```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include <sstream>

#include "Assembler.h"

using namespace std;

Assembler::Assembler()
{
}

int Assembler::assemble(fstream& in, fstream& out)
{
    string line;
    string opcode;
    int rd, rs, addr, constant;
    int instruction;

    const int success = false;
    const int error = true;
    //const int debug = false;

    getline(in, line);
    while (!in.eof()) {
        if (line[0] == '!' or line == "") { // ignore comment or empty lines
            getline(in, line);
            continue;
        }

        istringstream str(line.c_str());
        str >> opcode;

        if (opcode == "load") {
            str >> rd >> addr;
            if (rd < 0 || rd > 3)
                return error;
            if (addr < 0 || addr > 255)
                return error;
            instruction = 0;
            instruction = instruction | rd<<9 | addr;

        } else if (opcode == "loadi") {
            str >> rd >> constant;
            if (rd < 0 || rd > 3)
                return error;
            if (constant < -128 || constant > 127)
                return error;
            instruction = 0;
            instruction = instruction | rd<<9 | 1<<8 | (0xff & constant);

        } else if (opcode == "store") {
            str >> rd >> addr;
            if (rd < 0 || rd > 3)
                return error;
            if (addr < 0 || addr > 255)
                return error;
            instruction = 1;
            instruction = instruction<<11 | rd<<9 | 1<<8 | addr;

        } else if (opcode == "add") {
            str >> rd >> rs;
            if (rd < 0 || rd > 3)
                return error;
            if (rs < 0 || rs > 3)
                return error;
            instruction = 2;
            instruction = instruction<<11 | rd<<9 | rs<<6;

...
        } else if (opcode == "noop") {
            instruction = 25;
            instruction = instruction<<11;

        } else
            return error;

        out << setfill('0') << setw(5) << instruction << endl;
        getline(in, line);
    }
    return success;
} // assemble
```