بسمه تعالی

دانشکده مهندسی کامپیوتر

کارگاه کامپیوتر (۴۰۱۰۸) نیمسال اول ۱۴۰۳-۱۴۰۲

تمرین ششم
آشنایی با گیت

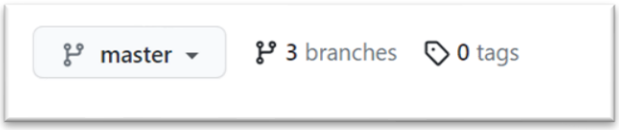
مهلت ارسال: ۱۴۰۲/۹/۱۰

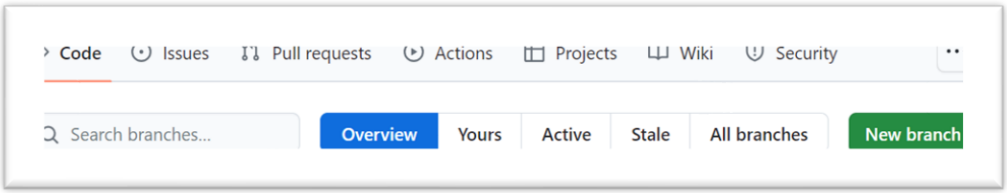
برای انجام این تمرین نیاز است که ابزار گیت را روی سیستم عامل خود نصب کنید. نحوه نصب بر روی ویندوز و لینوکس در درسنامه توضیح داده شده است. در گام بعدی اگر حساب شخصی در سایت github ندارید یک حساب شخصی ایجاد کنید.

توجه: لطفا پس از انجام سوال‌ها فولدر شامل مخزن‌تان را به همراه فایل پاسخ‌ها در یک پوشه قرار دهید و فایل فشرده آن را بر روی سایت آپلود کنید. در هر مرحله از انجام موارد ذکر شده در سوال‌ها جهت نشان دادن گام‌های انجام شده و تغییرات را اسکرین شات بگیرید و در فایل پاسخ قرار دهید.

۱- یک repository برای پروژه خود ایجاد کرده و آن را روی گیت‌هاب بارگذاری کنید. لینک آدرس مخزن ایجاد شده را در فایل پاسخ تمرین قرار دهید. لطفا تمامی مراحل بعدی را پس از انجام دادن در مخزن ریموت نیز push کنید.

۲- یک branch جدید به نام develop در ریموت ریپازیتوری خود ایجاد کنید. ابتدا روی branches رفته و سپس گزینه New branch.





فایلی به نام calculator.c با محتوای زیر ایجاد کرده و آن را در شاخه develop اضافه نمایید.

```c
#include <stdio.h>

int calculate(int a, int b) {
    // Some calculation
    return a * b;
}
```

۳- در مخزن local خود ابتدا pull گرفته تا شاخه develop را از ریموت دریافت نمایید. سپس روی شاخه develop بروید و بر روی آن یک شاخه جدید به اسم feature-conflict ایجاد نمایید. به شاخه feature-conflict بروید و سپس فایل calculator.c را با محتویات زیر تغییر دهید و کامیت کنید. شاخه جدید و محتویات آن را در ریموت خود قرار دهید.

```
#include <stdio.h>

int calculate(int a, int b) {
    // A different calculation
    return a + b;
}
```

حال مجدد به شاخه develop باز گردید (توجه کنید که هنوز در local خود هستید) و تغییرات زیر را بر روی فایل calculator.c اعمال کنید و سپس تغییرات‌تان را کامیت نمایید.

```
#include <stdio.h>

int calculate(int a, int b) {
    // A different calculation
    return a * b * 2 -10 ;
}
```

اکنون بر شاخه develop بروید و شاخه feature-conflict را با شاخه develop ادغام کنید. همان طور که مشاهده می کنید با خطای conflict مواجه خواهد شد. توضیح مختصری بفرمایید که چرا با این موضوع مواجه شده‌اید و نحوه رفع آن را توضیح دهید. سپس conflict به وجود آمده را به صورتی رفع نمایید که خط آخر تابع به شکل زیر باشد. (مراحل تغییرات را اسکرین شات بگیرید و در فایل پاسخ قرار دهید.)

بعد از رفع conflict، ادغام شاخه feature-conflict را با شاخه develop را انجام دهید. (تمامی تغییرات را در ریموت نیز push کنید).

```
#include <stdio.h>

int calculate(int a, int b) {
    // A combination of both calculations
    return a * b * 2 -10 + (a + b);
}
```

۴- فرض کنید یک توسعه دهنده نتواند تشخیص دهد که در حال حاضر در کدام شاخه است و به اشتباه، به جای کامیت کردن در شاخه مورد نظرش، در شاخه دیگری کامیت را انجام دهد. حالا برای رفع این مساله باید ابتدا git show را اجرا کند و تغییرات commit را ذخیره کند. بعد شاخه مورد نظرش را بررسی کند، تغییرات مورد نیاز را مجدد در آنجا اعمال کند و با همان پیام commit کند. اما همه اینها را می توان به طور خودکار با استفاده از یک دستور یعنی cherry-pick انجام داد. cherry-picking به انتخاب و اعمال تغییرات خاص از یک شاخه به شاخه دیگر اشاره دارد و این امکان را به شما می‌دهد تا به جای ادغام کامل یک شاخه فقط تغییرات در یک یا چند کامیت را انتخاب کرده و آن‌ها را به شاخه دیگری اعمال کنید.

روی شاخه develop یک شاخه جدید به اسم cherry-pick-test در مخزن local ایجاد کنید. ۴ مرحله زیر را به ترتیب توسط ۴ کامیت انجام دهید و تمامی این تغییرات و شاخه جدید را در ریموت نیز قرار دهید:

- فایل جدیدی به اسم greeting.c با هر محتوای دلخواه ایجاد کنید و کامیت نمایید.
- محتوای greeting.c را تغییر دهید و مجدد با پیام مناسب کامیت کنید.
- فایل greeting.c را حذف کنید و کامیت نمایید.
- متد زیر را به فایل calculator.c اضافه نمایید و سپس تغییرات‌تان را با پیام مناسب کامیت نمایید

```
int pow2(int a) {

    return a * a;
}
```

الف) با دستور git log لیست کامیت‌های این برنچ را در فایل گزارش خود قرار دهید.

ب) سپس hash کامیت آخر که تغییر فایل calculator.c بود را پیدا کنید و سپس به برنچ develop بروید و این کامیت را با دستور cherry-pick به برنچ develop اضافه کنید. و سپس با دستور log لیست کامیت های برنچ develop را نمایش دهید تا کامیت جدید مشخص شود. (اسکرین شات تمام دستورات این مرحله را در فایل گزارش قرار دهید.)

ج) توضیح دهید که چرا بهتر است که به ندرت از cherry-pick استفاده کنیم و استفاده از آن چه مساله‌ای را می تواند ایجاد کند؟ (امتیازی)

۵- در مورد فایل gitignore. توضیح دهید و هم چنین بفرمایید به چه علت استفاده از این فایل در پروژه‌ها ضروری است؟ حال برای پروژه خود یک فایل gitignore. با محتوای زیر ایجاد کنید. و سپس تغییرات را در ریموت نیز قرار دهید.

```
__pycache__
*.py[cod]
*$py.class

# Distribution / packaging
.Python build/
dist/
lib/
lib64/
sdist/
var/
*.manifest
*.spec

# Log files
pip-log.txt
pip-delete-this-directory.txt
*.log

# Environments
.env
.venv
env/
venv/
ENV/

### VisualStudioCode ###
.vscode/*
!.vscode/settings.json
!.vscode/tasks.json
```

```
!.vscode/launch.json
!.vscode/extensions.json
*.code-workspace
.history/
```

نکات مهم در مورد تحویل پاسخ تمرین

- خروجی تمرین شما می‌بایست یک فایل zip شده شامل فولدر پروژه و فایل گزارش با فرمت PDF با نام زیر باشد و در صفحه‌ی تمرین در کوئرا ارسال شود.

CW-HW#<STDID>.pdf.....(<STDID> شماره دانشجویی شماست)

- در صورت بروز سؤال و ابهام، می‌توانید از طریق صفحه‌ی کوئرای اختصاصی گروه خود موضوع را مطرح نمایید.
- تأخیر در ارسال تمرین پذیرفته نمی‌باشد.
- پاسخ تمرین باید به صورت تایپ شده و مرتب (با مرزبندی مشخص برای هر سؤال) باشد.
- در صورت ارسال چندین نسخه از پاسخ تمرین در زمان‌های مختلف، فقط نسخه‌ی آخر بررسی می‌شود.
- پاسخ هر سؤال باید تا حد امکان دقیق و متناسب با سؤال باشد. از ذکر مطالب مبهم، نامرتبط و زائد خودداری نمایید.
- شما متعهد هستید که دانش و دانسته‌های خود را در قالب جمله‌بندی‌های خودتان به عنوان پاسخ تمرین تحویل دهید. لذا به هیچ‌وجه اقدام به کپی کردن مطالب از منابع مختلف نکنید.
- در صورت استفاده از هر گونه منبع برای پاسخ به سؤالات (البته در سؤالاتی که مجاز به این کار باشید)، ذکر نام و نشانی دقیق دسترسی به صفحه مورد نظر الزامی است. در صورت تشخیص رونویسی از منابع و وب‌سایتها بدون ارجاع، به عنوان تقلب محسوب می‌شود.
- در صورت کشف تقلب، نمره کل تمرین برابر با «منفی ۱۰۰» لحاظ می‌گردد و در صورت تکرار، برابر مقررات آموزشی اقدام خواهد شد .

موفق باشید.