שפת C – תרגיל 3

<u>תאריך הגשה:</u>

<u>הגשה מאוחרת (בהפחתת 10 נקודות):</u>

<u>הנחיות חשובות:</u>

א. בכל התרגילים יש לעמוד בהנחיות הגשת התרגילים וסגנון כתיבת הקוד. שני המסמכים נמצאים באתר הקורס – הניקוד יכלול גם עמידה בדרישות אלו.

ב. אין להגיש קבצים נוספים על אלו שתדרשו

ג. עליכם לקמפל עם הדגלים -Wextra -Wall -Wvla ולוודא שהתוכנית מתקמפלת ללא אזהרות, **תכנית שמתקמפלת עם אזהרות תגרור הורדה בציון התרגיל.** למשל, בכדי לקמפל קובץ מקור בשם ex2.c יש להריץ את הפקודה:

```
gcc -Wextra -Wall -Wvla  -c ex3.c -o ex3
```

ד. עליכם לוודא שהתרגילים שלכם תקינים ועומדים **בכל דרישות הקימפול והריצה במחשבי בית הספר** מבוססי מעבדי bit-64 (מחשבי האקווריום, לוי, השרת river). חובה להריץ את התרגיל במחשבי בית הספר לפני ההגשה. (ניתן לוודא שהמחשב עליו אתם עובדים הנו בתצורת bit-64 באמצעות הפקודה "uname –a" ווידוא כי הארכיטקטורה היא 64, למשל אם כתוב x86_64.)

ה. לאחר ההגשה, בדקו את הפלט המתקבל בקובץ ה-PDF שנוצר מה presubmission script בזמן ההגשה. באם ישנן שגיאות, תקנו אותן על מנת שלא לאבד נקודות.

ו. בדיקת הקוד לפני ההגשה, גם על ידי קריאתו וגם על ידי כתיבת בדיקות אוטומטיות (tests) עבורו היא אחריותכם

חישבו על מקרי קצה לבדיקת הקוד, בדיקות אוטומטיות יהיו חלק מבדיקת הקוד לציון

חלק מקבצי הבדיקה לדוגמה שמסופקים על ידי צוות הקורס נבדקים על ידי הבדיקה האוטומטית בהגשה (presubmission script). שימוש בקבצי הבדיקה לדוגמה הוא באחריותכם. **במהלך הבדיקה הקוד שלכם ייבדק מול קלטים נוספים לשם מתן הציון.**

משימת תכנות ראשונה – ספריית DFS גנרי (50 נק')

בקובץ ה tar ישנו קובץ בשם genericdfs.h, עליכם לממש את הפונקציה getBest שמוכרזת ומתועדת בקובץ הנ"ל.

- עליכם להוסיף ב makefile מטרה בשם genericdfs.a ליצירת ספרייה בשם זה שתכיל את המימוש לספריה.

- את המימוש שלכם הגישו בקובץ genericdfs.c.

- אין צורך להגיש את genericdfs.h.

משימת תכנות שנייה – פתירת סודוקו (50 נק')

סודוקו הוא תשבץ מספרים שבו צריך למקם ספרות על לוח משובץ שגודלו N×N (כש N מספר שלם כלשהו, ובתרגיל הזה בעל שורש שלם גם כן), המורכב מ-N ריבועים בני N משבצות כל אחד. מטרת המשחק – למקם מספרים על גבי לוח המשחק כך שבכל טור, בכל שורה, ובכל ריבוע, יופיע כל מספר מ 1 עד N בדיוק פעם אחת. (ראו דוגמה לסודוקו של 9X9, הריבועים מודגשים)

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

דוגמה לסודוקו

בתרגיל זה תידרשו לכתוב תוכנה שפותרת סודוקו.

התכנה תקבל כפרמטר קובץ, השורה הראשונה בקובץ מכילה מספר אחד – גודל כל שורה או עמודה בסודוקו. יתר השורות מכילות את שאר המספרים בסודוקו, כאשר מספרים חסרים מופיעים כאפסים, המספרים מופרדים באמצעות רווחים ויש ירידת שורה ('\n') בין השורות.

דוגמה לקובץ קלט אפשרי:

```
4

2 0 3 1

1 0 0 2

0 0 2 0

3 0 0 0
```

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

דוגמה לסודוקו פתור

ניתן להסתכל על פתרון הסודוקו כחיפוש בגרף – כאשר הקודקודים בגרף הם תשבצים מלאים

חלקית, והבנים הישירים של כל קדקוד הם התשבץ כשעליו יש מספר נוסף.

על תכניתכם לפתור את הסודוקו תוך שימוש בספריית ה DFS מהסעיף הראשון, ולהדפיס את הפתרון. (במבנה דומה לקובץ הקלט) אם יש כמה פתרונות, עליכם להדפיס את זה שהשורה העליונה בו מתחילה ברצף המספרים הקטן ביותר (הרצף מתחיל משמאל, אם השורה העליונה זהה יש להסתכל על זו שמתחתיה וכן הלאה).

עבור הדוגמה שהופיעה קודם למשל, על התכנית להדפיס:

4

2 4 3 1

1 3 4 2

4 1 2 3

3 2 1 4

- את הפונקציות שאתם מעבירים כפרמטרים לפונקציה getBest ממשו בקובץ sudukutree.c ו sudukutree.h, את פונקציית ה main ממשו בקובץ sudukusolver.c, ניתן להגיש קבצים נוספים לפי הצורך.

- הרצת התוכנית תיעשה בצורה הבאה:

sudukusolver <filename>

- אם הקובץ לא נמצא על התכנית להדפיס

please supply a file!

usage: sudukusolver <filename>

ולצאת

- במקרה של בעיה בפתיחת הקובץ, על התכנית להדפיס "<filename>: no such file\n" ולצאת.

- אם הקובץ המתקבל במבנה שונה מהאמור לעיל (N לא חיובי או עם שורש לא שלם, מספרים חסרים/שליליים/ גדולים מ N, תווים במקום מספרים) על התוכנית להדפיס: "<filename>:not a valid suduku file\n" ולצאת.

-אם אין פתרון לתשבץ, עליכם להדפיס "no solution!\n" ולצאת.

-אתם רשאים להניח ש N לא גדול מ 100. (פתרון כזה סודוקו לא בהכרח ייקח זמן סביר)

ניהול זיכרון

בתרגיל זה (ובבאים אחריו) תידרשו להשתמש בניהול זיכרון דינמי.

הנכם נדרשים לבדוק שניהול הזיכרון של התכנה שלכם תקין, זאת תעשו ע"י שימוש ב valgrind.

בכדי ש valgrind יוכל לעבוד ולהציג לכם שגיאות יש לקמפל התכנה שלכם עם הדגל g- ולהריץ את התכנה באמצעות הפקודה

valgrind --leak-check=full --show-possibly-lost=yes --show-reachable=yes –undef-value-errors=yes <command>

ולוודא שהתכנה לא מתריעה על שגיאות.

makefile

על קובץ ה makefile שלכם לתמוך בפקודות הבאות:

make genericdfs.a

ליצירת קובץ הספריה עבור המשימה הראשונה.

make sudukuSolver

ליצירת קובץ הרצה עבור המשימה השנייה

make all

ליצירת שני הקבצים.

make clean

למחיקת קבצי ההרצה, הספריות וקבצים נוספים שיוצרו במהלך הבנייה (קבצי o למשל)

הגשה

עליכם להגיש קובץ tar בשם ex3.tar המכיל לפחות את הקבצים הבאים:
makefile sudukusolver.c sudukutree.c sudukutree.h genericdfs.c

ניתן ליצור קובץ tar כדרוש על ידי הפקודה:

tar cvf ex3.tar makefile sudukusolver.c sudukutree.c sudukutree.h genericdfs.c

לפני ההגשה, פתחו את הקובץ ex3.tar בתיקיה נפרדת וודאו שהקבצים מתקמפלים ללא

שגיאות וללא אזהרות.

מומלץ גם להריץ בדיקות אוטומטיות וטסטרים שכתבתם על הקוד אותו אתם עומדים להגיש.

בנוסף, אתם יכולים להריץ בעצמכם בדיקה אוטומטית עבור סגנון קידוד בעזרת הפקודה:

```
~labc/public/codingStyleCheck <file or directory>
```

כאשר <directory or file> מוחלף בשם הקובץ אותו אתם רוצים לבדוק או תיקייה שיבדקו כל הקבצים הנמצאים בה.

דאגו לבדוק לאחר ההגשה את קובץ הפלט (submission.pdf) וודאו שההגשה שלכם עוברת את ה-presubmission script ללא שגיאות או אזהרות.

בהצלחה!