中南大學
CENTRAL SOUTH UNIVERSITY

第 九 讲

函　　数

9、 函 数

什么是函数?

打印图形:

```
**********
Hello world!
**********
**********
```

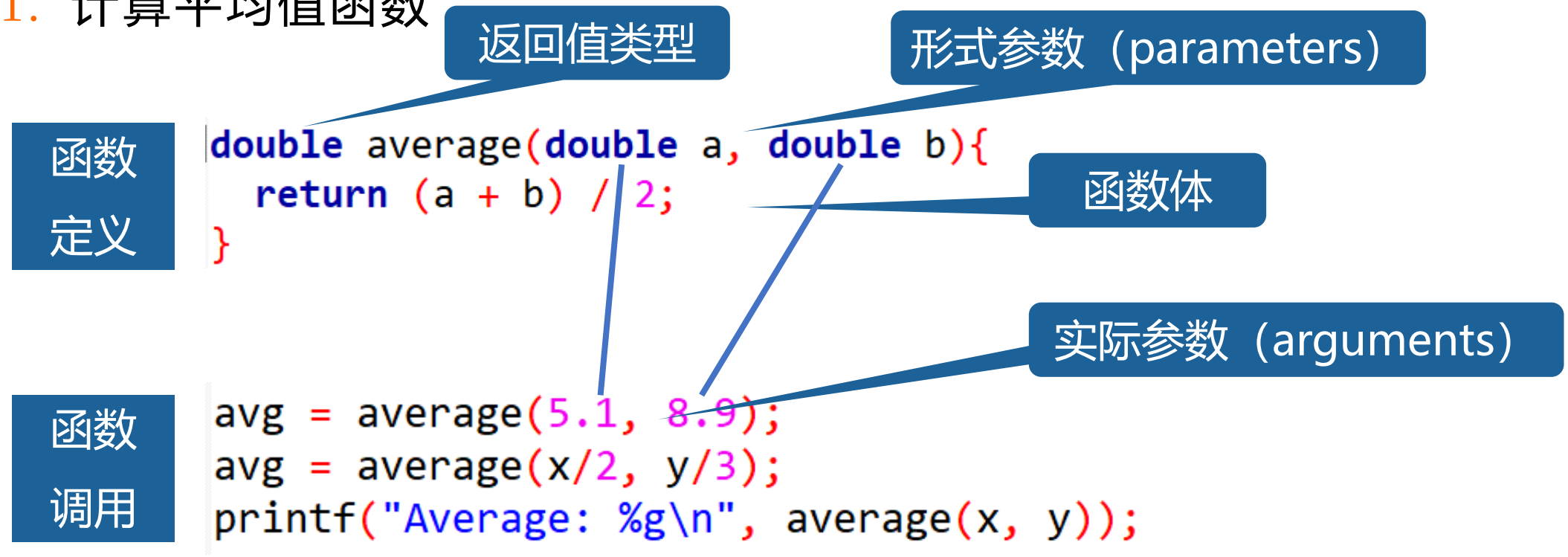
```c
#include <stdio.h>
int main(void)
{   printf("**********\n");
    printf("Hello world!\n");
    printf("**********\n");
    printf("**********\n");
    return 0;
}
```

```c
#include <stdio.h>
void printstar(void);
int main(void){
   printstar();
   printf("Hello world!\n");
   printstar();
   printstar();
   return 0;
}
//三个语句一样,可编写成函数:
void printstar(void){
   printf("**********\n");
}
```

每行打印*的个数不同?

9.1函数的定义和调用

1. 计算平均值函数



9、函数

9.1函数的定义和调用

- 计算平均值函数

```
Enter three numbers: 3.5 9.6 10.2
Average of 3.5 and 9.6: 6.55
Average of 9.6 and 10.2: 9.9
Average of 3.5 and 10.2: 6.85
```

```c
/* Computes pairwise averages of three numbers */
#include <stdio.h>

double average(double a, double b){
  return (a + b) / 2;
}
int main(void){
  double x, y, z;
  printf("Enter three numbers: ");
  scanf("%lf%lf%lf", &x, &y, &z);
  printf("Average of %g and %g: %g\n", x, y, average(x, y));
  printf("Average of %g and %g: %g\n", y, z, average(y, z));
  printf("Average of %g and %g: %g\n", x, z, average(x, z));
  return 0;
}
```

9.1函数的定义和调用

2. 显示倒计数

```c
/* Prints a countdown */
#include <stdio.h>
void print_count(int n){
  printf("T minus %d and counting\n", n);
}
int main(void){
  int i;
  for (i = 10; i > 0; --i)
    print_count(i);
  return 0;
}
```

9.1函数的定义和调用

3. 显示双关语

```c
/* Prints a bad pun */
#include <stdio.h>
void print_pun(void){
    printf("To C, or not to C: that is the question.\n");
}
int main(void){
   print_pun();
   return 0;
}
```

9.1函数的定义和调用

函数定义

返回值类型 函数名(形式参数){
声明
语句
}

```
double average(double a, double b){
  double sum;          /* declaration */

  sum = a + b;         /* statement */
  return sum / 2;      /* statement */
}
```

函数返回值类型 不能返回数组;void类型表示没有返回值;如果省略则C89中默认int型,但C99中不允许省略。

形式参数有关说明 没有参数用void表示;每个形参即使类型相同也要分别进行类型说明。 **double average(double a, b);**

9.1函数的定义和调用

函数调用

- 在一个函数中调用另一个函数需要具备的条件
  - 被调用函数已经存在(库函数或用户自定义);
  - 使用库函数或别的文件中的函数,在本文件开头用#include命令进行文件包含
  - 若使用用户自己定义的函数,而且该函数与主调函数在同一个文件中,则需进行函数声明

```
average(x, y);
print_count(i);        print_pun;    /*** WRONG ***/
print_pun();
```

9.1函数的定义和调用

判定素数

```c
/* Tests whether a number is prime */
#include <stdbool.h>   /* C99 only */
#include <stdio.h>
bool is_prime(int n){
  int divisor;
  if (n <= 1)
    return false;
  for (divisor = 2; divisor * divisor <= n; divisor++
    if (n % divisor == 0)
      return false;
  return true;
}
```

```c
int main(void){
  int n;

  printf("Enter a number: ");
  scanf("%d", &n);
  if (is_prime(n))
    printf("Prime\n");
  else
    printf("Not prime\n");
  return 0;
}
```

1. 两个n有何区别?
2. return语句多个?

9.2 函数声明

```
8       warning: implicit declaration of function 'average' [-Wimplicit-function-declaration]
14      error: conflicting types for 'average'
8       note: previous implicit declaration of 'average' was here
        === Build failed: 1 error(s), 1 warning(s) (0 minute(s), 0 second(s)) ===
```

```c
#include <stdio.h>

int main(void){
  double x, y, z;

  printf("Enter three numbers: ");
  scanf("%lf%lf%lf", &x, &y, &z);
  printf("Average of %g and %g: %g\n", x, y, average(x, y));
  printf("Average of %g and %g: %g\n", y, z, average(y, z));
  printf("Average of %g and %g: %g\n", x, z, average(x, z));

  return 0;
}
double average(double a, double b){
   return (a + b) / 2;
}
```

编译器为函数创建了一个隐式声明，假设函数返回int类型。
无法检查实参个数和实参类型

9.2 函数声明

返回值类型 函数名(形式参数);

函数声明（函数原型）

```
double average(double,double);
```

```c
#include <stdio.h>
double average(double a, double b);    /* DECLARATION */
int main(void){
   double x, y, z;
   printf("Enter three numbers: ");
   scanf("%lf%lf%lf", &x, &y, &z);
   printf("Average of %g and %g: %g\n", x, y, average(x, y));
   printf("Average of %g and %g: %g\n", y, z, average(y, z));
   printf("Average of %g and %g: %g\n", x, z, average(x, z));
   return 0;
}
double average(double a, double b)     /* DEFINITION */
{
    return (a + b) / 2;
}
```

9.3 实际参数

- 实际参数传的是**值**，**单向**传递
- 函数调用是，**计算**出每个实际参数的值并且把它**赋值**给相应的形式参数
- 函数执行过程中，对形式参数的改变**不会影响**实际参数的值

```
void decompose(double x, long int_part,double frac_part){
   int_part = (long) x;
   frac_part = x - int_part;
}

decompose(3.14159, i, d);
```

变量 i 和 d 不会受到影响，函数调用前后的值是完全一样的

9.3 实际参数

数组作为函数参数

- 数组经常被用为实际参数
- 一维数组时，可以（通常）不说明数组的长度

```
int f(int a[])  /* no length specified */
{
    …
}
```

```
#define LEN 100
int main(void){
   int b[LEN], total;
   …
   total = sum_array(b, LEN);
   …
}
```

```
int sum_array(int a[], int n){
   int i, sum = 0;
   for (i = 0; i < n; i++)
      sum += a[i];
   return sum;
}
```

```
int sum_array(int a[], int n);

int sum_array(int [], int);
```

9.3 实际参数

数组作为函数参数

- 函数可以改变数组型形式参数的元素，且改变会在相应的实际参数中体现出来

```
void store_zeros(int a[], int n){
   int i;
   for (i = 0; i < n; i++)
     a[i] = 0;
}

store_zeros(b, 100);
```

会将**数组b**的前100元素中赋值0

9.4 return 语句

return 表达式;

```
return 0;
return status;

return n >= 0 ? n : 0;
```

不一致　若函数返回值类型与return语句中表达式类型不一致时，系统隐式转换为函数返回值类型。

void　非void函数中必须有return语句返回一个值，void函数中也可以用return语句提前返回。

9.5 程序终止

- main参数
  - 以往C程序中常省略main函数返回值类型,默认int
  - main函数中的返回值是状态码,用return语句和exit函数都可以终止程序
- exit函数
  - 非main函数只能用exit函数终止程序
  - <stdlib.h>
    - 正常终止: exit(0); 或exit(EXIT_SUCCESS);
    - 异常终止: exit(1); 或exit(EXIT_FAILURE);

9.6 递归

- 如果函数调用它本身，那么此函数就是递归的

计算fact(n)=n!=n*(n-1)!

分治法

递归条件

必须有一个终止条件。

一个大问题转化成一个小问题,解题方法相似,而且每一步转化都越接近终止条件。

$$fact(n) = \begin{cases} 1 & (n \le 1) \quad \text{终止条件} \\ n*fact(n-1) & (n>1) \end{cases}$$

求n!与求(n-1)!的方法一样

9.6 递归

```c
#include <stdio.h>
int fact(int n)
{
    if (n <= 1)
        return 1;
    else
      return n * fact(n - 1);
 }
int main(void)
{
    printf("%d!=%d\n",3,fact(3));
    return 0;
}
```

→main()
→fact(3)中3不满足≤1执行 return 3*fact(2);
→fact(2)中2不满足≤1执行 return 2*fact(1);
→fact(1)中1满足≤1执行 return 1;返回1
→fact(2)返回2*1即2
→fact(3)返回3*2即6
→main函数输出

9.6 递归

快速排序算法

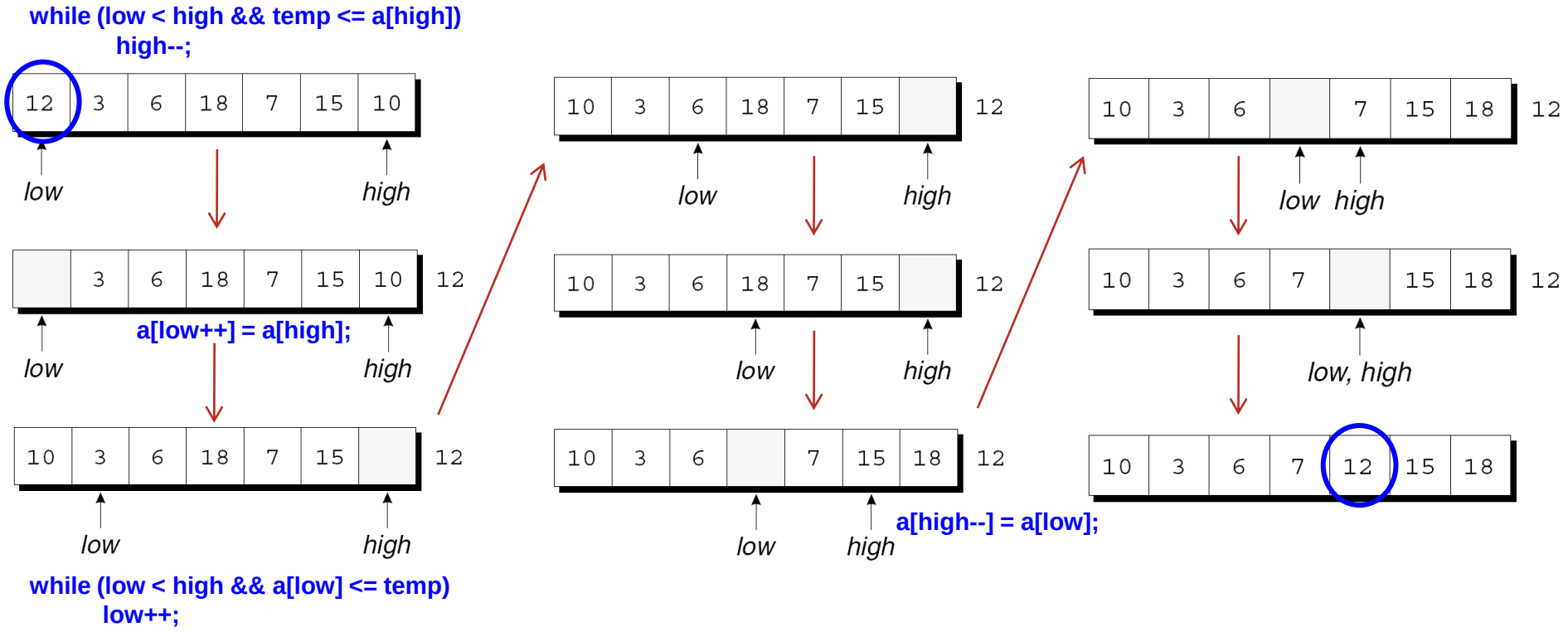
- 快速排序（英语：Quicksort），又称分区交换排序
- 使用分治法策略来把一个序列分为较小和较大的2个子序列，然后递归地排序两个子序列
- 步骤为：
  - 挑选基准值：从数列中挑出一个元素，称为“基准”（pivot）
  - 分割：重新排序数列，所有比基准值小的元素摆放在基准前面，所有比基准值大的元素摆在基准后面（与基准值相等的数可以到任何一边）。在这个分割结束之后，对基准值的排序就已经完成
  - 递归排序子序列：递归地将小于基准值元素的子序列和大于基准值元素的子序列排序

9.6 递归

http://vlab.csu.edu.cn/pljp/#/problems?id=1004

快速排序算法

```
Enter 7 numbers to be sorted: 12 3 6 18 7 15 10
In sorted order: 3 6 7 10 12 15 18
```



9.6 递归

求x的平方根

分析:在0~x之间,以0.0001的间距查找某个满足t^2等于x的t.

注意两个实数是否相等的判定

算法分析

- ①t取值0
- ②如果t^2等于x,则t即为所求的解,转④
- ③t的值增加0.0001,转②
- ④输出t的值

- 问题:当数很大,精度要求高时,穷举效率很低,如何优化?

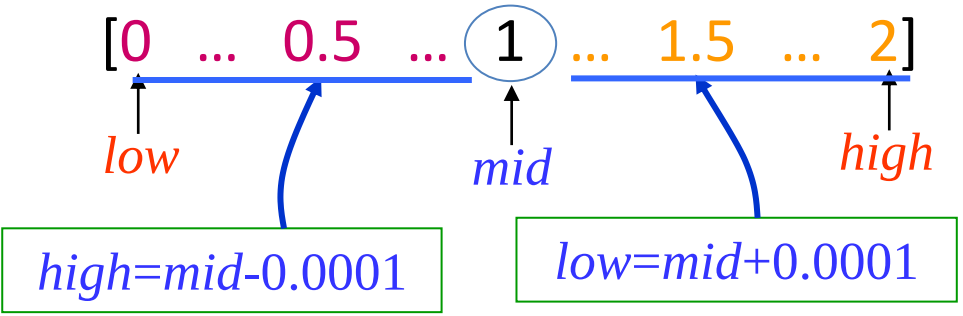
找4的平方根要找20000次

9.6 递归

求x的平方根

- 分治法计算任意数x的平方根,保留到小数点后4位。

分析:找到0~x的中间位置,根据该位置数据的平方与x的关系确定找到,左边或者右边继续查找,不断将问题的规模缩小.



算法分析

①$mid=(low+high)/2$

②如果mid^2等于2,则t即为所求的解,转③;

如果mid^2小于2,则在右边查找,$low=mid$+0.0001,转①;

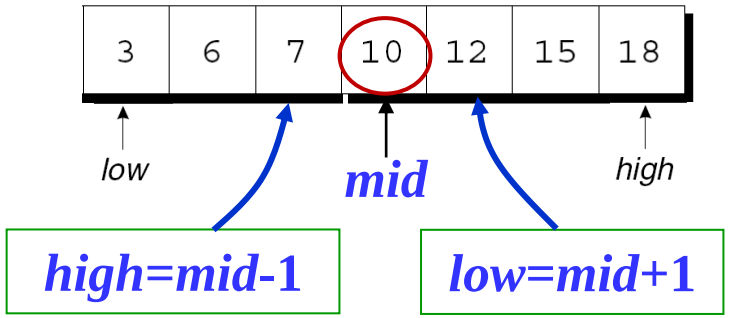
如果mid^2大于2,则在左边查找,$high=mid$-0.0001,转①;

③输出mid的值

9.6 递归

折半查找函数

函数原型:int BinSearch(int a[],int low,int high,int x)



算法分析

①如果*low*>*high*,则返回-1

②*mid*=(*low*+*high*)/2

③如果a[*mid*]==*x*,则返回*mid*

如果a[*mid*]>*x*,则在**右边**查找,*low*=*mid*+1

如果a[*mid*]<*x*,则在**左边**查找,*high*=*mid*-1

分别用循环和递归处理

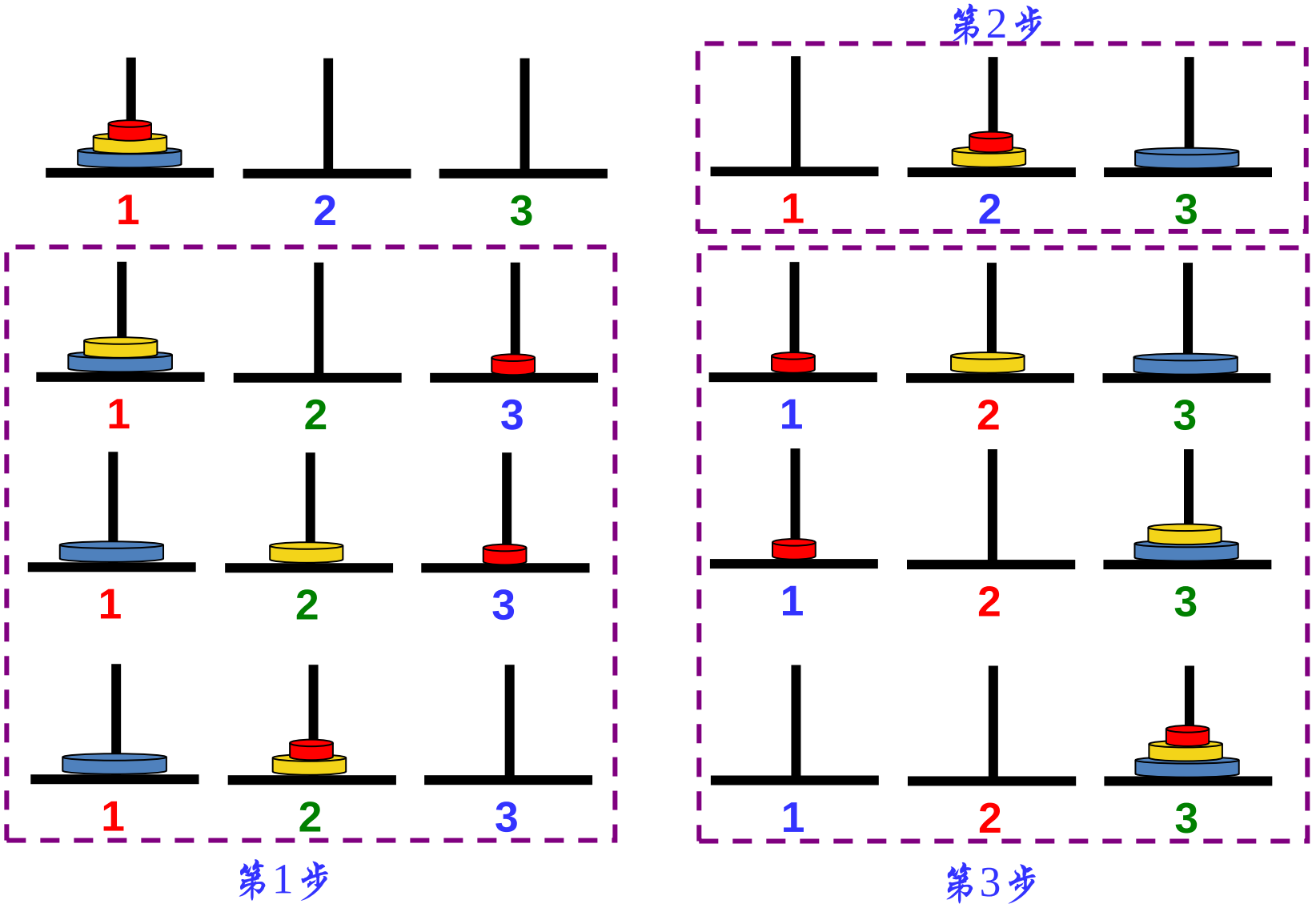
9.6 递归

汉诺塔

- 汉诺塔问题是源于印度一个古老传说,大梵天创造世界的时候做了三根金刚石柱子,在一根柱子上从下往上按照大小顺序摞着64片黄金圆盘.大梵天命令婆罗门把圆盘从下面开始按大小顺序重新摆放在另一根柱子上,并且规定,在小圆盘上不能放大圆盘,在三根柱子之间一次只能移动一个圆盘.

9.6 递归

汉诺塔



9.6 递归

汉诺塔

http://vlab.csu.edu.cn/pljp/#/problems?id=1734

函数原型:void hanoi(int n,int start,int trans,int dest)
void move(int here, int there)

算法分析

①如果原始柱上只剩下1个圆盘,直接将其移到目标柱上;
②否则,执行以下三个步骤:
第1步:先将原始柱上的n-1个圆盘通过目标柱移到中间柱上
第2步:再将原始柱上剩下的1个圆盘移到目标柱上
第3步:将中间柱上的n-1个圆盘通过原始柱移到目标柱上