第7章 数组

——数组的其他应用之文曲星猜数游戏

文曲星猜数游戏

- 由计算机随机生成一个各位相异的4位数字，由人来猜
- 每次提示：xAxB
  - A前面的数字表示有几个数字猜对位置也对了
  - B前面的数字表示有几个数字猜对但位置不对
- 思路
  - 用数组a存计算机随机生成的各位相异的4位数:MakeDigit(a)
  - 用数组b存人猜的4位数:InputGuess(b)
  - 比较a和b的相同位置元素，得到A前面数字:IsRightPosition(a, b)
  - 比较a和b的不同位置元素:IsRightDigit(a, b)

```
a[i] 4213
b[i] 1234
     1A3B
     4231
     2A2B
     4213
     4A0B
```

- 部分主函数

```
int main()
{  ……
   MakeDigit(a);          /*随机生成一个各位相异的4位数字 */
   printf("How many times do you want to guess?");
   scanf("%d", &level); /*最多允许猜的次数*/
   count = 0;             /*记录用户猜的次数*/
   do{
       printf("No.%d of %d times\n", count, level);
       printf("Input your guess:\n");
       if (InputGuess(b) == 0)  continue;
       count++;          /*记录已经猜的次数*/
       rightPosition = IsRightPosition(a, b);          /*统计数字和位置都猜对的个数*/
       rightDigit = IsRightDigit(a, b) - rightPosition;/*统计数字猜对位置不对的个数*/
       printf("%dA%dB\n", rightPosition, rightDigit);
   }while (rightPosition != 4 && count < level );
   if (rightPosition == 4)
       printf("Congratulations,you guess the right number at No.%d\n", count);
   else
       printf("Sorry,you haven't guess the right number,see you next time!\n");
   printf("Correct answer is:%d%d%d%d\n", a[0], a[1], a[2], a[3]);
   return 0;
}
```

文曲星猜数游戏

- 随机生成一个各位相异的4位数字——第1种方法

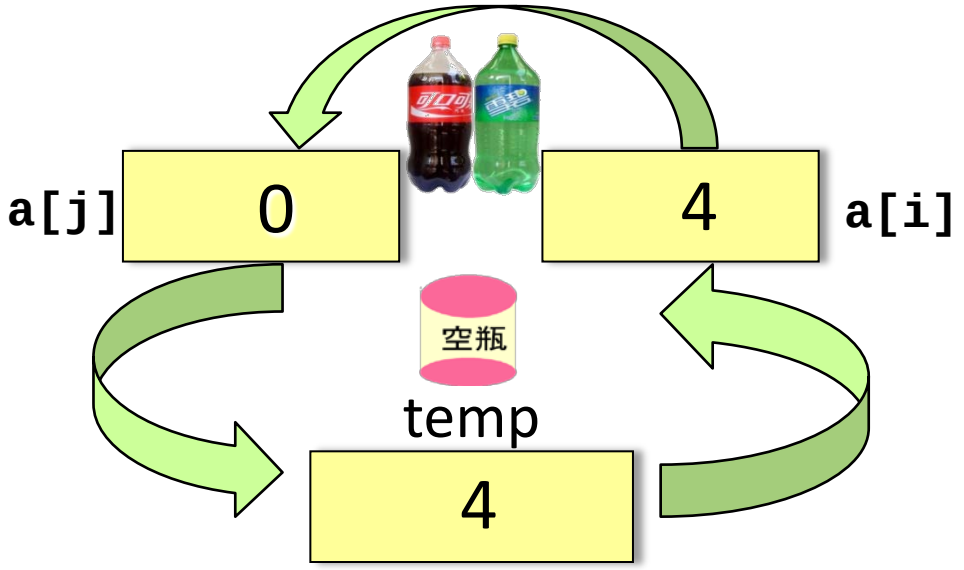
```
void MakeDigit(int a[])
{
   srand(time(NULL));
   a[0] = rand()%10;           /*千位数字 */
   do
   {
       a[1] = rand() % 10;     /*百位数字 */
   }while (a[0] == a[1]);
   do
   {
       a[2] = rand() % 10;     /*十位数字 */
   }while (a[0] == a[2] || a[1] == a[2]);
   do
   {
       a[3] = rand() % 10;     /*个位数字 */
   }while (a[0] == a[3] || a[1] == a[3] || a[2] == a[3]);
}
```

文曲星猜数游戏

- 随机生成一个各位相异的4位数字——第2种方法

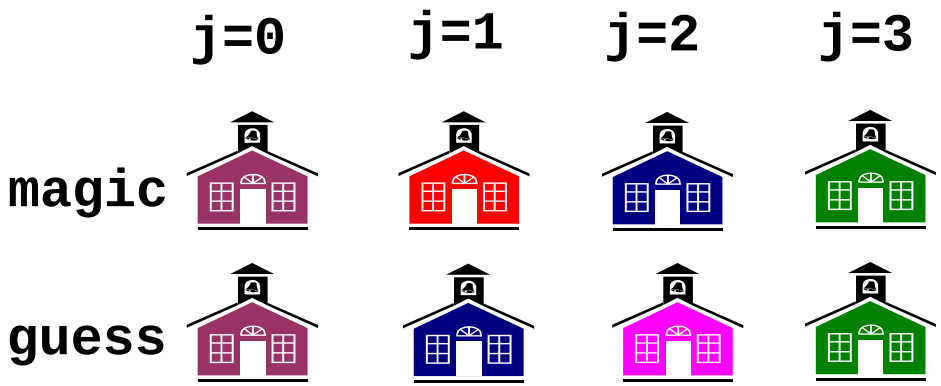
```
void MakeDigit(int a[])
{
   int i, j, temp;
   srand(time(NULL));
   for (i=0; i<10; i++)
   {
        a[i] = i;
   }
   for (i=0; i<10; i++)
   {
        j = rand() % 10;
        temp = a[j];
        a[j] = a[i];
        a[i] = temp;
   }
}
```

0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9
4	1	2	3	0	5	6	7	8	9



文曲星猜数游戏

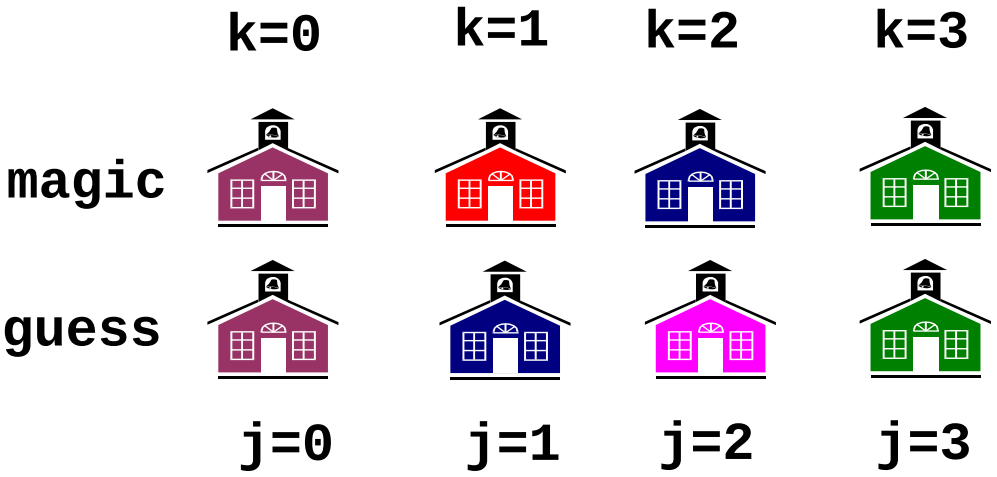
- 统计**数字和位置都猜对**的个数，对guess和magic的**相同**位置的元素进行比较，得到**A**前面的数字




```
int IsRightPosition(int magic[], int guess[])
{
   int rightPosition = 0;
   int j;
   for (j=0; j<4; j++)
   {
        if (guess[j] == magic[j])
        {
            rightPosition++;
        }
   }
   return rightPosition;
}
```

文曲星猜数游戏

- 统计数字猜对（不管位置是否猜对）的数字个数，对guess和magic的不同位置的元素进行比较




```
int IsRightDigit(int magic[], int guess[])
{
   int rightDigit = 0;
   int j, k;
   for (j=0; j<4; j++)
   {
        for (k=0; k<4; k++)
        {
           if (guess[j] == magic[k])
           {
               rightDigit++;
           }
        }
   }
   return rightDigit;
}
```

文曲星猜数游戏

- 统计**数字猜对但位置没猜对**的数字个数，得到**B**前面的数字

```
rightPosition = IsRightPosition(a, b);  /*统计数字和位置都猜对的个数*/
rightDigit = IsRightDigit(a, b);         /*统计人猜对的数字个数*/
rightDigit = rightDigit - rightPosition; /*统计数字猜对但位置不对的个数*/
```

```
int InputGuess(int b[])
{
   int i, ret;
   for (i=0; i<4; i++)
   {
       ret = scanf("%1d", &b[i]);
       if (ret != 1)
       {
          printf("Input Data Type Error!\n");
          fflush(stdin); /*清空输入缓冲区 */
          return 0; /*输入数据不合法 */
       }
   }
   if (b[0]==b[1]||b[0]==b[2]||b[0]==b[3]||b[1]==b[2]||b[1]==b[3]||b[2]==b[3])
   {
       printf("The numbers must be different from each other, input again\n");
       return 0;   /*输入数据不合法 */
   }
   else
   {
       return 1;   /*输入数据合法 */
   }
}
```

- 输入用户猜的数

问题：为什么结果会这样？

```
#include <stdio.h>
double Fact(unsigned int n);
int main()
{
   int i;
   for (i=1; i<=40; i++)
   {
        printf("%d! = %.0f\n", i, Fact(i));
   }
   return 0;
}
double Fact(unsigned int n)
{
   unsigned int i;
   double result = 1;
   for (i=1; i<=n; i++)
   {
        result = result * i;
   }
   return result;
}
```

```
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880
10! = 3628800
11! = 39916800
12! = 479001600
13! = 6227020800
14! = 87178291200
15! = 1307674368000
16! = 20922789888000
17! = 355687428096000
18! = 6402373705728000
19! = 121645100408832000
20! = 2432902008176640000
21! = 51090942171709440000
22! = 1124000727777607700000
23! = 25852016738884978000000
24! = 620448401733239410000000
25! = 15511210043330986000000000
26! = 403291461126605650000000000
27! = 10888869450418352000000000000
28! = 304888344611713840000000000000
29! = 8841761993739700800000000000000
30! = 265252859812191030000000000000000
31! = 8222838654177922400000000000000000
32! = 263130836933693520000000000000000000
33! = 8683317618811885900000000000000000000
34! = 295232799039604120000000000000000000000
35! = 10333147966386144000000000000000000000000
36! = 371993326789901180000000000000000000000000
37! = 13763753091226343000000000000000000000000000
38! = 523022617466601040000000000000000000000000000
39! = 20397882081197442000000000000000000000000000000
40! = 815915283247897680000000000000000000000000000000
```

讨论

- 1）挑战类型表示的极限 —— 计算50位的n!
  - 大数的存储问题
- 2）将文曲星猜数游戏写完整，看看有什么窍门猜得更快？

```
Enter a number to be calculated:
40
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880
10! = 3628800
11! = 39916800
12! = 479001600
13! = 6227020800
14! = 87178291200
15! = 1307674368000
16! = 20922789888000
17! = 355687428096000
18! = 6402373705728000
19! = 121645100408832000
20! = 2432902008176640000
21! = 51090942171709440000
22! = 1124000727777607680000
23! = 25852016738884976640000
24! = 620448401733239439360000
25! = 15511210043330985984000000
26! = 403291461126605635584000000
27! = 10888869450418352160768000000
28! = 304888344611713860501504000000
29! = 8841761993739701954543616000000
30! = 265252859812191058636308480000000
31! = 8222838654177922817725562880000000
32! = 263130836933693530167218012160000000
33! = 8683317618811886495518194401280000000
34! = 295232799039604140847618609643520000000
35! = 10333147966386144929666513375232000000000
36! = 371993326789901217467999448150835200000000
37! = 13763753091226345046315979581580902400000000
38! = 523022617466601111760007224100074291200000000
39! = 20397882081197443358640281739902897356800000000
40! = 815915283247897734345611269596115894272000000000
```