目的：

    1，大家共同维护一个清晰可读代码提交历史，便于代码审查、代码回滚、剔除事故代码

    2，尽可能减少事故发生

规范一：

    1，每一个新项目从主干（master）开一个新分支

    2，不允许多个项目共用一个分支，哪怕可以同时上线

    3，建议分支名：feature/项目名称，例如：feature/v79

    4，如果多人共同开发，每个人都要从项目分支（feature/v79）再开出个人的开发分支，建议名称 feature/项目名称/人名，例如：feature/v79/mh

规范二：每次提交代码都必须有含义明确的注释，比如：“修改了XXX的BUG”，“完成XXX功能”，禁止：“调试”，“测试”，“debug”

规范三：推荐频繁的提交代码，并把开发分支push到远程

规范四：切换分支之前一定要确保本地工作区是干净的，所有修改和添加的文件都提交后在切换分支

规范五：

    1，推荐频繁的将个人开发分支中的内容到项目分支中，频次可以自行掌握

    2，禁止将个人开发分支中未完成无法运行的代码合并到项目分支中

    3，个人开发分支不要和其他人的开发分支相互合并，如果需要用别人写的内容，先让他把自己的内容合并到项目分支中，然后在把项目分支的内容合过来

规范六：

    1，推荐频繁的将项目分支中的内容合并到个人开发分支中

    2，推荐频繁的将主干分支中的内容合并到项目分支中

规范七：推荐个人开发分支和项目分支合并的时候，用gitlab提交merge request，让其他同事做下人工代码审查，或者自己进行审查

规范八：合并代码时要加 --no-ff 选项，例如： git merge --no-ff

规范九：推送分支到远程时禁止使用 --force 选项，例如：git push --force

规范十：拉取代码时应使用 --ff-only 选项，例如 git pull --ff-only

规范十一：从远程仓库更新本地仓库时应该加 -p 选项，例如： git fetch -p

规范十二：提交代码时要加 -a 选项，把修改的文件自动加入缓冲区，列如：git commit -a -m '注释'

规范十二：谨慎使用 rebase，reset，revert等改变Git提交历史的操作

规范十三：提交点（CommitId）和标签（Tag）不是真正的分支，检出时要加 -b 选项生成新分支，例如：git checkout -b branch\_name tag\_name

规范十四：

1，每次代码上线，创建一个上线分支，把项目分支合并到上线分支中，然后把上线分支发布上线

2，建议上线分支名：release/日期-序号，例如release/20180509-1

3，上线成功后把上线分支合回主干

4，同一时间只能有一个上线分支

5，建议在把项目分支合并到上线分支的时候过一遍单元测试

规范十五：每次上线分支合并到主干，马上打一个轻量级的Tag，例如：git tag v20180509.1

规范十六：禁止直接在主干分支上进行修改、提交、推送等操作，如果需要紧急修改线上BUG应新开热修复分支，例如：hotfix/20180509-1

场景一：创建新功能分支

    #检查本地工作区是否干净，如果有修改，添加并提交新建和修改的文件

    git status

    git add filename

    git commit -a -m '注释'

    #切换到主干，并拉取最新的代码

    git checkout master

    git fetch -p

    git pull --ff-only

    #从主干拉出项目分支

    git checkout -b feature/v79

    #从项目分支拉出个人开发分支

    git checkout -b feature/v79/mh

场景二：日常提交代码及分支合并

    #切换到项目分支并拉取最新内容

    git checkout feature/v79

    git fetch -p

    git pull --ff-only

    #切回到开发分支合并项目分支内容

    git checkout feature/v79/mh

    git merge --no-ff feature/v79

    git push origin feature/v79/mh

    #添加新建文件到缓冲区

    git add filename

    #自动添加修改文件到缓冲区并提交

    git commit -a -m '注释'

    #推送分支代码到远程仓库

    git push origin feature/v79/mh

场景三：上线后打Tag

    #切换到主干并拉取最新代码

    git checkout master

    git fetch -p

    git pull --ff-only

    #打Tag

    git tag v20180509.1