

二维数组

含义

一维数组是相同数据类型元素的集合，但是只能表示一行数据。若是存在行和列相关的信息(例如矩阵)，我们就需要用二维数组来表示。

本质

元素为数组的数组

定义方式

dataType arrayName[length1][length2];

- dataType 为数据类型
- arrayName 为数组名
- length1 为第一维下标的长度
- length2 为第二维下标的长度

数据类型 数组名{行数}[列数];

二维数组看做一个 Excel 表格，有行有列，length1 表示行数，length2 表示列数，要在二维数组中定位某个元素，必须同时指明行和列

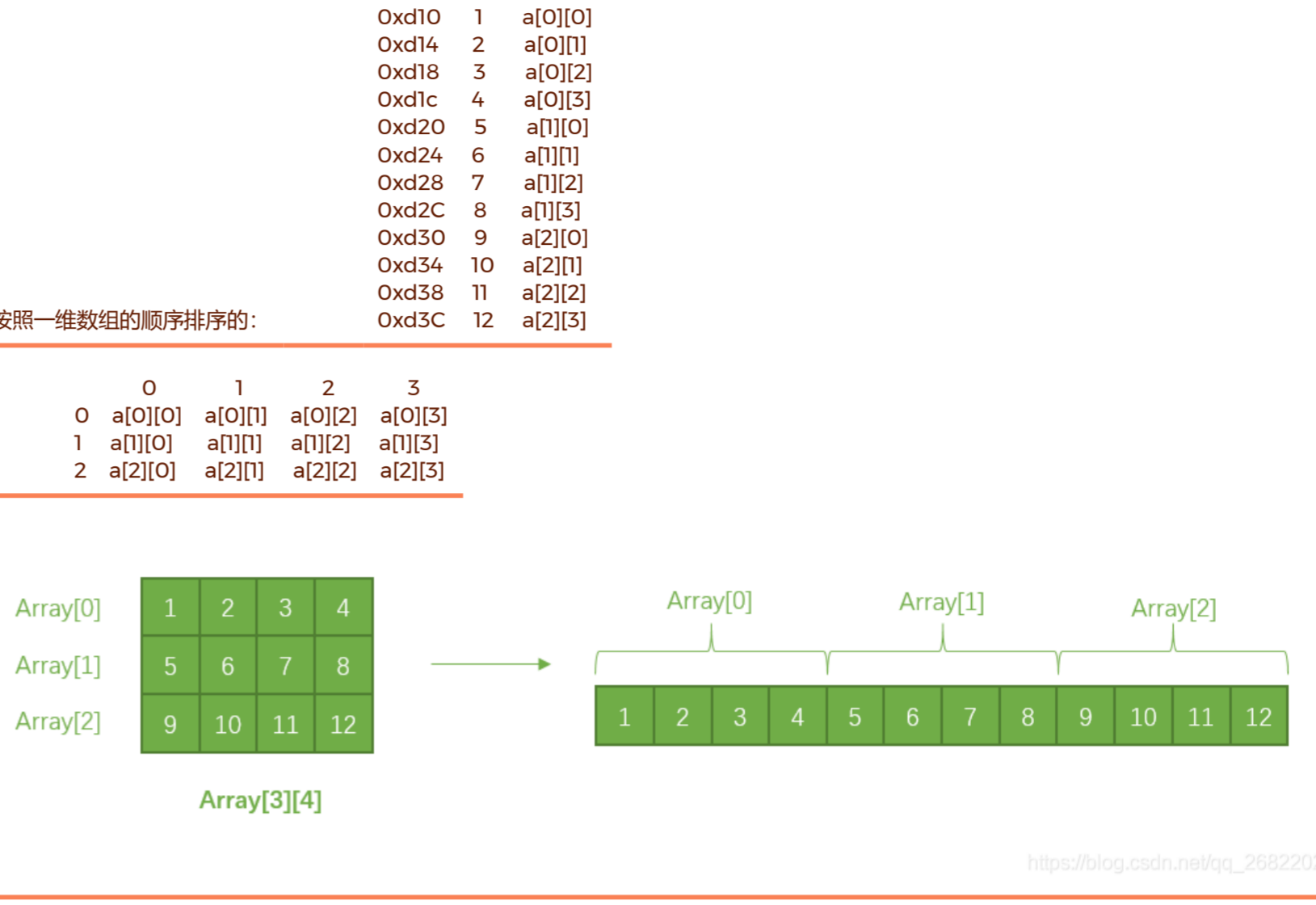
类比

例如: int a[3][4]; //三行四列

在内存中按照还是一维数组的顺序排序的:

人为理解

	0	1	2	3
0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
2	a[2][0]	a[2][1]	a[2][2]	a[2][3]



数组a的类型

定义 - 名字 = 类型

例如: int a[3][4] - a = int [3][4]

二维数组的类型为 int [3][4]

二维数组大小

两种方法

- 元素个数 * 一个元素的大小 行数 * 列数 * sizeof(a[0][0])
- 数组总大小sizeof(a)

二维数组的初始化

二维数组初始化的形式

第一种定义 数据类型 数组名{行数}[列数] = {值};

- int a[3][3] = {1, 2, 3, 4, 5, 6, 7}; //正确
- int a[][3] = {1, 2, 3, 4, 5, 6, 7}; //正确
- a[3][] = {1, 2, 3, 4, 5, 6, 7}; //错误
- a[][] = {1, 2, 3, 4, 5, 6, 7}; //错误

第二种定义 数据类型 数组名{行数}[列数] = { (第一行的值), (第二行的值),};

- int a[3][3] = { {1, 2, 3}, {4, 5, 6}, {7} }; //正确
- { {1, 2}, {3, 4, 5}, {6, 7} } //错误

二维数组的输入和输出

对二维数组进行输入输出需要对行和列进行循环

例如 int a[3][2]

1.定义

- 定义数组
- 定义行数变量 i
- 定义列数变量 j

```
10 printf("please input %d data:\n", M * N);
11 for (i = 0; i < M; i++)
12 {
13     for (j = 0; j < N; j++)
14     {
15         scanf("%d", &a[i][j]);
16     }
17 }
```

2.循环输入进数组

```
19 for (i = 0; i < M; i++)
20 {
21     for (j = 0; j < N; j++)
22     {
23         printf("%d ", a[i][j]);
24     }
25     putchar('\n');
26 }
```

3.循环输出数组内容

总结

例: int a[3][2]

- 1.a的类型 int [3][2]
- 2.元素的表达式 a[0][0], a[0][1], a[1][0] ... a[2][1]
- 3.元素个数 行数 * 列数 sizeof(a) / sizeof(a[0][0])
- 4.数组大小 元素的个数 * 一个元素的大小 <====> 6 * sizeof(a[0][0]) <====> sizeof(a)
- 5.数组行数 sizeof(a) / sizeof(a[0])
- 5.数组的最后一个元素 a[行数 - 1][列数 - 1]
- 6.内存的存放方式 按行优先存放
- 7.定义二维数组，行数和列数哪个可以省略? 行数可以省略不写，系统会根据你定义的默认确定

练习

作业1:
定义 整型3行4列的二维数组，给它输入值。
例如:
1 2 3 10
4 5 6 20
7 8 9 30

int line, column;

要求用户输入行数和列数，若是查看对应的数据。(line和column从1开始计数)

解析

1.将数据输入对应的二维数组中

```
11 printf("please input %d data:\n", M * N);
12 for (i = 0; i < M; i++)
13 {
14     for (j = 0; j < N; j++)
15     {
16         scanf("%d", &a[i][j]);
17     }
18 }
```

2.输入行数列数，查看对应数据 (限制行数和列数的数值是正确的，不正确则重新开始输入)

```
20 loop:
21     printf("please enter the line and column you want to look:\n");
22     scanf("%d%d", &line, &column);
23     if(!(line >= 1 && line <= 3))
24     {
25         printf("input wrong line!\n");
26         goto loop;
27     }
28     if(!(column >= 1 && column <= 4))
29     {
30         printf("input wrong column!\n");
31         goto loop;
32     }
33 }
```

代码

```
1 #include <stdio.h>
2 #define M 3
3 #define N 4
4
5 int main(int argc, const char *argv[])
6 {
7     int a[M][N];
8     int i = 0, j = 0;
9     int line = 0, column = 0;
10
11     printf("please input %d data:\n", M * N);
12     for (i = 0; i < M; i++)
13     {
14         for (j = 0; j < N; j++)
15         {
16             scanf("%d", &a[i][j]);
17         }
18     }
19
20 loop:
21     printf("please enter the line and column you want to look:\n");
22     scanf("%d%d", &line, &column);
23     if(!(line >= 1 && line <= 3))
24     {
25         printf("input wrong line!\n");
26         goto loop;
27     }
28     if(!(column >= 1 && column <= 4))
29     {
30         printf("input wrong column!\n");
31         goto loop;
32     }
33
34     printf("the data: %d\n", a[line - 1][column - 1]);
35
36     return 0;
37 }
38 }
```

解析

1.设定二维数组，每一行的首位都为1，从第三行开始排列

```
12 int a[10][10] = {0};
13 int i = 0, j = 0;
14
15 //i = 0, 排好
16 //i = 1, 排好
17 //i = 2, a[2][1]
18 //i = 3, a[3][1] a[3][2]
19 //i = 4, a[4][1] a[4][2] a[4][3]
20 for (i = 0; i < 10; i++)
21 {
22     a[i][0] = a[i][1] = 1;
23     for (j = 1; j < i; j++) //从第三行开始排列
24     {
25         a[i][j] = a[i-1][j-1] + a[i-1][j];
26     }
27 }
```

2.以行列的形式输出二维数组

```
29 for (i = 0; i < 10; i++) //行数
30 {
31     for (j = 0; j <= i; j++) //每行输出的个数
32     {
33         printf("%-4d", a[i][j]);
34     }
35     putchar('\n');
36 }
```

代码

```
1 #include <stdio.h>
2 // 1
3 // 1 1
4 // 1 2 1
5 // 1 3 3 1
6 // 1 4 6 4 1
7 // .....
8
9 int main(int argc, const char *argv[])
10 {
11     int a[10][10] = {0};
12     int i = 0, j = 0;
13
14     //i = 0, 排好
15     //i = 1, 排好
16     //i = 2, a[2][1]
17     //i = 3, a[3][1] a[3][2]
18     //i = 4, a[4][1] a[4][2] a[4][3]
19     for (i = 0; i < 10; i++)
20     {
21         a[i][0] = a[i][1] = 1;
22         for (j = 1; j < i; j++) //从第三行开始排列
23         {
24             a[i][j] = a[i-1][j-1] + a[i-1][j];
25         }
26     }
27
28     for (i = 0; i < 10; i++) //行数
29     {
30         for (j = 0; j <= i; j++) //每行输出的个数
31         {
32             printf("%-4d", a[i][j]);
33         }
34         putchar('\n');
35     }
36
37     return 0;
38 }
```

作业2:
打印杨辉三角。定义int a[10][10];
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
.....

特点:
1. 每行第一个数和最后一个数都是1
2. 除了头和尾外，该数列里面的数 a[i][j] = a[i-1][j-1] + a[i-1][j] (i和j表示行数和列数)

第i行排列 ----- 3, 2

例: a[3][2] = a[2][1] + a[2][2]