Structured Programming

-Array

Donglong Chen

With Thanks to Dr. Xin Feng and Dr. Haipeng Guo

Outline

- The concept of array
- One-dimension array
- Multi-dimension array

Array

If we are required to write a program to calculate the average of 50 grades for a class of students, how can we write that program?

```
int grade1, grade2, grade3, …., grade50;
```

50 variables!

Array

- An array (数组) offers a solution to this problem
- Array is a derived data type
  - It itself is not a type
  - Every element in the array has same type
  - E.g., instead of the declaration like

```
int grade1, grade2, grade3, …., grade50;
```

  - We can declare

```
        int grade[50];
```

Array

- If we declare

```
int grade[50];
```

- We can refer to each element in the array with index.

E.g.

```
int grade[50];

grade[0] = 10;
grade[1] = 20;
```

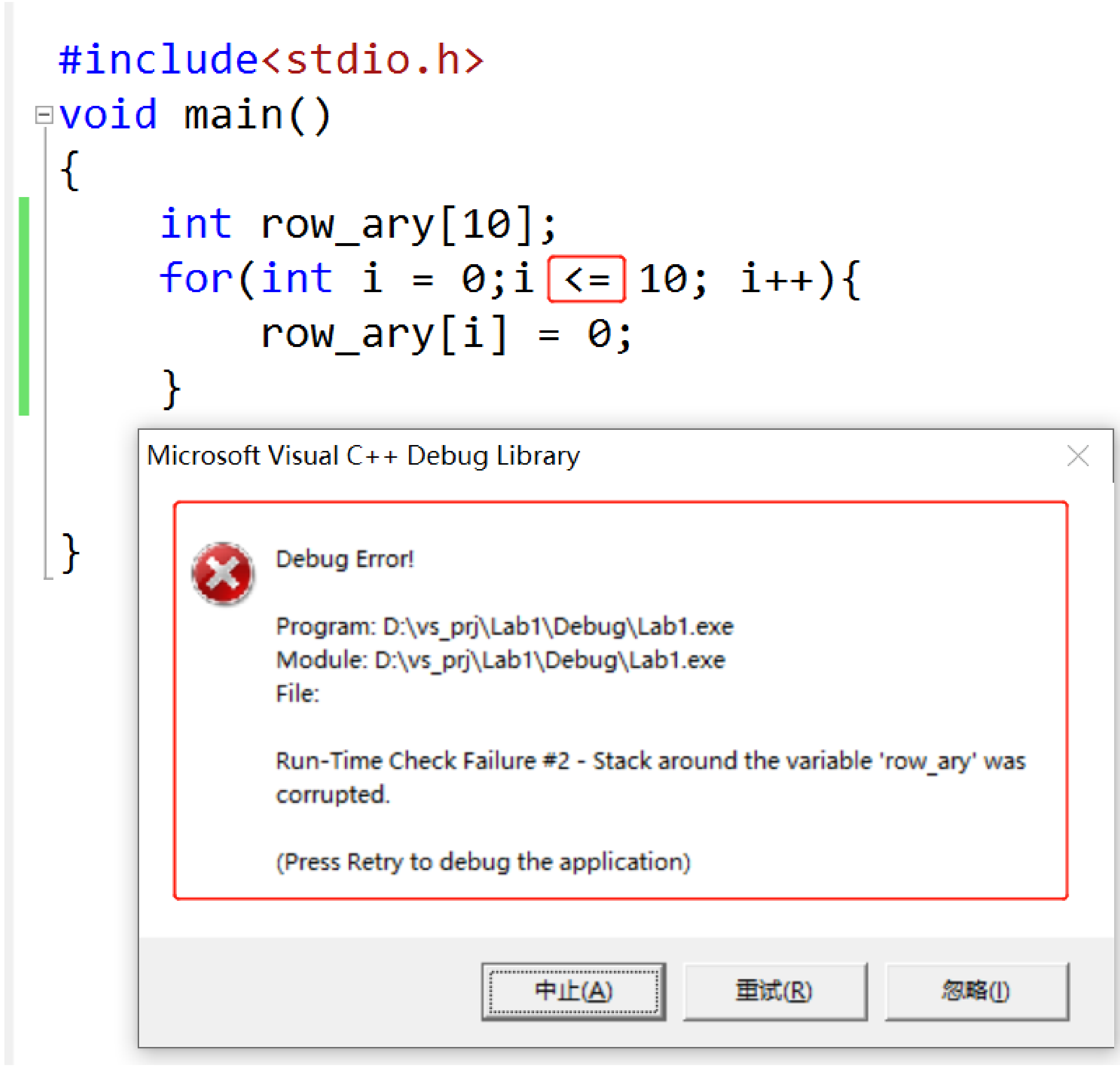
```
int grade[50];
int i;

for (i = 0; i < 50; i++)
   grade[i] = 100;
```

- Attention
  - The first element in an array has the index 0
  - The last element in an array has the index 49 (in this example), i.e., grade[50] is not allowed.

Bounds of An Array

- The size of an array is the total number of elements in the array
- Remember that the array index is from 0 to size – 1.
- If an index exceeds the size – 1, on Unix systems this leads to a memory *segmentation fault*.
- Writing over the bounds of an array is a common source of error.

Error - Bounds of An Array



```
#include<stdio.h>
void main()
{
    int row_ary[10];
    for(int i = 0;i <= 10; i++){
        row_ary[i] = 0;
    }

}
```

Microsoft Visual C++ Debug Library

Debug Error!

Program: D:\vs_prj\Lab1\Debug\Lab1.exe
Module: D:\vs_prj\Lab1\Debug\Lab1.exe
File:

Run-Time Check Failure #2 - Stack around the variable 'row_ary' was corrupted.

(Press Retry to debug the application)

Initializing An Array

- Three ways to initialize an array

```
int grade[4];

grade[0] = 10;
grade[1] = 20;
grade[2] = 30;
grade[3] = 40;
```

```
int grade[4] = {10, 20, 30, 40 };
```

```
int grade[] = {10, 20, 30, 40 };
```

An Example

```
int main() {
   int a[5];
   int i;
   for (i = 0; i < 5; i++)
     a[i] = i;
   for (i = 0; i < 5; i++)
     printf("a[%d] = %d\n", i, a[i]);
   return 0;
   }
```

1. What is the output of this program?
2. Can we change i < 5 to i <= 5?

An Example

```
int main() {
  int grade[5]= {1, 2, 3, 4, 5};
  int i, sum;
  float average;
  for (i = 0; i <= 5; i++)
    sum = sum + grade[i];
  average = (float)sum / i;
  return 0;
}
```

1. What is this program supposed to do?
2. Are there any problems in this program?

Multi-Dimensional Arrays

- Arrays in C programs can have virtually as many dimensions as you want.
- Declaration is accomplished by adding additional subscripts when it is defined.
- E.g. `int table[4][3];`
  - defines a two-dimensional array

Multi-Dimensional Arrays

- E.g. `int table[4][3];`
  - We can understand this as
    - table is an array of table[0], table[1], table[2], and table[3]

`table[0]`	`table[0][0]`	`table[0][1]`	`table[0][2]`
`table[1]`	`table[1][0]`	`table[1][1]`	`table[1][2]`
`table[2]`	`table[2][0]`	`table[2][1]`	`table[2][2]`
`table[3]`	`table[3][0]`	`table[3][1]`	`table[3][2]`

Initializing An Array

- Three ways to initialize a multi-dimensional array

```
int grade[2][3];

grade[0][0] = 10;
grade[0][1] = 20;
grade[0][2] = 30;
grade[1][0] = 40;
grade[1][1] = 50;
grade[1][2] = 60;
```

```
int grade[2][3] = {10, 20, 30, 40, 50, 60};
```

```
int grade[2][3] = {{10, 20, 30}, {40, 50, 60}};
```

An Example

```
int main ()
{
   int random[2][2];
   int i, j;
   for (i = 0; i < 2; i++)
       for (j = 0; j < 2; j++)
          random [i][j] = rand()%2;
    for (i = 0; i < 2; i++){
       for (j = 0; j < 2; j++)
           printf ("%c " , random[i][j] ? 'x' : 'o');
       printf("\n");
   }
    return 0;
}
```

String

- A string is an array of chars
- E.g., `char s[10];`
  - s is a string which can store at most 10 characters

An Example

```
#include<stdio.h>
#include <string.h>
int main ()  {
  char word[20];

  word[0] = 'H';
  word[1] = 'e';
  word[2] = 'l';
  word[3] = 'l';
  word[4] = 'o';
  word[5] = '\0';

  printf("The word is %s\n", word );
  printf("The length of string is %d", strlen(word));
  return 0;
}
```

Exercises

- What is the results of the example in the previous slide if (`word[5] = '\0';`) is deleted? Why? Search from internet and find out the answer.

- What is the value of `a[2]` if we have the following initialization of an array
  - `int a[5] = {2, 4, 6, 8, 10}`

- What is the value of `a[2][1]` if we have the following initialization of an array
  - `int a[2][3] = {2, 4, 6, 8, 10, 12}`

Reading recommendation:
https://stackoverflow.com/questions/14461695/what-does-0-stand-for/14461711
https://blog.csdn.net/supreme42/article/details/7300451

Summary

- Array can be used to store a set of data with same data type.
- Index of an array should not exceed the upper limit.