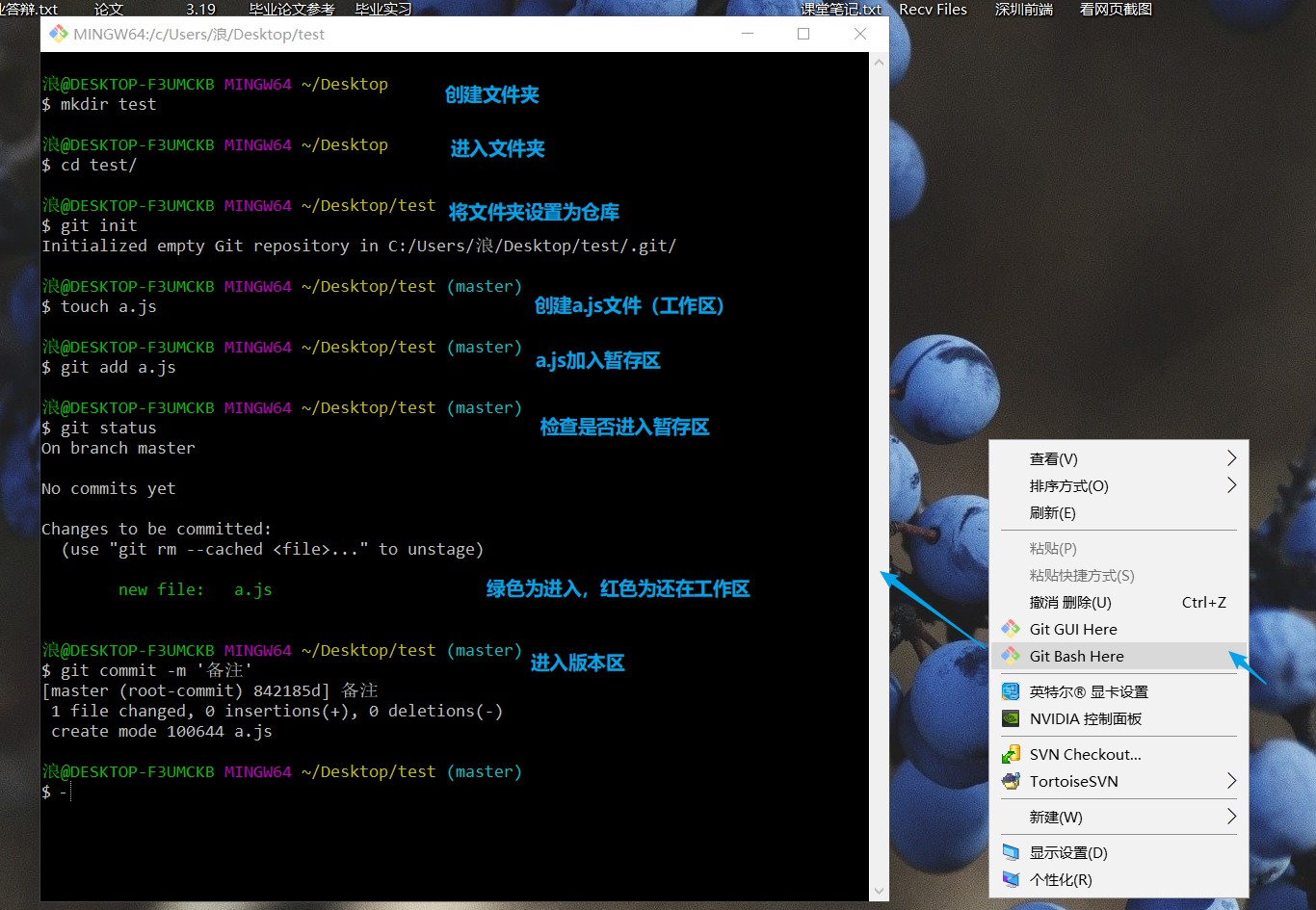
基本命令看python的linux命令行



1. 分支：

主干：svn（trunk） git（master） 存放上线用的代码版本

其他分支

branches / development 开发分支： 程序员开发代码的分支

tags 里程碑/重要的代码版本备份的分支

bugs 解决bug的分支

test 测试代码的分支

今后会创建n分支，不同分支会起不同的作用： 能让对代码进行不同的操作，从而做不同的任务

2. 分支操作

git checkout -b dev （1. 新建dev分支 2. 切换到dev分支 3. 将当前分支的最新版本复制一份到dev上）

git checkout dev 切换到dev分支

git branch 查看当前版本库中所有分支（当前所在分支会高亮显示）

git merge dev 在当前分支上合并其他分支代码（当前分支会变，dev分支不变）

3. 重要指令

git init 初始化一个空的版本库

git add . 将工作区文件添加到暂存区

git commit -m 'xxx' 将暂存区文件添加到版本区

git status 查看状态

git diff 对比工作区和暂存区文件的差异

git log 查看完成的版本历史记录

git reflog 查看简版版本历史记录

git reset --hard HEAD^ / git reset --hard xxx 回退版本

git checkout -b dev 1. 新建dev分支 2. 切换到dev分支 3. 将当前分支的最新版本复制一份到dev上

git checkout dev 切换到dev分支

git branch 查看当前版本库中所有分支（当前所在分支会高亮显示）

git merge dev 在当前分支上合并其他分支代码（当前分支会变，dev分支不变）

4. github操作

- 需要将本地仓库内容提交到远程仓库去保管

- 创建远程仓库

- 本地仓库的操作

- git init

- git add .

- git commit -m 'xxxx'

- git remote add origin xxxx 本地仓库和远程仓库关联起来

- git push -u origin master 将本地仓库master分支的版本区所有内容提交到远程仓库保管（第一次加上-u参数，第二次以后就不用了）

第二次：

- git add .

- git commit -m 'xxxx'

- git push origin master

- 本地没东西，需要远程仓库代码

- git clone xxxx 将远程仓库所有版本信息全部克隆下来（只克隆master分支）

- 需求：需要dev分支的内容

- git fetch origin dev1:dev2 将远程仓库的dev1分支的内容保存在本地的dev2分支上

- 远程仓库更新了，本地需要拉去远程仓库的最新的内容

- git pull origin dev 拉取远程仓库dev分支的内容