دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر

تمرین اول

آزمون نرم‌افزار

نگارش

محمدرضا اخگری زیری

محمدعلی کشاورز

علی نظری

استاد درس

دکتر معصومه طارمی‌راد

زمستان ۹۹

فهرست مطالب

صفحه

سوال اول

علی‌بابا (*کشاورز*): علاوه بر اینکه توی هر تیم یک تستر وجود دارد، یک چپتر برای تست وجود دارد که جلسات مشترک دارند و درآن به انتقال دانش و تجربه می‌پردازند و هدف‌گذاری کلی می‌کنند، از نظر بلوغ تست، شرایط در سطح دو است؛ به این شکل که همیشه بین تیم توسعه و تیم تست حالت خصمانه وجود دارد. برای رفع این موضوع تلاش می‌شود که تست‌ها از ابتدا مشخص شوند که این در عمل هنوز به صورت کامل انجام نشده است.

کوییز آو کینگز (*اخگری*): به نظر من تیم ما در راستای رسیدن به مرحله سوم قرار دارد، برای این منظور ابتدا تیم تست با تیم ما همراه شد و اینگونه فاصله‌ی دو تیم کاسته شد تا مشکلات مرحله دوم تا حدی از بین برود. بعد از آن، تصمیمی که هنوز اجرایی نشده است، تمرکز روی تست‌کیس‌هاست به صورتی که ریسک‌ها کاهش پیدا کند و همکاری تیم توسعه و تست در آن مشهود باشد. اما همچنان در سطح دو هستیم و به سطح سه نرسیدیم.

بله (*نظری*): در تیم ما کسی به تنهایی وظیفه تست را بر عهده ندارد، تیم تست جداگانه یا تضمین کیفیت هم نداریم. در شرکت ما به این صورت عمل می‌شود که خود توسعه‌دهندگان به کمک هم‌دیگر تست‌ها را طراحی میکنند و با هم دیگه اجرا میکنند، هدف هم کم کردن ریسک‌های محصول هست، پس به نوعی میتوان گفت که در سطح سه هستیم.

سوال دوم

به طور کلی نزدیک‌شدن تیم تست و توسعه، در وهله‌ی اول؛ همچنین ترکیب فرآیند تست و توسعه به رسیدن از سطح ۲ به ۴ کمک می‌کند؛ یعنی اگر تلاش کنیم که فرهنگی در تیم ایجاد کنیم و تست را جزو فاکتورهای اساسی کیفیت نرم‌افزارمان ببینیم و به عنوان مثال فرآیند تولید محصول را به صورت Test driven جلو ببریم باعث می‌شود که ابتدا تیم تست و توسعه و محصول با کمک یک‌دیگر تست‌ها را طراحی کنند (البته فشار اصلی کار روی تیم تست است و تیم‌های دیگر همکاری خواهند کرد) و در ادامه تیم توسعه با یک هدف مشخص به سمت پیاده‌سازی بروند و تست‌ها را با موفقیت پشت سر بگذارند. این باعث می‌شود که هم نزاع بین تیم تست و توسعه به وجود نیاید، ریسک‌هایی که در ابتدا به ذهن رسیده نیز مورد توجه قرار گیرند و در نهایت هم کیفیت کلی نرم‌افزار با این رویکردی که تیم داشته به طور کلی افزایش پیدا می‌کند.

سوال سوم

الف)

تابعی که نوشتیم به صورت زیر است:

```
public static Vector union(Vector a, Vector b) {

    Vector c = new Vector(new HashSet(a));

    for(Enumeration e = b.elements(); e.hasMoreElements();) {
        Object o = e.nextElement();
        if (!c.contains(o))
            c.addElement(o);
    }
    return c;
}
```

ب)

ابهاماتی که دیده می‌شد از این قرارند:

- در صورت تعریف الف، برای پیاده سازی این تابع، گفته شده که اشیایی را که در یکی از این دو Vector هست را برگردانیم که این بیشتر شبیه تعریف تفاضل متقارن[1] است و باعث ابهام شده است در صورتی که احتمالا با توجه به اسم تابع، منظور اجتماع بوده است.
- تایپ اشیا ورودی و خروجی مشخص نشده‌اند که ممکن است موجب خطا شود و حتی ممکن است تعریف درستی از تساوی اشیا نتوانیم برای اجتماع در نظر بگیریم.
- در امضای این تابع، مشخص نیست که رفتار آن در قبال ورودی‌هایی که هنوز ساخته نشده اند و null هستند چگونه است.
- خروجی این تابع در صورت خالی بودن هر دو ورودی، مشخص نیست که باید یک Vector خالی برگردانده شود یا null.
- اگر در خود یک ورودی عناصر تکراری وجود داشته باشند، چه اتفاقی برای خروجی خواهد افتاد؟
- اگر تایپ ورودی‌های ورودی برابر نبود باید به چه صورتی این اجتماع را انجام دهیم؟

---

[1] Δ: برای تعریف تابع تفاضل به سایت مراجعه کنید.

ج)

آزمون‌هایی که برای اشکالات احتمالی طراحی کردیم به این صورت‌اند:

- در طراحی این تست‌ها، هر وکتور را برای سادگی به صورت یک آرایه در نظر گرفته‌ایم.

تست اول: در این تست هدف این بود که مطمئن شویم این تابع، اجتماع دو مجموعه را می‌گیرد (اگر هدف ما تابع تفاضل بود باید خروجی فرق می‌کرد).

$$a = [1,2,3], b = [1,4,5] \xrightarrow{\text{Expected}} [1,2,3,4,5]$$

```
a: [1, 2, 3]
b: [1, 4, 5]
result: [1, 2, 3, 4, 5]
```

تست دوم: در این تست هدف این است که نشان دهیم اجتماع‌گیری ما تایپ‌های مختلف را پشتیبانی می‌کند.

$$a = ["A","B"], b = [1,2] \xrightarrow{\text{Expected}} ["A","B",1,2]$$

```
a: [A, B]
b: [1, 2]
result: [A, B, 1, 2]
```

تست سوم: در این تست هدف ما این بود که نشان دهیم این متد در صورت مواجه شدن با وکتور null خطا می‌دهد. (هر کدوم از متغیرها نال بود باید این خروجی را بدهد)

$$a = null\ , b = [1,2] \xrightarrow{\text{Expected}} \text{NullPointerException}$$

```
a: null
b: [1, 2]
result: Exception in thread "main" java.lang.NullPointerException
```

تست چهارم: در این تست هدف این بود که رفتار تابع در قبال وکتور خالی را نشان دهیم.

$$a = [\,]\ , b = [1] \xrightarrow{\text{Expected}} [1]$$

```
a: []
b: [1]
result: [1]
```

تست پنجم: در این تست هدف نشان دادن رفتار تابع در صورت گرفتن دو Vectorخالی بود.

$$a = [\,], b = [\,] \xrightarrow{\text{Expected}} [\,]$$

```
a: []
b: []
result: []
```

تست ششم: در این تست می‌خواستیم نشان دهیم که اگر در یکی از مجموعه‌ها عنصر تکراری وجود داشته باشد، در خروجی این عناصر تکراری حذف خواهند شد.

$$a = [1,1,2], b = [1,3,4,4] \xrightarrow{\text{Expected}} [1,2,3,4]$$

```
a: [1, 1, 2]
b: [1, 3, 4, 4]
result: [1, 2, 3, 4]
```

د)

سعی شده است که تا جای امکان ابهامات را با امضای زیر حل کنیم، برای مثال با اضافه کردن *@NotNull* نشان داده شده که نباید ورودی *null* باشد و با عبارت *throws* نشان می‌دهیم که اکسپشن خواهیم داد. و همچنین با قرار دادن تایپ *Object* تعیین می‌کنیم که ورودی برای تمامی تایپ‌های جاوا ممکن است و برای تساوی از *equals* استفاده خواهد شد.

```
public static Vector<Object> union(@NotNull Vector<Object> a,@NotNull Vector<Object> b)
        throws NullPointerException
```

گزارش انجام تمرین به صورت گروهی:

در سوال اول از هر کس از تجربیاتش در مورد تست در شرکت‌هایی که کار کرده بود، صحبت کردیم و هر کدام این موارد را نوشتیم.

در سوال دوم هم ابتدا مقداری با هم صحبت کردیم و روش‌هایی که فکر می‌کردیم می‌تواند در را ستای بهبود فرایند تست موثر باشد را با همدیگر نوشتیم.

سوال سوم را هم با همکاری یکدیگر ابهامات و اشکالات را در آوردیم و برای آن‌ها تست‌هایی طراحی کردیم.

در کل می‌توان گفت که کل تمرین را با همکاری یکدیگر و به صورت همزمان (در جلسه آنلاین) انجام دادیم و تقسیم کاری بر اساس هر سوال انجام ندادیم و هر سوال نتیجه بحث و جدل همه‌ی ما بود.