

Refactoring Code Examples

1 A Simple Class

The following class maps a n-dimensional real vector and its magnitude into C++.

Listing 1: The MagVector Class

```
1 #ifndef _MAGVECTOR_H_
2 #define _MAGVECTOR_H_
3
4 #include <vector>
5 #include <iostream>
6 #include <cmath>
7 #include <stdexcept>
8
9 class MagVector
10 {
11 private:
12
13     std::vector<float> _content;
14
15 public:
16
17     MagVector(const int& _size):
18         _content(_size,0.){
19     };
20
21     virtual ~MagVector(){};
22
23     void print(){
24         std::vector<float>::const_iterator vItr = _content.begin();
25         std::vector<float>::const_iterator vEnd = _content.end();
26         std::cout << "vector\t";
27         for (; vItr!=vEnd; ++vItr){
28             std::cout << *vItr << ", ";
29         }
30         std::cout << std::endl;
31     }
32
33     void append(const float& _new){
34         _content.push_back(_new);
35     }
36
37     void set(const int& _index, const float& _new){
38         if(_index>-1 && _index<_content.size()){
39             _content[_index] = (_new);
40         }
41         else
42             throw std::out_of_range("MagVector:: set >> _index out of range");
43     }
44
45     float get(const int& _index){
46         try{
47             return _content.at(_index);
48         }
49         catch(...){
50             std::cerr << __FILE__ << ":\t" << "index out of range!\n";
51             return 0.;
52         }
53     }
54
55     float magnitude() const {
56         float valueSquared = 0.;
57         for (short index=0; index<_content.size(); ++index){
58             valueSquared += ((_content[index])*(_content[index]));
59         }
60
61         if(valueSquared >= 0.)
62             return std::sqrt(valueSquared);
63         else
64             return -1.;
65     }
66 };
67
68 #endif /* _MAGVECTOR_H_ */
```

2 Simple Tests

Using boost, we write tests to ensure that all behavior and exceptions are covered.

Listing 2: The MagVector Class Tests

```
1 #define BOOST_TEST_DYN_LINK
2 #define BOOST_TEST_MODULE MagVectorTests
3 #include "MagVector.hh"
4
5 #include <boost/test/unit_test.hpp>
6
7 // BOOST_AUTO_TEST_CASE( testFails )
8 // {
9 //     BOOST_FAIL("This test fails!");
10 // }
11
12 BOOST_AUTO_TEST_SUITE( MagVectorSuite )
13
14 BOOST_AUTO_TEST_CASE( testValueGet )
15 {
16     MagVector aVector(3);
17     BOOST_CHECK_MESSAGE(aVector.get(0) == 0., "Content_on_0_not_on_default_value!");
18 }
19
20 BOOST_AUTO_TEST_CASE( testValueGetAllDefaultZeros )
21 {
22     MagVector aVector(3);
23     BOOST_CHECK_MESSAGE(aVector.get(0) == 0.
24                         && aVector.get(1) == 0.
25                         && aVector.get(2) == 0.,
26                         "Content_on_0_not_on_default_value!");
27 }
28
29 BOOST_AUTO_TEST_CASE( testSetValueAtIndex )
30 {
31     MagVector aVector(3);
32     aVector.set(0,42);
33     BOOST_CHECK_MESSAGE(aVector.get(0) == 42.,
34                         "Content_on_0_not_at_42!");
35 }
36
37 BOOST_AUTO_TEST_CASE( testSetValueAtOutOfRangeIndex )
38 {
39     MagVector aVector(3);
40     BOOST_CHECK_THROW(aVector.set(4,42), std::out_of_range );
41 }
42
43 BOOST_AUTO_TEST_CASE( testMagnitudeOf0 )
44 {
45     MagVector aVector(3);
46     BOOST_CHECK_MESSAGE(aVector.magnitude()==0., "vector_magnitude_not_on_default" );
47 }
48
49 BOOST_AUTO_TEST_CASE( testMagnitudeOf4 )
50 {
51     MagVector aVector(3);
52     aVector.set(1,4);
53     BOOST_CHECK_MESSAGE(aVector.magnitude()==4, "vector_magnitude_unequal_4" );
54 }
55
56 BOOST_AUTO_TEST_SUITE_END()
```

3 Simple Tests on a Fixture

Often classes require input data that is complex to instantiate. Instead of setting up `MagVector` per test case, a class can do that automatically every time a test case is called (this class is dubbed a `TestFixture`).

Listing 3: The `MagVector` Class Tests on a Fixture

```
1 #define BOOST_TEST_DYN_LINK
2 #define BOOST_TEST_MODULE MagVectorTestsSuiteAndFixture
3 #include "MagVector.hh"
4 #include <boost/test/unit_test.hpp>
5
6 class MagVectorFixture
7 {
8
9 public:
10
11     MagVector ThreeVector;
12
13     MagVectorFixture():
14         ThreeVector(3)
15     {
16         BOOST_MESSAGE( "setup_fixture" );
17     };
18
19     virtual ~MagVectorFixture() { BOOST_MESSAGE( "teardown_fixture" ); };
20
21 };
22
23 BOOST_FIXTURE_TEST_SUITE( MagVectorSuite, MagVectorFixture )
24
25 BOOST_AUTO_TEST_CASE( testValueGet )
26 {
27     BOOST_CHECK_MESSAGE( ThreeVector.get(0) == 0., "Content_on_0_not_on_default_value!" );
28 }
29
30 BOOST_AUTO_TEST_CASE( testValueGetAllDefaultZeros )
31 {
32     BOOST_CHECK_MESSAGE( ThreeVector.get(0) == 0.
33                         && ThreeVector.get(1) == 0.
34                         && ThreeVector.get(2) == 0.,
35                         "Content_on_0_not_on_default_value!" );
36 }
37
38 BOOST_AUTO_TEST_CASE( testSetValueAtIndex )
39 {
40     ThreeVector.set(0,42);
41     BOOST_CHECK_MESSAGE( ThreeVector.get(0) == 42.,
42                         "Content_on_0_not_at_42!" );
43 }
44
45 BOOST_AUTO_TEST_CASE( testSetValueAtOutOfRangeIndex )
46 {
47     BOOST_CHECK_THROW( ThreeVector.set(4,42), std::out_of_range );
48 }
49
50 BOOST_AUTO_TEST_CASE( testMagnitudeOf0 )
51 {
52     BOOST_CHECK_MESSAGE( ThreeVector.magnitude()==0., "vector_magnitude_not_on_default" );
53 }
54
55 BOOST_AUTO_TEST_CASE( testMagnitudeOf4 )
56 {
57     ThreeVector.set(1,4);
58     BOOST_CHECK_MESSAGE( ThreeVector.magnitude()==4, "vector_magnitude_unequal_4" );
59 }
60
61 BOOST_AUTO_TEST_SUITE_END()
```