**Rope Cutting Problem**

Given a rope of length**N** meters, and the rope can be cut in only 3 sizes **A**, **B** and **C**. The task is to maximizes the number of cuts in rope. If it is impossible to make cut then print the number else print **-1**. **Examples:**

**Input:**

N = 17, A = 10, B = 11, C = 3

**Output:**3

**Explanation:** The maximum cut can be obtain after making 2 cut of length 3 and one cut of length 11.

**Input:**N = 10, A = 9, B = 7, C = 11

**Output:**-1

**Explanation:**It is impossible to make any cut so output will be -1.

**Naive Approach: Using Recursion**

C++Java

/\*package whatever //do not write package name here \*/

import java.io.\*;

import java.util.\*;

class GFG {

static int maxCuts(int n, int a, int b, int c)

{

if(n == 0)

return 0;

if(n <= -1)

return -1;

int res = Math.max(maxCuts(n-a, a, b, c),

Math.max(maxCuts(n-b, a, b, c),

maxCuts(n-c, a, b, c)));

if(res == -1)

return -1;

return res + 1;

}

public static void main(String [] args)

{

int n = 5, a = 2, b = 1, c = 5;

System.out.println(maxCuts(n, a, b, c));

}

}

**Output:**

5

**Time Complexity** : O(3^n)

**Space Complexity** : O(n), due to recursive call stack.