26 | 牛刀小试（上）：实现测试框架前的基础准备

2020-03-19 胡光

人人都能学会的编程入门课 进入课程 >

讲述：胡光
时长 14:38　大小 13.41M

你好，我是胡光。经历了千难万险，我们终于来到了这次编程探险旅程的最后一个阶段“综合项目篇”。

还记得开篇的时候，我跟你讲过的，要带你看看这片编程森林中最有趣的地方，不知道你感受到这份乐趣了吗？接下来的最后几节课，是我为你精心准备的，作为你编程入门学习之行的完成礼物，同时，也是对整个专栏学习内容的一个总结升华。

对于本章的学习，你需要综合运用“语言基础篇”“编码能力训练篇”以及“算法数据篇”中所学知识，完成两个小项目，一个是“个人测试框架开发”，另一个是“自制简易计算器”。我希望通过这两个项目，打通你编程学习的“任督二脉”，让你在后续的其他编程知识学习中，可以游刃有余，乐在其中。

软件开发流程

在开始做项目之前呢，让我们先来了解下一般的项目开发流程。

拿传统的软件工程开发流程来说，一个软件的项目开发流程，包括：需求分析、概要设计、详细设计、编码、测试、软件交付、验收和维护。虽然包括了 8 个阶段，可总的来说，你可以把它总结为软件开发的前中后三个部分。

软件开发的前期，由产品经理、项目经理跟进需求，做需求分析，然后是概要设计，出一份系统的详细设计。这是项目的头部阶段，主要是从理论上验证需求可行，并且最终产出一份切实可行的软件系统的详细设计。就像我在咱们每节课中，给你留思考题，此时我的角色就是项目经理和产品经理，我首先需要分析给你留的思考题是否可做，以及跟你说明白，大致怎么做。

软件开发的中期，是由技术人员负责，主要是根据系统的详细设计，进行编码和测试，把图纸上的系统实现出来。你也可以认为这是一个施工的过程。这个过程，类比到咱们的思考题上，就是你根据我的提示，具体完成每一道思考题的程序，并且对完成的程序做简单测试，以保证程序的正确性。

软件开发的后期，由项目经理带领技术支持人员做软件交付、验收及后期维护的相关工作。这一部分，你可以理解为，是你交作业的过程，客户就是老师，你的作业要是不合格，老师就会给你退回去并提出修改建议。

从这个过程中，你可以看到作为技术人员，与我们相关的，就是软件开发的中期阶段，也就是编码和测试。

在之前的学习中，我们把学习任务主要放在了编码阶段，一直没有提测试阶段的事情，下面我们就来说说测试阶段主要做的事情。

黑盒测试与白盒测试

关于测试阶段的测试方法，可以大致分为五种：白盒测试，黑盒测试，灰盒测试，静态测试与动态测试。

看着这些测试方法的名称，你可能有点儿懵，其实你完全没有必要掌握全部的测试方法，只需要了解其中的黑盒测试和白盒测试就行，知道了这两种测试方法，就可以满足你对项目开发流程的概念认知。

我们先来说说白盒测试与黑盒测试都是什么意思。首先名字里面的“盒”，其实指的就是项目中的系统，你可以理解成为是我们写的程序，它也是被测试的对象。至于这个白与黑呢，意思就是在测试过程中，是否关注代码实现逻辑。

白盒测试，就是关注代码实现逻辑，从而产生的测试行为。这种行为就像你把一个盒子打开，仔细检查其内部有无错误一样。

而不关注代码实现逻辑，而产生的测试行为，就是黑盒测试。在这个测试过程中，我们只关注系统的输出是否满足我们的要求。

下面我来举几个具体的例子，你来分析一下哪一种是白盒测试，哪一种是黑盒测试。

场景一：你写完了一个程序，运行以后输入数据，测试程序的输出结果是否符合预期。

场景二：当你发现，程序的结果不符合预期的时候，开始检查你程序的代码逻辑，并且针对于每一个函数功能做测试。

在场景一中，虽然程序是你写的，可你在做测试的时候，并没有关心程序内部的代码逻辑，而是关注整体程序的功能是否符合预期，所以这个属于黑盒测试。

而在场景二中，你在测试程序的时候，关注到了程序内部的代码逻辑是否正确，并且针对代码中的函数，开始做针对性测试。由于这个测试过程关注到了代码本身的逻辑，而不单单是程序功能本身，所以，这属于白盒测试。

通过这两个日常写程序的场景，你会发现，其实测试行为对于我们来说并不陌生，测试就是为了保证程序功能的正确性的，而所谓的黑盒测试或者白盒测试，并没有优劣之分。在实际工作中，我们也会经常综合运用这两种测试，来查找程序中存在的潜在问题。

总的来说，想要保证程序的正确性，必然会涉及到运用相关的测试方法。所以请你记住，测试并不是测试人员的专属，很多时候也是开发人员需要掌握的技能。

对于开发人员来说，最基本的测试方法，就是针对自己程序中的每一个功能模块，编写对应的单元测试，而单元测试中的单元，是程序中最小的测试单位。例如 C 语言中，一个单元就是一个函数，C++ 中的一个单元，就是一个类。

所以，我们说到 C 语言的单元测试，指的是对每一个函数，编写一段测试程序。而可以辅助开发人员编写这些单元测试的程序，我们叫做“单元测试框架”，也可以简称为“测试框架”。

初识：Google 测试框架

好了，关于软件开发流程的基本情况呢，我讲完了，下面我们正式说说测试框架。

咱们要完成的第一个项目，就是编写一个简易的 C 语言测试框架。不过在写这个 C 语言测试框架之前，咱们先看看之前的大厂或大牛们是怎么做的，这样，我们在写自己的测试框架的时候，也能有所借鉴。

今天，我要带你认识的是一个由 Google 开发的单元测试框架 Google Test，我们一般称它为 gtest。

关于如何安装 Google 测试框架呢，由于大家的编程环境不同，所以，你可以按照网上其他资料中给出的安装教程，结合自己的编程环境，准备好 gtest 的相关环境。

gtest 是一个 C++ 的单元测试框架，如果你不方便准备 gtest 相关环境，也不碍事儿，你可以看我下面的讲解。我的环境中，已经安装好了 gtest 的相关模块，为了简单说明 gtest 的使用与运行效果，给你准备了如下代码：

复制代码

```
#include <stdio.h>
#include <gtest/gtest.h>

// 判断一个数字 x 是否是素数
int is_prime(int x) {
    for (int i = 2; i * i < x; i++) {
        if (x % i == 0) return 0;
    }
    return 1;
}

// 第一个测试用例
```

```
TEST(test1, test_is_prime) {
    EXPECT_EQ(is_prime(3), 1);
    EXPECT_EQ(is_prime(5), 1);
    EXPECT_EQ(is_prime(7), 1);
}

// 第二个测试用例
TEST(test2, test_is_prime) {
    EXPECT_EQ(is_prime(4), 0);
    EXPECT_EQ(is_prime(0), 0);
    EXPECT_EQ(is_prime(1), 0);
}

int main() {
    return RUN_ALL_TESTS();
}
```

就是一份简单的 gtest 的使用代码。代码中，包含了 gtest/gtest.h 头文件以后，程序就具备了一些魔法效果，下面让我来给你具体讲一讲。

从程序的结构上来说，先是一个 is_prime 函数，接下来是两段以 TEST 作为开头的代码（这两段代码的作用，咱们稍后做介绍），最后是一个主函数。主函数中原本 return 0 的位置，替换成了 return RUN_ALL_TESTS()，也就是说，主函数中只执行了一个 RUN_ALL_TESTS 函数，而这个 RUN_ALL_TESTS 函数有什么作用呢？咱们来看一下程序的具体输出：

复制代码

```
[==========] Running 2 tests from 1 test suite.
[----------] Global test environment set-up.
[----------] 2 tests from test_is_prime
[ RUN      ] test_is_prime.test1
[       OK ] test_is_prime.test1 (1 ms)
[ RUN      ] test_is_prime.test2
gtest_test.cpp:25: Failure
Expected equality of these values:
  is_prime(4)
    Which is: 1
  0
gtest_test.cpp:26: Failure
Expected equality of these values:
  is_prime(0)
    Which is: 1
  0
gtest_test.cpp:27: Failure
Expected equality of these values:
```

```
19    is_prime(1)
20      Which is: 1
21    0
22  [  FAILED  ] test_is_prime.test2 (0 ms)
23  [----------] 2 tests from test_is_prime (1 ms total)
24
25  [----------] Global test environment tear-down
26  [==========] 2 tests from 1 test suite ran. (1 ms total)
27  [  PASSED  ] 1 test.
28  [  FAILED  ] 1 test, listed below:
29  [  FAILED  ] test_is_prime.test2
30
31   1 FAILED TEST
```

由于咱们的环境有所不同，所以你环境中的输出内容，可能和我这个输出结果略有差别，不过，这不影响我们接下来的讨论。

先看输出内容的第 4 行和第 6 行，意思是说，执行测试用例 test_is_prime.test1 和 test_is_prime.test2，这不就是上面两个以 TEST 开头的两段代码相关的输出内容么？

接下来从第 7 行到第 21 行是一段报错信息，意思就是说 is_prime(4)，is_prime(0) 与 is_prime(1) 函数返回值错误，也就意味着 is_prime 函数实现有错误，这段错误所涉及的信息，在源代码中的第二个测试用例中有涉及。

以上就是我们对 gtest 单元测试框架的一个感性认识，从这些感性认知，我们逐渐走向理性层面，逐步展开属于我们自己的思考。

对于 gtest 的三个思考

面对刚才的演示代码和输出内容，你可能会产生如下三个问题：

1. 源代码中的 EXPECT_EQ 是做什么的？
2. 以 TEST 开头的代码段，和我们学习的函数很不一样，那它究竟是什么？
3. 主函数中只调用了 RUN_ALL_TESTS 函数，为什么好像是执行了程序中所有的 TEST 代码段？这个功能是怎么实现的？

第一个问题不难，查看相关 gtest 的文档资料，你就可以知道，EXPECT_EQ 是 gtest 里面自带的宏，主要作用是判断传入的两部分的值是否相等。如果不相等，就会产生类似于输出

内容中第 7 行到第 21 行的输出内容。

第二个问题，以 TEST 开头的这段代码，明显不符合我们对 C 语言的语法认知，我们确实没有见过不用规定返回值类型，也不用规定参数类型的函数定义方式。关于 TEST 究竟是个什么的问题，更加合理的猜测，就是 TEST 实际上是一个宏。

我们来回顾一下宏的作用，宏就是做简单的替换。正是因为 TEST(test_is_prime, test1) 这段代码实际上是一个宏，所以展开以后，和后面的大括号中的内容一起组成了一段合法的代码内容，这样理解，原本的代码内容也就解释得通了。

而 TEST 宏展开以后会被替换成什么内容呢？关于这个问题，我留下充足的时间请你去思考，同时，包括第三个问题的答案，都留作今天的思考题。我希望你认真想一想，并将你的思考结果写在留言区，我们一起讨论。

下面这段代码，就是作为我们后续的项目测试代码，对于这份源代码，我们不会对它做任何的改动，所以我建议你把代码内容保存下来，以备咱们后面课程练习使用。

复制代码

```
#include <stdio.h>
#include "geek_test.h" // 替换掉原 gtest/gtest.h 头文件

// 判断一个数字 x 是否是素数
int is_prime(int x) {
    for (int i = 2; i * i < x; i++) {
        if (x % i == 0) return 0;
    }
    return 1;
}

// 第一个测试用例
TEST(test1, test_is_prime) {
    EXPECT_EQ(is_prime(3), 1);
    EXPECT_EQ(is_prime(5), 1);
    EXPECT_EQ(is_prime(7), 1);
}

// 第二个测试用例
TEST(test2, test_is_prime) {
    EXPECT_EQ(is_prime(4), 0);
    EXPECT_EQ(is_prime(0), 0);
    EXPECT_EQ(is_prime(1), 0);
}

```

```
26 int main() {
27     return RUN_ALL_TESTS();
28 }
```

我们后续的目标，就是开发一个自己的测试框架，替换掉上述代码中的 gtest/gtest.h 头文件，使得程序具有和之前类似的功能和输出内容。整个过程，我们不会改动上述代码中的内容，这一点，请你一定要牢记。

从这一课开始，我希望你能脱离我的指导和提示，可以独立完成咱们每节课的任务，这也是为了检验你这段时间以来的学习成果。当然，如果你实在想不出答案，也可以回来看看我的做法，以供参考。

课程小结

最后呢，我来给你做一下今天的课程小结：

1. 测试行为，不是测试工程师的专属，你应该把它作为一个开发工程师的习惯。
2. 单元测试属于白盒测试范畴，Google 的 gtest 就是一种辅助我们编写单元测试的框架。
3. gtest 中的 TEST 本质上是一个宏，而这个宏应该展开成怎样的代码内容，还需要你认真思考，这个思考过程对你来说是很有价值的。

好了，今天就到这里了，我是胡光，我们下期见。

上一篇　做好闭环（四）：二分答案算法的代码统一结构

下一篇　27 | 牛刀小试（下）：实现一个自己的测试框架

精选留言 (2)

写留言

胖胖胖
2020-03-20

提问：
我去官方的github（https://github.com/google/googletest）上下了包，解压后，在readme里面看见：
If you use Windows and have Visual Studio installed, a `gtest.sln` file and several `.vcproj` files will be created. You can then build them using Visual...

展开

作者回复: 现在 VS 下面也支持 cmake 编译，你可以试着在 VS 下面使用 cmake 编译工程。

一步

2020-03-19

写单元测试是一个程序员的职业素养

展开