01背包： 494. 目标和

- **核心思想**：通过数学推导，将添加 + 和 - 号的问题转化为子集和问题。
- **动态规划**：使用二维数组 dp[i][j]，记录使用前 i 个元素，和为 j 的子集数量。
- **状态转移方程**：
- 不选择当前元素：dp[i][j] = dp[i-1][j]
- 选择当前元素：dp[i][j] += dp[i-1][j - nums[i-1]]
- **边界条件**：初始化 dp[0][0] = 1，并检查 abs(target) <= sum 和 (sum + target) % 2 == 0。

代码：

```cpp
class Solution {
public:
    int findTargetSumWays(vector<int>& nums, int target) {
        int count = 0;
        int sum = 0;
        for(auto s : nums)
        {
            sum += s;
        }
        // 检查 target 是否在有效范围内
        if(abs(target) > sum) return 0;

        // 检查 (sum + target) 是否为偶数
        if((sum + target) % 2 != 0) return 0;

        int m = nums.size();
        int n = (sum + target)/2;
        vector<vector<int>> dp(m+1,vector<int>(n+1));
        dp[0][0] = 1;
        for(int i = 1; i <=m ;i++)
        {
            for(int j = 0; j <= n;j++)
            {
                dp[i][j] = dp[i-1][j];
                if(j >= nums[i-1])
                dp[i][j] += dp[i-1][j-nums[i-1]];
            }
        }
        return dp[m][n];

    }
};
```

BFS： 127. 单词接龙

解答：

代码：

```cpp
int ladderLength(string beginWord, string endWord, vector<string>& wordList) {
    unordered_set<string> vis;
    unordered_set<string> hash(wordList.begin(), wordList.end());
    string change = "abcdefghijklmnopqrstuvwxyz";

    if (beginWord == endWord)
        return 0;
    if (!hash.count(endWord))
        return 0;
    queue<string> q;
    q.push(beginWord);
    vis.insert(beginWord);
    int ret = 0;
    while (q.size()) {
        ret++;
        int sz = q.size();
        while (sz--) {
            string t = q.front();
            q.pop();
            int size = t.size();
            for (int i = 0; i < size; i++) {
                string tmp = t;
                for (int j = 0; j < 26; j++) {
                    tmp[i] = change[j];
                    if (hash.count(tmp) && !vis.count(tmp)) {
                        if (tmp == endWord)
                            return ret + 1;
                        q.push(tmp);
                        vis.insert(tmp);
                    }
                }
            }
        }
    }
    return 0;
}
```