שאלה 1

10 נקודות

בכל סעיף, עליכם לכתוב האם "**תמיד נכון**" בשפת ANSI-C, "**לפעמים נכון ולפעמים אינו נכון**" או "**תמיד אינו נכון**". עליכם לנמק את תשובתכם, תשובה לא מנומקת, גם אם היא נכונה, לא תזכה בנקודות.

סעיף א'

5 נקודות

קובץ הקלט הסטנדרטי stdin הוא **תמיד** המקלדת, קובץ הפלט הסטנדרטי stdout הוא **תמיד** המסך, וקובץ השגיאות הסטנדרטי stderr הוא **תמיד** המסך.

תמיד אינו נכון

<u>נימוק:</u>

stdin, stdout ו־stderr הם תיאורי הקבצים שמופעלים בידי מערכת ההפעלה ומאפשרים לתוכנה לתקשר באמצעות קלט, פלט והודעות שגיאה (בהתאמה). מערכת ההפעלה היא שקובעת כיצד אותם קבצים יתקבלו ויובאו למול המשתמש. את ברירת המחדל, תהיה אשר תהיה, המשתמש יכול לשנות באמצעות redirection, הפניה של זרמי הנתונים כך שינותבו מחדש למקור נתונים ויעדי פלט שונים נוספים.

כך למשל, יצרתי Makefile להלן: (הדגש בהבאתו, שהוא כולל דגלי **-ansi**)

```
example: example.o
  gcc -ansi -Wall -pedantic -g example.o -o $@
example.o: example.c example.h
  gcc -ansi -Wall -pedantic -g -c example.c -o $@
```

ולאחר מכן הרצתי:

```
./example <./i.txt >./o.txt 2>&1
```

stdin הופנה ל־`./i.txt`, stdout ל־`./o.txt`, והביטוי `2>&1` גרם להפניה של stderr השמור באינדקס השלישי (#2) אל מיקומו של האינדקס השני (#1) המציין את stdout, כך שגם stderr הופנה ל־`./o.txt`. כך, הובאה לכל אחד מהתיאורים דוגמה נגדית.

יתר על כן, אפילו אם המשתמש לא היה מסוגל לשנות את ברירת המחדל של מערכת ההפעלה, בעבר היו כאלו שבהן זרמי הנתונים הועברו בברירת מחדל באמצעים אחרים, ייתכן שאף פיזיים ממש כגון דיסקטים ומדפסות. כל זאת באופן לחלוטין בלתי־תלוי בשפה ANSI-C עצמה.

סעיף ב'

5 נקודות

אם נגדיר משתנה כ־**static register**, נוכל לייעל את זמן השימוש במשתנה, וכן לשמור על ערכו מקריאה לקריאה.

בהנחה שאין כאן כוונה לומר שהאמרה נכונה באופן ריק; אני פשוט מתייחס לנטען כשקול ל־"**ניתן** להגדיר משתנה כ־static register, ולייעל את...".

תמיד אינו נכון

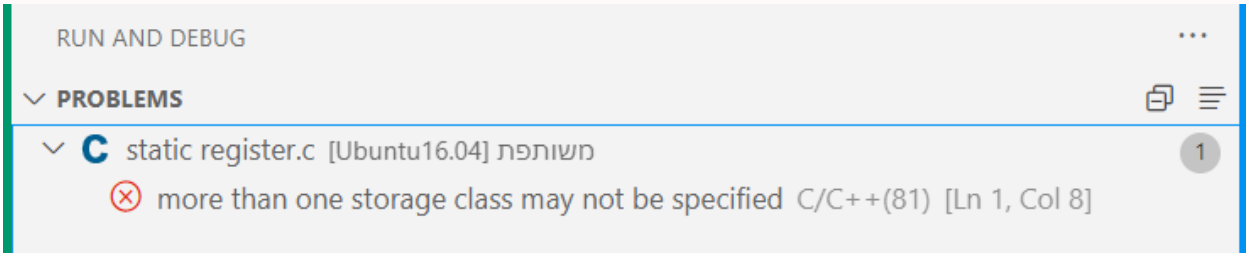
<u>נימוק:</u>

ניסיתי ליצור את הקוד:

```c
static register int Does_static_register_works_in_anci_c;
/*?      Let's check */
```

והתקבלה השגיאה –

```
user@ubuntu:/mnt/hgfs/SHARED$ gcc -ansi -Wall -pedantic -c "static register.c" -o "static register.o"
static register.c:1:1: error: multiple storage classes in declaration specifiers
 static register int Does_static_register_works_in_anci_c; /*?      Let's check */
 ^
static register.c:1:21: warning: 'Does_static_register_works_in_anci_c' defined but not used [-Wunused-variable]
 static register int Does_static_register_works_in_anci_c; /*?      Let's check */
                     ^
user@ubuntu:/mnt/hgfs/SHARED$
```



– על גרסאותיה השונות.

לפיה, ניתן לצרף רק אחד מתוך ה־storage class specifiers: `static`, `extern` או `register`.

ה־`static` וה־`register` בנפרד אכן ישמרו על ערכו מקריאה לקריאה, או לחילופין בהתאמה ייעלו את זמן השימוש במשתנה, אך לא ביחד.

אני מצליח לזהות היגיון אפשרי שיסביר את המגבלה הזאת (לא הצלחתי להתעמק בדפי המפרט):
כדי לשמור על ערך המשתנה מקריאה לקריאה יש לאחסן אותו למשכי זמן יחסית ארוכים שבהם לא יהיה בשימוש, ובמהלכם, תפיסת המקום בזיכרון הרגיסטרים המועט (למטרות ייעול זמן השימוש) תגביל את פעולת המעבד יותר מאשר תייעל אותה.