Username : class-1

Password : dctq

Please do not shut down the computer

Check VS 2019 whether can use

We will start our course in 18:30

we will start demonstrate the exercises at 19:15.

Do not use scanf_s

Please make sure the TA has recorded your exercise score here before leaving.

If you're ready for demonstration, go to the following link: nightynight.xyz/case/

Array and String

Introduction to Computers and Programming

Outline

- Array
- String

Array

Array – when to use

Q. If we have 100 students in this class, how to store their score?

Before

```
int student1 = 90;
int student2 = 98;
//     ….
int student99 = 77;
int student100 = 89;
```

Now with Array

```
int students[100] = {90, 98, ..., 77, 89};
```

Array - declare a 1D array

Format

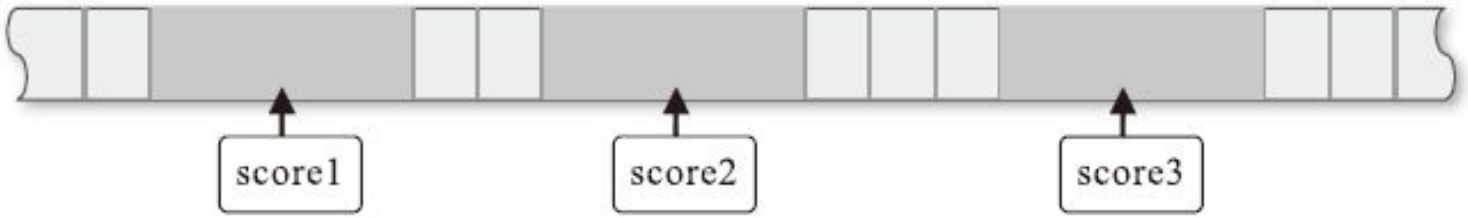
```
datatype arrayname[n_elements] = {elements};
```

Example

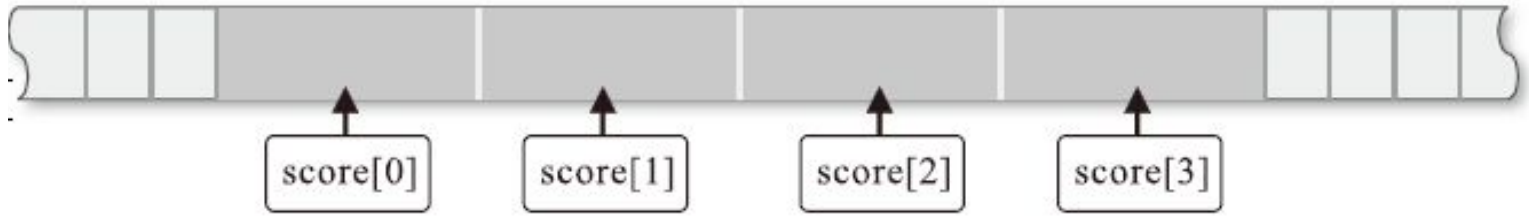
```
// declaring an "integer" array named "a" with "5" elements
int a[5] = { 20, 30, 40, 50, 60};
// a[0]=20, a[1]=30, a[2]=40, a[3]=50, a[4]=60

// declaring a "float" array named "weight" with "3" elements
float weight[3] = { 52.5, 60.8, 70.5};
// weight[0]=52.5, weight[1]=60.8, weight[2]=70.5
```

| Array - storage of 1D array

- Individually declared variables are stored in non-continuous space. The efficiency of operation with these data is low.



- The space allocated to an array is continuous.



Array - operator: sizeof()

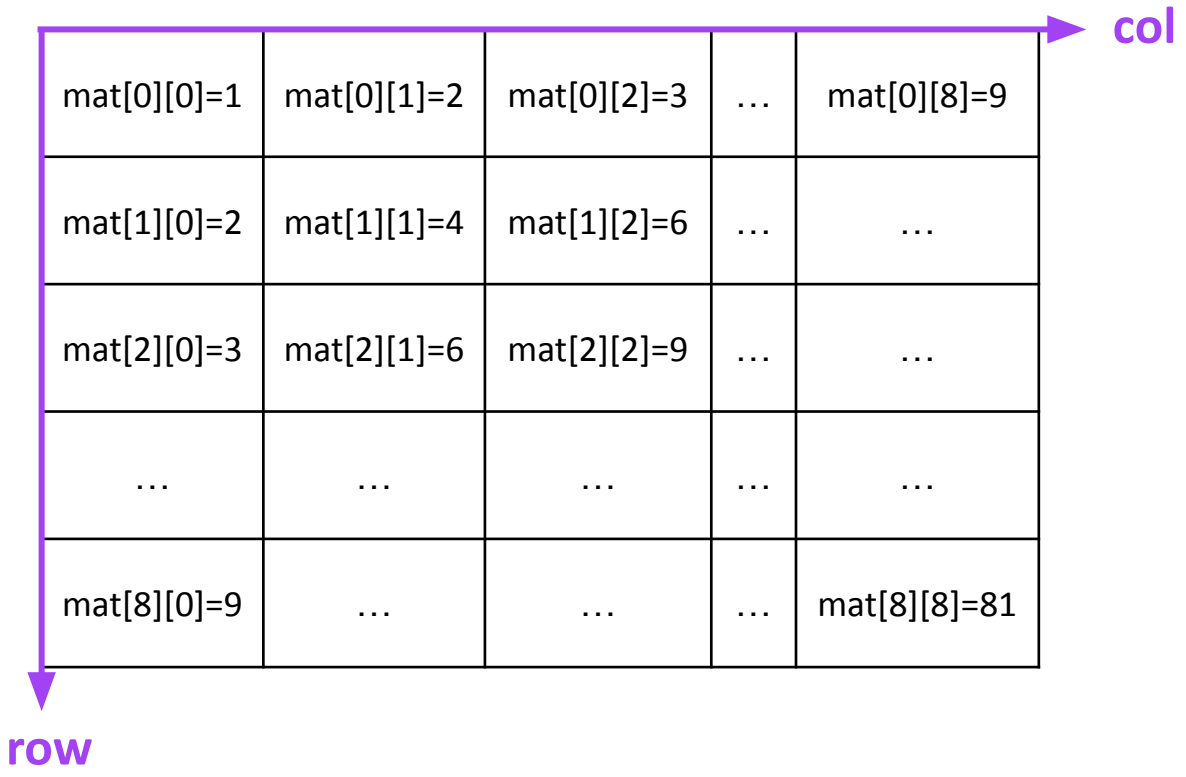
```c
int a[5] = { 20, 30, 40, 50, 60};
// a[0]=20, a[1]=30, a[2]=40, a[3]=50, a[4]=60

for (int i = 0; i < 5; i++)
{
  printf("a[%d] = %d, pos = %p \n", i, a[i], &(a[i]));
}

int size = sizeof(a);

printf("Total size: %d \n", size);
printf("Element size: %d \n", sizeof(a[0]));
printf("There are %d elements in the array \n", size/sizeof(a[0]));
```

```
a[0] = 20, pos = 000000FAB56FF5C8
a[1] = 30, pos = 000000FAB56FF5CC
a[2] = 40, pos = 000000FAB56FF5D0
a[3] = 50, pos = 000000FAB56FF5D4
a[4] = 60, pos = 000000FAB56FF5D8
Total size: 20
Element size: 4
There are 5 elements in the array
```

| Array - declare a 2D array

Example

```
int mat[9][9]; // mat[row][col]

for (int i = 0; i < 9; i++){
    for (int j = 0; j < 9; j++){
        mat[i][j] = (i+1)*(j+1);
    }
}
```



mat[0][0]=1	mat[0][1]=2	mat[0][2]=3	…	mat[0][8]=9
mat[1][0]=2	mat[1][1]=4	mat[1][2]=6	…	…
mat[2][0]=3	mat[2][1]=6	mat[2][2]=9	…	…
…	…	…	…	…
mat[8][0]=9	…	…	…	mat[8][8]=81

String

| String - an array of characters

- String is just an array of characters
- '\0' : terminating character (null character)
- 'c' is not the same as ''c''
- 'c' is a character, and ''c'' is a string (character array with elements 'c' and '\0')

Dec	Hx	Oct	Char
0	0	000	NUL (null)
1	1	001	SOH (start of heading)
2	2	002	STX (start of text)
3	3	003	ETX (end of text)
4	4	004	EOT (end of transmission)
5	5	005	ENQ (enquiry)

ASCII Code Example

| String - 2D character array

```
char sentences[3][200] = {"I love to Code!",
                          "What language is your favorite?",
                          "C and Cpp are the best!"};
```

[0][0] ‘I’	[0][1] ‘ ’	[0][2] ‘l’	[0][3] ‘o’	[0][4] ‘v’	[0][5] ‘e’	[0][6] ‘ ’	[0][7] ‘t’	[0][8] ‘o’	[0][9] ‘ ’	[0][10] ‘c’	[0][11] ‘o’	[0][12] ‘d’	[0][13] ‘e’	[0][14] ‘!’	[0][15] **‘\0’**	…	[0][199]
[1][0] ‘W’	[1][1] ‘h’	[1][2] ‘a’	[1][3] ‘t’	[1][4] ‘ ’	[1][5] ‘l’	[1][6] ‘a’	[1][7] ‘n’	[1][8] ‘g’	[1][9] ‘u’	[1][10] ‘a’	[1][11] ‘g’	[1][12] ‘e’	[1][13] ‘ ’	[1][14] ‘i’	[1][15] ‘s’	…	[1][199]
[2][0] ‘C’	[2][1] ‘ ’	[2][2] ‘a’	[2][3] ‘n’	[2][4] ‘d’	[2][5] ‘ ’	[2][6] ‘C’	[2][7] ‘p’	[2][8] ‘p’	[2][9] ‘ ’	[2][10] ‘a’	[2][11] ‘r’	[2][12] ‘e’	[2][13] ‘ ’	[2][14] ‘t’	[2][15] ‘h’	…	[2][199]

String - sizeof

Example

```
char s[] = "apple";

printf("%s\n", s);
s[0] = 'z';
printf("%s\n", s);

printf("Total size: %d\n", sizeof(s));
printf("Total size: %d\n", sizeof(s[0]));
printf("There are %d elements in the array\n",
     sizeof(s) / sizeof(s[0]));
```

s[0] = 'a' s[1] = 'p' s[2] = 'p' s[3] = 'l' s[4] = 'e' s[5] = '\0'

```
apple
zpple
Total size: 6
Total size: 1
There are 6 elements in the array
```

String – strlen()

Example

s[0] = 'a' s[1] = 'p' s[2] = 'p' s[3] = 'l' s[4] = 'e' s[5] = '\0'

```c
#include <string.h>
char s[] = "apple";

printf("%s\n", s);
s[0] = 'z';
printf("%s\n", s);

printf("Total size: %d\n", sizeof(s));
printf("Total size: %d\n", sizeof(s[0]));
printf("There are %d elements in the array\n", strlen(s));
```

```
apple
zpple
Total size: 6
Total size: 1
There are 5 elements in the array
```

String - input a character or string

Example1

```
printf("Input char: ");
char a;
a = getchar();
printf("Your char: %c\n", a);
```

```
Input string: a
Your string: a
```

Example2

```
printf("Input string: ");
char a[5];
gets_s(a);
printf("Your string: %s\n", a);
```

```
Input string: qwer
Your string: qwer
```

Exercise 1 – Search Characters

Input any string form and find the occurrences of a given character in the string.

(The length of the string won't be more than 30.)

Input

Enter any string: I love watermelon.

Enter character to be searched: e

Output

'{char}' is found at index {position}.

```
Enter any string: I love watermelon
Enter character to be searched: e
'e' is found at index 5.
'e' is found at index 10.
'e' is found at index 13.
```

| Exercise 2 - Matrix Multiplication

First, input m, n, p, which is the dimension of two matrices, A_{m*n}, B_{n*p}. Then, input the two matrices respectively with row-major. Finally, output the multiplication of A and B.

Input

m, n, p (max dimension is 10), two matrices A, B

Output

Print out the multiplication result of two matrices A and B.

It should be a matrix with dimension: m*p.

```
6 6 6

1 2 3 4 5 6
7 8 9 10 11 12
13 14 15 16 17 18
19 20 21 22 23 24
25 26 27 28 29 30
31 32 33 34 35 36

37 38 39 40 41 42
43 44 45 46 47 48
49 50 51 52 53 54
55 56 57 58 59 60
61 62 63 64 65 66
67 68 69 70 71 72
1197 1218 1239 1260 1281 1302
3069 3126 3183 3240 3297 3354
4941 5034 5127 5220 5313 5406
6813 6942 7071 7200 7329 7458
8685 8850 9015 9180 9345 9510
10557 10758 10959 11160 11361 11562
```