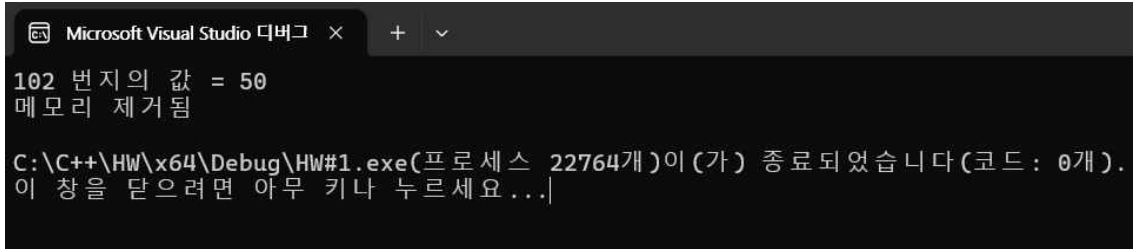


◆ 소스 수행 결과 화면



```
Microsoft Visual Studio 디버그 × + v
102 번지의 값 = 50
메모리 제거됨
C:\C++\HW\x64\Debug\HW#1.exe(프로세스 22764개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

◆ 소스 구현 설명

- 문제 정의

문제에서 주어진 main() 함수와 Ram 클래스의 선언부를 참고하여 Ram 클래스 멤버에 대한 구현부를 작성하는 문제이다.

- 문제 해결 방법 / 아이디어 평가

1. 문제의 소스를 헤더 파일인 Ram.h와 cpp 파일인 Ram.cpp, main.cpp 파일로 분리한다.

2. main.cpp과 Ram.h 파일에는 문제에서 제시한 main() 함수 Ram 클래스 코드를 동일하게 작성한다.

문제의 코드를 변경하지 않고 문제의 실행 결과와 동일하게 나오게 하기 위해서이다.

3. main.cpp 파일과 Ram.cpp 파일 코드 상단에 아래의 코드를 작성한다.

```
#include "Ram.h"
```

이 코드로 Ram 클래스와 메소드를 사용할 수 있다.

4. Ram.cpp 파일에 문제의 Ram 클래스에서 사용된 모든 멤버 함수를 작성한다.

```
Ram::Ram() {
    size = 100 * 1024;
    for (int i = 0; i < size; i++) {
        mem[i] = 0;
    }
}
```

위의 코드는 생성자 함수 Ram()을 작성한 것이다. 먼저, size를 100*1024로 초기화하고 for 문을 사용하여 메모리 배열을 0으로 초기화하게 만들었다.

```
Ram::~~Ram() {
```

```
        cout << "메모리 제거됨" << endl;
    }
```

위의 코드는 소멸자 함수 ~Ram()을 작성한 것이다. 소멸자 함수 ~Ram()은 메모리 객체가 소멸될 때 "메모리 제거됨" 문자열이 출력되게 만들었다.

```
char Ram::read(int address) {
    return mem[address];
}
```

위의 코드는 read 메소드이다. read 메소드는 address 주소의 메모리 바이트를 읽어 반환한다.

```
void Ram::write(int address, char value) {
    mem[address] = value;
}
```

마지막으로 write 메소드이다. write 메소드는 주어진 주소에 한 바이트로 값을 저장한다.

5. Ram.cpp 파일을 완성한 후 main.cpp 파일을 실행시키면 소스 수행 결과 화면이 나온다.

- 문제를 해결한 키 아이디어

문제를 해결한 키 아이디어는 'size' 변수이다. 문제를 해결하면서 메모리 크기를 어떻게 초기화하는 게 좋은지 의문이 들었는데 size를 먼저 설정한 후에 메모리 배열을 초기화하는 방법이 중요하다는 것을 알 수 있었다. size가 적절한 값으로 설정된 후에 mem 배열이 올바르게 초기화되는 것이다. 이를 통해 문제의 실행 결과가 올바르게 나올 수 있었다.