第7章 数组

——一维数组下标越界问题分析

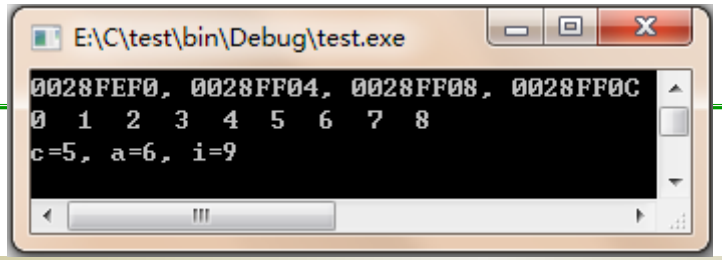
数组元素的访问

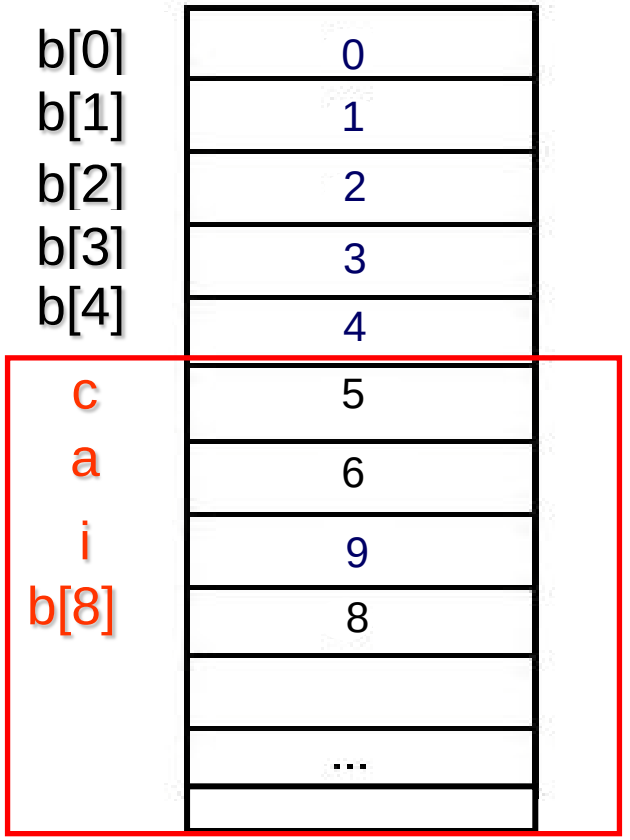
- 访问数组元素时，下标越界是大忌！
  - 编译器通常不检查下标越界，导致程序运行时错误
  - 下标越界，将访问数组以外的空间
  - 那里的数据是未知的，不受我们掌控，可能带来严重后果
- 后果有多严重呢？

一维数组元素的越界访问

当下标值小于0或超过数组长度时会出现什么情况？

```
#include <stdio.h>
int main()
{
    int i, a = 1, c = 2, b[5] = {0};
    printf("%p, %p, %p, %p\n", b, &c, &a, &i);
    for (i=0; i<=8; i++)   //越界访问
    {
        b[i] = i;
        printf("%d  ", b[i]);
    }
    printf("\nc=%d, a=%d, i=%d\n", c, a, i);
    return 0;
}
```

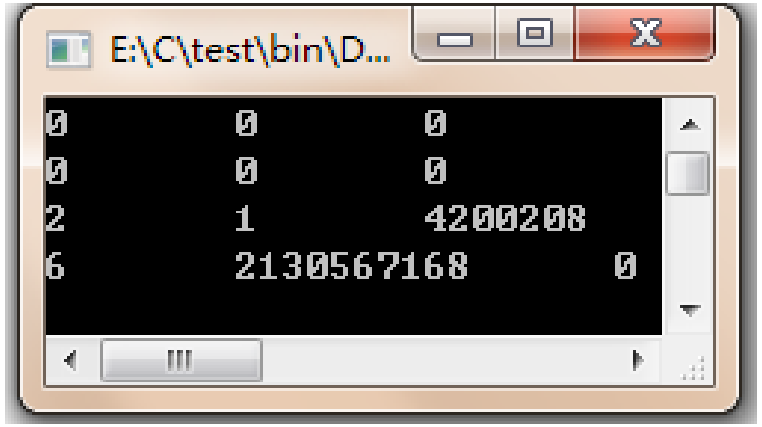




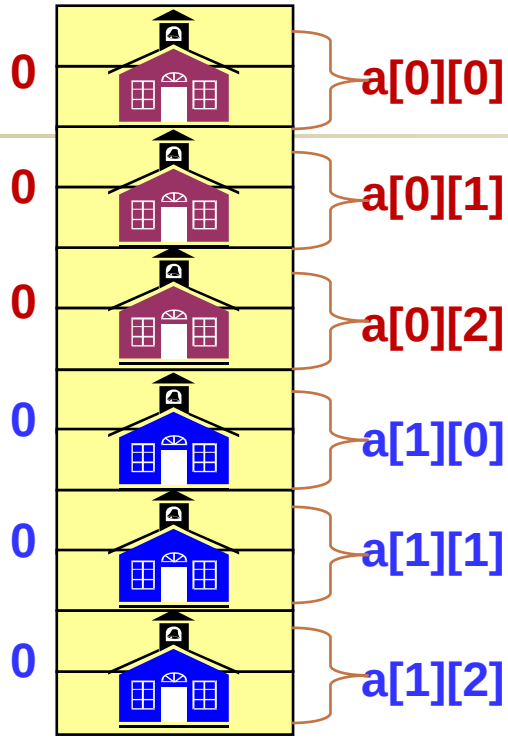
二维数组元素的越界访问

```
#include <stdio.h>
int main()
{
    int i, j;
    int a[2][3] = {0};
    a[3][0] = 6;            //越界访问
    for (i=0; i<4; i++) //越界访问
    {
        for (j=0; j<3; j++)
        {
            printf("%d\t", a[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

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



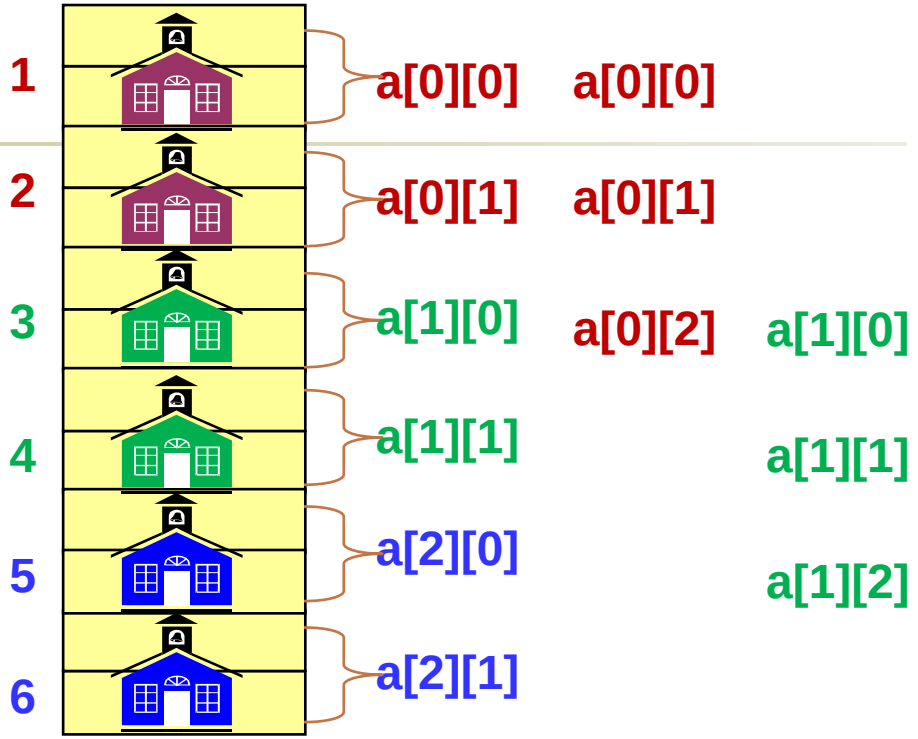
```c
#include <stdio.h>
int main()
{
    int i, j;
    int a[2][3] = {0};
    a[1][0] = 4;
    for (i=0; i<2; i++)
    {
        for (j=0; j<3; j++)
        {
            printf("%d\t", a[i][j]);
        }
        printf("\n");
    }
    a[0][3] = 5;
    for (i=0; i<2; i++)
    {
        for (j=0; j<3; j++)
        {
            printf("%d\t", a[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```




```
0	0	0
4	0	0
0	0	0
5	0	0
```

a[0][3]和a[1][0]指的是同一元素，不检查下标越界，a[0][3]的写法虽然合法，但隐患严重

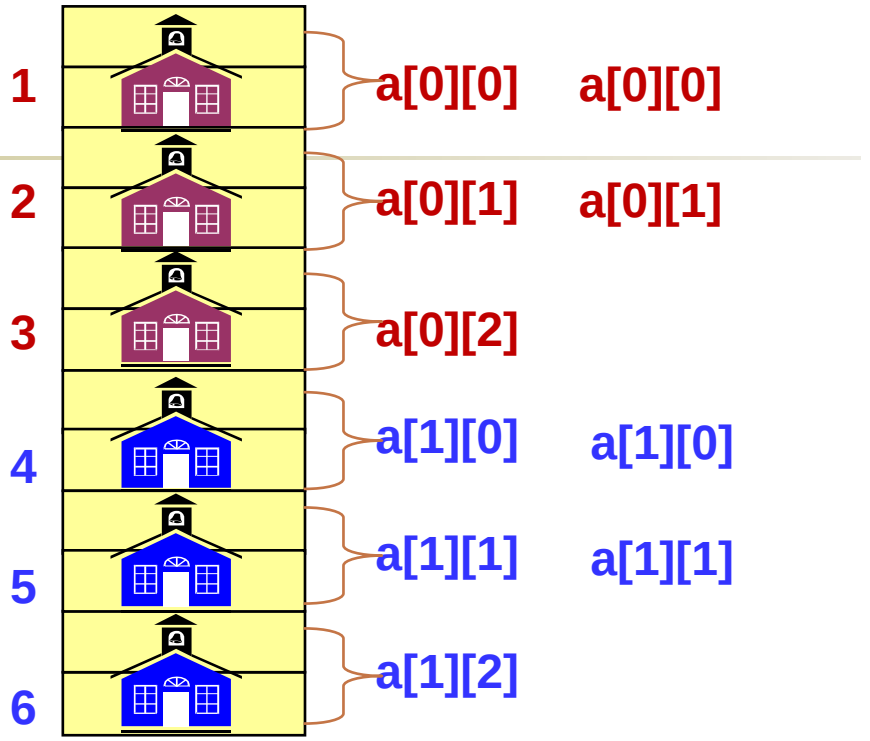
```
#include <stdio.h>
int main()
{
    int i, j;
    int a[3][2] = {1,2,3,4,5,6};
    for (i=0; i<3; i++)
    {
        for (j=0; j<2; j++)
        {
            printf("%d\t", a[i][j]);
        }
        printf("\n");
    }
    for (i=0; i<2; i++)
    {
        for (j=0; j<3; j++)
        {
            printf("%d\t", a[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```




```
1   2
3   4
5   6

1   2   3
3   4   5
```

```
#include <stdio.h>
int main()
{
    int i, j;
    int a[2][3] = {1,2,3,4,5,6};
    for (i=0; i<2; i++)
    {
        for (j=0; j<3; j++)
        {
            printf("%d\t", a[i][j]);
        }
        printf("\n");
    }
    for (i=0; i<3; i++)  //越界访问
    {
        for (j=0; j<2; j++)
        {
            printf("%d\t", a[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```




```
1   2   3
4   5   6

1   2
4   5
2   1
```

小结

- 使用数组的基本原则
  - 永远清楚每个数组有多大，永远不要让下标越界
  - 字符数组永远留意'\0'