



Problem Reduction in Transform and Conquer Technique



ojasvigupta

Read

Discuss

Courses

Practice

Video

What is Problem Reduction?

Problem reduction is an algorithm design technique that takes a complex problem and reduces it to a simpler one. The simpler problem is then solved and the solution of the simpler problem is then transformed to the solution of the original problem.

Problem reduction is a powerful technique that can be used to simplify complex problems and make them easier to solve. It can also be used to reduce the time and space complexity of algorithms.

Example:

Let's understand the technique with the help of the following problem:

Calculate the LCM (Least Common Multiple) of two numbers X and Y .

Approach 1:

To solve the problem one can iterate through the multiples of the bigger element (say **X**) until that is also a multiple of the other element. This can be written as follows:

- Select the bigger element (say **X** here).
- Iterate through the multiples of X:
 - If this is also a multiple of Y, return this as the answer.
 - Otherwise, continue the traversal.

Algorithm:

```
Algorithm LCM(X, Y):  
    if Y > X:  
        swap X and Y  
    end if  
    for i = 1 to Y:  
        if X*i is divisible by Y  
            return X*i  
        end if  
    end for
```

C++

```
#include<bits/stdc++.h>  
using namespace std;  
  
// Function to find the LCM of two numbers  
int LCM(int x,int y) {  
    if (y > x) {  
        swap(x, y); // swapping the values of x and y using destructuring assignment  
    }  
  
    for (int i = 1; i <= y; i++) {  
        if ((x*i) % y == 0) {  
            return i*x;  
        }  
    }  
}
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

```
}

int main()
{
    int x=10, y=15;
    cout<<LCM(10, 15); // Output: 30
}
```

Python3

```
# code
def LCM(x, y):
    if y > x:
        x, y = y, x

    for i in range(1, y+1):
        if (x*i) % y == 0:
            return i*x

    return x*y
print(LCM(10, 15))
#Code is contributed by Siddharth Aher
```

Javascript

```
// Function to find the LCM of two numbers
function LCM(x, y) {
    if (y > x) {
        [x, y] = [y, x]; // swapping the values of x and y using destructuring assignm
    }

    for (let i = 1; i <= y; i++) {
        if ((x*i) % y === 0) {
            return i*x;
        }
    }

    return x*y;
}

console.log(LCM(10, 15)); // Output: 30
```

Output

30

Time Complexity: O(Y) as the loop can iterate for maximum Y times [because X*Y is

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

Approach 2 (Problem Reduction): The above method required a linear amount of time and if the value of Y is very big it may not be a feasible solution. This problem can be reduced to another problem which is to “**calculate GCD of X and Y**” and the solution of that can be transformed to get the answer to the given problem as shown below:

- Calculate the [GCD of X and Y using Euclid’s algorithm](#).
- Now we know that $GCD * LCM = X * Y$. So the LCM can be calculated as $(X * Y / GCD)$.

Algorithm:

```
GCD (X, Y):
    if X = 0:
        return Y
    end if
    return GCD(Y%X, X)
```

```
Algorithm LCM(X, Y):
    G = GCD (X, Y)
    LCM = X * Y / G
```

Python3

```
def gcd(x, y):
    if x == 0:
        return y
    return gcd(y % x, x)

def lcm(x, y):
    g = gcd(x, y)
    lcm = (x*y)//g
    return lcm

x = 10
y = 15
print(lcm(x, y))
#Code is contributed by Siddharth Aher
```

Javascript

```
function gcd(x, y) {
    if (x === 0) {
        return y;
    }
}
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

```
function lcm(x, y) {  
    const g = gcd(x, y);  
    const lcm = (x * y) / g;  
    return lcm;  
}
```

```
const x = 10;  
const y = 15;  
console.log(lcm(x, y));
```

Output

30

Time Complexity: $O(\log(\min(X, Y)))$

Auxiliary Space: $O(1)$

Must Remember points about Problem Reduction:

- Reducing a problem to another one is only practical when the total time taken for transforming and solving the reduced problem is lower than solving the original problem.
- If problem A is reduced to problem B, then the lower bound of B can be higher than the lower bound of A, but it can never be lower than the lower bound of A.

Related Articles:

- [Transform and Conquer Technique](#)
- [Instance Simplification method in Transform and Conquer Technique](#)
- [Representation Change in Transform and Conquer Technique](#)

Last Updated : 26 Apr, 2023

2

Similar Reads

1. Representation Change in Transform and Conquer Technique

2. Instance Simplification Method in Transform and Conquer Technique

3. Transform and Conquer Technique

4. Swiss Roll Reduction with LLE in Scikit Learn

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

6. Comparison among Greedy, Divide and Conquer and Dynamic Programming algorithm

7. Code Optimization Technique (logical AND and logical OR)

8. Merge K sorted arrays | Set 3 (Using Divide and Conquer Approach)

9. Merge K sorted arrays of different sizes | (Divide and Conquer Approach)

10. Difference between Backtracking and Branch-N-Bound technique

Related Tutorials

1. Learn Data Structures with Javascript | DSA Tutorial

2. Introduction to Max-Heap – Data Structure and Algorithm Tutorials

3. Introduction to Set – Data Structure and Algorithm Tutorials

4. Introduction to Map – Data Structure and Algorithm Tutorials

5. What is Dijkstra's Algorithm? | Introduction to Dijkstra's Shortest Path Algorithm

[Previous](#)[Next](#)

Article Contributed By :



ojasvigupta
ojasvigupta

Vote for difficulty

[Easy](#)[Normal](#)[Medium](#)[Hard](#)[Expert](#)

Improved By : [poojaagrawal2](#), [rkbhola5](#), [mallelagowtalm](#), [vutv1kcj](#), [ahersiddhba7n](#)

Article Tags : [Picked](#), [Technical Scriptor 2022](#), [Algorithms](#), [DSA](#), [Technical Scriptor](#)

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

[Improve Article](#)

[Report Issue](#)



A-143, 9th Floor, Sovereign Corporate Tower, Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

Company

[About Us](#)

[Careers](#)

[In Media](#)

[Contact Us](#)

[Terms and Conditions](#)

[Privacy Policy](#)

[Copyright Policy](#)

[Third-Party Copyright Notices](#)

[Advertise with us](#)

Explore

[Job Fair For Students](#)

[POTD: Revamped](#)

[Python Backend LIVE](#)

[Android App Development](#)

[DevOps LIVE](#)

[DSA in JavaScript](#)

Languages

[Python](#)

[Java](#)

[C++](#)

[GoLang](#)

[SQL](#)

[R Language](#)

[Android Tutorial](#)

Data Structures

[Array](#)

[String](#)

[Linked List](#)

[Stack](#)

[Queue](#)

[Tree](#)

[Graph](#)

Algorithms

[Sorting](#)

[Searching](#)

[Greedy](#)

Web Development

[HTML](#)

[CSS](#)

[JavaScript](#)

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

[Recursion](#)[AngularJS](#)[Backtracking](#)[NodeJS](#)

Data Science & ML

Interview Corner

[Data Science With Python](#)[Company Preparation](#)[Data Science For Beginner](#)[Preparation for SDE](#)[Machine Learning Tutorial](#)[Company Interview Corner](#)[Maths For Machine Learning](#)[Experienced Interview](#)[Pandas Tutorial](#)[Internship Interview](#)[NumPy Tutorial](#)[Competitive Programming](#)[NLP Tutorial](#)[Aptitude](#)

Python

GfG School

[Python Tutorial](#)[CBSE Notes for Class 8](#)[Python Programming Examples](#)[CBSE Notes for Class 9](#)[Django Tutorial](#)[CBSE Notes for Class 10](#)[Python Projects](#)[CBSE Notes for Class 11](#)[Python Tkinter](#)[CBSE Notes for Class 12](#)[OpenCV Python Tutorial](#)[English Grammar](#)

UPSC/SSC/BANKING

Write & Earn

[SSC CGL Syllabus](#)[Write an Article](#)[SBI PO Syllabus](#)[Improve an Article](#)[IBPS PO Syllabus](#)[Pick Topics to Write](#)[UPSC Ethics Notes](#)[Write Interview Experience](#)[UPSC Economics Notes](#)[Internships](#)[UPSC History Notes](#)[Video Internship](#)

@geeksforgeeks , Some rights reserved