דו"ח תרגיל 1 – קורס אוטומציה:

שם התרגיל: Git – A

מספר הקבוצה: 13

חברי הקבוצה: גל בוטבול 208854745

מיתר חשדי 206911117

אלה גריאני 316120997

מטרת הפרויקט:

במסגרת התרגיל וכחלק מפתיח הקורס, התנסינו בשני תחומים עיקריים:

- היכרות ראשונית עם תוכנת GIT & GITHUB – ההיכרות כללה הורדה ראשונית, פתיחת משתמש קבוצתי, הגדרת ענף וביצוע "דחיפה" של הקוד בהמשך לתוכו.
- ריענון המרה בין בסיסים - בתרגיל זה חזרנו לטכניקות של המרת מספרים בין בסיסים שונים. לאחר מכן, תרגמנו את הטכניקות לקוד כתוב באופן בו המשתמש מזין קלט בבסיס עשרוני/הקסדצימלי ומתקבל פלט הקסדצימלי/עשרוני (בהתאמה).

הנחות יסוד בפיתוח המערכת:

- הקוד מובנה כך שבעת הפעלתו המשתמש יזין את שיטת המעבר (בין בסיסים) הרצויה, לאחר מכן קבלת תוצאה, ולבסוף ההרצה מגיעה לסיומה.

כלומר, אין לולאה אינסופית שנעצרת כשהמשתמש מחליט שמבחינתו השימוש הסתיים, אלא לאחר המרה אחת בלבד. (כמובן שניתן להריץ שוב ושוב ככל שיידרש).

- במידה והתוכנית קלטה מצב קיצון כמו קלט לא תקין, תופעל דרך ההתמודדות כמפורט בסעיף הבא, וככלל התוכנית תיעצר.

תיאור מצבי קיצון ושיטות פתרון:

מצב קיצון	שיטת פתרון
בחירה שגויה מתפריט	הקוד נבנה על בסיס Switch-Case כך שניתן לבחור באופציות 1 (המרה מעשרוני להקסדצימלי) או 2 (המרה מהקסדצימלי לעשרוני). כל אופציה אחרת תזרוק את המשתמש לברירת המחדל שם תודפס הודעת שגיאה: Error: Invalid choice. Please enter 1 or 2.
קלט עשרוני לא חוקי	עטיפה של קריאת הקלט ב-Try/Catch כך שאם הבדיקה הראתה שמדובר במספר עשרוני לא חוקי (הבדיקה נעשתה באמצעות שורת הקוד (!scanner.hasNextInt()) תיזרק שגיאה של InputMismatchException שלאחר תפיסתה תדפיס את ההודעה הבאה: Error: Please enter a valid numeric choice.
קלט עשרוני שלילי	תודפס הודעת השגיאה: Error: Negative numbers are not supported.
קלט ריק	תודפס הודעת השגיאה: Error: input cannot be empty.
תווים לא חוקיים בהקסדצימלי	תיזרק שגיאה שלאחר תפיסתה תופיעה ההודעה הבאה: Error: Invalid characters in hexadecimal input.
אותיות קטנות בהקסדצימלי	כאן התוכנית לא תיעצר, אלא תבוצע המרה לאותיות גדולות באמצעות toUpperCase() ולאחר מכן ימשיך החישוב.
מספר הקסדצימלי ארוך מאוד	יוכל לגרום לבעיית OverFlow ולכן באם קרה מצב זה נזרקת שגיאה אריתמטית המציגה את ההודעה הבאה: Overflow: Hexadecimal number too large.

תיעוד קוד התוכנית:

1. הודעות פתיחה המכוונות את המשתמש לאפשרויות המוצעות וקליטת הבחירה באמצעות Scanner.

```
public class Ex1Auto {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);
        try {
            System.out.println("Which base conversion would you like to choose?");
            System.out.println("1. Decimal To Hex");
            System.out.println("2. Hex To Decimal");
            System.out.print("Enter your choice (1 or 2): ");
            int choice = scanner.nextInt();
```

2. שימוש ב- Switch-Case לפי בחירת המשתמש.

ב-Case הראשון מתבצעת קליטה של מספר עשרוני, לאחר מכן בדיקות תקינות באופן הבא:

התנאי הראשון בודק האם הקלט הוא מספר עשרוני ולא הוכנסו תווים לא חוקיים.

התנאי השני בודק שהקלט אינו מספר שלילי.

במידה ואין בעיות תקינות, מבוצעת פנייה לפונקציית ההמרה.

```
switch (choice) {
case 1:
    System.out.print("Enter a decimal number to convert: ");
    if (!scanner.hasNextInt()) throw new InputMismatchException();
    int decimalNumber = scanner.nextInt();
    if (decimalNumber < 0) {
        System.out.println("Error: Negative numbers are not supported.");
        return;
    }
    String hexResult = decimalToHex(decimalNumber);
    System.out.println("The result in hexadecimal: " + hexResult);
    break;
```

3. ב-Case השני באופן דומה, מתבצעת קליטה (הפעם של מספר הקסדצימלי), לאחריו יגיעו בדיקות התקינות, המרה לאותיות גדולות ואז פנייה לפונקציית ההמרה שגם בה ישנן בדיקות נוספות.

```
case 2:
    System.out.print("Enter a hexadecimal number to convert: ");
    String hexNumber = scanner.next();
    if (hexNumber.trim().isEmpty()) {
        System.out.println("Error: Hexadecimal input cannot be empty.");
        return;
    }
    hexNumber = hexNumber.toUpperCase();
    try {
        int decimalResult = hexToDecimal(hexNumber);
        System.out.println("The result in decimal: " + decimalResult);
    } catch (IllegalArgumentException e) {
        System.out.println("Error: Invalid characters in hexadecimal input.");
    }
    break;
```

4. במידה ולא נבחרה אפשרות נכונה, הקוד יגיע למצב הdefault בו תודפס הודעת שגיאה.

לאחר מכן ניתן לראות את התפיסה של החריגה בעת הזנת קלט של מספר עשרוני לא תקין לבסוף סגירה של ה-Scanner.

```
        default:
            System.out.println("Error: Invalid choice. Please enter 1 or 2.");
        }
    } catch (InputMismatchException e) {
        System.out.println("Error: Please enter a valid numeric choice.");
    } finally {
        scanner.close();
    }
```

5. פונקציית המרה מבסיס עשרוני להקסדצימלי:

- המקרה הפשוט: החזר '0' אם הקלט הינו '0'
- בניית מחרוזת לתוצאה שתוחזר
- חלוקה ב16 בכל איטרציה, אם השארית היא פחות מ10 נחזיר תו עשרוני ואם השארית בין 10 ל15 נמיר לתווים 'A' – 'F'
- הכנסת התו במקום הראשון
- חלוקת המספר ב16 בסיום עד שמגיעים ל-0.

```
// Function to convert decimal to hexadecimal
public static String decimalToHex(int decimal) {
    if (decimal == 0) return "0";
    StringBuilder hex = new StringBuilder();
    while (decimal > 0) {
        int remainder = decimal % 16;
        char hexChar = (char) (remainder < 10 ? '0' + remainder : 'A' + (remainder - 10));
        hex.insert(0, hexChar);
        decimal /= 16;
    }
    return hex.toString();
}
```

6. פונקציית המרה מבסיס הקסדצימלי לבסיס עשרוני:

- ריצה על התווים במחרוזת החל מהתו הראשון
- בדיקה אם התו הוא בין 0 ל9 – מחשבים את הערך המספרי שלו באמצעות החסרה של '0'.
  אם התו בין 'A' – 'F' – מחשבים את הערך בעזרת החסרה של A והוספה של 10
- תמיכה גם באותיות קטנות
- אם יש תו לא בסט התנאים הללו הערך יהיה (-1) ולאחר מכן תיזרק שגיאה של תווים לו חוקיים במחרוזת.
- הכפלה של המספר העשרוני ב16 והוספת הערך של התו הנוכחי
- בדיקה שהערך לא גדול מדי

- החזרת התוצאה

```java
// Function to convert hexadecimal to decimal
public static int hexToDecimal(String hex) {
    int decimal = 0;
    for (int i = 0; i < hex.length(); i++) {
        char hexChar = hex.charAt(i);
        int value = (hexChar >= '0' && hexChar <= '9') ? hexChar - '0' :
            (hexChar >= 'A' && hexChar <= 'F') ? hexChar - 'A' + 10 : -1;
        if (value == -1) {
            throw new IllegalArgumentException("Invalid hexadecimal character");
        }
        decimal = decimal * 16 + value;
        if (decimal < 0) { // Overflow check
            throw new ArithmeticException("Overflow: Hexadecimal number too large.");
        }
    }
    return decimal;
}
```

סיכום ומסקנות:

בתרגיל זה שילבנו והעשרנו את הידע שלנו כיחידים וכקבוצה בנושא המרות בין בסיסים, כתיבת קוד בפורמטים ובטכניקות שלמדנו במהלך השנים, במסגרת הקורסים השונים בתואר. בנוסף לכך, התנסינו לראשונה בעבודה למול תוכנות ה- GIT & GITHUB . נוכחנו לראות את הדרך בה עובדים עם תוכנה זו שמוכרת מאוד בתעשייה ולצבור ניסיון ראשוני בתחום.