9주차 3차시 포인터 배열과 연산 실습

【학습목표】

1. 실습을 통한 포인터 연산을 학습하고 실행할 수 있다.
2. 실습을 통한 함수 인자로 배열 전달을 학습하고 실행할 수 있다.

학습내용1 : 포인터 연산 실습

√길이가 5인 int형 배열 arr을 선언하고 1,2,3,4,5로 초기화한 다음, 이배열의 첫 번째 요소를 가리키는 포인터 변수 ptr을 선언한다. 그 다음 포인터 변수 ptr에 저장된 값을 증가시키는 형태의 연산을 기반으로 배열요소에 접근하면서 모든 배열요소의 값을 2씩 증가시키고, 정상적으로 증가가 이뤄졌는지 확인하는 예제를 작성해 보자.

√위의 문제에서 포인터 변수 ptr에 저장된 값을 변경시켜가면서 배열요소에 접근하라고 하였다. 그런데 이번에는 포인터 변수 ptr에 저장된 값을 변경시키지 않고, ptr을 대상으로 덧셈연산을 하여 그 결과로 반환되는 주소 값을 통해서 모든 배열요소에 접근하여 값을 2씩 증가시키는 예제를 작성해 보자.

√길이가 5인 int형 배열 arr을 선언하고 이를 1,2,3,4,5로 초기화한 다음, 이 배열의 마지막 요소를 가리키는 포인터 변수 ptr을 선언한다. 그 다음 포인터 변수 ptr에 저장된 값을 감소시키는 형태의 연산을 기반으로 모든 배열 요소에 접근하여, 배열에 저장된 모든 정수를 더하여 그 결과를 출력하는 프로그램을 작성해 보자.

√길이가 6인 int형 배열 arr을 선언하고 이를 1,2,3,4,5,6으로 초기화한 다음, 배열에 저장된 값의 순서가 6,5,4,3,2,1이 되도록 변경하는 예제를 작성해 보자. 단, 배열의 앞과 뒤를 가리키는 포인터 변수 두 개를 선언해서 이를 활용하여 저장된 값의 순서를 뒤바꿔야 한다.

학습내용2 : 함수 인자로 배열 전달 실습

1. Call-by-value & Call-by-reference

√변수 num에 저장된 값의 제곱을 계산하는 함수를 정의하고 이를 호출하는 main함수를 작성해 보자.
단 여기서는 다음 두 가지 형태로 함수를 정의해야 한다.
- Call-by-value 기반의 SquareByValue 함수
- Call-by-reference 기반의 SquareByReference 함수

SquareByValue 함수는 인자로 전달된 값의 제곱을 반환해야 하며, SquareByReference 함수는 정수가 저장되어 있는 변수의 주소 값을 인자로 받아서 해당 변수에 저장된 값의 제곱을 그 변수에 다시 저장해야 한다.

√세 변수에 값을 서로 뒤바꾸는 함수를 정의해 보자. 예를 들어서 함수의 이름이 Swap3라 하면, 다음의 형태로 함수가 호출되어야 한다.

```
Swap3(&num1, &num2, &num3);
```

그리고 함수호출의 결과로 num1dp 저장된 값은 num2에, num2에 저장된 값은 num3에 그리고 num3에 저장된 값은 num1에 저장되어야 한다.

【학습정리】

비교조건 \ 비교대상	포인터 변수	배열의 이름
이름이 존재하는가?	존재한다	존재한다
무엇을 나타내거나 저장하는가?	메모리의 주소 값	메모리의 주소 값
주소 값의 변경이 가능한가?	가능하다	불가능하다.

배열 이름과 포인터 변수의 비교

1. 포인터 변수에 저장된 값을 대상으로 하는 증가 및 감소연산을 진행할 수 있다.(곱셈, 나눗셈 등등은 불가).이것도 포인터 연산의 일종이다.

2. 배열을 함수의 매개변수에 전달하는 이유는 함수 내에서 배열에 저장된 값을 참조하도록 하기 위함이다. 그러나 배열을 통째로 복사하는 방법은 C언어에 존재하지 않는다. 따라서 배열을 통째로 복사해서 전달하는 방식 대신에, 배열의 주소 값을 전달하는 방식을 대신 취한다.

3. call-by-value 형태의 함수에서는 함수 외부에 선언된 변수에 접근이 불가능하다. 그러나 call-by-reference 형태의 함수에서는 외부에 선언된 변수에 접근이 가능하다.