Void Swap(int\* a, int\* b) {

int temp = \*a;

\*a = \*b;

\*b = temp

}

int main() {

int num1 = 1;

int num2 = 2;

cout << num1 << num2;

Swap(&num1, &num2);

cout << num1 << num2;

return 0;

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *메모리* | *변수명* | *Swap 전 값* | *Swap 중 값* | *Swap 후 값* |
| 0x0000 | num1 | 1 | 2 | 2 |
| 0x0004 | num2 | 2 | 1 | 1 |
| 0x0008 | A | Null | 0x0000 | 메모리에 들어있는 값 자체는 설거지 안한채로 남아있.  그치만 delete하면 다시는 이 주소의 a,b,temp에는 접근할 수 없게 돼 |
| 0x0012 | B | Null | 0x0004 |
| 0x0016 | temp | null | 1 |

외부 함수의 호출 형태

1. Call by value

매개변수 값을 복사해서 (외부에서 선언한) 함수 내 변수에 복사

-> 매개변수 자체는 변치 않음

-> 함수 호출 시마다 값을 복사 -> 메모리 매번 새로운거 지정

2. Call by refence

1) 주소값을 사용(call by address)

2) 참조자를 사용(call by reference)

새로 선언되는 변수 앞에 &가 붙으면 “참조자”

// 하지만 원래 있던 변수 앞에 &이면 변수의 “주소 값을 반환”하는 연산자 역할!