```cpp
using namespace std;

class test_progress {
private:
    progress mainProgress;  // Testing progress variable
    bool setValues();
    bool getValues();
    void breakline();
public:
    bool executeTests();
};

void test_progress::breakline() {
    // Visual function, outputs an 80 character breakline
    // Requriements: N/A

    // Output a newline for ease of use
    cout << endl;

    // Create 80 dashes
    for (int i = 0; i <= 80; i++) {
        cout << "-";
    }

    // Another newline, for luck
    cout << endl;
}

bool test_progress::setValues() {
    bool returnVal = true;  // Whether the test succeeded

    try {
        // Attempt to set the values of the class
        mainProgress.activate();
        mainProgress.setStudentName("Matthew Bowker");
        mainProgress.answerFalse();
        mainProgress.answerFalse();
        mainProgress.answerTrue();
        mainProgress.answerTrue();
        mainProgress.answerFalse();
    }
    catch (...) {
        // If it fails, the test fails
        returnVal = false;
    }

    // Return whether the test succeeded
    return returnVal;
}

bool test_progress::getValues() {
    bool returnValue = true; // Whether the test succeed

    try {
        // Output the stored values from the module
        cout << "Is the module active? " << mainProgress.getActive() << endl;
        cout << "Student Name: " << mainProgress.getStudentName() << endl;
        cout << "Number true: " << mainProgress.getNumTrue() << endl;
        cout << "Number false: " << mainProgress.getNumFalse() << endl;
    }
    catch (...) {
        // If it fails, the test fails
        returnValue = false;
    }

    // Return whether he test succeeded
    return returnValue;
}

bool test_progress::executeTests() {
    bool returnValue = true;

    cout << "Getting values for an inactive module..." << endl;
    if (getValues() == false) {
        returnValue = false;
    }

    breakline();

    cout << "Setting values and activating module..." << endl;
    if (setValues() == false) {
        returnValue = false;
    }

    breakline();

    cout << "Getting values for an active module..." << endl;
    if (getValues() == false) {
        returnValue = false;
    }

    return returnValue;
}
```