#### **חלק 1 (50 נקודות):**

**שאלה 1:** מהי התפקיד של האנוטציה @BeforeEach בבדיקות JUnit?

A. להפעיל קוד לפני כל בדיקה במחלקה.

B. להפעיל קוד אחרי כל בדיקה במחלקה.

C. להפעיל קוד פעם אחת בלבד לפני הבדיקה הראשונה במחלקה.

D. להפעיל קוד פעם אחת בלבד אחרי הבדיקה האחרונה במחלקה.

**פרט:**

**מתבצעת לפני כל טסט במחלקה**

**שאלה 2:** מהו השימוש העיקרי בטסטים פרמטריים ב-JUnit?

A. לבצע אותו טסט עם ערכי קלט שונים במקום לכתוב מספר טסטים נפרדים.

B. לבדוק את זמן הריצה של הטסטים ולאופטימזציה.

C. לקבוע את סדר הרצת הטסטים.

D. לקבוע אילו טסטים ירוצו במקביל.

**פרט:**

**הטסט מבצע את אותה הבדיקה כל פעם על הפרמטר הבא ובכך חוסך קוד כפול**

**שאלה 3:** באילו מקרים מומלץ להשתמש באנוטציה @Disabled ב-JUnit 5?

A. כאשר רוצים להריץ טסט באופן אוטומטי בזמן ההרצה.

B. כאשר הטסט עדיין בפיתוח ולא מוכן להרצה.

C. כאשר הטסט מבוצע על פונקציונליות שהוסרה או שונתה.

D. כאשר המנהל מבקש לדלג על הטסט.

**פרט:**

**נועד לבטל זמנית טסט למשל כשלא סיימנו לפתח אותו או שהוא לא תקף אז במקום למחוק אותו משביתים אותו באופן זמני**

#### **חלק 2 (50 נקודות):**

בתרגיל הזה תכתבו בדיקות למחלקה Calculator.

מחלקה זו מספקת פעולות בסיסיות: חיבור, חיסור, כפל וחילוק. התרגיל כולל יצירת Extension מותאם אישית ב-JUnit 5, שימוש בבדיקות פרמטריות, ויישום TestWatcher למעקב אחרי תוצאות הבדיקות.

**מחלקת Calculator:**

|  |
| --- |
| **public class Calculator {   public int add(int a, int b) {  return a + b;  }   public int subtract(int a, int b) {  return a - b;  }   public int multiply(int a, int b) {  return a \* b;  }   public int divide(int a, int b) {  if (b == 0) {  throw new ArithmeticException("Cannot divide by zero");  }  return a / b;  } }** |

### **דרישות למחלקת הבדיקות:**

1. **Lifecycle Annotations**:
   * השתמשו ב-@BeforeAll, @BeforeEach, @AfterEach, ו-@AfterAll לניהול פעולות הכנה וניקוי סביבת הבדיקות.
2. **Custom JUnit Extension**:
   * צרו הרחבה מותאמת אישית שתאתחל לוג של פעולות לפני תחילת הבדיקות ותנקה אותו לאחר סיום הבדיקות.
3. **Parameterized Tests**:
   * השתמשו בבדיקות פרמטריות כדי לבדוק את שיטות add ו-divide עם סטים שונים של ערכים. בחרו 2 שיטות שונות לביצוע פרמטריות.
4. **TestWatcher**:
   * יישמו TestWatcher כדי לרשום תוצאות הצלחה או כישלון של כל בדיקה.