دانشکده‌ی مهندسی کامپیوتر
برنامه‌سازی پیشرفته
راهنمای بازبینی کد (Code Review)

سارا کدیری، علی حیدری
استاد: سید صالح اعتمادی

زمستان- بهار ۹۸-۱۳۹۷

فهرست مطالب

۱ مقدمه ۲
۲ بازبینی کد چیست؟ ۲
۳ فرهنگ بازبینی کد ۲
۱.۳ نکات مورد توجه ۲
۱.۱.۳ جدا نگه‌داشتن جنبه‌های شخصی از جنبه‌های فنی ۲
۲.۱.۳ فرهنگ انتقادپذیری ۲
۳.۱.۳ فرهنگ انتقاد سازنده ۲
۴.۱.۳ فرهنگ شکرگزاری ۲
۵.۱.۳ صراحت در بازخورد ۳
۶.۱.۳ صراحت در قبول یا رد بازخورد ۳
۴ روند بازبینی کد برای درس برنامه‌سازی پیشرفته ۳
۱.۴ مرحله‌ی اول: تحویل تمرین ۳
۲.۴ مرحله دوم: بازبینی کد ۳
۳.۴ الگوهای ارزیابی ۴
۵ پیوست‌ها ۴
۱.۵ نحوه‌ی اضافه کردن بازبیننده به پروژه ۴
۲.۵ جدول بازبینندگان ۴

۱ مقدمه

بازبینی کد [۱] از مهم‌ترین و موثرترین راه‌های یادگیری و افزایش کیفیت کد است که در تمام شرکت‌های نرم‌افزاری حرفه‌ای از قبیل مایکروسافت، گوگل، آمازون استفاده می‌شود. لازم است بازبینی کد دارای بار آموزشی فنی و فرهنگی برای بازبیننده کد [۲] و بازبین‌شونده کد [۳] باشد. لذا برای داشتن یک بازبینی کد موثر، به بلوغ فنی و فرهنگی در هر دو طرف لازم است.

۲ بازبینی کد چیست؟

روند بازبینی کد به این شکل است که بعد از این‌که برنامه‌نویس کد خود را به بهترین وجهی که می‌تواند کامل کرد، کد خود را (یا تغییرات کد خود را) با ابزار بازبینی کد (در درس ما `Pull Request`) برای بازبیننده(ها) می‌فرستد. بازبیننده کد را از جنبه‌های مختلف مثل خوانایی، کارایی، خطاپذیری و آزمون بررسی کرده و بازخوردهای مناسب را یادداشت می‌کند. بازبین‌شونده بازخوردها را با دقت بررسی کرده و برای هر بازخورد یا پیشنهاد بابیننده را قبول کرده و تغییرات را اعمال می‌کند و یا توضیح مناسبی در جواب بازبیننده می‌دهد. این روند تا رضایت کامل بازبیننده کد ادامه پیدا می‌کند.

۳ فرهنگ بازبینی کد

بازبینی کد در بسیاری از شرکت‌ها منشا کدورت و از هم پاشیدگی تیم‌های نرم‌افزاری می‌شود. جلوگیری از این مساله نیاز به فرهنگ‌سازی در جامعه نرم‌افزاری دارد. بهترین زمان و مکان برای شروع این فرهنگ‌سازی دوران تحصیل دانشجویان در دانشگاه است. از این جهت که:

۱. این محیط عاری از رقابت‌های درون شرکتی برای ترفیع و برتری است

۲. دانشجو برای یادگیری در دانشگاه است و نسبت به بازخوردها کمتر جنبه دفاع خواهد داشت.

۱.۳ نکات مورد توجه

رعایت نکات زیر برای برگزاری یک بازبینی کد موثر لازم است:

۱.۱.۳ جدا نگه‌داشتن جنبه‌های شخصی از جنبه‌های فنی

بازخوردی که برای کد شما گذاشته می‌شود برای «کد» شماست، نه «شما». برداشت شخصی نکنید. هم‌چنین برای بازبیننده: بازخورد شما باید متوجه «کد» باشد نه نویسنده کد.

۲.۱.۳ فرهنگ انتقادپذیری

همه ما می‌توانیم برنامه‌نویس‌های بهتری بشویم. بازبینی کد شما بهترین راه پیدا کردن فرصت‌های بهتر شدن است. از این جهت که شما سعی خود را برای ارائه بهترین کار خود در محدودیت زمانی که داشته‌اید کرده‌اید و بازبینی کد شما فرصت‌های بهتر شدن را به شما نشان می‌دهد.

۳.۱.۳ فرهنگ انتقاد سازنده

بازبینندگان کد هم لازم است علاقه به پیشرفت و بهتر شدن کد و برنامه‌نویس را در بازخوردها و لحن آن رعایت کنند. علاوه بر این سطح برنامه‌نویس را در بازخوردهای خود در نظر بگیرید. مثلا تذکرهای بسیار ظریف را به برنامه‌نویس مبتدی ندهید. و سلسله مراتب بازخوردها را رعایت کنید. یعنی اول بازخوردهای اساسی‌تر و مهم‌تر بعد بازخوردهای جزئی‌تر و ظریف‌تر.

۴.۱.۳ فرهنگ شکرگزاری

ازبیننده کد از وقت و کار خود گذشته که کد شما را بررسی کند و راه‌های بهبود کد شما را به شما نشان بدهد. این بهترین راه یادگیری برای شماست. شکرگزار این فرصت باشید و در پیام‌های خود لحن مناسب بکار ببرید.

[1] Code Reviewing
[2] Code Reviewer
[3] Code Reviewee

۵.۱.۳ صراحت در بازخورد

برای جلوگیری از پیام‌های بدون هدف و اتمام به موقع بازبینی کد، در بازخورد خود صراحت داشته باشید به طوری‌که بازبین‌شونده بتواند آن را در یکی از طبقه‌بندی‌های زیر جای دهد و تکلیف خود را بداند.

آ) کد به صورت فعلی قابل قبول نیست و باید تصحیح شود.

ب) کد به صورت فعلی قابل قبول است ولی اگر تصحیح شود بهتر است.

ج) کد به صورت فعلی قابل قبول است ولی سلیقه من این است که تغییر کند.

د) چه کد جالبی. این رو قبلا نمی‌دونستم...

۶.۱.۳ صراحت در قبول یا رد بازخورد

چنانچه با بازخورد موافق یا مخالف هستید، لازم است نظر موافق یا مخالف شما به صراحت در جواب شما با ذکر دلیل بیان شود.

۴ روند بازبینی کد برای درس برنامه‌سازی پیشرفته

با توجه به اهمیت بازبینی کد در آموزش در این درس از ابتدای ترم بازبینی کد بخشی از روند یادگیری برای شما می‌باشد. برای آشنایی با مراحل اضافه کردن بازبیننده کد به پروژه AP97982 در سایت Azure DevOps به مراحل ذکر شده در ضمیمه ۱.۵ مراجعه کنید.

۱.۴ مرحله‌ی اول: تحویل تمرین

- تحویل اولیه تمرین با کامل شدن `Pull Request` به `master` انجام می‌شود. دقت کنید که هنگام کامل کردن `Pull Request` گزینه `"Delete Source Branch"` انتخاب نشده باشد.
- برای اطمینان از کامل شدن درست `Pull Request` در کامپیوتر خود دستورات زیر را اجرا کنید و از وجود پروژه خود در شاخه `master` اطمینان حاصل کنید.

```
git checkout master
git pull
```

- چنانچه به اشتباه گزینه `Delete Source Branch` را انتخاب کرده باشید، لازم است با دستور زیر تمام شاخه‌های محلی را بار دیگر را روی سرور `push` کنید.

```
git push --all origin
```

۲.۴ مرحله دوم: بازبینی کد

- مهلت بازبینی کد یک هفته بعد از تحویل تمرین می‌باشد.
- اولین مرحله مطلع‌کردن بازبیننده کد از اتمام تمرین و تقاضای بازبینی کد می‌باشد.
- سپس بازبیننده کد بعد یک روز نتیجه بازبینی کد را در قالب کامنت‌ها لازم و نمره اولیه تمرین منعکس می‌کند.
  - Grade1://<AssignmentNumber>/<FirstGrade>
  - e.g., Grade1://A1/90
- بعد از گرفتن کامنت‌ها، دانشجویان یک روز فرصت دارند تغییرات لازم در کد را برای جلب نظر بازبیننده کد در همان شاخه تمرین اعمال کرده و `Pull Request` جدید با نام `Pull Request` اصلی/قبلی به‌علاوه‌ی پسوند `(_review)` برای فرستادن به `master` ایجاد کنند. مثال:

Pull Request name before review	Pull Request name after review
HW3	HW3_review
HW4	HW4_review
HW5	HW5_review

- این `Pull Request` تا پایان مهلت بازبینی کد بازمانده و روند بازبینی کد با مهلت ۲۴ ساعت برای پاسخ از هر دو طرف ادامه پیدا می‌کند. در نهایت نمره بازبینی شده به صورت زیر در `Pull Request` جدید منعکس می‌شود. نمره نهایی شما بر اساس نمره `Pull Request` اولیه و `Pull Request` جدید محاسبه خواهد شد.
  - Grade2://<AssignmentNum>/<FinalGrade>/<TotalComments>/<AddressedComments>
  - e.g., Grade2://A1/90/10/5
  - عدد `AddressedComments` نشان‌گر تعداد کامنت‌هایی است که دانشجو مطابق نظر و رضایت بازبیننده کد تغییرات لازم را انجام یا جواب مناسب داده باشد.
- در نهایت `Pull Request` دوم شما باید قبل از اتمام مهلت بازبینی کد کامل شده و در شاخه‌ی `master` ادغام (`merge`) شود.
- نمره نهایی شما بر اساس میزان مشارکت فعال شما در روند بازبینی‌کد خواهد بود.

۳.۴ الگوهای ارزیابی

نمره تمرین‌ها بر اساس جدول زیر محاسبه می‌شود. این جدول به تناسب تمرین در مستند تمرین بروز خواهد شد.

Subject	Program logic	Unit tests	Meaningful variable and method names and comments where necessary	Observe coding conventions in the names of variables and methods	no redundancy	small methods
Grade	30	30	10	10	10	10

۵ پیوست‌ها

۱.۵ نحوه‌ی اضافه کردن بازبیننده به پروژه

ابتدا ایمیل بازبیننده‌ی خود را از جدول بخش ۲.۵ پیدا کنید سپس برای اضافه کردن بازبیننده کد به آدرس:
`http://<StudentNumber>.visualstudio.com/_settings/users`
مراجعه کرده و طبق شکل ۱.۵ بازبیننده کد را اضافه کنید.

۲.۵ جدول بازبینندگان

برای اطلاع از ایمیل بازبیننده‌ی خود به جدول زیر مراجعه کنید.

RandomId	Rev. e-mail
1001	mohammad.m.a.77@outlook.com
1004	mahnaz.hagh@outlook.com
1073	sadra_h_m@outlook.com
1099	bluxixi@outlook.com
1111	nejati.mohammadsajjad@gmail.com
1234	sauleh@gmail.com
1357	41ir92a@gmail.com
1376	41ir92a@gmail.com

RandomId	Rev. e-mail
1379	gh_mhdi@outlook.com
2022	sadra_h_m@outlook.com
2265	nargeshm99@outlook.com
2320	miladisjobs@gmail.com
2710	gh_mhdi@outlook.com
2853	nili_505077@yahoo.com
2999	ehsandaramir@outlook.com
3333	nejati.mohammadsajjad@gmail.com
3911	sauleh@gmail.com
4251	dani77ghaemi@gmail.com
4279	mohammad.m.a.77@outlook.com
5322	mah_d10@outlook.com
5718	41ir92a@gmail.com
5831	hosseinSadatipour@gmail.com
5871	nili_505077@yahoo.com
6556	pooya_kabiri@yahoo.com
6654	nargeshm99@outlook.com
6674	bluxixi@outlook.com
6859	hosseinSadatipour@gmail.com
7223	mohammad.m.a.77@outlook.com
7225	nejati.mohammadsajjad@gmail.com
7678	bluxixi@outlook.com
7982	pooya_kabiri@yahoo.com
8301	gh_mhdi@outlook.com
8456	mahnaz.hagh@outlook.com
8534	hosseinSadatipour@gmail.com
8569	sauleh@gmail.com
8662	miladisjobs@gmail.com
8810	mah_d10@outlook.com
9204	mahnaz.hagh@outlook.com
9622	sadra_h_m@outlook.com
0002	nili_505077@yahoo.com
0021	dani77ghaemi@gmail.com
0098	miladisjobs@gmail.com

شکل ۱: نحوه‌ی اضافه کردن بازبیننده‌ی کد