NAME - AMEY HUSHAR PATKAR

ASSIGNMENT NO 09

Question 1

Given an integer `n`, return `true` if it is a power of two. Otherwise, return `false`.

An integer `n` is a power of two, if there exists an integer `x` such that `n == 2x`.

Example 1:

Input: n = 1

Output: true

Example 2:

Input: n = 16

Output: true

Example 3:

Input: n = 3

Output: false

ANS –

To determine if an integer is a power of two, we can use the following approach:

Check if the given number n is positive and greater than zero. If n is zero or negative, it cannot be a power of two, so we return false.

Perform a bitwise AND operation between n and n-1. If the result is zero, it means n is a power of two.

Return true if the result is zero; otherwise, return false.

Here's the implementation in Python:

def isPowerOfTwo(n):

if n <= 0:

return False

return n & (n - 1) == 0

Let's test the function with the provided examples:

print(isPowerOfTwo(1)) # Output: True

print(isPowerOfTwo(16)) # Output: True

print(isPowerOfTwo(3)) # Output: False

The function correctly identifies whether the given integers are powers of two or not.

Question 2

Given a number n, find the sum of the first natural numbers.

Example 1:

Input: n = 3

Output: 6

Example 2:

Input : 5

Output : 15

ANS –

To find the sum of the first n natural numbers, we can use the formula for the sum of an arithmetic series: sum = (n \* (n + 1)) / 2.

Here's the implementation in Python:

def sumOfNaturalNumbers(n):

return (n \* (n + 1)) // 2

Let's test the function with the provided examples:

print(sumOfNaturalNumbers(3)) # Output: 6

print(sumOfNaturalNumbers(5)) # Output: 15

The function correctly calculates the sum of the first n natural numbers.

Question 3

Given a positive integer, N. Find the factorial of N.

Example 1:

Input: N = 5

Output: 120

Example 2:

Input: N = 4

Output: 24

ANS –

To find the factorial of a positive integer N, we can use a simple iterative approach where we multiply all the numbers from 1 to N together.

Here's the implementation in Python:

def factorial(N):

result = 1

for i in range(1, N + 1):

result \*= i

return result

Let's test the function with the provided examples:

print(factorial(5)) # Output: 120

print(factorial(4)) # Output: 24

The function correctly calculates the factorial of the given positive integers.

Question 4

Given a number N and a power P, the task is to find the exponent of this number raised to the given power, i.e. N^P.

Example 1 :

Input: N = 5, P = 2

Output: 25

Example 2 :

Input: N = 2, P = 5

Output: 32

ANS –

To find the exponent of a number N raised to a power P, you can use the power operator (\*\*) in Python. Here's the implementation:

def exponent(N, P):

return N \*\* P

Let's test the function with the provided examples:

print(exponent(5, 2)) # Output: 25

print(exponent(2, 5)) # Output: 32

The function correctly calculates the exponent of the given number N raised to the power P.

Question 5

Given an array of integers arr, the task is to find maximum element of that array using recursion.

Example 1:

Input: arr = {1, 4, 3, -5, -4, 8, 6};

Output: 8

Example 2:

Input: arr = {1, 4, 45, 6, 10, -8};

Output: 45

ANS –

To find the maximum element in an array arr using recursion, we can use a divide-and-conquer approach.

Here's the recursive implementation in Python:

def findMax(arr, start, end):

if start == end:

return arr[start]

mid = (start + end) // 2

left\_max = findMax(arr, start, mid)

right\_max = findMax(arr, mid + 1, end)

return max(left\_max, right\_max)

def getMaxElement(arr):

return findMax(arr, 0, len(arr) - 1)

Let's test the function with the provided examples:

arr1 = [1, 4, 3, -5, -4, 8, 6]

print(getMaxElement(arr1)) # Output: 8

arr2 = [1, 4, 45, 6, 10, -8]

print(getMaxElement(arr2)) # Output: 45

The function correctly finds the maximum element in the given array using recursion.

Question 6

Given first term (a), common difference (d) and a integer N of the Arithmetic Progression series, the task is to find Nth term of the series.

Example 1:

Input : a = 2 d = 1 N = 5

Output : 6

The 5th term of the series is : 6

Example 2:

Input : a = 5 d = 2 N = 10

Output : 23

The 10th term of the series is : 23

ANS –

To find the nth term of an arithmetic progression series, we can use the formula:

nth term = a + (n - 1) \* d

where:

a is the first term of the series

d is the common difference between consecutive terms

n is the position of the term we want to find

Here's the implementation in Python:

def findNthTerm(a, d, N):

nthTerm = a + (N - 1) \* d

return nthTerm

Let's test the function with the provided examples:

print(findNthTerm(2, 1, 5)) # Output: 6

print(findNthTerm(5, 2, 10)) # Output: 23

The function correctly calculates the nth term of the arithmetic progression series.

Question 7

Given a string S, the task is to write a program to print all permutations of a given string.

Example 1:

Input:

S = “ABC”

Output:

“ABC”, “ACB”, “BAC”, “BCA”, “CBA”, “CAB”

Example 2:

Input:

S = “XY”

Output:

“XY”, “YX”

ANS –

To generate all permutations of a given string, we can use a recursive approach.

Here's a recursive function to generate permutations in Python:

def generatePermutationsHelper(s, l, r, result):

if l == r:

result.append("".join(s))

else:

for i in range(l, r + 1):

s[l], s[i] = s[i], s[l]

generatePermutationsHelper(s, l + 1, r, result)

s[l], s[i] = s[i], s[l]

def generatePermutations(S):

result = []

s = list(S)

n = len(s)

generatePermutationsHelper(s, 0, n - 1, result)

return result

Let's test the function with the provided examples:

S1 = "ABC"

print(generatePermutations(S1))

# Output: ['ABC', 'ACB', 'BAC', 'BCA', 'CBA', 'CAB']

S2 = "XY"

print(generatePermutations(S2))

# Output: ['XY', 'YX']

The function correctly generates all permutations of the given string S.

Question 8

Given an array, find a product of all array elements.

Example 1:

Input : arr[] = {1, 2, 3, 4, 5}

Output : 120

Example 2:

Input : arr[] = {1, 6, 3}

Output : 18

ANS –

To find the product of all elements in an array, we can multiply each element together in a loop.

Here's the implementation in Python:

def productOfArray(arr):

product = 1

for num in arr:

product \*= num

return product

Let's test the function with the provided examples:

arr1 = [1, 2, 3, 4, 5]

print(productOfArray(arr1)) # Output: 120

arr2 = [1, 6, 3]

print(productOfArray(arr2)) # Output: 18

The function correctly calculates the product of all elements in the given array.