NAME:	Shubham Vishwakarma
UID No.	2021700071
BRANCH:	S.Y CSE-DS
BATCH:	D
SUBJECT	Design and Analysis of Algorithms
EXPERIMENT No.	1A
DATE:	25/01/2023

AIM:	**To implement the various functions e.g. linear, non-linear, quadratic, exponential etc.**
PROGRAM:	

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

double fun1(double n)
{
    return log(n);
}
double fun2(double n)
{
    return n * log2(n);
}
double fun3(double n)
{
    return n;
}
double fun4(double n)
{
    return log2(log2(n));
}
double fun5(double n)
{
    return pow(sqrt(2), log2(n));
}
double fun6(double n)
{
    return pow(2, log2(n));
}
double fun7(double n)
{
    return pow(2, sqrt(2 * log2(n)));
}
double fun8(double n)
{
    return sqrt(log2(n));
}
```

```
double fun9(double n)
{
    return log2(n);
}
double fun10(double n)
{
    return pow(n, 1 / log2(n));
}

double fun11(double n)
{
    double fac = 1;
    for (int i = 1; i <= n; i++)
    {
        fac = fac * i;
    }
    return fac;
}

void main()
{
    int A[] = {0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100};

printf("\nX\tFun_1\tFun_2\tFun_3\tFun_4\tFun_5\tFun_6\tFun_7\tFun_8\tFun_
9\tFun_10\n\n");
    for (int i = 0; i <= 10; i++)
    {
        printf("%d ", A[i]);
        printf("\t");
        printf("%0.2f", fun1(A[i]));
        printf("\t");
        printf("%0.2f", fun2(A[i]));
        printf("\t");
        printf("%0.2f", fun3(A[i]));
        printf("\t");
        printf("%0.2f", fun4(A[i]));
        printf("\t");
        printf("%0.2f", fun5(A[i]));
        printf("\t");
        printf("%0.2f", fun6(A[i]));
        printf("\t");
        printf("%0.2f", fun7(A[i]));
        printf("\t");
        printf("%0.2f", fun8(A[i]));
        printf("\t");
        printf("%0.2f", fun9(A[i]));
        printf("\t");
        printf("%0.2f\n", fun10(A[i]));
```

```
    }
    printf("\nX\tFun_11\n");
    for (int i = 0; i <= 20; i += 2)
    {
        printf("%d ", i);
        printf("\t");
        printf("%0.2f\n", fun11(i));
    }
}
```

RESULT:

```
PS C:\Users\smsha\Desktop\SEM 4\DAA\Practicals> & .\"Exp1A.exe"

 X       Fun_1   Fun_2   Fun_3   Fun_4   Fun_5   Fun_6   Fun_7   Fun_8   Fun_9   Fun_10

 0       -1.#J   -1.#J   0.00    -1.#J   0.00    0.00    -1.#J   -1.#J   -1.#J   1.00
 10      2.30    33.22   10.00   1.73    3.16    10.00   5.97    1.82    3.32    2.00
 20      3.00    86.44   20.00   2.11    4.47    20.00   7.67    2.08    4.32    2.00
 30      3.40    147.21  30.00   2.29    5.48    30.00   8.77    2.22    4.91    2.00
 40      3.69    212.88  40.00   2.41    6.32    40.00   9.60    2.31    5.32    2.00
 50      3.91    282.19  50.00   2.50    7.07    50.00   10.27   2.38    5.64    2.00
 60      4.09    354.41  60.00   2.56    7.75    60.00   10.83   2.43    5.91    2.00
 70      4.25    429.05  70.00   2.62    8.37    70.00   11.32   2.48    6.13    2.00
 80      4.38    505.75  80.00   2.66    8.94    80.00   11.76   2.51    6.32    2.00
 90      4.50    584.27  90.00   2.70    9.49    90.00   12.15   2.55    6.49    2.00
 100     4.61    664.39  100.00  2.73    10.00   100.00  12.51   2.58    6.64    2.00

 X       Fun_11
 0       1.00
 2       2.00
 4       24.00
 6       720.00
 8       40320.00
 10      3628800.00
 12      479001600.00
 14      87178291200.00
 16      20922789888000.00
 18      6402373705728000.00
 20      2432902008176640000.00
PS C:\Users\smsha\Desktop\SEM 4\DAA\Practicals> S
```

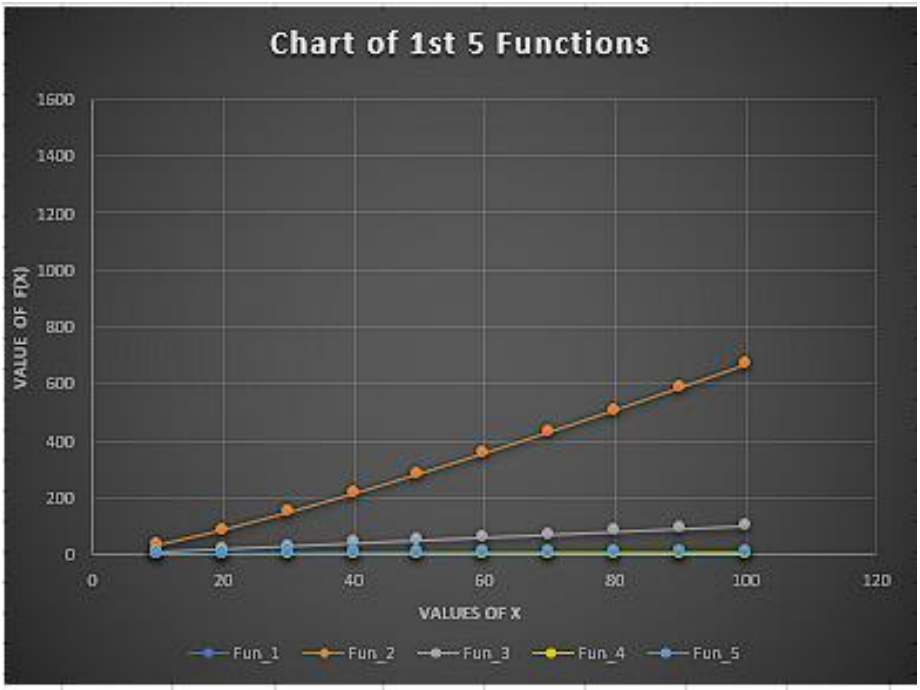


Chart of 1st 5 Functions
1600
1400
1200
1000
800
600
400
200
0
VALUE OF F(X)
0
20
40
60
80
100
120
VALUES OF X
Fun_1
Fun_2
Fun_3
Fun_4
Fun_5

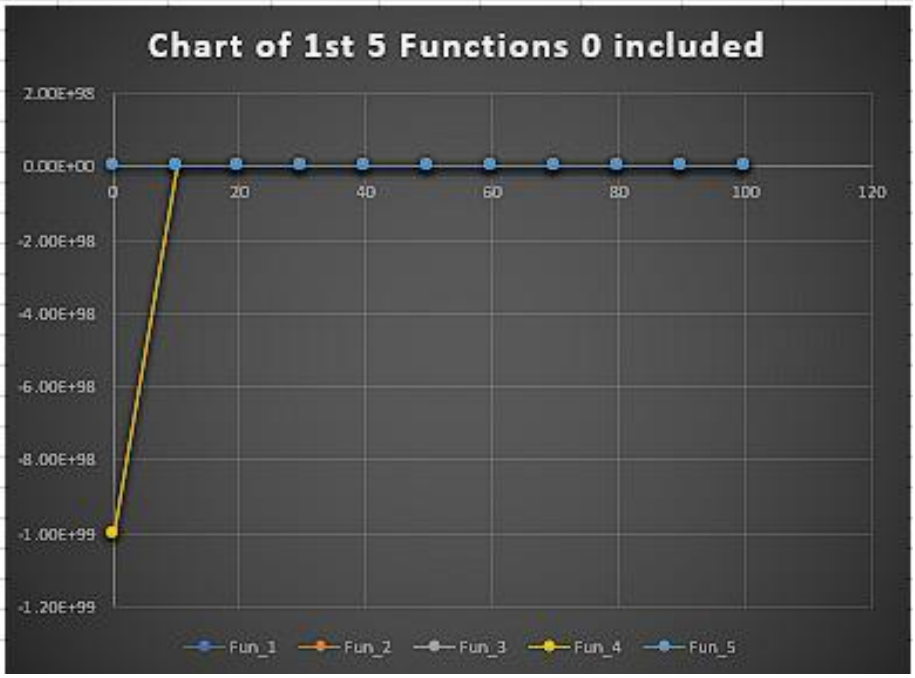


Chart of 1st 5 Functions 0 included
2.00E+98
0.00E+00
-2.00E+98
-4.00E+98
-6.00E+98
-8.00E+98
-1.00E+99
-1.20E+99
0
20
40
60
80
100
120
Fun_1
Fun_2
Fun_3
Fun_4
Fun_5

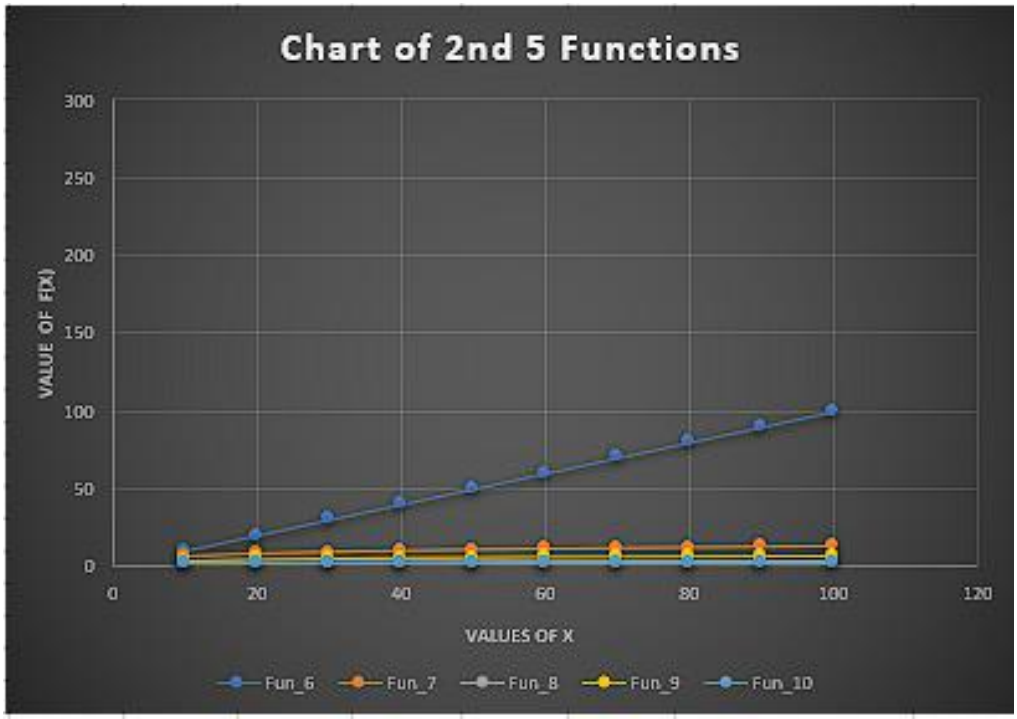


Chart of 2nd 5 Functions
300
250
200
150
100
50
0
VALUE OF F(X)
0
20
40
60
80
100
120
VALUES OF X
Fun_6
Fun_7
Fun_8
Fun_9
Fun_10

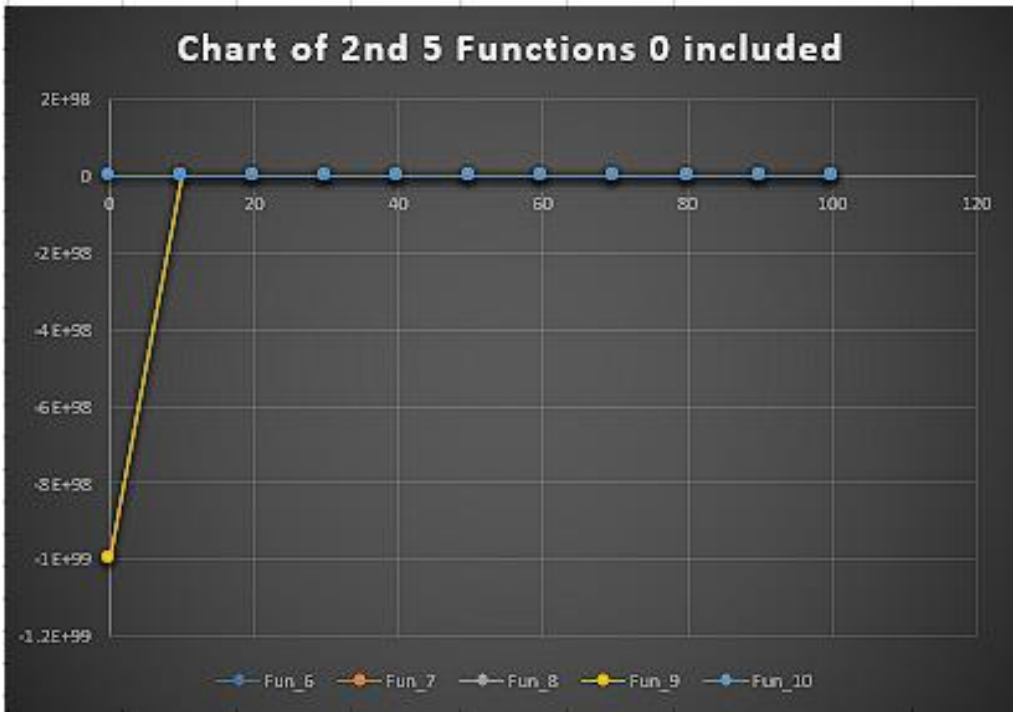
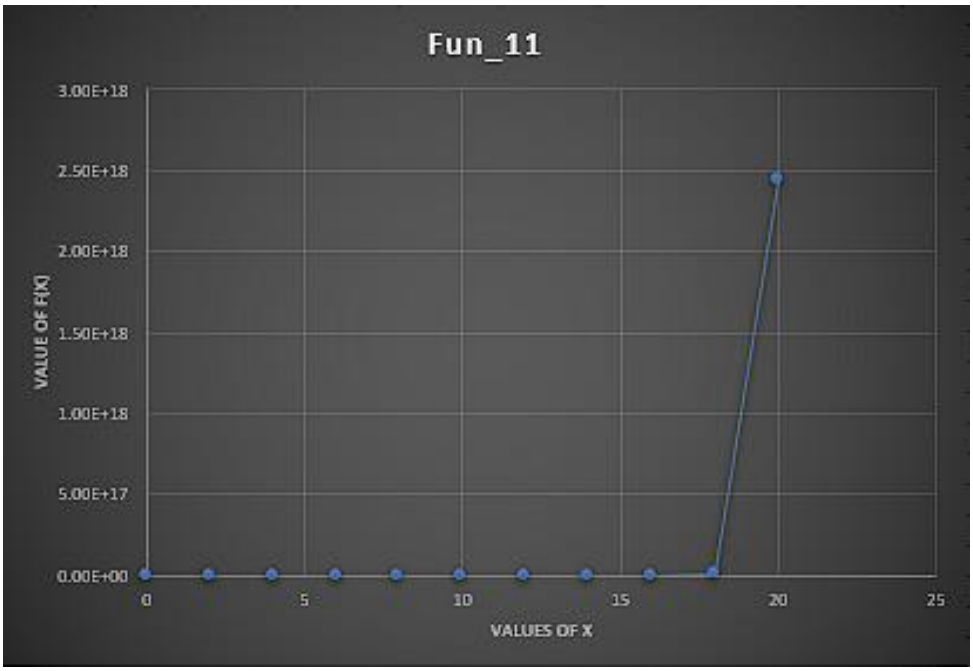


Chart of 2nd 5 Functions 0 included
2E+98
0
-2E+98
-4E+98
-6E+98
-8E+98
-1E+99
-1.2E+99
0
20
40
60
80
100
120
Fun_6
Fun_7
Fun_8
Fun_9
Fun_10

	
CONCLUSION:	By studying the behaviour graph of different functions we can conclude that the straight line passing through origin have time complexity of O(n), proportional to n elements. The curves below the straight line have time complexiy less than O(n), hence the lower curve function are less complex and efficient to execute. The curves above the straight have time complexity greater than O(n), thus are very complex and takes more time to execute.