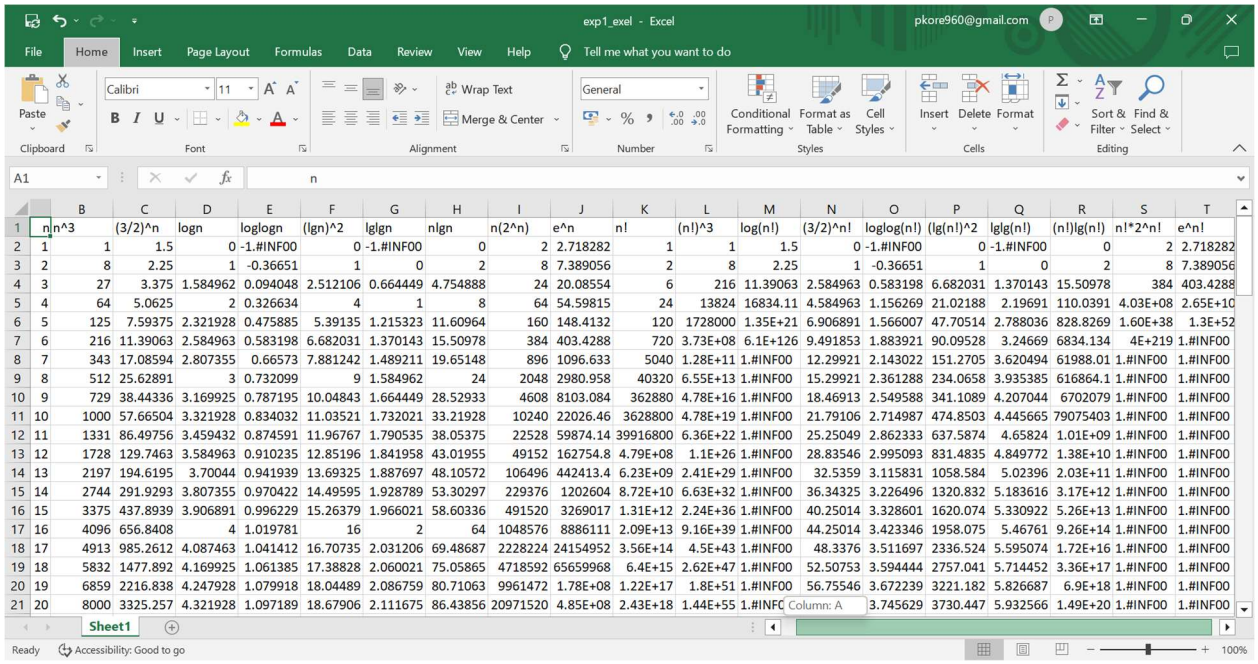
NAME:	Pranav Sadanand Kore
UID:	2021300065
SUBJECT	Design and Analysis of Algorithm
EXPERIMENT NO :	1
AIM:	To implement the various functions e.g. linear, non-linear, quadratic, exponential etc.
PROGRAM:	

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

double Factorial(double n)
{
    if(n==1 || n==0)
    {
        return 1;
    }
    double a;
    a=Factorial(n-1);
    return n*a;
}

double n3(double n)
{
    double cube;
    cube=pow(n,3);
    return cube;
}

double pow32n(double n)
{
    double result;
    result=pow(3.0/2.0,n);
    return result;
}

double lgn(double n)
```

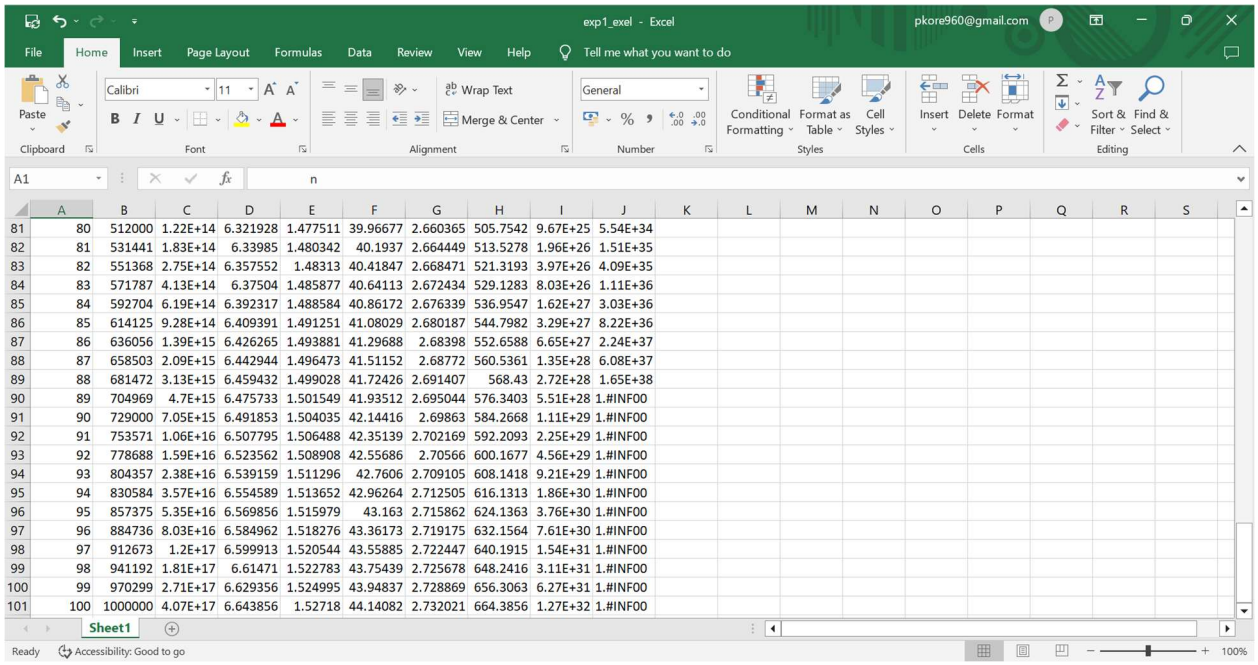
```
{
    double result;
    result=log2(n);
    return result;
}

double lnlnn(double n)
{
    double result;
    result=log(log(n));
    return result;
}

double lglgn(double n)
{
    double result;
    result=log2(log2(n));
    return result;
}

double lg2n(double n)
{
    double result;
    result=pow(log2(n),2);
    return result;
}

double nlgn(double n)
{
    double result;
    result=n*log2(n);
    return result;
}

double n2n(double n)
{
    double result;
    result=n*pow(2,n);
    return result;
}

double en(double n)
{
    double result;
```

```
    result=exp(n);
    return result;
}

int main()
{
    printf("n      n^3      (3/2)^n      logn      loglogn
(lgn)^2      lglgn      nlgn      n(2^n)      e^n\n");
    for(int i=1; i<=100; i++)
    {
        printf("%lf     %lf       %lf           %lf        %lf
%lf
%lf",i,n3(i),pow32n(i),lgn(i),lnlnn(i),lg2n(i),lglgn(i));
        printf("        %lf        %lf
%lf",nlgn(i),n2n(i),en(i));

        if(i<=20)
        {
            double a;
            a=Factorial(i);
            printf("      %lf     %lf       %lf           %lf
%Lf           %lf
%lf",a,n3(a),pow32n(a),lgn(a),lnlnn(a),lg2n(a),lglgn(a));
            printf("      %lf        %lf
%Lf",nlgn(a),n2n(a),en(a));
        }
        printf("\n");
    }
}
```

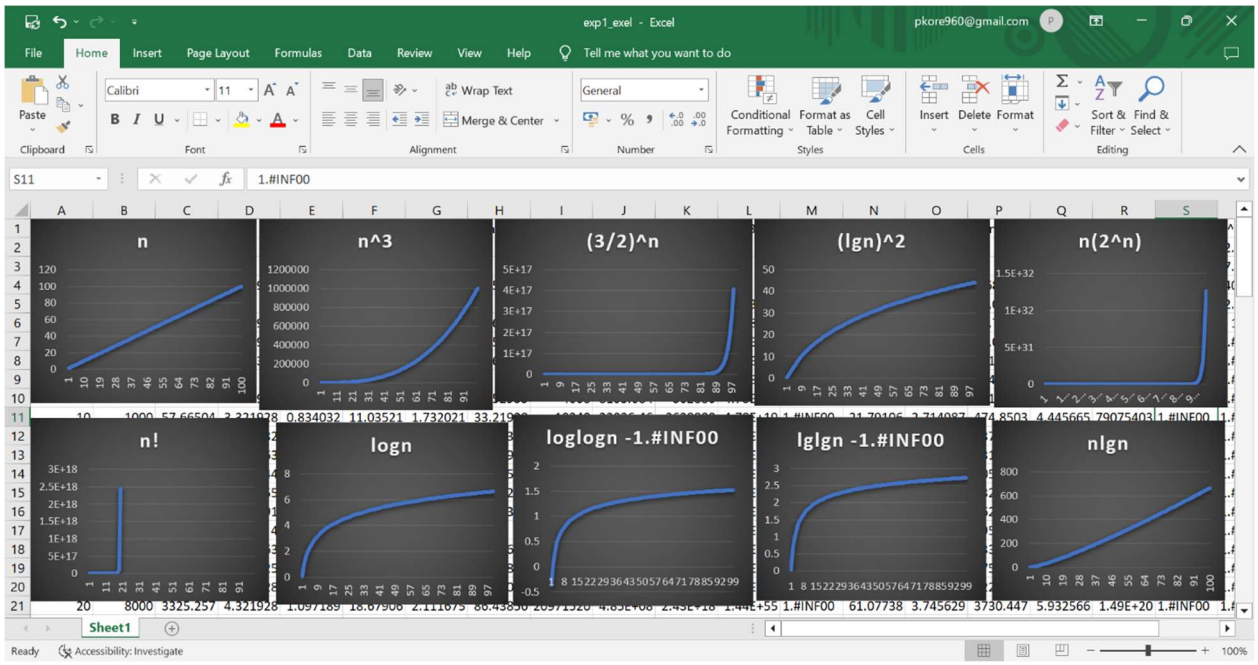
RESULT (SNAPSHOT)



	n	n^3	(3/2)^n	logn	loglogn	(lgn)^2	lglgn	nlgn	n(2^n)	e^n	n!	(n!)^3	log(n!)	(3/2)^n!	loglog(n!)	(lg(n!)^2	lglg(n!)	(n!)lg(n!)	n!*2^n!	e^n!
2	1	1	1.5	0	-1.#INF00	0	-1.#INF00	0	2	2.718282	1	1	1.5	0	-1.#INF00	0	-1.#INF00	0	2	2.718282
3	2	8	2.25	1	-0.36651	1	0	2	8	7.389056	2	8	2.25	1	-0.36651	1	0	2	8	7.389056
4	3	27	3.375	1.584962	0.094048	2.512106	0.664449	4.754888	24	20.08554	6	216	11.39063	2.584963	0.583198	6.682031	1.370143	15.50978	384	403.4288
5	4	64	5.0625	2	0.326634	4	1	8	64	54.59815	24	13824	16834.11	4.584963	1.156269	21.02188	2.19691	110.0391	4.03E+08	2.65E+10
6	5	125	7.59375	2.321928	0.475885	5.39135	1.215323	11.60964	160	148.4132	120	1728000	1.35E+21	6.906891	1.566007	47.70514	2.788036	828.8269	1.60E+38	1.3E+52
7	6	216	11.39063	2.584963	0.583198	6.682031	1.370143	15.50978	384	403.4288	720	3.73E+08	6.1E+126	9.491853	1.883921	90.09528	3.24669	6834.134	4E+219	1.#INF00
8	7	343	17.08594	2.807355	0.66573	7.881242	1.489211	19.65148	896	1096.633	5040	1.28E+11	1.#INF00	12.29921	2.143022	151.2705	3.620494	61988.01	1.#INF00	1.#INF00
9	8	512	25.62891	3	0.732099	9	1.584962	24	2048	2980.958	40320	6.55E+13	1.#INF00	15.29921	2.361288	234.0658	3.935385	616864.1	1.#INF00	1.#INF00
10	9	729	38.44336	3.169925	0.787195	10.04843	1.664449	28.52933	4608	8103.084	362880	4.78E+16	1.#INF00	18.46913	2.549588	341.1089	4.207044	6702079	1.#INF00	1.#INF00
11	10	1000	57.66504	3.321928	0.834032	11.03521	1.732021	33.21928	10240	22026.46	3628800	4.78E+19	1.#INF00	21.79106	2.714987	474.8503	4.445665	79075403	1.#INF00	1.#INF00
12	11	1331	86.49756	3.459432	0.874591	11.96767	1.790535	38.05375	22528	59874.14	39916800	6.36E+22	1.#INF00	25.25049	2.862333	637.5874	4.65824	1.01E+09	1.#INF00	1.#INF00
13	12	1728	129.7463	3.584963	0.910235	12.85196	1.841958	43.01955	49152	162754.8	4.79E+08	1.1E+26	1.#INF00	28.83546	2.995093	831.4835	4.849772	1.38E+10	1.#INF00	1.#INF00
14	13	2197	194.6195	3.70044	0.941939	13.69325	1.887697	48.10572	106496	442413.4	6.23E+09	2.41E+29	1.#INF00	32.5359	3.115831	1058.584	5.02396	2.03E+11	1.#INF00	1.#INF00
15	14	2744	291.9293	3.807355	0.970422	14.49595	1.928789	53.30297	229376	1202604	8.72E+10	6.63E+32	1.#INF00	36.34325	3.226496	1320.832	5.183616	3.17E+12	1.#INF00	1.#INF00
16	15	3375	437.8939	3.906891	0.996229	15.26379	1.966021	58.60336	491520	3269017	1.31E+12	2.24E+36	1.#INF00	40.25014	3.328601	1620.074	5.330922	5.26E+13	1.#INF00	1.#INF00
17	16	4096	656.8408	4	1.019781	16	2	64	1048576	8886111	2.09E+13	9.16E+39	1.#INF00	44.25014	3.423346	1958.075	5.46761	9.26E+14	1.#INF00	1.#INF00
18	17	4913	985.2612	4.087463	1.041412	16.70735	2.031206	69.48687	2228224	24154952	3.56E+14	4.5E+43	1.#INF00	48.3376	3.511697	2336.524	5.595074	1.72E+16	1.#INF00	1.#INF00
19	18	5832	1477.892	4.169925	1.061385	17.38828	2.060021	75.05865	4718592	65659968	6.4E+15	2.62E+47	1.#INF00	52.50753	3.594444	2757.041	5.714452	3.36E+17	1.#INF00	1.#INF00
20	19	6859	2216.838	4.247928	1.079918	18.04489	2.086759	80.71063	9961472	1.78E+08	1.22E+17	1.8E+51	1.#INF00	56.75546	3.672239	3221.182	5.826687	6.9E+18	1.#INF00	1.#INF00
21	20	8000	3325.257	4.321928	1.097189	18.67906	2.111675	86.43856	20971520	4.85E+08	2.43E+18	1.44E+55	1.#INF	Column: A	3.745629	3730.447	5.932566	1.49E+20	1.#INF00	1.#INF00



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
81	80	512000	1.22E+14	6.321928	1.477511	39.96677	2.660365	505.7542	9.67E+25	5.54E+34									
82	81	531441	1.83E+14	6.33985	1.480342	40.1937	2.664449	513.5278	1.96E+26	1.51E+35									
83	82	551368	2.75E+14	6.357552	1.48313	40.41847	2.668471	521.3193	3.97E+26	4.09E+35									
84	83	571787	4.13E+14	6.37504	1.485877	40.64113	2.672434	529.1283	8.03E+26	1.11E+36									
85	84	592704	6.19E+14	6.392317	1.488584	40.86172	2.676339	536.9547	1.62E+27	3.03E+36									
86	85	614125	9.28E+14	6.409391	1.491251	41.08029	2.680187	544.7982	3.29E+27	8.22E+36									
87	86	636056	1.39E+15	6.426265	1.493881	41.29688	2.68398	552.6588	6.65E+27	2.24E+37									
88	87	658503	2.09E+15	6.442944	1.496473	41.51152	2.68772	560.5361	1.35E+28	6.08E+37									
89	88	681472	3.13E+15	6.459432	1.499028	41.72426	2.691407	568.43	2.72E+28	1.65E+38									
90	89	704969	4.7E+15	6.475733	1.501549	41.93512	2.695044	576.3403	5.51E+28	1.#INF00									
91	90	729000	7.05E+15	6.491853	1.504035	42.14416	2.69863	584.2668	1.11E+29	1.#INF00									
92	91	753571	1.06E+16	6.507795	1.506488	42.35139	2.702169	592.2093	2.25E+29	1.#INF00									
93	92	778688	1.59E+16	6.523562	1.508908	42.55686	2.70566	600.1677	4.56E+29	1.#INF00									
94	93	804357	2.38E+16	6.539159	1.511296	42.7606	2.709105	608.1418	9.21E+29	1.#INF00									
95	94	830584	3.57E+16	6.554589	1.513652	42.96264	2.712505	616.1313	1.86E+30	1.#INF00									
96	95	857375	5.35E+16	6.569856	1.515979	43.163	2.715862	624.1363	3.76E+30	1.#INF00									
97	96	884736	8.03E+16	6.584962	1.518276	43.36173	2.719175	632.1564	7.61E+30	1.#INF00									
98	97	912673	1.2E+17	6.599913	1.520544	43.55885	2.722447	640.1915	1.54E+31	1.#INF00									
99	98	941192	1.81E+17	6.61471	1.522783	43.75439	2.725678	648.2416	3.11E+31	1.#INF00									
100	99	970299	2.71E+17	6.629356	1.524995	43.94837	2.728869	656.3063	6.27E+31	1.#INF00									
101	100	1000000	4.07E+17	6.643856	1.52718	44.14082	2.732021	664.3856	1.27E+32	1.#INF00									



CONCLUSION:	Through this experiment, I gained a comprehensive understanding of utilizing logarithmic and exponential functions in C programming language and the implementation of recursive functions, enhancing my programming skills and knowledge.