集训课-19：递推算法

课程目标

01 递推算法思想

02 斐波那契数列

03 平面分割

04 走楼梯

05 猴子分桃

Part 1

递推算法的思想

递推算法的思想

找规律：

①：4，7，10，13，16，（），（）

②：1，2，4，8，16，（），（）

③：2，4，8，14，22，（），44，（）

④：1，1，2，3，5，8，13，（），（）

⑤：1，8，27，64，125，（）

递推算法的思想

找规律：

①：4，7，10，13，16，（19），（22）

②：1，2，4，8，16，（32），（64）

③：2，4，8，14，22，（32），44，（58）

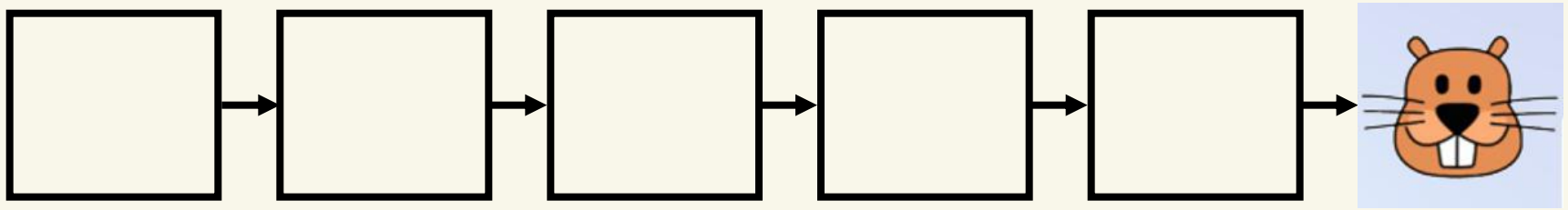
④：1，1，2，3，5，8，13，（21），（34）

⑤：1，8，27，64，125，（216）

递推算法的思想

再来一波：

如下所示，相邻画框中的图片只有一个特征发生了改变，尝试推理出这些图片。

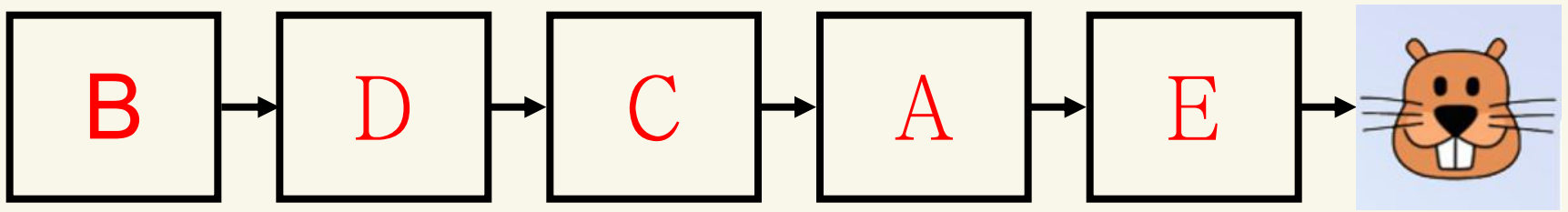


A　　B　　C　　D　　E

递推算法的思想

再来一波：

如下所示，相邻画框中的图片只有一个特征发生了改变，尝试推理出这些图片。



A　　B　　C　　D　　E

递推算法的思想

递推算法的思想是指从已知的初始条件出发，依据某种递推关系(规律)，逐次推出所要求的各中间结果及最后结果，而且会把每一次的状态都存储起来，以便下次直接使用。

递推算法的思想

递推公式

递推基础项

递推算法的思想是指从已知的初始条件出发，依据某种递推关系(规律)，逐次推出所要求的各中间结果及最后结果，而且会把每一次的状态都存储起来，以便下次直接使用。

递推算法的思想

递推算法分类

递推算法
- 顺推：根据初始值一步步推导
- 逆推：根据结果值一步步逆向推导

Part 2

斐波那契数列

斐波那契数列

- 斐波那契数列

  第一项和第二项都为1

  从第三项开始每一项都等于前面两项之和

  1 1 2 3 5 8 13 21 34 55 89……

斐波那契数列

- 斐波那契数列

  第一项和第二项都为1

  从第三项开始每一项都等于前面两项之和

下标i	0	1	2	3	4	5	i
数据a[i]	0	1	1				

占位作用　　递推基础项

斐波那契数列

- 斐波那契数列

  第一项和第二项都为1

  从第三项开始每一项都等于前面两项之和

下标i	0	1	2	3	4	5	i
数据a[i]	0	1	1	2			

占位作用　　递推基础项　　a[3]=a[1]+a[2];

斐波那契数列

- 斐波那契数列

  第一项和第二项都为1

  从第三项开始每一项都等于前面两项之和

下标i	0	1	2	3	4	5	i
数据a[i]	0	1	1	2	3		

占位作用　　递推基础项　　a[4]=a[2]+a[3];

斐波那契数列

- 斐波那契数列

第一项和第二项都为1

从第三项开始每一项都等于前面两项之和

下标i	0	1	2	3	4	5	i
数据a[i]	0	1	1	2	3	5	

占位作用　　递推基础项　　a[5]=a[3]+a[4];

斐波那契数列

- 斐波那契数列

  第一项和第二项都为1

  从第三项开始每一项都等于前面两项之和

下标i	0	1	2	3	4	5	i
数据a[i]	0	1	1	2	3	5	

占位作用　　递推基础项　　a[i]=a[i-1]+a[i-2];

斐波那契数列

输出斐波那契数列第n项(1<=n<=20)

（1）递推基础项

（2）递推公式

（3）递推求解

斐波那契数列

（1）递推基础项

```
int a[21] = {0,1,1};
```

（2）递推公式

```
a[i] = a[i-1] + a[i-2];
```

（3）递推求解

```
for (int i = 3; i <= n; i++)
{
    a[i] = a[i-1] + a[i-2];
}
```

输出

```
cout << a[n];
```

斐波那契数列

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

int main()
{
		int n, a[21] = {0, 1, 1};
		cin >> n;
		for (int i = 3; i <= n; i++)
		{
			a[i] = a[i-1] + a[i-2];
		}
	cout << a[n];
		return 0;
}
```

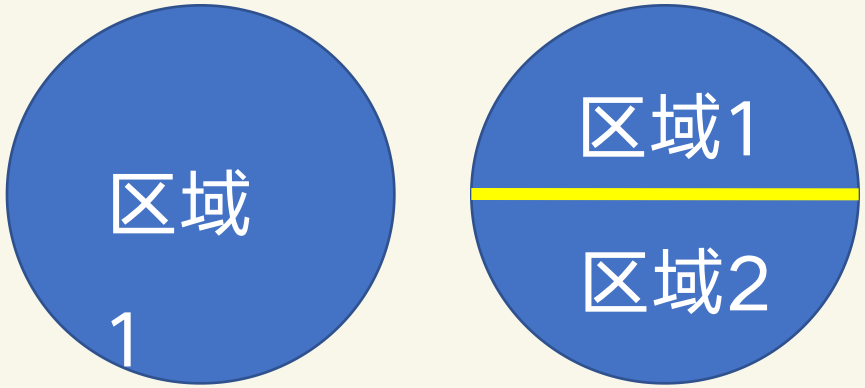
Part 3

平面分割

平面分割

【问题描述】
n条直线，最多可以把平面分为多少个区域？
(0<=n<=100)
【输入格式】
一行一个整数 n，表示直线的数量。
【输出格式】
一行一个整数，表示最多可以把平面分为多少个区域。
【输入样例】
2
【输出样例】
4

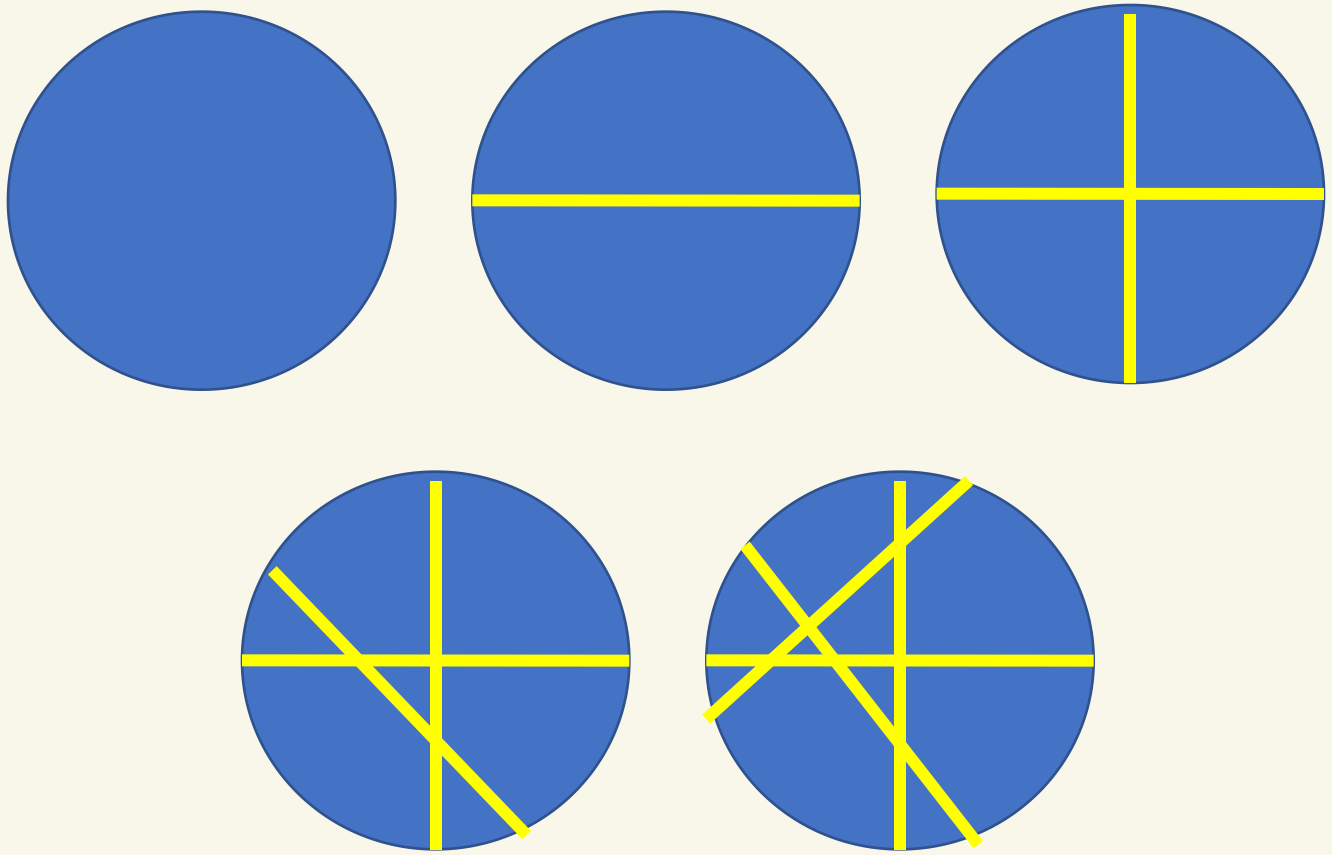
平面分割



直线数量i	0	1	2	3	4	……	n
区域数量a[i]	1	2				……	

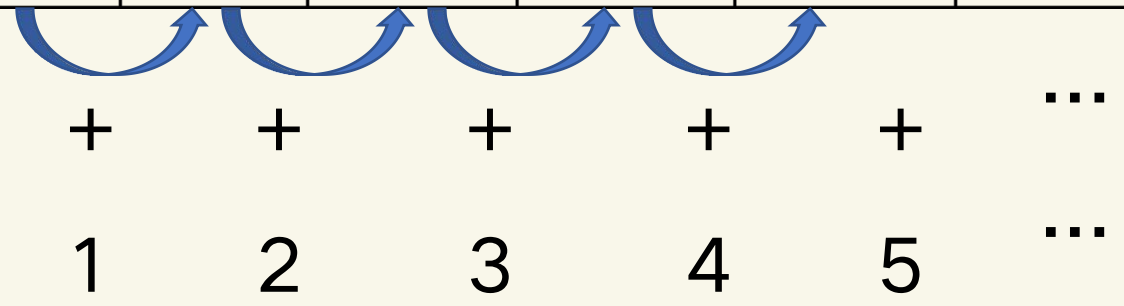
平面分割

每增加1条线，需要穿过前面所有的线



平面分割

直线数量i	0	1	2	3	4	……	n
区域数量a[i]	1	2	4	7	11	……	



平面分割

（1）递推基础项

```
int a[101] = {1};
```

（2）递推公式

```
a[i] = a[i-1] + i;
```

（3）递推求解

```
for (int i = 1; i <= n; i++)
{
    a[i] = a[i-1] + i;
}
```

平面分割

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

int main()
{
		int n, a[101] = {1};
		cin >> n;
		for (int i = 1; i <= n; i++)
		{
				a[i] = a[i-1] + i;
		}
		cout << a[n];
		return 0;
}
```

Part 4

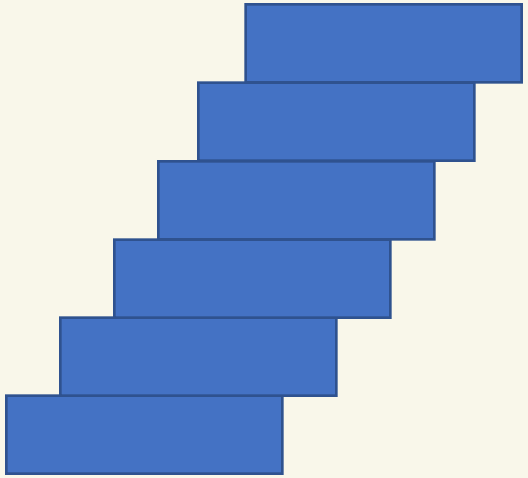
走樓梯

走楼梯

【问题描述】
小明是个非常好奇的人，他每天都会思考一些奇怪的问题，比如爬楼梯的时候，他就会想，如果每次可以上一级台阶，两级台阶或者三级台阶，那么上n(1<=n<=30)级台阶一共有多少种方案？
【输入格式】
一行一个整数 n，表示台阶数量。
【输出格式】
一行一个整数，表示一共有多少种方案。
【输入样例】
4
【输出样例】
7

走楼梯

每次可以上一级台阶，两级台阶或者三级台阶



方案

台阶数量i	0	1	2	3	4	5	6	……	n(n<=30)
方案数量a[i]	0	1	2	4	7	13		……	

走楼梯

台阶数量i	0	1	2	3	4	5	6	……	n(n<=30)
方案数量a[i]	0	1	2	4	7	13	24	……	

第一项为1，第二项为2，第三项为4

从第四项开始等于前三项之和

走楼梯

台阶数量i	0	1	2	3	4	5	6	……	n(n<=30)
方案数量a[i]	0	1	2	4	7	13	24	……	

（1）递推基础项

```
int a[31] = {0,1,2,4};
```

（2）递推公式

```
a[i] = a[i-1] + a[i-2] + a[i-3];
```

（3）递推求解

```
for (int i = 4; i <= n; i++)
{
    a[i] = a[i-1] + a[i-2] +a[i-3];
}
```

走樓梯

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

int main()
{
	int n, a[31] = {0, 1, 2, 4};
	cin >> n;
	for (int i = 4; i <= n; i++)
	{
		a[i] = a[i-1] + a[i-2] + a[i-3];
	}
	cout << a[n];
	return 0;
}
```

Part 5

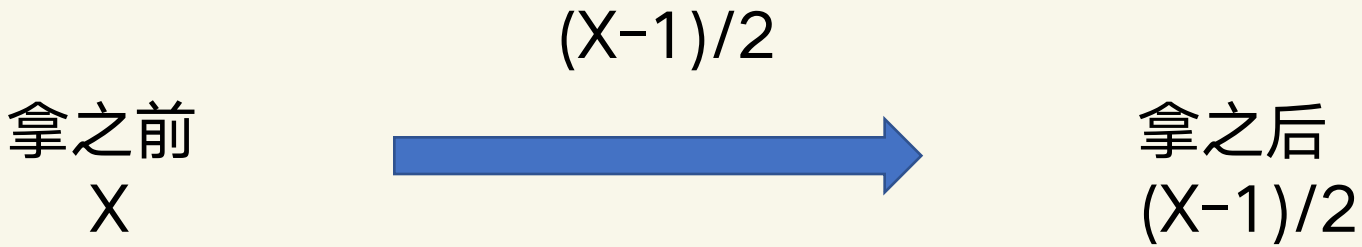
猴子分桃

猴子分桃

【问题描述】
海滩上有一堆桃子，N 只猴子来分。第一只猴子把这堆桃子中的一个扔入海中，然后将剩余的桃子平均分成2份，最后这只猴子拿走了 1 份。第二只猴子来了接着把剩下的桃子中的一个扔入海中，然后把剩余的桃子平均分成 2 份，开开心心拿走了 1 份。第三、第四、……，第 N 只猴子都是将剩下的桃子中的一个扔入海中，然后将剩余的桃子平均分成 2 份，并拿走其中的 1 份。最后只剩下 1 个桃子。编写程序，输入猴子的数量 N，输出海滩上原有多少桃子。
【输入格式】：一行，包含一个正整数 N（0<N<21）
【输出格式】：一个整数， 海滩上原有多少桃子
【输入样例】：2
【输出样例】：7
【数据范围】：0<N<21

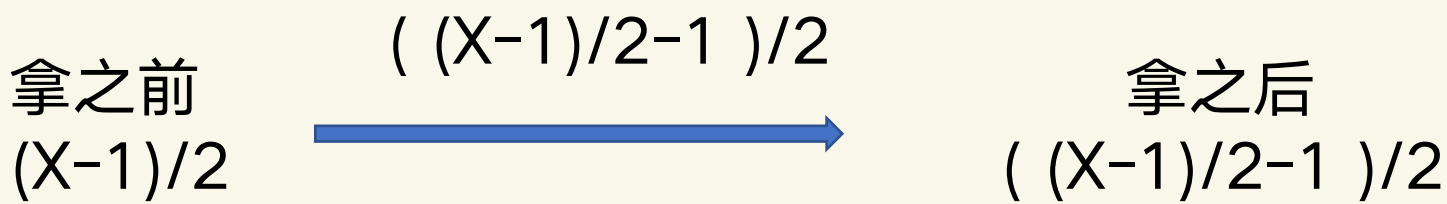
猴子分桃

第一只猴子把这堆桃子中的一个扔入海中，然后将剩余的桃子平均分成2份，最后这只猴子拿走了 1 份。



猴子分桃

第二只猴子来了接着把剩下的桃子中的一个扔入海中，然后把剩余的桃子平均分成 2 份，开开心心拿走了 1 份。



猴子分桃

(拿之前 - 1) / 2 = 拿之后

拿之后 → 拿之前

猴子分桃

第一只猴子拿完剩下的 =（原来有的-1）/ 2

第二只猴子拿完剩下的 =（第一只猴子拿完剩下的-1）/ 2

第三只猴子拿完剩下的 =（第二只猴子拿完剩下的-1）/ 2

第四只猴子拿完剩下的 =（第三只猴子拿完剩下的-1）/ 2

1

猴子分桃

第四只猴子拿完剩下的 =（第三只猴子拿完剩下的-1）/ 2=1

第三只猴子拿完剩下的 = 第四只猴子拿之前的

= 第四只猴子拿完剩下的*2+1

= 1*2+1

= 3

猴子分桃

第三只猴子拿完剩下的 =（第二只猴子拿完剩下的-1）/ 2=3

第二只猴子拿完剩下的 = 第三只猴子拿之前的

= 第三只猴子拿完剩下的*2+1

= 3*2+1

= 7

猴子分桃

每一轮拿之前 = 每一轮拿之后 * 2 + 1;

猴子分桃

1、定义个数变量，初始值为1

2、重复执行n次

3、每一轮使用个数变量 * 2 + 1

4、输出个数变量

猴子分桃

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

int main()
{
    int cnt = 1, n; //个数变量 轮数
    cin >> n;
    for (int i = 1; i <= n; i++)
    {
        cnt = cnt * 2 + 1; //每一轮使用个数变量 * 2 + 1
    }
    cout << cnt;
    return 0;
}
```

Class is over