**ES215: Computer Organisation and Architecture**

[**Github Repo Link**](https://github.com/Madhav-Kanda/ES215_Assignment_1)

**https://github.com/Madhav-Kanda/ES215\_Assignment\_1**

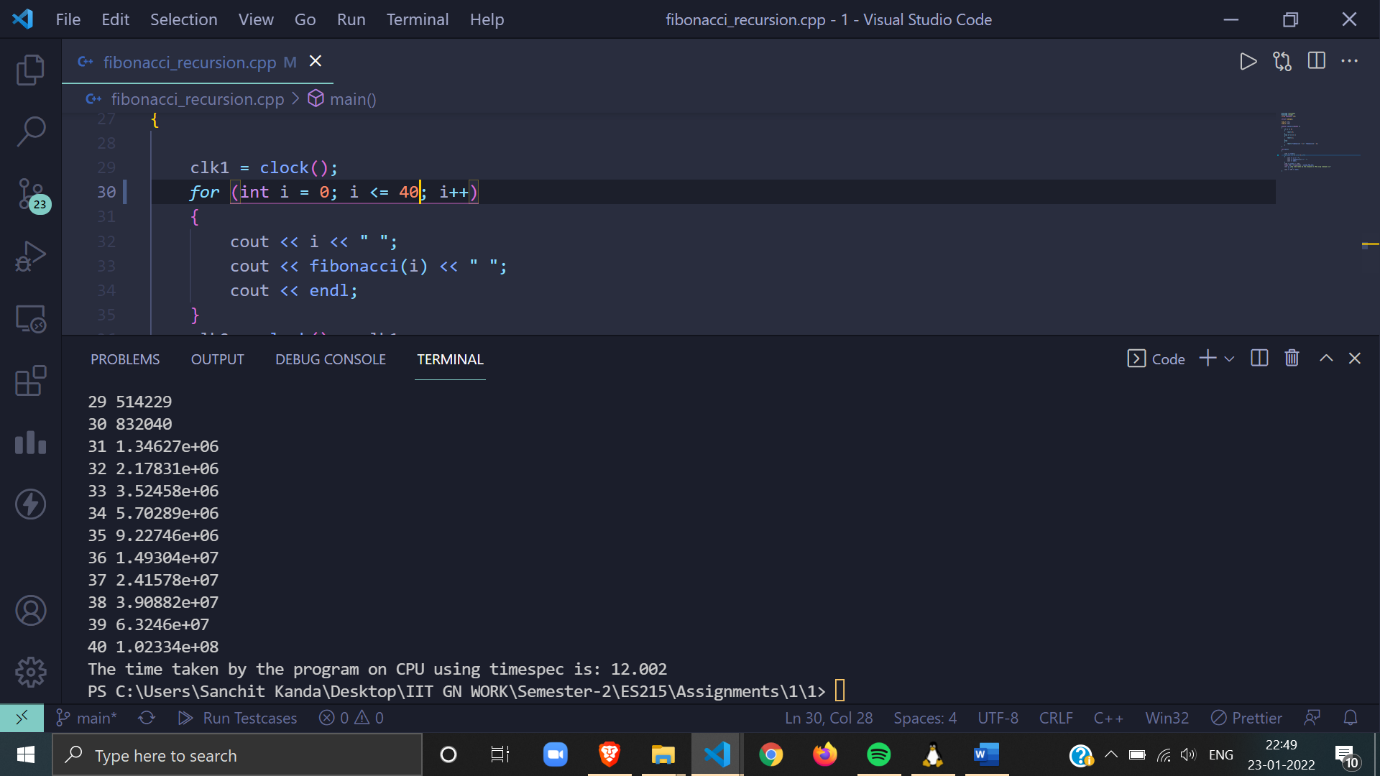
**Q1.**

1. Since we cannot run the program for 100 Fibonacci numbers as it would take a very long time, therefore, using the time calculated for the first 40 Fibonacci numbers we calculate the time for

Time taken for first 40 Fibonacci =**12.002 sec**

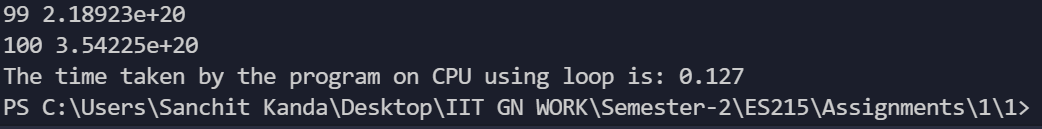
Formula to calculate time taken for 100 Fibonacci = ((1.168)^(60)))\*(12.002) sec = **133,605.45 sec**

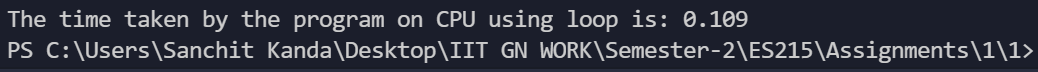
Time taken using recursion: **133,605.45 sec**

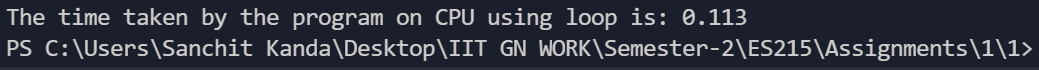


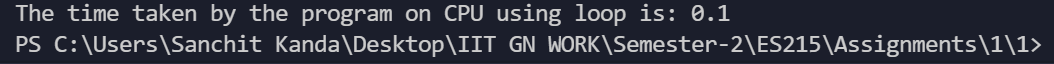
The above image shows the time taken by the Recursion to calculate first 40 Fibonacci numbers.

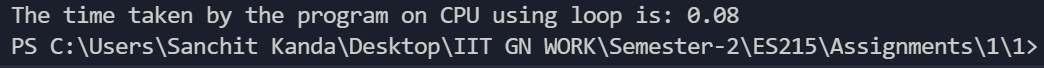
1. **Time taken using loop for 5 observations are as follows:**









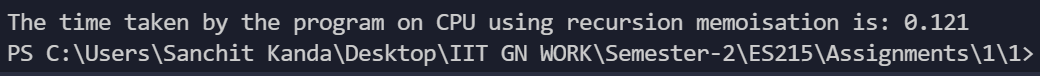


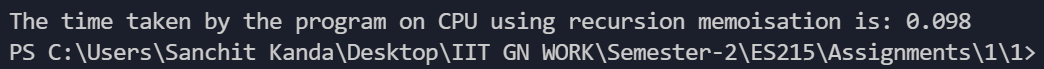
The values in the sorted order are as follows:

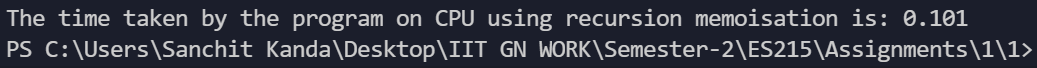
**0.08,0.1,0.109,0.113,0.127**

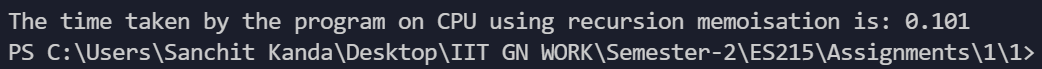
Thus, the median value of Fibonacci using loop is **0.109 s**

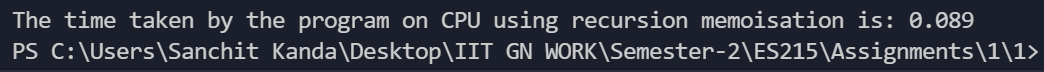
1. **Time taken using recursion and memoisation for 5 observations are as follows:**









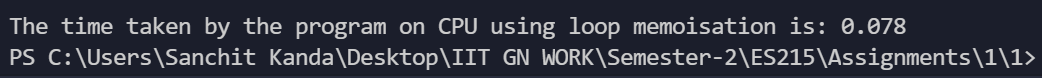


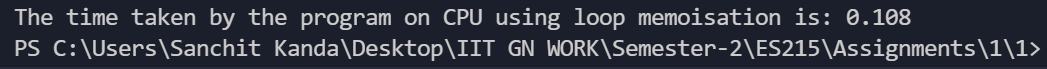
The values in the sorted order are as follows:

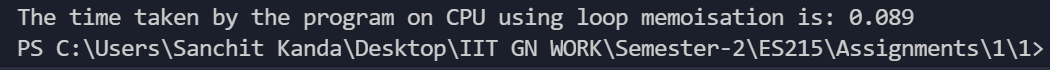
**0.089s ,0.098s ,0.101s, 0.101s ,0.121s**

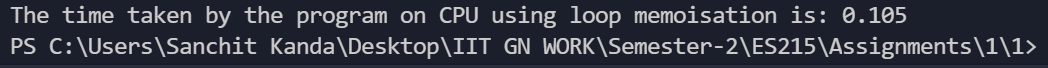
Thus, the median value of Fibonacci using loop is **0.101s**

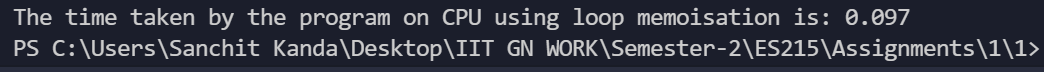
1. **Time taken using loop and memoisation for 5 observations are as follows:**











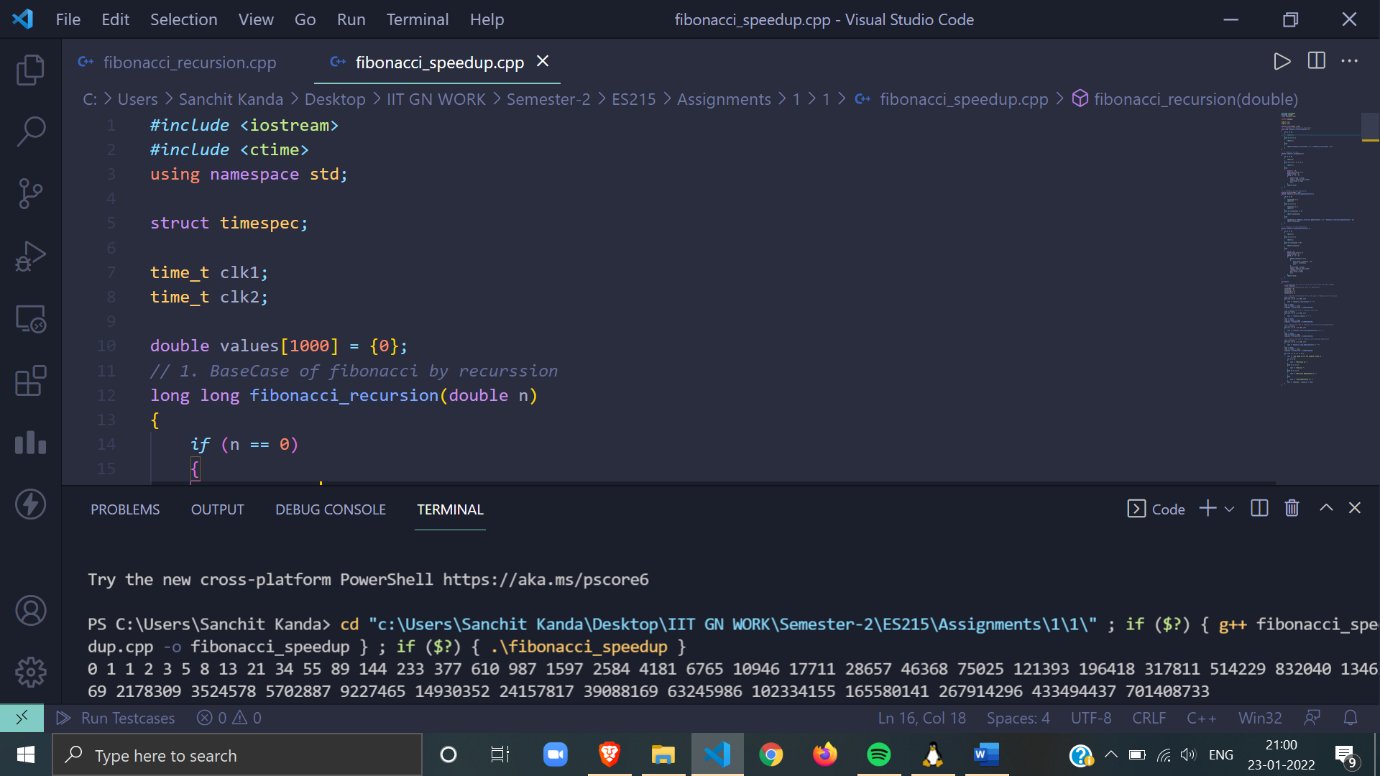
The values in the sorted order are as follows:

**0.078s ,0.089s ,0.097s, 0.105s ,0.108s**

Thus, the median value of Fibonacci using loop is **0.097s**

**Speedup of all the programs:**

1. Speedup for recursion is: **(Time taken using recursion/Time taken using recursion)** = **1**
2. Speedup for loop is**: (Time taken using recursion/ Time taken using loop) = 133,605/0.109 = 1225733.95**
3. Speedup for recursion and memoisation is: **133,605/0.101 = (Time taken using recursion/ Time taken using recursion memoisation) = 1322821.78**
4. Speedup for loop memoisation is: **133,605/0.097 = (Time taken using recursion/ Time taken using loop memoisation) = 1377371.13**



**I wrote a separate program to calculate the speed up for the individual program but since the Recursion program does not come to completion thus speed up could not be calculated using the code.**

**Q2.**

1. **Output Time**

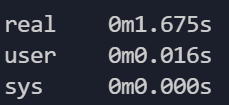
**Execution Time = real (in the images showing the data)**

1. **For C++ using Double**
2. **N=32**

CPU Time=user +sys

CPU Time = 0.016s + 0.000s = **0.016s**

System Time =sys= **0.000s**

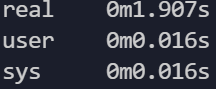


1. **N=64**

CPU Time=user +sys

CPU Time = 0.016s + 0.016s = **0.032s**

System Time= sys= **0.016s**

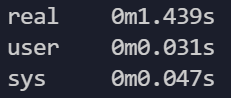


1. **N=128**

CPU Time=user +sys

CPU Time = 0.031s + 0.047s = **0.078s**

System Time= sys= **0.047s**

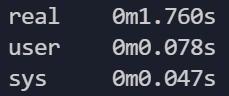


1. **N=256**

CPU Time=user +sys

CPU Time = 0.078s + 0.047s = **0.125s**

System Time= sys= **0.047s**

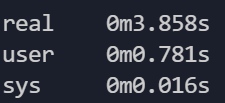


1. **N=512**

CPU Time=user +sys

CPU Time = 0.781s + 0.016s = **0.797s**

System Time= sys= **0.016s**

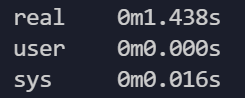


1. **For C++ using Integer**
2. **N=32**

CPU Time=user +sys

CPU Time = 0.000s + 0.016s = **0.016s**

System Time= sys= **0.016s**

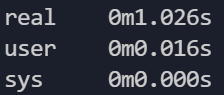


1. **N=64**

CPU Time=user +sys

CPU Time = 0.016s + 0.000s = **0.016s**

System Time= sys= **0.000s**

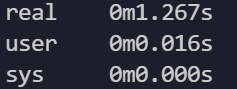


1. **N=128**

CPU Time=user +sys

CPU Time = 0.016s + 0.000s = **0.016s**

System Time= sys= **0.000s**

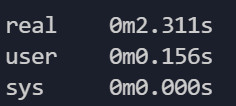


1. N=256

CPU Time=user +sys

CPU Time = 0.156s + 0.000s = **0.156s**

System Time= sys= **0.000s**

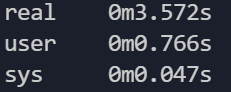


1. N=512

CPU Time=user +sys

CPU Time = 0.766s + 0.047s = **0.813s**

System Time= sys= **0.047s**

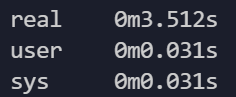


1. **For Python using integer**
2. **N=32**

CPU Time=user +sys

CPU Time = 0.031s + 0.031s = **0.062s**

System Time= sys= **0.031s**

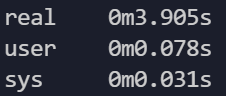


1. **N=64**

CPU Time=user +sys

CPU Time = 0.078s + 0.031s = **0.109s**

System Time= sys= **0.031s**

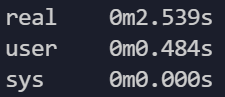


1. **N=128**

CPU Time=user +sys

CPU Time = 0.484s + 0.000s = **0.484s**

System Time= sys= **0.000s**

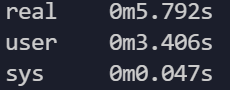


1. **N=256**

CPU Time=user +sys

CPU Time = 3.406s + 0.047s = **3.453s**

System Time= sys= **0.047s**

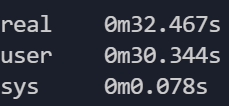


1. **N=512**

CPU Time=user +sys

CPU Time = 30.344s + 0.078s = **30.422s**

System Time= sys= **0.078s**

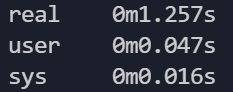


1. **For Python using Double**
2. **N=32**

CPU Time=user +sys

CPU Time = 0.047s + 0.016s = **0.063s**

System Time= sys= **0.016s**

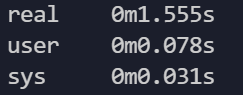


1. **N=64**

CPU Time=user +sys

CPU Time = 0.078s + 0.031s = **0.141s**

System Time= sys= **0.031s**

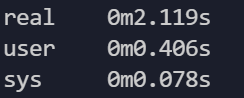


1. **N=128**

CPU Time=user +sys

CPU Time = 0.406s + 0.078s = **0.484s**

System Time= sys= **0.078s**

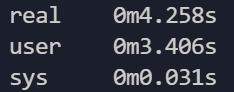


1. **N=256**

CPU Time=user +sys

CPU Time = 3.406s + 0.031s = **3.437s**

System Time= sys= **0.031s**

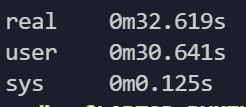


1. **N=512**

CPU Time=user +sys

CPU Time = 30.641s + 0.125s = **30.766s**

System Time= sys= **0.125s**



**B) Using Language Hooks**

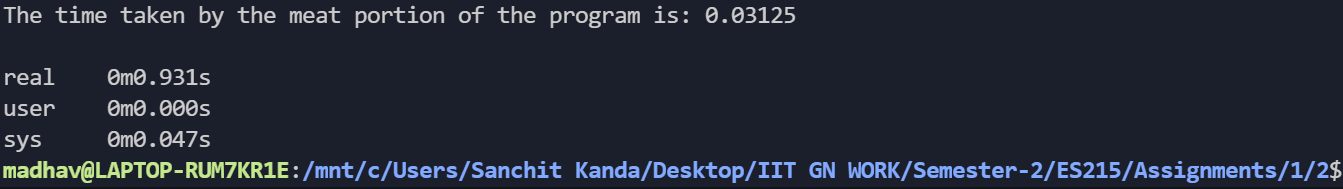
**a) C++ program for integer**

**1. N=32**

Execution Time: **0.931s**

Meat Portion: **0.03125s**

**(Meat Portion/Execution Time): 0.0335**

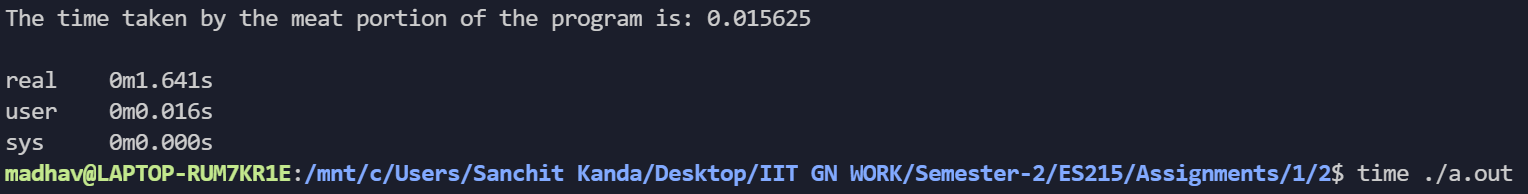


**2. N=64**

Execution Time: **1.641s**

Meat Portion: **0.015625s**

**(Meat Portion/Execution Time):** **0.009521**



3. **N=128**

Execution Time: **1.745s**

Meat Portion: **0.015625s**

**(Meat Portion/Execution Time): 0.0089**

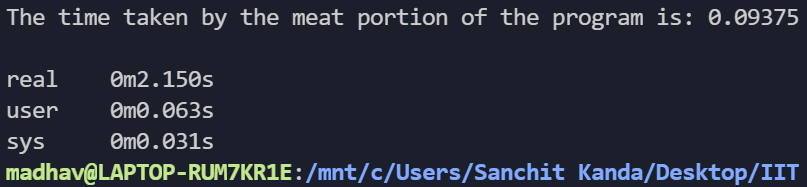


4. **N=256**

Execution Time: **2.150s**

Meat Portion: **0.09375s**

**(Meat Portion/Execution Time): 0.043**

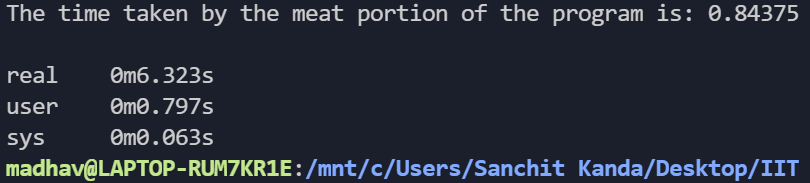


**5. N=512**

Execution Time: **6.323s**

Meat Portion: **0.84375s**

**(Meat Portion/Execution Time): 0.133**



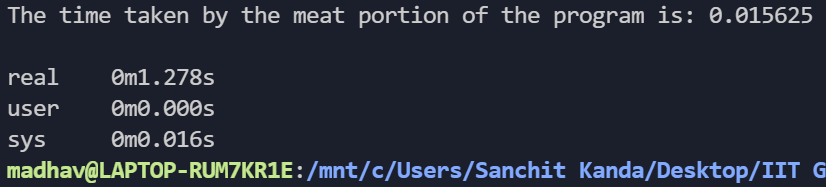
**b) C++ program for double**

**1. N=32**

Execution Time: **1.278s**

Meat Portion**: 0.015625s**

**(Meat Portion/Execution Time): 0.1222**

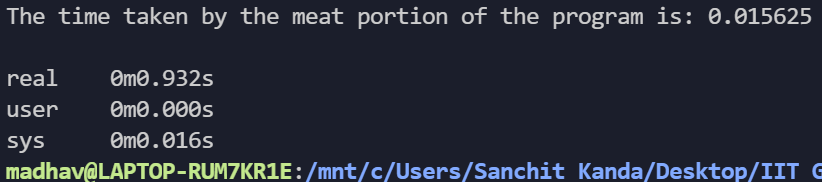


**2. N=64**

Execution Time: **0.932s**

Meat Portion: **0.015625s**

**(Meat Portion/Execution Time): 0.0167**

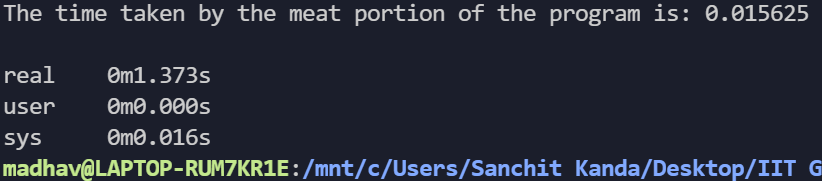


**3. N=128**

Execution Time: **1.373s**

Meat Portion: **0.015625s**

**(Meat Portion/Execution Time): 0.0113**

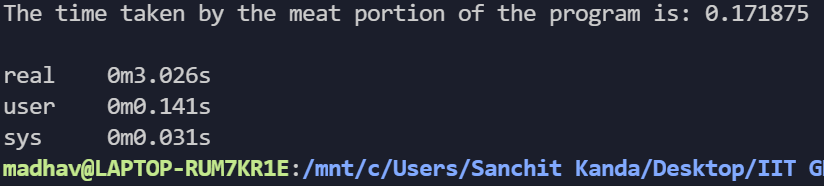


**4. N=256**

Execution Time: **3.026s**

Meat Portion: **0.171875s**

**(Meat Portion/Execution Time): 0.056799**

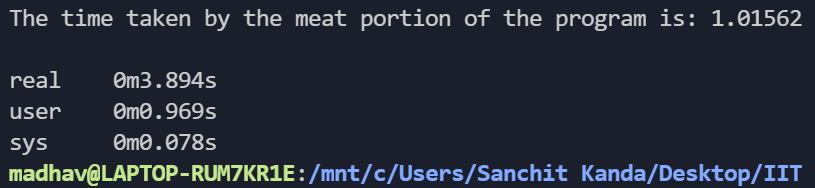


**5. N=512**

Execution Time: **3.894s**

Meat Portion: **1.0156s**

**(Meat Portion/Execution Time): 0.2608**



**a) Python program for Double**

**1. N=32**

Execution Time: **1.126s**

Meat Portion: **0.0110s**

**(Meat Portion/Execution Time): 0.00976**

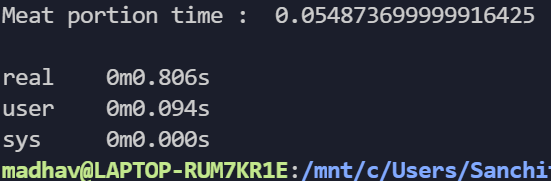


**2. N=64**

Execution Time: **0.806s**

Meat Portion: **0.05487s**

**(Meat Portion/Execution Time): 0.067**

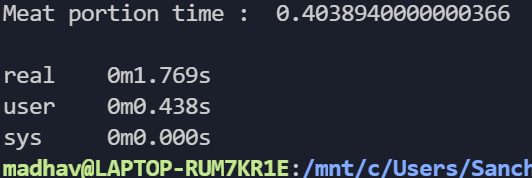


**3. N=128**

Execution Time: **1.769s**

Meat Portion: **0.403s**

**(Meat Portion/Execution Time): 0.2278**

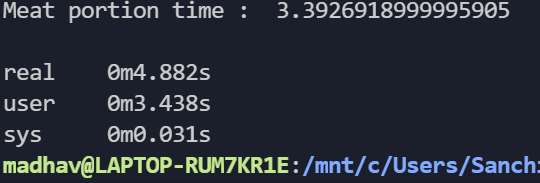


**4. N=256**

Execution Time: **4.882s**

Meat Portion: **3.392s**

**(Meat Portion/Execution Time): 0.69**

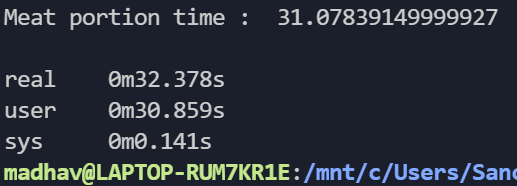


**5. N=512**

Execution Time: **32.378s**

Meat Portion: **31.07s**

**(Meat Portion/Execution Time): 0.95**



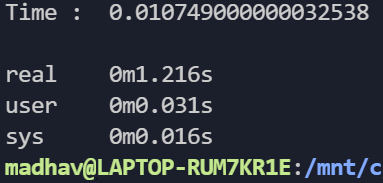
**b) Python program for Integer**

1. **N=32**

Execution Time: **1.216s**

Meat Portion: **0.0107s**

**(Meat Portion/Execution Time): 0.087**

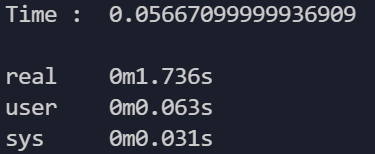


**2. N=64**

Execution Time: **1.736s**

Meat Portion: **0.0566s**

**(Meat Portion/Execution Time): 0.032**

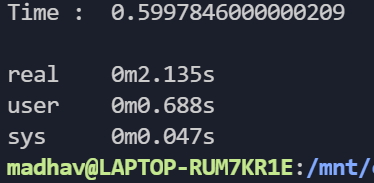


**3. N=128**

Execution Time: **2.135s**

Meat Portion: **0.599s**

**(Meat Portion/Execution Time): 0.28**

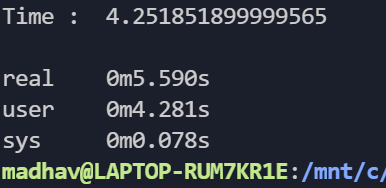


**4. N=256**

Execution Time**: 5.590s**

Meat Portion: **4.251s**

(Meat Portion/Execution Time): **0.76**

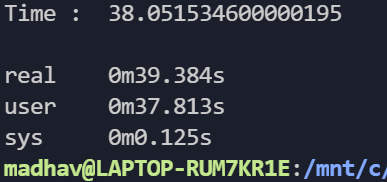


**5. N=512**

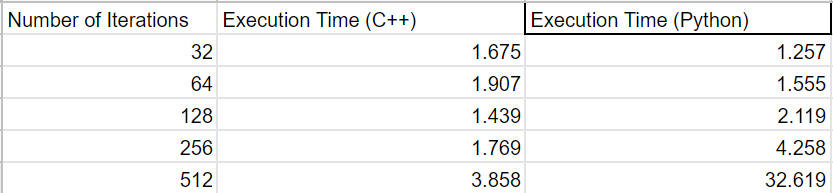
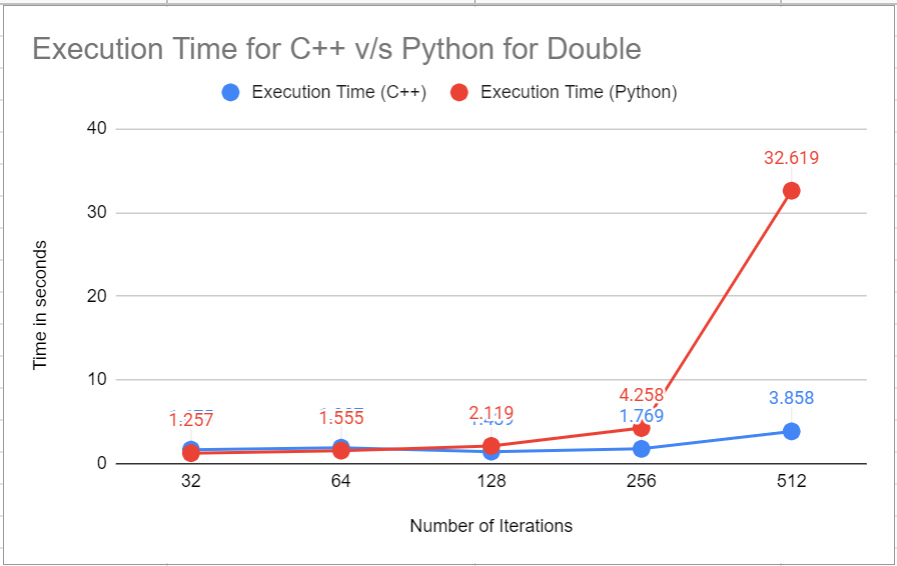
Execution Time: **39.384s**

Meat Portion: **38.05s**

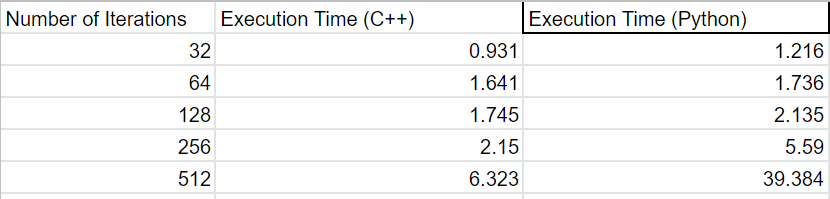
**(Meat Portion/Execution Time): 0.96**

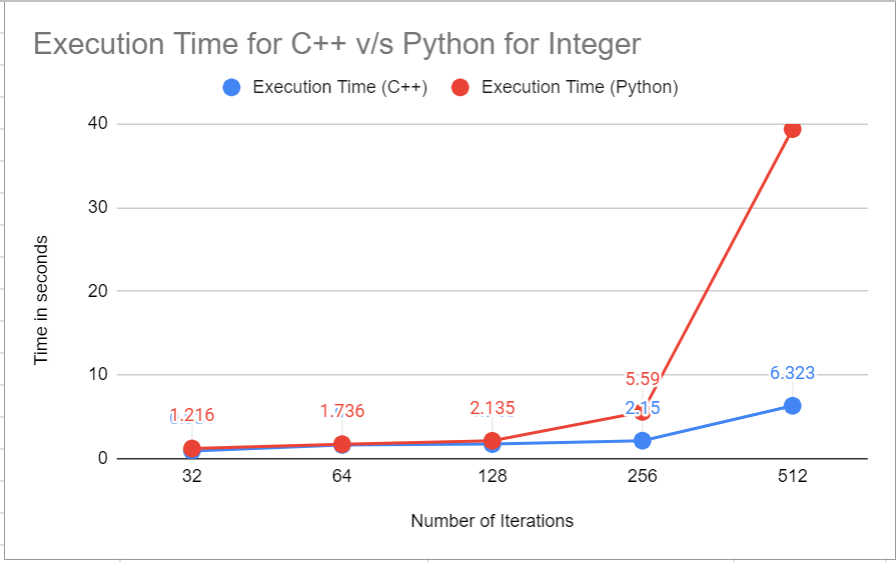


1. **Plots for Execution Time**
2. **Execution Time for Double of the Two programming languages**

****

1. **Execution Time for Integer of the Two programming languages**

****

****

**Observations**

According to the above graphs, it is clear that the execution time for python in both Integer and Double matrix multiplication is higher than that of C++.

This also goes well with the fact that Python is a high-level language whereas C++ is middle-level language and execution time for high level language is more than the execution time on a middle-level language given the same program.

From the data mentioned in the a) part it is clear that the system time (real) is less than the program execution time for any program on any language. This is because execution time takes into consideration the time taken from the start to finish of the call. Thus it also includes the time that we take to give the input. System time is the time spent in the kernel within the process. Since execution time includes the system time as it calculates the time from start to finish of the program thus, it is always greater than system time.