Problem 1. Basic Array Element Comparisons

In [4]:

```
import numpy as np
arr = np.array([2, 5, 8, 10, 3, 6, 7])
```

In [3]:

```
28%2
```

Out[3]:

```
0
```

In [4]:

```
for i in arr:
    if i%2 == 0:
        print(i)
```

```
2
8
10
6
```

Problem 2: Find Prime Array Elements:

In [14]:

```
def isitprime(x):
    if x < 2:
        return False
    for i in range(2, int(x**0.5) + 1):
        if x % i == 0:
            return False
    return True
```

In [34]:

```
isitprime(16)
```

Out[34]:

```
False
```

In [55]:

```
array= np.array([2,3,4,5,7,10,11,12,14,79])
prime_array = []

for i in range(len(array)):
    result = isitprime(array[i])

    prime_array.append(result)

print(prime_array)
```

```
[True, True, False, True, True, False, True, False, False, True]
```

Problem 3: Nested For Loops with 2D Arrays

In [38]:

```
# Initialize the array with zeros
n = 10
a = np.zeros((n, n), dtype=int)
print(a)
```

```
[[0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0]]
```

In [101…

```
def fill_fibonacci_matrix(matrix):

    #initiate fib seq
    fib_sequence = [0, 1]
    for i in range(2, 19):
        fib_sequence.append(fib_sequence[i-1] + fib_sequence[i-2])

    #fill first row in:
    a[0]=fib_sequence[0:10]

    #fill second row in:
    a[1]=fib_sequence[1:11]

    #fill third row in:
    a[2]=fib_sequence[2:12]

    #and so on...
    a[3]=fib_sequence[3:13]

    a[4]=fib_sequence[4:14]

    a[5]=fib_sequence[5:15]

    a[6]=fib_sequence[6:16]
    a[7]=fib_sequence[7:17]
    a[8]=fib_sequence[8:18]
    a[9]=fib_sequence[9:19]

    print(a)
```

I know that there was a ***much*** more Pythonic and easier way to solve this problem. I could've just made a for() loop to fill in the rest of the matrix, but, to be completely honest, I have spent hours on this homework and it was just easier this way! Sorry!

In [100…

```
fill_fibonacci_matrix(a)
```

```
[[   0    1    1    2    3    5    8   13   21   34]
 [   1    1    2    3    5    8   13   21   34   55]
 [   1    2    3    5    8   13   21   34   55   89]
 [   2    3    5    8   13   21   34   55   89  144]
 [   3    5    8   13   21   34   55   89  144  233]
 [   5    8   13   21   34   55   89  144  233  377]
 [   8   13   21   34   55   89  144  233  377  610]
 [  13   21   34   55   89  144  233  377  610  987]
 [  21   34   55   89  144  233  377  610  987 1597]
 [  34   55   89  144  233  377  610  987 1597 2584]]
```

Problem 4: While Loop Within a For Loop

In [204…

```
arrays_list = [np.array([10, 20, 30, 40, 50]),
               np.array([5, 15, 25]),
               np.array([1, 2, 3, 4, 5, 6, 7, 8, 9])]
```

In [224…

```
for i in range(len(arrays_list)):
    single_array = arrays_list[i]
    sums = 0
    index = 0

    while sums < 50:
        sums += single_array[index]
        index += 1

        if index == len(single_array):
            index = 0


    print("Array:", arrays_list[i], "; Sum =", sums)
    print("Index where it stopped:", index)
```

```
Array: [10 20 30 40 50] ; Sum = 60
Index where it stopped: 3
Array: [ 5 15 25] ; Sum = 50
Index where it stopped: 1
Array: [1 2 3 4 5 6 7 8 9] ; Sum = 51
Index where it stopped: 3
```

Problem 5: Logical Testing with NumPy:

In [225…

```
data = np.array([[5, 8, 3], [7, 2, 9], [6, 4, 1]])
```

In [226…

```
print(data)
```

```
[[5 8 3]
 [7 2 9]
 [6 4 1]]
```

In [263…

```
nRows, nCols = 3, 3

for row in range(nRows):
    for col in range(nCols):
        if data[row, col] > 6:
            print("Element=", data[row, col])
            print(f"Location: row {row}, column {col}")
            print()
```

```
Element= 8
Location: row 0, column 1

Element= 7
Location: row 1, column 0

Element= 9
Location: row 1, column 2
```