DAA PRACTICAL - 1

## PRACTICAL-1

Implement a function for each of following problems and count the number of steps executed/Time taken by each function on various inputs and write complexity of each function. Also draw a comparative chart. In each of the following function N will be passed by user.

- 1. To calculate sum of 1 to N number using loop.
- 2. To calculate sum of 1 to N number using equation.3.To calculate sum of 1 to N numbers using recursion.

## CODE:

```
#include <stdio.h>
int count_loop=0; int count_rec=0;
int count_eq=0;
void loop(int ); int
rec(int); void
eq(int);
int main()
{ int n;
  printf("enter value of n =");
scanf("%d",&n);
  loop(n);
             eq(n);
  int recursion=rec(n);
  printf("sum of rec=%d\n",recursion);
  printf("rec count=%d",count rec);
  return 0;
}
void loop(int n){  int
```

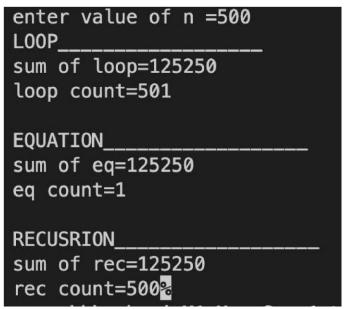
Patel Guru 21012011074

DAA PRACTICAL - 1

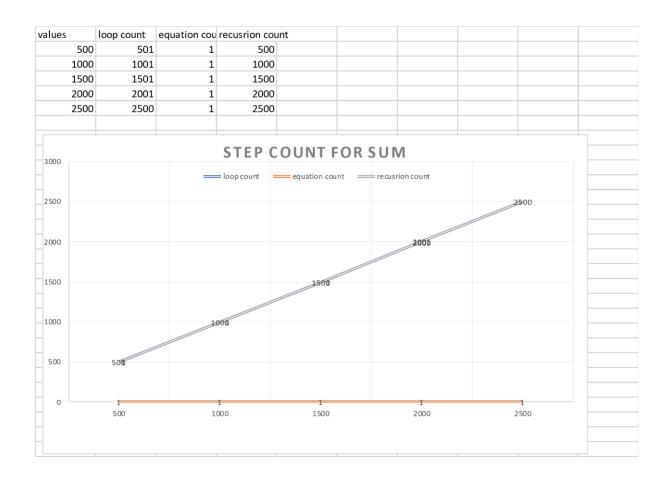
```
sum_loop=0; for(int i=0;i<=n;i++){</pre>
sum loop=sum loop+i;
count_loop++;
 }
 printf("LOOP_____\n"); printf("sum
of loop=%d\n",sum loop); printf("loop
count=%d\n",count_loop);
}
void eq(int n){  int sum_eq=0;  count_eq;
sum_eq=(n*(n+1))/2; count_eq++;
                                 ___\n");
printf("\nEQUATION___
printf("sum of eq=%d\n",sum_eq); printf("eq
count=%d\n",count eq);
printf("\nRECUSRION_____\n");
}
int rec(int n){
if (n <=
1){
count_rec++; return n;
     } else
count_rec++;
               return
n + rec(n - 1);
  }
}
  OUTPUT:
```

Patel Guru 21012011074

DAA PRACTICAL - 1



## Graph:



Patel Guru 21012011074