Answer

Step 1

Write a program to enter a natural number n and find all sphenic numbers from 1 to n using Recursion (combined with iteration if necessary).
Note: A sphenic number is a product of p*q*r where p, q, and r are three distinct prime numbers.
Example: 30 = 2 * 3* 5; 42 = 2*3*7; 66 = 2*3*11.
30, 42, 66, 70, 78, 102, 105, 110, 114, 130, 138, 154, 165, … are sphenic numbers.

Step 2

```
#include <iostream>

#include <cstring>

using namespace std;

bool primes[1001]; // array will store all the prime numbers from 0 to 1000. Primes will be marked as true

// seive of eratosthenes to find all primes les sthan 1000

void primeSieve()

{

    // initialize all entries of 'primes' as true

    memset(primes, true, sizeof(primes));

    // traverse all numbers and mark their multiples as false

    // do not traverse false numbers

    for (int p = 2; p * p < 1001; p++)

    {

        if (primes[p])

        {

            for (int i = p * 2; i < 1001; i = i + p)
```

```
                primes[i] = false;
            }
        }
    }
// function to return true if number is sphenic
bool isSphenic(int number)
{
    int arr1[8] = {0}; // array to store 8 divisors
    int count = 0;     // track the number of divisors
    int j = 0;
    for (int i = 1; i <= number; i++)
    {
        if (number % i == 0 && count < 9)
        {
            count++;
            arr1[j++] = i;
        }
    }
    // if there are 8 divisors and the divisors are primes, the number is sphenic
    if (count == 8 && (primes[arr1[1]] && primes[arr1[2]] && primes[arr1[3]]))
        return true;
    return false;
}
// main function
```

```
int main()
{
    int n;
    cout << "Enter n : ";
    cin >> n;
    // generate all prime numbers
    primeSieve();
    cout << "The sphenic numbers between 1 and " << n << " are:\n";
    for (int i = 1; i <= n; i++)
    {
        if(isSphenic(i))
            cout << i << "\n";
    }

}
```