1. Perfect Squares

Given a positive integer *n*, find the least number of perfect square numbers (for example, 1, 4, 9, 16, ...) which sum to *n*.

**Example 1:**

Input: n = 12  
Output: 3   
Explanation: 12 = 4 + 4 + 4.

**Example 2:**

Input: n = 13  
Output: 2  
Explanation: 13 = 4 + 9.

**解法1** 暴力求解

class Solution {  
public:  
 int numSquares(int n) {  
 int \*s = new int[n+1];  
 for(int i = 0; i <= n; ++i)s[i] = i\*i;  
 return minSquares(n, s, n);  
   
 }  
 int minSquares(int k, int \*squares, int n){  
 for(int i = 1; i <= n; ++i){  
 if(squares[i] == k)return 1;  
 if(squares[i] > k)break;  
 }  
 int ans = INT\_MAX;  
 for(int i = 1; i <= n; ++i){  
 if(squares[i] > k)break;  
 ans = min(ans, minSquares(k - squares[i], squares, n)+1);  
 }  
 return ans;  
 }  
};

**解法2** 将暴力求解的中间结果存储起来，避免重复搜索

class Solution {  
public:  
 int numSquares(int n) {  
 int \*s = new int[n+1];  
 int \*dp = new int[n+1];  
 for(int i = 0; i <= n; ++i){  
 s[i] = i\*i;  
 dp[i] = INT\_MAX;  
 }  
 dp[0] = 0;  
 return minSquares(n, s, n, dp);  
   
 }  
 int minSquares(int k, int \*squares, int n, int \*dp){  
 if(dp[k] != INT\_MAX)return dp[k];  
 int ans = INT\_MAX;  
 for(int i = 1; i <= n; ++i){  
 if(squares[i] > k)break;  
 ans = min(ans, minSquares(k - squares[i], squares, n, dp)+1);  
 }  
 dp[k] = ans;  
 return ans;  
 }  
};

**解法3** 动态规划，可以认为是解法2的迭代版本

class Solution {  
public:  
 int numSquares(int n) {  
 int \*s = new int[n+1];  
 int \*dp = new int[n+1];  
 for(int i = 0; i <= n; ++i){  
 s[i] = i \* i;  
 dp[i] = INT\_MAX;  
 }  
 dp[0] = 0;  
 int max\_s\_index = sqrt(n) + 1;  
 for(int i = 1; i <= n; ++i){  
 for(int j = 1; j < max\_s\_index; ++j){  
 if(i < s[j])break;  
 dp[i] = min(dp[i], dp[i - s[j]]+1);  
 }  
 }  
 return dp[n];  
 }  
};