**[Perfect Squares](https://leetcode.com/problems/perfect-squares/)**

Given an integer n, return *the least number of perfect square numbers that sum to* n.

A **perfect square** is an integer that is the square of an integer; in other words, it is the product of some integer with itself. For example, 1, 4, 9, and 16 are perfect squares while 3 and 11 are not.

**Example 1:**

**Input:** n = 12

**Output:** 3

**Explanation:** 12 = 4 + 4 + 4.

**Example 2:**

**Input:** n = 13

**Output:** 2

**Explanation:** 13 = 4 + 9.

**Constraints:**

* 1 <= n <= 104

Code :

#pragma GCC optimize("O3", "unroll-loops")

class Solution {

public:

    bool is\_square(int n){

        double sqrt\_n=sqrt(n);

        return sqrt\_n==int(sqrt\_n);

    }

    int numSquares(int n) {

        //Use Lagrange's four-square theorem and Legendre's three-square theorem

        while (n%4==0) n/=4;

        if (n%8==7) return 4;

        if (is\_square(n)) return 1;

        double&& sqrt\_n=sqrt(n);

        for(int i=1; i<=sqrt\_n; i++)

        {

            int&& y=n-i\*i;

            if (is\_square(y)) return 2;//2 squares;

        }

        return 3;

    }

};

auto init = []()

{

    ios::sync\_with\_stdio(0);

    cin.tie(0);

    cout.tie(0);

    return 'c';

}();

Link : - <https://leetcode.com/problems/perfect-squares/?envType=daily-question&envId=2024-02-08>