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

Task 1: NdArray

Create a list and 1-dimensional numpy array with identical elements and show that slicing in both is the same.

In [5]:
```
x = ["a", "b", "c", "d", "e"]
y = np.array(["a", "b", "c", "d", "e"])

print("List slice: ", x[1:])
print("NumPy array slice: ", y[1:])
```

```
List slice:  ['b', 'c', 'd', 'e']
NumPy array slice:  ['b' 'c' 'd' 'e']
```

Choose 3 movies and create 2, 3x4 numpy arrays as follows:

The first numpy array called cast contains the name of 4 actors/actresses from each movie.

The second numpy array called movies contains the name of the movie corresponding to each cast.

Calculate the boolean indexing for one of the movies in the second numpy array.

In [10]:
```
cast = np.array([["Samuel L. Jackson", "John Travolta", "Uma Thurman", "Bruce Willis"],
                 ["Timothee Chalamet", "Oscar Isaac", "Jason Momoa", "Javier Bardem"],
                 ["Florence Pugh", "Saoirse Ronan", "Emma Watson", "Laura Dern"]])

movies = np.array(["Pulp Fiction", "Dune", "Little Women"])

# Boolean Index for the movie "Little Women"
a = movies[2]
index = (movies == a)
print("Boolean Index for 'Little Women':", index)
```

```
Boolean Index for 'Little Women': [False False  True]
```

Using the boolean indexing from the previous step, select the actors/actresses that played in that movie and put it in a new numpy array called selected_movie.

In [15]:
```
selected_movie = cast[index]
print("Cast of 'Little Women':", selected_movie)
```

```
Cast of 'Little Women': [['Florence Pugh' 'Saoirse Ronan' 'Emma Watson' 'Laura Dern']]
```

In [18]:
```
# Confirming all three are numpy arrays
print(type(selected_movie))
print(type(cast))
print(type(movies))
```

```
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
```

Save all of the three numpy arrays from the previous steps to disk.

In [19]:
```
np.savez("task_1_arrays.npz", cast = cast, movies = movies, selected_movie = selected_movie)
```

Reload the saved file and load the three numpy arrays to three new variables called cast_new, movies_new and selected_movie_new.

In [27]:
```
cast_new, movies_new, selected_movie_new = np.load('task_1_arrays.npz').values()
```

In [33]:
```
print("1. cast_new array:", cast_new)
print("2. movies_new array:", movies_new)
print("3. selected_movie_new array:", selected_movie_new)
```

```
1. cast_new array: [['Samuel L. Jackson' 'John Travolta' 'Uma Thurman' 'Bruce Willis']
 ['Timothee Chalamet' 'Oscar Isaac' 'Jason Momoa' 'Javier Bardem']
 ['Florence Pugh' 'Saoirse Ronan' 'Emma Watson' 'Laura Dern']]
2. movies_new array: ['Pulp Fiction' 'Dune' 'Little Women']
3. selected_movie_new array: [['Florence Pugh' 'Saoirse Ronan' 'Emma Watson' 'Laura Dern']]
```

Task 2: Pandas Basics

Create a Series object as follows:

The Series object contains at least 7 car models with the index being car brands like 'BMW' , e.g. data being X3 and the index is BMW.

In [41]:
```
car = pd.Series(["X3", "A4", "Compass", "Altima", "Corolla", "Model 3", "Mustang", "911"],
                index = ["BMW", "Audi", "Jeep", "Nissan", "Toyota", "Tesla", "Ford", "Porsche"])
print(car)
```

```
BMW              X3
Audi             A4
Jeep        Compass
Nissan       Altima
Toyota      Corolla
Tesla       Model 3
Ford        Mustang
Porsche         911
dtype: object
```

Use boolean filtering to filter cars containing the letter 'a'

In [42]:
```
car.str.contains("a")
```

Out[42]:
```
BMW        False
Audi       False
Jeep        True
Nissan      True
Toyota      True
Tesla      False
Ford        True
Porsche    False
dtype: bool
```

Rename the index column to "manufacturer."

In [44]:
```
car.index.name = "manufacturer"
print(car)
```

```
manufacturer
BMW              X3
Audi             A4
Jeep        Compass
Nissan       Altima
Toyota      Corolla
Tesla       Model 3
Ford        Mustang
Porsche         911
dtype: object
```

Task 3: DataFrame

Create a DataFrame as follows:

The DataFrame columns are "fruit" (string), "number_of_seeds" (integer), "edible" (boolean) and "season" (one of the values for 'summer', 'fall' or 'spring'), and the DataFrame contains at least 5 rows.

In [46]:
```
x = {'fruit': ["apple", "orange", "banana", "strawberry", "blueberry"],
     'number_of_seeds': [1, 2, 3, 4 , 5],
     'edible': [True, True, True, True, True],
     'season': ['spring', "fall", "summer", "spring", "fall"]}

df = pd.DataFrame(x)
```

Add a new column to the DataFrame called "Color" and show that all the values for the column are NaN after adding the column

In [56]:
```
new_df = pd.DataFrame(df, columns = ["fruit", "number_of_seeds", "edible", "season", "Color"])

new_df
```

Out[56]:

	fruit	number_of_seeds	edible	season	Color
0	apple	1	True	spring	NaN
1	orange	2	True	fall	NaN
2	banana	3	True	summer	NaN
3	strawberry	4	True	spring	NaN
4	blueberry	5	True	fall	NaN

Set all of the values for the "Color" column to "Unknown"

In [58]:
```
new_df["Color"] = "Unknown"

new_df
```

Out[58]:

	fruit	number_of_seeds	edible	season	Color
0	apple	1	True	spring	Unknown
1	orange	2	True	fall	Unknown
2	banana	3	True	summer	Unknown
3	strawberry	4	True	spring	Unknown
4	blueberry	5	True	fall	Unknown

Use slicing properly to extract "number_of_seeds" and "fruit" for 2 rows.

In [67]:
```
new_df.iloc[0:2,0:2]
```

Out[67]:

	fruit	number_of_seeds
0	apple	1
1	orange	2

Write a function called "seed_count_calculator" and use the apply() method to sum the number_of_seeds row. (do not use the built-in sum functions for this part)

In [78]:
```
def seed_count_calculator(seed):
    count = 0
    for i in seed:
        count += i
    return count
```

In [86]:
```
seeds = new_df[["number_of_seeds"]].apply(seed_count_calculator)
print(seeds)
```

```
number_of_seeds    15
dtype: int64
```

Try to sort the DataFrame by fruit names.

In [87]:
```
new_df.sort_values(by = "fruit")
```

Out[87]:

	fruit	number_of_seeds	edible	season	Color
0	apple	1	True	spring	Unknown
2	banana	3	True	summer	Unknown
4	blueberry	5	True	fall	Unknown
1	orange	2	True	fall	Unknown
3	strawberry	4	True	spring	Unknown

Find both the index and name of the fruit with the maximum amount of seeds.

In [97]:
```
max_index = new_df["number_of_seeds"].idxmax()
max_fruit = new_df.loc[max_index, "fruit"]
print("Index:", max_index)
print("Fruit:", max_fruit)
```

```
Index: 4
Fruit: blueberry
```

Save the DataFrame to disk as a CSV file.

In [99]:
```
new_df.to_csv("DataFrame.csv")
```

Reload the saved CSV in a new variable.

In [101…
```
reload_csv = pd.read_csv("DataFrame.csv")

reload_csv
```

Out[101]:

	Unnamed: 0	fruit	number_of_seeds	edible	season	Color
0	0	apple	1	True	spring	Unknown
1	1	orange	2	True	fall	Unknown
2	2	banana	3	True	summer	Unknown
3	3	strawberry	4	True	spring	Unknown
4	4	blueberry	5	True	fall	Unknown