```cpp
#include <iostream>

int main()
{
    std::cout << "Enter two numbers to multiply:\n";

    double x, y;
    std::cin >> x >> y;
    if (!std::cin) {
        std::cerr << "Could not read numbers!\n";
        return 1;
    }

    std::cout << x << " * " << y
              << " == " << x * y << "\n";
}
```

```cpp
#include <catch.h>

void inc_p(int* p)
{
    *p += 1;
}

TEST_CASE("C-style increment")
{
    int x = 0;
    inc_p(&x);
    CHECK( x == 1 );
}

void inc_r(int& r)
{
    r += 1;
}

TEST_CASE("C++-style increment")
{
    int x{0};
    inc_r(x);
    CHECK( x == 1 );
}

void swap_p(int* p, int* q) {
    int temp = *p;
    *p = *q;
    *q = temp;
}

TEST_CASE("C-style swap")
{
    int x = 3, y = 4;
    swap_p(&x, &y);
    CHECK( x == 4 );
    CHECK( y == 3 );
}

void swap_r(int& r, int& s)
{
    int temp = r;
    r = s;
    s = temp;
}

TEST_CASE("C++-style swap")
{
    int x = 3, y = 4;
    swap_r(x, y);
    CHECK( x == 4 );
    CHECK( y == 3 );
}

void broken_swap_r(int& r, int& s)
{
    int& temp = r;
    r = s;
    s = temp;
}

TEST_CASE("broken swap")
{
    int x = 3, y = 4;
```

```
    broken_swap_r(x, y);
    CHECK( x == 4 );
    CHECK( y == 4 );
}
```

```cpp
#include <catch.h>
#include <stdexcept>
#include <vector>

using VI = std::vector<int>;

TEST_CASE("vector creation and access")
{
    std::vector<int> v1{ 2, 4, 6, 8 };
    std::vector<double> v2(10, 3.5);

    CHECK( v1.size() == 4 );
    CHECK( v2.size() == 10 );

    CHECK( v1[1] == 4 );
    CHECK( v2[1] == 3.5 );

    v1[1] = 15;
    CHECK( v1[1] == 15 );
}

TEST_CASE("growing and shrinking")
{
    VI v;
    CHECK( v == VI{} );
    v.push_back(2);
    CHECK( v == VI{2} );
    v.push_back(5);
    v.push_back(9);
    CHECK( v == VI{2, 5, 9} );
    v.pop_back();
    CHECK( v == VI{2, 5} );
}

TEST_CASE("bounds checking (or not)")
{
    std::vector<int> v{2, 3, 4};

    CHECK(v.at(2) == 4);
    v.at(2) = 8;
    CHECK(v.at(2) == 8);

    CHECK_THROWS_AS(v.at(3), std::out_of_range);

    v[10] = 12;           // UB!
    CHECK( v[10] == 12 ); // also UB!
}

static void inc_vec_wrong(std::vector<int> v)
{
    for (size_t i = 0; i < v.size(); ++i)
        ++v[i];
}

TEST_CASE("vector passed by value")
{
    VI v{2, 3, 4};
    inc_vec_wrong(v);
    CHECK( v == VI{2, 3, 4} );
}

static void inc_vec(std::vector<int>& v)
{
    for (size_t i = 0; i < v.size(); ++i)
        ++v[i];
```

```
}

TEST_CASE("vector passed by reference")
{
    VI v{2, 3, 4};
    inc_vec(v);
    CHECK( v == VI{3, 4, 5} );
}

static double sum_vec(std::vector<double> const& v)
{
    double result = 0;
    for (double d : v) result += d;
    return result;
}

TEST_CASE("sum vector passed by const&")
{
    std::vector<double> v{1.1, 20.2, 400};
    CHECK( sum_vec(v) == Approx(421.3) );
}

static void dec_vec_wrong(std::vector<int> &v)
{
    for (int z : v) --z;
}

TEST_CASE("for-each by value")
{
    VI v{3, 4, 5};
    dec_vec_wrong(v);
    CHECK( v == VI{3, 4, 5} );
}

static void dec_vec(std::vector<int> &v)
{
    for (int& z : v) --z;
}

TEST_CASE("for-each by reference")
{
    VI v{3, 4, 5};
    dec_vec(v);
    CHECK( v == VI{2, 3, 4} );
}
```