LAB-1
Date:4/6/24

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

CODE:

```
def recur_fibo(n):

if n <= 1:

return n

else:

return(recur_fibo(n-1) + recur_fibo(n-2))

nterms = 10

# check if the number of terms is valid

if nterms <= 0:

print("Plese enter a positive integer")

else:

print("Fibonacci sequence:")

for i in range(nterms):

print(recur_fibo(i))
```

OUTPUT:

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

CODE:

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

sum = 0

n1 = len(str(num))
```

```
temp = num
while temp > 0:
digit = temp % 10
sum += digit ** n1
temp //= 10
if num == sum:
print(num,"is an Armstrong number")
else:
print(num,"is not an Armstrong number")
```

OUTPUT:

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

CODE:

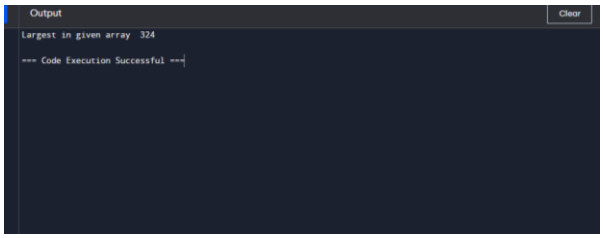
```
def gcd(a, b):
if a == b:
return a
elif a < b:
return gcd(b, a)
else:
return gcd(b, a - b)

a = 25
b = 45
print(gcd(a, b))
```

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

CODE:

```
# in arr[] of size n
# in arr[] of size n



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)
```



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

CODE:

```
def factorial(n):

# Checking the number
# is 1 or 0 then
# return 1
# other wise return
# factorial
if (n==1 or n==0):

return 1

else:

return (n * factorial(n - 1))

# Driver Code
num = 5;
print("number : ",num)
print("Factorial : ",factorial(num))
```

```
number :  5
Factorial :  120

=== Code Execution Successful ===
```

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

CODE:

```
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("GEEKShi")
s2=[""]*len(s1)
print(myCopy(s1,s2))
```

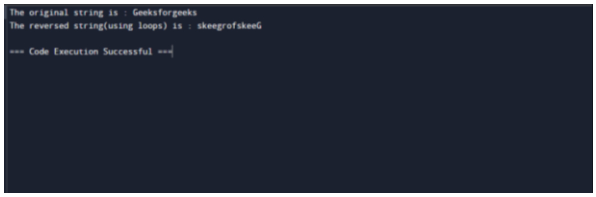
```
GEEKhi

=== Code Execution Successful ===
```

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

CODE:

```
def reverse(s):
str = ""
for i in s:
str = i + str
return str

s = "Geeksforgeeks"

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

```
print(s)

print("The reversed string(using loops) is : ", end="")
print(reverse(s))
```



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

CODE:

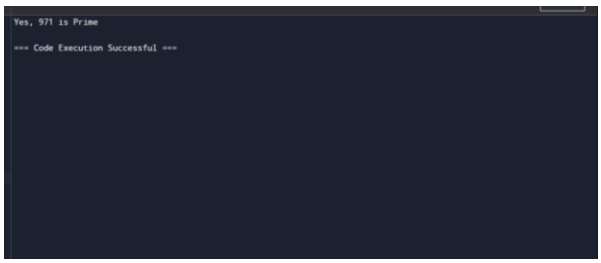
```
def is_prime(number, divisor=None):
# If divisor is not set, initialize it to number - 1
if divisor is None:
divisor = number - 1

# Base case: if divisor is 1, then number is prime
if divisor == 1:
return True
# If number is divisible by divisor, it's not prime
if number % divisor == 0:
return False
# Recurse with the next smaller divisor
return is_prime(number, divisor-1)

# Number to be checked
num_to_check = 29
# Check if the number is prime and print the result
if is_prime(num_to_check):
print(f"{num_to_check} is a prime number.")
else:
```

print(f"{num_to_check} is not a prime number.")

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

CODE:

```
def Prime_Number(n, i=2):
if n == i:
return True
elif n % i == 0:
return False
return Prime_Number(n, i + 1)


n = 971
if Prime_Number(n):
print("Yes,", n, "is Prime")
else:
print("No,", n, "is not a Prime")
```



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

CODE:

```
def is_palindrome(s):
```

```
if len(s) < 1:
return True
else:
if s[0] == s[-1]:
return is_palindrome(s[1:-1])
else:
return False
a=str(input("Enter string:"))
if(is_palindrome(a)==True):
print("String is a palindrome!")
else:
print("String isn't a palindrome!")
```