**Day 01**

#Fibbonacci using recursion def fibbo(n): if(n==0): return 0 elif (n==1 or n==2): return 1 else: return(fibbo(n-1)+fibbo(n-2))n=int(input("enter no. of terms:"))for i in range(n): print(fibbo(i))2.#Amstrong numbern = int(input("Enter a number: "))s = 0t = nwhile t > 0: digit = t % 10 s += digit \*\* 3 t //= 10if n == s: print(n,"is an Armstrong number")else: print(n,"is not an Armstrong number"3.#2.Write a program to check the given no is Armstrong or not using recursive function.def armstrong(n,n1,s,t): if t==0: if s==n: return 1 else: return 0 else: d=t%10 s=s+d\*\*n1 t=t//10 return armstrong(n,n1,s,t)n=int(input("enter number"))s=0n1=len(str(n))t=nans=armstrong(n,n1,s,t)print(ans)if ans==1: print(n,"is an Armstrong number")else: print(n,"is not an Armstrong number")4.def find\_largest\_element(arr): # Initialize the first element as the largest largest = arr[0] # Iterate through the array to find the largest element for element in arr: if element > largest: largest = element return largest# Example usagearray = [10, 24, 45, 90, 72, 56, 32]largest\_element = find\_largest\_element(array)print(f"The largest element in the array is: {largest\_element}")5.def gcd(a, b): # Base case if b == 0: return a # Recursive case else: return gcd(b, a % b)# Example usagenum1 = 48num2 = 18print(f"The GCD of {num1} and {num2} is {gcd(num1, num2)}")6.#include<stdio.h>int revNum=0, rem;int reverse\_function(int num){ if(num) { rem=num%10; revNum=revNum\*10+rem; reverse\_function(num/10); } else return revNum; return revNum;}int main(){ int num, reverse\_number; printf("Enter any number:"); scanf("%d",&num); reverse\_number=reverse\_function(num); printf("The reverse of entered number is :%d",reverse\_number); return 0;}7. reverse a number using recursion #include <stdio.h>int main(){ int number, rem, sum = 0, i; printf("Enter a Number: "); scanf("%d", &number); for (i = 1; i <= (number - 1); i++) { rem = number % i; if (rem == 0) { sum = sum + i; } } if (sum == number) printf("%d is perfect number", number); else printf("%d is not a perfect number", number); return 0;}8.perfect number#include <stdio.h>// Recursive function to calculate the nth Fibonacci numberint fibonacci(int n) { if (n <= 1) { return n; } return fibonacci(n - 1) + fibonacci(n - 2);}int main() { int n = 10; // Example: Calculate the 10th Fibonacci number int fib = fibonacci(n); printf("The %dth Fibonacci number is: %d\n", n, fib); return 0;}9.recursive algorithm#include <stdio.h>// Recursive function to calculate the nth Fibonacci numberint fibonacci(int n) { if (n <= 1) { return n; } return fibonacci(n - 1) + fibonacci(n - 2);}int main() { int n = 10; // Example: Calculate the 10th Fibonacci number int fib = fibonacci(n); printf("The %dth Fibonacci number is: %d\n", n, fib); return 0;}