מבוא למדעי המחשב מ'/ח' (234114/7), סמסטר אביב 2022

תרגיל בית 3

מועד אחרון להגשה:

המתרגל האחראי על תרגיל זה: אדיר רחמים

שאלות לגבי התרגיל יש לפרסם ב-piazza

טרם הפנייה, בדקו אם השאלה שלכם כבר נענתה.

E-mail: adir.rahamim@campus.technion.ac.il לעניינים מנהלתיים בלבד.

- **על נושא המייל להתחיל במספר הקורס (234114/234117) והתרגיל ולהמשיך בנושא השאלה שתופיע בגוף המייל**

בתרגיל זה מותר להשתמש בפונקציות שנלמדו מהספרייה `stdio.h,stdbool.h` **בלבד**. כמו כן, **החומר המותר לתרגיל הוא עד תרגול 7 כולל.**

הנחיות כלליות:

- הגשה ב**בודדים**. עליכם לכתוב את הפתרונות לבד ולהגיש ביחידים.
- קראו את השאלות בעיון לפני שתתחילו בפתרונן.
- הקפידו לתעד את הקוד שלכם בהערות באנגלית.
- שאלות ותשובות נפוצות בנוגע לתרגיל יתפרסמו באתר כל כמה זמן תחת סעיף F.A.Q- **חובה להיכנס ולהתעדכן! כל דגש שמפורסם שם הוא מחייב!**
- מלבד מילואים, לא יתקבלו תרגילים אחרי מועד הגשה. הגשה באיחור לאחר מועד הגשה נחשבת כאי-הגשה.
- כל יום מילואים = יום דחייה. על מנת לקבל את הדחייה, עליכם לשלוח באי-מייל, עותק של האישור המראה שהייתם במילואים (טופס 3010). אם האישור יגיע אליכם בתאריך מאוחר, יש להודיע על כך למתרגל האחראי לפני תאריך הגשת התרגיל.
- ערעורים ניתן להגיש עד שבוע לאחר קבלת הציון.
- **לא ניתן לערער על תוצאות הבדיקה האוטומטית.**
- **שימו לב! הבדיקה הינה בחלקה אוטומטית, ולכן הקפידו להדפיס בדיוק בפורמט שהתבקשתם ובידקו עם DiffMerge את הפלט שלכם מול הפלט של הדוגמאות שקיבלתם.**
- השתמשו באתר הבדיקה העצמית.
- ההגשה הינה אלקטרונית ו**בבודדים** דרך אתר הקורס. קובץ ההגשה יהיה מסוג **zip** (ולא אף פורמט אחר) ויכיל בתוכו את הקבצים הבאים בלבד, ללא כל תיקיות:
  - קובץ **students.txt** עם מספר תעודת הזהות שלך וכתובת האי-מייל שלך.
  - קובץ פתרון **hw3q1.c**.
  - **חובה לשמור את אישור ההגשה (ולא רק את קוד האישור!!) שמקבלים מהמערכת לאחר שמגישים, עד לסיום הקורס.**
  - יש להקפיד להגיש את כל הקבצים בדיוק עם השמות שמופיעים לעיל. הגשה שלא תעמוד בתנאי זה **לא תתקבל ע"י המערכת!** אם המערכת לא מקבלת את התרגיל שלכם, חפשו את הפתרון לבעיה באתר הקורס תחת הכפתור FAQ.

הנחיות לתכנון וכתיבת קוד: חשוב לקרוא לפני התרגיל(!)

בתרגיל זה המטרה היא לתרגל אתכם בפירוק בעיה לגורמים קלים לתכנות (ותכנון). לצורך כך, ישנן כמה מגבלות על כתיבת הקוד. **שימו לב, התוכנית תיבדק באופן ידני, ויורדו נקודות על חריגה מהכללים.**
בנוסף, הדגש בתרגיל זה הוא על תכנון נכון של הקוד, ובהתאם לכך יינתן לבדיקה הידנית משקל גבוה יותר מהרגיל: 67% מהציון.

1. אורך כל פונקציה לא יעלה על 16 שורות קוד (ראו הגדרות מדויקות בהמשך הדף). חשוב: עמידה בתנאי זה דורשת תכנון מוקדם של הקוד!
   a. בונוס 5 נקודות אם הקוד יעמוד בכל הדרישות, ואורך הפונקציה המקסימלית =< 13.
2. רוחב כל שורה (כולל הערות והזחות) לא יעלה על 75 תווים. ניתן לבדוק אורך של שורה בקודבלוקס (ראו תמונה מצורפת עם התרגיל). אם השורות ארוכות הן יגלשו בהדפסה, מה שיקשה על בדיקת התרגיל שלכם (ויגרור הורדת ניקוד).
3. לפני כל פונקציה, יש לכתוב בהערה (בקצרה) מה הפונקציה עושה
   a. ההערה צריכה להסביר מה הפונקציה עושה ולא כיצד היא עושה זאת. לדוגמה:
   "הפונקציה מכניסה את x ו y למערך moves" אינו הסבר, לעומת "הפונקציה שומרת את המהלך האחרון במערך moves" הוא כן הסבר.
   או: "הפונקציה מקדמת את num אם isMove חיובי" אינו הסבר, לעומת "אם בוצע מהלך חוקי, הפונקציה מקדמת את מספר המהלכים החוקיים" (או בקצרה, "הפונקציה סופרת את מספר המהלכים החוקיים").
   b. ההערה צריכה להיות באנגלית. לצערנו עברית מודפסת כג"בריש.
   c. ההערה צריכה להופיע לפני המימוש של הפונקציה ולא ההצהרה שלה.
4. חובה לתת שמות משמעותיים לפונקציות.
   a. השם צריך לשקף את מה שכתבתם בהערה (אך בתמצות).

בנוסף, יש להקפיד על הכללים הרגילים לגבי כתיבת קוד נכון:

5. חובה לתת שמות משמעותיים למשתנים שאתם מגדירים.
6. חובה להשתמש בהזחות תקינות כפי שנלמדו בתרגולים.
7. חובה להשתמש ב define במקומות המתאימים, כפי שנלמדו בתרגולים.
8. אסור להשתמש במשתנים גלובאליים או סטאטיים.
9. אסור לשכפל קוד שלא לצורך

הגדרות לגבי אורך פונקציה

1. מגבלת השורות תקפה לגבי כל פונקציה בקוד, **כולל ה-main**. יש לרשום את האורך בהערה לפני הפונקציה כדי להקל על הבדיקה.
2. אסור לכתוב 2 פקודות באותה שורה (למשל ;x=0; counter++). לעומת זאת, אפשר להגדיר כמה משתנים באותה שורה ולאתחל אותם.
   מקרה יוצא דופן הוא הפקודות case ו – break בבלוק של switch בלבד, אותן ניתן לכתוב באותה שורה.
3. שורה המכילה הערות בלבד, כותרת הפונקציה ,סוגר '{', '}' אחד בלבד, או מילת else בלבד אינה נחשבת לספירת אורך הפונקציה.
4. כל שורה אחרת נחשבת לאורך הפונקציה, כולל: הגדרות משתנים, פקודת if, פקודת return וכל פקודה אחרת שאינה מופיעה ב 3.
5. שורה ארוכה שנשברה לשתיים (כדי שרוחב השורה יהיה קטן מ 75 תווים) נחשבת כ 2 שורות.

צוללות (Battleships)

בתרגיל זה נממש את המשחק צוללות.
בקישור הבא ניתן לראות המחשה:
https://he.wikipedia.org/wiki/%D7%A6%D7%95%D7%9C%D7%9C%D7%95%D7%AA_(%D7%9E%D7%A9%D7%97%D7%A7)

החוקים:

במשחק שני שחקנים (במקרה של התרגיל זה מחשב מול המשתמש). לכל משתמש שני לוחות. לוח אחד למיקום הצוללות שלו ולוח שני למעקב אחר צוללות היריב. כל לוח מורכב מ – 10 עמודות ו –10 שורות.

על כל משתמש למקם בתחילת המשחק 4 צוללות (הערה: מספר זה יכול להשתנות! שימו לב שאתם כותבים קוד כך שתוכלו לעדכן שינויים בקלות). במשחק ישנם 4 צוללות לכל שחקן והצוללות הינן בגודל: 3,3,4,5.

כל שחקן בתורו, בוחר באיזה מקום לתקוף את לוח היריב. לוח המעקב אחר היריב יתמלא באינפורמציה הנכונה – אם נמצאה שם צוללת אז יסומן V אחרת יסומן X. צוללת טבעה אם תקפו את כולה.
המשחק מסתיים כששחקן הטביע את כל צוללות היריב.

בתרגיל זה, המשחק יהיה בין מחשב למשתמש. על מנת שהמחשב יבחר ערכי אתחול ותקיפה (פרטים נוספים בהמשך) תצטרכו להשתמש ברנדומליות:

רנדומליות:
על מנת להגריל ערכים "אקראיים" עליכם להשתמש בפונקציות rand_range, srand_range המסופקות לכם בקובץ rand.c.
את פונ' אלו יש להעתיק בשלמותן לקובץ c שהנכם מגישים.

בתחילת התכנית יש לקרוא פעם אחת לפונקציה srand_range עם ה – seed שהמשתמש הזין.
לאחר מכן, כדי להגריל מספר בטווח low עד high (כולל) יש לקרוא לפונקציה rand_range.

הערה: ב-C קיימות פונ' דומות: rand() ו – srand() (שמוגדרות ב – stdlib). מאחר והמחשב לא באמת יכול להמציא ערכים רנדומליים, הוא בוחר את המספר שהוא מחזיר ע"י חישוב מסובך. הערך ההתחלתי של החישוב מכונה seed. אם הערך ההתחלתי זהה בשתי הרצות שונות הערכים ה"אקראיים" יהיו זהים בשתי ההרצות. לא נשתמש בפונ' srand(), rand() מאחר ועבור הרצות במערכות הפעלה שונות לא יוחזרו אותם ערכים "רנדומליים".

שימו לב שבמימוש הפונ' rand_range השתמשנו במשתנה סטטי. אסור לכם להשתמש במשתנים סטטיים למעט מקרה זה.

מהלך המשחק

בתחילת המשחק תוצג הודעת פתיחה:

```
Welcome to Battleship!
```

התוכנית תבקש מהמשתמש seed:

```
Please enter seed:
```

המשתמש יזין seed כקלט עבור srand_range כפי שהוסבר לעיל.
לאחר מכן המשתמש יכניס רמת קושי – כל מספר טבעי גדול (ממש) מ-0. **יפורט בהמשך.**

```
Please enter level:
```

ניתן להניח את נכונות הקלט עבור משתנים אלו.

אתחול:

המשתמש והמחשב צריכים לבחור מקומות עבור הצוללות שלהם. מיקום צוללת בגודל מסוים מורכב מבחירת **מיקום התחלתי** – כלומר בחירת 2 אינדקסים לתא המתאים בלוח, **ובחירת כיוון**.
בחירת הכיוון תיוצג ע"י בחירת שני מספרים כל אחד בתחום 1,0,1- (אך לא שניהם 0), ותיקבע בדומה לכיוון וקטור במערכת צירים דו ממדית, כאשר המספר הראשון הוא הכיוון בציר ה-X, והשני הכיוון בציר ה-Y. לדוגמא עבור בחירת (1,1-) הכיוון יהיה אלכסון שמאלה-מעלה. עבור בחירת (0,1) הכיוון יהיה מעלה.

אתחול עבור המחשב:

המחשב יבחר את מקומות הצוללות האלו לפי סדר עולה: כלומר קודם את הצוללת בגודל 2, לאחר מכן בגודל 3 וכו'..
עבור כל צוללת התוכנית תבחר ערכים רנדומליים למיקום הצוללות, ולאחר מכן ערכים רנדומליים לכיוון הצוללת, כפי שהוסבר לעיל.
בחירת הערכים הרנדומליים תהיה בתחום הרצוי. כלומר עבור מיקום צוללות ייבחרו שני ערכים, x,y רנדומליים, כל אחד בתחום 0,...,9.
עבור כיוון הצוללת צריך לבחור 2 מספרים, dirX,dirY , כל אחד בין 1- ל-1.
במקרה של שגיאה כלשהי בבחירת ערכי הצוללת – **יפורט בהמשך** - יש לבחור את ערכי הצוללת מהתחלה.
שימו לב שאתם בוחרים את הערכים בדיוק כפי שמתואר. בחירה בסדר שגוי או בתחום לא נכון תכשיל את הטסטים ותגרום להורדת ציון משמעותית.

אתחול עבור המשתמש:

התוכנית תבקש מהמשתמש קלט עבור מיקום הצוללות בסדר עולה. כלומר, בתחילה תבקש קלט עבור צוללת בגודל 3 לאחר מכן 3 וכו'.. הקלט יהיה מהצורה: (dirX,dirY) (x,y) כאשר:
x,y – זוג קואורדינאטות המייצג מיקום התחלתי.
dirX,dirY – כיוון הצוללת פי שהוסבר לעיל.
לפני בקשת קלט עבור כל צוללת יודפס לוח המשחק של המשתמש ולאחר מכן תודפס ההודעה:

```
Enter location for Battleship of size *:
```

כאשר * הינו גודל הצוללת הרצוי.

במקרה של שגיאה כלשהי בבחירת ערכי הצוללת – יפורט בהמשך - יש להדפיס את ההודעה:

```
Error: Incorrect parameters! Please enter location for the Battleship size *
again:
```

ולבחור את ערכי הצוללת מהתחלה.

התוכנית תבצע את כל הנ"ל עד לקליטת כל הצוללות הרצויות.

שגיאות בבחירת ערכי הצוללת:

- מיקום הצוללת חורג מגבולות המשחק: צריך לבדוק לא רק את x,y. ייתכן ולאחר בחירת הכיוון הצוללת תחרוג מגבולות המשחק.
- לפחות אחד מ- dirX, dirY לא בין 1- ל- 1.
- אם dirX, dirY שניהם 0.
- אם הצוללת שנבחרה חופפת לצוללת אחרת שכבר מוקמה, כלומר קיים תא ששייך לצוללת שמוקמה בעבר, והצוללת הנוכחית כוללת גם את תא זה. שימו לב שהמקרה הבא תקין:

```
9 |
8 |
7 |       *     *
6 |         * *
5 |         * *
4 |       *     *
3 |               *
2 |
1 |
0 |
    - - - - - - - - - -
    0 1 2 3 4 5 6 7 8 9
```

שימו לב שלא כל השגיאות הנ"ל רלוונטיות לאתחול לוח המחשב.

דוגמת הרצה לקליטת צוללת:

(מפאת חיסכון במקום, דוגמאות ההרצה הן עבור לוח בגודל 5*5 וצוללות בגדלים 2,3,3)

```
Welcome to Battleship!
Please enter seed:
5
Please enter level:
2
4 |
3 |
2 |
1 |
0 |
    - - - - -
    0 1 2 3 4
Enter location for Battleship of size 2:
0,0 1,2
Error: Incorrect parameters! Please enter location for the Battleship size 2 again:
0,4 1,0
4 | * *
3 |
2 |
1 |
0 |
    - - - - -
    0 1 2 3 4
Enter location for Battleship of size 3:
0,2 1,1
4 | * * *
3 |   *
2 | *
1 |
0 |
    - - - - -
    0 1 2 3 4
Enter location for Battleship of size 3:
4,0 1,0
Error: Incorrect parameters! Please enter location for the Battleship size 3 again:
2,0 1,1
All battleships have been located successfully!
```

ביצוע תקיפות:

בעוד שהמחשב פחות חכם מכם, הוא מסוגל לירות הרבה יותר מהר! בכל תור המחשב מצליח לתקוף כמה פעמים את צוללותיכם! מספר הפעמים שהמחשב יכול לתקוף אתכם בכל תור הינו כרמת הקושי שנקלטה בתחילת המשחק.

המשחק בתורות כאשר השחקן מתחיל ראשון.

<u>ביצוע תור המשתמש</u>

בתחילת כל תור של השחקן יודפס מצב המשחק – לוח המעקב של השחקן (בפרט בתחילת המשחק יוצג הלוח "הריק") ולאחר מכן לוח המעקב של המחשב עם מיקום צוללות השחקן.
לוח ריק יהיה למעשה מלא ע"י התו ' ' (רווח).
תא בלוח מעקב שהתגלתה בו צוללת יסומן ע"י 'V' ואילו תא שהתגלה שאין בו צוללת יסומן ע"י 'X'.
חלקי צוללות השחקן שמוצגים על גבי לוח המעקב של המחשב ועדיין לא הותקפו יסומנו ע"י '*'.
מתחת לכל עמודה ומצד שמאל של כל שורה יוצג מספר העמודה/שורה 0-9.

מתחת להדפסת מצב המשחק, תוצג הודעה לשחקן הנוכחי, שתבקש ממנו להזין את המיקום בלוח היריב שבו הוא רוצה לתקוף.

```
Your following table:
4 |   V
3 |
2 |
1 |
0 |
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | * * *
3 |   *
2 | *     X *
1 |       *
0 |   X *
    - - - - -
    0 1 2 3 4
It's your turn!
Enter coordinates for attack:
```

השחקן יכניס את המיקום שברצונו לתקוף בצורה הבאה: x,y

במקרה של שגיאה כלשהי בבחירת מיקום התקיפה – יפורט בהמשך - יש להדפיס את ההודעה:

```
Error: Incorrect parameters! Please enter coordinates for attack again:
```

והתוכנית תבקש קלט מחדש עבור תור זה. הודעה זו תימשך עד לקבלת קלט תקין.

<u>ביצוע תור המחשב</u>

לאחר שהשחקן יבצע את תורו באופן תקין, המחשב יבצע את תורו בצורה רנדומלית, ולאחר מכן יודפס מצב המשחק החדש.
ביצוע מהלכי המחשב יהיה באופן דומה לאתחול לוח המחשב: המחשב יגריל בצורה רנדומלית 2 מספרים בתחום 0-9 אשר יהוו זוג קואורדינאטות עד לקבלת זוג קואורדינאטות תקין.
קואורדינאטות תקינות הן קואורדינאטות שלא התבצעה מהן אף אחת מהשגיאות המופיעות להלן (למעשה, השגיאה האפשרית היחידה במהלך של המחשב היא תקיפה במקום שכבר הותקף בעבר. שימו לב שלא תודפס הודעת שגיאה במידה וזהו מהלכו של המחשב).
המחשב יגריל מספר זוגות תקינים x,y לפי רמת המשחק כפי שנקלטה בתחילת התוכנית.

שגיאות אפשריות בבחירת מיקום תקיפה–

- הזנת ערך החורג מגבולות המשחק
- תקיפה במקום שכבר הותקף בעבר

התוכנית תודיע במידה וצוללת טבעה: לאחר כל תור (של המשתמש או של המחשב), התוכנית תדפיס את ההודעה הבאה במידה וצוללת כלשהי הותקפה בשלמותה.

- אם הצוללת של המשתמש טבעה:

```
Your battleship of size * has been drowned!
```

- אם הצוללת של המחשב טבעה:

```
The computer’s battleship of size * has been drowned!
```

כאשר * הינו גודל הצוללת שטבעה.

אם בתור של המחשב הוטבעו כמה צוללות, יש להדפיס את ההודעות לפי סדר ההטבעות.

יחס לקלט:

- ניתן להניח שתמיד יתקבל "מספיק קלט", כלומר כל עוד המשחק לא נגמר לא נגיע לסוף הקלט (EOF).
- לא ניתן להניח את נכונות הקלט מבחינה סינטקטית. לדוגמא לא מובטח שעבור בקשה לקלט מהצורה i,j הקלט אכן יהיה בצורה הזו (למשל מספרים שלא מופרדים בעזרת פסיק, או תווים שאינם מספרים או פסיק). במקרים כאלו תודפס השגיאה:

```
Error: Invalid input!
```

והתוכנית תסתיים (שימו לב שאסור להשתמש בפונקציה exit מאחר ובאופן כללי לא רצוי להשתמש בפונקציה זו וגם היא נמצאת בספריה stdlib שאסורה לשימוש).

מקרה הרצה לדוגמה: (המשך של דוגמת האתחול)

```
Your following table:
4 |
3 |
2 |
1 |
0 |
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | * * *
3 |   *
2 | *       *
1 |       *
0 |     *
    - - - - -
    0 1 2 3 4
It's your turn!
Enter coordinates for attack:
1,4
Your following table:
4 |   V
3 |
2 |
1 |
0 |
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | * * *
3 |   *
2 | *     X *
1 |       *
0 |   X *
    - - - - -
    0 1 2 3 4
It's your turn!
Enter coordinates for attack:
0,4
Your following table:
4 | X V
3 |
2 |
1 |
0 |
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | V * *
3 |   *     X
2 | *     X *
1 |       *
0 |   X *
    - - - - -
    0 1 2 3 4
It's your turn!
```

```
Enter coordinates for attack:
2,4
Your following table:
4 | X V X
3 |
2 |
1 |
0 |
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | V * *
3 |   *     X
2 | *   X X *
1 |       V
0 |   X *
    - - - - -
    0 1 2 3 4
It's your turn!
Enter coordinates for attack:
2,3
The computer's battleship of size 2 has been drowned!
Your following table:
4 | X V X
3 |     V
2 |
1 |
0 |
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | V * V
3 |   *     X
2 | *   X X *
1 |       V
0 |   X *   X
    - - - - -
    0 1 2 3 4
It's your turn!
Enter coordinates for attack:
3,1
Your following table:
4 | X V X
3 |     V
2 |
1 |       V
0 |
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | V * V
3 | X *     X
2 | *   X X *
1 |   X   V
0 |   X *   X
    - - - - -
    0 1 2 3 4
It's your turn!
Enter coordinates for attack:
4,1
```

```
Your battleship of size 2 has been drowned!
Your following table:
4 | X V X
3 |     V
2 |
1 |       V V
0 |
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | V V V
3 | X *     X
2 | *   X X *
1 |   X X V
0 |   X *   X
    - - - - -
    0 1 2 3 4
It's your turn!
Enter coordinates for attack:
2,1
The computer's battleship of size 3 has been drowned!
Your following table:
4 | X V X
3 |     V
2 |
1 |     V V V
0 |
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | V V V
3 | X * X   X
2 | *   X X *
1 | X X X V
0 |   X *   X
    - - - - -
    0 1 2 3 4
It's your turn!
Enter coordinates for attack:
0,0
Your following table:
4 | X V X
3 |     V
2 |
1 |     V V V
0 | X
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | V V V X
3 | X * X   X
2 | *   X X *
1 | X X X V X
0 |   X *   X
    - - - - -
    0 1 2 3 4
It's your turn!
Enter coordinates for attack:
3,4
Your following table:
```

```
4 | X V X X
3 |     V
2 |
1 |     V V V
0 | X
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | V V V X
3 | X * X X X
2 | * X X X *
1 | X X X V X
0 |   X *   X
    - - - - -
    0 1 2 3 4
It's your turn!
Enter coordinates for attack:
3,1
Error: Incorrect parameters! Please enter coordinates for attack again:
1,0
Your following table:
4 | X V X X
3 |     V
2 |
1 |     V V V
0 | X V
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | V V V X
3 | X * X X X
2 | * X X X *
1 | X X X V X
0 | X X V   X
    - - - - -
    0 1 2 3 4
It's your turn!
Enter coordinates for attack:
2,0
Your battleship of size 3 has been drowned!
Your following table:
4 | X V X X
3 |     V
2 |
1 |     V V V
0 | X V V
    - - - - -
    0 1 2 3 4
The computer's following table:
4 | V V V X
3 | X V X X X
2 | V X X X *
1 | X X X V X
0 | X X V   X
    - - - - -
    0 1 2 3 4
It's your turn!
Enter coordinates for attack:
3,0
The computer's battleship of size 3 has been drowned!
Congrats! You are the winner!
```