递推

信智学院　陈琰宏

教学内容

01 理解应用递思想

02 掌握用递推思想解题

1 递推算法

递推是计算机数值计算中的一个重要算法，通过数学推导，将复杂的运算化解为若干重复的简单运算，以充分发挥计算机善长于重复处理的特点。

递推算法以初始{起点}值为基础，用相同的运算规律，逐次重复运算，直至运算结束。这种从“起点”重复相同的方法直至到达一定“边界”，犹如单向运动，用循环可以实现。递推的本质是按规律逐次推出（计算）下一步的结果。

1.1 引例1

例：植树节那天，有五位参加了植树活动，他们完成植树的棵数都不相同。问第一位同学植了多少棵时，他指着旁边的第二位同学说比他多植了两棵；追问第二位同学，他又说比第三位同学多植了两棵；如此追问，都说比另一位同学多植两棵。最后问到第五位同学时，他说自己植了10棵。到底第一位同学植了多少棵树？

【分析】设第一位同学植树的棵数为a1，欲求a1，需从第五位同学植树的棵数a5入手，根据“多两棵”这个规律，按照一定顺序逐步进行推算：

1.1 引例1

①a5=10；

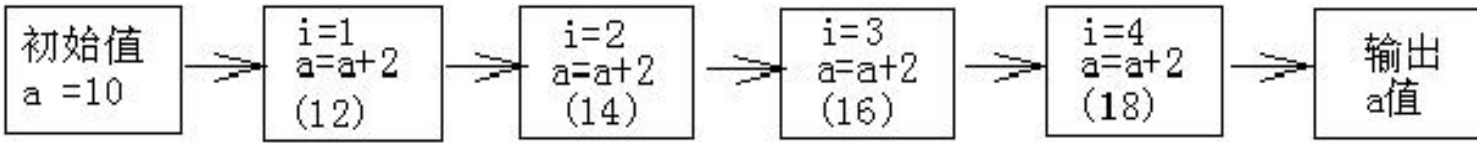
②a4=a5+2=12；

③a3=a4+2=14；

④a2=a3+2=16；

⑤a1=a2+2=18；

本程序的递推运算可用如下图示描述：



1.1 引例1

程序如下：

```
#include<iostream>
using namespace std;
int main()
{
   int a=10;    //以第五位同学的棵数为递推的起始值
   for (int i=4; i>=1; i--) //还有4人，递推计算4次
   a+=2;                    //递推运算规律
   cout<<" The Num is "<<a<<endl;
   return 0;
}
```

1.1 引例2

楼梯有n个台阶，上楼可以一步上一阶，也可以一步上两阶。一共有多少种上楼的方法？

这是一道计数问题。在没有思路时，n=5时，一共有8种方法：

5=1+1+1+1+1　　5=2+1+1+1

5=1+2+2　　5+2+2+1

5=1+2+1+1　　5=1+1+2+1

5=1+1+1+2　　5=2+1+2

其中有：

5种方法第1步走了1阶，

3种方法第1步走了2阶，没有其他可能。

假设f(n)为n个台阶的走法总数，**把n个台阶的走法分成两类**。

第1类：第1步走1阶，剩下还有n-1阶要走，有f(n-1)种方法。

第2类：第1步走2阶，剩下还有n-2阶要走，有f(n-2)种方法。

这样，就得到了递推式：f(n)=f(n-1)+f(n-2),不要忘记边界情况：f(1)=1,f(2)=2。把f(n)的前几项列出：1，2，3，5，8，……。

```
using namespace std;
int main(){
  int n;
  int a[41];
  a[1]=1;
  a[2]=2;
  while(cin>>n)
  {
    if(n==0)
      break;
    for(int i=3;i<=n;i++)
      a[i]=a[i-1]+a[i-2];
    cout<<a[n]<<endl;
  }
  return 0;
}
```

1.2 理解递推法

所谓递推，是指从已知的初始条件出发，依据某种递推关系，逐次推出所要求的各中间结果及最后结果。其中初始条件或是问题本身已经给定，或是通过对问题的分析与化简后确定。

可用递推算法求解的题目一般有以下二个特点：

1、问题可以划分成多个状态；

2、除初始状态外，其它各个状态都可以用固定的递推关系式来表示。

在我们实际解题中，题目不会直接给出递推关系式，而是需要通过分析各种状态，找出递推关系式。

1.2 解决递推问题的一般步骤

- **建立递推关系式**
- **确定边界条件**
- **递推求解**

递推的两种形式

- **顺推法和倒推法**

1.3.1 [2046] 杨辉三角形

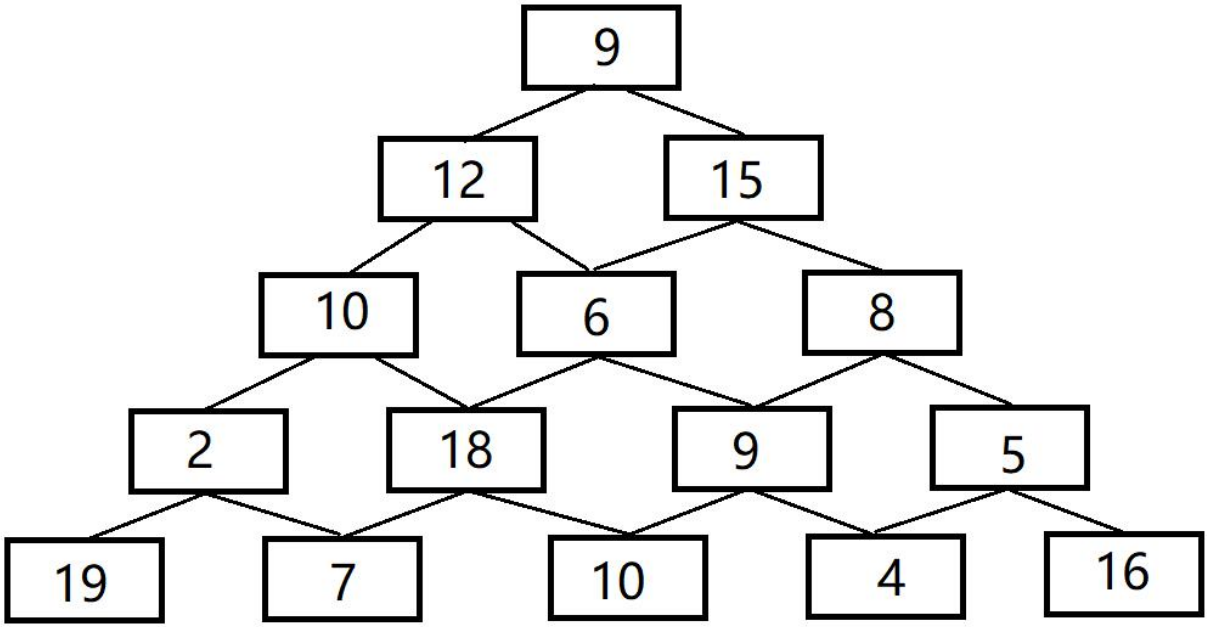
打印杨晖三角形的前10行。杨晖三角形的前5行如左下图所示。

问题分析：我们观察左上图不太容易找到规律，但如果将左上图转化为右上图就不难发现杨辉三角形其实就是一个二维表（数组）的下三角部分。

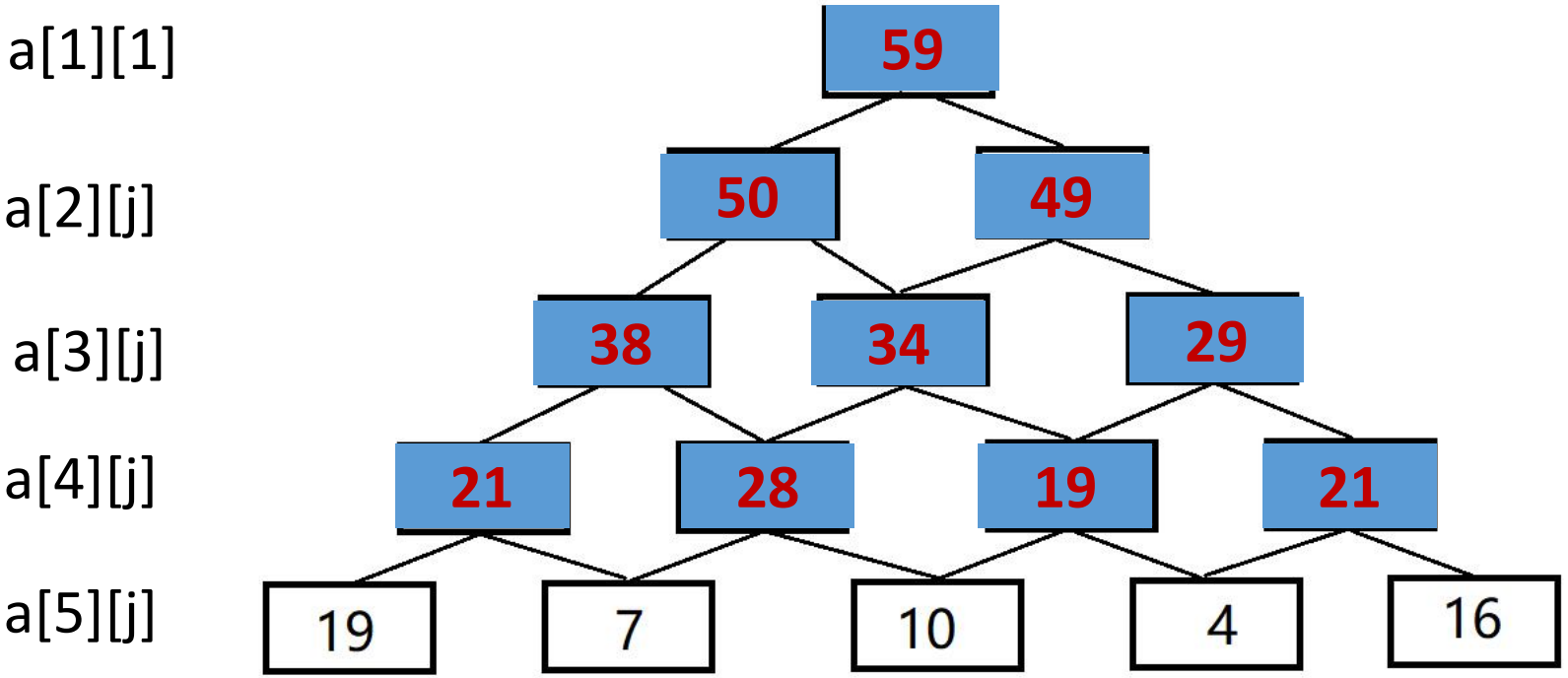
```cpp
#include<iostream>
using namespace std;
int main()
{
    int a[31][31],i,j,n;
    cin>>n;
    for(i=1;i<=n;i++) {
        a[i][i]=1;a[i][1]=1;
    }
    for(i=3;i<=n;i++)
        for(j=2;j<i;j++)
            a[i][j]=a[i-1][j-1]+a[i-1][j];
    for(i=1;i<=n;i++) {
        for(j=1;j<=i;j++)
            printf("%d ",a[i][j]);
        printf("\n");
    }
   return 0;
}
```

1.3.2 [2997] 数塔问题

有如下所示的数塔，要求从顶层走到底层，若每一步只能走到相邻的结点，则经过的结点的数字之和最大是多少？



1.3.2 [2997] 数塔问题

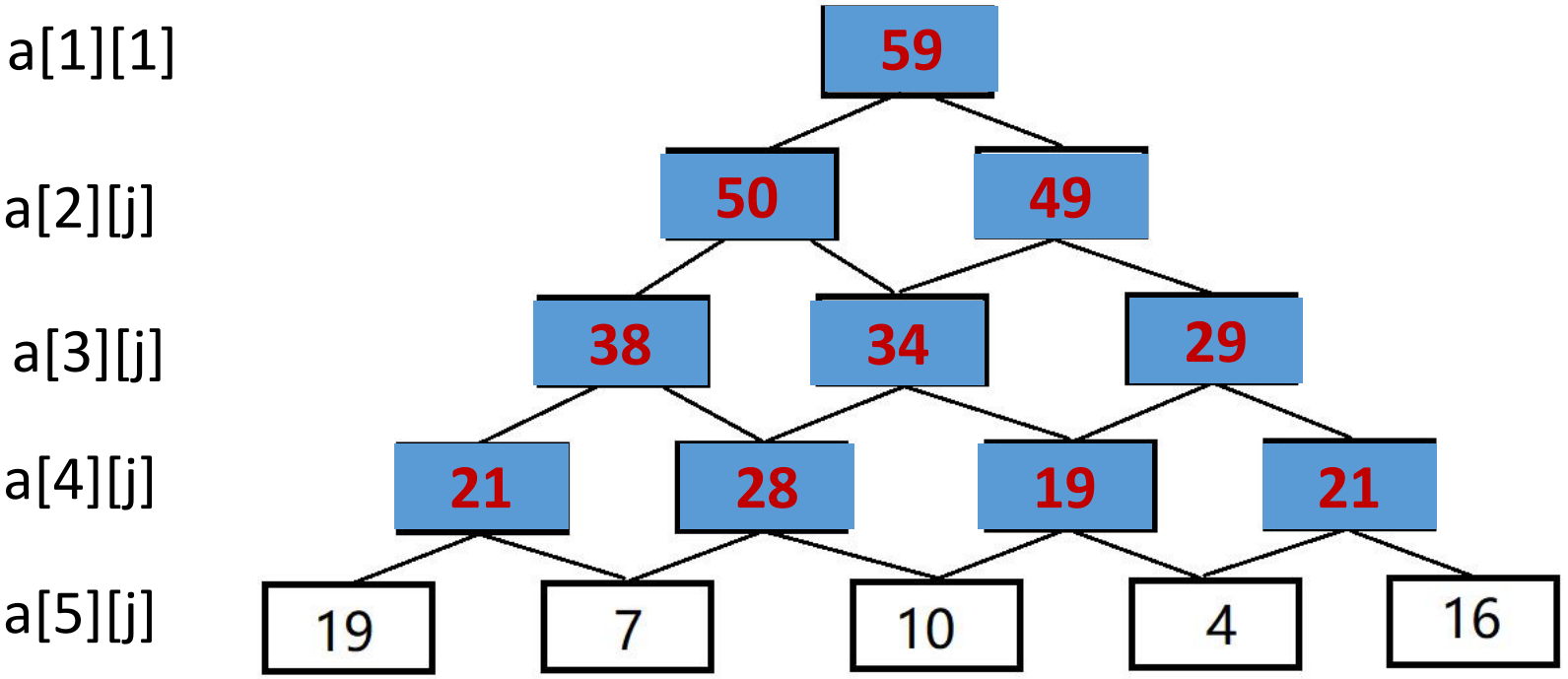


递推关系： f[i][j]=a[i][j]+max(f[i+1][j],f[i+1][j+1]);

求解： cout<<f[1][1];

边界： memset(a,0,sizeof(a));

1.3.2 [2997] 数塔问题



递推关系： f[i][j]=a[i][j]+max(f[i+1][j],f[i+1][j+1]);

求解： cout<<f[1][1];

边界： memset(a,0,sizeof(a));

1.3.2 [2997] 数塔问题

```
#include<iostream>
#include<cstring>
using namespace std;

int a[100][100],f[100][100];
int main(){
    int n,i,j;
    cin>>n;
    memset(f,0,sizeof(a));
    for(i=1;i<=n;i++)
        for(j=1;j<=i;j++)
            cin>>a[i][j];
    for(i=n;i>=1;i--)
        for(j=1;j<=i;j++)
            f[i][j]=a[i][j]+max(f[i+1][j],f[i+1][j+1]);
    cout<<"max="<<f[1][1];
    return 0;
}
```

```
1  #include<iostream>
2  #include<cstring>
3  using namespace std;
4  int main(){
5      int n,a[21][21],f[21][21];
6      cin>>n;
7      for(int i=1;i<=n;i++)
8          for(int j=_____1_____)
9              cin>>a[i][j];
10     memset(f,0,sizeof(a));
11     for(int i=_____2_____)
12         for(int j=____3_____)
13             f[i][j]=max(_____4_____);
14     cout<<"max="<<f[1][1]<<endl;
15     return 0;
16 }
```

1.3.2 [2998] 汉诺塔1

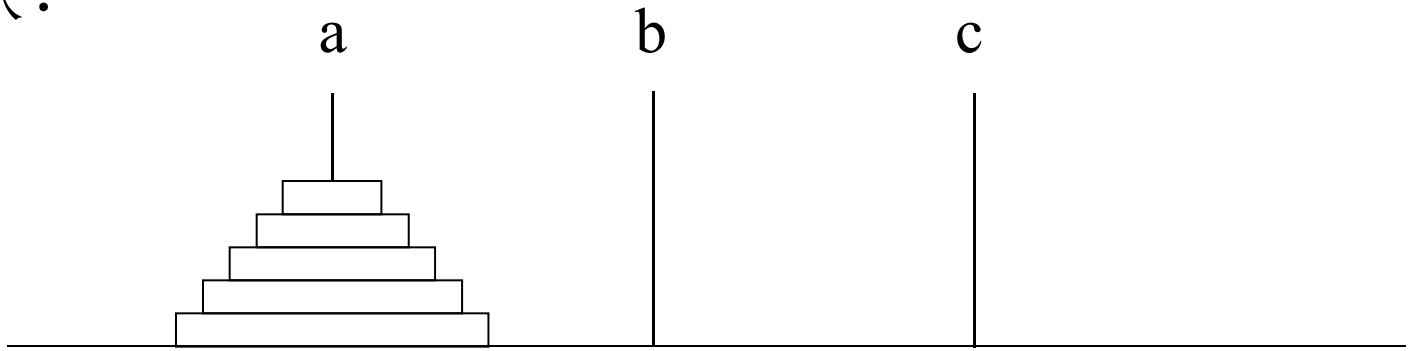
Hanoi塔由n个大小不同的圆盘和三根木柱a, b, c组成。开始时，这n个圆盘由大到小依次套在a柱上，如图1所示。要求把a柱上n个圆盘按下述规则移到c柱上：

(1) 一次只能移一个圆盘；

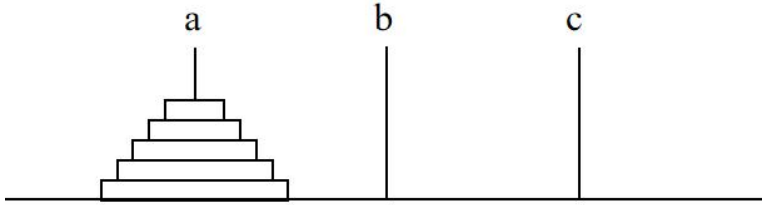
(2) 圆盘只能在三个柱上存放；

(3) 在移动过程中，不允许大盘压小盘。

问将这n个盘子从a柱移动到c柱上，总计需要移动多少个盘次？



分析



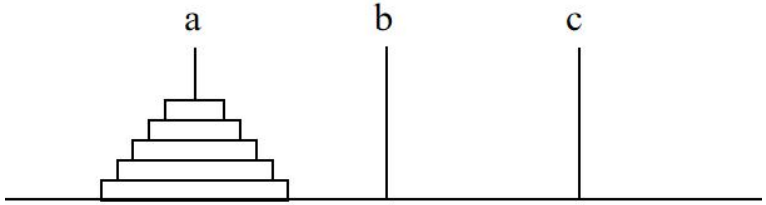
第一步：先借助c柱把a柱上面的n-1个盘子移动到b柱上，所需的移动次数为f(n-1)。

第二步：然后再把a柱最下面的一个盘子移动到c柱上，只需要1次盘子。

第三步：再借助a柱把b柱上的n-1个盘子移动到c上，所需的移动次数为f(n-1)。

由以上3步得出总共移动盘子的次数为：f(n-1)+1+ f(n-1)。
所以：**递推式　f(n)=2f(n-1)+1**

分析



确定边界条件：

当n=1时，f(1)=1。

当n=2时，f(2)=3。

所以

边界条件：$h_1=1$

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main()
4  {
5      int sum=1,n;
6      cin>>n;
7      for(int i=1;i<n;i++)
8          sum=sum*2+1;
9      cout<<sum;
10     return 0;
11 }
```

1.3.3 [2974]传球游戏

老师带着同学们一起做传球游戏。

游戏规则是这样的：n（3<=n<=30）个同学站成一个圆圈，其中的一个同学手里拿着一个球，当老师吹哨子时开始传球，每个同学可以把球传给自己左右的两个同学中的一个（左右任意），当老师再吹哨子时，传球停止，此时，拿着球没传出去的那个同学就是败者，要给大家表演一个节目。

聪明的小蛮提出一个有趣的问题：**有多少种不同的传球方法可以使得从小蛮手里开始传的球，传了m（3<=m<=30）次后，又回到小蛮手里。**两种传球被视作不同的方法，当且仅当这两种方法中，接到球的同学按照顺序组成的序列是不同的。比如3个同学1号、2号、3号，并假设小蛮为1号，球传了3次回到小蛮手里的方式有1->2->3->1和1->3->2->1，共两种。

分析-描述状态

设f[i][k]表示经过k次传到编号为i的人手中的方案数，传到i号同学的球只能来自于i的左边一个同学和右边一个同学，这两个同学的编号分别是i-1和i+1.

f[i-1][k-1]：经过k-1次传到左手边同学的方案数

f[i+1][k-1]：经过k-1次传到右手边同学的方案数

分析-描述状态

设f[i][k]表示经过k次传到编号为i的人手中的方案数，传到i号同学的球只能来自于i的左边一个同学和右边一个同学，这两个同学的编号分别是i-1和i+1.

f[i-1][k-1]：经过k-1次传到左手边同学的方案数

f[i+1][k-1]：经过k-1次传到右手边同学的方案数

分析-递推式

经过k次传到编号为i的人手中的方案数=

经过k-1次传到左手边同学的方案数+

经过k-1次传到右手边同学的方案数

f[i][k]=f[i-1][k-1]+f[i+1][k-1]

分析-边界

f[i][k]=f[i-1][k-1]+f[i+1][k-1]

观察递推式，需要考虑哪些边界条件？

1. 求什么？　　answer=f[1,m]

2. 边界？

f[1,k]=f[n,k-1]+f[2,k-1],　　当i=1时

f[n,k]=f[n-1,k-1]+f[1,k-1],　　当i=n时

边界条件：f[1,0]=1；

代码

```
#include<stdio.h>
int main()
{
    int n,m;
    scanf("%d %d",&n,&m);
    int f[31][31]={0};
   f[1][0]=1;
    for(int j=1;j<=m;j++)//m次传球，n个同学
    {
        for(int i=1;i<=n;i++)
        {
            if(i==1)f[i][j]=f[n][j-1]+f[2][j-1];
            else if(i==n)f[i][j]=f[n-1][j-1]+f[1][j-1];
            else f[i][j]=f[i-1][j-1]+f[i+1][j-1];
        }
    }
    printf("%d",f[1][m]);
    return 0;
}
```

1.3.4[2864]昆虫繁殖

科学家在热带森林中发现了一种特殊的昆虫，这种昆虫的繁殖能力很强。**每只成虫过x个月产y个卵，每个卵要过两个月长成成虫**。假设每个成虫不死，第一个月只有一只成虫，且卵长成成虫后的第一个月不产卵(过X个月产卵)，问过Z个月以后，共有成虫多少只？0≤X≤20,1≤Y≤20,X≤Z≤50。

输入：x,y,z的数值。

输出：过Z个月以后，共有成虫只数。

样例输入：1 2 8

样例输出：37

分析-递推式

当前成虫数=上一个月的成虫+能变成功的卵的数量

用 a[i]表示第i个月拥有的成虫数目，

b[i]表示第i个月产生的新增卵，显然

递归式为：

a[i]=a[i-1]+b[i-2]（**每个卵要过两个月长成成虫**）

分析-递推式

第i个月产生的新增卵怎么求？

1. 每只成虫过x个月产y个卵
2. 每个卵要过两个月长成成虫。

b[i]为x月前的成虫产的卵;x月前的成虫数为a[i-x],每只成虫产y个卵，可以得到：

b[i]=y*a[i-x]

分析-边界

第一个月只有一只成虫

a[1]=1,b[1]=0

第一个月到第x个月都没有产卵

a[i]=1,b[i]=0

```cpp
#include<iostream>
using namespace std;
int main()
{
    int x,y,z;
    int a[505],b[505];
    cin>>x>>y>>z;
    for(int i=1;i<=x;i++)
    {
        a[i]=1; //成虫数量
        b[i]=0; //卵的数量
    }
    for(int i=x+1;i<=z+1;i++)
    {
        b[i]=y*a[i-x];
    //卵的数量等于i-x个月时成虫的数量产的卵
        a[i]=a[i-1]+b[i-2];
//成虫的数量等于i-1个月的成虫数量加上i-2个月的卵的数量
    }
    cout<<a[z+1];
    return 0;
}
```

1.3.5 [2757]移动路线

X桌子上有一个m行n列的方格矩阵，将每个方格用坐标表示，行坐标从下到上依次递增，列坐标从左至右依次递增，左下角方格的坐标为(1,1)，则右上角方格的坐标为(m,n)。

小明是个调皮的孩子，一天他捉来一只蚂蚁，不小心把蚂蚁的右脚弄伤了，于是蚂蚁只能向上或向右移动。小明把这只蚂蚁放在左下角的方格中，蚂蚁从

左下角的方格中移动到右上角的方格中，每步移动一个方格。蚂蚁始终在方格矩阵内移动，请计算出不同的移动路线的数目。

对于1行1列的方格矩阵，蚂蚁原地移动，移动路线数为1；对于1行2列（或2行1列）的方格矩阵，蚂蚁只需一次向右（或向上）移动，移动路线数也为1.....对于一个2行3列的方格矩阵，如下图所示：

(2,1)　(2,2)　(2,3)

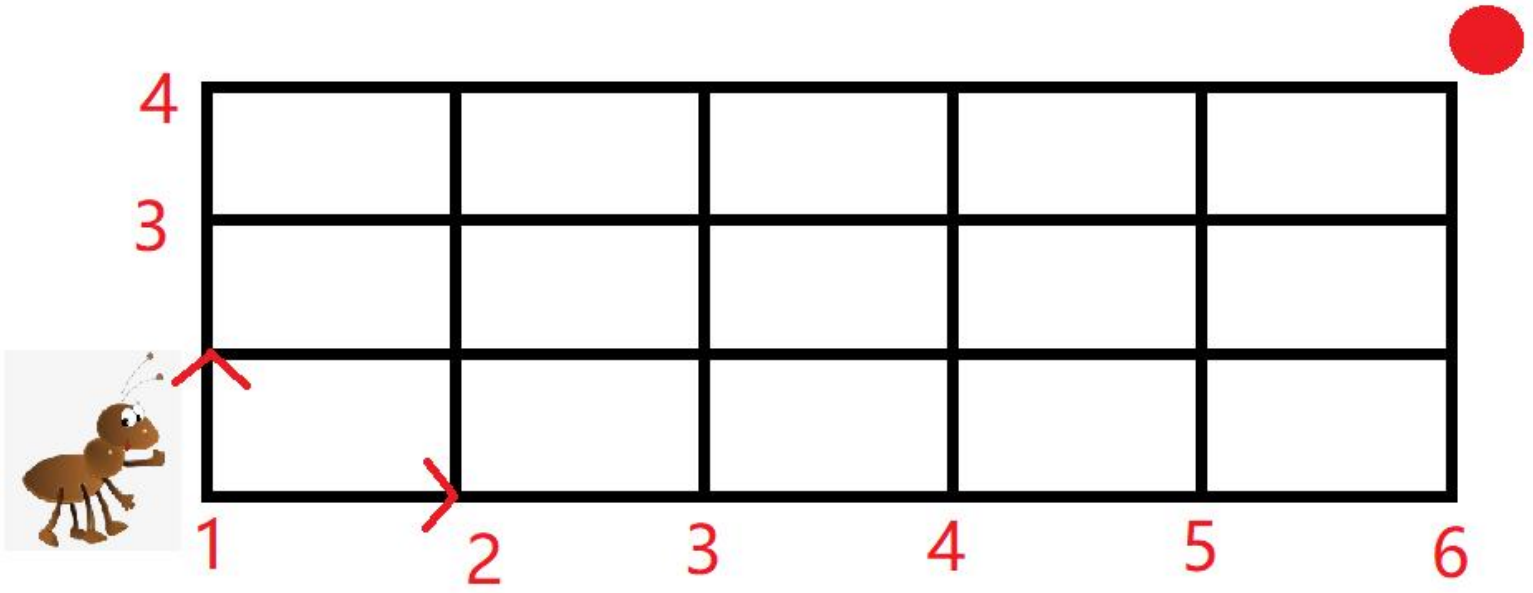
(1,1)　(1,2)　(1,3)

[2757]移动路线

蚂蚁共有3种移动路线：
路线1：(1,1) → (1,2) → (1,3) → (2,3)
路线2：(1,1) → (1,2) → (2,2) → (2,3)
路线3：(1,1) → (2,1) → (2,2) → (2,3)
输入
输入只有一行，包括两个整数m和n（$0 < m+n \le 20$），代表方格矩阵的行数和列数，m、n之间用空格隔开。
输出
输出只有一行，为不同的移动路线的数目。
样例输入
2 3
样例输出
3

思考

1.目标状态如何描述？
2. 递推式是什么？
3. 初始条件时什么？



代码

```
1  #include<cstdio>
2  #include<cstring>
3  #include<iostream>
4  using namespace std;
5  int a[30][30];
6  int main(){
7      int m,n,i,j;
8      scanf("%d%d",&m,&n);
9      for(i=1;i<=m;i++)a[i][1]=1;
10     for(j=1;j<=n;j++)a[1][j]=1;
11     for(i=2;i<=m;i++)
12         for(j=2;j<=n;j++)
13             a[i][j]=a[i-1][j]+a[i][j-1];
14     printf("%d",a[m][n]);
15     return 0;
16 }
```

1.3.6 [3443] 筷子

A先生有很多双筷子。确切的说应该是很多根，因为筷子的长度不一，很难判断出哪两根是一双的。这天，A先生家里来了K个客人，A先生留下他们吃晚饭。加上A先生，A夫人和他们的孩子小A，共K+3个人。每人需要用一双筷子。A先生只好清理了一下筷子，共N根，长度为T1,T2,T3,……,TN.现在他想**用这些筷子组合成K+3双，使每双的筷子长度差的平方和最小。**

[3443] 筷子

输入
共有两行，第一行为两个用空格隔开的整数，表示N,K(1≤N≤100, 0<K<50)，第二行共有N个用空格隔开的整数，为Ti.每个整数为1～50之间的数。
输出
仅一行。如果凑不齐K+3双，输出-1，否则输出长度差平方和的最小值。
样例输入
10 1
1 1 2 3 3 3 4 6 10 20
样例输出
5

思考

- 1. 目标状态如何描述？
- 2. 递推式是什么？
- 3. 初始条件时什么？

分析

1. 目标状态:使每双的筷子长度差的平方和最小

f[i][j]表示前i根组成j双筷子每双长度差的和的最小值

2. 递推推导

在考虑第i根筷子时,需要做出的决策是：要不要把这只筷子加入到最优解中去,

若加入，则其与第(i-1)根筷子组成一对，则
f[i][j]=f[i-2][j-1]+(a[i]-a[i-1])*(a[i]-a[i-1]),
否则f[i][j]=f[i-1][j] 。

分析

f[i][j]=f[i-2][j-1]+(a[i]-a[i-1])*(a[i]-a[i-1]),
否则：f[i][j]=f[i-1][j] 。

因为i与i-1 配对，产生一双筷子，这样剩下的筷子就是从1~i-2。该递推式的意思就是“前i根组成j双筷子每双长度差的和的最小值 等于 前i-2根组成j-1双筷子每双长度差的和的最小值+最后一双筷子长度差的平方”。

递推表达式：

```
f[i][j]=min(f[i-1][j],
            f[i-2][j-1]+(a[i]-a[i-1])*(a[i]-a[i-1]));
```

```cpp
using namespace std;
int main()
{
    int a[101],f[101][101];
    int k,i,j,n,t;
    scanf("%d%d",&n,&k);
    k=k+3;
    for (i=1;i<=n;i++) scanf("%d",&a[i]);
    sort(a+1, a + n+1);//将筷子按照从小到大排序
    if (k*2>n) printf("-1");//如果筷子不够则输出-1
    else
    {
        for(i=0;i<=n;i++)//初始化，f[i][j]表示i根筷子里取j双的最优值
        {   f[i][0]=0;
            for (j=1;j<=k;j++)f[i][j]=99999999;
        }

        for (i=2;i<=n;i++)//因为一根筷子没意义，所以从2根筷子开始更新
        for (j=1;j<=k;j++)
        f[i][j]=min(f[i-2][j-1]+(a[i]-a[i-1])*(a[i]-a[i-1]),f[i-1][j]);
        //每一个更新都有2种选择，
        //筷子减2、人减1的情况用i和i-1的筷子配对一个人，
        //以及用i-1根筷子配对j个人情况
        printf("%d",f[n][k]);
    }
}
```

今天的课程结束啦……

下课了…
同学们再见！