Foundations of C Programming (Structured Programming)
- Array

Outline

- The concept of array
- One-dimension array (一维数组)
- Multi-dimension array (多维数组)

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] = 10;
```

```
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.
- Writing over the bounds of an array is a common source of error. It might change the values of other variables.

Attention: Always assure that the index is within the bounds.

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?

Array Size

```
int a[5];
```
√

```
int n = 5;
int a[n];
```
√

```
int n;
scanf("%d", &n);
int a[n];
```
√

```
int n;
int a[n];
scanf("%d", &n);
```
X

Attention: Array size must be determined before the array is declared.

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 (二维数组)

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(初始化) a multi-dimensional array (多维数组)

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

String

- A string (字符串) is an array of chars
  - E.g., `char s[10];`
    - `s` is a string which can store at most 10 characters
- A string must end at '\0'.

An Example to String Initialization

```
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 );
   //%s: output a string ending at '\0'
   return 0;
}
```


Attention: '\0' must be added

Another Way to Initialize String

```
int main (){
   char word[20]= "Hello";

  printf("The word is %s\n", word );
   return 0;
}
```

√

```
int main (){
   char word[20];

    word = "Hello"; //illegal assignment
   printf("The word is %s\n", word );
   return 0;
}
```

X

Attention: '\0' had been automatically added to the end of string in the first example.

Class Exercise

```
int main (){
   char word[20]= "Hello";

  printf("The second char is %c \n", word[1]);
  return 0;
}
```

Output?

Class Exercises

- 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}`

An Example： Sorting Data in An Array

- Sometimes we need to sort an array in descending order (降序) or ascending order(升序)
- For example
  - An array: 10 30 20 40
  - Descending order: 40 30 20 10
  - Ascending order: 10 20 30 40
- Program in the next page
  - implements decedent ordering
- More interesting algorithms can be used in sorting
  - Explore after the class

```c
#include <stdio.h>

int main(){
  int n, i, j;

 printf("Input the total number of integers: \n");
  scanf("%d", &n);
  int a[n];

  printf("Please input the integers to sort: \n");
  for (i = 0; i <= n-1; i++)
       scanf("%d", &a[i]);

  for (i = 0; i <= n-2; i++)
    for (j = i+1; j <= n-1; j++){
      if (a[i]<a[j]){
         int temp;
         temp = a[j];
         a[j] = a[i];
         a[i] = temp;
      }
    }

  printf("Sorted integers in descendent order\n");
  for (i = 0; i <= n - 1; i++)
    printf(" %d ", a[i]);
  return 0;
}
```

```
n = 4;

a[0]    a[1]    a[2]      a[3]
10       30      20        40
-------------------------
i = 0
j: 1 ~ 3

j=1: 30 10 20 40
j=2: 30 10 20 40
j=3: 40 10 20 30
-------------------------
i = 1
j: 2 ~ 3

j=2: 40 20 10 30
j=3: 40 30 10 20
-------------------------
i = 2
j: 3 ~ 3

j=3: 40 30 20 10

Descending
```

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.
- Array can be used to store a string.