NAME:	Pooja Patil
UID:	2022301011
SUBJECT	Design and Analysis of Algorithms
EXPERIMENT NO :	1A
DATE OF PERFORMANCE	30.01.2023
DATE OF SUBMISSION	05.02.2023
AIM:	To implement the various functions e.g., linear, non-linear, quadratic, exponential etc. 1) Print the values of each function value for all n starting 0 to 100 in tabular format for both aforementioned cases 2) Draw two 2D plot of all functions such that x-axis represents the values of n and y-axis represent the function value for different n values using LibreOffice Calc/MS Excel.
THEORY	A function is a relation between a set of inputs and a set of permissible outputs with the property that each input is related to exactly one output. Let A & B be any two non-empty sets; mapping from A to B will be a function only when every element in set A has one end, only one image in set B. 1) n 2) n^3 3) log n 4)$n.2^n$ 5)log (log n)

	6) 2^n 7) e^n 8) 3/2.n 9) $(\log n)^{1/2}$ 10) n.(log n)
ALGORITHM	➢ Initialize variables n and result. 1. n Take the value of n from 0-100 and print all the values. 2. n^3 • result = n*n*n • Apply a for loop for values of n from 0-100 and print all thevalues for result. 3. log(n) • result = log(n) • Apply a for loop for values of n from 0-100 and print all thevalues for result. 4. n*2^n • result = n*pow(2,n) • Apply a for loop for values of n from 0-100 and print all thevalues for result. 5. (3/2)^n • result = log(log(n)) • Apply a for loop for values of n from 0-100 and print all thevalues for result.

6. e^n
   - result = pow(2,n)
   - Apply a for loop for values of n from 0-100 and print all thevalues for result.
7. 2^n
   - result = exp(n) (e^n)
   - Apply a for loop for values of n from 0-100 and print all thevalues for result.

8. lg(lg n)
   - result = 3/2*n
   - Apply a for loop for values of n from 0-100 and print all thevalues for result.

9. (logn)^1/2
   - result = pow(log(n),0.5)
   - Apply a for loop for values of n from 0-100 and print all thevalues for result.

10. n*log(n)
    - result = n*log(n)
    - Apply a for loop for values of n from 0-100 and print all thevalues for result.

11.

i. Initialize a variable n.

ii. Create a function to find the factorial.

iii. factorial(n)

```
if(n==1 || n==0)
return i
else
return n*factorial(n-1)
```

iv. Apply a for loop for values of n from 0-19 and print al the values for result in the main function.

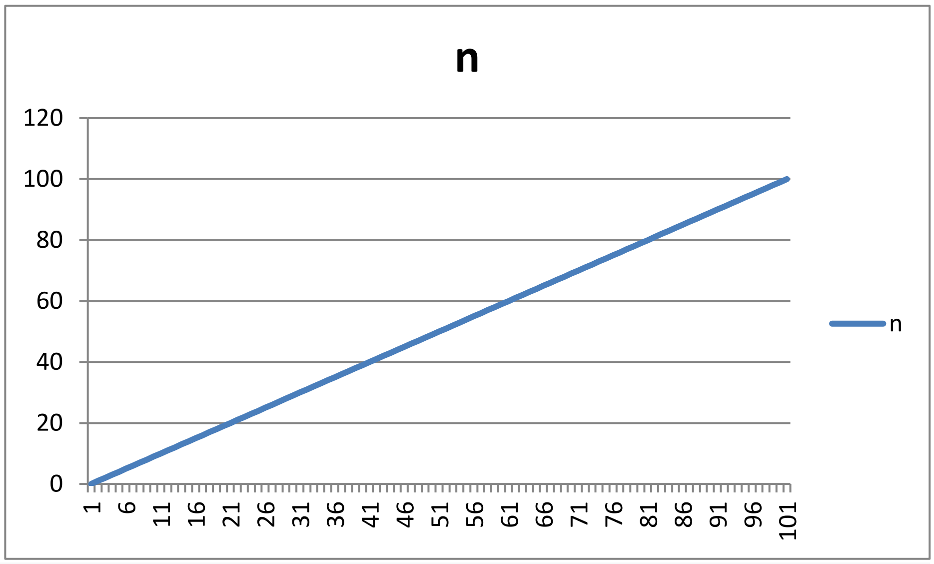
PROGRAM:	(code below)

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

long double factorial(int n){
 if(n==0 || n==1){
 return 1;
 }
 else{
 return n*factorial(n-1);
 }
}

int main(){
 int n,f1,f2,x=0;
 long double f4,f7,f11;
 double f3,f5,f6,f8,f9,f10;
 while(x<=11){
 printf("Enter the function no.:\n");
 scanf("%d",&x);
 if(x==1){
 printf("n\n");
 for(int i=0; i<=100; i++){
 f1=i;
 printf("%d\n",f1);
 }
 }
 else if(x==2){
 printf("n^3\n");
 for(int i=0; i<=100; i++){
 f2=i*i*i;
 printf("%d\n",f2);
 }
 }
 else if(x==3){
 printf("log(n)\n");
 for(int i=0; i<=100; i++){
 f3=log(i);
 printf("%.2lf\n",f3);
 }
 }
 else if(x==4){
 printf("n.2^n\n");
 for(int i=0; i<=100; i++){
 f4=i*(pow(2,i));
 printf("%.Lf\n",f4);
 }
 }
 else if(x==5){
```

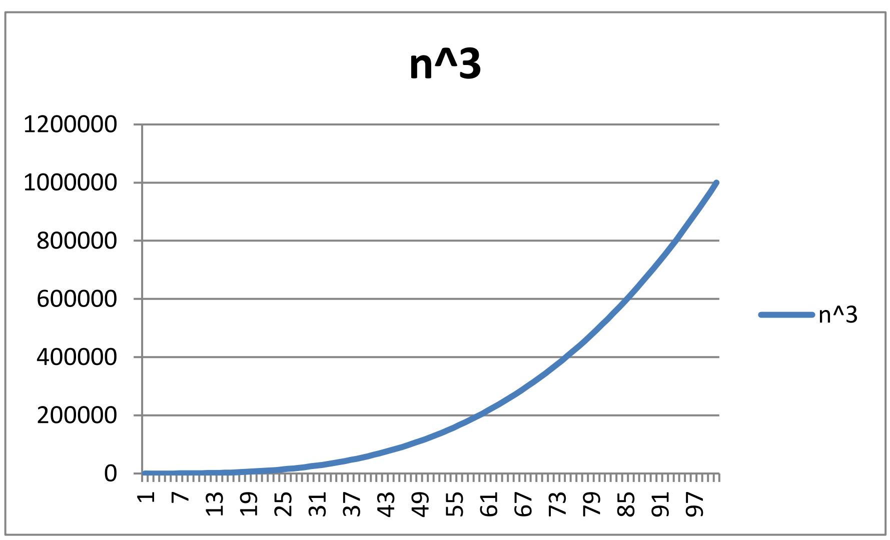
```
printf("(3/2)n\n");
for(int i=0; i<=100; i++){
f5=1.5*i;
printf("%.2lf\n",f5);
}
}
else if(x==6){
printf("e^n\n");
for(int i=0; i<=100; i++){
f6=exp(i);
printf("%.2lf\n",f6);
}
}
else if(x==7){
printf("2^n\n");
for(int i=0; i<=100; i++){
f7=(pow(2,i));
printf("%.Lf\n",f7);
}
}
else if(x==8){
printf("log(log(n))\n");
for(int i=0; i<=100; i++){
f8=log(log(i));
printf("%.2lf\n",f8);
}
}
else if(x==9){
printf("(logn)^1/2\n");
for(int i=0; i<=100; i++){
f9=pow(log(i),0.5);
printf("%.2lf\n",f9);
}
}
else if(x==10){
printf("n*log(n)\n");
for(int i=0; i<=100; i++){
f10=i*log(i);
printf("%.2lf\n",f10);
}
}
else if(x==11){
printf("n!\n");
for(int i=0; i<20; i++){
f11=factorial(i);
printf("%.2Lf\n",f11);
}
}
else{
```

```
printf("Invalid function number entered. Please enter a number from 1 to 11.\n");
}
}
return 0;
}
```

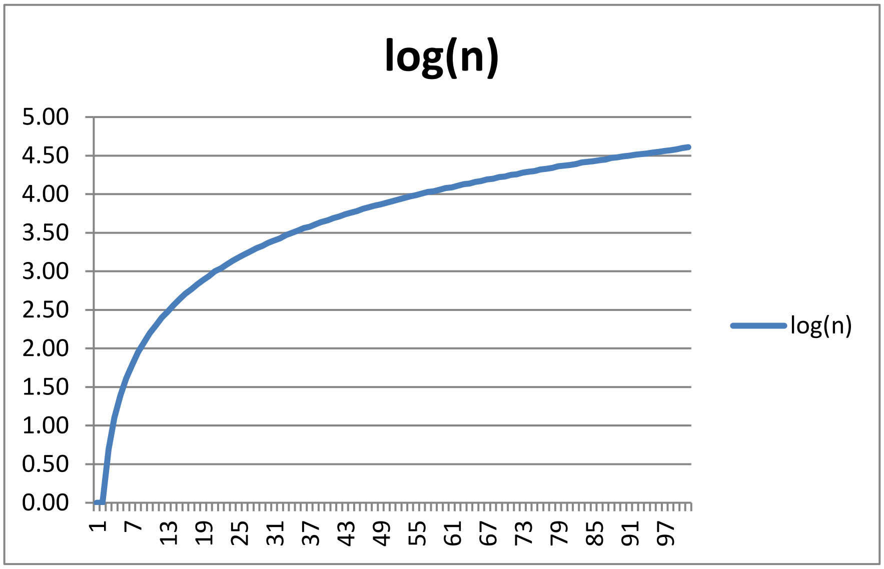
RESULT:



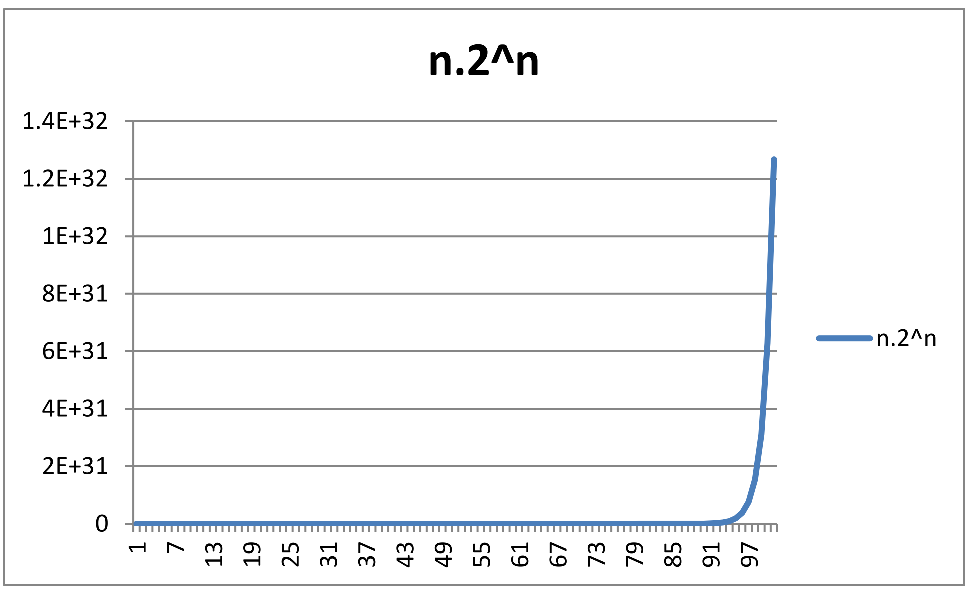
The observation from the output is that the function f1(n) increases linearly as n increases, with each value of f1(n) being equal to n.



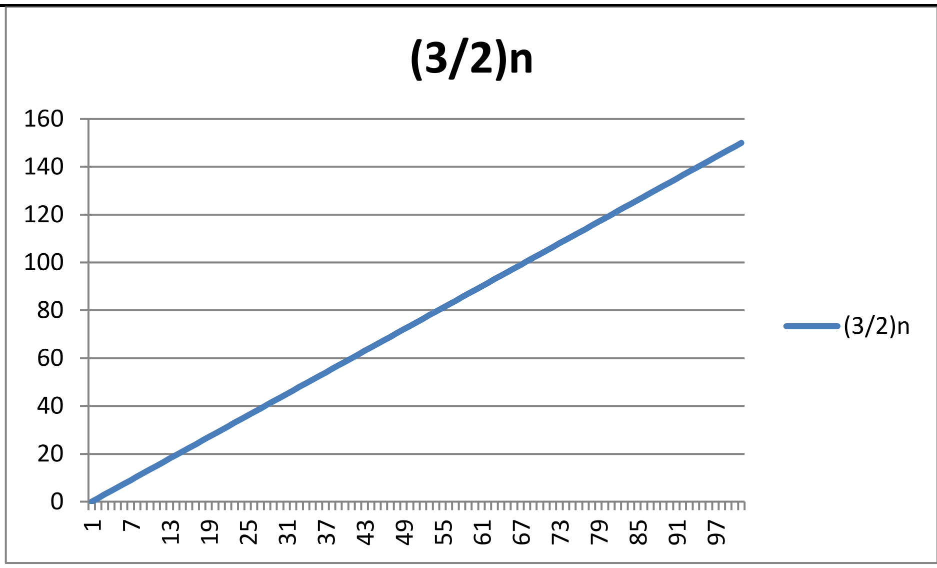
The data represents the cube of numbers from 0 to 100. The values increase rapidly as n increases, indicating that the cube function grows quickly. Additionally, the graph of the data would show a smooth curve that increases at an increasing rate, with a concave upward shape.



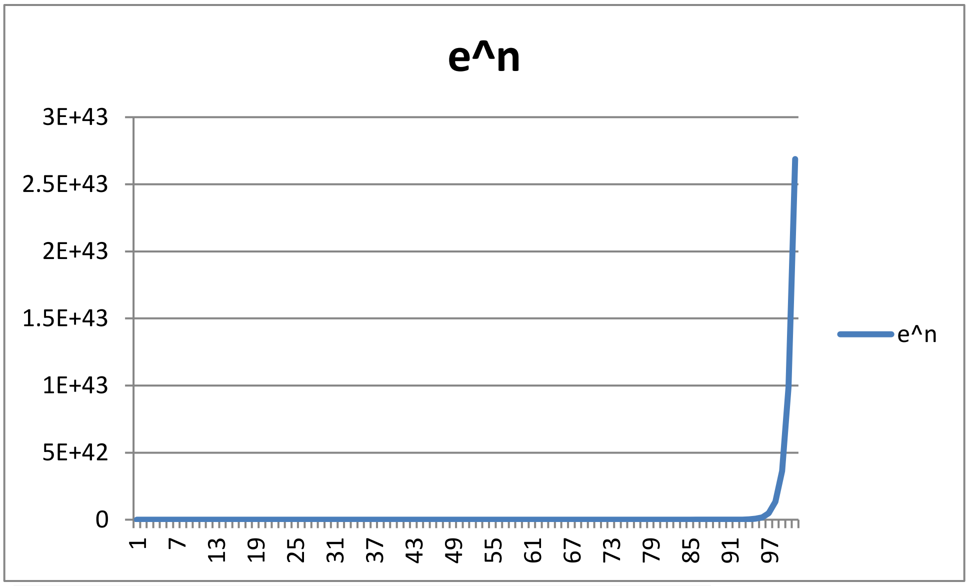
As n increases, the logarithms increase as well, but at a decreasing rate. In other words, the function log(n) grows more slowly than n itself. Additionally, the logarithm of 1 is 0, and the logarithm of 0 is negative infinity.



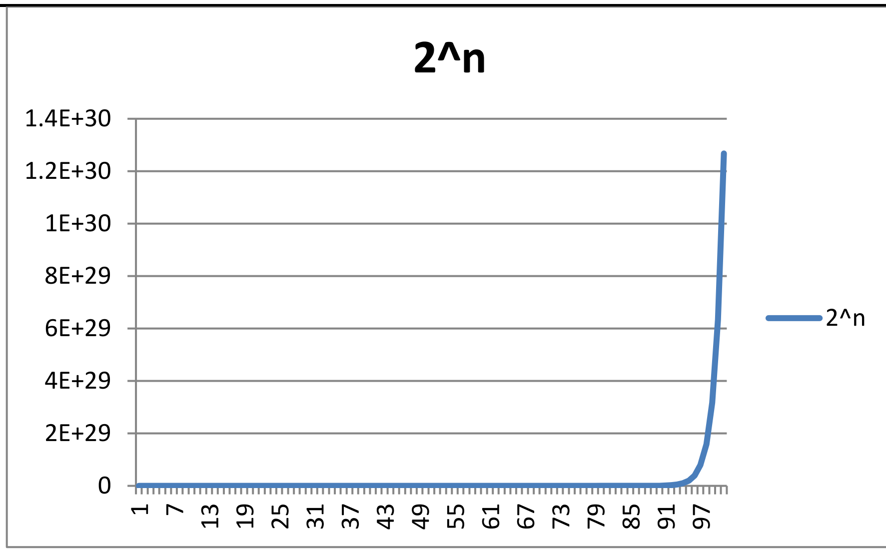
From this graph we observe that, for the values 0-90 the curve increases steadily and sort of linearly but from 90-100 it shoots rapidly



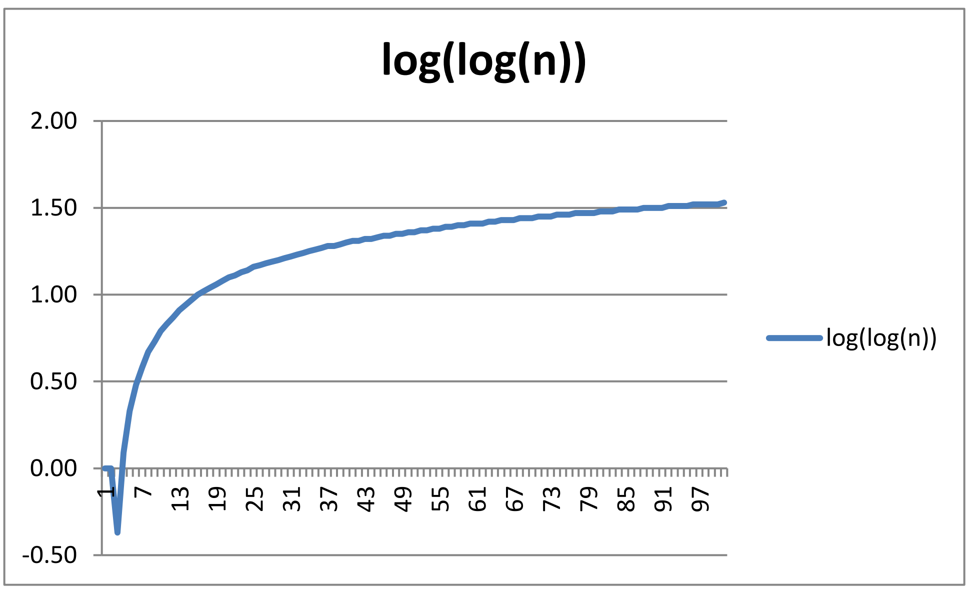
From this graph we can observe that the value of function increases linearly as the value of n increases



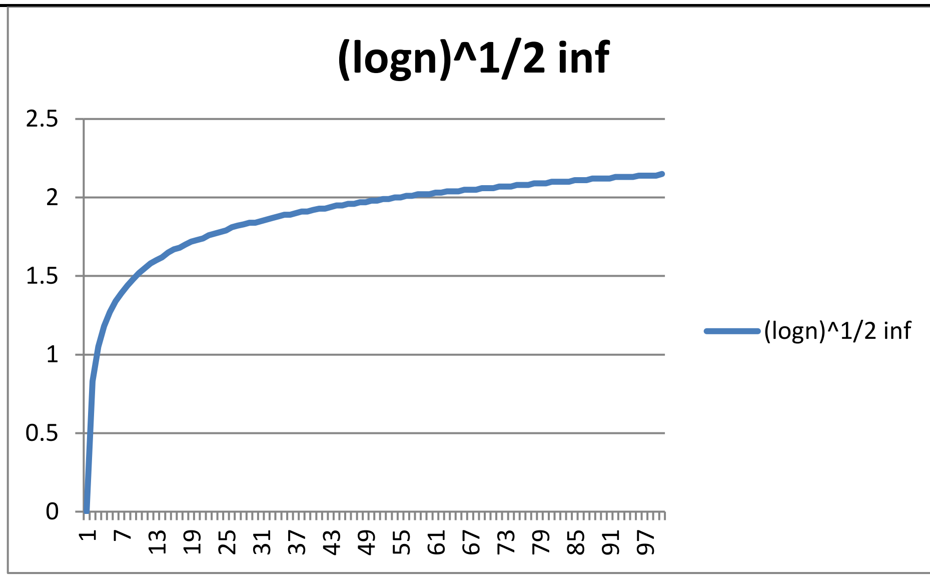
As n increases, the value of e^n also increases rapidly.



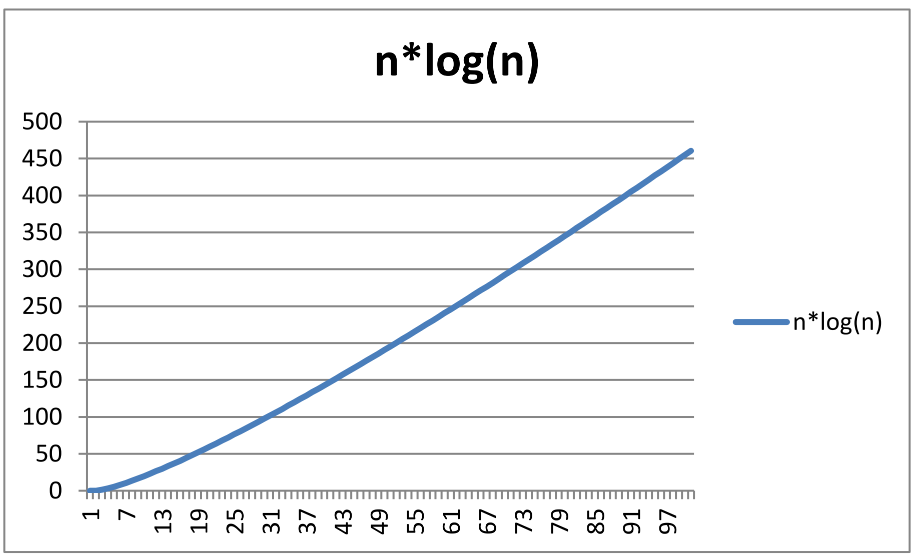
It increases steadily in the beginning and from 94 or 95 increases sharply.



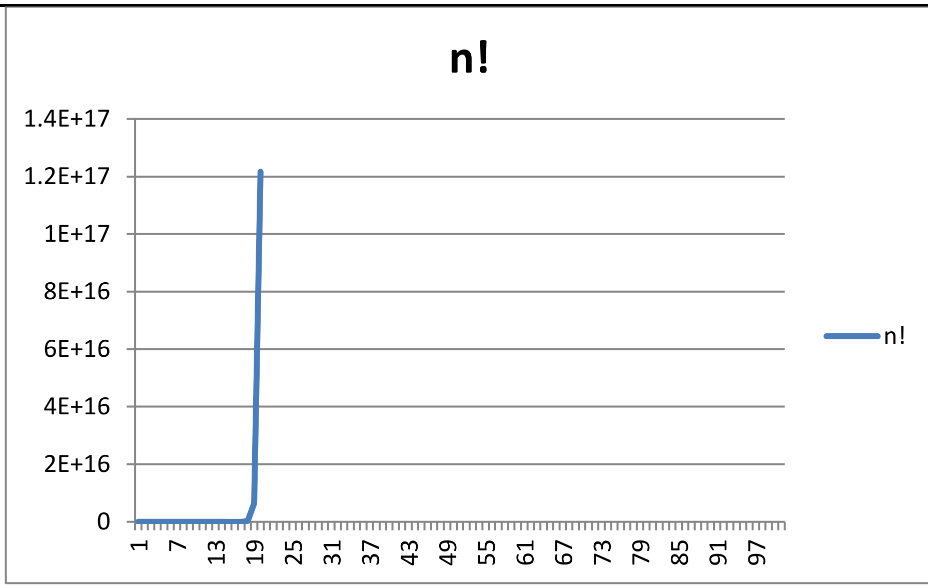
In this graph, we observe that for values 2-25, the graph grows exponentially and then increases linearly and steadily. Here we don’t consider n=0,1 as log 0 is not defined and as log 1 is 0 then log(log 1) cannot be defined too.



This graph increases rapidly from 1 to 12-13 and then becomes steady.



The curve in this graph increases somewhat linearly as the value of n increases

	 The curve for factorial is linear from 0-17 and then shoots rapidly from 18-19.
CONCLUSION:	Thus we have implemented and plotted graphs for different functions and observed their growth for various cases.