LAB Logbook

Ela Dogruyol - 2351343

Lab 1

For the Lab 1 in Week 1, students were asked to create a vector using np.arange method and doing some changes on that vector to be able to practice NumPy and Python.

Firstly, because my student ID is 2351343, I created a vector of 43 elements. Secondly, I changed this matrix into a 2-d array with 1 row using **reshape** method. Thirdly, I used NumPy's **empty_like** method and **slicing** to be able to create an independent array and save the values of the matrix to that independent array. I checked the **shape** attribute values of both matrixes. I printed all results at the end of the steps.

My code and results:

Week 1 Assignment

Name: Ela Dogruyol

Student ID: 2351343

Installation process is done in the below cell.

In [1]:
```
import numpy as np
```

1) A vector that has 43 elements is created with np.arange method.

In [22]:
```
vector = np.arange(43)
print(vector)
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42]
```

2) Matrix is changed into a to 2-d array with 1 row.

In [23]:
```
vector = vector.reshape(43,1)
print(vector)
```

```
[[ 0]
 [ 1]
 [ 2]
 [ 3]
 [ 4]
 [ 5]
 [ 6]
 [ 7]
 [ 8]
 [ 9]
 [10]
 [11]
 [12]
 [13]
 [14]
 [15]
 [16]
 [17]
 [18]
 [19]
 [20]
 [21]
 [22]
 [23]
 [24]
 [25]
 [26]
 [27]
 [28]
 [29]
 [30]
 [31]
 [32]
 [33]
 [34]
 [35]
 [36]
 [37]
 [38]
 [39]
 [40]
 [41]
 [42]]
```

3) The constructed array is saved into another array.

In [24]:
```
new_array_2d = np.empty_like(vector)
new_array_2d[:,:] = vector
print(new_array_2d)
```

```
[[ 0]
 [ 1]
 [ 2]
 [ 3]
 [ 4]
 [ 5]
 [ 6]
 [ 7]
 [ 8]
 [ 9]
 [10]
 [11]
 [12]
 [13]
 [14]
 [15]
 [16]
 [17]
 [18]
 [19]
 [20]
 [21]
 [22]
 [23]
 [24]
 [25]
 [26]
 [27]
 [28]
 [29]
 [30]
 [31]
 [32]
 [33]
 [34]
 [35]
 [36]
 [37]
 [38]
 [39]
 [40]
 [41]
 [42]]
```

4) Shape attribute value is checked for both arrays.

In [26]:
```
print(vector.shape)
print(new_array_2d.shape)
```

```
(43, 1)
(43, 1)
```

Lab 2

For the Lab 2 in Week 2, Pandas and its main functions are studied. According to requirements, "adult_data_mini.csv" is used and some operations is done.

Firstly, n is determined as 3 (n=3) because of my student ID. Secondly, data is grouped by "relationship" and "hours-per-week". In other words, "relationship" column is grouped based on the "hours-per-week" column values. Thirdly, "hours-per-week" column values is reduced by n=3. At this step, the function "change_data(x)" is created and used. To apply this function to the dataset, apply() method is used and original DataFrame is updated. Lastly, grouping by "relationship" and reduced "hours-per-week" operation is done again.

My code and results:

1) Data is grouped by 'relationship' and 'hours-per-week'. (group 'relationship' based on 'hours-per-week')

In [49]:
```
group_by_hours = data.groupby(['relationship', 'hours-per-week'])
group_by_hours.size()
```

Out[49]:
```
relationship    hours-per-week
 Husband        13.0             1
                40.0             2
                45.0             1
                80.0             1
 Not-in-family  16.0             1
                40.0             2
                50.0             2
 Own-child      30.0             1
 Wife           40.0             2
dtype: int64
```

2) In order to change values of the original DataFrame, a function is created. Then, all values of 'hours-per-week' is reduced by 3 (because n=3).

In [50]:
```
def change_data(x):
    return x - 3

data['hours-per-week'] = data['hours-per-week'].apply(change_data)
data
```

Out[50]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	Answer	IsHo
0	39	State-gov	77516.0	Bachelors	13.0	Never-married	Adm-clerical	Not-in-family	White	Male	2174.0	NaN	37.0	United-States	<=50K	
1	50	Self-emp-not-inc	83311.0	Bachelors	13.0	Married-civ-spouse	Exec-managerial	Husband	White	Male	0.0	0.0	10.0	United-States	<=50K	
2	38	Private	215646.0	HS-grad	9.0	Divorced	Handlers-cleaners	Not-in-family	White	Male	0.0	NaN	37.0	United-States	<=50K	
3	53	Private	234721.0	11th	7.0	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0.0	NaN	37.0	United-States	<=50K	
4	28	Private	338409.0	Bachelors	13.0	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0.0	NaN	37.0	Cuba	<=50K	
5	37	Private	284582.0	Masters	14.0	Married-civ-spouse	Exec-managerial	Wife	White	Female	0.0	NaN	37.0	United-States	<=50K	
6	49	Private	160187.0	9th	5.0	Married-spouse-absent	Other-service	Not-in-family	Black	Female	0.0	0.0	13.0	Jamaica	<=50K	
7	52	Self-emp-not-inc	209642.0	HS-grad	9.0	Married-civ-spouse	Exec-managerial	Husband	White	Male	0.0	0.0	42.0	United-States	>50K	
8	31	Private	45781.0	Masters	14.0	Never-married	Prof-specialty	Not-in-family	White	Female	14084.0	NaN	47.0	United-States	>50K	
10	37	Private	280464.0	Some-college	10.0	Married-civ-spouse	Exec-managerial	Husband	Black	Male	0.0	NaN	77.0	United-States	>50K	
12	23	Private	122272.0	Bachelors	13.0	Never-married	Adm-clerical	Own-child	White	Female	0.0	NaN	27.0	United-States	<=50K	
13	32	Private	205019.0	Assoc-acdm	12.0	Never-married	Sales	Not-in-family	Black	Male	0.0	NaN	47.0	United-States	<=50K	
14	40	Private	121772.0	Assoc-voc	11.0	Married-civ-spouse	Craft-repair	Husband	Asian-Pac-Islander	Male	0.0	NaN	37.0	?	>50K	
15	25	Private	NaN	Some-college	NaN	NaN	NaN	NaN	White	Male	0.0	NaN	NaN	NaN	NaN	

3) Grouping is done again with 'relationship' and reduced 'hours-per-week'.

In [51]:
```
group_by_reduced_hours = data.groupby(['relationship', 'hours-per-week'])
group_by_reduced_hours.size()
```

Out[51]:
```
relationship    hours-per-week
 Husband         10.0             1
                 37.0             2
                 42.0             1
                 77.0             1
 Not-in-family   13.0             1
                 37.0             2
                 47.0             2
 Own-child       27.0             1
 Wife            37.0             2
dtype: int64
```

Lab 3

Lab 4

Lab 5

Lab 6

Lab 7

Lab 8

Lab 9

Lab 10

Lab 11

Lab 12