|  |  |
| --- | --- |
| **NAME:** | Akshaya Karande |
| **UID:** | 2021300056 |
| **SUBJECT** | Design and Analysis of Algorithms |
| **EXPERIMENT NO :** | 1 (A) |
| **AIM:** | To implement the various functions e.g. linear, non-linear, quadratic, exponential etc. |
| **ALGORITHM:** | **Function n:**  i. Initialize a variable n.  ii. Take the value of n from 0-100 and print all of them.  **Function n^3:**  i. Initialize variables n and result.  ii. result = n\*n\*n  iii. Apply a for loop for values of n from 0-100 and print all  the values for result.  **Function 3n/2:**  i. Initialize variables n and result.  ii. result = 3n/2  iii. Apply a for loop for values of n from 0-100 and print all  the values for result.  **Function logn:**  i. Initialize variables n and result.  ii. result = log10(n)  iii. Apply a for loop for values of n from 0-100 and print all  the values for result.  **Function ln n:**  i. Initialize variables n and result.  ii. result = ln(n)  iii. Apply a for loop for values of n from 0-100 and print all  the values for result.  **Function 2^n:**  i. Initialize variables n and result.  ii. result = pow(2,n)  iii. Apply a for loop for values of n from 0-100 and print all  the values for result.  **Function e^n:**  i. Initialize variables n and result.  ii. result = pow(e,n)  iii. Apply a for loop for values of n from 0-100 and print all  the values for result.  **Function 2^log n:**  i. Initialize variables n and result.  ii. result = pow(2,log10(n))  iii. Apply a for loop for values of n from 0-100 and print all  the values for result.  **Function n\*2^n:**  i. Initialize variables n and result.  v. result = n\*pow(2,n)  vi. Apply a for loop for values of n from 0-100 and print all  the values for result.  **Function sqrt(log n):**  i. Initialize variables n and result.  ii. result = sqrt(log10(n))  iii. Apply a for loop for values of n from 0-100 and print all  the values for result. |
| **PROGRAM:** | #include<stdio.h>  #include<math.h>  double fact(double n);  int main()  {  printf("\tn\tn^3\t2^n\tln n\tlg n\tn lg n\te^n\t(3/2)^n\tn.2^n\tn!");  for(double i=0;i<=100;i++)  {  printf("\n%.0lf",i);  printf("\t%.2lf",i);  printf("\t%.2lf",pow(i,3));  printf("\t%.2lf",pow(2,i));  printf("\t%.2lf",log(i));  printf("\t%.2lf",log2(i));  printf("\t%.2lf",i\*log2(i));  // printf("\t%.2lf",pow(2,pow(2,i)));  printf("\t%.2lf",exp(i));  printf("\t%.2lf",pow(1.5,i));  printf("\t%.2lf",i\*pow(2,i));  // printf("\n");  printf("\t%.2lf",fact(i));  }  return !4;  }  double fact(double n)  {  if(n==0)  return 1;  else  return n\*fact(n-1);  } |
| Observation: | |
| **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, and understanding the pattern of various functions and observing their behavior across a wide range of values. | |