SENG 437 - Software Testing

Lab. Report #4 – Mutation Testing and Analysis

Group #:	16
Student Names:	Burcu İskender
	Kürşat Aktaş

1 INTRODUCTION

Bu labarotuvarda bir white-box test tekniği olan mutasyon testinin uygulamasını gerçekleştirdik.Bunun için Jumble toolundan faydalandık.Geçen lablarda oluşturduğumuz Range ve DataUtilies test classları için mutasyon testini uyguladık.Mutasyon testi rastgele mutantlar oluşturmakta ve bizim oluşturduğumuz test durumlarının bunların kaçını öldürdüğünü göstermekte ve bize bir oran oluşturmaktadır.Bu oran da yazdığımız testin kalitesini ölçmede yardımcı olmaktadır.Oluşan ve öldürülemeyen mutantlara göre test durumlarımızda iyileştirmeler yapıp ya da yeni test durumları ekleyip bu oranı yükseltmeye çalıştık, ve daha güçlü test classları oluşturmuş olduk.

2 JFREECHART MUTATION TESTING

2.1 ANALYSIS OF 10 MUTANTS OF THE RANGE CLASS INCLUDING CODE SNIPPETS

-203. Satırda mutant combine metodu içinde range2 == NULL değerini range2 != NULL olarak değiştiriyor.Bu durumda bizim yazdığımız test caseler bu durumu cover etmediği için mutant öldürülemiyor. Biz bunun için , range2 değerini null olarak gönderdiğimiz yeni bir test metodu oluşturduk ve bu mutantı yok ettik.

-260. Satırda oluşan mutant expand metodu içinde lower = length * lowerMargin ifadesinde * yı / ye çeviriyor.Burada da 0 ile bölünme durumu ortaya lowerMargin ifadesi 0 olduğunda 0 ile bölünme belirsizliği ortaya çıkıyor. Biz bu mutantı yok etmek için herhangi bir test durumu oluşturamadık.

- 300. Satırda oluşan metod shift metodu içinde if (allowZeroCrossing) durumunu if (!allowZeroCrossing) olarak değiştiriyor. Biz de shift metodu için shift metodu için oluşturduğumuz yeni test case bu mutantı öldürdü.

- 324. Satırda oluşan 2 mutant shiftWithNoZeroCrossing metodunda valuenun küçük olması durumunu tersine çeviriyor. Biz bu mutantla shift ile gelen Range değerinin lower limitinin 0 dan küçük olma durumunu cover etmediğimizi farkettik.Bu yüzden bu mutant test metodlarımızla öldürülemedi. Bunun için yeni bir test case oluşturup mutant öldürdük.

-325. Satırda oluşan iki mutant ise dönüş değerinde bulunan value + deltayı value-deltaya ve mindeki 0 I 1 yapıyor.Yukarıda da bahsettiğim sebeplerden oluşturduğum test metodu bu mutantları da öldürdü.

-341. Satırda oluşan mutant equals metodunda gelen nesne range nesnesi değilse return false değerini return true olarak değiştiriyor. Biz farklı bir nesne olması durumunu cover etmediğimizden bu mutant test classımız öldüremiyor. Bunun için testequalsShouldBeFalse metodumuzda iyileştirme yaparak bu mutantı öldürdük.

-348. Satırda oluşan mutant yine equals metodunda bir if ifadesinden sonra dönüş değerini falsetan true ya çeviriyor. Bu öldürülemeyen mutant ile gelen upper değerinin rangein upper değerine eşit olmadığı durumu test etmediğimizi farkettik bunun için de yine testEquals metodunda iyileştirme yaptık ve test classımız bu mutantı da öldürdü.

- 86. Satırda Range constructorında oluşan mutant ise exception mesajında bir değişiklik yapıyor.Biz bu mutant öldürecek herhangi bir test case üretemedik.

2.2 ALL THE STATISTICS AND THE MUTATION SCORE FOR EACH MUTATED CLASS WITH EACH TEST SUITE CLASS

Range Classı için :

Geçen laboratuvarda oluşturduğumuz RangeTest classı ile Range classına mutation testi yaptığımızda Jumble 98 mutant üretti , test classımızda ‘test class is broken’ hatasını veren metodlarımızı düzelttik düzeltemediklerimizi silmek zorunda kaldık. Düzeltmeleri yaptıktan sonra bizim test classımız oluşan mutantların 79 tanesini öldürdü. Mutation score %80 olarak çıktı. Daha sonra oluşan mutantların konumlarını inceledik ve test classımızın neden öldüremediğini yorumladık ve eksik test caseleri tamamladık , bazılarında da düzenlemeler yaptık. Test classımızda yaptığımız bu iyileştirmelerden sonra oluşan mutantların 91 tanesini test classımız öldürdü ve mutation scoreumuz %92 ye yükseldi.

```
Mutating org.jfree.data.Range
Tests: org.jfree.data.RangeTest
Mutation points = 98, unit test time limit 2.02s
M FAIL: org.jfree.data.Range:86: CP[23] "Range(double, double): require lower (" -> "___jumble___"
M FAIL: (org.jfree.data.Range.java:87): CP[32] ") <= upper (" -> "___jumble___"
M FAIL: (org.jfree.data.Range.java:87): CP[37] ")." -> "___jumble___"
.M FAIL: (org.jfree.data.Range.java:257): CP[100] "Null 'range' argument." -> "___jumble___"
................................M FAIL: (org.jfree.data.Range.java:199): negated conditional
.M FAIL: (org.jfree.data.Range.java:203): negated conditional
.........M FAIL: (org.jfree.data.Range.java:260): * -> /
....M FAIL: (org.jfree.data.Range.java:279): 0 -> 1
.M FAIL: (org.jfree.data.Range.java:300): negated conditional
....M FAIL: (org.jfree.data.Range.java:321): 0.0D -> 1.0D
M FAIL: (org.jfree.data.Range.java:321): negated conditional
.M FAIL: (org.jfree.data.Range.java:322): 0.0D -> 1.0D
.M FAIL: (org.jfree.data.Range.java:324): 0.0D -> 1.0D
M FAIL: (org.jfree.data.Range.java:324): negated conditional
M FAIL: (org.jfree.data.Range.java:325): + -> -
M FAIL: (org.jfree.data.Range.java:325): 0.0D -> 1.0D
.M FAIL: (org.jfree.data.Range.java:328): + -> -
..M FAIL: (org.jfree.data.Range.java:341): 0 -> 1
.....M FAIL: (org.jfree.data.Range.java:348): 0 -> 1
..............
Jumbling took 42.668s
Score: 80%
```

```
<terminated> Run Jumble [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (2 Oca 2017 20:40:32)
Mutating org.jfree.data.Range
Tests: org.jfree.data.RangeTest
Mutation points = 98, unit test time limit 2.01s
M FAIL: org.jfree.data.Range:86: CP[23] "Range(double, double): require lower (" -> "___jumble___"
M FAIL: (org.jfree.data.Range.java:87): CP[32] ") <= upper (" -> "___jumble___"
M FAIL: (org.jfree.data.Range.java:87): CP[37] ")." -> "___jumble___"
.M FAIL: (org.jfree.data.Range.java:257): CP[100] "Null 'range' argument." -> "___jumble___"
..........................................M FAIL: (org.jfree.data.Range.java:260): * -> /
..........M FAIL: (org.jfree.data.Range.java:321): 0.0D -> 1.0D
..M FAIL: (org.jfree.data.Range.java:322): 0.0D -> 1.0D
.............................
Jumbling took 42.966s
Score: 92%
```

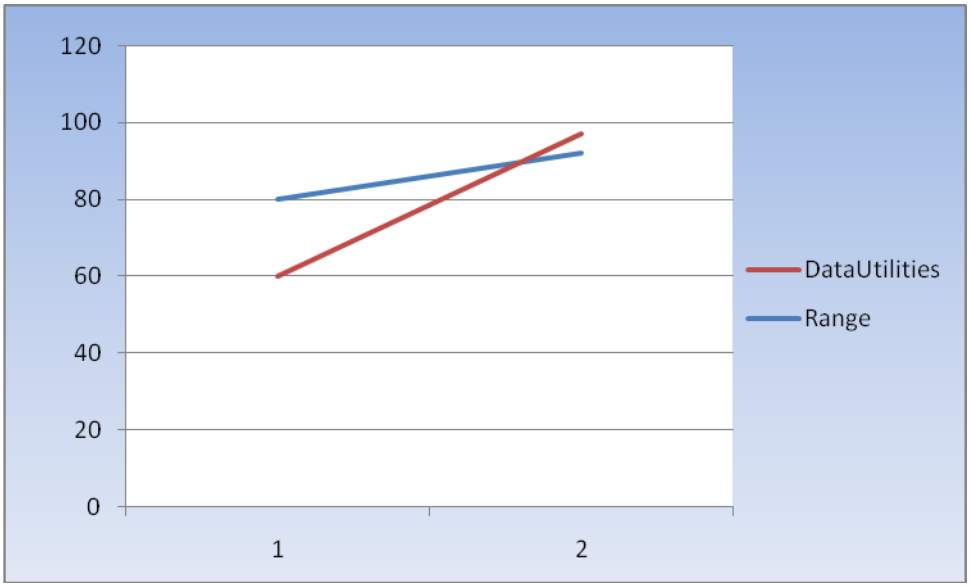
DataUtilies Classı için :

Programı çalıştırdığımızda 40 mutant oluştu ve önceki laboratuvarda oluşturduğumuz test classı bu mutantların 27 tanesini öldürdü ve mutation score %67 olarak çıktı. Daha sonra bu mutantlardan yola çıkarak test classımızdaki eksik yerleri belirledik.Yeni test case oluşturup test classımızı iyileştirdik.İyileştirme yaptıktan sonra çalıştırdığımızda test classımız 39 mutantı öldürdü ve yeni mutation score %97 ye çıktı.

```
<terminated> Run Jumble [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (20 Ara 2016 19:09:56)
Mutating org.jfree.data.DataUtilities
Tests: org.jfree.data.DataUtilitiesTest
Mutation points = 40, unit test time limit 2.8s
.........................M FAIL: (org.jfree.data.DataUtilities.java:178): 0.0D -> 1.0D
M FAIL: (org.jfree.data.DataUtilities.java:179): 0 -> 1
M FAIL: (org.jfree.data.DataUtilities.java:181): negated conditional
M FAIL: (org.jfree.data.DataUtilities.java:182): + -> -
M FAIL: (org.jfree.data.DataUtilities.java:179): += -> -=
M FAIL: (org.jfree.data.DataUtilities.java:179): negated conditional
M FAIL: (org.jfree.data.DataUtilities.java:185): 0.0D -> 1.0D
M FAIL: (org.jfree.data.DataUtilities.java:186): 0 -> 1
M FAIL: (org.jfree.data.DataUtilities.java:188): negated conditional
M FAIL: (org.jfree.data.DataUtilities.java:189): + -> -
M FAIL: (org.jfree.data.DataUtilities.java:191): / -> *
M FAIL: (org.jfree.data.DataUtilities.java:186): += -> -=
M FAIL: (org.jfree.data.DataUtilities.java:186): negated conditional
.
Jumbling took 23.155s
Score: 67%
```

```
Mutating org.jfree.data.DataUtilities
Tests: org.jfree.data.DataUtilitiesTest
Mutation points = 40, unit test time limit 2.97s
.................................M FAIL: (org.jfree.data.DataUtilities.java:186): 0 -> 1
......
Jumbling took 22.347s
Score: 97%
```

2.3 ANALYSIS DRAWN ON THE EFFECTIVENESS OF EACH OF THE TEST SUITES

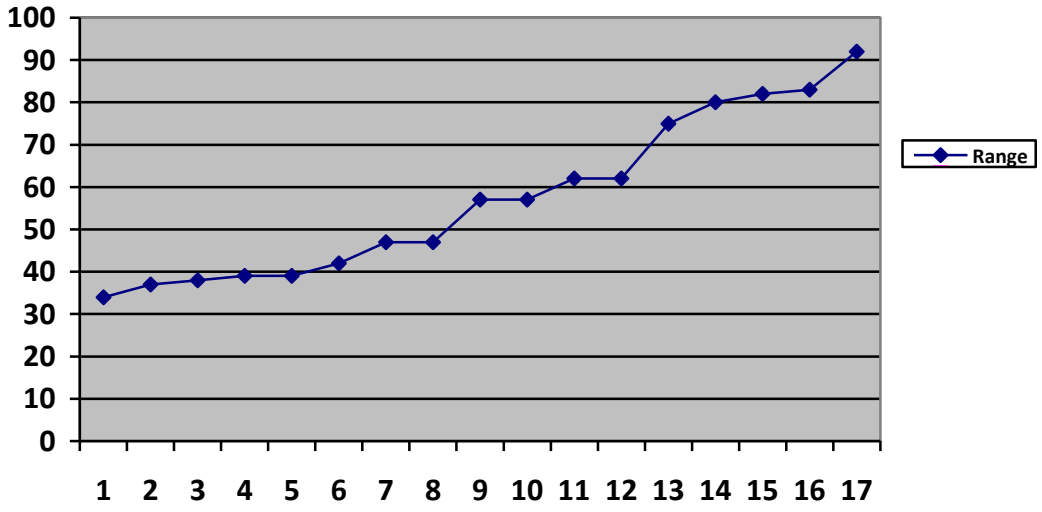


2.4 A DISCUSSION ON THE EFFECT OF EQUIVALENT MUTANTS ON MUTATION SCORE ACCURACY

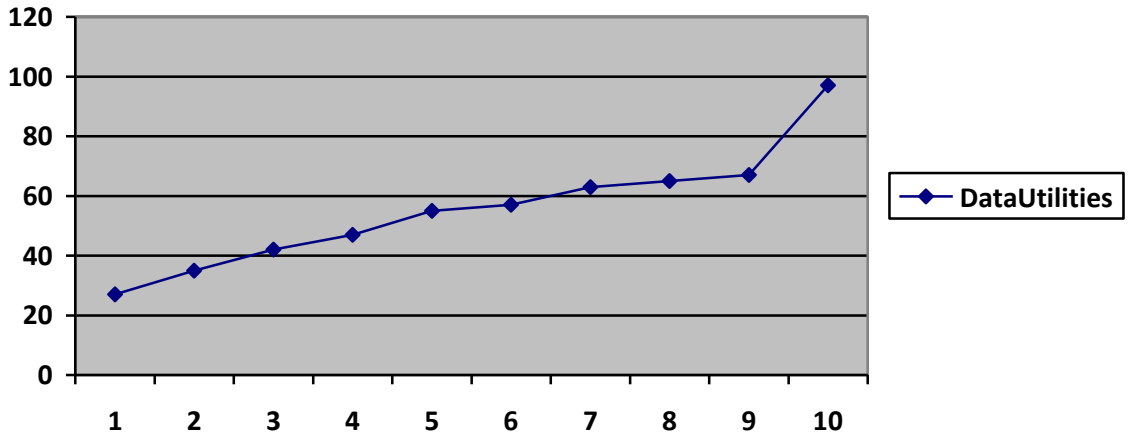
Eşdeğer mutantlar asıl programla aynıdır.Herhangi bir test metodu ile öldürülmemelidir.Mutasyon testinin problem yaratan bir parçası olarak kabul edilebilirler.Öldürülemedikleri için mutation scoreu düşürürler , bu da test classının bir zayıflığı olarak düşünülür.

2.5 ANALYSIS AND CONCLUSIONS DRAWN ON THE EFFECT OF REMOVING OR INCLUDING TEST CASES ON MUTATION SCORE

Range classı için oluşturduğumuz test caseleri birer birer açarak mutation scoru gözlemledik. Tek bir metod varken mutation scoru %34 olarak verdi. Metodları açtıkça skorda artış bazı yerlerde ise sabitlik gözlemlendi. Buradan bazı test metodlarımızın oluşan mutantlardan hiç birini öldüremediği sonucuna vardık.Oluşan mutantlar testlerimizde eksik olan yerleri bulmamıza yardımcı oldu.Gerekli test caseleri ekledik ve düzeltmeler yaparak en son test metodu ile skoru % 92ye yükselttik.



DataUtilities classı için tek bir metodla mutasyon skoru %35 olarak verdi.Geçen laboratvarda oluşturduğumuz metodları açtıkca bu skor %67 ye kadar çıktı. Son olarak bu test ile cover etmediğimiz yeri keşfettik ve yeni bir test metodu ekledik.
‘testCumulativepercantagesThreeKeysAndValues’ metodu ile skoru %97 ye çektik.



2.6 A DISCUSSION OF WHAT COULD HAVE BEEN DONE TO IMPROVE THE MUTATION SCORE OF THE TEST SUITES

Mutation scorunu yükseltmek için oluşan mutantlardan yola çıkılarak yeni test caseler üretilebilir ya da olan test caseler içinde bizim yaptığımız gibi düzenlemeler yapılabilir.Kodun herhangi biryerinde değişiklik yapılarak rastgele oluşan mutantlar bizim test classımız tarafından öldürülmesi gerekmektedir. Bunu bu labda Jumble ile gerçekleştirdik.Düzenlemeleri yaparak mutation scoreları yükselttik.

3 GENERAL ASPECTS

3.1 ADVANTAGES AND DISADVANTAGES OF MUTATION TESTING

Bir önceki laboratuvarda oluşturduğumuz test suit üzerinde code coverage uygulaması yapmıştık.Bütün kaynak kodu ne ölçüde test ettiğimizi oranlamıştık.Bu laboratuvarda uyguladığımız mutasyon testi ile oluşturduğumuz test suiti yeniden test etmiş olduk. Oluşturulan ve öldürülemeyen mutantlar eksik test durumlarını belirlememize ya da olan testler üzerinde iyileştirme yapmamıza yardımcı oldu.Yani mutasyon testi ile ortaya çıkan belirsizlikler, eksikliklerin giderilmesi ile daha kaliteli, güçlü test classları ortaya çıkardık.

Dezavantajları incelenecek olursa mutasyon testi için kaynak kod üzerinde işlem yapıldığı için blacbox test yöntemleri arasında değildir.Ayrıca bir tool yardımı olmadan gerçekleştirileceği düşünülürse oldukça zaman alıcı ve karmaşıktır.

3.2 How the team work/effort was divided and managed

Bu ödevin her aşamasında beraber tek bilgisayar üzerinden çalıştık. Öncelikle Eclipsein Jumble a uyumlu sürümünü araştırdık.Eclipse neon üzerinde Jumble toolunu kurduk.Daha sonra geçen labda oluşturduğumuz test classları ile ve Jumble ile mutasyon testi yaptık.Eksik test caseleri belirledik ve tamamladık.Bazıları üzerinde de iyileştirmeler yaptık.Sonuçları inceleyip, beraber yorumlayıp raporladık.

3.3 Difficulties encountered, challenges overcome, and lessons learned

Bu ödevde eclipse üzerinde Jumble kurarken çok sıkıntı yaşadık. İndirdiğimiz toolda güncelleme yapamıyor çalıştıramıyorduk.Bunun için baya bir zaman kaybettik.Eclipse in neon sürümünü indirerek bu sıkıntıyı hallettik.Onun dışında bu labdan öğrendiğimiz şeyler; bir tool yardımıyla mutasyon testinin nasıl yapılacağı , sonuçların nasıl yorumlandığı ve oluşan mutantların nasıl öldürüldüğüydü.Son olarak da mutasyon skorunun nasıl artırıldığını yeni metodlar ekleyerek ya da olan test metodlarında iyileştirme yaparak deneyimlemiş olduk.

3.4 Comments/feedback on the lab itself

.Geçen labda yaptığımız covaregein yeterli bir test yöntemi olmadığını bu lab ile beraber anlamış olduk.,Bunun dışında ödevin föyü yeterince açıklayıcıydı.Ne yapmamız gerektiği adım adım belirtilmişti, bu da zorlanmadan bu labı tamamlamamıza yardımcı oldu.