**Day – 1 . CSA0676- Design analysis of algorithms k.Edukondalu (192365010)**

1. **Write a program to Print Fibonacci Series using recursion.**

def recur\_factorial(n):

if n == 1:

return n

else:

return n\*recur\_factorial(n-1)

num = 7

# check if the number is negative

if num < 0:

print("Sorry, factorial does not exist for negative numbers")

elif num == 0:

print("The factorial of 0 is 1")

else:

print("The factorial of", num, "is", recur\_factorial(num))

1. **Write a program to check the given no is Armstrong or not using recursive function.**

**import math as m**

# recursive method to calculate armstrong number to power

def getSum(num, num\_length):

if num == 0:

return num

else:

return m.pow((num % 10), num\_length) + getSum(num//10, num\_length)

# get input from user

num = int(input("Enter a number: "))

# get input number length

num\_length = len(str(num))

sum = getSum(num, num\_length)

# display output

if sum == int(num):

print(num, "is an Armstrong Number.")

else:

print(num, "is not an Armstrong Number.")

1. **Write a program to find the GCD of two numbers using recursive factorization**

def gcd(a,b):

if(b==0):

return a

else:

return gcd(b,a%b)

a=int(input("Enter first number:"))

b=int(input("Enter second number:"))

GCD=gcd(a,b)

print("GCD is: ")

print(GCD)

1. **Write a program to get the largest element of an array.**

def largest(arr, n):

# Initialize maximum element

max = arr[0]

# Traverse array elements from second

# and compare every element with

# current max

for i in range(1, n):

if arr[i] > max:

max = arr[i]

return max

# Driver Code

arr = [10, 324, 45, 90, 9808]

n = len(arr)

Ans = largest(arr, n)

print("Largest in given array ", Ans)

1. **Write a program to find the Factorial of a number using recursion.**

def recur\_factorial(n):

if n == 1:

return n

else:

return n\*recur\_factorial(n-1)

num = 7

# check if the number is negative

if num < 0:

print("Sorry, factorial does not exist for negative numbers")

elif num == 0:

print("The factorial of 0 is 1")

else:

print("The factorial of", num, "is", recur\_factorial(num))

1. **Write a program for to copy one string to another using recursion.**

def myCopy(s1,s2):

# traversing the string s1 from start to end

for i in range(len(s1)):

# copying value one by one

s2[i]=s1[i]

return "".join(s2)

#Driver code

s1=list("KINGSFORVIKINGS")

s2=[""]\*len(s1)

print(myCopy(s1,s2))

'''Code is contributed by RAJAT KUMAR (rajatkumargla19)'''

1. **Write a program to print the reverse of a string using recursion**

def reverse(s):

if len(s) == 0:

return s

else:

return reverse(s[1:]) + s[0]

s = "Geeksforgeeks"

print("The original string is : ", end="")

print(s)

print("The reversed string(using recursion) is : ", end="")

print(reverse(s))

1. **Write a program to generate all the prime numbers using recursion**

def is\_prime(n, i=2):

if n <= 2:

return True if n == 2 else False

if n % i == 0:

return False

if i \* i > n:

return True

return is\_prime(n, i + 1)

def generate\_primes(n):

if n > 1:

generate\_primes(n - 1)

if is\_prime(n):

print(n)

# Generate prime numbers up to a specific number

generate\_primes(20)

1. **Write a program to check a number is a prime number or not using recursion.**

def is\_prime(num, i=2):

if num <= 2:

return num == 2

if num % i == 0:

return False

if i \* i > num:

return True

return is\_prime(num, i + 1)

num = int(input("Enter a number: "))

if is\_prime(num):

print(num, "is a prime number")

else:

print(num, "is not a prime number")

1. **.Write a program for to check whether a given String is Palindrome or not using recursion.**

num=127

temp=num

rev=0

while num>0:

rem=num%10

rev=rev\*10+rem

num=num//10

if temp==rev:

print("palindrome")

else:

print("not palindrome")