עבודת בית מספר 2

חלק א'

מערכים, פונקציות ובעיית הספיקות

מבוא למדעי המחשב, סמסטר א' תשע"ט

המחלקה למדעי המחשב, אוניברסיטת בן-גוריון בנגב

מבוא למדעי המחשב – סמסטר א' תשע"ט

עבודת בית מספר 2: מערכים, פונקציות ובעיית הספיקות

צוות העבודה:

מרצה אחראי: פרופ' מיכאל קודיש

מתרגלים אחראים: אבי יצחקוב ונועה בן דוד

תאריך פרסום: 08/11/18

מועד אחרון להגשה: 15/11/18 ב-12:00 (צהריים)

בעבודת בית זו נתרגל עבודה עם מערכים ופונקציות בג'אווה ונפגוש את בעיית הספיקות יחד עם כמה מושגים חשובים נוספים במדעי המחשב.

נכתוב תכנית לפתרון בעיית *הטיול הגדול*: בבעיה זו נתונים קבוצה של ערים וקווי תעופה ביניהן. יש לתכנן מסלול לטיול שיוצא מנמל הבית, עובר בכל שאר הערים וחוזר לנמל הבית, כך שכל עיר תופיע בדיוק פעם אחת במסלול וכן בסיומו נחזור לעיר המקור.

האלגוריתם שנממש לפתרון הבעיה מבוסס על רדוקציה ל"בעיית הספיקות", או באנגלית:
The Boolean Satisfiability Problem (SAT). את הנוסחה שנקבל מהרדוקציה נפתור בעזרת "פותרן של בעיית הספיקות" (SAT Solver).

לעבודה זו שני חלקים שמפורסמים בנפרד: בחלק א' נממש מספר פונקציות לצורך בדיקת תקינות של מופע נתון לבעיית הטיול הגדול וכן מספר פונקציות לווידוא נכונות של פתרון למופע נתון. בחלק ב', נממש את האלגוריתם לפתרון הבעיה שמבוסס על רדוקציה לבעיית הספיקות .

מרכיבי הציון לעבודה: חלק א' – 35 נקודות, חלק ב' – 65 נקודות.

הוראות מקדימות:

הגשת עבודות בית

1. **קראו את העבודה מתחילתה ועד סופה לפני שאתם מתחילים לפתור אותה.** ודאו שאתם מבינים את כל המשימות. רמת הקושי של המשימות אינה אחידה: הפתרון של חלק מהמשימות קל יותר, ואחרות מצריכות חקירה מתמטית - שאותה תוכלו לבצע בספרייה או בעזרת מקורות דרך רשת האינטרנט. בתשובות שבהן אתם מסתמכים על עובדות מתמטיות שלא הוצגו בשיעורים, יש להוסיף כהערה במקום המתאים בקוד את ציטוט העובדה המתמטית ואת המקור (כגון ספר או אתר).
2. עבודה זו תוגש ביחידים. כדי להגיש את העבודה יש להירשם למערכת ההגשות (Submission System). את הרישום למערכת ההגשות מומלץ לבצע כבר עכשיו, טרם הגשת העבודה (קחו בחשבון כי הגשה באיחור אינה מתקבלת). את הגשת העבודה ניתן לבצע רק לאחר הרישום למערכת.
3. לעבודה זו מצורף קובץ Assignment2A.java. עליכם לערוך קובץ זה בהתאם למפורט בתרגיל ולהגישו כפתרון, מכווץ כקובץ ZIP יחיד. שימו לב: עליכם להגיש רק את קובץ ה-Java. אין לשנות את שם הקובץ, ואין להגיש קבצים נוספים. שם קובץ ה-ZIP יכול להיות כרצונכם, אך באנגלית בלבד. בנוסף, הקובץ שתגישו יכול להכיל טקסט המורכב מאותיות באנגלית, מספרים וסימני פיסוק בלבד. טקסט אשר יכיל תווים אחרים (אותיות בעברית, יוונית וכד'..) לא יתקבל. בנוסף מצורפים קבצים עם שמות Task<n>Test.java, כאשר n מסמן את מס' המשימה. קבצים אלה נועדו עבורכם לבדיקת הקוד, והם אינם להגשה.
4. קבצים שיוגשו שלא על פי הנחיות אלו לא ייבדקו. את קובץ ה-ZIP יש להגיש ב-Submission System. פרטים בעניין ההרשמה ואופן הגשת העבודה תוכלו למצוא באתר.

בדיקת עבודות הבית

5. עבודות הבית נבדקות גם באופן ידני וגם באופן אוטומטי.
6. סגנון כתיבת הקוד ייבדק באופן ידני. יש להקפיד על כתיבת קוד ברור, על מתן שמות משמעותיים למשתנים, על הזחות (אינדנטציה), ועל הוספת הערות בקוד המסבירות את תפקידם של מקטעי הקוד השונים. אין צורך למלא את הקוד בהערות סתמיות, אך חשוב לכתוב הערות בנקודות קריטיות המסבירות קטעים חשובים בקוד. הערות יש לרשום אך ורק באנגלית. כתיבת קוד אשר אינה עומדת בדרישות אלו תגרור הפחתה בציון העבודה.

עזרה והנחיה

7. לכל עבודת בית בקורס יש צוות שאחראי לה. ניתן לפנות לצוות בשעות הקבלה. פירוט שמות האחראים לעבודה מופיע במסמך זה וכן באתר הקורס, כמו גם פירוט שעות הקבלה. בשאלות טכניות אפשר גם לגשת לשעות התגבורים, שבהן ניתנת עזרה במעבדה. כמו כן, אתם יכולים להיעזר בפורום ולפנות בשאלות לחבריכם לכיתה. צוות הקורס עובר על השאלות ונותן מענה במקרה הצורך.
8. בכל בעיה אישית הקשורה בעבודה (מילואים, אשפוז וכו'), אנא פנו אלינו דרך מערכת הפניות, כפי שמוסבר באתר הקורס.

הערות ספציפיות לעבודת בית זו

9. בעבודה זו 20 משימות (7 משימות בחלק א' ו-13 משימות בחלק ב') וסך הנקודות המקסימלי הוא 100. הניקוד לכל משימה שווה (5 נקודות).
10. בעבודה זו מותר להשתמש בידע שנלמד עד הרצאה 8 (כולל), וכן עד תרגול 4 (כולל).

יושר אקדמי

הימנעו מהעתקות! ההגשה היא ביחידים. אם מוגשות שתי עבודות עם קוד זהה או אפילו דומה - זוהי העתקה, אשר תדווח לאלתר לוועדת משמעת. אם טרם עיינתם בסילבוס הקורס, אנא עשו זאת כעת.

מומלץ לקרוא היטב את כל ההוראות המקדימות ורק לאחר מכן להתחיל בפתרון המשימות. ודאו שאתם יודעים לפתוח

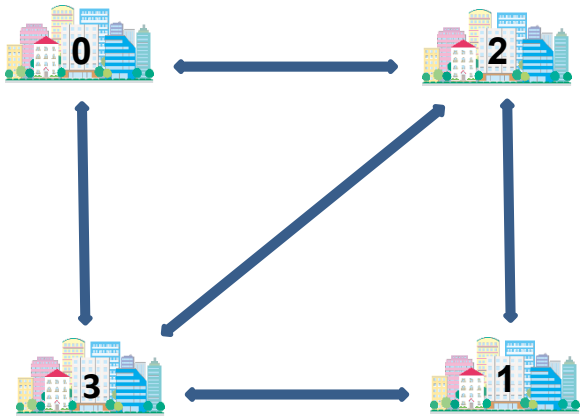
קבוצת הגשה (עבור עצמכם) במערכת ההגשות.

1. בעיית הטיול הגדול

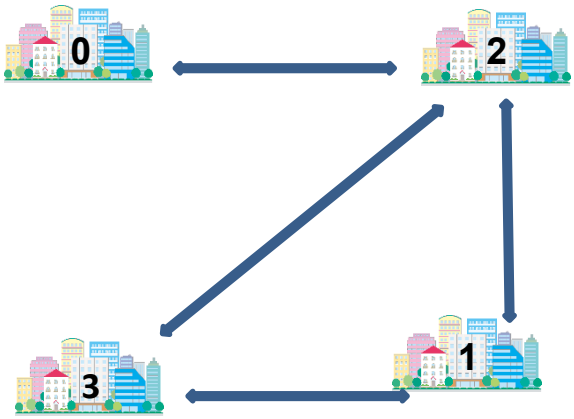
מבוא

בהינתן רשימה של קווי תעופה בין n ערים הממוספרות ב- $\{0 \dots n-1\}$, יש לתכנן מסלול המתחיל מנמל הבית הנמצא בעיר המקור 0, מבקר בכל אחת מן הערים האחרות $\{1 \dots n-1\}$ בדיוק פעם אחת ובסיומו חוזר לנמל הבית שבעיר המקור (מסלול מעגלי). קיומו של קו תעופה $\{i,j\}$ בין שני ערים שונות i,j , מציין שקיימת טיסה בשני הכיוונים $(i \to j,\ j \to i)$.

דוגמא 1: נניח שיש 4 ערים $\{0,1,2,3\}$ ושקווי התעופה הם: $\{1,3\},\{2,3\},\{1,2\},\{0,3\},\{0,2\}$, כפי שמוצג באיור מספר 1. ניתן לתכנן מסלול שפותר את בעיית הטיול הגדול. למשל המסלול: $0 \to 3 \to 1 \to 2 \to 0$.
דוגמא 2: נניח שיש 4 ערים $\{0,1,2,3\}$ ושקווי התעופה הם:$\{1,3\},\{2,3\},\{1,2\},\{0,2\}$, כפי שמוצג באיור מספר 2. בדוגמא זו לא ניתן למצוא מסלול שפותר את בעיית הטיול הגדול.



איור 1. הצגה גרפית של קווי התעופה מדוגמא 1



איור 2. הצגה גרפית של קווי התעופה מדוגמא 2

ייצוג מופע של בעיית הטיול הגדול ב-Java

מופע של בעיית *הטיול הגדול* עבור n ערים מיוצג באמצעות מטריצה בוליאנית flights בגודל $n \times n$, כאשר הערך בתא ה (i,j) במטריצה הוא true אם ורק אם קיים קו תעופה $\{i,j\}$.

הגדרה 1: מערך דו-ממדי flights מייצג מופע חוקי של בעיית הטיול הגדול אם הוא מטריצה בוליאנית שהיא:
א. ריבועית – מכילה n מערכים שכל אחד מהם באורך n.
ב. סימטרית - לכל $0 \le i \le j < n$ מתקיים flights[i][j] = flights[j][i].
ג. אנטי-רפלקסיבית – לכל $0 \le i < n$ מתקיים flights[i][i]=false.

דוגמה: המופע של בעיית הטיול הגדול שמוצג באיור מספר 1 ייוצג ב-Java באופן הבא:

```
boolean[][] flights = {{false, false, true,  true },
                       {false, false, true,  true },
                       {true,  true,  false, true },
                       {true,  true,  true,  false}};
```

ייצוג פתרון לבעיית הטיול הגדול ב Java

פתרון למופע של בעיית *הטיול הגדול* עבור n ערים מיוצג באמצעות מערך חד-ממדי בגודל n של מספרים שלמים, כאשר הערך בתא ה-i מציין את מספר העיר שמבקרים בה בשלב ה-i של הטיול. למשל, המסלול $0 \to 3 \to 1 \to 2 \to 0$ המהווה פתרון עבור המופע שמתואר בדוגמא 1, מיוצג באמצעות המערך [0,2,1,3]. נשים לב שהחזרה לעיר המקור (0) בסוף הטיול לא מיוצגת באופן מפורש במערך.

הגדרה 2: מערך חד-ממדי A של מספרים שלמים, מייצג פתרון למופע של בעיית הטיול הגדול על nערים אם:

א. A הוא מערך באורך n המכיל את כל המספרים $0 \dots n-1$ (כל הערים מופיעות במסלול בדיוק פעם אחת).
ב. A[0]=0 - העיר הראשונה במסלול היא 0.
ג. לכל $0 \le i < n-1$, קיים קו תעופה בין A[i] ל-A[i+1] וכן קיים קו תעופה בין $A[n-1]$ ל-A[0].

וידוא תקינות קלט

במשימות הבאות נממש מספר פונקציות שיסייעו לנו לבדוק האם מופע נתון לבעיית הטיול הגדול הוא תקין.

משימה 1 (מטריצה ריבועית) (5 נקודות):

בהינתן מערך בוליאני דו-ממדי באורך כלשהו, נרצה לוודא כי המערך מייצג מטריצה ריבועית. השלימו את הפונקציה הבאה בקובץ Assignment2A.java:

```
public static boolean isSquareMatrix (boolean[][] matrix)
```

על הפונקציה להחזיר ערך true אם ורק אם המערך matrix מכיל n מערכים באורך n ($n \ge 0$).

הנחות על הקלט וחריגות:

- אין להניח שום הנחות על הקלט.
- יש להחזיר ערך false אם הקלט אינו תקין.
- פונקציה זו לא זורקת חריגות.

דוגמאות:

```
boolean[][] matrix1 = {{false,false},{true,true}} ;
System.out.println(isSquareMatrix(matrix1)); // true ;

boolean[][] matrix2 = {{true,false,true},{false,false}} ;
System.out.println(isSquareMatrix(matrix2)); // false ;

boolean[][] matrix3 = null;
System.out.println(isSquareMatrix(matrix3)); // false ;
```

משימה 2 (מטריצה סימטרית) (5 נקודות):

בהינתן מטריצה בוליאנית בגודל $n \times n$, נרצה לוודא כי המטריצה היא סימטרית. השלימו את הפונקציה הבאה בקובץ Assignment2A.java:

public static boolean isSymmetricMatrix (boolean[][] matrix)

על הפונקציה להחזיר ערך true אם ורק אם לכל $0 \leq i < j < n$ מתקיים matrix[i][j]=matrix[j][i].

הנחות על הקלט וחריגות:
- הניחו כי matrix היא מטריצה ריבועית.
- פונקציה זו לא זורקת חריגות.

דוגמאות:

```
boolean[][] matrix1 = {{false, false, true},
                       {false, false,  true},
                       {true,  true,  true}};
System.out.println(isSymmetricMatrix(matrix1)); // true
```

משימה 3 (מטריצה אנטי-רפלקסיבית) (5 נקודות):

בהינתן מטריצה בוליאנית בגודל $n \times n$, נרצה לוודא כי המטריצה היא אנטי-רפלקסיבית. השלימו את הפונקציה הבאה בקובץ Assignment2A.java:

public static boolean isAntiReflexiveMatrix (boolean[][] matrix)

הנחות על הקלט וחריגות:
- הניחו כי matrix היא מטריצה ריבועית.
- פונקציה זו לא זורקת חריגות.

דוגמאות:

```
int[][] matrix1 = {{false, false},
                   {true,  false }};
System.out.println(isAntiReflexiveMatrix(matrix1)); // true

int[][] matrix2 = {{false, false},
                   {true,  true }};
System.out.println(isAntiReflexiveMatrix(matrix2)); // false
```

משימה 4 (מופע חוקי) (5 נקודות):

בהינתן מערך דו-ממדי, נבדוק שהוא מייצג מופע תקין של בעיית הטיול הגדול. השלימו את הפונקציה הבאה בקובץ Assignment2A.java:

public static boolean isLegalInstance (boolean[][] matrix)

הפונקציה תחזיר ערך true אם ורק אם המערך הדו-ממדי matrix מקיים את התנאים של מופע תקין לפי **הגדרה 1**.

הנחות על הקלט וחריגות:
- אין להניח שום הנחות על הקלט.
- יש להחזיר ערך false אם הקלט אינו תקין.
- פונקציה זו לא זורקת חריגות.

<u>**וידוא פתרון לבעיית הטיול הגדול**</u>

במשימות הבאות נממש מספר פונקציות לצורך וידוא פתרון של בעיית הטיול הגדול.

<u>**משימה 5** (המסלול עובר בכל הערים) (5 נקודות):</u>

הגדרה (פרמוטציה): מערך $array$ יקרא פרמוטציה אם הוא מכיל את על המספרים השלמים בין 0 ל- $array.length - 1$ כולל. כלומר, כל ערך בטווח הנ"ל יופיע <u>בדיוק פעם אחת</u>
דוגמאות: המערכים [0,2,1,3], [1,0] הם פרמוטציות ואילו המערכים [0,1,2,2], [0,2], [1,4,3,2] אינם פרמוטציות.

השלימו את הפונקציה הבאה בקובץ Assignment2A.java::

```
public static boolean isPermutation (int[] array)
```

הפונקציה תחזיר את הערך true אם ורק אם המערך array הוא פרמוטציה.

הנחות על הקלט וחריגות:
- הניחו כי array אינו null.
- פונקציה זו לא זורקת חריגות.

דוגמאות:

```
int[] array1 = {0,2,3,1};
System.out.println(isPermutation(array1)); //true

int[] array2 = {1,4,3,2};
System.out.println(isPermutation(array2)); //false
```

<u>**משימה 6** (כל הטיסות במסלול קיימות) (5 נקודות)</u>

בהינתן מערך דו ממדי בוליאני flights המייצג מופע של בעיית הטיול הגדול ומערך חד-ממדי של מספרים שלמים, tour, שמייצג מסלול, נבדוק שבין כל שני ערים עוקבות במסלול קיימת טיסה. השלימו את הפונקציה הבאה בקובץ Assignment2A.java:

```
public static boolean hasLegalSteps (boolean[][] flights, int[] tour)
```

הפונקציה מקבלת מערך flights דו מימדי בגודל $n \times n$ ומערך *tour* באורך *n*. הפונקציה תחזיר ערך true אם ורק אם לכל $0 \leq i < n - 1$ יש קו תעופה בין tour[i] ל-tour[i+1] וגם יש קו תעופה חזרה מ-tour[n-1] ל- tour[0].

הנחות על הקלט וחריגות:

- הניחו שהמערך flights מייצג מופע תקין על n ערים.
- המערך tour הוא באורך n וערכיו הם מהתחום [0,n-1].
- פונקציה זו לא זורקת חריגות.

<u>משימה 7</u> <u>(פתרון חוקי) (5 נקודות):</u>

בהינתן מערך דו-ממדי flights המייצג מופע של בעיית הטיול הגדול ומערך חד-ממדי tour, נבדוק שהמערך מהווה פתרון למופע. השלימו את הפונקציה הבאה בקובץ Assignment2A.java:

```
public static boolean isSolution(boolean[][] flights, int[] tour)
```

הפונקציה תחזיר ערך true אם ורק אם המערך tour מקיים את התנאים לפתרון לפי **הגדרה 2** עבור המופע flights.

הנחות על הקלט וחריגות:
- הניחו שהמערך flights מייצג מופע תקין על $n \geq 0$ ערים.
- אין להניח שום הנחות על המערך tour.
- פונקציה זו זורקת חריגה אם tour אינו מערך באורך n.

בהצלחה!