Ram.h

```
1  #ifndef RAM_H
2  #define RAM_H
3
4  class Ram {
5      char mem[100 * 1024];
6      int size;
7  public:
8      Ram();
9      ~Ram();
10     char read(int address);
11     void write(int address, char value);
12 };
13
14 #endif
```

1, 2, 14번 라인의 #ifndf, #define, #endif 명령어를 통하여 RAM_H가 이미 정의 되어있다면 중복으로 정의되는것을 방지합니다.

4~12번 라인에서 Ram클래스를 정의하고 char mem[100*1024]; 100kb의 메모리 공간을 정의하고, int size;를 통하여 정수형 변수 size를 선언합니다.
8~9번 라인에서 Ram의 생성자와 소멸자를 정의하고 10번 라인을 통하여 주어진 라인의 주소를 읽고, 11번 라인을 통하여 주어진 주소에 특정 값을 저장합니다.

Ram.cpp

```
1  #include <iostream>
2  using namespace std;
3
4  #include "Ram.h"
5
6  Ram::Ram() {
7      size = 100 * 1024;
8      for (int i = 0; i < size; ++i)
9          mem[i] = 0;
10 }
11 Ram::~Ram() {
12     cout << "메모리 제거됨" << endl;
13 }
14 char Ram::read(int address) {
15     return mem[address];
16 }
17 void Ram::write(int address, char value) {
18     mem[address] = value;
19 }
```

4번 라인에서 Ram.h 헤더파일을 include 합니다.
6~10번 라인에 Ram클래스의 Ram생성자를 이용하여 size를 100kb로 정의하고 for문을 이용하여 메모리를0으로 초기화하고 배열을 생성한후 값을 1씩 증가하여 size값과 같아질때까지 반복합니다.

Ram.cpp

```cpp
#include <iostream>
using namespace std;

#include "Ram.h"

Ram::Ram() {
    size = 100 * 1024;
    for (int i = 0; i < size; ++i)
        mem[i] = 0;
}
Ram::~Ram() {
    cout << "메모리 제거됨" << endl;
}
char Ram::read(int address) {
    return mem[address];
}
void Ram::write(int address, char value) {
    mem[address] = value;
}
```

11~13번 라인에 Ram클래스의 소멸자Ram을 이용하여 객체가 소멸될 때 "메모리 제거됨" 이라는 문구가 화면에 출력되도록 만듭니다.
14~16번 라인을 통하여 배열의 address위치에 있는 값을 반환합니다.
17~19번 라인을 통하여 배열의 address위치에 값을 저장합니다.

main.cpp

```
1  #include <iostream>
2  using namespace std;
3
4  #include "Ram.h"
5
6  int main() {
7      Ram ram;
8      ram.write(100, 20);
9      ram.write(101, 30);
10     char res = ram.read(100) + ram.read(101);
11     ram.write(102, res);
12     cout << "102 번지의 값 = " << (int)ram.read(102) << endl;
13 }
```

4번 라인에서 Ram.h 헤더파일을 include 한다.
6라인에서 메인 함수를 선언하고, 7번 라인에서 Ram클래스의 ram객체를 생성하고 이때 Ram.cpp에 작성된 내용을 통하여 생성자가 호출되며 메모리가 0으로 초기화 됩니다.
8~9번 라인 명령어를 통하여 100번지에 20의 값을, 101번지에 30의 값을 저장합니다.
10번 라인에서 res에 100번지의 값(20)과 101번지의 값(30)을 읽어와 그 둘을 더한 값을 저장합니다. 따라서 res=50이 됩니다.
11번 라인에서 102번지에 res의 값(50)을 저장합니다.
12번 라인에서 "102 번지의 값 = " 뒤에 (int)ram.read(102)명령어를 통해102번지의 값을 읽어와 정수형으로 화면에 출력합니다.
따라서 실행시 화면에 "102 번지의 값 = 50"이 출력됩니다.