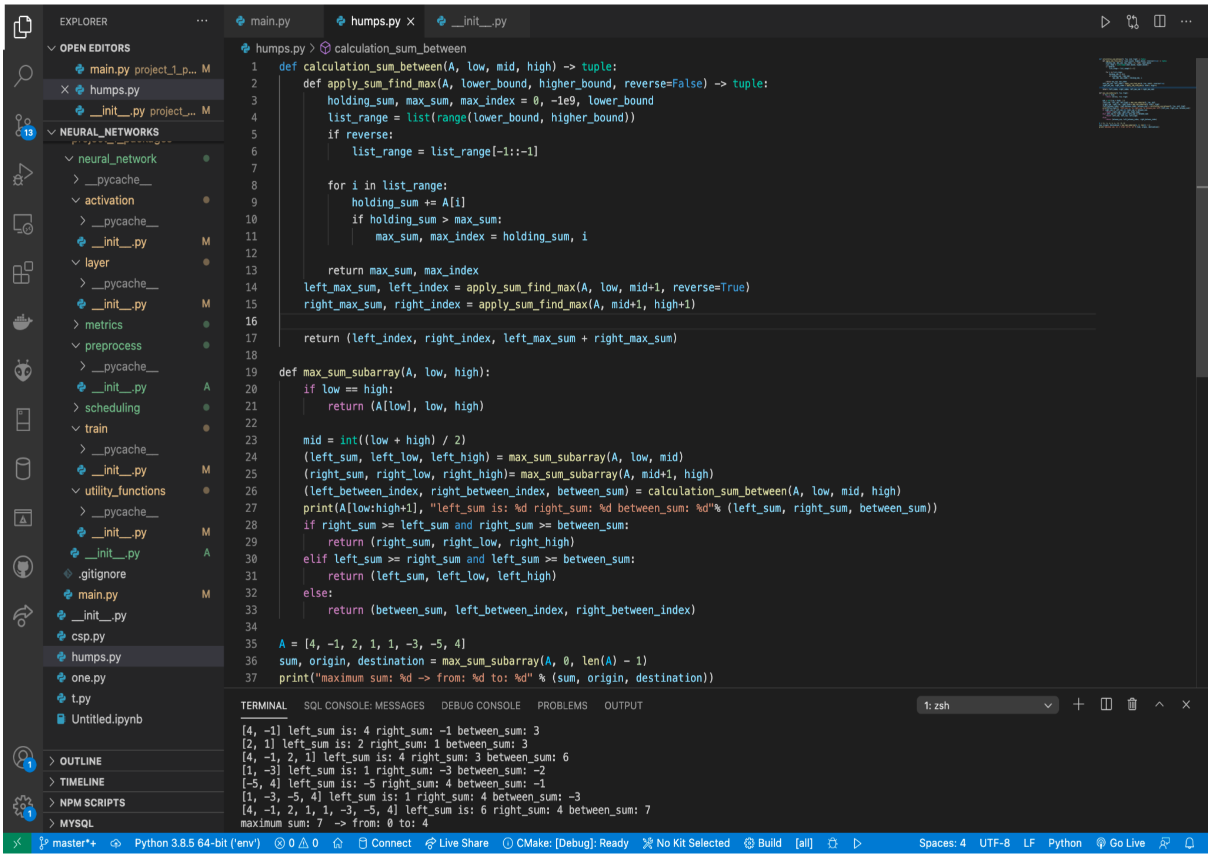
**Algorithm Design 3992**
**April 2021**
**Instructor : Dr.Shafiei**

## Question 1.



```python
def calculation_sum_between(A, low, mid, high) -> tuple:
    def apply_sum_find_max(A, lower_bound, higher_bound, reverse=False) -> tuple:
        holding_sum, max_sum, max_index = 0, -1e9, lower_bound
        list_range = list(range(lower_bound, higher_bound))
        if reverse:
            list_range = list_range[-1::-1]

        for i in list_range:
            holding_sum += A[i]
            if holding_sum > max_sum:
                max_sum, max_index = holding_sum, i

        return max_sum, max_index
    left_max_sum, left_index = apply_sum_find_max(A, low, mid+1, reverse=True)
    right_max_sum, right_index = apply_sum_find_max(A, mid+1, high+1)

    return (left_index, right_index, left_max_sum + right_max_sum)

def max_sum_subarray(A, low, high):
    if low == high:
        return (A[low], low, high)

    mid = int((low + high) / 2)
    (left_sum, left_low, left_high) = max_sum_subarray(A, low, mid)
    (right_sum, right_low, right_high)= max_sum_subarray(A, mid+1, high)
    (left_between_index, right_between_index, between_sum) = calculation_sum_between(A, low, mid, high)
    print(A[low:high+1], "left_sum is: %d right_sum: %d between_sum: %d"% (left_sum, right_sum, between_sum))
    if right_sum >= left_sum and right_sum >= between_sum:
        return (right_sum, right_low, right_high)
    elif left_sum >= right_sum and left_sum >= between_sum:
        return (left_sum, left_low, left_high)
    else:
        return (between_sum, left_between_index, right_between_index)

A = [4, -1, 2, 1, 1, -3, -5, 4]
sum, origin, destination = max_sum_subarray(A, 0, len(A) - 1)
print("maximum sum: %d -> from: %d to: %d" % (sum, origin, destination))
```

```
[4, -1] left_sum is: 4 right_sum: -1 between_sum: 3
[2, 1] left_sum is: 2 right_sum: 1 between_sum: 3
[4, -1, 2, 1] left_sum is: 4 right_sum: 3 between_sum: 6
[1, -3] left_sum is: 1 right_sum: -3 between_sum: -2
[-5, 4] left_sum is: -5 right_sum: 4 between_sum: -1
[1, -3, -5, 4] left_sum is: 1 right_sum: 4 between_sum: -3
[4, -1, 2, 1, 1, -3, -5, 4] left_sum is: 6 right_sum: 4 between_sum: 7
maximum sum: 7  -> from: 0 to: 4
```

## Question 2.

The first matrices are

$$\begin{aligned} S_1 &= 6 & S_6 &= 8 \\ S_2 &= 4 & S_7 &= -2 \\ S_3 &= 12 & S_8 &= 6 \\ S_4 &= -2 & S_9 &= -6 \\ S_5 &= 6 & S_{10} &= 14. \end{aligned}$$

The products are

$$\begin{aligned} P_1 &= 1 \cdot 6 = 6 \\ P_2 &= 4 \cdot 2 = 8 \\ P_3 &= 6 \cdot 12 = 72 \\ P_4 &= -2 \cdot 5 = -10 \\ P_5 &= 6 \cdot 8 = 48 \\ P_6 &= -2 \cdot 6 = -12 \\ P_7 &= -6 \cdot 14 = -84. \end{aligned}$$

The four matrices are

$$\begin{aligned} C_{11} &= 48 + (-10) - 8 + (-12) = 18 \\ C_{12} &= 6 + 8 = 14 \\ C_{21} &= 72 + (-10) = 62 \\ C_{22} &= 48 + 6 - 72 - (-84) = 66. \end{aligned}$$

The result is

$$\begin{pmatrix} 18 & 14 \\ 62 & 66 \end{pmatrix}.$$

**Algorithm Design 3992**
**April 2021**
**Instructor : Dr.Shafiei**

---

**Question 3.**

- Master Theory :
    - T(n) = a . T(n / b) + O(n ^ d)

a. $T(n) = 3T(n / 9) + \sqrt{n}$

A = 3 , b = 9 , d = ½ → a = b ^ d → $O(n^{1/2} . log(n))$

b. $T(n) = T(n - 4) + n$

$\rightarrow T(n) = n + (n - 4) + (n - 8) + ... + 4 + T(0)$

$T(n) = \frac{n}{8}[2a + (\frac{n}{4} - 1) . 4] + T(0) = \frac{n}{8}[n + 4] + T(0) = \frac{n^2}{8} + \frac{n}{2} + T(0) \in$

c. $T(n) = 6T(n / 4) + n^2$

A = 6 , b = 4 , d = 2 → a < b ^ d → O(n ^ 2)

d. $T(n) = 5T(n / 2) + n^2$

A = 5 , b = 2 , d = 2 → a > b ^ d → $O(n^{Log\ 5\ base\ 2})$

---

**Question 4.**

$T(n) = 2T(n / 3 ) + n \ \ for\, n \geq 5$

A = 2 , b = 3 , d = 1 → b ^ d > a → O(n)

---

**Question 5.**

We can get to correct answer by testing the options and also we can solve it by writing characteristic equation:

$x^2 - 5x + 6 = 0 \rightarrow x = 2, 3 \rightarrow g(n) = \alpha 1\ (3)\ n + \alpha 2\ (2)$

---

**Question 6.**

Step 1 : moving n-1 discs from A to B using C
Step 2: moving one disc from A to C
Step 3: moving n-1 discs from B to A using C
Step 4: moving one disc from C to B
Step 5: moving n-1 disc from A to B using C
In Conclusion :

$$T(n-1) + 1 + T(n-1) + 1 + T(n-1) = 3T(n-1) + 2$$