Name	Stephen David Vaz
UID no.	2021700070
Experiment No.	1

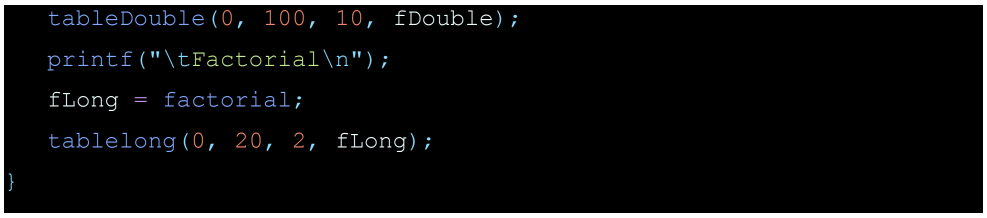
AIM:	To implement the various functions e.g. linear, non-linear, quadratic, exponential etc.
Program 1	
PROBLEM STATEMENT :	For this experiment, you have to implement at least 10 functions from the given list. The input (i.e. n) to all the above functions varies from 0 to 100 with increment of 10. Then add the function n! in the list and execute the same for n from 0 to 20 with increment of 2.
ALGORITHM/ THEORY:	**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.

PROGRAM:

```c
#include <stdio.h>
#include <math.h>
// 11th func is n factorial for 0 to 20 (0,2,4,6,8,10)

void tableDouble(int start, int end, int incr, double (*f)())
{
   FILE *fp = fopen("output.csv", "a+");
   fprintf(fp, "n, f(n)\n");
   for (int i = start; i <= end; i += incr)
   {
       printf("\t%d\t| %f\n", i, f(i));
       fprintf(fp, "%d, %f\n", i, f(i));
   }
   fclose(fp);
   printf("\n");
}
void tableInt(int start, int end, int incr, int (*f)())
```

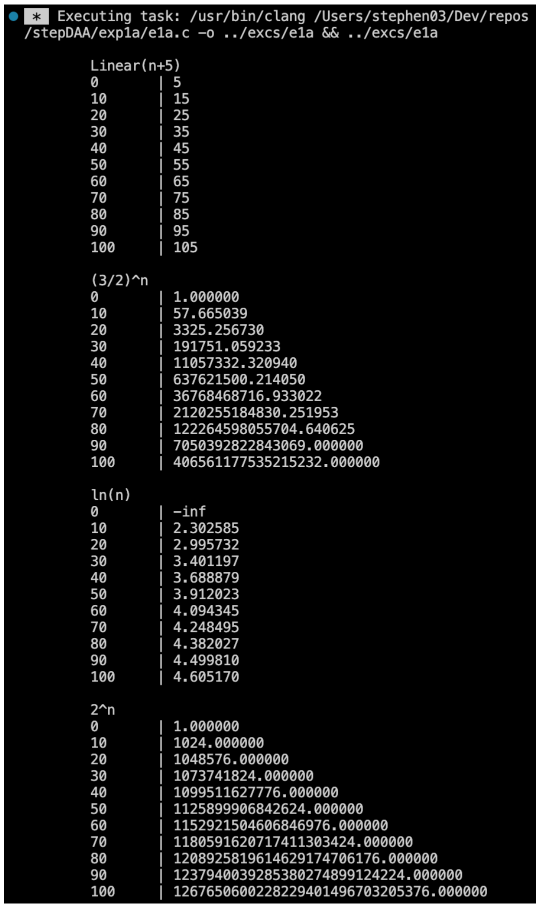
```c
{
    FILE *fp = fopen("output.csv", "a+");
    fprintf(fp, "n, f(n)\n");
    for (int i = start; i <= end; i += incr)
    {
        printf("\t%d\t| %d\n", i, f(i));
        fprintf(fp, "%d, %d\n", i, f(i));
    }
    fclose(fp);
    printf("\n");
}
void tablelong(int start, int end, int incr, unsigned long long (*f)())
{
    FILE *fp = fopen("output.csv", "a+");
    fprintf(fp, "n, f(n)\n");
    for (int i = start; i <= end; i += incr)
    {
        printf("\t%d\t| %lld\n", i, f(i));
        fprintf(fp, "%d, %lld\n", i, f(i));
    }
    fclose(fp);
    printf("\n");
}

int linear(int x)
{
    return x + 5;
    // linear function
}
double fun1(int x) { return pow(1.5, x); }
double ln(int x) { return log(x); }
double twon(int x) { return pow(2, x); }
double lnlnn(int x) { return log(log(x)); }
double rtln(int x) { return sqrt(log10(x)); }
double en(int x) { return exp(x); }
double logsqn(int x) { return pow(log2(x), 2); }

unsigned long long factorial(unsigned long f)
{
    if (f)
```

```c
        return (f * factorial(f - 1));
    return 1;
}

double nloglogn(int x) { return pow(x, log2(log2(x))); }
double sqrt2logn(int x) { return pow(sqrt(2), log2(x)); }

int (*fInt)(int);
double (*fDouble)(int);
unsigned long long (*fLong)(unsigned long f);
int main()
{
    printf("\tLinear(n+5)\n");
    fInt = linear;
    tableInt(0, 100, 10, fInt);
    printf("\t(3/2)^n\n");
    fDouble = fun1;
    tableDouble(0, 100, 10, fDouble);
    printf("\tln(n)\n");
    fDouble = ln;
    tableDouble(0, 100, 10, fDouble);
    printf("\t2^n\n");
    fDouble = twon;
    tableDouble(0, 100, 10, fDouble);
    printf("\tln ln n\n");
    fDouble = lnlnn;
    tableDouble(0, 100, 10, fDouble);
    printf("\tsqroot(log(n))\n");
    fDouble = rtln;
    tableDouble(0, 100, 10, fDouble);
    printf("\te^n\n");
    fDouble = en;
    tableDouble(0, 100, 10, fDouble);
    printf("\t(log n)^2\n");
    fDouble = logsqn;
    tableDouble(0, 100, 10, fDouble);
    printf("\tn^(log(log(n)))\n");
    fDouble = nloglogn;
    tableDouble(0, 100, 10, nloglogn);
    printf("\tsqrt(2)^logn\n");
    fDouble = sqrt2logn;
```

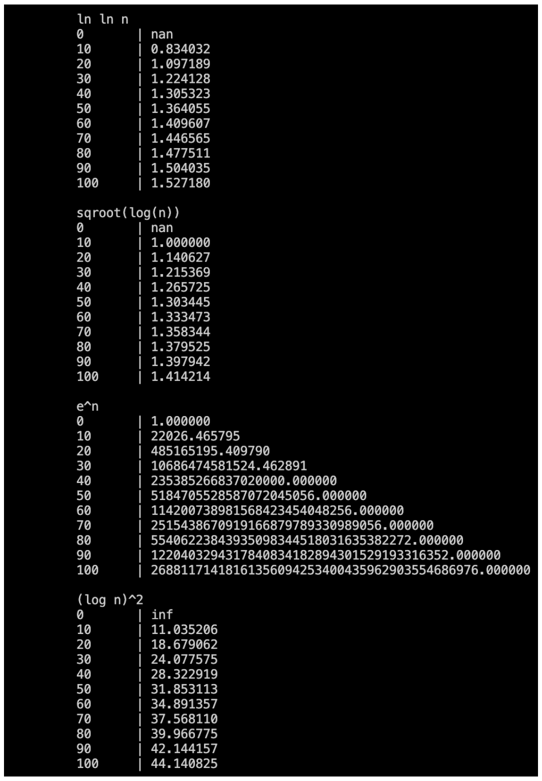


```
    tableDouble(0, 100, 10, fDouble);
    printf("\tFactorial\n");
    fLong = factorial;
    tablelong(0, 20, 2, fLong);
}
```

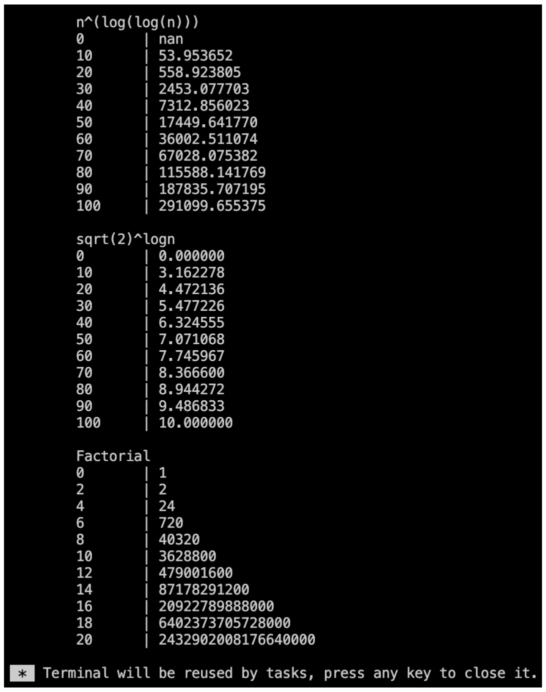
RESULT:



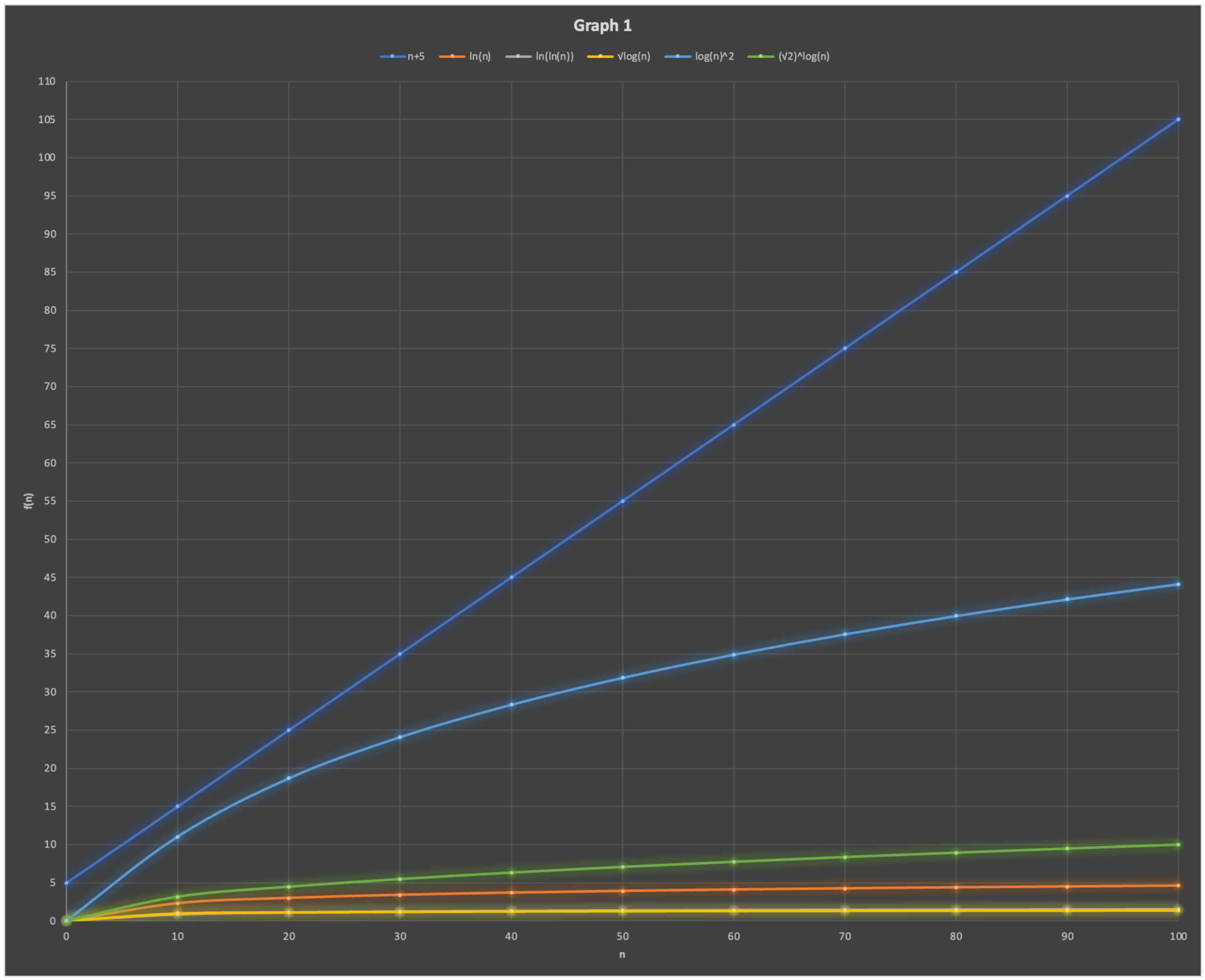
```
● * Executing task: /usr/bin/clang /Users/stephen03/Dev/repos
  /stepDAA/exp1a/e1a.c -o ../excs/e1a && ../excs/e1a

        Linear(n+5)
        0       | 5
        10      | 15
        20      | 25
        30      | 35
        40      | 45
        50      | 55
        60      | 65
        70      | 75
        80      | 85
        90      | 95
        100     | 105

        (3/2)^n
        0       | 1.000000
        10      | 57.665039
        20      | 3325.256730
        30      | 191751.059233
        40      | 11057332.320940
        50      | 637621500.214050
        60      | 36768468716.933022
        70      | 2120255184830.251953
        80      | 122264598055704.640625
        90      | 7050392822843069.000000
        100     | 406561177535215232.000000

        ln(n)
        0       | -inf
        10      | 2.302585
        20      | 2.995732
        30      | 3.401197
        40      | 3.688879
        50      | 3.912023
        60      | 4.094345
        70      | 4.248495
        80      | 4.382027
        90      | 4.499810
        100     | 4.605170

        2^n
        0       | 1.000000
        10      | 1024.000000
        20      | 1048576.000000
        30      | 1073741824.000000
        40      | 1099511627776.000000
        50      | 1125899906842624.000000
        60      | 1152921504606846976.000000
        70      | 1180591620717411303424.000000
        80      | 1208925819614629174706176.000000
        90      | 1237940039285380274899124224.000000
        100     | 1267650600228229401496703205376.000000
```

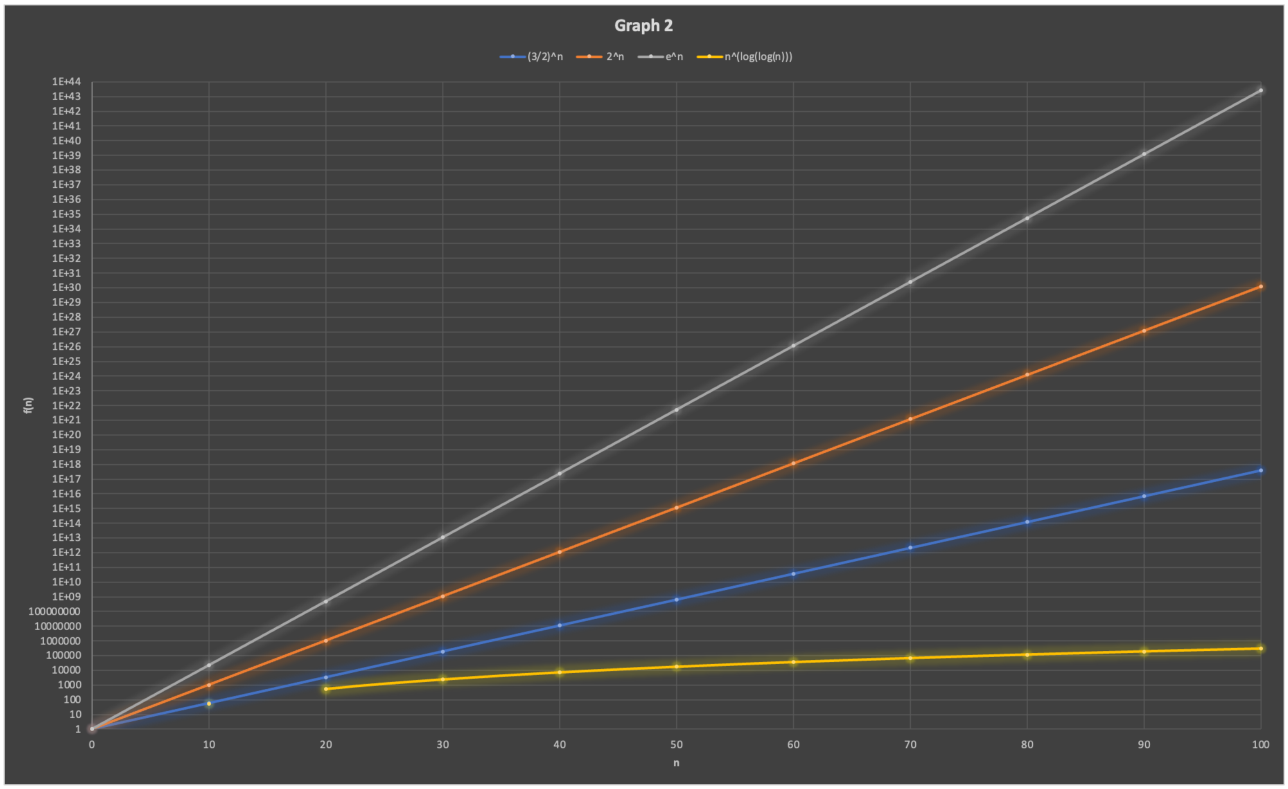


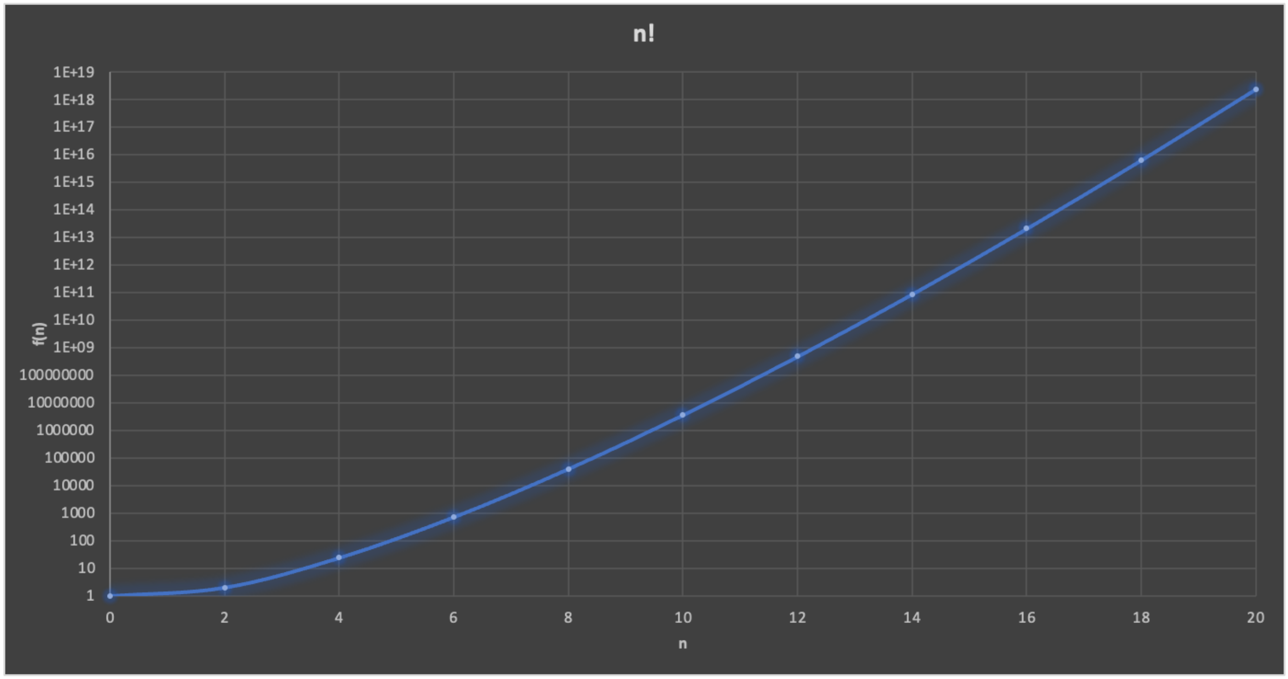
```
ln ln n
0       | nan
10      | 0.834032
20      | 1.097189
30      | 1.224128
40      | 1.305323
50      | 1.364055
60      | 1.409607
70      | 1.446565
80      | 1.477511
90      | 1.504035
100     | 1.527180

sqroot(log(n))
0       | nan
10      | 1.000000
20      | 1.140627
30      | 1.215369
40      | 1.265725
50      | 1.303445
60      | 1.333473
70      | 1.358344
80      | 1.379525
90      | 1.397942
100     | 1.414214

e^n
0       | 1.000000
10      | 22026.465795
20      | 485165195.409790
30      | 10686474581524.462891
40      | 235385266837020000.000000
50      | 5184705528587072045056.000000
60      | 114200738981568423454048256.000000
70      | 2515438670919166879789330989056.000000
80      | 55406223843935098344518031635382272.000000
90      | 1220403294317840834182894301529193316352.000000
100     | 26881171418161356094253400435962903554686976.000000

(log n)^2
0       | inf
10      | 11.035206
20      | 18.679062
30      | 24.077575
40      | 28.322919
50      | 31.853113
60      | 34.891357
70      | 37.568110
80      | 39.966775
90      | 42.144157
100     | 44.140825
```



```
n^(log(log(n)))
0       | nan
10      | 53.953652
20      | 558.923805
30      | 2453.077703
40      | 7312.856023
50      | 17449.641770
60      | 36002.511074
70      | 67028.075382
80      | 115588.141769
90      | 187835.707195
100     | 291099.655375

sqrt(2)^logn
0       | 0.000000
10      | 3.162278
20      | 4.472136
30      | 5.477226
40      | 6.324555
50      | 7.071068
60      | 7.745967
70      | 8.366600
80      | 8.944272
90      | 9.486833
100     | 10.000000

Factorial
0       | 1
2       | 2
4       | 24
6       | 720
8       | 40320
10      | 3628800
12      | 479001600
14      | 87178291200
16      | 20922789888000
18      | 6402373705728000
20      | 2432902008176640000

 *  Terminal will be reused by tasks, press any key to close it.
```


Graph 1
n+5
ln(n)
ln(ln(n))
√log(n)
log(n)^2
(√2)^log(n)
f(n)
n
0
10
20
30
40
50
60
70
80
90
100
0
5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
100
105
110





CONCLUSION:	Successfully implemented various functions in C and observed their outputs for a set of numbers both in tabular as well as graphical format.