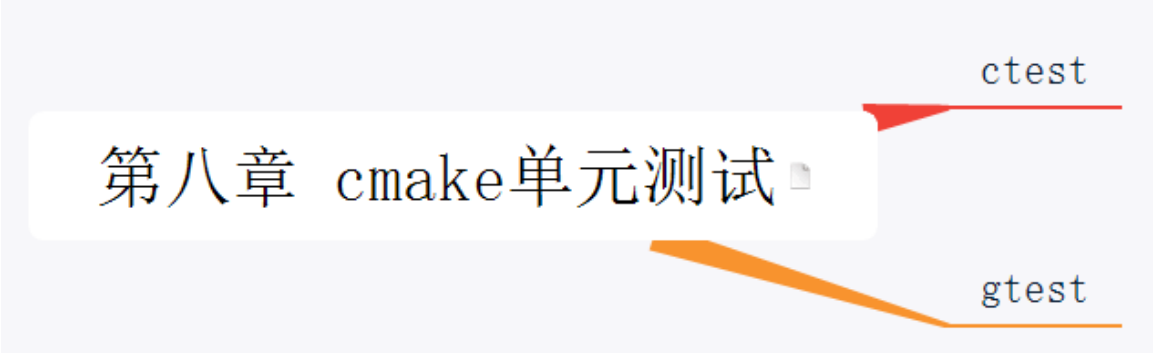
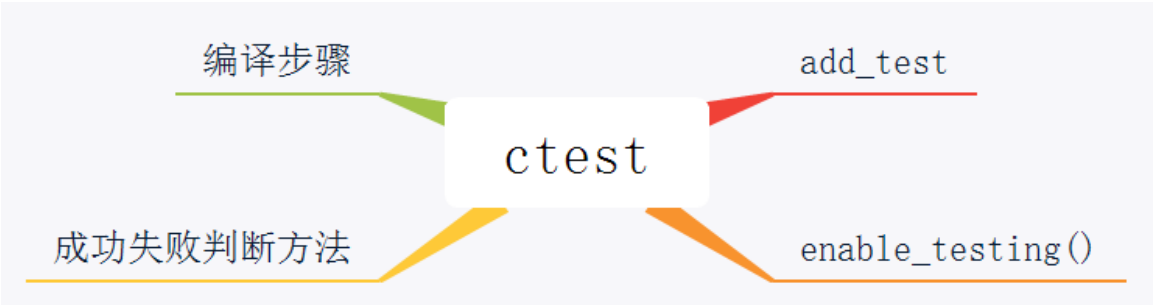
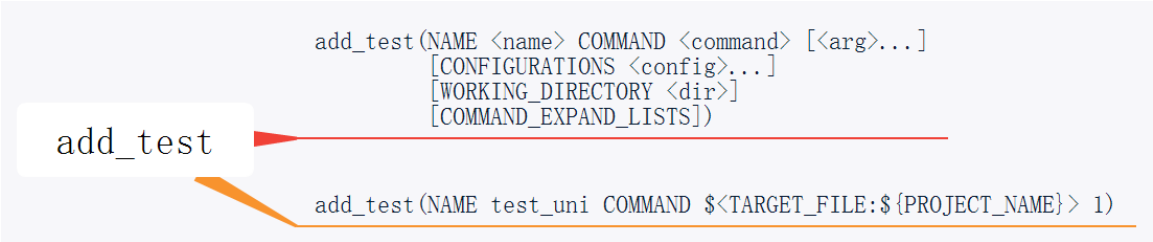
第八章 cmake单元测试



1. ctest



1.1. add_test



1.1.1. add_test(NAME <name> COMMAND <command> [<arg>...]

[CONFIGURATIONS <config>...]

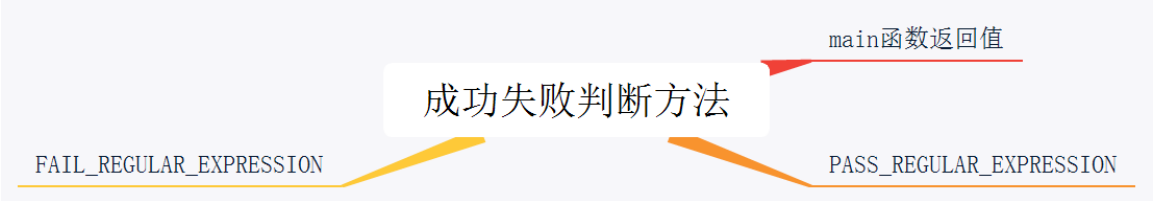
[WORKING_DIRECTORY <dir>]

[COMMAND_EXPAND_LISTS])

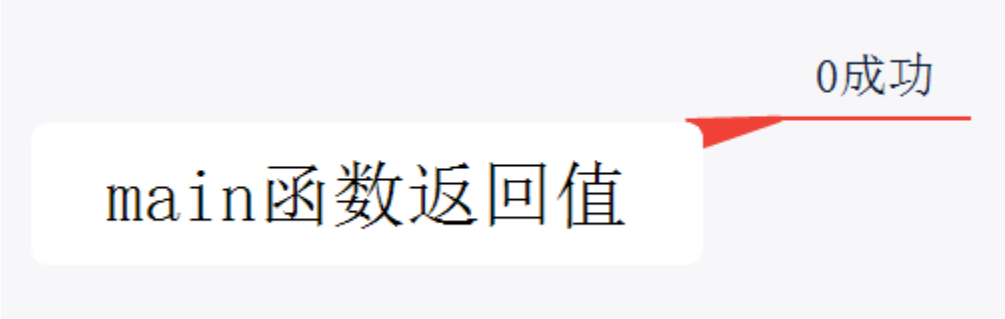
1.1.2. add_test(NAME test_uni COMMAND $<TARGET_FILE:${PROJECT_NAME}> 1)

1.2. enable_testing()

1.3. 成功失败判断方法



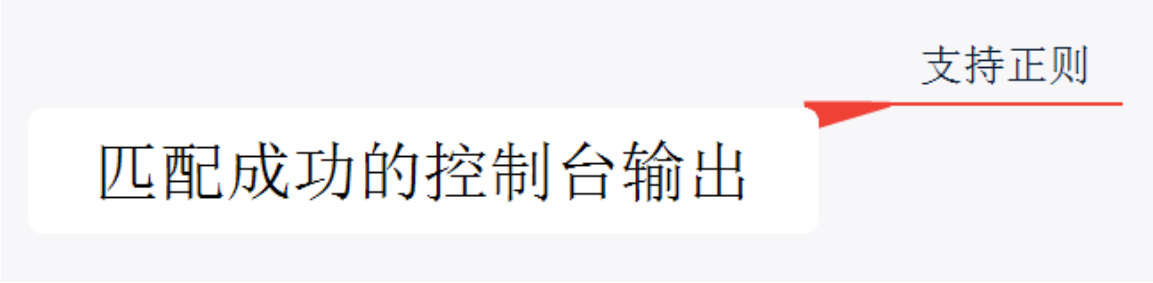
1.3.1. main函数返回值



0成功

1.3.2. PASS_REGULAR_EXPRESSION

匹配成功的控制台输出

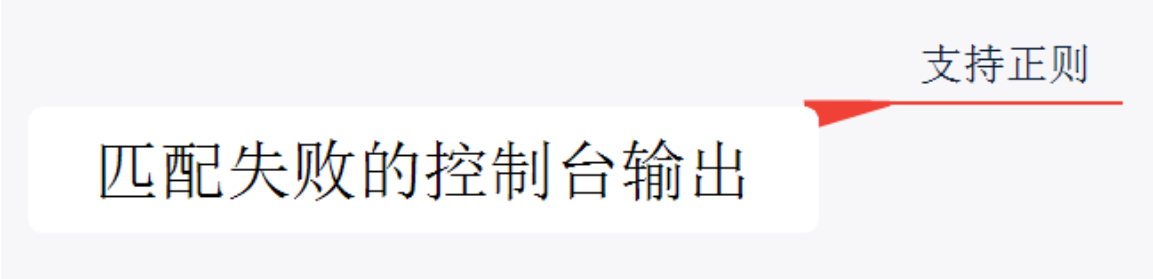


支持正则

```
set_tests_properties(test
  PROPERTIES PASS_REGULAR_EXPRESSION "99"
  )
```

1.3.3. FAIL_REGULAR_EXPRESSION

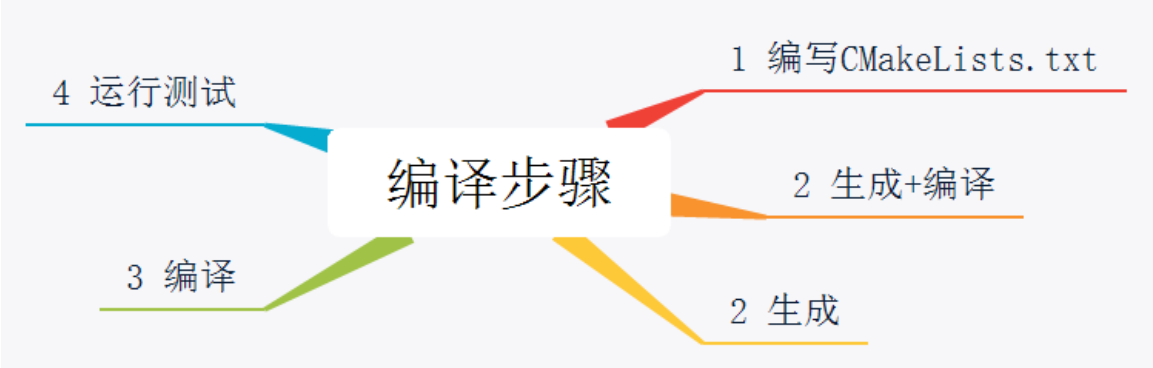
匹配失败的控制台输出



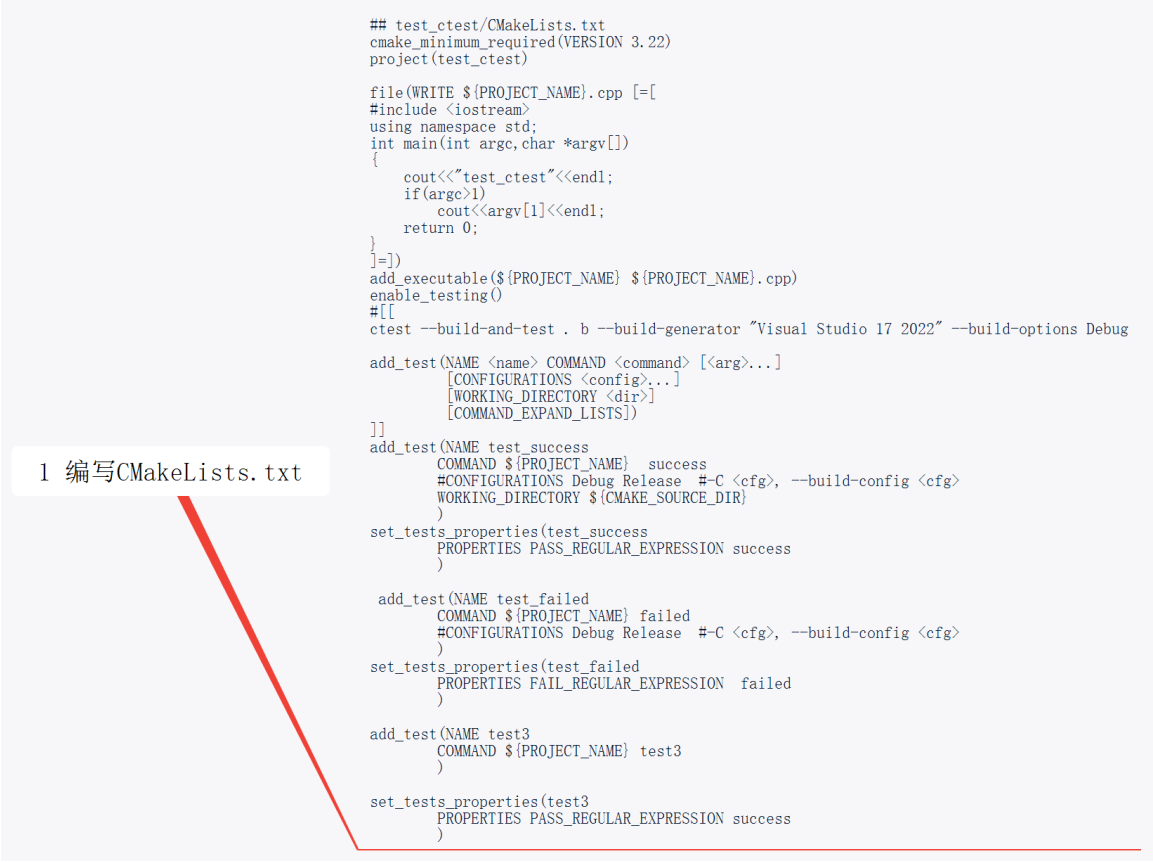
支持正则

```
set_tests_properties(test
  PROPERTIES FAIL_REGULAR_EXPRESSION "fail"
)
```

1.4. 编译步骤



1.4.1. 1 编写CMakeLists.txt




```
## test_ctest/CMakeLists.txt
cmake_minimum_required(VERSION 3.22)
project(test_ctest)

file(WRITE ${PROJECT_NAME}.cpp [=[
#include <iostream>
using namespace std;
int main(int argc,char *argv[])
{
   cout<<"test_ctest"<<endl;
   if(argc>1)
      cout<<argv[1]<<endl;
   return 0;
```

```
}
]=])
add_executable(${PROJECT_NAME} ${PROJECT_NAME}.cpp)
enable_testing()
#[[
ctest --build-and-test . b --build-generator "Visual Studio 17 2022" --build-
options Debug

add_test(NAME <name> COMMAND <command> [<arg>...]
        [CONFIGURATIONS <config>...]
        [WORKING_DIRECTORY <dir>]
        [COMMAND_EXPAND_LISTS])
]]
add_test(NAME test_success
      COMMAND ${PROJECT_NAME}  success
      #CONFIGURATIONS Debug Release  #-C <cfg>, --build-config <cfg>
      WORKING_DIRECTORY ${CMAKE_SOURCE_DIR}
      )
set_tests_properties(test_success
      PROPERTIES PASS_REGULAR_EXPRESSION success
      )

 add_test(NAME test_failed
      COMMAND ${PROJECT_NAME} failed
      #CONFIGURATIONS Debug Release  #-C <cfg>, --build-config <cfg>
      )
set_tests_properties(test_failed
      PROPERTIES FAIL_REGULAR_EXPRESSION  failed
      )

add_test(NAME test3
```

COMMAND ${PROJECT_NAME} test3

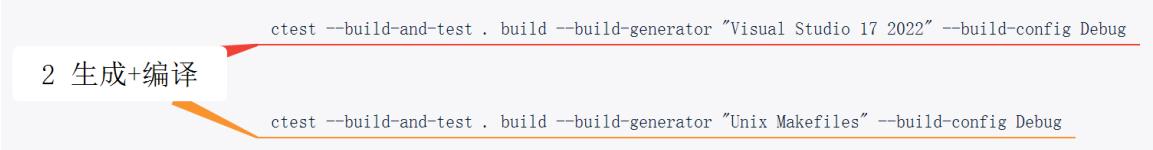
)

set_tests_properties(test3

PROPERTIES PASS_REGULAR_EXPRESSION success

)

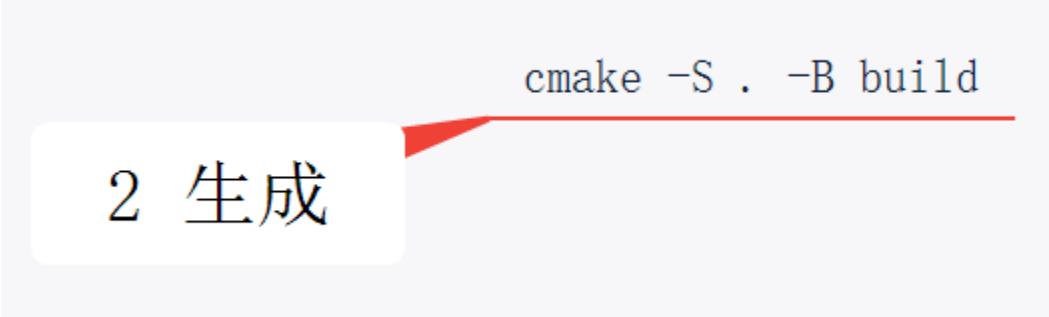
1.4.2. 2 生成+编译



ctest --build-and-test . build --build-generator "Visual Studio 17 2022" --build-config Debug

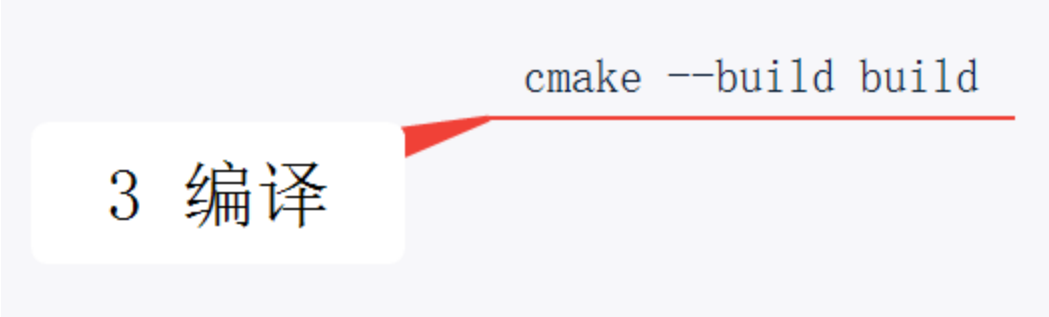
ctest --build-and-test . build --build-generator "Unix Makefiles" --build-config Debug

1.4.3. 2 生成



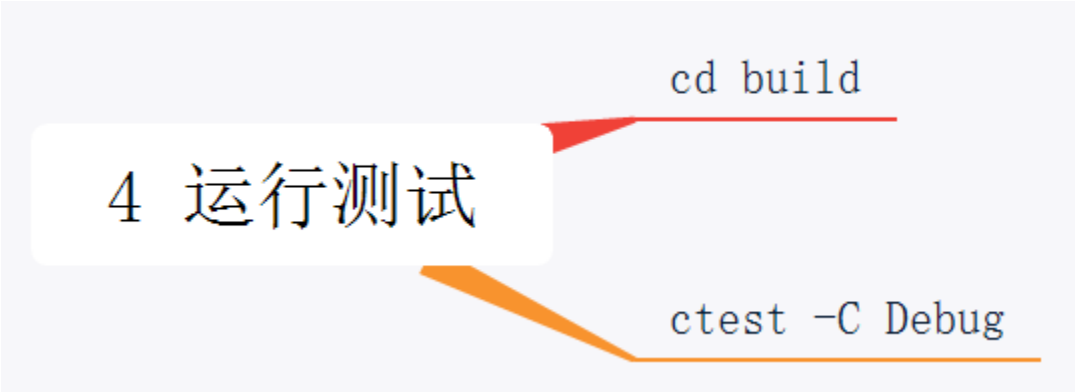
cmake -S . -B build

1.4.4. 3 编译



cmake --build build

1.4.5. 4 运行测试



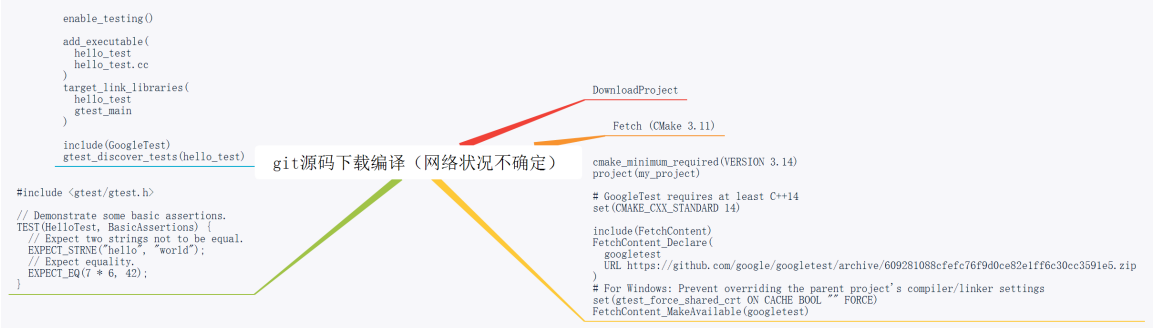
cd build

ctest -C Debug

2. gtest

2.1. 安装方法

2.1.1. git源码下载编译（网络状况不确定）



DownloadProject

Fetch (CMake 3.11)

cmake_minimum_required(VERSION 3.14)
project(my_project)

 # GoogleTest requires at least C++14
set(CMAKE_CXX_STANDARD 14)

include(FetchContent)
FetchContent_Declare(
googletest
URL
https://github.com/google/googletest/archive/609281088cfefc76f9d0ce82e1ff6c30cc3591e5.zip
)
 # For Windows: Prevent overriding the parent project's compiler/linker settings

```
set(gtest_force_shared_crt ON CACHE BOOL "" FORCE)
FetchContent_MakeAvailable(googletest)

#include <gtest/gtest.h>

// Demonstrate some basic assertions.
TEST(HelloTest, BasicAssertions) {
  // Expect two strings not to be equal.
  EXPECT_STRNE("hello", "world");
  // Expect equality.
  EXPECT_EQ(7 * 6, 42);
}

enable_testing()

add_executable(
  hello_test
  hello_test.cc
)
target_link_libraries(
  hello_test
  gtest_main
)

include(GoogleTest)
gtest_discover_tests(hello_test)
```

2.1.2. 直接下载发布库和头文件

2.1.3. 手动下载源码编译安装

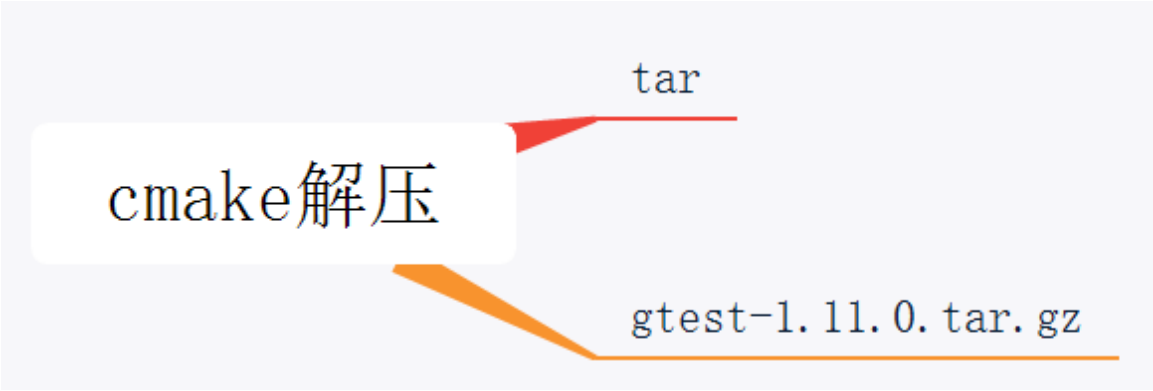
2.2. execute_process

2.2.1. execute_process(COMMAND <cmd1> [<arguments>]

[COMMAND <cmd2> [<arguments>]]...

[WORKING_DIRECTORY <directory>]

2.2.2. cmake解压



tar

gtest-1.11.0.tar.gz

2.2.3. cmake配置

2.2.4. cmake编译

2.2.5. cmake安装

2.3. FetchContent_Declare

2.3.1. FetchContent_Declare(

googletest

GIT_REPOSITORY https://github.com/google/googletest.git

GIT_TAG 703bd9caab50b139428cea1aaff9974ebee5742e # release-1.10.0

)

FetchContent_Declare(

myCompanyIcons

URL https://intranet.mycompany.com/assets/iconset_1.12.tar.gz

URL_HASH MD5=5588a7b18261c20068beabfb4f530b87

)

FetchContent_Declare(

myCompanyCertificates

SVN_REPOSITORY svn+ssh://svn.mycompany.com/srv/svn/trunk/certs

SVN_REVISION -r12345

)

2.4. 简单测试

2.4.1. TEST(TestSuiteName, TestName) {

... test body ...

}

2.4.2. // Tests factorial of 0.

TEST(FactorialTest, HandlesZeroInput) {

EXPECT_EQ(Factorial(0), 1);

}

// Tests factorial of positive numbers.

TEST(FactorialTest, HandlesPositiveInput) {

EXPECT_EQ(Factorial(1), 1);

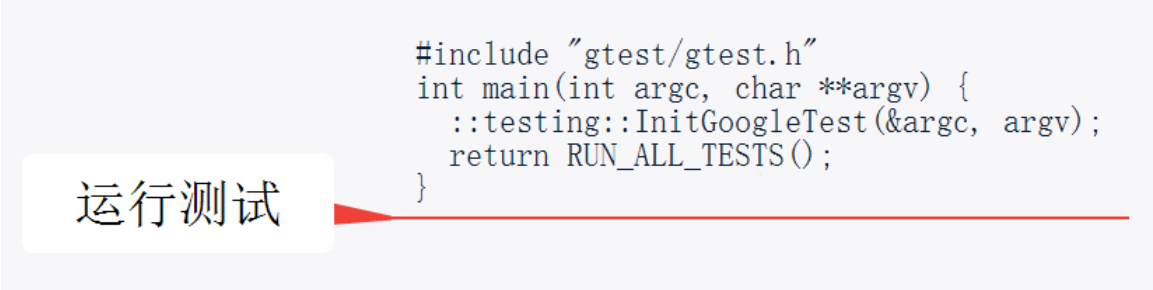
EXPECT_EQ(Factorial(2), 2);

EXPECT_EQ(Factorial(3), 6);

EXPECT_EQ(Factorial(8), 40320);

}

2.5. 运行测试



2.5.1. #include "gtest/gtest.h"

```
int main(int argc, char **argv) {
  ::testing::InitGoogleTest(&argc, argv);
  return RUN_ALL_TESTS();
}
```