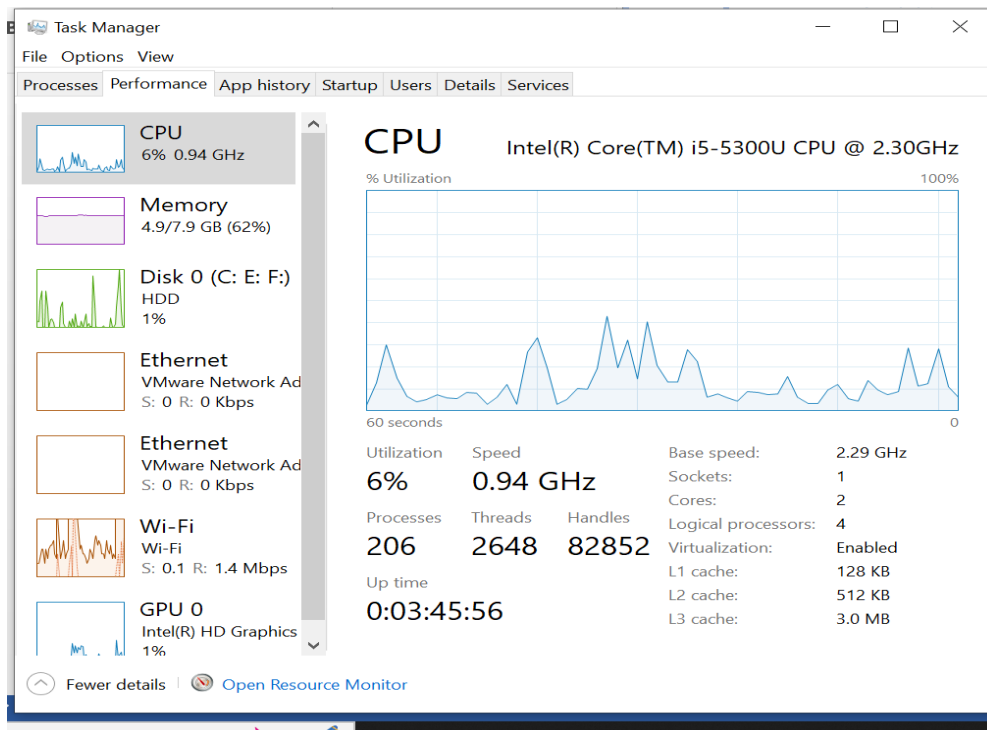


OPERATING SYSTEMS ASSIGNMENT #3



Name	Abdul Hadi
Registration Number	200901080
Batch & Section	CS01 A
Instructor's Name	Mam Asia

SYSTEM CORES: 2



MAC ADDRESS:

Network Connection Details	
Network Connection Details:	
Property	Value
Connection-specific DNS ...	
Description	Intel(R) Dual Band Wireless-AC 7265
Physical Address	48-45-20-8B-E9-FD
DHCP Enabled	Yes
IPv4 Address	10.4.21.122
IPv4 Subnet Mask	255.255.248.0
Lease Obtained	Wednesday, December 28, 2022 9:03:00 F
Lease Expires	Wednesday, December 28, 2022 10:59:38
IPv4 Default Gateway	10.4.16.1
IPv4 DHCP Server	10.4.16.1
IPv4 DNS Servers	8.8.8.8
	8.8.4.4
IPv4 WINS Server	
NetBIOS over Tcpi...	Yes
Link-local IPv6 Address	fe80::391d:65d9:78d9:9e30%2
IPv6 Default Gateway	
IPv6 DNS Server	
Close	

CODE:

```
import threading
```

```
def merge(arr, l, m, r):
```

```
    # Find sizes of two subarrays to be merged
```

```
    n1 = m - l + 1
```

```
    n2 = r - m
```

```
    # Create temp arrays
```

```
    L = [0] * n1
```

```
    M = [0] * n2
```

```
    # Copy data to temp arrays L[] and M[]
```

```
    for i in range(0, n1):
```

```
        L[i] = arr[l + i]
```

```
    for j in range(0, n2):
```

```
        M[j] = arr[m + 1 + j]
```

```
    # Merge the temp arrays back into arr[l..r]
```

```
    i = 0 # Initial index of first subarray
```

```
    j = 0 # Initial index of second subarray
```

```
    k = l # Initial index of merged subarray
```

```
    while i < n1 and j < n2:
```

```
        if L[i] <= M[j]:
```

```
            arr[k] = L[i]
```

```
            i += 1
```

```
        else:
```

```
    arr[k] = M[j]

    j += 1

    k += 1
```

```
# Copy remaining elements of L[], if there are any
```

```
while i < n1:
```

```
    arr[k] = L[i]

    i += 1

    k += 1
```

```
# Copy remaining elements of M[], if there are any
```

```
while j < n2:
```

```
    arr[k] = M[j]

    j += 1

    k += 1
```

```
def merge_sort(arr, l, r):
```

```
    if l < r:
```

```
        # Find the middle point
```

```
        m = (l + (r - 1)) // 2
```

```
        # Sort first and second halves
```

```
        t1 = threading.Thread(target=merge_sort, args=(arr, l, m))
```

```
        t2 = threading.Thread(target=merge_sort, args=(arr, m + 1, r))
```

```
        # Wait for the threads to finish
```

```
        t1.start()
```

```
        t2.start()
```

```
        t1.join()
```

```
t2.join()
```

```
# Merge the sorted halves
```

```
merge(arr, l, m, r)
```

```
arr = [5, 4, 7, 1, 3, 2, 6]
```

```
merge_sort(arr, 0, len(arr) - 1)
```

```
# Print the sorted array
```

```
print(arr)
```

Github Link:

<https://github.com/ABDULHADI44>