



CS100

# 信息学算法入门

第二讲

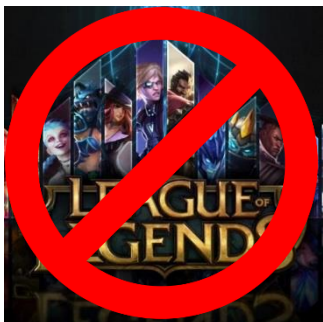
# 课程资料下载地址

---

CS100信息学算法入门公布资料的固定网站  
<http://pan.baidu.com/s/1jHDcMou>

请每次课前自行将资料下载到电脑

# 课堂纪律



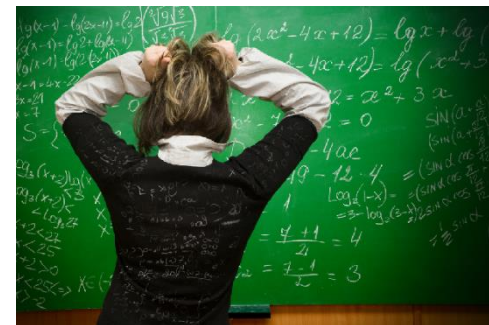
不准打游戏



手机静音



说话前先举手



遇到困难及时解决

# C++编程环境

windows



mac



# 最小公倍数

```
1  #include<iostream>
2  using namespace std;
3  long long x,y;
4  long long gcd(long long a, long long b){
5      if(a%b==0) return b;
6      else return gcd(b,a%b);
7  }
8  long long lcm(long long a, long long b){
9      return a/gcd(a,b)*b;
10 }
11 int main(){
12     cin>>x>>y;
13     cout<<lcm(x,y);
14     return 0;
15 }
```

# 砖头和金字塔

```
5 void pyramid(int m, char x){
6     int i, j;
7     for(i=0; i<m; i++){
8         for(j=0; j<m-i-1; j++) cout<< ' ';
9         for(j=0; j<i+i+1; j++) cout<<x;
10        cout<<endl;
11    }
12 }
13 int main(){
14     cin>>n>>c;
15     pyramid(n, c);
16     pyramid(2*n, c);
17     return 0;
18 }
```

# 卡特兰数 – 循环实现

```
1  #include<iostream>
2  using namespace std;
3  long long i,n,h[50]={1,1};
4  int main(){
5      cin>>n;
6      for(i=2;i<=n;i++)
7          h[i]=h[i-1]*(4*i-2)/(i+1);
8      cout<<h[n]<<endl;
9      return 0;
10 }
```

# 卡特兰数 – 递归实现

```
1  #include<iostream>
2  using namespace std;
3  long long h(int m){
4      if(m==0) return 1;
5      return h(m-1)*(4*m-2)/(m+1);
6  }
7  int main(){
8      long long i,n;
9      cin>>n;
10     cout<<h(n)<<endl;
11     return 0;
12 }
```



# 模块化编程—函数

C++中，通常采用**函数来进行模块封装**，对于函数，我们所关心的是对给定的输入，能否得到想要的输出，也就是执行结果

有些函数系统已经做好了，可直接调用，比如诸多的数学函数。有些时间，我们需要创作自己的函数。

# 四类函数

void 函数名()

void launch()

void 函数名(参数)

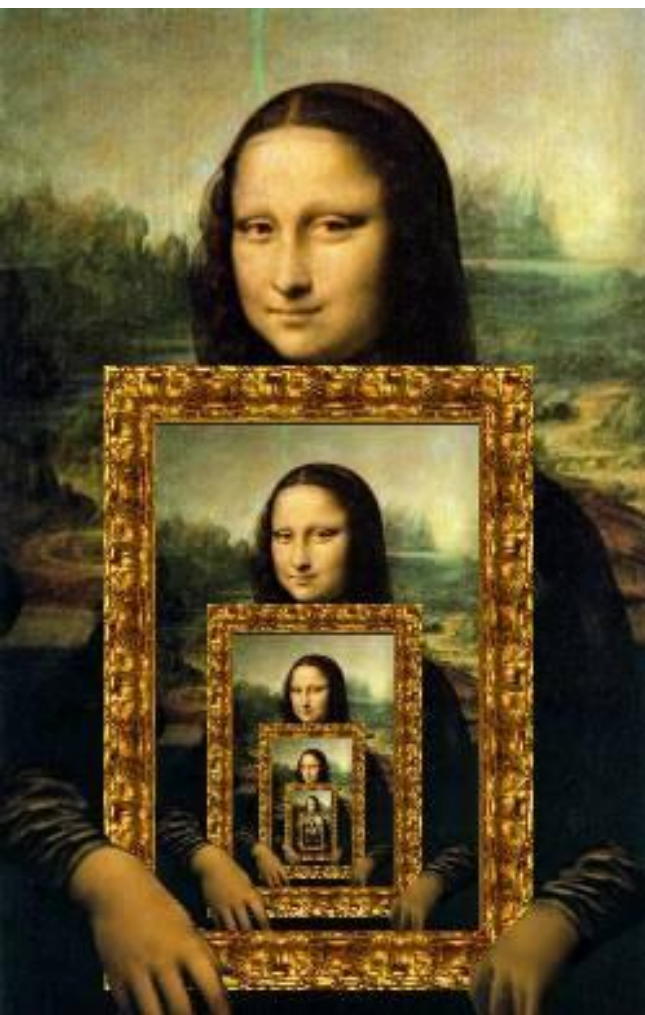
void launch(double x, double y)

返回类型 函数名()

int launch()

返回类型 函数名(参数)

int launch(double x, double y)



递归  
(recursion)



# 吓得我抱起了

抱着抱着抱着我的小鲤鱼的我的我的我







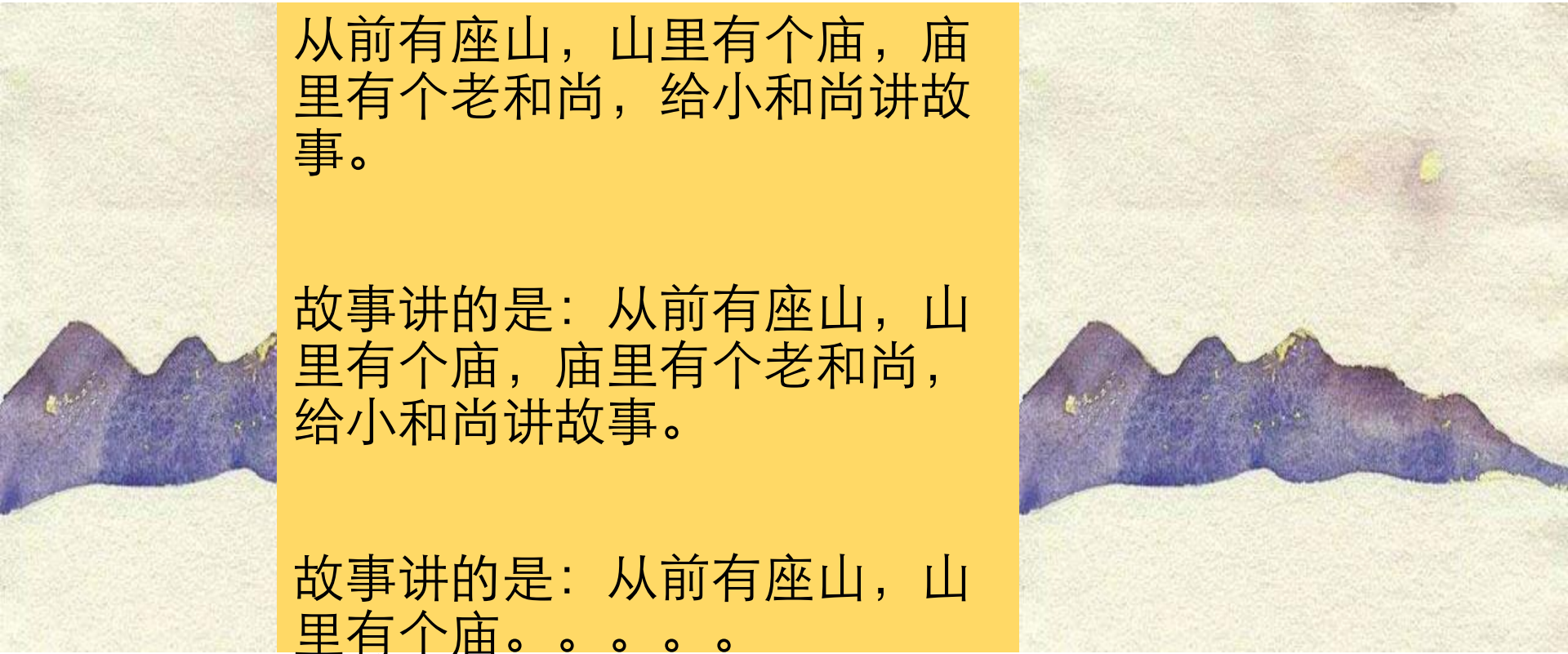




Hyphaene Compressa - Doum Palm

© Shlomit Pinter

# 生活中的递归现象



从前有座山，山里有个庙，庙里有个老和尚，给小和尚讲故事。

故事讲的是：从前有座山，山里有个庙，庙里有个老和尚，给小和尚讲故事。

故事讲的是：从前有座山，山里有个庙。。。。。



# 递归

你乘坐电梯时，前后各有一块镜子，你会发现你在镜子的成像数不过来。那是因为你在镜子**A**中的成像又在镜子**B**中成像，镜子**B**的成像又在镜子**A**中有成像，如此反复……。

# 递归

小明去找**A**经理解决问题，**A**经理说：。这件事情不归我管，去找**B**经理。。于是小明去找**B**经理。**B**经理说。这件事情不归我管，去找**A**经理。。如果两个经理的说辞不变，小明又始终听话，小明将永远往返于两者之间。

# 递归(recursion)

递归(recursion)是一个函数/功能在其定义中调用自身的一种方法。

递归通常把一个大型复杂的问题层层转化为一个与原问题相似的规模较小的问题来求解。

递归策略只需少量的程序就可描述出解题过程所需要的多次重复计算，大大地减少了程序的代码量。

递归的能力在于用有限的语句来定义对象的无限集合。用递归思想写出的程序往往十分简洁易懂。

# 阶乘函数

$$n! = n \times (n - 1) \times \cdots \times 2 \times 1$$

```
3  = long long f(int x){  
4      long long ans=1;  
5      for(int i=2;i<=x;i++)  
6          ans*=i;  
7      return ans;  
8  }
```

# 阶乘函数 - 递归算法

$$n! = n \times (n - 1) \times \cdots \times 2 \times 1$$

```
3 3 long long f(int x){  
4     if(x==0) return 1;  
5     return f(x-1)*x;  
6 }
```

—

# 阶乘函数 - 递归算法

$$n! = n \times (n - 1) \times \cdots \times 2 \times 1$$

```
3  = long long f(int x){  
4      if(x==0) return 1;  
5      return f(x-1)*x;  
6  }
```


递归终  
止条件

递归前  
进步骤

# 最大公约数： gcd()函数

```
3 ☐ int gcd(int x,int y){  
4     int tmp;  
5 ☐ while (tmp=x%y){  
6         x=y;  
7         y=tmp;  
8     }  
9     return y;  
10 }
```

# 最大公约数： gcd()函数

```
3  int gcd(int x, int y){  
4     if(x%y==0) return y;  
5     return gcd(y, x%y);  
6 }
```

递归终止条件

递归前进步骤



# 递归方法解汉诺塔游戏

有三根杆子A，B，C。A杆上有N个( $N > 1$ )穿孔圆盘，盘的尺寸由下到上依次变小。要求按下列规则将所有圆盘移至C杆：

- 1、每次只能移动一个圆盘；
- 2、大盘不能叠在小盘上面。

提示：可将圆盘临时置于B杆，也可将从A杆移出的圆盘重新移回A杆，但都必须遵循上述两条规则。

问：如何移？最少要移动多少次？



# 递归方法解汉诺塔游戏

$N=1$ 时，1个圆盘的移动总次数=1

$N>1$ 时， $N$ 个圆盘移动总次数=  $(N-1)$ 个圆盘移动总次数 \* 2 + 1

# 作业

---

作业网站:

<http://120.132.20.20:8080/thrall-web/main#home>