מבוא לתכנות – עבודה 3

רקורסיה

מתרגל אחראי: אלעד מרקו

תאריך פרסום: **3.12.17**

תאריך אחרון להגשה: **17.12.17, עד לשעה 23:55**

הוראות מקדימות

1. יש לבצע ולהגיש את העבודה **כיחידים**, כלומר כל סטודנט יבצע ויגיש את עבודתו שלו.
2. לעבודה מצורפים שני קבצי שלד: "Warmup.java" ו-"Suduko.java".
   - יש להשלים את הקוד במקומות המתאימים.
   - בתחילת כל קובץ יש לשנות את השורה הראשונה בהערות שתכיל את שם המלא ותעודת הזהות של המגיש/ה. לדוגמה, במקום "Author: ADD YOUR NAME!" יש לכתוב "Author: Elad Marco 123456789".
   - צרו תיקייה חדשה והעתיקו אליה את הקבצים "Warmup.java" ו-"Sudoku.java".
   - עליכם להוסיף לקבצים אלו את הפתרונות שלכם בהתאם למפורט בתרגיל, ולהגיש את הקבצים כקובץ ZIP יחיד.
   - אין להגיש קבצים נוספים ואין ליצור תיקיות מיותרות (הנוצרות לדוגמה ביצירת package חדש ב-eclipse).
   - שם הקובץ המכווץ יהיה מספר תעודת הזהות של הסטודנט/ית. לדוגמה, 123456789.zip.
   - **עבודות שיוגשו בפורמט שונה מהמתואר לא ייבדקו.**
3. את קובץ ה-zip יש להגיש דרך מערכת ההגשה: https://subsys.ise.bgu.ac.il/submission/login.aspx
4. ניתן להגיש את הקובץ מספר פעמים עד לתאריך ההגשה, ההגשה האחרונה היא זו שתיבדק.
5. **העבודה תיבדק באופן אוטומטי לפי הפלט אשר התוכניות שלכם תדפיס למסך. לכן, אנא הדפיסו אך ורק את הפלט הנדרש, הקפידו על ההוראות ובצעו אותן במדויק. כל הדפסה אשר אינה עונה באופן מדויק לדרישות המופיעות בעבודה (כולל שורות, רווחים, סימני פיסוק, או כל תו אחר מיותר, חסר, או מופיע בסדר שונה מהנדרש), תגרור פגיעה משמעותית בציון (80 נקודות).**
6. סגנון כתיבת הקוד ייבדק באופן ידני (20 נקודות). יש להקפיד על כתיבת קוד ברור, מתן שמות משמעותיים למשתנים, הזחות (אינדנטציה), והוספת הערות בקוד המסבירות את תפקידם של מקטעי הקוד השונים שלדעתכם אינם

אינטואיטיביים להבנה במבט ראשון ובתחילת כל פונקציה שאתם כותבים. אין צורך למלא את הקוד בהערות סתמיות (כמו למשל לכתוב הערת הסבר עבור השורה int num = 0;), אך חשוב לכתוב הערות בנקודות קריטיות המסבירות קטעים חשובים בקוד. כתיבת קוד אשר אינו עומד בסטנדרטים אלו תגרור הפחתה בציון העבודה. בנוסף, הערות יש לרשום אך ורק באנגלית, כתיבת הערות בכל שפה אחרת שקולה לאי כתיבת הערות.

7. במידה ואינכם בטוחים מהו הפירוש המדויק להוראה מסוימת או אם יש לכם שאלה אחרת הקשורה לתוכן העבודה, אנא היעזרו בפורום או בשעות הקבלה של האחראי על העבודה, אלעד (שעות הקבלה מופיעות באתר הקורס תחת הלשונית "דרכים ליצירת קשר ושעות קבלה). בכל בעיה אישית הקשורה בעבודה (מילואים, אשפוז וכו') אנא צרו את הפניה המתאימה למייל הקורס (ise.intro.181@gmail.com).
8. ניתן לפתור את העבודה במעבדות המחשבים של האוניברסיטה או במחשבים האישיים. שימו לב! העבודות יבדקו במחשבי האוניברסיטה ולכן חובה לוודא שהעבודה עובדת כראוי על מחשבי האוניברסיטה!
9. אין להשתמש בחומר שאינו נלמד בכיתה עד לרגע פרסום התרגיל.
10. מותר להוסיף פונקציות עזר כרצונכם.
11. לחלק מהפונקציות הוספנו פקודת return על מנת לאפשר לקוד להתקמפל. כלומר, עבור פונקציה שאמורה להחזיר לדוגמה ערך boolean, הוספנו את השורה return false כברירת מחדל כדי שהקוד יתקמפל. עליכם לשנות את הערכים הללו כך שהפונקציה תחזיר את הערך הנכון.
12. החשיבות העליונה היא על **נכונות** הקוד. כמו-כן, על הקוד להיות קריא כפי שהוגדר בסעיף 8 ולתת את הפלט הנכון.
13. מטרות העבודה: פיתוח שליטה ברקורסיה.

חלק ראשון - חימום (32 נקודות)

השלד לחלק זה נתון בקובץ "Warmup.java".

משימה 1 – בדיקה האם ספרה מופיעה במספר (8 נקודות)

השלימו את הפונקציה **הרקורסיבית**

```
public static boolean doesDigitAppearInNumber(int number, int digit)
```

אשר מקבלת מספר שלם חיובי number וספרה digit. הפונקציה תחזיר true אם הספרה מופיעה במספר, אחרת תחזיר false.

דוגמאות:

עבור הקריאה *doesDigitAppearInNumber*(324, 1) יוחזר false.

עבור הקריאה *doesDigitAppearInNumber*(321, 1) יוחזר true.

עבור הקריאה *doesDigitAppearInNumber*(3212, 2) יוחזר true.

הניחו שהקלט תקין ואל תבדקו זאת. כלומר, הניחו ש-number הוא מספר שלם גדול מ-0, וערך של digit הינו בין 0 ל-9.

משימה 2 – מציאת כמות ספרות זוגיות (8 נקודות)

השלימו את הפונקציה **הרקורסיבית** `public static int countNumberOfEvenDigits(int number)`

אשר מקבלת מספר שלם חיובי ומחזירה את מספר הספרות הזוגיות שבו.

דוגמאות:

עבור הקריאה *countNumberOfEvenDigits*(32414) יוחזר 3, שכן המספר 32414 מכיל שלוש ספרות זוגיות: 2, 4, 4

עבור הקריאה *countNumberOfEvenDigits*(32514) יוחזר 2, כי מספר 32514 מכיל שתי ספרות זוגיות: 2 ו-4

עבור הקריאה *countNumberOfEvenDigits*(1357) יוחזר 0, כיוון שמספר 1357 אינו מכיל ספרות זוגיות.

הניחו שהקלט תקין ואל תבדקו זאת. כלומר, הניחו ש-number הוא מספר שלם גדול מ-0.

משימה 3 – ספירת מופעים של תו במחרוזת (8 נקודות)

השלימו את הפונקציה **הרקורסיבית**

```
public static int countTheAmountOfCharInString (String str, char ch)
```

המקבלת מחרוזת str ותו ch ומחזירה את מספר המופעים של התו במחרוזת.

הניחו שהקלט תקין, כלומר שהמחרוזת str אינה null ואל תבדקו זאת.

דוגמאות:

עבור הקריאה *countTheAmountOfCharInString* ("Information systems", 'o') יוחזר 2.

עבור הקריאה *countTheAmountOfCharInString* ("Information systems", 'a') יוחזר 1.

עבור הקריאה *countTheAmountOfCharInString* ("Information systems", 'b') יוחזר 0.

משימה 4 – (8 נקודות)

השלימו את הפונקציה **הרקורסיבית**:

```
public static boolean checkIfAllLettersAreCapitalOrSmall (String str)
```

המקבלת מחרוזת ומחזירה true אם כל התווים בה הם רק אותיות גדולות באנגלית או רק אותיות קטנות באנגלית, או false אחרת.

הניחו שהמחרוזת str אינה null ואל תבדקו זאת.

<u>דוגמה</u>: עבור ("ABCD") `checkIfAllLettersAreCapitalOrSmall` יוחזר true.
עבור ("abcd") `checkIfAllLettersAreCapitalOrSmall` יוחזר true.
עבור ("AkJL") `checkIfAllLettersAreCapitalOrSmall` יוחזר false.

חלק שני – מימוש משחק סודוקו (68 נקודות)

בחלק זה של העבודה תממשו משחק סודוקו.
השלד לחלק זה נתון בקובץ "Sudoku.java", ובמהלך כתיבת הפתרון מותר ומומלץ להוסיף פונקציות עזר משלכם.

<u>כללי המשחק</u>

במשחק הסודוקו יש לשבץ בלוח משבצות 9X9 מספרים מ-1 עד 9 כך שיתקיימו מספר אילוצים:

1) בכל שורה יופיעו כל המספרים.
2) בכל עמודה יופיעו כל המספרים.
3) הלוח מחולק ל-9 קוביות בגודל 3X3. יש להבטיח כי בכל קובייה כזו יופיעו כל המספרים.
4) בתחילת המשחק נתון חלק מהשיבוץ ועל השחקן להשלים לשיבוץ מלא וחוקי.

למידע נוסף על משחק סודוקו לחצו כאן.

במשימה זו תתבקשו לבצע חלק מתהליך הפתרון של משחק הסודוקו.
כל תא בלוח יכול להכיל מספר מ-1 עד 9. לכן נאמר כי תחום הערכים (Domain) ההתחלתי של תא הינו 1-9. אולם, בהינתן שיבוץ חלקי של לוח, ניתן להקטין את התחום של חלק מהתאים. כך למשל, אם נתון כי בשורה שלוש בתא הראשון מופיע המספר 7 הרי שניתן להסיר את המספר 7 מכל התאים האחרים בשורה שלוש, מכל התאים האחרים בעמודה אחד ומכל התאים בקובייה השמאלית עליונה. כאשר נצמצם את התחום של תאים, יתכן כי נגיע למצב בו יוותר מספר בודד בתחום של תא x. מכאן שתא x חייב לקבל את המספר הנ"ל, שכן זאת האפשרות היחידה. נציב את המספר בתא x, וכעת ניתן להסירו מהתחום של כל התאים התלויים בתא x, וחוזר חלילה.

בסיומו של תהליך זה יתכן כי כל המשבצות בלוח יכילו מספר והמשחק מסתיים, אולם יתכן כי גם לאחר סיום התהליך יהיו משבצות בהן יוותרו מספר מספרים בתחום. סודוקו הניתנים לפתרון בתהליך להלן ידועים כסודוקו קלים, בעוד סודוקו מהסוג השני ידועים כסודוקו קשים.

משימה 1 (24 נקודות)

בחלק זה של העבודה עליכם לממש את הפונקציה הלא רקורסיבית (איטרטיבית):

```
public static int[][][] eliminateDomains(int[][] board)
```

בהינתן מערך דו מימדי board בגודל 9X9 המכיל שיבוץ חלקי (משבצות פנויות יכילו את המספר 0 שאינו משתתף במשחק), עליכם להחזיר מערך תלת מימדי domains כאשר התא domains[row][column] יכיל מערך ובו כל המספרים שמותר למקם במשבצת [row][column].

הפונקציה בעצם מבצעת את תהליך האלימינציה שתואר בדף הקודם.

בסיום סבב האלימינציה על הפונקציה לעדכן השמה עבור התאים בהם קיימת אפשרות אחת בלבד (תחום של משבצת מכיל רק ערך אחד אפשרי). ההשמה תירשם בלוח שהתקבל כקלט (board).
במידה ובוצע עדכון בתא אחד או יותר, יש לחזור על תהליך האלימינציה. וחוזר חלילה.

הערות:

הניחו שהקלט תקין ואל תבדקו זאת. כלומר, הניחו שמערך board הינו מערך בגודל 9X9, וכל תא במערך מכיל ערך בין 0 ל-9, כאשר 0 מסמן תא פנוי, וכל ערך אחר מסמן שיבוץ בתא.

כמו כן, הניחו ששיבוץ חלקי שנתון במערך board הינו חוקי ואל תבדקו זאת.

<u>הדרכה:</u>

- חשבו על אופן מימוש הפונקציה. ייתכנו מספר אפשרויות למימוש המערך המוחזר.

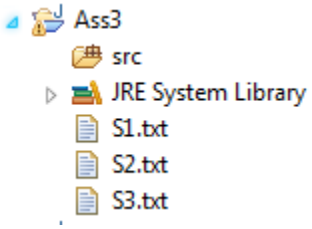
אפשרות אחת לפתרון (ניתן כמובן לממש את הפתרון בכל דרך בה תבחרו) היא הגדרת המימד השלישי במערך domains להיות בגודל 9, כלומר: int[][][] domains = new int[9][9][9].

אם ניתן לשים ספרה i בתא board[row][col] נסמן זאת על ידי domains[row][col][i-1] = 1.

אם לא ניתן לשים ספרה i בתא board[row][col] נסמן זאת על ידי domains[row][col][i-1] = -1.

- מומלץ להוסיף פונקציות עזר

לנוחיותכם צירפנו לעבודה מספר דוגמאות עבור תשבצי סודוקו. התשבצים ניתנים כקבציי טקסט (S1.txt, S2.txt, S3.txt).
כדי לטעון את הלוח ההתחלתי מקובץ טקסט, סיפקנו לכם פונקצית עזר (אין צורך לשנות אותה אלא רק להיעזר)
`public static int[][] readBoardFromFile(String fileToRead)`
המאפשרת לטעון דוגמא של סודוקו לתוך מערך מסוג int[][]. על הקבצים המסופקים כדוגמאות להיות בתיקיית הפרויקט ב-eclipse (מחוץ ל- src), כמו בתמונה למטה:



משימה 2 (10 נקודות)

משום שניתן לממש מערך domains שמוחזר מפונקציה `eliminateDomains` בדרכים שונות, לצורך בדיקת הפונקציה `eliminateDomains` אתם מתבקשים לכתוב ולממש את הפונקציה

```
public static void printBoard(int[][][] domains, int[][] board)
```

המבצעת הדפסה לפי הפורמט שמתואר בהמשך.

פונקציה `printBoard` תקבל כקלט

- domains – מערך אותו מחזירה הפונקציה eliminateDomains כפלט.
- board - הלוח אותו eliminateDomains מקבלת כקלט ומשבצת בו ערכים לאורך הפתרון.

הניחו שהקלט תקין ואל תבדקו זאת.

פורמט ההדפסה:

עליכם לוודא כי הפלט המתקבל מהתוכנית שתממשו יהיה בפורמט הנדרש.

התוכנית תיבדק על הדוגמאות המצורפות וגם על מספר דוגמאות נוספות.

- ראשית יודפס הלוח (ללא רווחים) שהתקבל בתום הקריאה ל eliminateDomains בפורמט הבא: (עבור הלוח הניתן בקובץ S1.txt).

```
060|104|050
008|305|600
200|609|001
---+---+---
800|437|006
006|852|300
700|961|004
---+---+---
500|703|002
007|206|900
642|598|173
```

- לאחר מכן עבור כל תא בלוח יודפסו כל הערכים שעדיין ניתן לשבץ בתום התהליך (לאחר הקריאה לפונקציה eliminateDomains) באופן הבא: (עבור הלוח הניתן בקובץ S1.txt)

```
0,0 = 3,9,
0,1 = 6,
0,2 = 3,9,
0,3 = 1,
0,4 = 2,7,8,
0,5 = 4,
0,6 = 2,7,8,
0,7 = 5,
0,8 = 7,8,9,
1,0 = 1,4,9,
1,1 = 1,7,9,
1,2 = 8,
1,3 = 3,
1,4 = 2,7,
1,5 = 5,
1,6 = 6,
```

1,7 = 2,4,9,
1,8 = 7,9,
2,0 = 2,
2,1 = 3,5,7,
2,2 = 3,4,5,
2,3 = 6,
2,4 = 7,8,
2,5 = 9,
2,6 = 4,7,8,
2,7 = 3,4,8,
2,8 = 1,
3,0 = 8,
3,1 = 1,2,5,9,
3,2 = 1,5,9,
3,3 = 4,
3,4 = 3,
3,5 = 7,
3,6 = 2,5,
3,7 = 1,2,9,
3,8 = 6,
4,0 = 1,4,9,
4,1 = 1,9,
4,2 = 6,
4,3 = 8,
4,4 = 5,
4,5 = 2,
4,6 = 3,
4,7 = 1,9,
4,8 = 7,9,
5,0 = 7,
5,1 = 2,3,5,
5,2 = 3,5,
5,3 = 9,
5,4 = 6,
5,5 = 1,
5,6 = 2,5,8,
5,7 = 2,8,
5,8 = 4,
6,0 = 5,
6,1 = 1,8,9,
6,2 = 1,9,
6,3 = 7,
6,4 = 1,4,
6,5 = 3,
6,6 = 4,8,
6,7 = 4,6,8,
6,8 = 2,
7,0 = 1,3,
7,1 = 1,3,8,

```
7,2 = 7,
7,3 = 2,
7,4 = 1,4,
7,5 = 6,
7,6 = 9,
7,7 = 4,8,
7,8 = 5,8,
8,0 = 6,
8,1 = 4,
8,2 = 2,
8,3 = 5,
8,4 = 9,
8,5 = 8,
8,6 = 1,
8,7 = 7,
8,8 = 3,
```

דוגמה לפלט printBoard הנדרש בהינתן הקלט מקובץ S2.txt:

```
534|678|912
672|195|348
198|342|567
---+---+---
859|761|423
426|853|791
713|924|856
---+---+---
961|537|284
287|419|635
345|286|179
0,0 = 5,
0,1 = 3,
0,2 = 4,
0,3 = 6,
0,4 = 7,
0,5 = 8,
0,6 = 9,
0,7 = 1,
0,8 = 2,
1,0 = 6,
1,1 = 7,
1,2 = 2,
1,3 = 1,
1,4 = 9,
1,5 = 5,
1,6 = 3,
1,7 = 4,
1,8 = 8,
2,0 = 1,
```

2,1 = 9,
2,2 = 8,
2,3 = 3,
2,4 = 4,
2,5 = 2,
2,6 = 5,
2,7 = 6,
2,8 = 7,
3,0 = 8,
3,1 = 5,
3,2 = 9,
3,3 = 7,
3,4 = 6,
3,5 = 1,
3,6 = 4,
3,7 = 2,
3,8 = 3,
4,0 = 4,
4,1 = 2,
4,2 = 6,
4,3 = 8,
4,4 = 5,
4,5 = 3,
4,6 = 7,
4,7 = 9,
4,8 = 1,
5,0 = 7,
5,1 = 1,
5,2 = 3,
5,3 = 9,
5,4 = 2,
5,5 = 4,
5,6 = 8,
5,7 = 5,
5,8 = 6,
6,0 = 9,
6,1 = 6,
6,2 = 1,
6,3 = 5,
6,4 = 3,
6,5 = 7,
6,6 = 2,
6,7 = 8,
6,8 = 4,
7,0 = 2,
7,1 = 8,
7,2 = 7,
7,3 = 4,
7,4 = 1,

```
7,5 = 9,
7,6 = 6,
7,7 = 3,
7,8 = 5,
8,0 = 3,
8,1 = 4,
8,2 = 5,
8,3 = 2,
8,4 = 8,
8,5 = 6,
8,6 = 1,
8,7 = 7,
8,8 = 9,
```

דוגמה לפלט printBoard הנדרש בהינתן הקלט מקובץ S3.txt:

```
842|135|796
653|927|841
719|648|532
---+---+---
394|762|158
587|391|264
261|584|973
---+---+---
926|853|417
475|216|389
138|479|625
0,0 = 8,
0,1 = 4,
0,2 = 2,
0,3 = 1,
0,4 = 3,
0,5 = 5,
0,6 = 7,
0,7 = 9,
0,8 = 6,
1,0 = 6,
1,1 = 5,
1,2 = 3,
1,3 = 9,
1,4 = 2,
1,5 = 7,
1,6 = 8,
1,7 = 4,
1,8 = 1,
2,0 = 7,
2,1 = 1,
2,2 = 9,
2,3 = 6,
```

2,4 = 4,
2,5 = 8,
2,6 = 5,
2,7 = 3,
2,8 = 2,
3,0 = 3,
3,1 = 9,
3,2 = 4,
3,3 = 7,
3,4 = 6,
3,5 = 2,
3,6 = 1,
3,7 = 5,
3,8 = 8,
4,0 = 5,
4,1 = 8,
4,2 = 7,
4,3 = 3,
4,4 = 9,
4,5 = 1,
4,6 = 2,
4,7 = 6,
4,8 = 4,
5,0 = 2,
5,1 = 6,
5,2 = 1,
5,3 = 5,
5,4 = 8,
5,5 = 4,
5,6 = 9,
5,7 = 7,
5,8 = 3,
6,0 = 9,
6,1 = 2,
6,2 = 6,
6,3 = 8,
6,4 = 5,
6,5 = 3,
6,6 = 4,
6,7 = 1,
6,8 = 7,
7,0 = 4,
7,1 = 7,
7,2 = 5,
7,3 = 2,
7,4 = 1,
7,5 = 6,
7,6 = 3,
7,7 = 8,

```
7,8 = 9,
8,0 = 1,
8,1 = 3,
8,2 = 8,
8,3 = 4,
8,4 = 7,
8,5 = 9,
8,6 = 6,
8,7 = 2,
8,8 = 5,
```

משימה 3 (34 נקודות)

בסעיף זה תידרשו לפתור משחק סודוקו בעזרת הפונקציה `eliminateDomains` שכתבתם.

בסיומו של תהליך הסרת הערכים יתכן כי כל המשבצות בלוח יכילו מספרים והמשחק מסתיים (לדוגמה הפלטים עבור הקבצים S2.txt, S3.txt לעיל), אולם יתכן כי גם לאחר סיום התהליך יהיו משבצות בהן יוותרו מספר ערכים אפשריים בתחום. לכל משבצת עבורה יש מספר ערכים אפשריים, עלינו לנסות להציב כל אחד מערכים אלו (באופן רקורסיבי), ולבדוק האם ניתן להגיע לפתרון מלא (שיבוץ תקין של מספרים בכל תא בלוח).

ניתן לנסח את התהליך באופן הבא:

1) הסר את כל הערכים שאינם חוקיים.
2) אם התשבץ פתור – סיים.
3) אם התחום של אחת המשבצות התרוקן – אין פתרון. סיים.
4) מצא משבצת עבורה עדיין לא מצאנו ערך.
5) נסה להציב עבור המשבצת ערך אפשרי ולפתור את הסודוקו שנוצר – אם הצלחת תרשום את הפתרון על גבי הלוח, אם נכשלת נסה ערך אחר.
6) אם ניסית את כל הערכים האפשריים ואף אחד לא פתר – אין פתרון.

עליכם לממש את הפונקציה **הרקורסיבית**:

```
public static boolean solveSudoku(int[][] board)
```

בהינתן מערך דו מימדי board בגודל 9X9 המכיל שיבוץ חלקי (משבצות פנויות יכילו את המספר 0 שאינו משתתף במשחק), עליכם למלא את הלוח בפתרון אפשרי (אם קיים כזה). הפונקציה תמומש באופן רקורסיבי ותחזיר true אם נמצא פתרון או false אם לא ניתן לפתור.

<u>שימו לב</u> – מאחר והניסיון לפתור את הסודוקו כולל ניסיונות להצבת ערכים שיתכן ואינם אפשריים, יש לבצע את שלב 4 על <u>עותק</u> של הלוח המקורי, ורק אם נמצא פתרון לשנות את הלוח המקורי.

- הניחו שהקלט תקין ואל תבדקו זאת. כלומר, הניחו שמערך board הינו מערך בגודל 9X9, וכל תא במערך מכיל ערך בין 0 ל-9, כאשר 0 מסמן תא פנוי, וכל ערך אחר מסמן שיבוץ בתא.

<u>עבודה נעימה!</u>