Google Test Cheat Sheet

Simple Test

```cpp
TEST(UserTests, isAdult_AgeLessThan18_False)
{
    User paul(17);
    ASSERT_FALSE(paul.isAdult());
}
```

Basic Assertions

```cpp
ASSERT_TRUE(condition);

ASSERT_FALSE(condition);

ASSERT_EQ(expected, actual);         // Equal

ASSERT_NE(expected, actual);         // Not equal

ASSERT_LT(value1, value2);               // value1 < value2

ASSERT_LE(value1, value2);           // value1 <= value2

ASSERT_GT(value1, value2);           // value1 > value2

ASSERT_GE(value1, value2);           // value1 >= value2

ASSERT_STREQ(expected, actual);      // Strings equal

ASSERT_STRNE(expected, actual);      // Strings not equal

ASSERT_STRCASEEQ(expected, actual);  // Strings equal, ignoring case

ASSERT_STRCASENE(expected, actual);  // Strings not equal, ignoring case
```

Exception Assertions

```cpp
ASSERT_THROW(statement, exception_type);

ASSERT_ANY_THROW(statement);

ASSERT_NO_THROW(statement);

    TEST(CarTests, crash_Normal_ShouldThrow)
    {
        Car car;
        ASSERT_THROW(car.crash(), CrashException);
    }
```

Test Fixture

```cpp
class UserTests : public ::testing::Test
{
    protected:
        virtual void SetUp()
        { // runs before each test }
        virtual void TearDown()
        { // runs after each test }
        User myUser(16);
};


TEST_F(UserTests, isAdult_AgeLessThan18_False)
{
    ASSERT_FALSE(myUser.isAdult());
}
```

Value Parameterized Tests

```cpp
class UserTests : public ::testing::TestWithParam<int>
{ // it is also a normal test fixture with setup and tear down };


TEST_P(UserTests, isAdult_AgeGreaterThan18_True)
{
    User user(GetParam());
    ASSERT_TRUE(user.isAdult());
}


INSTANTIATE_TEST_SUITE_P(InstantiationName, UserTests,
                         ::testing::Values(18, 20, 60));
```