最优子结构详解

最优子结构：从子问题的最优结果推导出更大规模的问题解。

dp数组遍历方向

dp数组遍历方向有：

1. 正向

```
std::vector<std::vector<int>> dp= std::vector<std::vector<int>>(m+1,
std::vector<int>(n+1,0));
for(int i = 0; i < m;i++) {
    for(int j =0; j < n;j++) {
        // 计算dp[i][j]
    }
}
```

2. 反向

```
std::vector<std::vector<int>> dp= std::vector<std::vector<int>>(m+1,
std::vector<int>(n+1,0));
for(int i = m-1; i >= 0;i--) {
    for(int j = n-1; j >= 0;j--) {
        // 计算dp[i][j]
    }
}
```

3. 斜向

```
std::vector<std::vector<int>> dp= std::vector<std::vector<int>>(m+1,
std::vector<int>(n+1,1));
for(int i = 2; i <= n;i++) {
    for(int j =0; j <= n-1;j++) {
        int k = i + j - 1;
        // 计算dp[i][j]
    }
}
```

遍历的过程中需要注意的两点：

> 1. 遍历的过程中，所需的状态必须是已经计算出来的；
> 2. 遍历终点必须是存储结果的位置。