מבוא למדעי המחשב מ'/ח' (234114/7), סמסטר אביב 2019

תרגיל בית 3

מועד אחרון להגשה: **יום רביעי 18/12/2019 עד שעה 23:55**

המתרגל האחראי על תרגיל זה: **עמית ברכה**

משרד: טאוב 400

E-mail: amit.bracha@cs.technion.ac.il

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

234114 hw3 Question on handling scanf errors

- **טרם הפנייה, בדקו בדף ה-FAQ של התרגיל אם השאלה שלכם כבר נענתה.**

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 שלכם תעבור בדיקת style, ויורדו נקודות על חריגה מהכללים. בנוסף, הדגש בתרגיל זה הוא על תכנון נכון של הקוד, ובהתאם לכך יינתן לבדיקת ה style משקל רב : 50% מהציון.**

- אורך כל פונקציה לא יעלה על 16 שורות קוד (ראו הגדרות מדויקות בהמשך הדף). הגבלה זו תקפה לכל הפונקציות, כולל main. יש **לרשום את האורך בהערה** לפני הפונקציה כדי להקל על הבדיקה. **קראו מסמך נלווה** שמפרט איך אנו סופרים שורות קוד.
  a. בונוס 10 נקודות אם הקוד יעמוד בדרישה לאורך הפונקציה המקסימלית =< 13.
- רוחב כל שורה (**כולל הערות והזחות**) לא יעלה על 150 תווים. ניתן לראות מהו אורך של שורה בקודבלוקס. אם השורות ארוכות הן תגלושנה בהדפסה, מה שיקשה על בדיקת התרגיל שלכם (ויגרור הורדת ניקוד).
  a. דהיינו, שורות ארוכות יש לשבור ידנית (ורצוי להימנע מלכתוב)
- לפני כל פונקציה, כדאי לכתוב בהערה (בקצרה) מה הפונקציה עושה
  b. ההערה צריכה להסביר מה הפונקציה עושה ולא כיצד היא עושה זאת. לדוגמה: "הפונקציה מקדמת את num אם isMove חיובי" אינו הסבר, לעומת "אם בוצע מהלך חוקי, הפונקציה מקדמת את מספר המהלכים החוקיים".
  c. כלל אצבע הוא שצריך להיות ברור מה משמעות משתני הקלט, מה הערך המוחזר והקשר ביניהם.
  d. התיעוד, אם בכלל, צריך להופיע לפני המימוש של כל פונקציה.
  e. ההערה צריכה להיות באנגלית. לצערנו עברית מודפסת כג"יבריש.
- חובה לתת שמות משמעותיים לפונקציות ולמשתנים.
  a. השם צריך לשקף את פעולת הפונקציה או את מטרת המשתנה.
  b. על השמות להיות באנגלית (לדוגמא לא tavla אלא table, לא luah אלא board).
- חובה להשתמש בהזחות תקינות כפי שנלמדו בתרגולים.
- חובה להשתמש ב define להגדרת קבועים בעלי משמעות (מספרים קבועים או תווים קבועים עם מטרה מוגדרת).
  a. שם הקבוע צריך להיות לפי המטרה שלו, ולא לפי התוכן. הימנעו מהגדרות כגון: #define ZERO 0, כיוון שהגדרה כזו לא מוסיפה מידע בקוד ומיותרת.
  b. על שמות קבועי #define (ורק הם) תמיד להופיע באותיות גדולות (כמו בהרצאות ובתרגולים). זוהי מוסכמה מקובלת שמטרתה להבדילם ממזהים (כמו משתנים ושמות פונקציות).
- אסור להשתמש במשתנים גלובאליים או סטאטיים
- לא כדאי לשכפל קוד שלא לצורך, למשל לכתוב שתי פונקציות שעושות פעולה דומה רק עם קבועים שונים.

בקרת עמיתים:

בקורס מבוא למדעי המחשב תלמדו את הבסיס לתכנות. בסיס זה ישמש אתכם בהמשך התואר, וכן בעבודה בחברות הייטק. בעוד שבקורסים סטנדרטים במדעי המחשב המיקוד הוא על כתיבת קוד, בתעשייה תדרשו בעיקר "לתקן" קוד קיים – להוסיף יכולות או לתקן באגים קיימים. מעבר ליכולת לכתוב קוד, נדרש גם (ובעיקר) לקרוא קוד קיים, ולהבין אותו לעומק. בנוסף, מכיוון שמתכנתים מתעסקים בעיקר בקריאת קוד קיים, הכתיבה של קוד קריא וברור הפכה ליכולת חשובה מאוד.

על תרגיל זה תופעל סדנה של בקרת עמיתים. בסדנה זו כל סטודנט יתבקש להעריך קוד של 2 סטודנטים, והקוד שהגיש יוערך ע"י 2 סטודנטים אחרים. הסטודנטים יספקו לעמיתיהם משוב וציון על קוד שקראו, ויקבלו ציון ומשוב מהסטודנטים שהעריכו את הקוד שלהם. חלק מציון המטלה יורכב מממוצע הציונים אשר יינתנו ע"י סטודנטים עמיתים. הציון על התרגיל יורכב מהציון שיינתן על הקוד ומציון על איכות ההערכה שהסטודנט נתן. ציון זה יהיה עד 2% מהציון הסופי בקורס.

הגשה: בנוסף להגשת התרגיל דרך אתר הקורס (באופן הרגיל) יש להגיש את התרגיל גם דרך מודל:
234114 on Moodle

למודל יש להגיש רק את הקובץ hw3q1.c. הקובץ חייב להיות אנונימי ולא להכיל את שמכם או פרט מזהה אחר. הסיסמא הראשונית למודל היא הת"ז שלכם.

הערכה: ההערכה תתחיל לאחר סיום תאריך ההגשה של התרגיל. ניתן למצוא טופס הערכה לדוגמה ביחד עם תרגיל זה.

לכל שאלה בקשר לבקרת עמיתים אתם מוזמנים לפנות לדמיטרי רבינוביץ' באימייל dmitry.ra@cs.technion.ac.il או בשעת קבלה.

AVOIDANCE TIC-TAC-TOE

- בתרגיל זה נכתוב גרסה של המשחק "איקס עיגול" עבור 2 שחקנים.
- בגרסה שלנו גודל הלוח n*n נבחר על ידי המשתמש בתחילת הריצה והוא יכול להיות לכל היותר N*N כאשר אצלינו N=11 (כלומר 0<n<N=11).
- מטרת המשחק הפוכה מ"איקס עיגול" – השחקן הראשון שיוצר רצף של n "איקסים" או "עיגולים" בשורה, עמודה או באלכסון הוא המפסיד ובמקרה זה השחקן השני הוא המנצח.
  התוכנית תריץ את המשחק, ובסופו תכריז על הזוכה מבין שני השחקנים.
- התוכנית תאפשר למשתמש לבצע undo כדי לנסות מהלכים שונים.
- **פלט התוכנית יתבצע רק על ידי פונקציות ההדפסה הנתונות בקובץ hw3q1_print_functions.c שסופק יחד עם התרגיל –** בקובץ זה מעל כל פונקציה יש הסבר למה היא עושה ומהו אורכה בשורות (כפי שהנכם צריכים לעשות לכל פונקציה שאתם מגדירים וממשים).

חוקי המשחק:
בתחילת המשחק הלוח כולו ריק (תא ריק יסומן ב'_') . בתחילת המשחק שחקן 1 בוחר תא בלוח (ע"י מתן אינדקס שורה ואינדקס עמודה כאשר האינדקסי מתחילים מ 1) ותא זה יסומן ב-X, לאחר מכן מגיע תור שחקן 2 שבוחר גם הוא תא בלוח (אם התא הנבחר לא ריק, יש לבחור תא אחר) ותא זה יסומן ב-O, לאחר מכן מגיע שוב תור שחקן 1 וכן הלאה, עד שאחד השחקנים מפסיד ואז השני הוא המנצח או שהלוח כולו מלא בלי שאף אחד הפסיד או ניצח (תיקו).
שוב, המפסיד הוא הראשון שמשיג רצף של n סימנים זהים באותה שורה, עמודה, אלכסון ראשי או אלכסון משני ולכן השחקן השני הוא המנצח.

דומגאות:

1] עבור הלוח הבא בגודל 3x3 (n=3):

```
|O|O|O|
|_|X|_|
|X|_|X|
```

שחקן 1 הוא המנצח שכן שחקן 2 הפסיד כיוון שעשה רצף של n "עיגולים" בשורה 1.

2] עבור הלוח הבא בגודל 2x2 (n=2):

```
|O|X|
|_|X|
```

שחקן 2 הוא המנצח שכן שחקן 1 הפסיד כיוון שעשה רצף של n "איקסים" בעמודה השנייה.

3] עבור הלוח הבא בגודל 3x3 (n=3):

```
|O|X|O|
|X|O|X|
|X|O|X|
```

אין מנצח (זהו תיקו) שכן אף שחקן לא הפסיד (לא עשה רצף של n).

קלט ומהלך המשחק:
בתחילת המשחק יש להציג הודעת פתיחה, לקלוט את גודל הלוח, להדפיס את הלוח ההתחלתי ולבקש משחקן 1 להכניס קלט.
בהכנסת קלט ניתן להקיש צעד, או בקשת undo.
Undo = שחזור מצב המשחק כפי שהיה בעבר, מצוין על ידי הכנסת מספר שלילי אי זוגי (במקום אינדקס השורה). כלומר, אם מוכנס מספר שלילי אי זוגי, יש להעביר את הלוח (ובעצם את המשחק) מספר צעדים אחורה כפי המספר שהוכנס. למשל, אם הוכנס 3- יש להעביר את הלוח למצב שהיה 3 צעדים אחורה ולהציגו. לאחר ביצוע ה-undo התור עובר למי שהיה תורו באותה נק' זמן אליה חזרנו (שימו לב

שההגבלה על האי זוגיות יוצרת מצב שבו תמיד לאחר ביצוע ה undo התור הבא יהיה של היריב, ביצוע undo עם מספר זוגי נחשבת לפקודת undo לא חוקית ועליכם לבדוק זאת).
צעד = הכנסת "איקס" (אם מדובר בשחקן 1) או "עיגול" (אם מדובר בשחקן 2) לתא בלוח על ידי קליטת אינדקס השורה והעמודה. אם הוכנס מספר אי שלילי עבור אינדקס השורה, יש להכניס עוד מספר עבור העמודה. הצעד חוקי אם:
מספר השורה ומספר העמודה הם מספרים בין 1 ל n והמקום המתאים בלוח ריק (לא הוכנס קודם).
אם הצעד חוקי יש לבצעו, להציג את הלוח העדכני ולעבור לשחקן הבא.
לא לשכוח לבדוק אם השחקן הפסיד ואז השני ניצח.
אם הצעד או פעולת ה undo לא חוקיים יש להדפיס הודעת שגיאה (print_error) ולתת לשחקן הזדמנות נוספת עד שיכניס פקודה חוקית.

<u>נכונות קלט:</u>

- ניתן להניח שהקלט הוא רק מספרים שלמים (יש לקלוט באמצעות %d), רווחים ו enter.
- ניתן להניח שאם המספר הראשון שהוכנס שלילי (כלומר השחקן רוצה לבצע undo) אז לאחריו יהיה enter. אחרת (אם הראשון אי שלילי) מובטח שיופיע עוד מספר אחריו (ולאחר מכן enter).
- ניתן להניח שגודל הלוח (בהתחלה) הוא חוקי (מספר בין 1 ל 11).
- לא ניתן להניח שהצעדים חוקיים (דוגמא לצעד לא חוקי – שחקן 1 מבקש לשים "איקס" בשורה 1, עמודה 3 כאשר גודל הלוח הוא 2 על 2).
- לא ניתן להניח שבקשת ה undo היא חוקית (דוגמא לבקשת undo לא חוקית – שחקן מבקש לעשות undo עם מספר צעדים גדול יותר ממה שהיה במשחק).
- במקרה של גילוי פעולה לא חוקית כלשהי יש להדפיס הודעת שגיאה (print_error) ולתת למשתמש הזדמנות להקיש מהלך נוסף.

<u>מקרה הרצה לדוגמה (טסט מספר 1):</u>

```
*** Welcome to AVOIDANCE TIC-TAC-TOE game ***

Please enter board size (1 to 11):
3

Current board:
|_|_|_|
|_|_|_|
|_|_|_|


Player ** 1 **, enter next move:
1 1

Current board:
|X|_|_|
|_|_|_|
|_|_|_|


Player ** 2 **, enter next move:
2 2

Current board:
|X|_|_|
|_|O|_|
|_|_|_|


Player ** 1 **, enter next move:
3 3

Current board:
|X|_|_|
|_|O|_|
|_|_|X|
```

```
Player ** 2 **, enter next move:
2 3

Current board:
|X|_|_|
|_|O|O|
|_|_|X|


Player ** 1 **, enter next move:
1 3

Current board:
|X|_|X|
|_|O|O|
|_|_|X|


Player ** 2 **, enter next move:
3 1

Current board:
|X|_|X|
|_|O|O|
|O|_|X|


Player ** 1 **, enter next move:
-1

Current board:
|X|_|X|
|_|O|O|
|_|_|X|


Player ** 2 **, enter next move:
3 2

Current board:
|X|_|X|
|_|O|O|
|_|O|X|


Player ** 1 **, enter next move:
4 1
Illegal move!!!, please try again:
3 1

Current board:
|X|_|X|
|_|O|O|
|X|O|X|


Player ** 2 **, enter next move:
2 1

Current board:
|X|_|X|
|O|O|O|
|X|O|X|

Player 1 Wins! Hooray!
```

שחקן 1 בוחר לשחזר את הלוח (ובעצם את המשחק) כפי שהיה צעד 1 אחורה. כלומר הצעד האחרון (שהוא הצעד של שחקן 2) מתבטל ומכיוון שאז התור היה של שחקן 2, כעת שוב תורו של שחקן 2.

צעד לא חוקי! (מחוץ ללוח כי אין שורה 4)

שחקן 2 עשה רצף של 3 "עיגולים" בשורה 2 ולכן הפסיד, כלומר שחקן 1 הוא המנצח!

עם התרגיל פורסמו 4 טסטים כדי שתוכלו לבדוק את עצמכם (מקרה ההרצה שלעיל הוא טסט מספר 1).

חשוב:

- אחרי קריאת התרגיל והבנתו – קראו והבינו היטב את פונקציות ההדפסה ומתי יש להשתמש בהן, מומלץ אפילו לנסות להריץ אותן, זה יחסוך לכם זמן יקר בהמשך.
- עליכם להשתמש בעקרונות שנלמדו בתרגול על Top-Down Design, זה יקל עליכם את התכנון, המימוש והדיבוג של התוכנית. בנוסף, הקוד שלכם יהיה הרבה יותר קריא.

בהצלחה!