**אבטחת איכות תוכנה תשפ"ד תרגיל 2**

1. אתם בודקים פונקציה שממירה ציונים למילים לאותיות על פי הטבלא
2. כתבו קוד שמממש את הפונקציה
3. תכננו מקרי בדיקה בשיטת ה-BVA לאחת המחלקות A-F
4. כתבו טסטים מתאמים ב-JUNIT למקרה הבדיקה מסעיף ב

|  |  |
| --- | --- |
| < 1 | error |
| 1-49 | F |
| 50-59 | E |
| 60-69 | D |
| 70-79 | C |
| 80-89 | B |
| 90-100 | A |
| 100< | error |

**טבלת מקרי בדיקה-**

|  |  |  |  |
| --- | --- | --- | --- |
| Test scenario | Test case | Data | Expected results |
| BVA-ex1- for F partition | Min | 1 | F |
| BVA-ex1- for F partition | Min +1 | 2 | F |
| BVA-ex1- for F partition | Nominal | 25 | F |
| BVA-ex1- for F partition | Max | 49 | F |
| BVA-ex1- for F partition | Max-1 | 48 | F |
| BVA-ex1- for F partition | Error | 50 | error |

1. עליכם לבדוק מחלקה שמסווגת משולשים כפי שנלמד בכיתה.
2. תכננו מקרי בדיקה בשיטת ה-EP לבדיקת המחלקה (הגישו טבלא)
3. הורידו את הפרוייקט שמממש את המחלקה מgithub ל-eclipse. זהו פרוייקט maven שניתן לממש בו גם בדיקות**.** [קישור לפרוייקט](https://github.com/chaya7282/Traingle_Casification_class_1.git)
4. ממשו טסטים מתאימים ב-JUNIT למקרה הבדיקה מסעיף א
5. דווחו על התקלות שנתגלו
6. במקרה ונתגלו תקלות במחלקה תקנו אותן והריצו שוב את הבדיקות
7. הפקידו גירסא מתוקנת לgit-hub עם הערות commit מתאימות .
8. תנו קישור לRepository שמכיל את הפרויקט עם הקוד המתוקן והבדיקות. חשוב שה-repository יהיה public

|  |  |  |  |
| --- | --- | --- | --- |
| Test Scenario | Test Case | Data | Expected Results |
| Equilateral Triangle Classification | All sides are equal | a = 2, b = 2, c = 2 | Triangle\_Types.equilateral |
| Isosceles Triangle Classification | Two sides are equal | a = 2, b = 2, c = 3 | Triangle\_Types.isosceles |
| Scalene Triangle Classification | All sides are different | a = 2, b = 3, c = 4 | Triangle\_Types.scalene |
| Not a Triangle Classification | Sum of two sides less than third side | a = 1, b = 2, c = 3 | Triangle\_Types.notTriangle |
| Test Scenario | Test Case | Data | Expected Results |
| Equilateral Triangle Classification | All sides are equal | a = 2, b = 2, c = 2 | Triangle\_Types.equilateral |
| Isosceles Triangle Classification | Two sides are equal | a = 2, b = 2, c = 3 | Triangle\_Types.isosceles |

1. עליכם לבדוק מחלקה שמסווגת שורשים כפי שנלמד בכיתה.
2. תכננו מקרי בדיקה בשיטת ה-BVA לבדיקת המחלקה (הגישו טבלא)
3. הורידו את הפרוייקט שמממש את המחלקה מgithub ל-eclipse. זהו פרוייקט maven שניתן לממש בו גם בדיקות**.** [קישור לפרוייקט](https://github.com/chaya7282/Root_classification.git)
4. ממשו טסטים מתאימים ב-JUNIT למקרה הבדיקה מסעיף א
5. דווחו על התקלות שנתגלו
6. במקרה ונתגלו תקלות במחלקה תקנו אותן והריצו שוב את הבדיקות
7. הפקידו גירסא מתוקנת לgit-hub עם הערות commit מתאימות .
8. תנו קישור לRepository שמכיל את הפרויקט עם הקוד המתוקן והבדיקות. חשוב שה-repository יהיה public

|  |  |  |  |
| --- | --- | --- | --- |
| Test Scenario | Test Case | Data | Expected Results |
| Boundary check for parameter a | a is below lower boundary | a = -1, b = 10, c = 10 | Root\_Types.ERROR |
| Boundary check for parameter a | a is at lower boundary | a = 0, b = 10, c = 10 | Root\_Types.Two\_roots |
| Boundary check for parameter a | a is above lower boundary | a = 1, b = 10, c = 10 | Root\_Types.Equal\_roots |
| Boundary check for parameter a | a is below upper boundary | a = 49, b = 10, c = 10 | Root\_Types.Equal\_roots |
| Boundary check for parameter a | a is at upper boundary | a = 50, b = 10, c = 10 | Root\_Types.Equal\_roots |
| Boundary check for parameter a | a is above upper boundary | a = 51, b = 10, c = 10 | Root\_Types.ERROR |
| Boundary check for parameter b | b is below lower boundary | a = 10, b = -1, c = 10 | Root\_Types.ERROR |