



Institute of Space Technology, Islamabad

BS CS-01

Operating Systems

Assignment - 03

Due Date: Wednesday, 28th Dec 2022

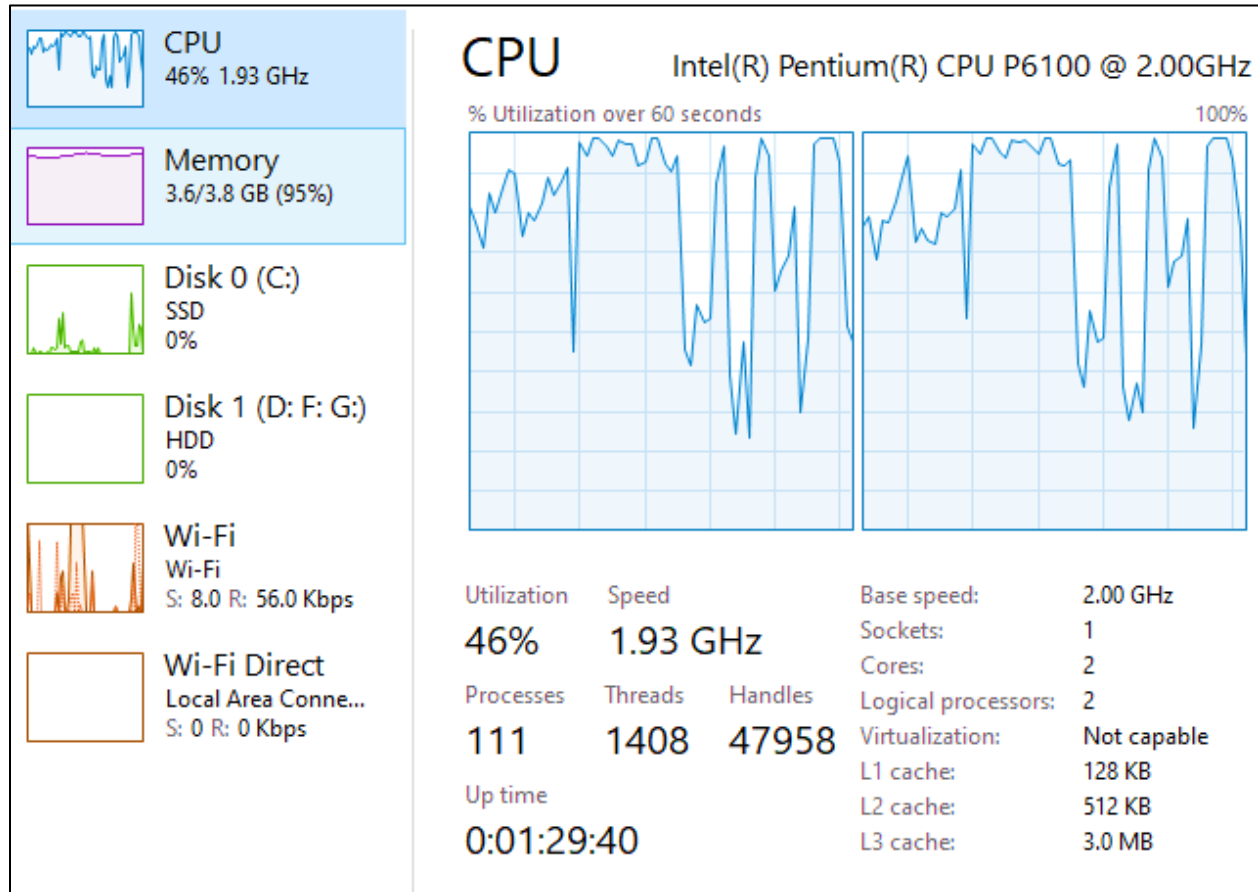
Submitted by: Faraz Ahmad Qureshi

Regno: (200901045)

Section: B

Course Instructor: Ma'am Asia

Find Number of Cores on your system:



As we can see from this figure, there are 2 cores and 2 threads in our system. As windows does not allow a process/program to take over both processes for an extensive period of time (to ensure the operating system is working reliably), we will use 2 threads on one logical processor to run our code.

In task manager we can also see the total number of processes running at the moment and the number of threads being used by those processes. We can also see in real-time the resource usage of our system including CPU, RAM, Disk, Networking etc.

Device Mac Address:

View hardware and connection properties	
Connectivity (IPv4/IPv6):	Disconnected
Name:	Wi-Fi
Description:	Broadcom 802.11n Network Adapter
Physical address (MAC):	1c:65:9d:5f:1b:a3
Status:	Operational
Maximum transmission unit:	1472
Link speed (Receive/Transmit):	144/72 (Mbps)
DHCP enabled:	Yes
DHCP servers:	192.168.100.1
DHCP lease obtained:	Wednesday, 28 December 2022 8:46:18 pm
DHCP lease expires:	Thursday, 29 December 2022 12:46:18 am
IPv4 address:	192.168.100.13/24
IPv6 address:	fe80::61ec:5ab:1888:9f8d%13/64
Default gateway:	fe80::1%13, 192.168.100.1
DNS servers:	fe80::1%13, 8.8.8.8, 8.8.4.4

Merge Sort Multithreading Code:

```
//Faraz Ahmad Qureshi 200901045
//OS Assignment
//Section B
//=====|
#include <iostream>
#include <thread>
#include <cstdlib>
using namespace std;

void merge(int arr[], int left, int middle, int right)
{
    int i, j, k;    //indexes that we use to denote left right and center
    int n1 = middle - left + 1; //name of sub arrays we use for combining
    int n2 = right - middle;

    int L[n1], R[n2];    //declaring the sub arrays

    for (i = 0; i < n1; i++)//copying values of the array from sub array to the combined array
        L[i] = arr[left + i];
    for (j = 0; j < n2; j++)
        R[j] = arr[middle + 1 + j];
```

```

i = 0;
j = 0;
k = left;
while (i < n1 && j < n2)//logic 2 decide whether to combin the index w/left array or right
{
    if (L[i] <= R[j])
    {
        arr[k] = L[i];
        i++;
    }
    else
    {
        arr[k] = R[j];
        j++;
    }
    k++;
}

while (i < n1)
{
    arr[k] = L[i];
    i++;
    k++;
}

while (j < n2)
{
    arr[k] = R[j];
    j++;
    k++;
}
}

void mergeSort(int arr[], int left, int right)
{
    this_thread::sleep_for (std::chrono::seconds(1));
    if (left < right)
    {
        int middle = left+(right-left)/2;

        thread leftSorter(mergeSort,arr,left,middle);
        thread rightSorter(mergeSort,arr,middle+1,right);

        cout << "< Left-sort-threadID: " << leftSorter.get_id();
        cout << endl;
        cout << "> Right-sort-threadID: " << rightSorter.get_id() ;

        leftSorter.join();
        rightSorter.join();
        merge(arr, left, middle, right);
        //
    }
}

void printArray(int A[], int size)
{
    int i;
    for (i=0; i < size; i++)

```

```

        cout<<A[i]<<" ";
        cout<<endl;
    }

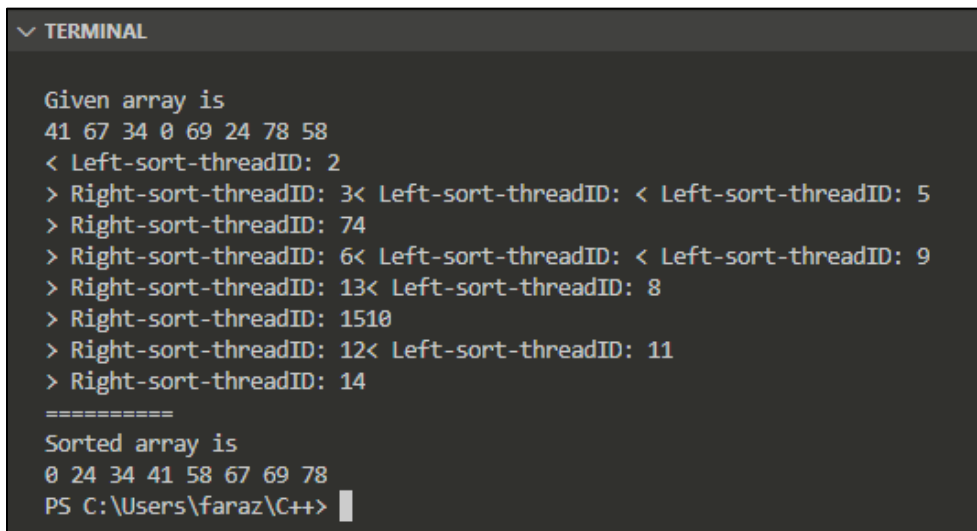
int main()
{
    int* array1;
    int arr_size;
    cout << "Number of cores on this system: 2\n";
    cout << "Enter size of array you want to sort: ";
    cin >> arr_size;
    array1 = new int[arr_size] ;
    for (int i = 0; i < arr_size; i++)
    {
        array1[i] = (rand()%100);
    }
    cout<<"Given array is \n";
    printArray(array1, arr_size);

    mergeSort(array1, 0, arr_size - 1);

    cout<<"\n===== \nSorted array is \n";
    printArray(array1, arr_size);
    return 0;
}

```

Merge Sort Multithreading Code Output:



```

v TERMINAL

Given array is
41 67 34 0 69 24 78 58
< Left-sort-threadID: 2
> Right-sort-threadID: 3< Left-sort-threadID: < Left-sort-threadID: 5
> Right-sort-threadID: 74
> Right-sort-threadID: 6< Left-sort-threadID: < Left-sort-threadID: 9
> Right-sort-threadID: 13< Left-sort-threadID: 8
> Right-sort-threadID: 1510
> Right-sort-threadID: 12< Left-sort-threadID: 11
> Right-sort-threadID: 14
=====
Sorted array is
0 24 34 41 58 67 69 78
PS C:\Users\faraz\C++>

```

As you can see from the output, The given code runs in parallel (multithreading) and sorts the array faster than if it were to run on a single thread. We start with only using 2 available threads on our processor, The code runs these threads recursively and we end up with multiple threads running which sort the given array (entered by the user) as quickly as possible.