实验课程名称：__软件工程基础实验_____

实验项目名称	Git 实战			实验成绩	
实 验 者	亢柏涵	专业班级	软件 1703	组　　别	
同 组 者				实验日期	2019.6.13

第一部分：实验预习报告（包括实验目的、意义，实验基本原理与方法，主要仪器设备及耗材，实验方案与技术路线等）

一、实验目的与意义

1) 熟练掌握 git 的基本指令和分支管理指令；

2）掌握 git 支持软件配置管理的核心机理；

二、实验内容及要求

1 安装 Git

(1) 本地机器上安装 git 如果在 linux 系统下安装 git，给出安装命令和安装后的 运行界面； 如果在 windows 下安装 git，给出安装的 git 版本 号和在本地机器上安装 git 后的运行界面（给出主要界面即 可）；

(2) 给出自己在 github 上申请的帐号名称和本次实验中涉及 的项目的 URL 地址。 给出 github 网站上你的账号信息和项目信息的截图。2 Git 操作过程

(1)仓库创建与提交

(2)分支管理

(3)远程分支管理

3 在 Eclipse 中安装和使用 Git Plugin

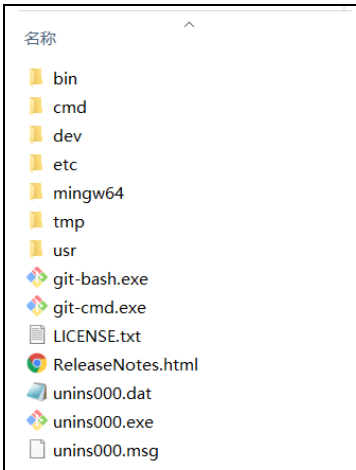
Eclipse 中 Git plugin 的安装和配置

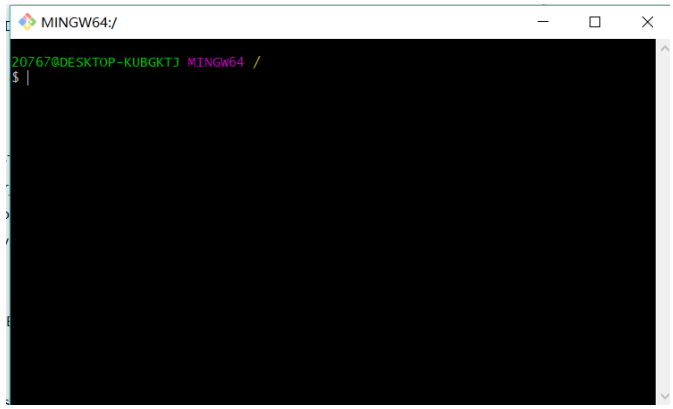
三、实验基本原理与方法

Git 是一个开源的分布式版本控制系统，用于敏捷高效地处理任何或小或大的项目。Git 是 Linus Torvalds 为了帮助管理 Linux 内核开发而开发的一个开放源码的版本控制软件。Git 与常用的版本控制工具 CVS, Subversion 等不同，它采用了分布式版本库的方式，不必服务器端软件支持。

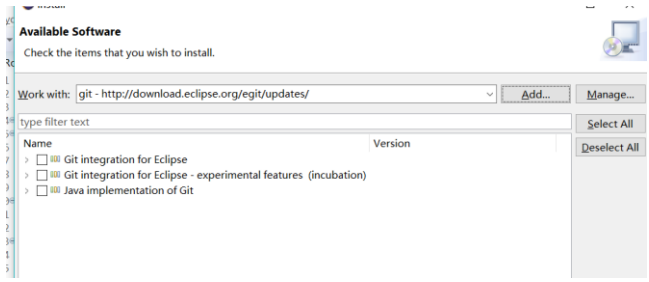
四、实验方案与技术路线

1.本次实验我在 windows 环境下使用 git，首先在官网 https://git-scm.com/下载安并装 git 后打开 git-batch.exe，界面如图

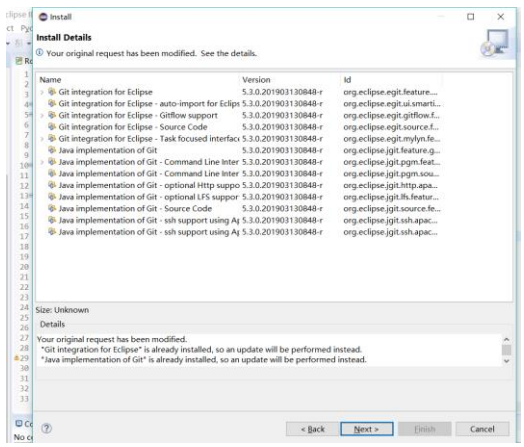




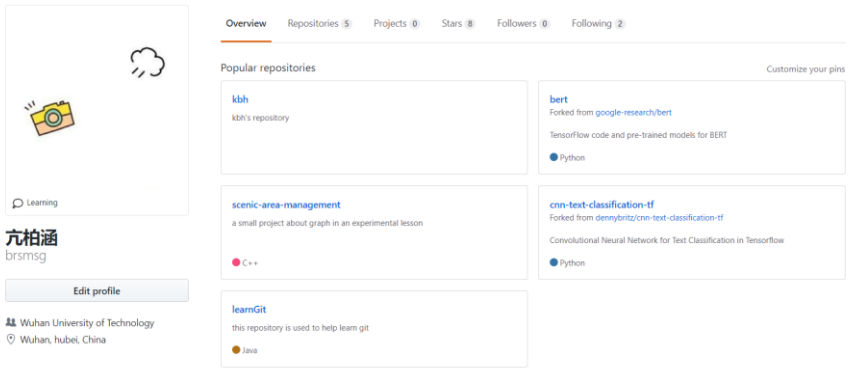
2.我们的 Eclipse 已经自带了 git，如果没有也可以通过如下方法安装：help->install new software 输入如下 url



三个全选，然后点 add next 即可

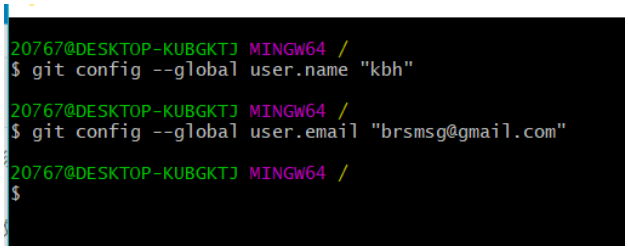


3.在 github 上注册账号，我之前已经注册，附上主页截图和 url：https://github.com/brsmsg

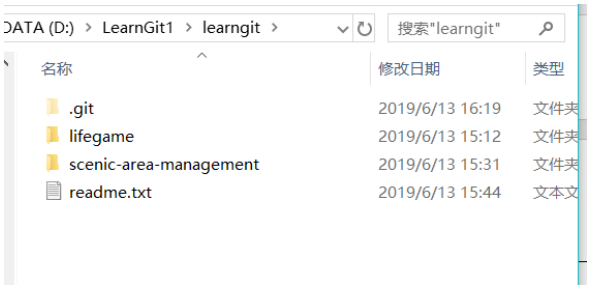


4.在本地把文件加入仓库

安装好后打开 git-bash.exe。首先在命令行进行如下输入名称和邮箱地址，对 git 进行配置。

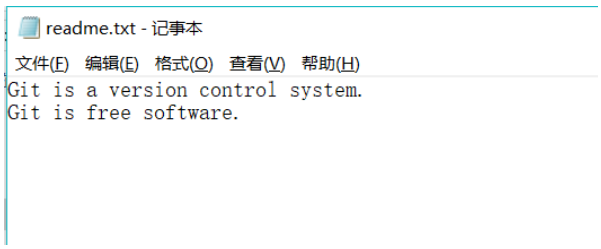


然后新建一个文件夹 learngit 使用 git init 命令把该目录变为一个 git 仓库

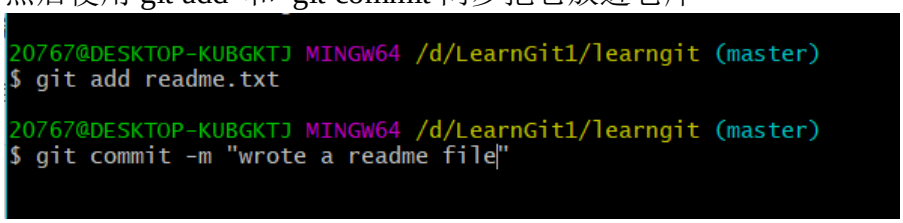


目录下的一些文件是后来添加的，可以看到目录下有一个.git 的目录，说明该目录已经变为一个 git 仓库了。这个.git 是 Git 用来跟踪管理版本库的，万不得已不要改目录的文件，不然可能会把 git 仓库破坏。

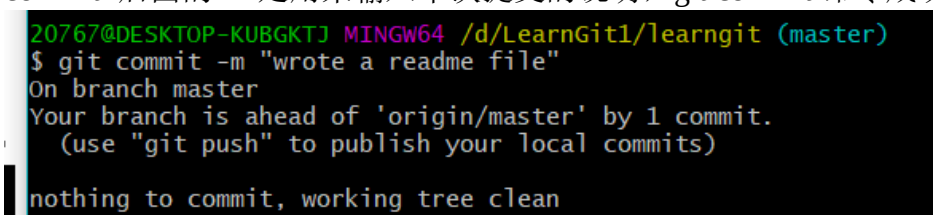
之后我们在这个目录下编写一个 readme.txt 文件如图：



然后使用 git add 和 git commit 两步把它放进仓库



commit 后面的-m 是用来输入本次提交的说明，git commit 命令成功后会进行如下提示。



5.将文件上传到远程仓库。

先在 GitHub 中创建一个新的 repository 叫 learnGit。然后根据 github 的提示，在命令行上

输入 git remote add origin https://github.com/brsmsg/learnGit.git 连接上后也会收到 github 发来的邮件：

[GitHub] A personal access token has been added to your account

发件人：GitHub <noreply@github.com>
(由 bounces+848413-c9bd-2076706661=qq.com@sgmail.github.com 代发)
时 间：2019年6月13日(星期四) 下午3:06
收件人：华尔街骆士奇 <2076706661@qq.com>

纯文本 |

Hey brsmsg!

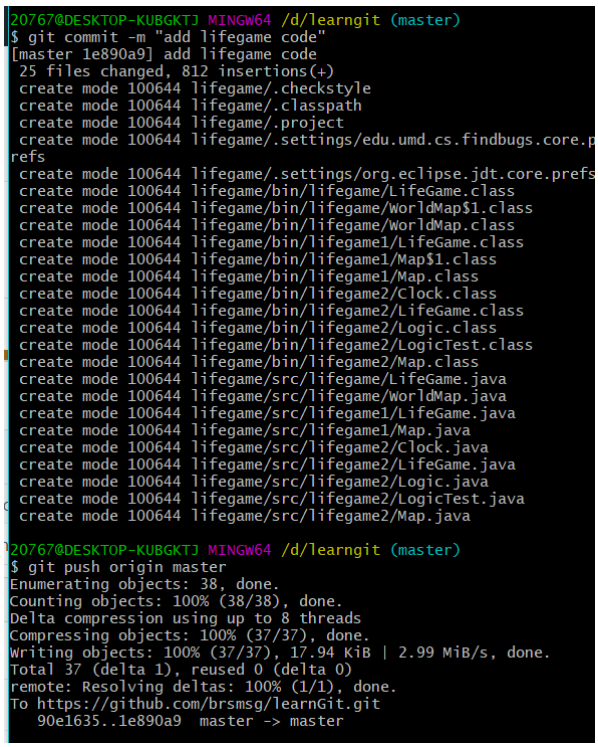
A personal access token (git: https://github.com/ on DESKTOP-KUBGKTJ at 13-6月-2019 15:06) with gist and repo scopes was recently added to your account. Visit https://github.com/settings/tokens for more information.

To see this and other security events for your account, visit https://github.com/settings/security

If you run into problems, please contact support by visiting https://github.com/contact

Thanks,
Your friends at GitHub

之后就可以把本地版本库的内容推送到 GitHub 上。命令行输入 git push origin master。我在把 readme 文件 push 上去。成功后，打开 github 的 learnGit repository 可以看到已经添加成功。我把 readme 文件 push 上去后又把之前实验写的声明游戏的代码也推送到了这个 repository。



```
20767@DESKTOP-KUBGKTJ MINGW64 /d/learngit (master)
$ git commit -m "add lifegame code"
[master 1e890a9] add lifegame code
 25 files changed, 812 insertions(+)
 create mode 100644 lifegame/.checkstyle
 create mode 100644 lifegame/.classpath
 create mode 100644 lifegame/.project
 create mode 100644 lifegame/.settings/edu.umd.cs.findbugs.core.p
refs
 create mode 100644 lifegame/.settings/org.eclipse.jdt.core.prefs
 create mode 100644 lifegame/bin/lifegame/LifeGame.class
 create mode 100644 lifegame/bin/lifegame/WorldMap$1.class
 create mode 100644 lifegame/bin/lifegame/WorldMap.class
 create mode 100644 lifegame/bin/lifegame1/LifeGame.class
 create mode 100644 lifegame/bin/lifegame1/Map$1.class
 create mode 100644 lifegame/bin/lifegame1/Map.class
 create mode 100644 lifegame/bin/lifegame2/Clock.class
 create mode 100644 lifegame/bin/lifegame2/LifeGame.class
 create mode 100644 lifegame/bin/lifegame2/Logic.class
 create mode 100644 lifegame/bin/lifegame2/LogicTest.class
 create mode 100644 lifegame/bin/lifegame2/Map.class
 create mode 100644 lifegame/src/lifegame/LifeGame.java
 create mode 100644 lifegame/src/lifegame/WorldMap.java
 create mode 100644 lifegame/src/lifegame1/LifeGame.java
 create mode 100644 lifegame/src/lifegame1/Map.java
 create mode 100644 lifegame/src/lifegame2/Clock.java
 create mode 100644 lifegame/src/lifegame2/LifeGame.java
 create mode 100644 lifegame/src/lifegame2/Logic.java
 create mode 100644 lifegame/src/lifegame2/LogicTest.java
 create mode 100644 lifegame/src/lifegame2/Map.java

20767@DESKTOP-KUBGKTJ MINGW64 /d/learngit (master)
$ git push origin master
Enumerating objects: 38, done.
Counting objects: 100% (38/38), done.
Delta compression using up to 8 threads
Compressing objects: 100% (37/37), done.
Writing objects: 100% (37/37), 17.94 KiB | 2.99 MiB/s, done.
Total 37 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/brsmsg/learnGit.git
   90e1635..1e890a9  master -> master
```

6.分支管理。创建一个 dev 分支，并切换至 dev 分支。使用 git branch 命令查看当前分支，看到所有分支 *表示当前分支。然后我们就可以在 dev 分支上进行提交，对 readme.txt 进行一次修改试一下。加上一行

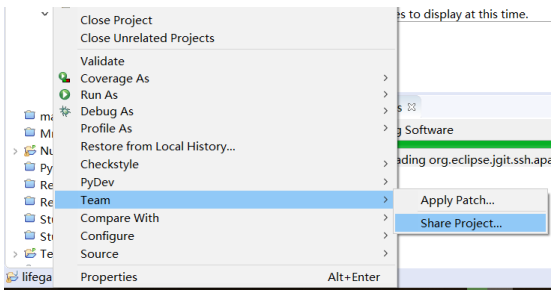
```
20767@DESKTOP-KUBGKTJ MINGW64 /d/learnGit1/learngit (master)
$ git checkout -b dev
Switched to a new branch 'dev'

20767@DESKTOP-KUBGKTJ MINGW64 /d/learnGit1/learngit (dev)
$ git brach
git: 'brach' is not a git command. See 'git --help'.

The most similar command is
        branch

20767@DESKTOP-KUBGKTJ MINGW64 /d/learnGit1/learngit (dev)
$ git branch
* dev
  master
```

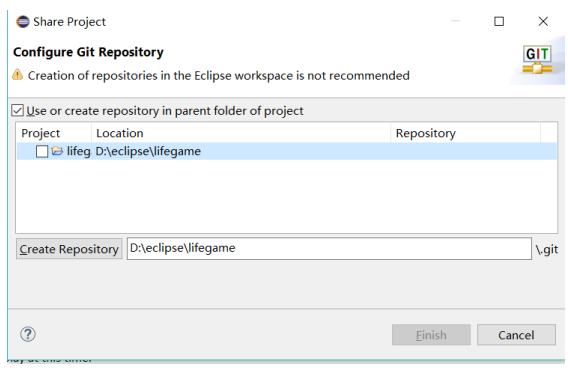
修改完后奇幻会 master 分支，并将 dev 分支的工作成功合并到 master 分支上，合并完成后删除 dev 分支，使用 git branch 可以看到只有 master 分支了。

```
20767@DESKTOP-KUBGKTJ MINGW64 /d/learnGit1/learngit (dev)
$ git add readme.txt

20767@DESKTOP-KUBGKTJ MINGW64 /d/learnGit1/learngit (dev)
$ git commit -m "branch test"
[dev 35fdd47] branch test
 1 file changed, 2 insertions(+), 1 deletion(-)

20767@DESKTOP-KUBGKTJ MINGW64 /d/learnGit1/learngit (dev)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

20767@DESKTOP-KUBGKTJ MINGW64 /d/learnGit1/learngit (master)
$ git merge dev
Updating 1e890a9..35fdd47
Fast-forward
 readme.txt | 3 ++-
 1 file changed, 2 insertions(+), 1 deletion(-)

20767@DESKTOP-KUBGKTJ MINGW64 /d/learnGit1/learngit (master)
$ git branch -d dev
Deleted branch dev (was 35fdd47).

20767@DESKTOP-KUBGKTJ MINGW64 /d/learnGit1/learngit (master)
$ git branch
* master
```

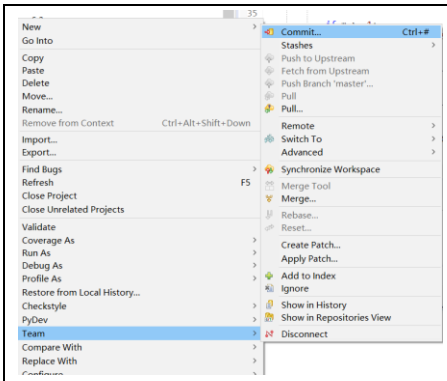
最后打开 readme.txt 可以看到修改成功。

7 使用 eclipse 进行 git 操作。与使用命令行类似，使用 eclipse 也要先把代码存放到本地版本库，然后再 push 到远端版本库上。具体操作如下：
右键我们要进行操作的 projrct 选择 team->share project，然后在如图界面对 git repository 进行配置

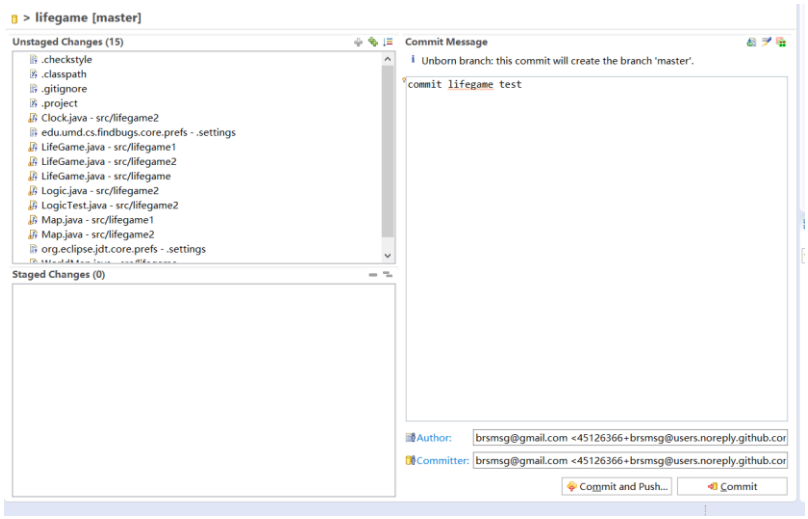




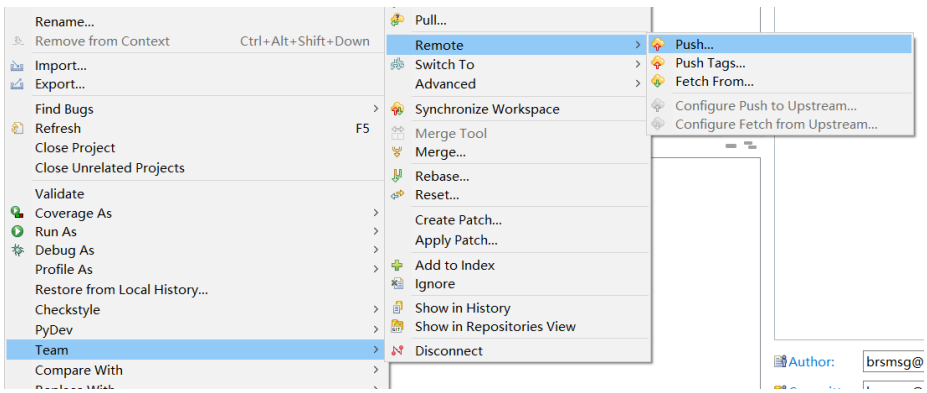
选择 team->commit



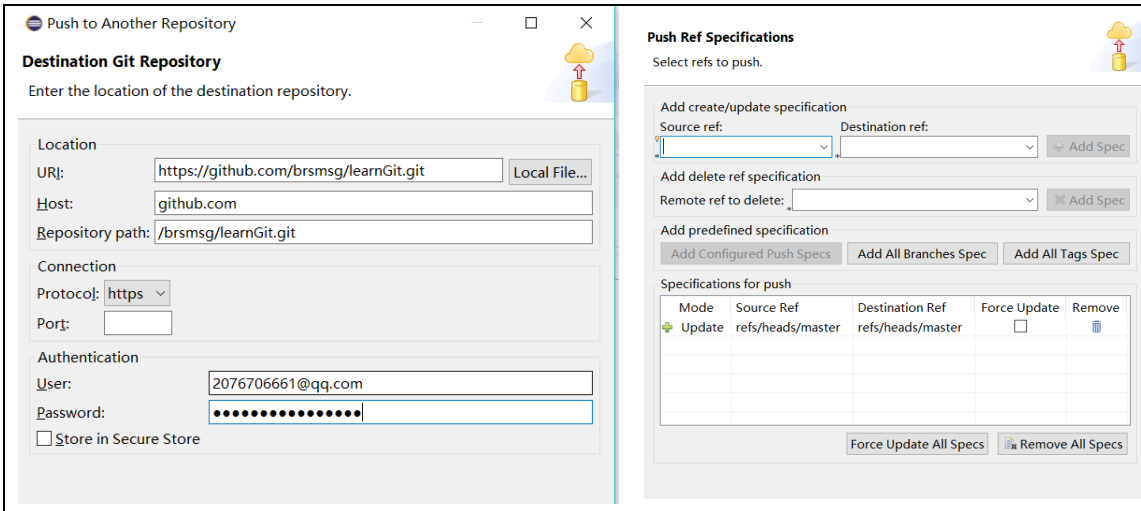
选择要 commit 的文件，然后在 commit message 中输入消息，点击 commit 就提交到本都版本库了

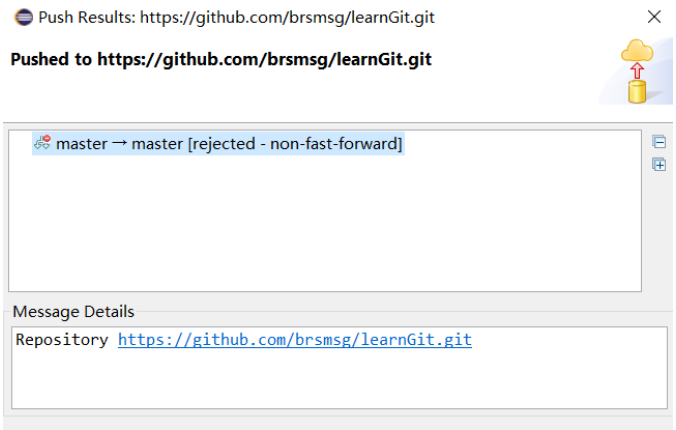


下一步就是推送到远端版本库，同样 team-> remote->push



然后根据 eclipse 的提示，输入远端版本库的各种信息，进行推送即可。推送成功后在 github 上也可以看到我们推送的文件





五、主要仪器设备

1）Eclipse

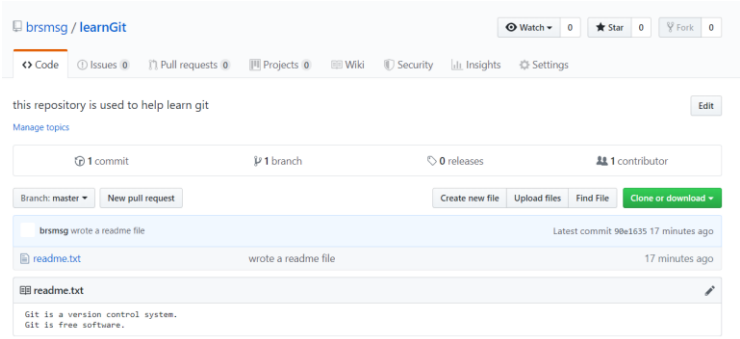
2）个人电脑

3）JProfiler

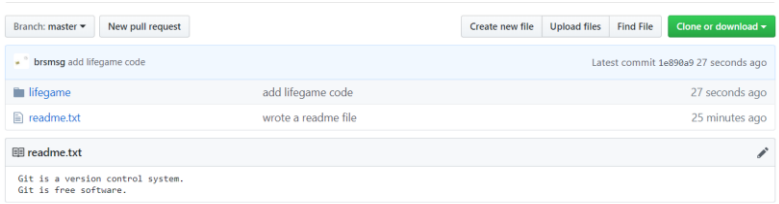
第二部分：实验过程记录（可加页）（包括实验原始数据记录，实验现象记录，实验过程发现的问题等）

一、实验数据和现象

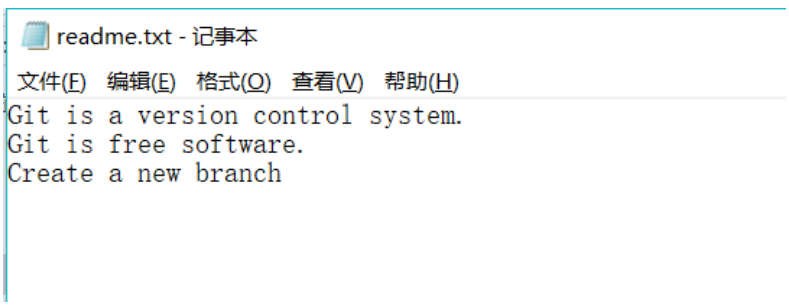
将 readme.txt 推送到远程库后



将 lifegame 也推送到远程库后



分支管理后，打开 readme.txt，可以看到 readme.txt 文件已经被修改成功。增加了新的一行 Create a new branch。说明们使用分支管理来修改文件成功。而且命令行上也显示了我们成功创建新分支 dev，删除它以及对文件进行修改等的一系列操作



之后使用 eclipse 中的 git 工具将本地版本库的代码文件推送到远程版本库 GitHub 中，成功后在 github 上也可以看到和上图一样的效果。

二、实验发现的问题：

1.有几次在推送到远端版本库的时候，忘记先 commit 到本地版本库，导致失败。

2.在命令行输入命令的时候，有时候会输入错，导致报错。

第三部分　结果与讨论（可加页）

一、实验结果分析（包括数据处理、实验现象分析、影响因素讨论、综合分析和结论等）

1）实验现象分析

通过我的 git 操作的结果，文件成功上传到了远端版本库，说明我使用命令进行的一系列 git 操作是正确的且成功了。在使用 git 的时候一定要注意，要先把代码提交到本地版本库，再推送到远端版本库。Git 提供的分支功能也是十分实用，让多人可以对代码进行修改与提交。通过实验我们发现 Git 的确是十分好用且强大的分布式版本控制系统。

2） 结论

我们很容易发现 git 有如下优点：

1 适合分布式开发，强调个体。

2 公共服务器压力和数据量都不会太大。

3 速度快、灵活。

4 任意两个开发者之间可以很容易的解决冲突。

5 离线工作。

缺点：

1 学习周期相对而言比较长。

2 不符合常规思维。

3 代码保密性差，一旦开发者把整个库克隆下来就可以完全公开所有代码和版本信息。

二、实验小结及体会

这次试验下载安装了 git 并进行了基本操作，我之前就注册了 github 账号，但是就只是把它当作一个看别人开源代码的平台，偶尔也放一点点自己的代码上去。也并没有很多的去了解 git 这么强大的工具。通过本次实验，我也初步类对 git 有了一定的了解，对版本库，分支，等有了初步的理解。同时，也学会了一些最基本的 git 命令。Git 确实是一种很强大的工具，不论是现在的学习，还是将来的工作当中，肯定都少不了接触 git。这次实验所学到的当然也仅仅只是皮毛，在之后我也会继续深入了解 git，去学习它，直到我能够熟练使用他为止。这次试验我也了解到了 git 的诞生过程，git 的诞生故事也充满传奇色彩，不得不佩服，Linus 不愧是 Linus，感谢他创造了如此强大且好用的分布式版本控制工具。

成绩评定表：

序号	评分项目	满分	实得分

1	**实验报告格式规范**	**2**	
2	**实验报告过程清晰，内容详实**	**4**	
3	**实验报告结果正确性**	**2**	
4	**实验分析与总结详尽**	**2**	
	总得分	**10**	