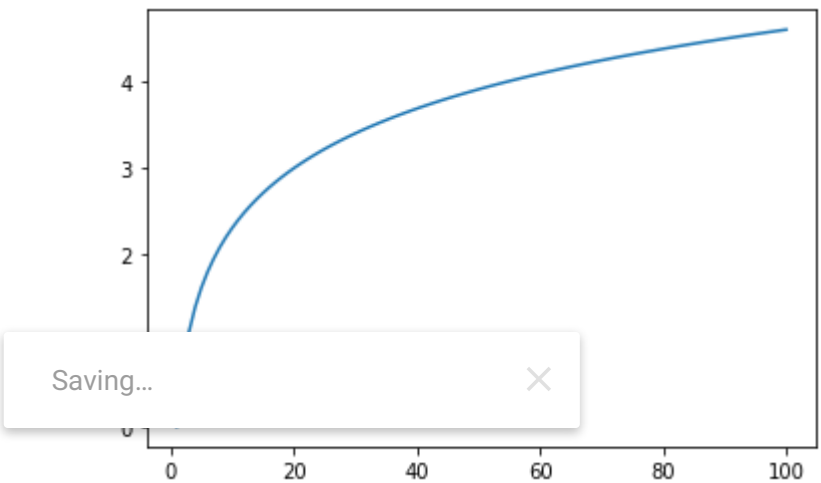
SEARCH STACK OVERFLOW

```
np_log = np.vectorize(math.log)
```

```
np_log([10, 100, 1000])
```

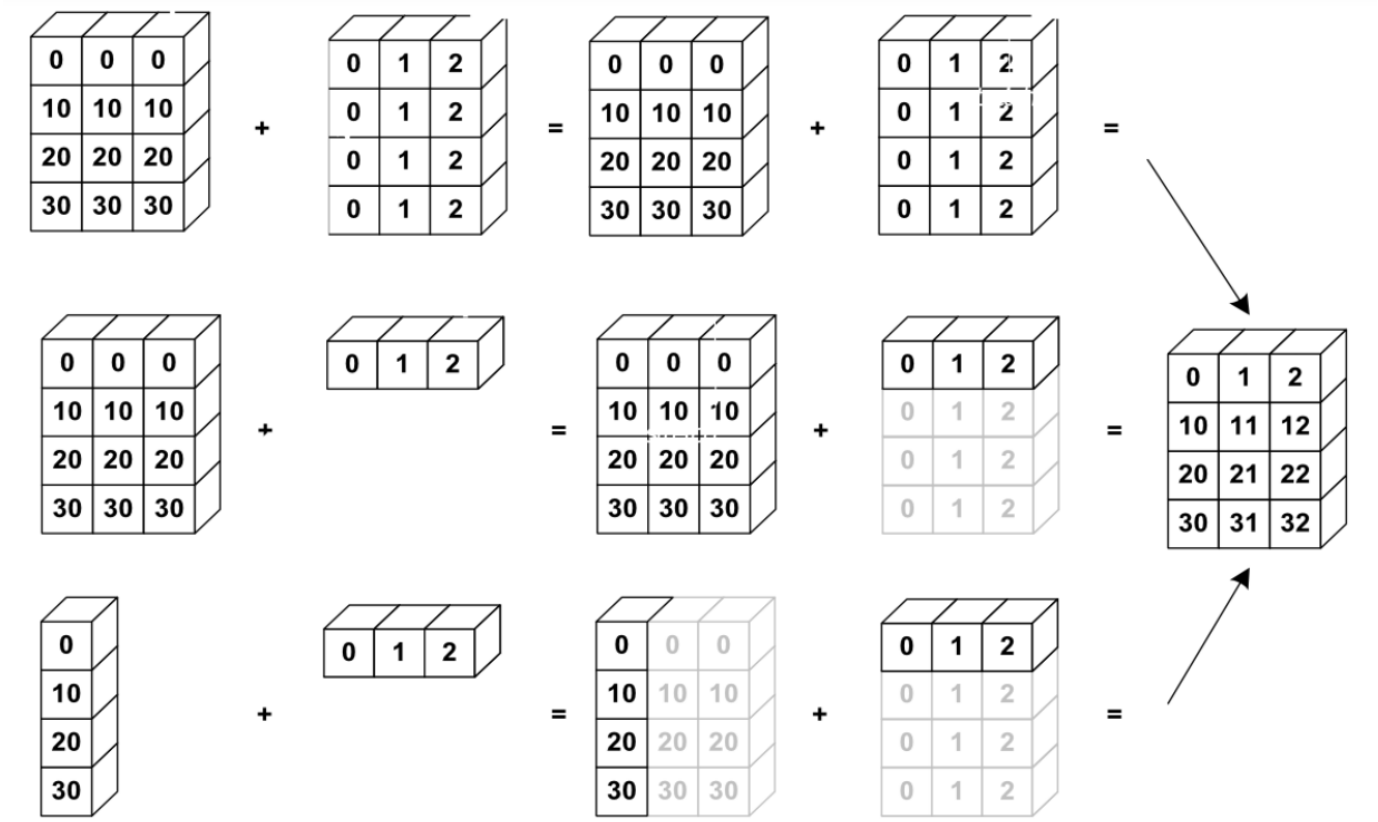
```
array([2.30258509, 4.60517019, 6.90775528])
```

```
import math
import matplotlib.pyplot as plt

x = np.arange(1, 101)
y = np.vectorize(math.log)(x)

plt.plot(x, y)
plt.show()
```



Saving...

```
# Broadcasting
```




```
a = np.tile(np.arange(0, 40, 10), (3, 1))
```

```
np.tile(np.arange(0, 40, 10), (3, 2))
```

```
array([[ 0, 10, 20, 30,  0, 10, 20, 30],
       [ 0, 10, 20, 30,  0, 10, 20, 30],
       [ 0, 10, 20, 30,  0, 10, 20, 30]])
```

```
a = a.T
```

```
a
```

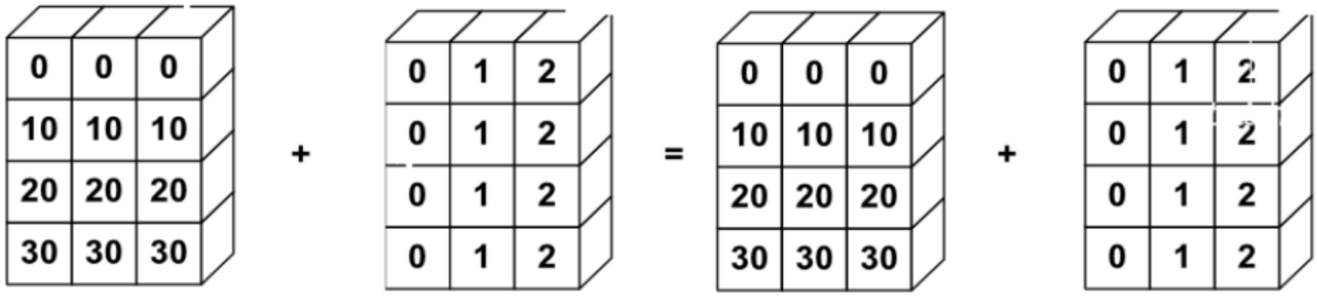
```
array([[ 0,  0,  0],
```

Saving...

```
       [30, 30, 30]])
```

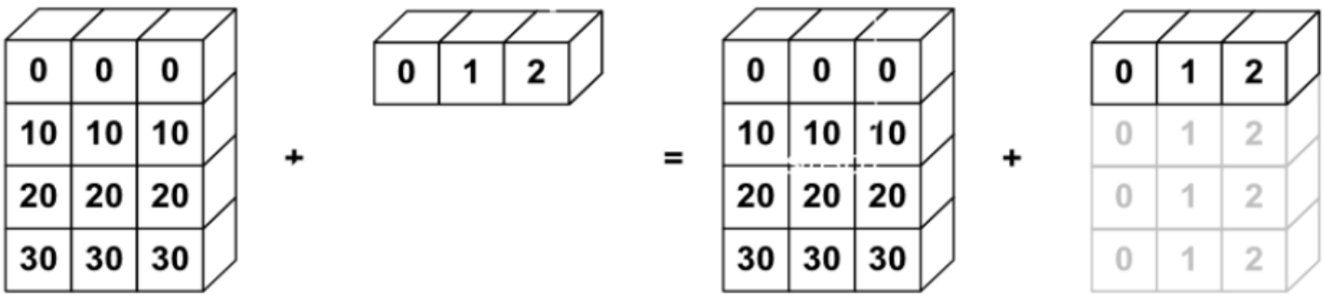
```
b = np.tile(np.arange(0,3), (4,1))
b
```

```
array([[0, 1, 2],
       [0, 1, 2],
       [0, 1, 2],
       [0, 1, 2]])
```




```
a + b
```

```
array([[ 0,  1,  2],
       [10, 11, 12],
       [20, 21, 22],
       [30, 31, 32]])
```




```
a
```

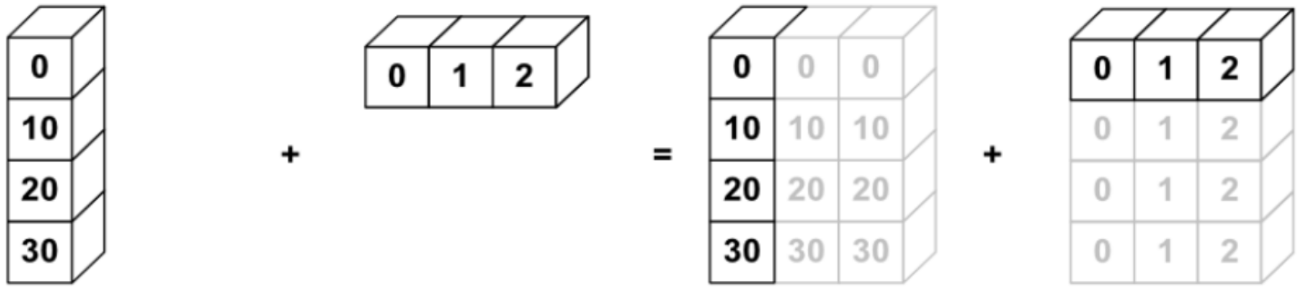
```
array([[ 0,  0,  0],
       [10, 10, 10],
       [20, 20, 20],
       [30, 30, 30]])
```

```
b = np.array([0,1,2])
```

Saving...

```
a + b
```

```
array([[ 0,  1,  2],
       [10, 11, 12],
       [20, 21, 22],
       [30, 31, 32]])
```




```
np.arange(0, 40, 10).reshape(4, 1)
```

```
array([[ 0],
       [10],
       [20],
       [30]])
```

```
np.arange(0, 40, 10)[:, np.newaxis]
```

```
array([[ 0],
       [10],
       [20],
       [30]])
```

```
a = np.arange(0, 40, 10).reshape(4, 1)
a
```

```
array([[ 0],
       [10],
       [20],
       [30]])
```

```
b = np.arange(0, 3)
b
```

```
array([0, 1, 2])
```

Saving...

```
array([[ 0,  1,  2],
       [10, 11, 12],
       [20, 21, 22],
       [30, 31, 32]])
```

```
A = np.arange(1,10).reshape(3,3) # (3, 3)
B = np.array([-1, 0, 1]) #(1, 3)
A * B
```

```
array([[-1,  0,  3],
       [-4,  0,  6],
```

```
       [-7,  0,  9]])
```

```
A = np.arange(1,10).reshape(3,3) #(3,3)
B = np.arange(3, 10, 3).reshape(3,1) #(3,1)
C = A + B
```

```
A = np.arange(12).reshape(3, 4) # 3, 4
B = np.array([1, 2, 3]) # 1, 3
A + B
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-187-0e3264f58792> in <module>()
      1 A = np.arange(12).reshape(3, 4) # 3, 4
      2 B = np.array([1, 2, 3]) # 1, 3
----> 3 A + B

ValueError: operands could not be broadcast together with shapes (3,4) (3,)
```

SEARCH STACK OVERFLOW

```
# split, vsplit, stack,. hstack, 3d arrays
```

```
# (x1, y1) (x2, y2)
```

Saving...

0s completed at 23:57

Saving...