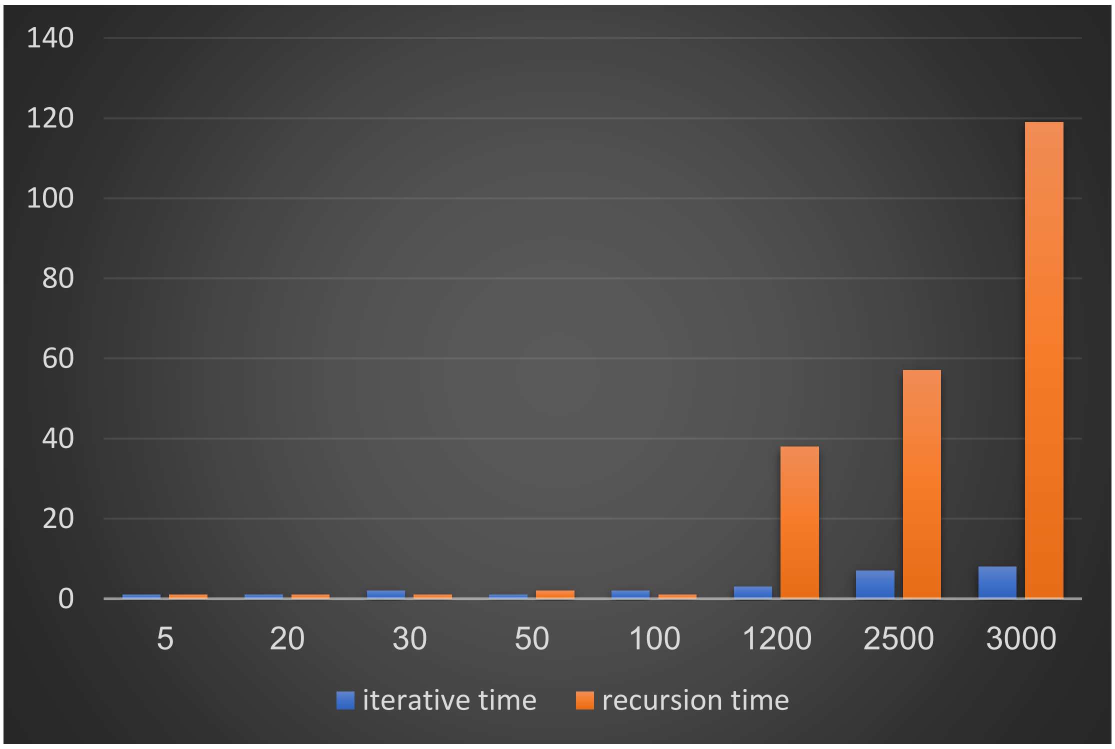
My name: ayah khamaysa

Assignment #1

n	Iterative time(ms)	Recursive time(ms)
5	1	1
20	1	1
30	2	1
50	1	2
100	2	1
1200	3	38
2500	7	57
3000	8	119



The Stack Overflow theme is a common problem. It occurs when memory allocated to functions and variables is used incorrectly and also occurs when the call is repeated endlessly, which overrides the amount of memory allocated to the .program

When Stack Overflow occurs, the program execution is terminated and an error is returned. To solve this problem, you should avoid calling unfinished functions or using memory incorrectly

---

From this code we can conclude that it has two functions to calculate the sequential factorial and the recurring factorial of a given number. The iterativefactorial function uses an iterative loop to calculate the factorial, while the recursivefactorial .function uses repeating to calculate the factorial

Next, the execution time of the selected function is measured using clock and the difference between the initial time and the final time is calculated. The workings are printed and the time it took to calculate

Based on the results, we can compare the performance of sequential factorism and recurring factorism and determine which is faster in calculating the factorism of the given number

---

I concluded that the two methods I used in the solution are almost identical in time Also, recursive when the number exceeds the required volume, it will cause the stack overflow.

Finally the code:

```
<include<iostream#

<include<time.h#

;using namespace std

}long long iterativefactorial(int n)
```

```
;long long f=1
}for(int i=1;i<=n;i++)
;f*=i
{
;return f
{
}long long recursivefactorial(int n)
;if (n==1)return 1
;return n*recursivefactorial(n-1)
{

}()int  main
;int n
;"cout<<"enter number\n
;cin>> n

;long long result
;()clock_t start = clock
;result= iterativefactorial(n)
;result= recursivefactorial(n)//
;()clock_t end =clock
;cout<<"The factorial number\t"<<n<<"\tis\t"<< result <<endl
;double real_time=(((double)end-start)/CLOCKS_PER_SEC)*1000000
;cout<<"excution time\t"<<"is\t"<<real_time<<endl

        {
```