

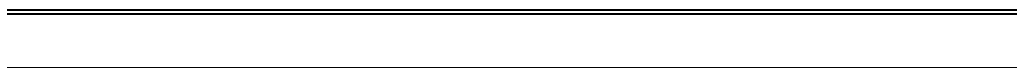
מטלה 2

לולאות ומערכים

לפניות ליבגני בנושאי הקורס והמטלה:

software.systems.a.ariel@gmail.com

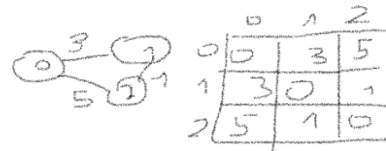
- המטלה היא בזוגות או ביחידים – **לא שלשות**
- עליכם לבצע את פקודת הקומפילציה עם הדגל – Wall על מנת לוודא שתוכניתכם ללא אזהרות. תוכנית שמתקמפלת עם אזהרות תגרור הורדת נקודות.
- עליכם לוודא שתוכניתכם מתקמפלת ורצה על גבי מערכת הפעלה ubuntu עם קומפיילר gcc .
- יש להגיש את המטלה ב- git . יש להגיש קובץ txt . השורה הראשונה תכלול את הכתובת של ה- git (לא URL) מאתר ה-github. השורה השנייה תכלול את מזהה ה-commit הרלוונטי והשורה השלישית את תעודת הזהות של הסטודנטים המגישים מופרדים ברווח.
- הנכם נדרשים לקוד קריא ונקי.
- בכל מקום בו יש צורך בשימוש בקבועים בעלי משמעות יש להגדיר אותם באמצעות define



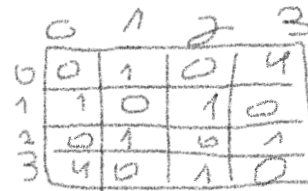
- מטריצת שכנות: היא שיטה לייצוג גרף מכוון/לא מכוון בעל N צמתים (קודקודים) בעזרת מטריצה ריבועית $N \times N$, שערכי תאיה הם 0 או מספר חיובי. תא (i, j) בגרף מתאר את משקלה של הקשת מקודקוד i לקודקוד j בגרף. אם אין קשת כזו, הערך בתא במטריצה יהיה 0.
אם יש קשת כזו, הערך יהיה מספר חיובי והוא ייצג את משקל המעבר מקודקוד i לקודקוד j .
- עליכם לבנות תוכנית המייצגת בעזרת מטריצה גרף לא מכוון ומבצעת את הפעולות הבאות:
(1) התוכנית קולטת מהמשתמש את ערכי המטריצה. (לא דווקא בתחילת התוכנית. יכול להיות מצב בו אתם מקבלים מטריצה מבצעים בדיקות ואז מבלי לצאת מהתוכנית אתם נדרשים לקבל עוד מטריצה)
(2) התוכנית קולטת מהמשתמש את ערכי i ו- j ומדפיסה "True" אם קיים מסלול מ- i ל- j . אחרת, תדפיס "False".

3) התוכנית קולטת מהמשתמש את ערכי i ו- j ומדפיסה את המסלול הקצר ביותר מ- i ל- j , אם לא קיים מסלול התוכנית תדפיס -1.

- יש להניח כי הקלט תקין.
- גודל המטריצה 10×10
- שימו לב יש דוגמאות של קלט-פלט מצורפות למטלה, חובה על הפלט שלכם להיות זהה לפלט שסופק.
- דוגמא:



מסלול הקצר ביותר מקודקוד 0 לקודקוד 2 הוא: 4



מסלול הקצר ביותר מקודקוד 0 לקודקוד 3 הוא: 3

אלגוריתמים העוזרים לכם לפתור בעיה זו:

• Floyd-Warshall_algorithm: מועדף

אלגוריתם פלוייד-וורשאל הוא אלגוריתם במדעי המחשב המשמש למציאת המסלולים הקצרים ביותר בין כל שני צמתים, בגרף ממושקל ומכוון/לא מכוון האלגוריתם מתבסס על האבחנה הבאה: אם ממספרים את קבוצת הצמתים של הגרף כך שהיא מסומנת $\{1, 2, \dots, n\}$ ומסמנים בתור $k^{יא}$ את משקל המסלול הקצר ביותר מהצומת

i אל הצומת j שעובר בצומתי ביניים השייכים אך ורק לקבוצה $\{1, 2, \dots, k\}$ אז מתקיים:

$$(d(i,j))^k = \min\{(d(i,j))^{k-1}, (d(i,k))^k + (d(k,j))^k\}$$

 עת, פעולת האלגוריתם היא זו: הוא מבצע $|V|$ איטרציות, כאשר באיטרציה ה- k הוא מחשב את $(d(i,j))^k$ לכל זוג צמתים i, j השייכים ל- V ושומר את המידע בטבלה. מידע נוסף על האלגוריתם ניתן למצוא בקישורים הבאים:

[הסבר ויקפדיה](#)

[סרטון הסבר](#)

• dykstra algorithm:

[להסבר לחץ כאן](#)

עליכם לכתוב ספריה בשם `my_mat`. אשר תכיל פונקציה עבור כל פעולה. התוכנית הראשית תקרא מהשתמש את הפעולות אחת אחת ותבצע כל פעולה על ידי קריאה לפונקציה המתאימה. ביצוע פעולות מסוימות ידרוש קליטת נתונים נוספים והעברתם לפונקציה המבצעת את הפעולה. על התוכנית הראשית לקרוא פעולות עד שהשתמש יבחר להפסיק פעולה.

המשתמש יקליד כל פעם אחת מאותיות אלו כדי לבצע את הפונקציות הבאות :

- A לבצע את פונקציה מס' 1
- B לבצע את פונקציה מס' 2
- C לבצע את פונקציה מס' 3
- D לצאת מהתוכנית

:Makefile

עליכם לכתוב makefile עם הפקודות:

- **all** יקמפל את כל הקוד שלכם ויצור תוכנית הרצה בשם **connections** אין מגבלה לגבי סוג הלינקוג'
- **clean** ינקה את התיקיה מכל הקבצים פרט לקבצי ההגשה.
- טארגטים נוספים לפי בחירתכם ולנוחיותכם (לא יבדקו)

: הגשה

במודל יש להגיש קובץ `txt` בשם `hw2_ss.txt` הקובץ יכיל 3 שורות. קישור ל `git`-שלכם, מספר `commit` ותעודות זהות המגישים מופרדים ברווח. על ה `git`-שלכם לכלול את הקבצים הבאים ברמה הראשונה:

- Makefile
- `my_mat.h`
- `my_mat.c`
- `main.c`

- יש לוודא שהתוכנית מתקמפלת ורצה על גבי `ubuntu` עם `gcc` אתם רשאים לעבוד עם כל עורך קוד, אך סביבת הבדיקה תהיה `ubuntu`.

בהצלחה!