포인터 1

C 프로그래밍 - 임진혁

포인터(Pointer)

주소 값을 저장할 수 있는 변수

주소는 컴퓨터의 저장 공간(메모리)에서 어느 위치에 해당 값이 존재하고 있는지를 표현합니다.

즉 포인터는 해당 데이터가 실제로 어디에 있는지를 나타내는 용도로 사용합니다.

포인터 사용 목적

1. 함수, 자료 구조에 대한 접근을 쉽게 하기 위해(해당 값들이 저장된 실제 위치를 찾아 접근, 수정하는 방식)
2. 정확한 위치를 알아두면, 그 다음부터는 다시 위치를 알 필요가 없어짐.(효율 증가)
3. 데이터에 대한 접근과 수정에 용이함.
4. 전역 변수 사용에 대한 억제(call by reference)
5. 저장 공간에 대한 효율적인 사용
6. 연속된 데이터인 배열 같은 구조에서 효율적으로 접근이 가능함.

포인터 사용 시 유의 사항

1. 주소에 대한 직접적인 컨트롤을 진행하는 작업이기에 예외 처리에 대한 설계가 제대로 이루어지지 않을 경우 문제가 많이 발생하는 설계 방식
2. 선언만 할 경우에도 무언가(쓰레기값)를 가리키고 있는 상태이기에 NULL 포인터를 지정해두는 것이 안전합니다.
3. 시스템 오류가 발생할 가능성이 있음(메모리의 절대 번지에 대한 접근)
4. 의도치않은 원본 값 변경이 발생할 수 있음(포인터는 원본의 실제 위치를 가리키기 때문)

포인터 선언 및 초기화

타입* 변수명;

ex) int* ptr;

변수명 = 주소값;

ptr = &a;

*(애스터리스크)는 타입 다음에 붙으면 참조 연산자로 주소 값으로부터 값을 반환하는 기능을 가진 연산자로 사용됩니다.

&(앰퍼샌드)는 이름 앞에 붙으면 주소 연산자로 해당 변수로부터 주소 값을 반환하는 연산자로 사용됩니다.

포인터 관련 출력문

```
int a = 10;

int* ptr = &a;


printf(“%d”, *ptr); //포인터 변수 ptr의 값 출력

printf(“%p”, ptr); //포인터 변수 ptr의 주소 출력(16진수)
```

NULL 포인터

ex)

int* ptr = NULL;

현재 가리키고 있는 값이 없음을 표현하는 NULL 포인터

선언 시의 안전성을 위해 사용합니다.

배열(포인터)

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

int* ptr = array; //배열의 이름은 주소이기에 &를 붙이지 않습니다.

printf(“%d”, ptr[0]); ptr은 배열을 가리키고 있기 때문에 인덱스 사용이 가능합니다.

printf(“%d”, *(array+1)); *(배열명 + 숫자)를 통해 배열의 n번째 값에 대한 표현 가능 (포인터 연산)

printf(“%d”, *(ptr+2));

함수(포인터)

```
void swap(int* a, int *b)
{
   temp = *a;
     *a = *b;
     *b = temp;
}
int a  =5; int b = 7;
swap(&a,&b); //매개변수가 포인터일 경우 인자 값은 주소 값으로 작성합니다.
```

함수가 종료되면 함수 안에 만들어진 매개변수와 지역변수는 메모리 상에서 제거됩니다.

하지만 실제 위치에 값을 대입했기 때문에 원본에 변화가 생겨 값에 대한 변화는 그대로 적용됩니다.(call by reference)

포인터 기본 연습 문제 1

각 배열의 합을 구하는 함수를 구현하시오.

int ArraySum(int* p, int size);

배열을 역순으로 출력하는 함수를 구현하시오.

void ArrayPrintReverse(int* p , int size);

문제 풀이 후 까페 개인 채팅 란에 이름 작성 후 문제 제출 시 답 첨삭 해드립니다.