Jiangxi University of Science and Technology

Chapter 8 Arrays

- lecture0801 One-Dimensional Arrays

Objectives

- 8.1 One-Dimensional Arrays
- 8.2 Array Initialization
- 8.3 Arrays as Function Arguments
- 8.4 Case Study: Computing Averages and Standard Deviations
- 8.5 Two-Dimensional Arrays
- 8.6 Common Programming and Compiler Errors

8.1 One-Dimensional Arrays

- **Atomic variable**原子变量: variable whose value cannot be further subdivided into a built-in data type 内置数据类型
  - Also called a **scalar variable** 标量变量
- **Data structure (aggregate data type** 聚合数据类型**)**: data type with two main characteristics
  1. Its values can **be decomposed into** individual data elements, each of which is either atomic or another data structure
  2. It provides **an access scheme** for locating individual data elements within the data structure

8.1 One-Dimensional Arrays

- One of the simplest data structures, called an **array** 数组, is used to store and process a set of values, all of the same data type, that forms a logical group

Grades	Codes	Prices
98	x	10.96
87	a	6.43
92	m	2.58
79	n	.86
85		12.27
		6.39

Figure 8.1 Three lists of items

8.1 One-Dimensional Arrays

- A **one-dimensional array** 一维数组, also called a **single-dimensional array** and a **single-subscript array**单下标数组, is a list of values of the same data type that is stored using a single group name

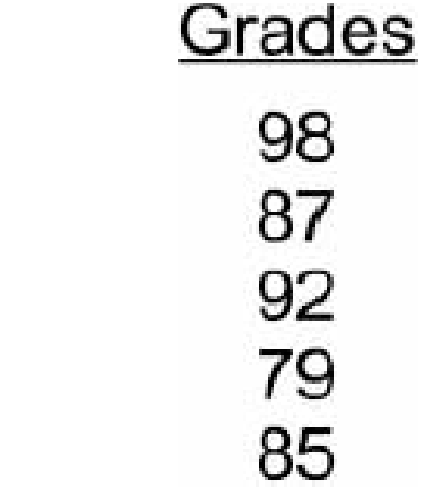


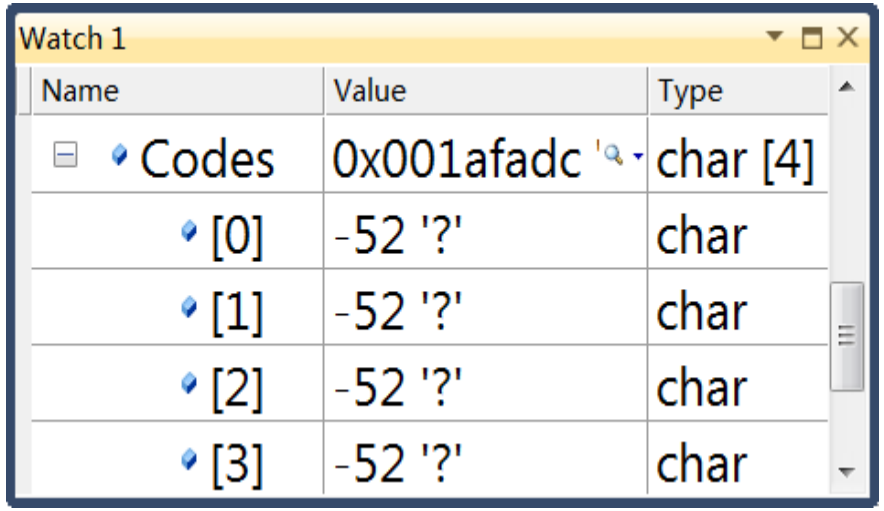
Figure 8.2
A list of grades

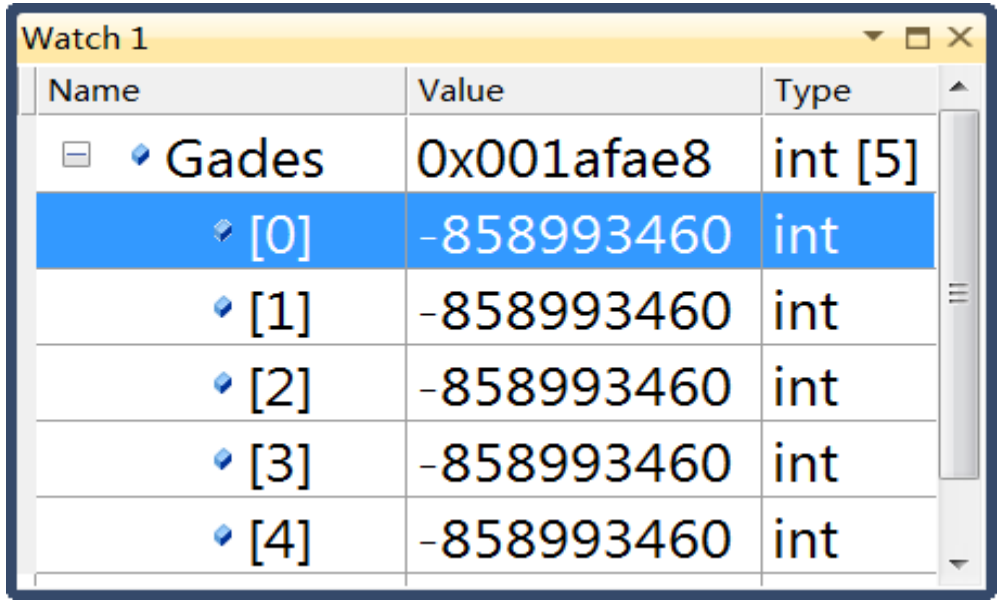
8.1 One-Dimensional Arrays

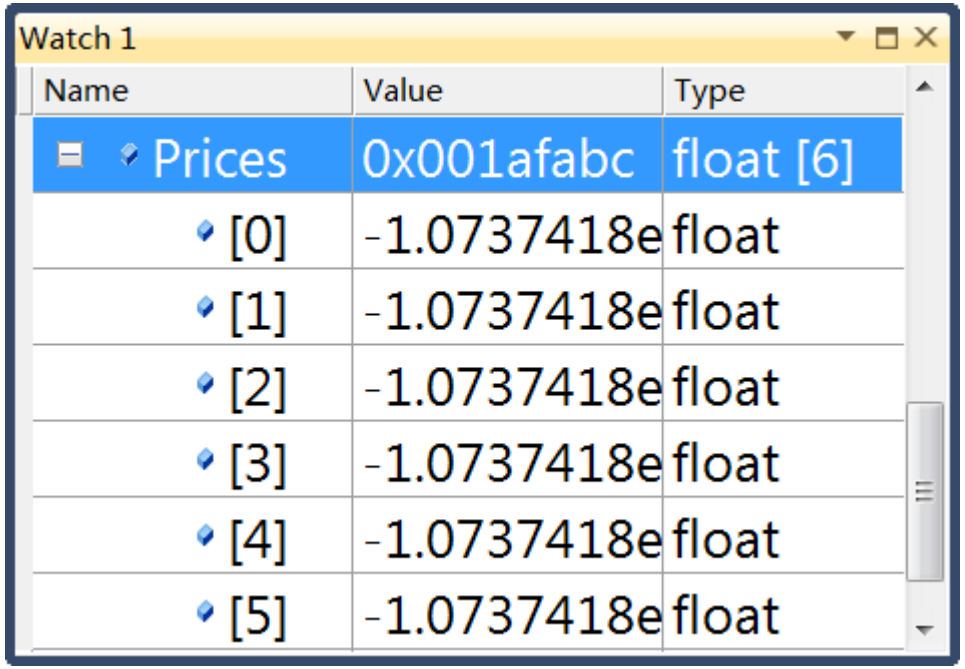
- **Create a one-dimensional array**:
  - **#define NUMELS 5**
  - **int grades[NUMELS];**
    - ***starting index value*** for all arrays is **0**
    - Each item in an array is called an ***element*** of the array
    - Any element can be accessed by giving the ***name*** of the array and the ***element's index***

8.1 One-Dimensional Arrays

- int Gades[5];
- char Codes[4];
- float Prices[6];







8.1 One-Dimensional Arrays

- Subscripted variables 下标变量 can be used anywhere scalar variables are valid
  - grades[**0**] = 98;
  - grades[**1**] = grades[**0**] - 11;
- **Any expression** that evaluates **an integer** may be used as a subscript
  - #define NUMELS 5
  - total = 0; /* initialize total to zero */
  - for (i = 0; i < NUMELS; i++)
  - total = total + grades[**i**]; /* add a grade */

8.1 One-Dimensional Arrays

- Input and Output of Array Values
  - Individual array elements can be assigned values using individual assignment statements or, interactively, using the ***scanf()*** function

```
1. #define NUMELS 5
2. for(i = 0; i < NUMELS; i++)
3. {
4.   printf("Enter a grade: ");
5.   scanf("%d", &grades[i]);
6. }
```

Be careful: C does not check the value of the index being used(called a bounds check)

8.1 One-Dimensional Arrays

➢ Program 8.1 Input of Array Values

```
1.  #include <stdio.h>
2.  int main(){
3.    #define MAXGRADES 5
4.    int grades[MAXGRADES];
5.   /* input the grades */
6.    for (int i = 0; i < MAXGRADES; i++)
7.    {
8.      printf("Enter a grade: ");
9.      scanf("%d", &grades[i]);
10.   }
11.           /* display the grades */
12.           for (int i = 0; i < MAXGRADES; i++)
13.                   printf("grades[%d] is %d\n", i, grades[i]);
14.           return 0;
15. }
```

```
Enter a grade: 85
Enter a grade: 90
Enter a grade: 78
Enter a grade: 75
Enter a grade: 92
```

```
grades[0] is 85
grades[1] is 90
grades[2] is 78
grades[3] is 75
grades[4] is 92
```

Name	Value	Type
grades	0x001ffdc0	int [5]
[0]	85	int
[1]	90	int
[2]	78	int
[3]	75	int
[4]	92	int

8.1 One-Dimensional Arrays

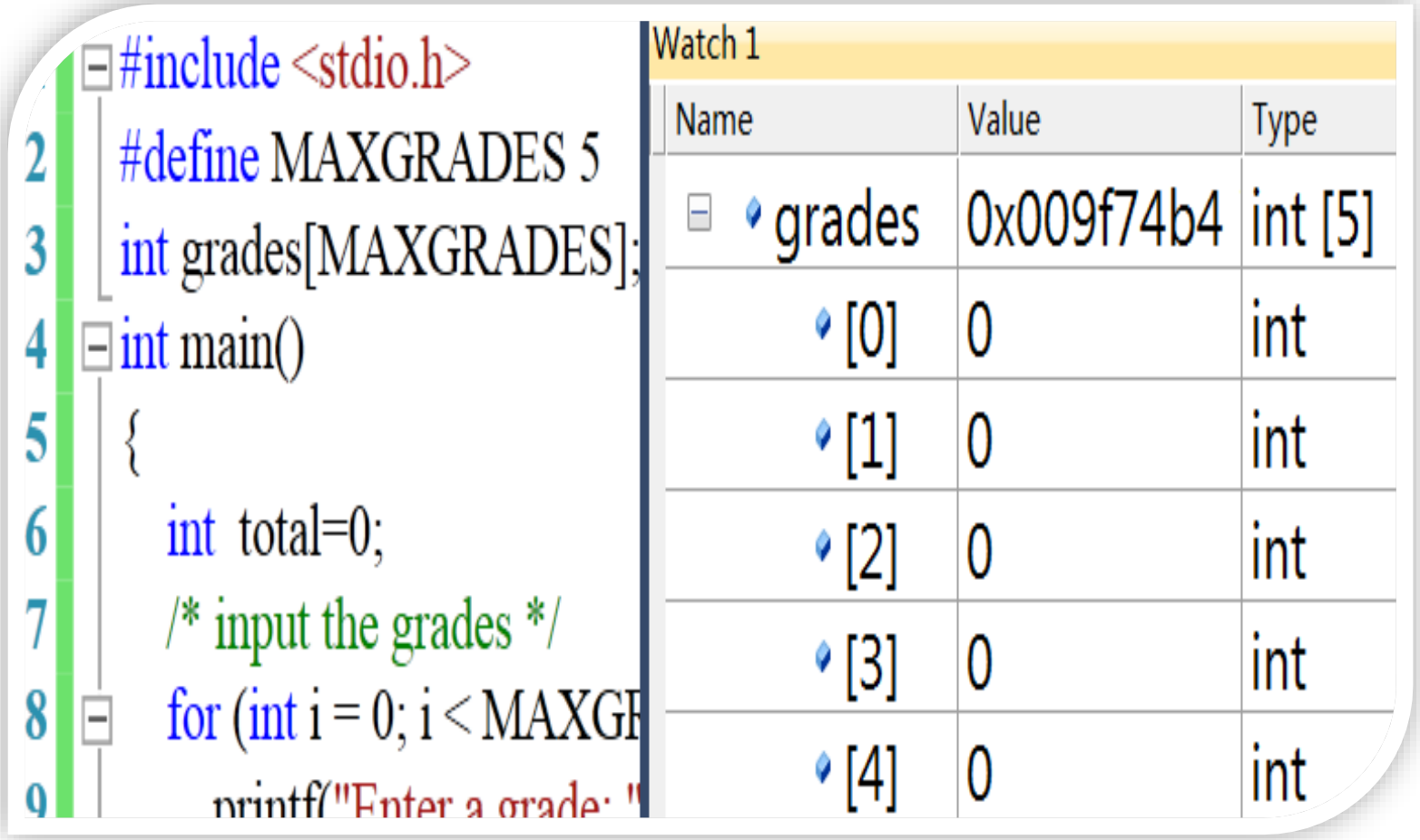
➢ Program 8.2 Input of Array Values

```
1.  #include <stdio.h>
2.  int main() {
3.          #define MAXGRADES 5
4.          int grades[MAXGRADES], total=0;
5.          /* input the grades */
6.          for(int i=0;i<MAXGRADES;i++){
7.          printf("Enter a grade: ");
8.          scanf("%d", &grades[i]);
9.          }
11. /*display and total the grades */
12.         printf("\nThe total of the grades ");
13.         for(int i=0;i<MAXGRADES;i++){
14.                 printf("%d ", grades[i]);
15.                 total += grades[i];
16.         }
17.         printf("is %d\n", total); //display the total
18.         return 0;
19. }
```

Statement is outside of the second for loop; total is displayed only once, after all values have been added

8.2 Array Initialization

- Initialization of **global and static arrays**
  - The individual elements of all global and static arrays (local or global) are, **by default**, set to 0 at compilation time



```
#include <stdio.h>
#define MAXGRADES 5
int grades[MAXGRADES];
int main()
{
    int  total=0;
    /* input the grades */
    for (int i = 0; i < MAXGR
        printf("Enter a grade: "
```

Watch 1

Name	Value	Type
grades	0x009f74b4	int [5]
[0]	0	int
[1]	0	int
[2]	0	int
[3]	0	int
[4]	0	int

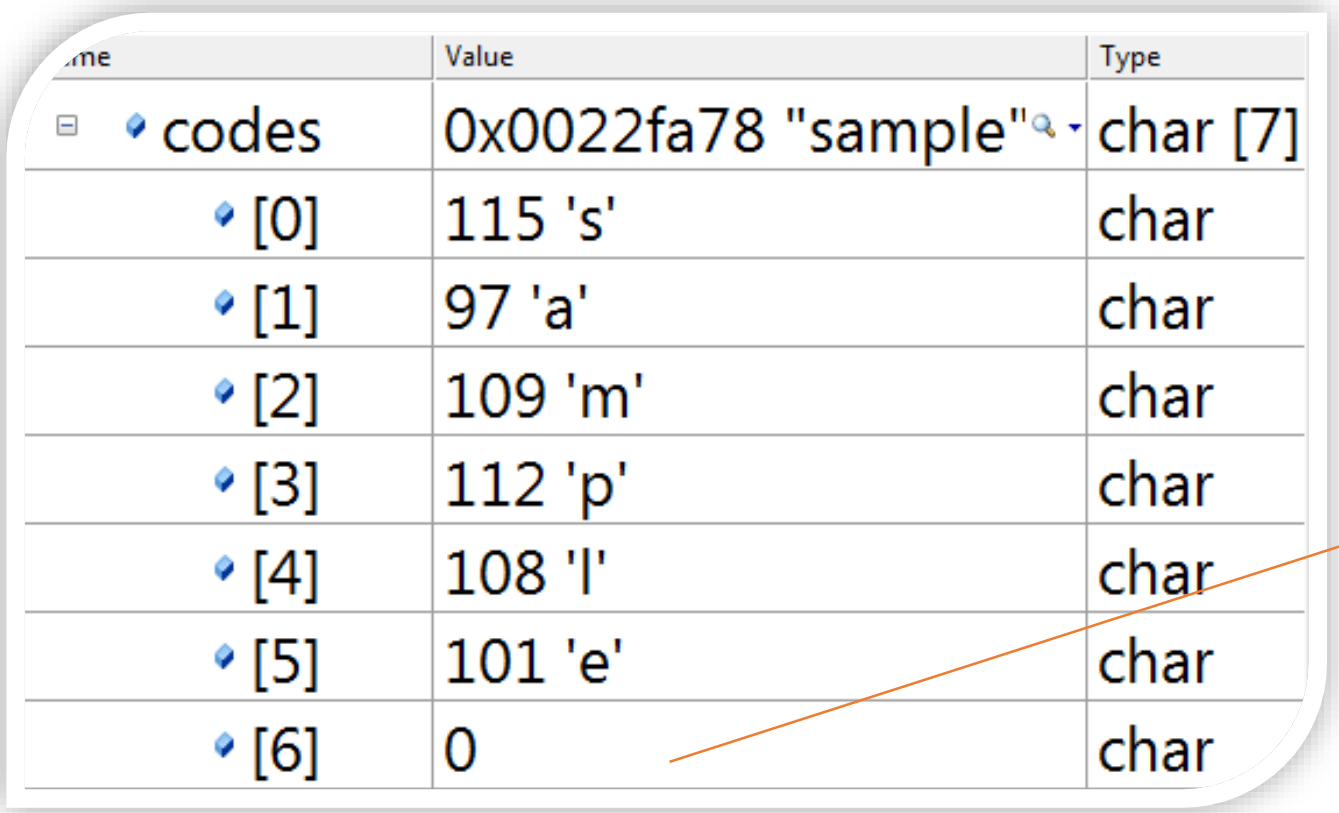
8.2 Array Initialization

- Initialization of **auto local arrays**
  - The values within auto local arrays are undefined
  - Examples of initializations:
    - `int grades[5] = {98, 87, 92, 79, 85};`
    - `double length[7] = {8.8, 6.4, 4.9, 11.2};`
    - `char codes[6] = {'s', 'a', 'm', 'p', 'l', 'e'};`
    - `char codes[] = {'s', 'a', 'm', 'p', 'l', 'e'};`
    - `char codes[] = "sample"; /* size is 7 */`
    - `int grades[5] = {98};`

如果对部分元素进行初始化，
其余元素也会自动设为0

8.2 Array Initialization

- The NULL character 空字符
  - The **NULL** character, which is the **escape sequence \0**, is **automatically** appended to all strings by the C compiler



me	Value	Type
codes	0x0022fa78 "sample"	char [7]
[0]	115 's'	char
[1]	97 'a'	char
[2]	109 'm'	char
[3]	112 'p'	char
[4]	108 'l'	char
[5]	101 'e'	char
[6]	0	char

字符串结尾符

8.2 Array Initialization

➢ Program 8.3 The maximum value 擂台法求最大值

```
1.  #include <stdio.h>
2.  int main(){
3.          #define MAXELS 5
4.          int nums[MAXELS] = {2, 18, 1, 27, 16};
5.          int max = nums[0];
6.          for (int i = 1; i < MAXELS; i++)
7.                  if (max < nums[i])
8.                          max = nums[i];
9.          printf("The maximum value is %d\n", max);
10.         return 0;
11. }
```

Reference

- https://www.codesdope.com/blog/article/int-main-vs-void-main-vs-int-mainvoid-in-c-c/