Task 1: Fundamentals

Create an array with a different length and two arrays of the same length after importing NumPy (3 arrays in total).

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

In [4]:
```
a = np.array([1,2,3,4,5,6,7,8,9,10])
b = np.array([1,2,3,4,5])
c = np.array([6,7,8,9,10])
```

Execute at least four different operations on arrays of the same size.

In [6]:
```
b + c
```

Out[6]:
```
array([ 7,  9, 11, 13, 15])
```

In [7]:
```
b - c
```

Out[7]:
```
array([-5, -5, -5, -5, -5])
```

In [8]:
```
b ** c
```

Out[8]:
```
array([      1,     128,    6561,  262144, 9765625])
```

In [9]:
```
c / b
```

Out[9]:
```
array([6.        , 3.5       , 2.66666667, 2.25      , 2.        ])
```

Create four variables containing a string, integer, float, and boolean and try to cast them to (in order) boolean, float, integer, and string.

In [10]:
```
string = "Hello World!"
integer = 15
float_var = 15.23
boolean = True
```

In [39]:
```
x = bool(string)
```

In [40]:
```
type(x)
```

Out[40]:
```
bool
```

In [42]:
```
x = float(integer)
```

In [43]:
```
type(x)
```

Out[43]:
```
float
```

In [44]:
```
x = int(float_var)
```

In [45]:
```
type(x)
```

Out[45]:
```
int
```

In [49]:
```
x = bool(string)
```

In [50]:
```
type(x)
```

Out[50]:
```
bool
```

Task 2: Control Flow

Create an array containing the first 10 letters of the alphabet. Iterate through the array using a for loop. Inside the for loop, use an if statement to print the word "four" instead of the letter "d" and "eight" instead of the letter "i", otherwise print the alphabet letter.

In [52]:
```
array = np.array(["a", "b", "c", "d", "e", "f", "g", "h", "i", "j"])
```

In [55]:
```
for i in array:
    if i == "d":
        print("four")
    elif i == "i":
        print("eight")
    else:
        print(i)
```

```
a
b
c
four
e
f
g
h
eight
j
```

Task 3: Data Structure

Create a tuple containing 4 strings and a list containing 7 strings. Unpack the tuple into 4 different variables and print them. Then try to remove an element from the list and instead insert one of the variables that were the result of unpacking the tuple. Finally, slice and sort the last 3 elements of the list.

In [58]:
```
x = ("alpha", "beta", "gamma", "delta")
y = ["epsilon", "zeta", "eta", "theta", "iota", "kappa", "lambda"]
```

In [61]:
```
i, j, k , l = x
print(i)
print(j)
print(k)
print(l)
```

```
alpha
beta
gamma
delta
```

In [62]:
```
y.remove("epsilon")
print(y)
```

```
['zeta', 'eta', 'theta', 'iota', 'kappa', 'lambda']
```

In [63]:
```
y.append(i)
print(y)
```

```
['zeta', 'eta', 'theta', 'iota', 'kappa', 'lambda', 'alpha']
```

In [65]:
```
y[4:]
```

Out[65]:
```
['kappa', 'lambda', 'alpha']
```

Create a dictionary with these keys: 'numbers', 'letters', 'words', 'sentences', 'dates', 'times' and 'months' and for each key, assign a value of that specific type. For instance, you should create a temporary small list to assign to the 'numbers' key, a temporary small string to assign to the 'sentences' key, and a temporary small tuple to assign to the 'dates' key.

In [67]:
```
dictionary = {
    "numbers": [5, 10, 15, 20],
    "letters": ["a", "b", "c", "d"],
    "words": ["hello", "these", "are", "words"],
    "sentences": "this is a sentence",
    "dates": ("5/16/23", "5/17/23", "5/18/23", "5/19/23"),
    "times": ("7:00 AM", "3:00 PM", "7:00 PM", "11:00 PM"),
    "months": ("January", "February", "March", "April")
}
```

Task 4: Sequence Function

Create a list containing numbers from 11 to 16, then using a for loop and proper indexing, produce the following output:

12 --> 11

13 --> 12

14 --> 13

15 --> 14

16 --> 15

17 --> 16

In [95]:
```
a = [11, 12, 13, 14, 15, 16]
print(a)
```

```
[11, 12, 13, 14, 15, 16]
```

In [100…
```
for i in range(len(a)):
    b = a[i] + 1
    c = a[i]
    print(b, "-->", c)
```

```
12 --> 11
13 --> 12
14 --> 13
15 --> 14
16 --> 15
17 --> 16
```

Task 5: Copying

By copying the previous list, create a new variable containing the same numbers from 11 to 16 elements from the previous list. Change an element in one of the variables and make sure the other doesn't change.

In [101…
```
a
new_a = a.copy()
print(new_a)
```

```
[11, 12, 13, 14, 15, 16]
```

In [102…
```
new_a[3] = 100
```

In [103…
```
print(new_a)
```

```
[11, 12, 13, 100, 15, 16]
```

Task 6: Exception Handling

Create a variable containing a tuple object. Then try to change its first element the way you change an element in a list and see what kind of error you will get. Then try to catch handle the error and instead of the error print "Tuples are immutable :)"

In [104…
```
tup_obj = (100, 200, 300, 400, 500)
```

In [107…
```
tup_obj[0] = 100000
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
/var/folders/vp/zv5p63wd1rz3_yn4kvl_kwb00000gq/T/ipykernel_36606/501815478.py in <module>
----> 1 tup_obj[0] = 100000

TypeError: 'tuple' object does not support item assignment
```

In [108…
```
try:
    tup_obj[0] = 100000
except TypeError:
    print("Tuples are immutable :)")
```

```
Tuples are immutable :)
```