

一、实验目的

□ □ □ □ □ □ □ □ □ □

实验二 自动化测试

班级	学号	姓名	指导老师
软件工程□□□□班	□□□□□□□□□□	董佩杰	毛锐

一、实验目的

- 掌握 Maven 的安装及其与 IDE 的集成。
- 利用 JUnit 进行单元测试。
- 掌握 Spring 中常用注解：@Autowired、@Qualifier、@Scope、@Transactional
- 掌握 Spring 中套件测试和参数化测试的方法。
- 掌握 Redis 的安装和使用。
- 基于 Mockito 的覆盖率测试对单元测试覆盖分析，提升测试质量。
- 利用 TestNG 进行自动化测试的配置和执行

二、实验步骤

- `libgdiplus.dll`的安装及其与`libjpeg.dll`的集成。
- 两种方法，`libgdiplus.dll`本地安装，`libjpeg.dll`集成安装
- 从[libgdiplus.dll](#)下载`libgdiplus.dll`压缩包，把`libgdiplus.dll`压缩包解压到一个物理路径
- 记录`libgdiplus.dll`文件所在目录。
- 设置环境变量`LIBGDIPLUS_PATH`。
- 在“系统”菜单“系统”的子项“系统”中选择“系统”，单击“系统”标签，添加`LIBGDIPLUS_PATH`，即选择`libgdiplus.dll`或`libjpeg.dll`，单击打开，就完成了`libgdiplus.dll`的安装。

二、实验步骤

□ □ □ □ □ □ □ 单元测试

□ □ 实习题一

利用 `TestRunner` 生成测试用例的框架，在框架中设计测试代码，完成对下面类 `TestRunner`

10

□ □ □ □ □ □ □ □ □ □ 求解最大公约数和最小公倍数以及查找功能

11

☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☒ ☐

求最大公约数

□ □ □ □

☐ ☐ ☐ ☐

□ □ □ □

□ □ □ □

□ □ □ □ □ □

□ □ □ □

☐ ☐ ☐ ☐

1

□ □ □ □ □ □ □ □

最小公倍数

1

[illegible]

二、实验步骤

[illegible]

测试类：

[illegible][illegible]

3

对 进行测试

11

[illegible][illegible]

二、实验步骤

[illegible]

Diagram illustrating a sequence of colored squares (yellow, green, red, blue, black) arranged in three rows, likely representing a data structure or a sequence of operations.

The diagram consists of three rows of colored squares. The first row contains 5 yellow squares. The second row contains 10 squares: 5 green, 3 red, and 2 blue. The third row contains 18 squares: 5 blue, 10 black, and 3 blue. A single black square is positioned below the first square of the third row.

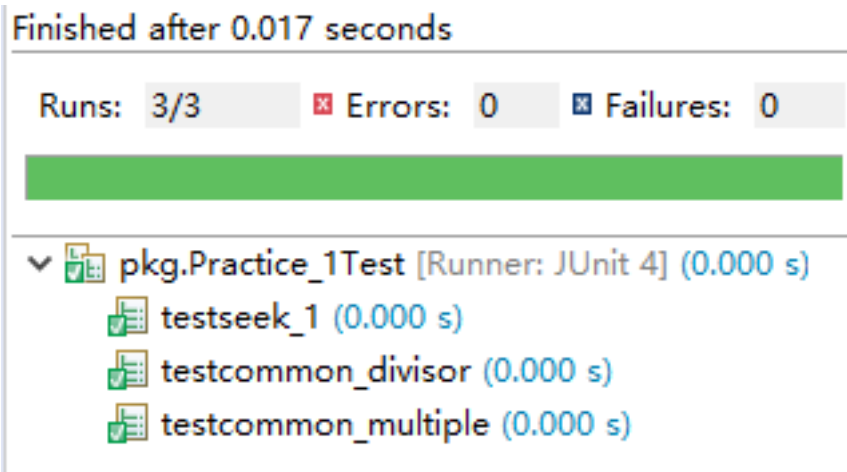
测试结果：

□ □ 实习题二

设计判断一个数是不是素数的程序，用基本断言类型实现测试，并用 `□ □ □ □ □ □` 初始化测试环境。参数化测试数据

判断素数类：

二、实验步骤



□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

□ □ □ □ □ □ □ □ □ □ 判断一个证书是否为素数

□ □

□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

□ □

□ □

□ □ □ □ □ □ □ □ □ □ □ □ □

□ □ □ □ □ □ □ □ □ □ □ □ □ □

□

□

□ □ □ □ □ □ □ □ □ □ □

□

□ □

□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

□ □

□

□

测试类：

□ □ □ □ □ □ □ □ □ □ □

二、实验步骤

[illegible][illegible]

Diagram illustrating the growth of a sequence of sets $S_0, S_1, S_2, S_3, S_4, S_5, S_6$. Each set is represented by a row of squares. The first four squares in each row are green, and the remaining squares are grey. The number of green squares is 4 for S_0 through S_4 , 5 for S_5 , and 6 for S_6 . The number of grey squares increases from 0 in S_0 to 1 in S_1 , 2 in S_2 , 3 in S_3 , 4 in S_4 , 5 in S_5 , and 6 in S_6 . The total number of squares in each row is 4, 5, 6, 7, 8, 9, and 10 respectively.

参数化测试，测试素数

A diagram showing a sequence of 25 colored squares arranged in four rows. Row 1: 7 yellow squares. Row 2: 10 green squares, 3 red squares, 5 blue squares, 2 black squares, 5 green squares, 5 black squares. Row 3: 5 black squares, 3 green squares, 5 blue squares, 3 black squares. Row 4: 1 black square.

Category	Count
Yellow	5
Red	6
Blue	4
Grey	2
Green	5
Black	8
Total	100

1

二、实验步骤

Diagram illustrating a sequence of blocks (represented by colored squares) arranged in four rows. The blocks are colored green, black, and blue. The sequence is as follows:

- Row 1: 10 blocks (Green, Black, Green, Black, Green, Black, Green, Black, Green, Black).
- Row 2: 12 blocks (Green, Black, Green, Black, Green, Black, Green, Black, Green, Black, Green, Black).
- Row 3: 12 blocks (Green, Black, Green, Black, Green, Black, Green, Black, Green, Black, Green, Black).
- Row 4: 8 blocks (Green, Black, Green, Black, Green, Black, Green, Black).

The blocks are arranged in a staggered pattern, with some blocks in the second row aligned with the first, and others shifted to the right. The blue blocks are located in the fourth row, starting from the 10th block of the sequence.

0
 1

A diagram illustrating a sequence of 30 colored squares arranged in three rows. The top row contains 4 yellow squares. The middle row contains 10 squares: 6 green, 4 red, and 10 blue. The bottom row contains 26 squares: 10 blue, 10 black, and 6 blue. A small black square is positioned below the first square of the bottom row.

测试结果：

使用□□□□□对以上两个测试类进行测试：

[illegible]

```

    0 0 0
    0 0 0 0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0 使用 0 0 0 0 进行批量测试，此处测试 0 0 0 0 0 0 0 0 和 0 0 0 0 0 的测试类
    0 0

```

□

测试结果为：

二、实验步骤



□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □



二、实验步骤

□ □ 实习题三

下面是使用 `is_holiday` 来跟踪一年中的那些天是节假日的程序。

111

□ □ □ □ □ □ □ □ □ 判断一个数是否存在于这个列表

11

[illegible]

□ □ □ □ □ □ □ □ □ □

□ □ 集合中假日是随机设定的，可根据今年的情况自行调整

□ □

1

1

[illegible]

1

Category	Count
Green	6
Red	6
Blue	6
Grey	2

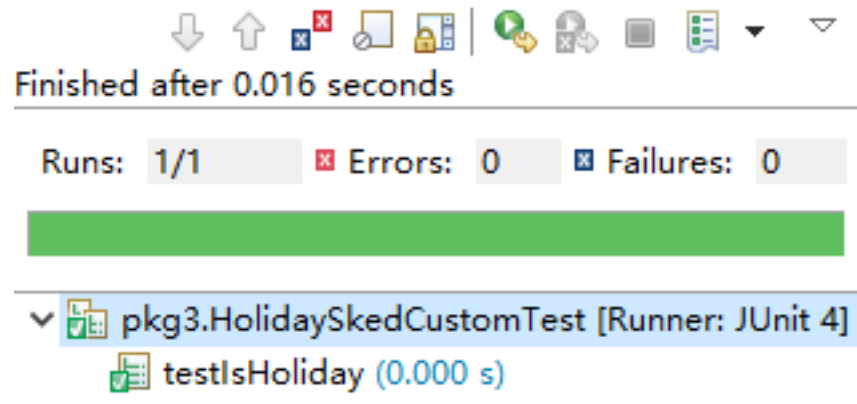
[illegible]

1

二、实验步骤

11

测试结果：

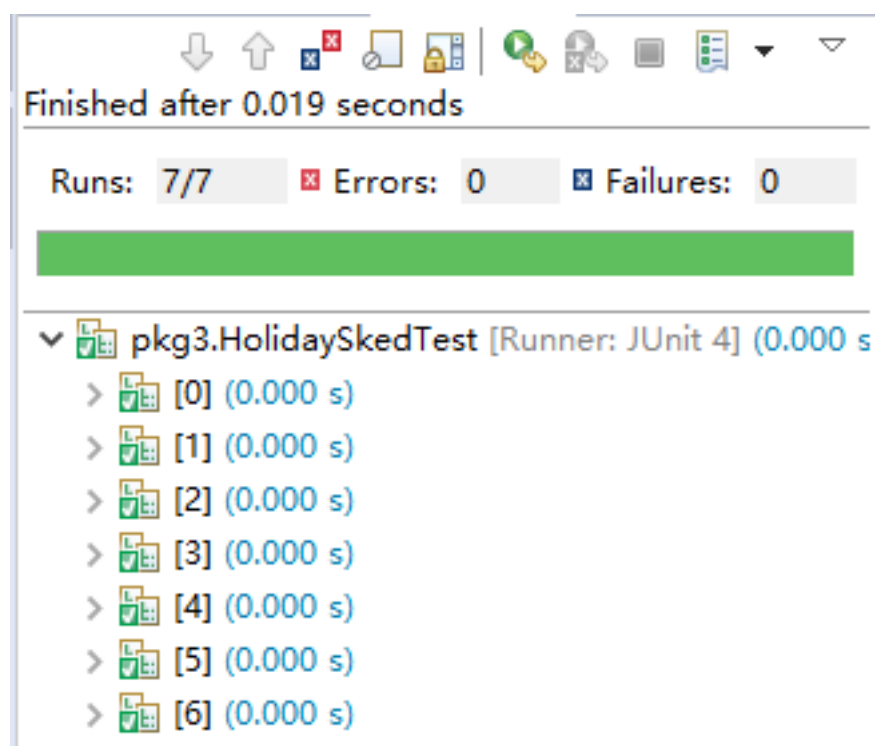


□ □ □ 用参数化的方法重新设计本题和实验题 □ 的测试用例。

二、实验步骤

11

二、实验步骤



□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

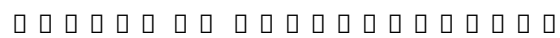
二、实验步骤

[illegible]

A diagram showing a sequence of 30 colored squares arranged in four rows. Row 1: 7 yellow squares. Row 2: 6 green squares, 4 red squares, 6 blue squares, 4 black squares, 5 green squares, 5 black squares. Row 3: 5 black squares, 3 green squares, 8 blue squares, 4 black squares. Row 4: 1 black square.

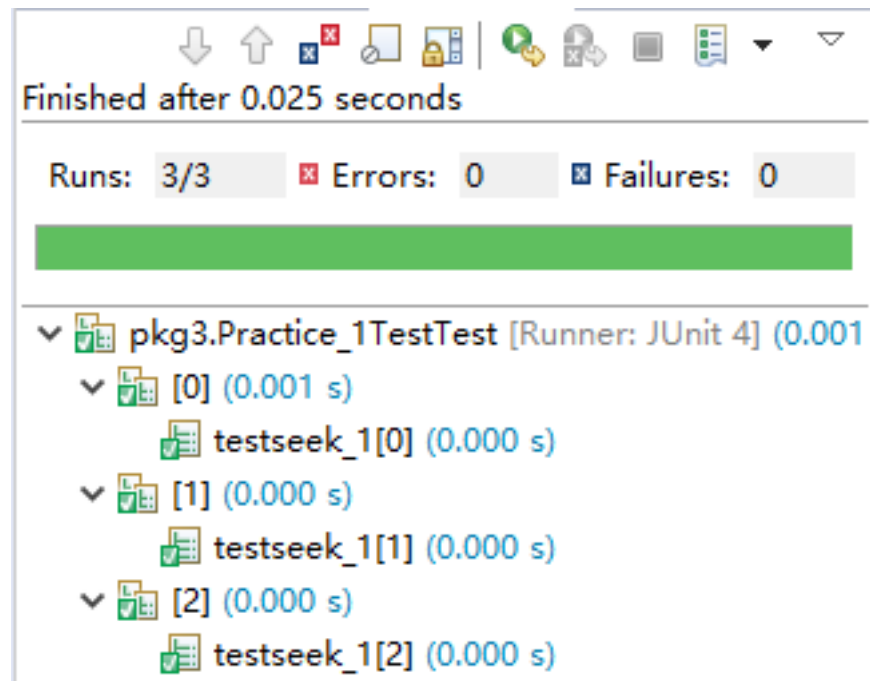
11

测试结果：



二、实验步骤

第二部分：查找功能



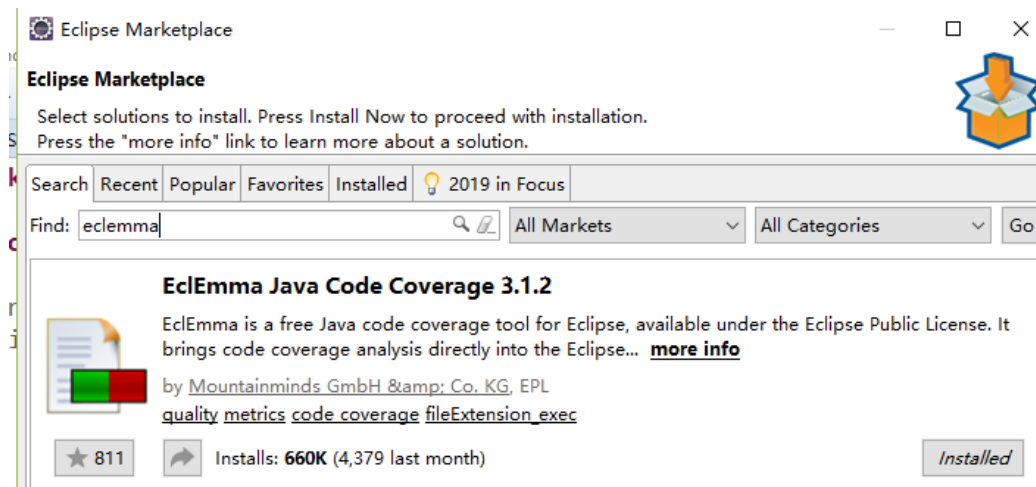
测试结果：



二、实验步骤

□ □ 实习题四

对课本□ □ □ 页的示例程序利用语句覆盖、判定□ 条件覆盖、条件组合及路径覆盖的角度分别设计测试用例进行自动化



□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

已安装好□ □ □ □ □ □

□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

□ □ 实习题五

针对静态测试试验中选择排序，三角形问题，隔一日问题的代码，或者自己开发实现的综合性□ □ □ □ 项目，在所学测

针对每一个类使用□ □ □ □ □ 进行自动化测试，然后使用□ □ □ □ □ 方法调用所有单个测试类，以下是执行的结果：

代码见附件

□ □ 实习题六

针对实验□ 的□ □ □ □ 项目，利用□ □ □ 结合□ □ □ □ □ 进行自动化测试构建运行。

通过使用□ □ □ □ □ □ 将□ □ □ □ □ □ □ □ □ 文件生成，然后运行即可。

在这个过程中遇到了一个比较大的问题，一开始将这些文件组织到一个新的工程中的时候，选择将其复制到工程中，然

二、实验步骤

▼	🟢	DateProcess	<div><div></div></div>	100.0 %	65	0	65
		● judge(int[])	<div><div></div></div>	100.0 %	26	0	26
		● myGetNextDate(int[])	<div><div></div></div>	100.0 %	36	0	36
▼	🟢	DateProcessTest	<div><div></div></div>	100.0 %	153	0	153
		● setUp()	<div><div></div></div>	100.0 %	6	0	6
		● test()	<div><div></div></div>	100.0 %	8	0	8
		● testJudge()	<div><div></div></div>	100.0 %	37	0	37
▼	🟢	SelectionSort	<div><div></div></div>	100.0 %	54	0	54
		● selectionSort(int[])	<div><div></div></div>	100.0 %	51	0	51
▼	🟢	SelectionSortTest	<div><div></div></div>	100.0 %	72	0	72
		● setUp()	<div><div></div></div>	100.0 %	6	0	6
		● test()	<div><div></div></div>	100.0 %	15	0	15
▼	🟢	Triangle	<div><div></div></div>	100.0 %	142	0	142
		● judge(int[])	<div><div></div></div>	100.0 %	37	0	37
		● judgeTriangle(int[])	<div><div></div></div>	100.0 %	51	0	51
		● selectionSort(int[])	<div><div></div></div>	100.0 %	51	0	51
▼	🟢	TriangleTest	<div><div></div></div>	100.0 %	135	0	135
		● setUp()	<div><div></div></div>	100.0 %	6	0	6
		● test()	<div><div></div></div>	100.0 %	33	0	33
		● testJudge()	<div><div></div></div>	100.0 %	7	0	7
		● testSort()	<div><div></div></div>	100.0 %	7	0	7

□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

```
Buildfile: E:\JavaSpace\exp3_pkg6\build.xml
DateProcessTest (1):
[junit] Running pkg6_test.DateProcessTest
[junit] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.109 sec
SelectionSortTest (1):
[junit] Running pkg6_test.SelectionSortTest
[junit] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.061 sec
TriangleTest (1):
[junit] Running pkg6_test.TriangleTest
[junit] Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.061 sec
TestSuites (1):
[junit] Running pkg6_test.TestSuites
[junit] Tests run: 6, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.069 sec
BUILD SUCCESSFUL
Total time: 3 seconds
```

□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

三、 总结

三、 总结

本次实习经历的时间比较长，前半部分做的比较快，后半部分由于一些环境配置还有其他奇奇怪怪的原因导致了代码无