Lecture #04

Array and Pointer(1)

SE271 Object-Oriented Programming (2020)

Yeseong Kim

Original slides from Prof. Shin at DGIST

Short Notice

- The first homework will be released today!
  - Due: Oct 5 (Monday) 11:59:59pm

Today's Topic

- Array
  - C-style String
- Pointer
  - Pointer Arithmetic
- Array and Pointer
- If we have tome – HW1 review

Example: swap operation?

```
#include <iostream>
void intSwap1(int num1, int num2) {
    int temp{num1};
    num1 = num2;
    num2 = temp; }
void intSwap2(int* num1, int* num2) {
    int temp{*num1};
    *num1 = *num2;
    *num2 = temp;}
int main(){
    int iNum1{ 1 };
    int iNum2{ 3 };
    std::cout << iNum1 << " " << iNum2 << std::endl;
    intSwap1(iNum1, iNum2);
    std::cout << iNum1 << " " << iNum2 << std::endl;
    intSwap2(&iNum1, &iNum2);
    std::cout << iNum1 << " " << iNum2 << std::endl;

     return 0;
}
```

← Call by value

← Call by reference

```
1 3
1 3
3 1
```

Array

- What is an **array**?
  - An aggregate **data type** that lets us access many variables of **the same type** through a **single** identifier

```
int studentID_1;
int studentID_2;
int studentID_3;
int studentID_4;
int studentID_5;
int studentID_6;
int studentID_7;
```

```
int studentID[7];
```

Array

- **Syntax**
  - Declaration

```
data_type variable_name [ #_of_elements ];
```

  - Usage

```
int studentID[10];
studentID[index] = 201811999; // 0 ≦ index ≦ 9
```

  - Initialization

```
int studentID[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
int studentID[10] {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
int studentID[ ] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
int studentID[10] = {1, };
```

Example: Char[] (string: not built-in type)

```
// “Hello World”, “John”
char name[10];
name[0] = 'J';
name[1] = 'o';
name[2] = 'h';
name[3] = 'n';
cout << name << endl;
```

- 'J' 'o' 'h' 'n' != "John"
- "John" == 'J' 'o' 'h' 'n' '\0'

```
John傲傲傲傲傲傲傲傲
傲傲傲傲傲傲傲傲傲傲
傲傲傲傲
```

Example: Char[] (string: not built-in type)

```
// c-style string initialization
char name [] = “John”;

cout << name[0] << endl;
cout << name << endl;
```

```
J
John
```

Example: Char[] (string: not built-in type)

```
#include <string.h>
char str[20] = "Hello";
char str2[] = "World";
cout << strlen(str) << endl;
cout << sizeof(str) << endl;
strncat_s(str, str2, 4);
cout << str << endl;
if (strcmp(str, "HelloWorld") == 0)
        cout << "OK" << endl;
else
        cout << "Fail" << endl;
char str01[] = "10";
char str02[] = "20";
cout << atoi(str01) * atof(str02) << endl;
```

```
5
20
HelloWorl
Fail
200
```

Multi-dimensional Array

```
int studentID_1;
int studentID_2;
int studentID_3;

char name_1[10] = "Alice";
char name_2[10] = "Bob";
char name_3[10] = "John";
```

```
int studentID[3];

char name[3][10];

char name[ ][3][10];
```

Multi-dimensional Array Initialization

```
int studentID[ 3 ] = {1, 2, 3};
int studentID[ ] = {1, 2, 3};

int studentID[2][3] = {1, 2, 3, 4, 5, 6};
int studentID[2][3] = { {1, 2, 3}, {4, 5, 6} };

int studentID[ ][3] = { {1 }, {4, 5, 6} };
```

Pointer

- **Variable**: a name for a piece of memory that holds a value
- **Pointer**: a variable that holds a *memory address* as its value

- Address-of (&)
  - To see what memory address is assigned to a variable
- Dereference (*)
  - To access the value at a particular address

```
int studentID = 201911999;
cout << “address of studentID :” << & studentID;
cout << “value at ” << & studentID << “ : “ << *(& studentID);
```

```
address of studentID : 010FFBD0
value at 010FFBD0 : 201911999
```

Pointer

- **Syntax**
  - Declaration

```
data_type * variable_name;
```

  - Initialization

```
char* name{ 0 };            // nullptr (C++11)
int iNum1 = 10;
int* pNum1 = &iNum1;
```

  - Usage

```
cout << "value:" << iNum1;
cout << "its address : " << pNum1;
*pNum1 = 10;
cout << "pointer Value" << *pNum1;
```

Pointer and Array

- Array Name can be used as a pointer

```
int iNum = 0;
int iNums[3] = { 1, 2, 3 };

int* pNum = &iNum;
int* pNums1 = &iNums[0];
int* pNums2 = iNums;                    // &iNums ?

cout << pNums1 << endl << pNums2 << endl;
cout << *pNums1 << endl << *pNums2 << endl;
cout << iNums[1] << endl << pNums1[1] << endl
<< pNums2[1] << endl;
```

```
0093FB24
0093FB24
1
1
2
2
2
```

Pointer Arithmetic

- Pointer + integer ?

```
int iNum[3]{ 1, 2, 3 };
int* pNum = iNum; // &iNum[0];
cout << pNum << endl;
cout << pNum + 1 << endl;
cout << pNum + 2 << endl;
cout << *pNum << endl;
cout << *(pNum + 1) << endl; // pNum[1]
```

```
0019FB20
0019FB24
0019FB28
1
2
```

Pointer and Array

- Multi-dimensional array and Pointer

```
// row: 2, col : 3
int iNums[2][3] = { 1, 2, 3, 4, 5, 6 };
cout << iNums << endl;
cout << *iNums << endl;
cout << **iNums << endl;

cout << **(iNums + 1) << endl;
cout << *((*iNums) + 1) << endl;
cout << *((*(iNums + 1)) + 1) << endl;
cout << *((*(iNums + 1)) + 1) + 1 << endl;
```

0053F604	iNums	iNums[0]	iNums[0][0]	1
0053F605				
0053F606				
0053F607				
0053F608			iNums[0][1]	2
0053F60C			iNums[0][2]	3
0053F610		iNums[1]	iNums[1][0]	4
0053F614			iNums[1][1]	5
0053F618			iNums[1][2]	6

Example: swap operation

```
#include <iostream>
void intSwap1(int  num1, int  num2) {
    int temp{num1};
    num1 = num2;
    num2 = temp; }
void intSwap2(int* num1, int* num2) {
    int temp{*num1};
    *num1 = *num2;
    *num2 = temp;}
void intSwap3(int & num1, int & num2) {
    int temp{num1};
    num1 = num2;
    num2 = temp; }
int * f1(){
    int iNums[3] {1,2,3};
    return iNums; }
```

```
int main(){
    int iNum1{ 1 };
    int iNum2{ 3 };
    std::cout << iNum1 << " " << iNum2
<< std::endl;
    intSwap3(iNum1, iNum2);
    std::cout << iNum1 << " " << iNum2
<< std::endl;

    int * pNums = f1();
    std::cout << *pNums << std::endl;

    return 0;
}
```

References

- Learn c++
  - https://www.learncpp.com/
  - Chapter 6.1~6.8

ANY QUESTIONS?