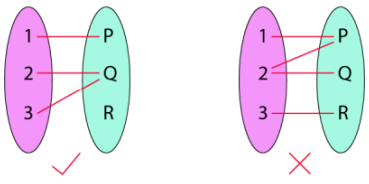
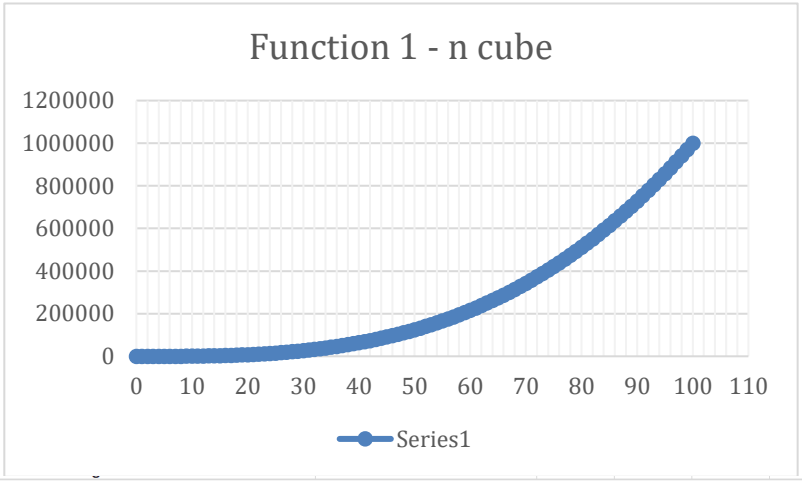
NAME:	Pratham Jain
UID:	2021300051- COMPS A (C-batch)
SUBJECT	DAA
EXPERIMENT NO :	1 A
DATE OF PERFORMANCE	30/01/23
DATE OF SUBMISSION	06/02/23
AIM:	To implement the various functions e.g. linear, non-linear, quadratic, exponential etc.
PROBLEM STATEMENT 1:	**Problem Definition & Assumptions** – For this experiment, you have to implement at least 10 functions from the following list. $(\frac{3}{2})^n$ n^3 $\lg^2 n$ $\lg(n!)$ 2^{2^n} $n^{1/\lg n}$ $\ln\ln n$ $\lg^* n$ $n \cdot 2^n$ $n^{\lg\lg n}$ $\ln n$ $2^{\lg^* n}$ $2^{\lg n}$ $(\lg n)^{\lg n}$ e^n $(\lg n)!$ $(\sqrt{2})^{\lg n}$ $\sqrt{\lg n}$ $\lg^*(\lg n)$ $2^{\sqrt{2\lg n}}$ n 2^n $n\lg n$ $2^{2^{n+1}}$ Note – *lg* denotes for log_2 and *le* denotes log_e The input (i.e. *n*) to all the above functions varies from 0 to 100 with increment of 1. Then add the function n! in the list and execute the same for n from 0 to 20.
THEORY	**Details** – 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. 

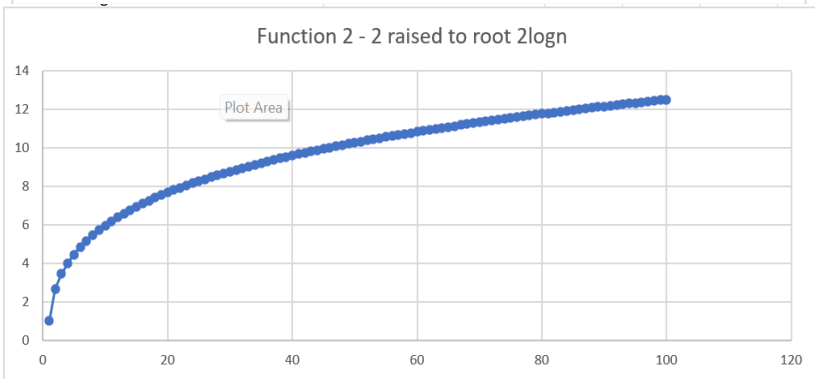
PROGRAM:

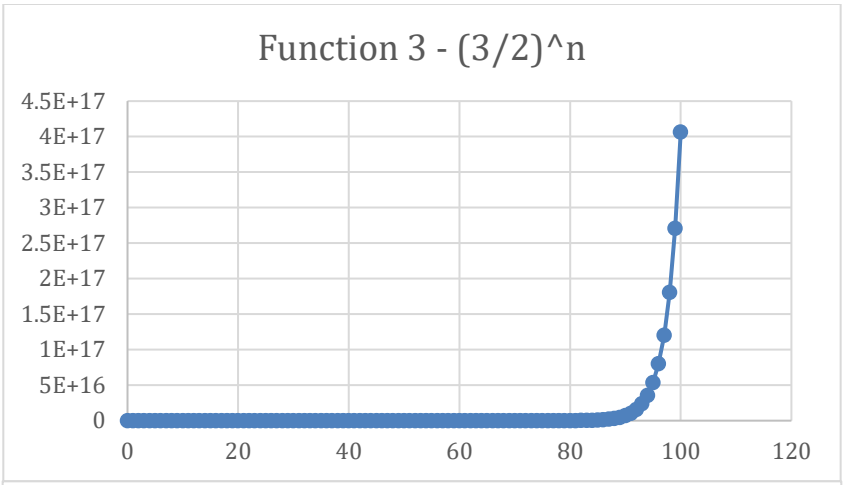
```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

long fact(int num){
    if(num == 0){
        return num;
    }
    else{
        return num*fact(num-1);
    }
}
void main(){
    printf("Function 1: n cube\n");
    for(double i=0;i<=100;i++){
        printf("%.0lf\n",pow(i,3));

    }
    printf("Function 2: 2 raised to root 2 log n\n");
    for(double i = 0;i<=100;i++){
        printf("%.3lf\n", pow(2, sqrt(2*log2(i))));
    }
    printf("Function 3: (3/2) raised to n\n");
    for(double i = 0;i<=100;i++) {
        printf("%.3lf\n",pow((3.0/2.0),i));
    }
    printf("Function 4: n log n\n");
    for(double i = 0;i<=100;i++) {
        printf("%.3lf\n",i * log2(i));
    }
    printf("Function 5: ln n\n");
    for(double i = 0;i<=100;i++) {
        printf("%.3lf\n",log(i));
    }
    printf("Function 6: 2 raised to 2 raised to n\n");
    for(double i = 0;i<=100;i++) {
        printf("%.3lf\n",pow(2,pow(2,i)));
    }
    printf("Function 7: n\n");
    for(double i = 0;i<=100;i++) {
        printf("%.3lf\n",i);
    }
    printf("Function 8: 2 raised to 2 raised to n+1\n");
    for(double i = 0;i<=100;i++) {
```

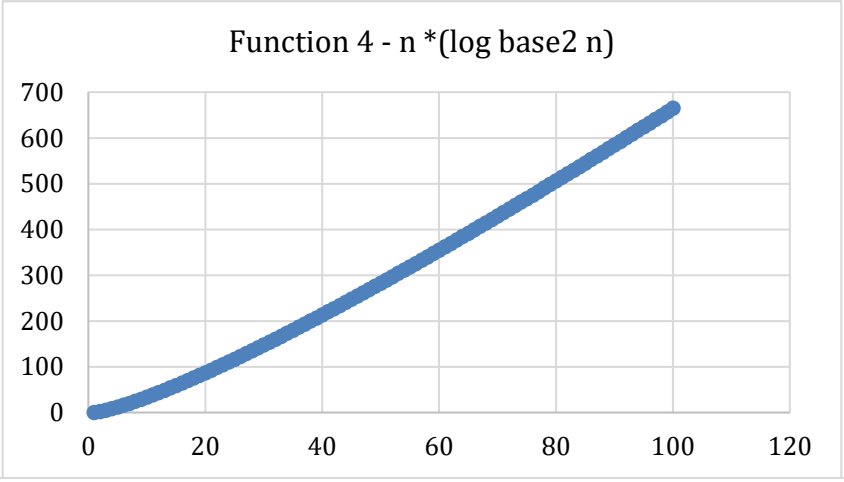
```
        printf("%.3lf\n",pow(2,pow(2,i+1)));
    }
    printf("Function 9: e raised to n\n");
    for(double i = 0;i<=100;i++) {
        printf("%.3lf\n",exp(i));
    }
    printf("Function 10: 2 raised to log n\n");
    for(double i = 0;i<=100;i++) {
        printf("%.3lf\n",pow(2,log2(i)));
    }

}
```

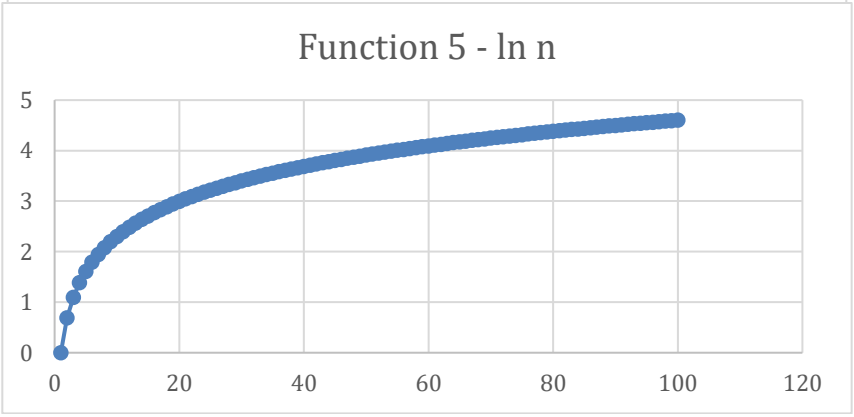
RESULT :

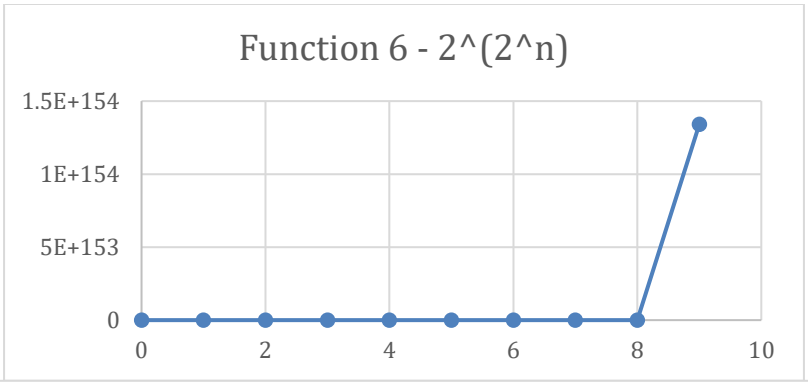


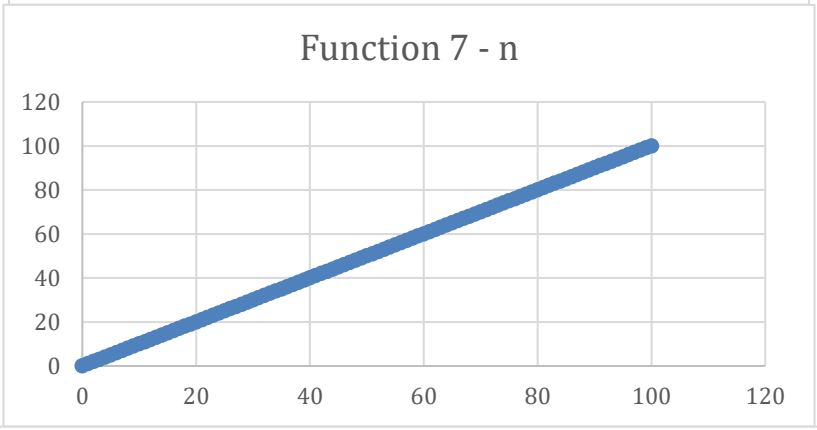


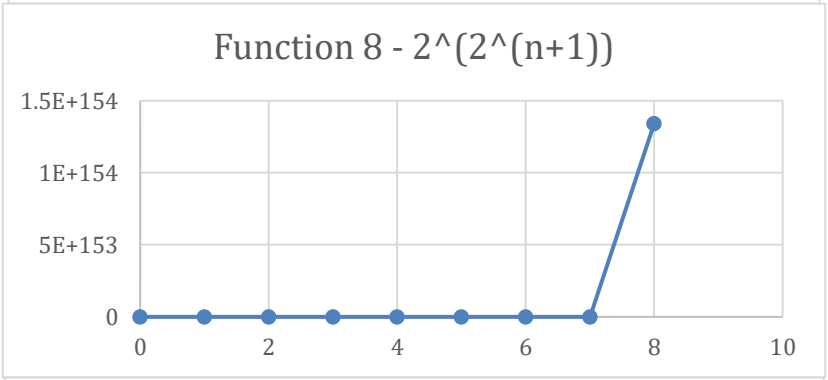

Function 3 - (3/2)^n
4.5E+17
4E+17
3.5E+17
3E+17
2.5E+17
2E+17
1.5E+17
1E+17
5E+16
0
0
20
40
60
80
100
120

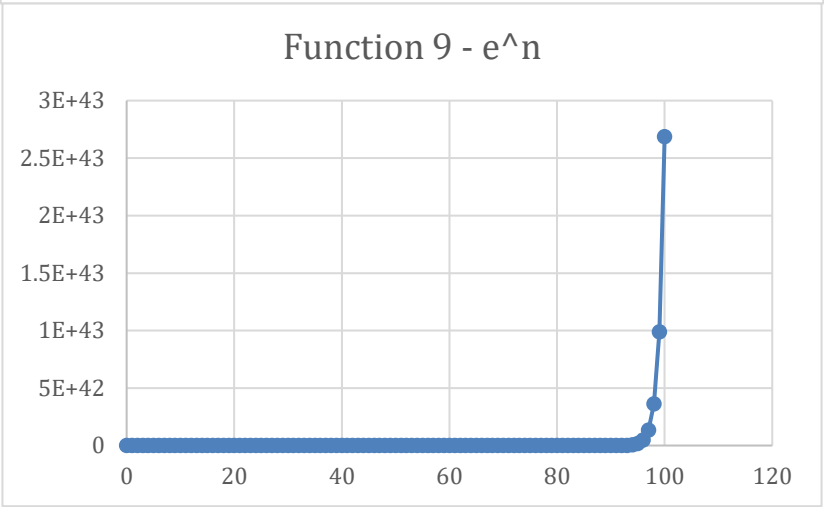

Function 4 - n *(log base2 n)
700
600
500
400
300
200
100
0
0
20
40
60
80
100
120

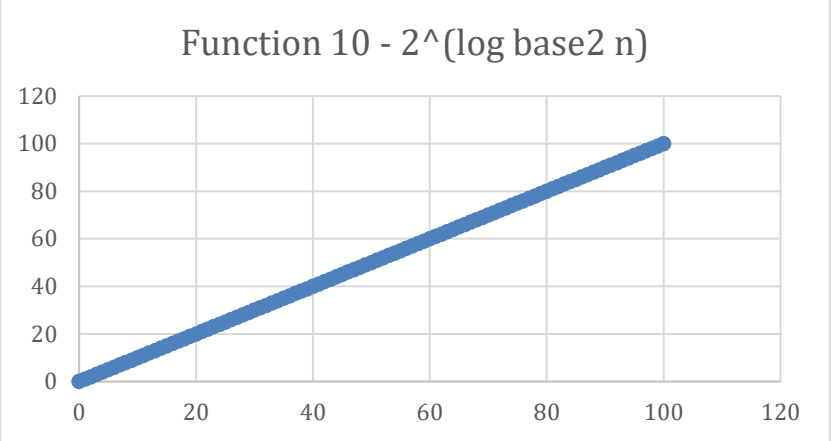

Function 5 - ln n
5
4
3
2
1
0
0
20
40
60
80
100
120


Function 6 - 2^(2^n)
1.5E+154
1E+154
5E+153
0
0
2
4
6
8
10


Function 7 - n
120
100
80
60
40
20
0
0
20
40
60
80
100
120


Function 8 - 2^(2^(n+1))
1.5E+154
1E+154
5E+153
0
0
2
4
6
8
10


Function 9 - e^n
3E+43
2.5E+43
2E+43
1.5E+43
1E+43
5E+42
0
0
20
40
60
80
100
120



CONCLUSION:	I have successfully understood learnt how to implement various linear, non-linear, exponential functions. Also compared their graphs for the given set.