```
#include "gtest/gtest.h"
#include "circle.hpp"
#include <cmath>

// NETIDS: ineto001, amiya017

// Tests for double area() const;
TEST(CircleArea, doubleRad)  {

    // Initially 10 circles with the radius being a double
    for (int i = 0; i < 10; i++) {
        double rad = i * 1.34567;
        Circle curr_test = Circle(rad);
        EXPECT_NEAR(M_PI * rad * rad, curr_test.area(), 0.0001);
    }

}

TEST(CircleArea, negRad) {

    // Initialize a circle with radius being a negative double
    double rad = -10.0;
    Circle curr_test = Circle(rad);
    ASSERT_DOUBLE_EQ(-1.0, curr_test.area());
}

TEST(CircleArea, zeroRad) {

    // Initialize a circle with the radius being zero (double form)
    double rad = 0.0;
    Circle curr_test = Circle(rad);
    EXPECT_DOUBLE_EQ(0.0, curr_test.area());
}


// Tests for double perimeter() const;
TEST(CirclePer, doubleRad) {

    // Initialize 10 circles with the radius being a double
    for (int i = 0; i < 10; i++) {
        double rad = i * 1.34567;
        Circle curr_test = Circle(rad);
        EXPECT_NEAR(2* M_PI * rad, curr_test.perimeter(), 0.0001);
    }
}

TEST(CirclePer, negRad) {

    // Initialize a circle with the radius being a negative double
    double rad = -10.0;
    Circle curr_test = Circle(rad);
    ASSERT_DOUBLE_EQ(-1.0, curr_test.perimeter());
}

TEST(CirclePer, zeroRad) {

    // Initialize a circle with the radius being zero (double form)
    double rad = 0.0;
    Circle curr_test = Circle(rad);
    EXPECT_DOUBLE_EQ(0.0, curr_test.perimeter());
}

int main(int argc, char** argv) {
    // Initialize the Google Test library
    ::testing::InitGoogleTest(&argc, argv);
    return RUN_ALL_TESTS(); // Run all tests
}
```