מבוא לתכנות מערכות 10010 סמסטר א' – תשפ"ב

תרגיל בית מס' 1

נושא התרגיל: מטריצות מצביעים

יש להגיש אך ורק דרך תפריט המטלות שבאתר הקורס.

הנחיות הגשה כלליות:

- התרגיל ייבדק בסביבת Ubuntu
- הקוד חייב לעבור קומפילציה, קוד שאינו מתקמפל לא ייבדק.
- **יש להגיש קוד ללא הערות קומפילציה, warnings, קוד בו יהיו הערות יגרור הורדה של 10 נקודות.**
- יש להגיש **רק קבצי c ו h ו Makefile** מקובצים לקובץ אחד **ששמו כשם הסטודנט**. שם פרטי ומשפחה. אין להגיש פרויקט ב eclipse או קבצי obj והרצה
- ב Makefile חייבת להיות אפשרות ל clean
- ניתן לעבוד בזוגות - **במידה ומגישים בזוג קובץ ההגשה יהיה מורכב משמות 2 המגישים ושני המגישים צריכים לעלות את העבודה למודל.**

הוראות כלליות:

1. יש להקפיד על כללי הנדסת התוכנה:
   1. פונקציה לא אמורה להיות ארוכה מ 25 שורות. אם יש קטע קוד שניתן לתת לו כותרת מה הוא מבצע יש לכתוב אותו בפונקציה נפרדת.
   2. יש לחלק את הקוד לתת פונקציות מתאימות. אין לשכפל קוד. קוד זהה אמור להיות בפונקציה מתאימה.
   3. **יש לחלק כל שאלה לקבצי c ו h נפרדים**, יש לכתוב פונקציות כלליות בקובץ נפרד.
   4. מבנה התכנית (הזחות) ותיעוד במידת הצורך .
   5. חובה להשתמש בקבועים במקומות המתאימים .
   6. יש להשתמש בפונקציות קצרות, כלליות, קריאות ושימושיות.
   7. יש להקפיד על בדיקת תקינות קלט, אפשר להניח שאם ביקשו מהמשתמש מספר הוא הכניס מספר אך יתכן ולא בטווח הנכון.
   8. הפלט צריך להיות כפי שניתן בתרגיל במידה וניתן.
   9. קוד קצר, לא מסורבל ויעיל הן מבחינת כתיבתו והן מבחינת ריצת התוכנית.

בתרגיל בית זה יש 3 משימות.

יש לכתוב תכנית שמציגה תפריט המאפשר למשתמש לבחור את המשימה אותה הוא רוצה להריץ וכן תהיה אפשרות לצאת. התפריט הראשי יוצג כל עוד לא בחרו לצאת. כלומר בסיום כל משימה יוצג התפריט הראשי שוב. בחירת E/e תצא הודעה למסך "Bye Bye" ונצא מהתוכנית.

התפריט הראשי

Please choose one of the following options

S/s - Biggest Matrix Sum

C/c – Color Game

B/b - Black and White Game

E/e - Quit

להצגת התפריט יש להשתמש בתבנית switch, יש לדאוג שהמשתמש יוכל להקליד אות קטנה או גדולה.

עבור כל אפשרות בתפריט יש לכתוב פונקציה נפרדת, אין קוד ב switch למעט **קריאה** לפונקציה זו.

1. **אין להשתמש בסוגרים מרובעים [ ]** באף אחת מפונקציות בתרגיל למעט בתחילת הפונקציה הראשית של כל תרגיל שם מגדירים את המטריצה הראשונית.
2. למרות ש Ubuntu מאפשר – אין להגדיר מטריצה עם פרמטר כמו בדוגמא!! רק ע"י קבוע.

```
int x;
int arr[x];
```

3. שים לב: בתתי פונקציות אף פעם אין שימוש בקבועים!!!

משימה 1 - Biggest Matrix Sum:

במשימה זו נחפש תת מטריצה בתוך מטריצה גדולה שסכום אבריה מקסימאלי.

- דוגמא: עבור מטריצה 20X20 אשר מחפשים בה תת מטריצה בגודל 2X2 נמצאה המטריצה המודגשת בצהוב. **שים לב: הדוגמא אומנם מתייחסת למטריצות מרובעות אך הדבר לא מחויב.**

50	50
50	49

23	35	40	-2	1	3	-28	-34	-45	46	-16	35	-21	29	-22	-24	17	-7	-4	13
-13	-20	11	21	-31	43	8	17	-40	0	47	-15	9	-45	47	13	11	22	-47	-36
34	4	-45	-12	-28	-10	36	32	-24	27	-49	50	50	-18	21	27	4	-15	-22	40
-11	-20	43	-39	13	-4	33	25	7	44	24	-24	-4	-13	-4	8	26	2	19	-16
18	-24	42	-30	15	-4	-23	19	-42	-10	-47	-44	-19	47	-45	-19	-32	15	8	20
-30	13	42	21	-35	-25	-21	13	-47	36	-26	6	16	48	24	43	40	-5	36	23
-48	48	-25	3	32	-36	3	-46	50	50	32	-25	33	-47	48	42	21	-37	-42	-46
-4	46	36	20	-36	48	50	4	50	49	14	19	-18	-21	10	-22	-2	-45	-6	15
-18	17	3	42	-24	-37	7	4	-46	-6	12	41	11	-43	17	-2	-26	17	38	38
50	7	-22	40	-20	-21	21	-19	3	46	35	1	-26	-20	23	19	20	28	-25	-3
-17	21	-9	20	28	21	-12	5	38	43	9	-27	25	-49	37	17	22	-21	-46	19
10	-19	-6	16	34	-32	-15	-46	24	-12	5	41	-10	35	21	-44	-13	18	16	-42
-37	14	12	-36	8	23	9	26	-15	0	-6	22	-32	-4	13	-43	21	9	-35	27
-24	-41	42	1	37	-1	34	48	30	0	9	-6	-44	31	10	-37	28	16	46	-9
20	16	-23	8	33	35	-41	-8	-42	11	19	18	45	36	32	-24	37	24	21	36
-46	38	-9	28	40	11	19	50	-20	50	-19	24	3	20	47	9	21	7	-17	-24
49	-17	2	40	2	-29	3	-22	16	40	3	41	43	-46	1	-17	4	6	44	-22
-25	20	5	50	1	-2	-26	40	10	4	-50	-29	18	49	-30	6	-43	-48	-18	-28
-25	15	-27	-22	46	19	17	45	49	43	15	-31	-31	-23	-24	-27	-3	-29	-26	19
-40	17	-20	-13	-29	-49	25	24	-3	-5	48	13	39	14	30	47	44	-45	30	-7

יש לחלק את העבודה לפונקציות כאשר הפונקציה ***doFindMaxSubMatrix*** **חייבת** להיות.

ממש את הפונקציה doFindMaxSubMatrix אשר מחפשת בתוך המטריצה הגדולה תת מטריצה אשר סכום איבריה הוא המקסימלי . **יש לקחת בחשבון רק מקומות שהמטריצה הקטנה נכנסת במלואה במטריצה הגדולה.**

הפרמטרים שפונקציה זו מקבלת הם:

- המטריצה
- מספר שורות
- מספר עמודות
- מספר שורות של תת המטריצה
- מספר עמודות של תת המטריצה

הפונקציה מחזירה:

- את הסכום של תת המטריצה המקסימאלית
- בעזרת מצביע את אינדקס, מתחיל ב 0, השורה העליונה בה נמצאת תת המטריצה (בדוגמא למעלה 6)
- בעזרת מצביע את אינדקס, מתחיל ב 0, העמודה השמאלית בה נמצאת תת המטריצה (בדוגמא למעלה 8)

המשימה מבצעת את הפעולות הבאות תוך חלוקה לפונקציות:

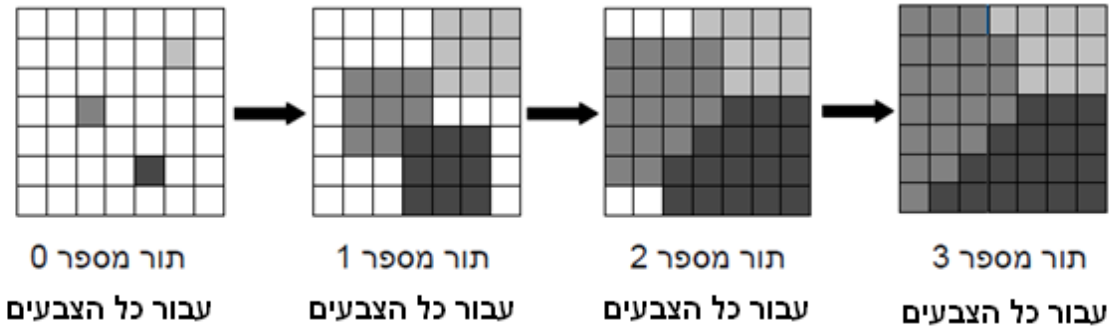
1. הגדר מטריצה בגודל ROWS שורות ו COLS עמודות בחר ערכים כרצונך. *הדפס ערכים אילו למסך.*
2. קלוט מהמשתמש שני מספרים אשר יגדירו את גודל המטריצה האפקטיבי כך שמספר השורות יהיה בין 1 ל ROWS כולל ומספר העמודות יהיה בין 1 ל COLS כולל. *הדפס ערכים אילו למסך.*
   משלב זה אין להשתמש ב סוגריים [ ] ואין להגדיר עוד מטריצה.
3. אתחל את המטריצה במספרים אקראיים, בחר טווח כרצונך, הגדר כמובן בקבועים. *הדפס ערכים אילו למסך.*
4. הדפס את המטריצה
5. קלוט מהמשתמש שני מספרים נוספים אשר יגדירו גודל תת המטריצה *הדפס ערכים אילו למסך.* (שים לב בקלט שגודל השורות והעמודות בתת המטריצה צריך להיות קטן מהשורות והעמודות במטריצה הגדולה).
6. קרא לפונקציה ***doFindMaxSubMatrix***
7. הדפס את תת מטריצת המקסימום, ואת סכומה (התקבל מהפונקציה בסעיף הקודם)
8. במידה ויש יותר מתת מטריצה אחת עם סכום מקסימאלי הדפס את הראשונה שמצאת.

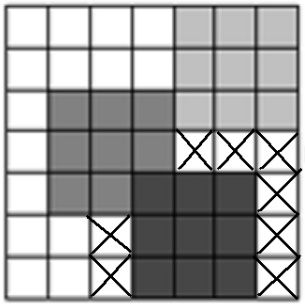
משימה 2 – Color Game:

נתונה מטריצה board [ROWS][COLS] המאותחלת ל 0 .

נתונים COLORS צבעים. כל צבע מקבל מספר. הצבע הראשון מקבל את המספר 1.

כל צבע מתחיל ממשבצת אחת במטריצה, משבצת שהוגרלה מראש. לפי סדר שנקבע.
צבע מתפשט כל פעם מסגרת אחת רחוקה יותר ממשבצת זו במידה והמשבצת לא תפוסה כבר. 0 מציין כי המשבצת לא תפוסה.
סדר התפשטות הצבעים במטריצה כסדר מספרם.
בתור מסוים כל צבע מתפשט לפי הסדר. כלומר במקרה של 3 צבעים, תור הינו התפשטות של שלושת הצבעים.
לדוגמא בציור הנתון:
COLORS = 3 , ערך הצבעים 1 השחור, 2 אפור כהה, 3 אפור בהיר. כלומר סדר ההתפשטות: שחור ראשון אחריו אפור כהה ואחרון אפור בהיר.





התפשטות צבע 1 בתור 2: מילוי כל המשבצות המסומנות ב X במספר 1 (שחור).

במשימה זו יש לצבוע מטריצה ב COLOR צבעים עד שהיא צבועה כולה (אין אף ערך 0 במטריצה)

<u>המשימה מבצעת את הפעולות הבאות תוך חלוקה לפונקציות:</u>

1. הגדר לוח משחק (מטריצה) בגודל ROWS שורות ו COLS עמודות שכל עריכה אפסים. בחר ערכים כרצונך. *הדפס ערכים אילו למסך.*
2. *הגדר קבוע COLOR ותן לו ערך. (שנה את הערך הזה כדי לבדוק את נכונות הקוד שמומש) . הדפס ערך זה למסך.*
3. הדפס את המטריצה
4. הגדר מטריצה בגודל startValues[COLOR][2] אשר תחזיק את נקודות ההתחלה של כל צבע.
5. הגרל נקודת התחלה על לוח המשחק לכל צבע וצבע. וודא שאין 2 צבעים שקיבלו אותה נקודת התחלה. (פונקציית עזר ללא שימוש ב [ ]
6. הדפס נתוני כל הנקודות למסך.

<u>משלב זה אין להשתמש ב סוגריים [ ] ואין להגדיר עוד מטריצה.</u>

7. ממש את הפונקציה *expandColor* אשר מממשת התפשטות של צבע נתון בתור נתון. הפונקציה מחזירה כמה משבצות היא הצליחה למלא בתהליך הצביעה.

<u>הפרמטרים שפונקציה זו מקבלת הם:</u>

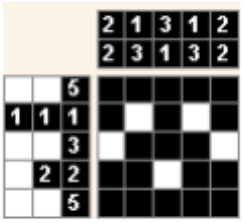
- לוח המשחק, המטריצה, עם הצביעה שנעשתה כבר.
- מספר השורות בלוח המשחק.
- מספר העמודות בלוח המשחק.
- אינדקס שורה של נקודת ההתחלה של הצבע המסוים (מתחיל מ 0).
- אינדקס עמודה של נקודת ההתחלה של הצבע המסוים (מתחיל מ 0).
- מספר הצבע הנתון.
- מספר התור הנתון.

8. ממש את הפונקציה colorTheBoard אשר מקבלת את הנתונים הנדרשים ומפעילה את הפונקציה *expandColor* לפי הסדר עבור כל צבע עד שלוח המשחק מלא. הפונקציה מדפיסה אחרי כל תור את לוח המשחק.

משימה 3 – Black and White Game:

שחור ופתור הוא חידה שבה יש לגלות ציור חבוי בטבלת משבצות. בראש כל שורה ועמודה מופיעה סדרה של מספרים המייצגים רצפים של משבצות שחורות, בסדר נתון. לדוגמא, הסדרה [2,3] מייצגת רצף של שתי משבצות המופיע לפני רצף של שלוש משבצות. המרחק של הרצף הראשון מקצה התמונה אינו ידוע. המרחק בין הרצפים גדול מ-1 (לפחות תא לבן אחד בינהם).

דוגמא לחידת שחור ופתור (ופתרונה):



במשימה זו נבדוק אם **השורות** בלוח שחור ופתור נתון הינו תקין.

נצייג את הקלט כשני מערכים דו ממדיים אחד עבור לוח המשחק בו ערכו של תא לבן הוא 0 וערכו של תא שחור הוא 1, והשני עבור נתוני השורות. נתוני השורות מכילים את סדרות הרצפים.

המשימה מבצעת את הפעולות הבאות תוך חלוקה לפונקציות:

1. *הגדר לוח משחק (מטריצה) בגודל ROWS שורות ו COLS ואתחל בערכים.*
   לדוגמא עבור ציור למעלה:

```
#define ROWS 5
#define COLS 5
```

```
int mat[ROWS][COLS] = { {1,1,1,1,1},{1,0,1,0,1},{0,1,1,1,0},
                        {1,1,0,1,1}, {1,1,1,1,1} };
```

2. *הגדר מטריצה המייצגת את הערכים משמאל. לדוגמא עבור הציור למעלה:*

```
int leftNums[ROWS][(COLS+1)/2] = { {0,0,5},{1,1,1},{0,0,3},{0,2,2},{0,0,5} };
```

3. *הדפס את לוח המשחק ואת ערכי הנתונים משמאל.*
4. *ממש את הפונקציה checkboard אשר בודקת את חוקיות השורות בלוח שחור ופתור אשר הגדרת.*
5. *הפעל את הפונקציה והדפס את תוצאת הבדיקה*

יש לבדוק את כל מצבי הקצה, לדוגמא:

1. *שורה כולה לבנה יכולה להיות תקינה אם המספרים מצד שמאל כולם אפסים*
2. *המספרים מצד שמאל יכולים להיות כולם אפסים אך יש שחור בשורה*
3. *מספרים משמאל שחורגים מאורך השורה {1,5,1}*

שנה את הערכים שבדוגמא לבדיקות מקיפות של הקוד!!!