

**2020年春季学期  
计算机学院《软件构造》课程**

**Lab 1实验报告**

|  |  |
| --- | --- |
| 姓名 | 郭茁宁 |
| 学号 | 1183710109 |
| 班号 | 1837101 |
| 电子邮件 | 847738428@qq.com |
| 手机号码 | 13905082373 |

**目录**

[1 实验目标概述 1](#_Toc29325521)

[2 实验环境配置 1](#_Toc29325522)

[3 实验过程 1](#_Toc29325523)

[3.1 Magic Squares 1](#_Toc29325524)

[3.1.1 isLegalMagicSquare() 1](#_Toc29325525)

[3.1.2 generateMagicSquare() 1](#_Toc29325526)

[3.2 Turtle Graphics 1](#_Toc29325527)

[3.2.1 Problem 1: Clone and import 2](#_Toc29325528)

[3.2.2 Problem 3: Turtle graphics and drawSquare 2](#_Toc29325529)

[3.2.3 Problem 5: Drawing polygons 2](#_Toc29325530)

[3.2.4 Problem 6: Calculating Bearings 2](#_Toc29325531)

[3.2.5 Problem 7: Convex Hulls 2](#_Toc29325532)

[3.2.6 Problem 8: Personal art 2](#_Toc29325533)

[3.2.7 Submitting 2](#_Toc29325534)

[3.3 Social Network 2](#_Toc29325535)

[3.3.1 设计/实现FriendshipGraph类 2](#_Toc29325536)

[3.3.2 设计/实现Person类 2](#_Toc29325537)

[3.3.3 设计/实现客户端代码main() 2](#_Toc29325538)

[3.3.4 设计/实现测试用例 3](#_Toc29325539)

[4 实验进度记录 3](#_Toc29325540)

[5 实验过程中遇到的困难与解决途径 3](#_Toc29325541)

[6 实验过程中收获的经验、教训、感想 3](#_Toc29325542)

[6.1 实验过程中收获的经验和教训 3](#_Toc29325543)

[6.2 针对以下方面的感受 3](#_Toc29325544)

# 实验目标概述

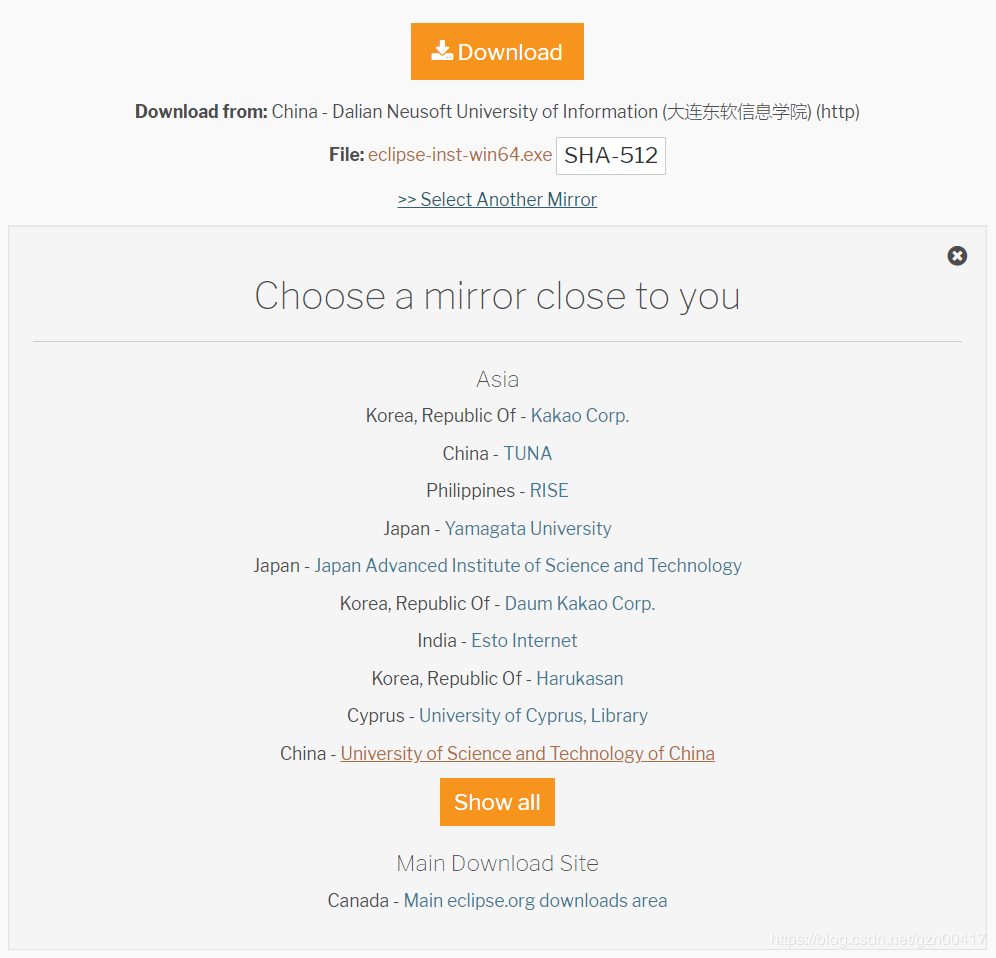
本次实验通过求解三个问题，训练基本 Java 编程技能，能够利用 Java OO 开发基本的功能模块，能够阅读理解已有代码框架并根据功能需求补全代码，能够为所开发的代码编写基本的测试程序并完成测试，初步保证所开发代码的正确性。另一方面，利用 Git 作为代码配置管理的工具，学会 Git 的基本使用方法。

* 基本的 Java OO 编程
* 基于 Eclipse IDE 进行 Java 编程
* 基于 JUnit 的测试
* 基于 Git 的代码配置管理

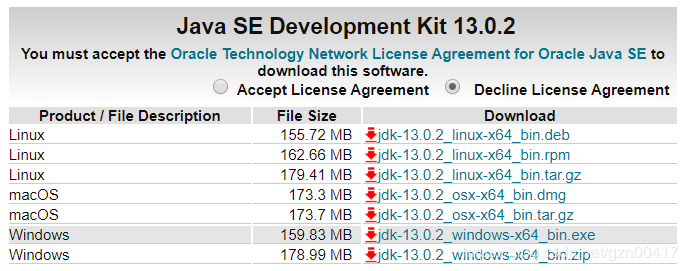
# 实验环境配置

## Java & Eclipse & Maven 使用配置方法

### 下载Eclipse

* Download 64 bit
* Select Another Mirror
  + China - University of Science and Technology of China
  + 作者选择中科大USTC镜像  
    

### 下载JDK13

* 点击 Accept License Agreement 后，点击灰条选项  
  

### 配置 JDK

* 点击 : Window -> Prefernces -> Java -> Installed JREs -> Add -> Standard VM -> 选择 JDK 目录 D:\Program Files\jdk-13.0.2 -> Finish -> 在列表勾选刚刚添加的 JDK -> Apply

### 自定义工具栏

* 点击 : Window -> Perspective -> Customize Perspective -> Tool Bar Visibility , 选择对应的选项即可

### 编辑器字体修改

* 点击：Window -> Prefernces -> General -> Appearance -> Colors and Fonts -> Java -> Java Editor Text Font
* 推荐：Consolas+小四号字体

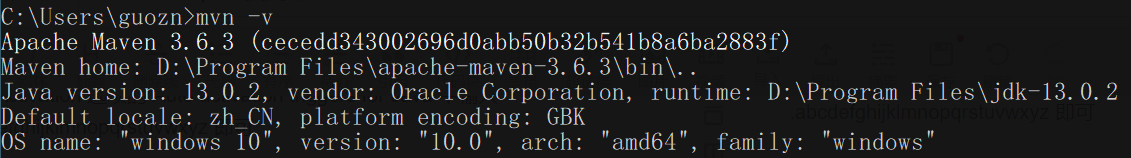
### 修改默认编码

* 点击 : Windows -> Preferences -> General -> Workspace , 在右侧的 Text file encoding 中选择 UTF-8
* 点击 : Windows -> Preferences -> General -> Content Types , 在右侧点开 Text , 选择 Java Source File , Java Properties File , JSP , 在下面的 Default encoding 输入 UTF-8 , 点击 Update
* 点击 : Windows -> Preferences -> Web -> JSP Files , 面板选择 ISO 10646/Unicode(UTF-8) 右键选择项目 -> Properties -> Resource -> 设置编码为 UTF-8

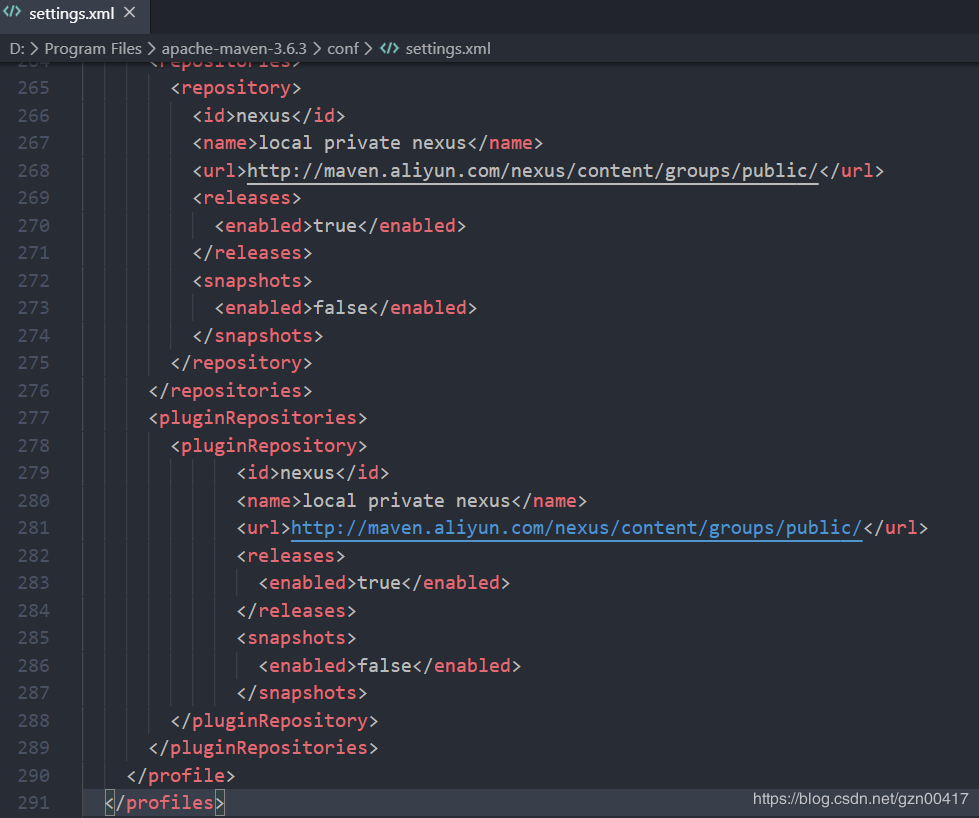
### 设置代码自动提示

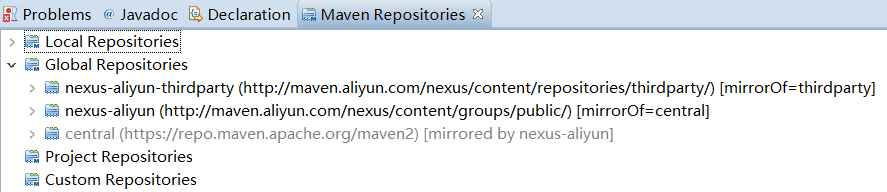
* 点击 Window -> Preferences -> Java -> Editor -> Content Assist , 在右侧的 Auto Activation 选项的 Auto activation triggers for Java 中输入 .abcdefghijklmnopqrstuvwxyz 即可

### 配置 Maven

* 首先配置系统环境变量 :
  + 右键计算机 -> 属性 -> 高级系统设置 -> 高级 -> 环境变量 , 在下方的 系统变量 添加 MAVEN*HOME , 值是 Maven 文件夹路径 , 如 : C:\MyProgram\Maven , 修改 Path 变量 , 添加 %MAVEN*HOME%\bin; , 注意分号 ; 配置完成之后查看是否成功 : 运行 CMD 输入 mvn -v  
    
* 然后配置 Eclipse :
  + 打开 Eclipse -> Window -> Preferences -> Maven -> User Settings -> Global Settings: 和 User Settings: -> Browse... , 选择 Maven 目录下的 \conf\settings.xml 文件 , 然后点击 Update Settings 按钮即可
* 添加镜像（国内阿里云镜像）
  + conf/setting.xml中插入代码
  + 在mirrors中间插入

<mirror>  
 <id>nexus-aliyun</id>  
 <mirrorOf>central</mirrorOf>  
 <name>Nexus aliyun</name>  
 <url>http://maven.aliyun.com/nexus/content/groups/public/</url>  
 </mirror>  
 <mirror>  
 <id>nexus-aliyun-thirdparty</id>  
 <mirrorOf>thirdparty</mirrorOf>  
 <name>Nexus aliyun thirdparty</name>  
 <url>http://maven.aliyun.com/nexus/content/repositories/thirdparty/</url>  
 </mirror>  
 - 在profiles中间插入

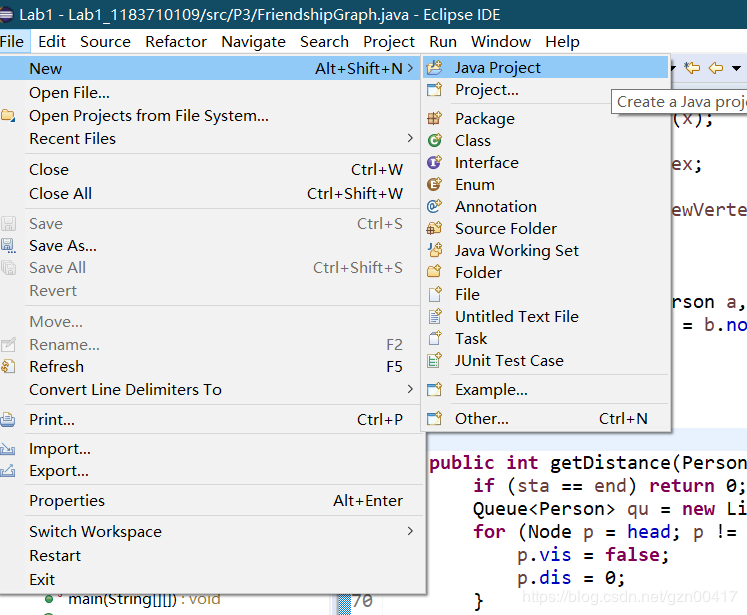


* 查看是否成功
  + Window -> Show View -> Other -> Maven -> Maven Repositories -> Open
  + 若打开后更改，可以点击Maven Repositories查看视图右边的Reload setting.xml  
    

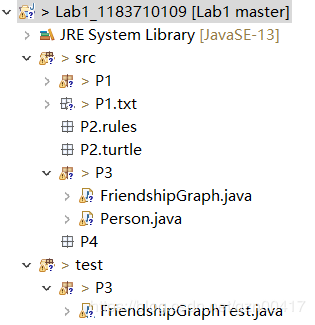
## JUnit

* JUnit is a simple framework to write repeatable tests. It is an instance of the xUnit architecture for unit testing frameworks.
* 使用 JUnit 为 Java 程序编写测试代码并执行测试

### 新建Java项目

* File -> New -> Java Project
* 根据自己的需求配置  
  

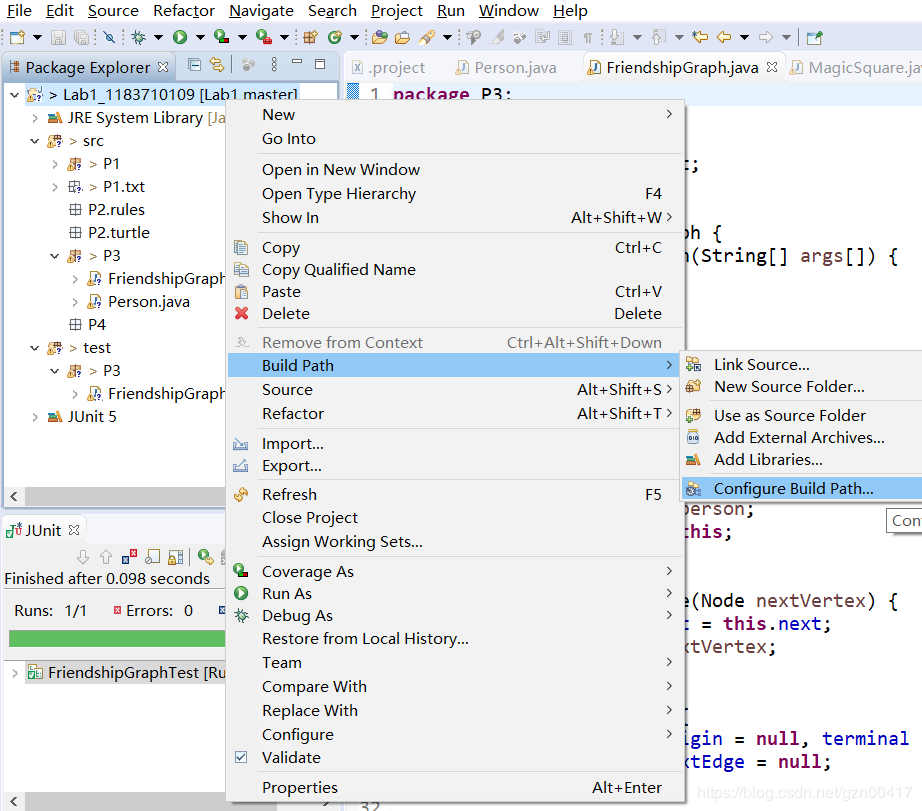
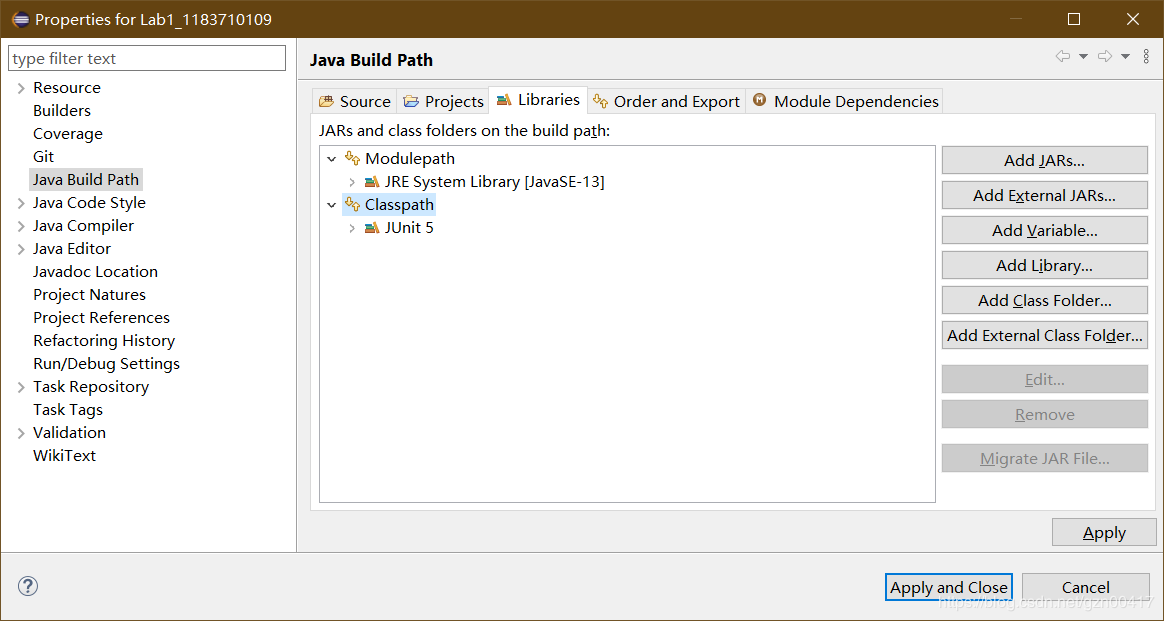
### 编写代码 & 打包

* 准备好src文件夹和test文件夹
* src文件夹中
  + 文件夹名与test中相同（推荐）
  + package名与test中相同（必须）  
    
* 示例：

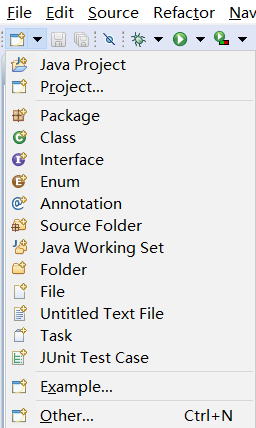
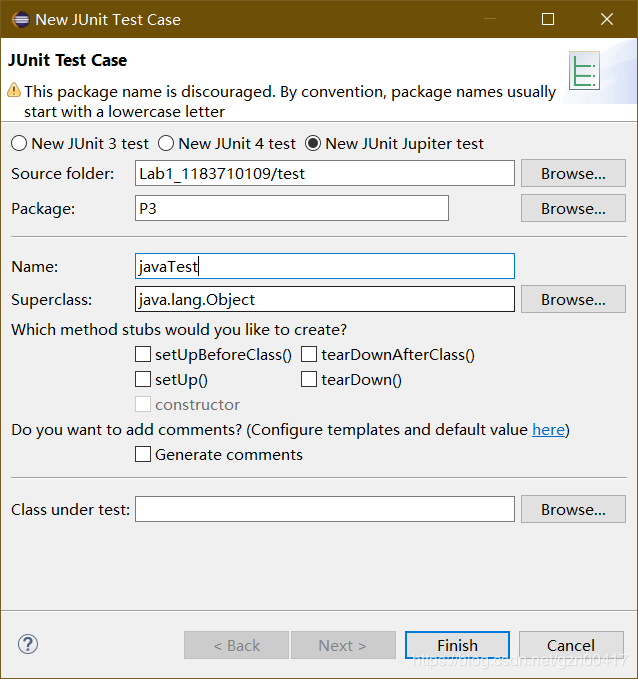
// ./src/P3/xxx.java  
package P3;  
...

// ./test/P3/xxxTest.java  
package P3;  
...

### 将JUnit加入Library

* 项目名字（右键）-> Build Path -> Configure Build Path  
  
* Library（中间上面） -> classpath（左边） -> Add Libraries（右边）  
  

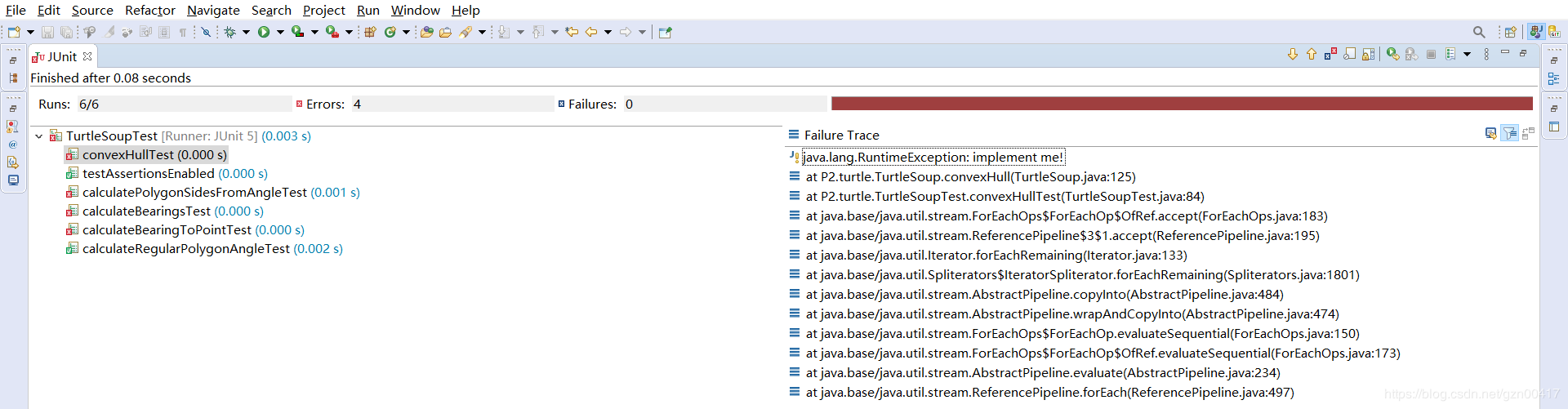
### 创建Junit Test Case

* 和创建project类似的方法创建JUnit Test Case  
  
* 弹框选项
  + New Junit Jupiter Test
  + Source folder：./.../test
  + package P3（与之前写好的类的package一样）  
    
* 创建好的示例：

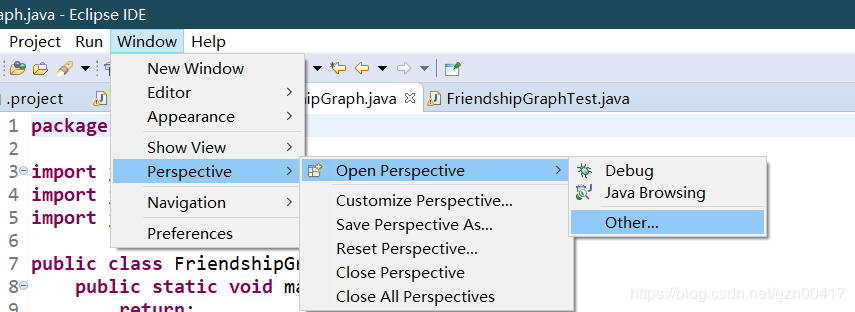
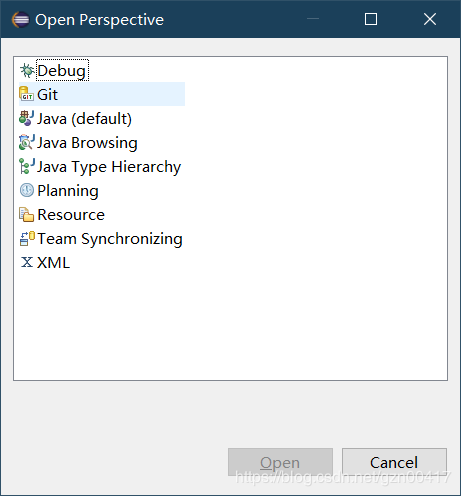
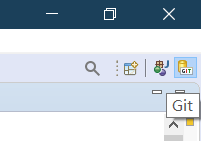
package P3;  
import static org.junit.jupiter.api.Assertions.\*;  
import org.junit.jupiter.api.Test;  
class Test1 {  
 @Test  
 void test() {  
 assertEquals(2.0, Calc(1.0, 1.0), 0);  
 fail("Not yet implemented");  
 }  
}

* 更改void test()中的内容即可！

### 查看测试结果

* 双击JUnit视图放大查看
* 每一个@Test为一组测试
* 每组测试会显示测试结果，正确为绿色，错误为红色，还有其他错误提示  
  

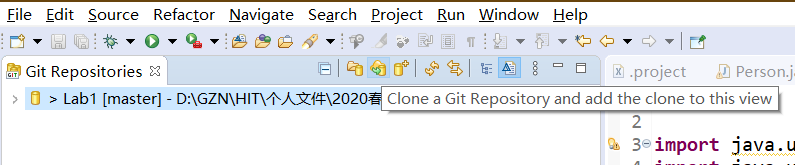
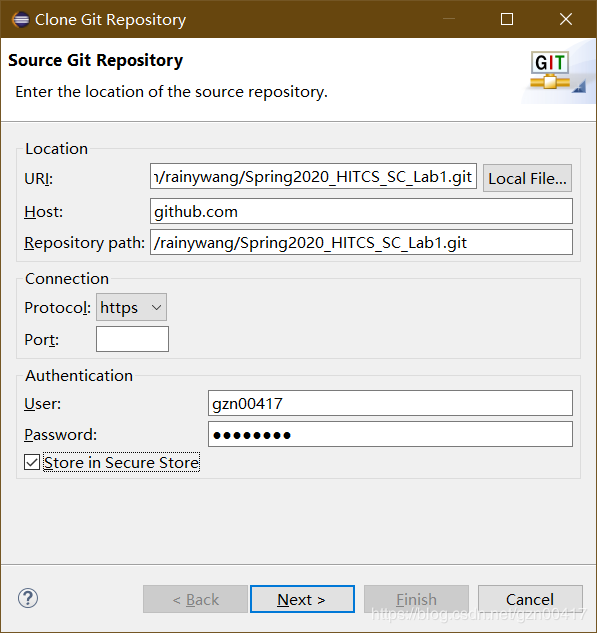
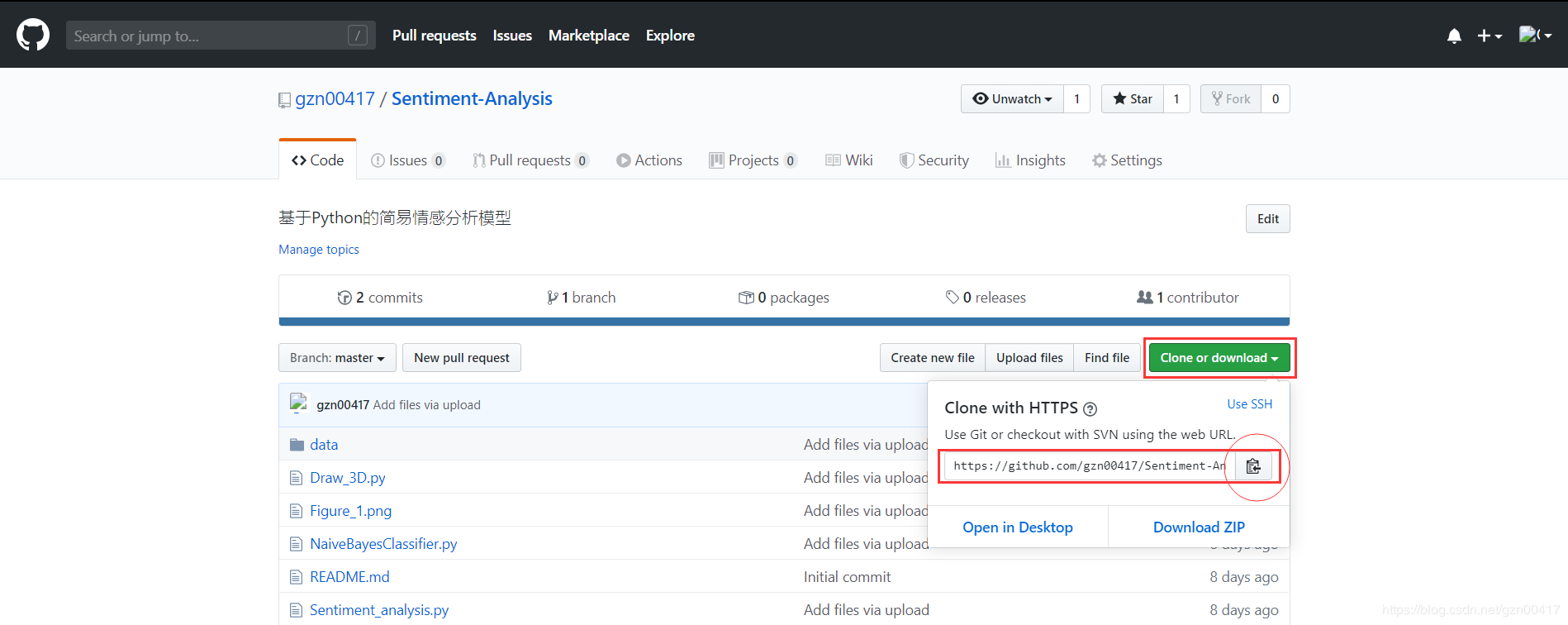
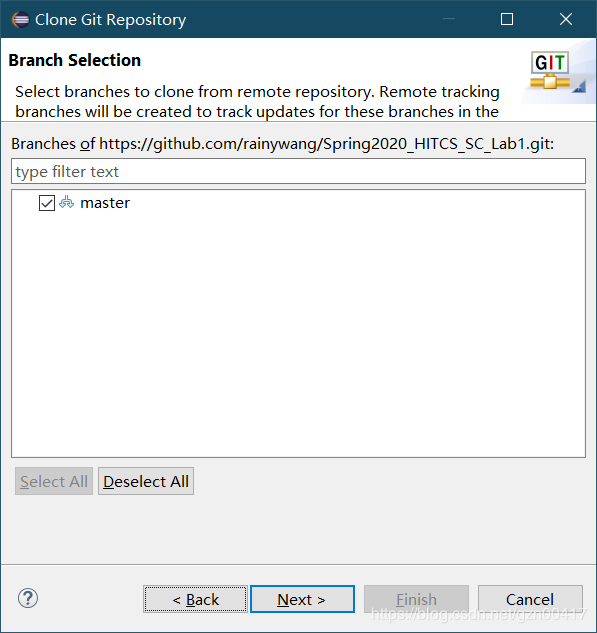
## 在Eclipse中打开Git视图

* Window -> Perspective -> Other Perspective -> Other...
* Git -> Open  
    
  
* 切换窗口：右上角视图 Java/Git  
  

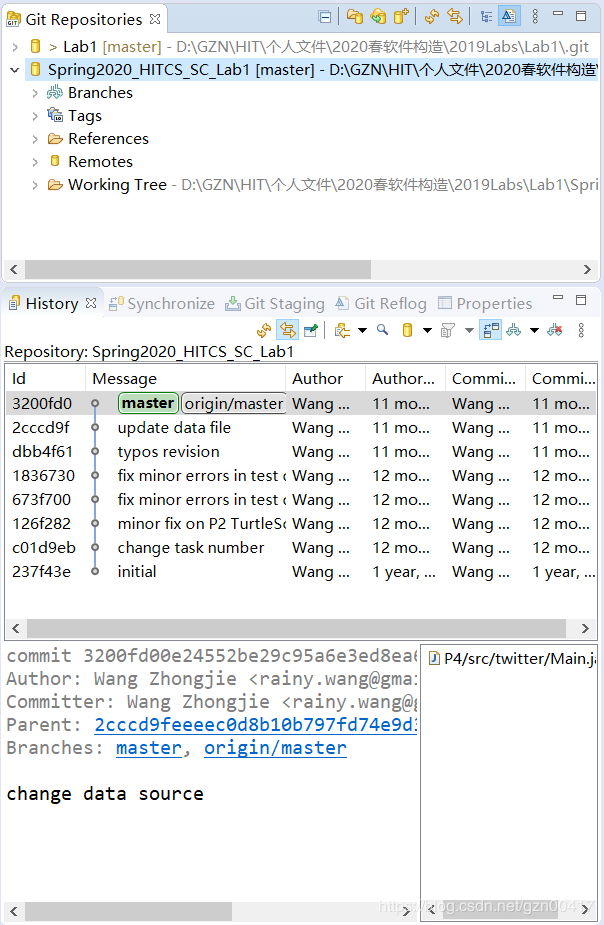
### 管理Git Repositories

* 该功能窗口有
  + Add exist...
  + Clone...
  + Create New...  
    fig:

### 下载Github上的Repositories

* 点击Clone a Git Repository...  
  
* 填写URL
  + Host和Repository Path会自动生成
  + URL从Github链接的**绿色按键**查找（下下图说明）
* 填写自己的GitHub账户密码  
  
* （接上面）在GitHub中复制URL
  + Clone or Download（绿色按键）
  + 点击右边按键复制URL  
    
* 选择master  
  
* Next -> Browse -> 选择目录 -> Finish

### Clone完成效果



## Git使用

# 实验过程

请仔细对照实验手册，针对四个问题中的每一项任务，在下面各节中记录你的实验过程、阐述你的设计思路和问题求解思路，可辅之以示意图或关键源代码加以说明（但无需把你的源代码全部粘贴过来！）。

为了条理清晰，可根据需要在各节增加三级标题。

## Magic Squares

幻方是一个有n\*n个不同数字、且每行、每列和斜线上都有相同的和的方形结构。要求写出程序判断输入一个矩阵是否是幻方，并且构造幻方。

### isLegalMagicSquare()

按步骤给出你的设计和实现思路/过程/结果。

### generateMagicSquare()

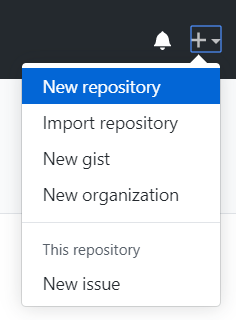
按步骤给出你的设计和实现思路/过程/结果。

## Turtle Graphics

在这里简要概述你对该任务的理解。

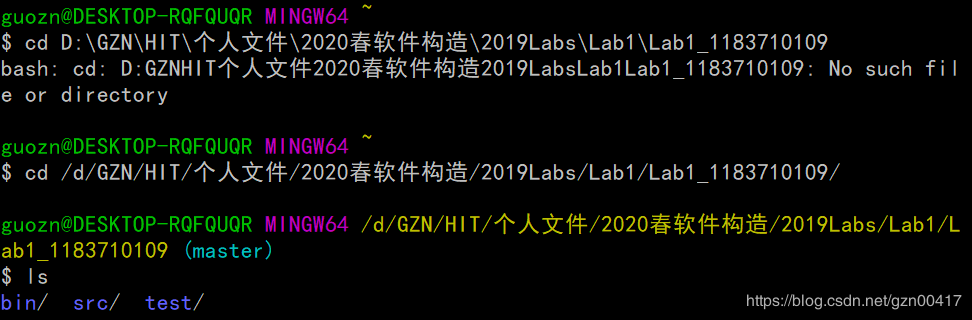
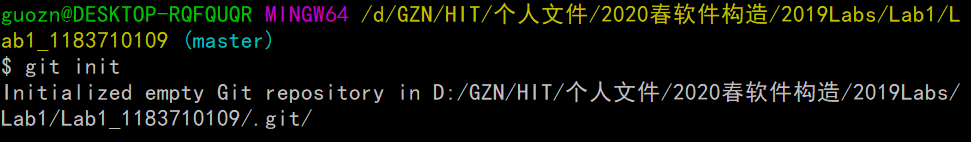
### Problem 1: Clone and import

#### 注册Github并新建仓库

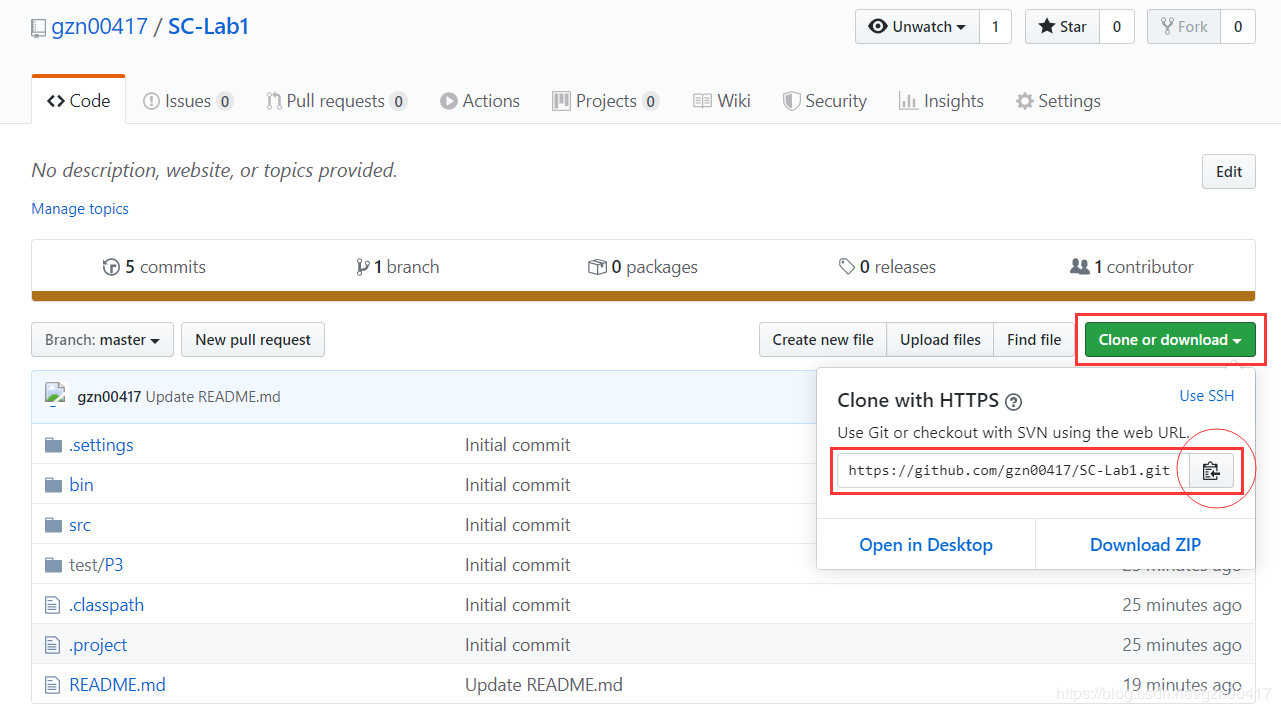
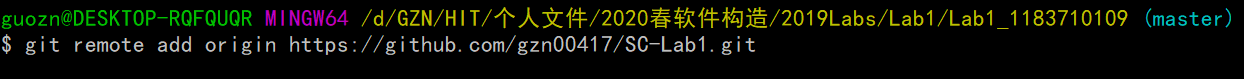
* GitHub网站右上角New repository  
  

#### 初始化git

打开 Git Bash（推荐）/ cmd

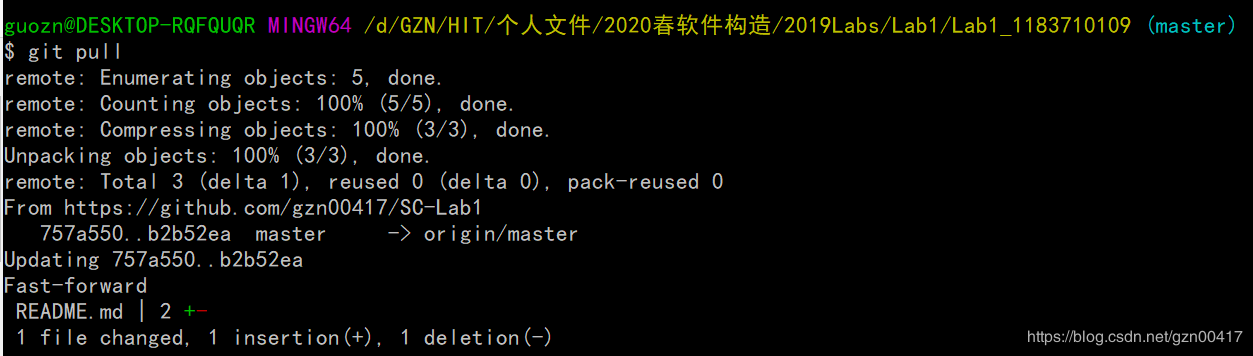
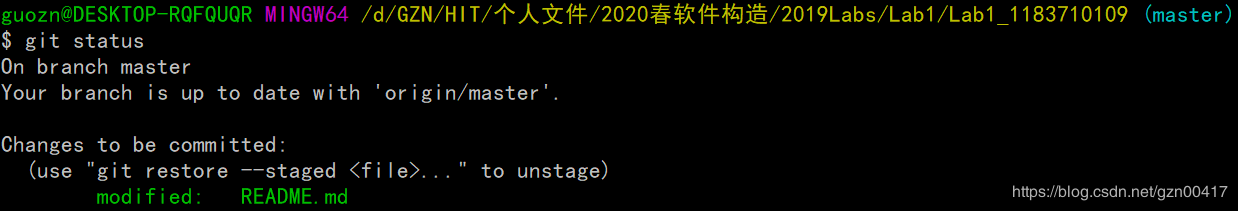
* cd /d/.../
* **注意：斜杠与windows不同，并且没有c/d盘的冒号，最后有斜杠**
* 可以用ls查看一下该目录文件  
  
* 输入git init初始化  
  

#### 添加远程仓库URL

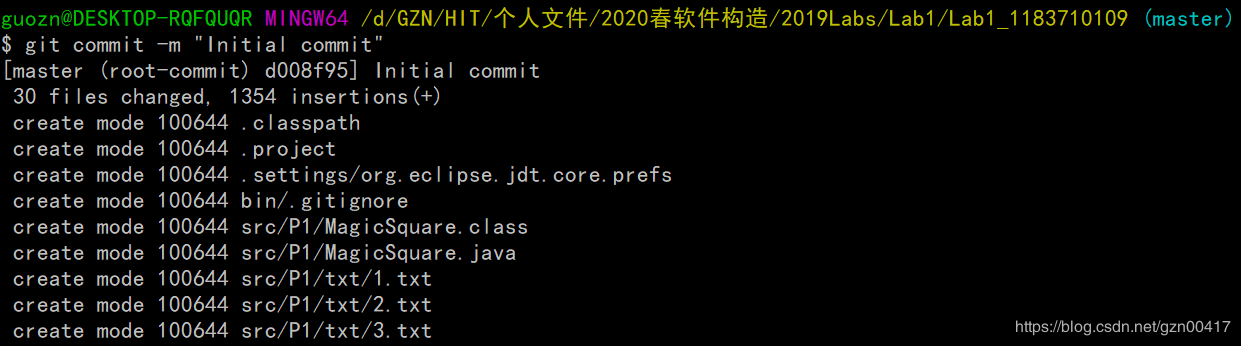
* 查找URL，图示位置（绿色按键-->复制URL）  
  
* 添加远程仓库链接，命令：git remote add origin https://github.com/.../x...x.git
  + 最后的链接就是复制来的URL  
    

#### 添加上传文件

**注意：若新建仓库有README.md，需要先在本地同步（下载）**

* + 命令：git pull
  + 若无效果则尝试：git pull --rebase origin master  
    
* 命令：git add .（后面是前后有空格的点，表示所有文件）
  + 也可以单独上传修改文件，如README.md
  + git add README.md
* 命令：git status可以查看状态  
  

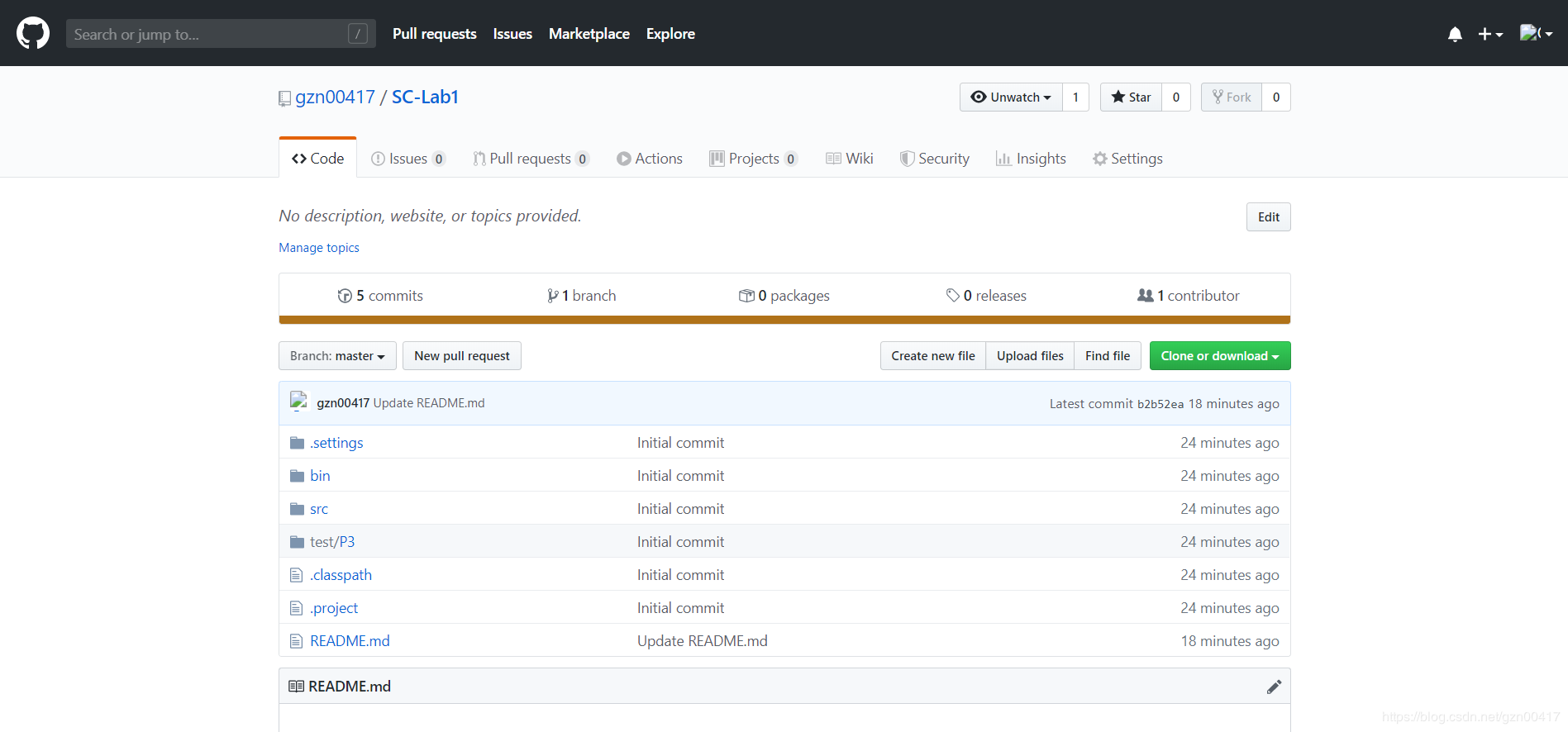
#### 添加修改日志

* 命令：git commit -m "Initial commit"
  + 引号内内容可以随意修改；
  + 引号内的内容只会称为有修改过的文件的新日志；
* 命令（在-m前加入-a）：git commit -a -m "Update 2.4"
  + 任何是否被git add的文件都将被commit  
    

#### 上传push

* 命令：git push -u origin master
* 多次使用后可以省略后面参数，只用git push

#### 查看上传情况

* F5刷新GitHub网页  
  

#### 下载/同步

* 命令：git pull 或 git pull --rebase origin master

#### 其他命令

* 查看提交历史记录：git log
* 查看更改：git diff
  + 加参数--staged即使提交准确记录

### Problem 3: Turtle graphics and drawSquare

### Problem 5: Drawing polygons

### Problem 6: Calculating Bearings

### Problem 7: Convex Hulls

### Problem 8: Personal art

### Submitting

如何通过Git提交当前版本到GitHub上你的Lab1仓库。

## Social Network

在这里简要概述你对该任务的理解。

### 设计/实现FriendshipGraph类

给出你的设计和实现思路/过程/结果。

### 设计/实现Person类

给出你的设计和实现思路/过程/结果。

### 设计/实现客户端代码main()

给出你的设计和实现思路/过程/结果。

### 设计/实现测试用例

给出你的设计和实现思路/过程/结果。

# 实验进度记录

请使用表格方式记录你的进度情况，以超过半小时的连续编程时间为一行。

每次结束编程时，请向该表格中增加一行。不要事后胡乱填写。

不要嫌烦，该表格可帮助你汇总你在每个任务上付出的时间和精力，发现自己不擅长的任务，后续有意识的弥补。

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 时间段 | 任务 | 实际完成情况 |
| 2020-02-03 | 下午 | Magicsquare | 按计划完成 |
| 2020-02-04 | 下午 | 建立本地git仓库并链接 | 遇到困难，但按计划完成 |
| 2020-02-04 | 晚上 | 配置、生成项目 | 按计划完成 |
| 2020-02-05 | 早上 | Turtle问题第二个小问 | 按计划完成 |
| 2020-02-05 | 下午 | Socialnetwork的两个类 | 按计划完成 |
| 2020-02-06 | 晚上 | 解决3.2.5 | 按计划完成 |
| 2020-02-08 | 晚上 | TurtleSoupTest调试 | 按计划完成，时间较长 |
| 2020-02-24 | 下午 | 迁移项目，调整工作目录和路径 | 完成 |
| 2020-02-26 | 上午 | 写报告，配置环境部分和问题1概述 | 完成 |
|  |  |  | 遇到困难，未完成 |

# 实验过程中遇到的困难与解决途径

|  |  |
| --- | --- |
| 遇到的困难 | 解决途径 |
| Java使用Eclipse出现editor does not contain a main type | 原因：当前源代码没有整合进package，即没有build path  部分已经build path，但新写的代码没有  解决方案  若已经有build path（有红褐色标记），则先Remove；没有（全白）则忽略  src（右键）-> build path -> remove from build path  此时，src文件全部变白  然后点击：src（右键）-> build path -> use as source folder |
| Exception in thread "main" java.lang.Error: Unresolved compilation problem，编译器/package不一致 | 检查编译器  打开cmd检查版本  项目名（右键）-> propriety -> Java Compiler  Enable…打勾，Compiler…level 与java和javac一致  可以把Use…勾去掉进行修改  若无误，则非编译器问题  发现package名字有红线\*\*（重要特征）\*\*  发现在build path之后不一致，统一package完成build path请见  解决  修改代码中package名字\*\*（每一个包中的代码）\*\*  再次运行check  注意！！还需要更改同一个package下所有代码的部分import（旧package名） |
| Exception in thread "main" java.io.FileNotFoundException: xxx.xxx (系统找不到指定的路径。) | 该情况为读取文件路径错误  src里的代码，在我们的直观感觉可以用相对路径./xxx  然而有时候行，有时候不行  作者确实都经历过  因此要加上以src为起始的路径  例如代码在src/P1里，读取的文件在src/P1/a里  写./a/xxx要碰运气  写src/P1/a/xxx才是最稳的 |

# 实验过程中收获的经验、教训、感想

## 实验过程中收获的经验和教训

## 针对以下方面的感受

1. Java编程语言是否对你的口味？
   1. 符合，Java的OOP非常顺手，又相近于之前熟悉的C++
2. 关于Eclipse IDE；
   1. 不够美观
   2. 经过一定装饰后还是不够美观
   3. 调试、Junit都很好用
   4. 相对vscode太丑
3. 关于Git和GitHub；
   1. 能掌握Git基本功能
   2. GitHub真香
4. 关于CMU和MIT的作业；
   1. 语言问题，有些地方理解比较困难
5. 关于本实验的工作量、难度、deadline；
   1. 工作量非常大
   2. 难度很大
   3. Deadline适中
6. 关于初接触“软件构造”课程；
7. 疫情期间，只能远程授课，个人在家里完成实验任务，你对该学习方式有什么想法？