שפת C – תרגיל 3

מצביעים לפונקציות, Makefiles, ספריות

תאריך הגשה: יום שני 30.11.15 עד שעה 23:55

הגשה מאוחרת (בהפחתת 10 נקודות): יום שלישי 1.12.15 עד שעה 23:55

תאריך ההגשה של הבוחן: יום שני 30.11.15 עד שעה 23:55

1. הנחיות חשובות:

1. בכל התרגילים יש לעמוד בהנחיות הגשת התרגילים וסגנון כתיבת הקוד. שני המסמכים נמצאים באתר הקורס – הניקוד יכלול גם עמידה בדרישות אלו.
2. בכל התרגילים עליכם לכתוב קוד ברור. בכל מקרה בו הקוד שלכם אינו ברור מספיק עליכם להוסיף הערות הסבר בגוף הקוד. יש להקפיד על תיעוד (documentation) הקוד ובפרט תיעוד של כל פונקציה.
3. במידה ואתם משתמשים בעיצוב מיוחד או משהו לא שגרתי, עליכם להוסיף הערות בקוד המסבירות את העיצוב שלכם ומדוע בחרתם בו.
4. בכל התרגילים במידה ויש לכם הארכה ואתם משתמשים בה, **חל איסור להגיש קובץ כלשהוא בלינק הרגיל (גם אם לינק ההגשה באיחור טרם נפתח)**. **מי שיגיש קבצים בשני הלינקים מסתכן בהורדת ציון משמעותית.**
5. אין להגיש קבצים נוספים על אלו שתדרשו.
6. עליכם לקמפל עם הדגלים Wall -Wextra -Wvla -std=c99 ולוודא שהתוכנית מתקמפלת ללא אזהרות, **תכנית שמתקמפלת עם אזהרות תגרור הורדה בציון התרגיל.** למשל, בכדי ליצור תוכנית מקובץ מקור בשם ex1.c יש להריץ את הפקודה:

```
gcc -Wextra -Wall -Wvla -std=c99 ex1.c -o ex1
```

7. עליכם לוודא שהתרגילים שלכם תקינים ועומדים בכל דרישות הקימפול והריצה במחשבי בית הספר מבוססי מעבדי bit-64 (מחשבי האקוריום, 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)
8. לאחר ההגשה, בדקו את הפלט המתקבל בקובץ ה-PDF שנוצר מהpresubmission script בזמן ההגשה. באם ישנן שגיאות, תקנו אותן על מנת שלא לאבד נקודות.

   שימו לב ! תרגיל שלא יעבור את ה presubmission script ציונו ירד משמעותית (הציון יתחיל מ-50, ויוכל לרדת) ולא יהיה ניתן לערער על כך.
9. בדיקת הקוד לפני ההגשה, גם על ידי קריאתו וגם על ידי כתיבת בדיקות אוטומטיות (tests) עבורו היא אחריותכם. חישבו על מקרי קצה לבדיקת הקוד.

קבצי בדיקה לדוגמה ניתן למצוא פה: ~slabc/www/ex1/tests_examples.tar

שימוש בקבצים אלו הוא באחריותכם. במהלך הבדיקה הקוד שלכם ייבדק מול קלטים נוספים לשם מתן הציון.

10. **הגשה מתוקנת -** לאחר מועד הגשת התרגיל ירוצו הבדיקות האוטמטיות ותקבלו פירוט על הטסטים בהם נפלתם. לשם שיפור הציון יהיה ניתן להגיש שוב את התרגיל לאחר תיקוני קוד קלים ולקבל בחזרה חלק מהנקודות - פרטים מלאים יפורסמו בפורום ואתר הקורס.

2. מידע חשוב נוסף:

1. ניתן להתחבר באמצעות SSH למחשבי בית הספר (למשל לשם בדיקת הקוד לפני הגשה מהבית)
http://wiki.cs.huji.ac.il/wiki/Connecting_from_outside
2. עליכם להכיר את ספריית הקלט-פלט של שפת C ובייחוד את השימוש בפונקציות printf ו scanf
http://www.cplusplus.com/reference/clibrary/cstdio

3. הנחיות ספציפיות לתרגיל זה:

1. חל איסור להשתמש במערכים בגודל דינמי (VLA). שימוש כזה יוביל לפסילת הסעיף הרלוונטי.
2. עליכם לוודא שהקוד שלכם רץ באופן תקין וללא דליפות זכרון. לשם כך עליכם להתשמש בתוכנת valgrind (ראו פירוט בהמשך).
3. אתם רשאים להשתמש בכל הספריות הסטנדרטיות של C.
4. בתרגיל זה אתם רשאים (ואף צריכים) לשנות את קבצי ה-header.
5. בתרגיל זה אתם רשאים להוסיף קבצים נוספים.

4. מספרים מורכבים:

עליכם לממש מחלקה של מספרים מורכבים.

1. עליכם לשנות את Complex.h ולהוסיף את החתימה של המתודה getCompareFunc.
2. עליכם לממש בקובץ Complex.c את כל הפונקציות המתועדות בקובץ Complex.h.
3. הדרכה והנחיות כלליות:
   - אתם רשאים להגדיר פונקציות נוספות לשימושכם הפנימי.
   - פונקציות פנימיות (שאינן מופיעות ב-interface) צריכות להיות מוגדרות כ- static function.

5. חבורה אבלית:

1. <u>רקע</u>

a. עליכם לייצר ספריה גנרית בשם GenGroup העוסקת בחיבור והרכבת פונקציות ובהגדרת חבורה אבלית. הספרייה היא למעשה אוסף פונקציות המוגדרות בקובץ GenGroup.h אותן עליכם ליישם. ככותבי הספריה אתם מתחייבים לממשק הפונקציות הנתון, כאשר המימוש הפנימי אינו חשוף למשתמש בספריה.

b. הגדרת הממשק האמור נמצאת בקובץ GenGroup.h המצורף לתרגיל. קראו היטב את הגדרות הממשק ואת הפונקציות שעליכם לממש. שימו לב שעליכם לשנות את הקובץ GenGroup.h בהתאם לתיאור שמופיע בקובץ.

2. <u>מימוש:</u>

a. עליכם להוסיף לקובץ GenGroup.h את החתימה של המתודות החסרות: addFunctions, composeFunctions.

b. בהתאם, עליכם גם לשנות את החתימה של המתודה isAbelianGroup.

c. עליכם לכתוב מימוש למודול GenGroup לפי הממשק המוגדר בקובץ GenGroup.h את המימוש עליכם להגיש בקובץ GenGroup.c

d. הדרכה והנחיות כלליות:

- לכל מתודה המוגדרת ב-GenGroup.h עליכם להוסיף תיעוד בקובץ GenGroup.c המתאר את מידת הסיבוכיות של המימוש שלכם.
- אתם רשאים להגדיר פונקציות נוספות לשימושכם הפנימי.
- פונקציות פנימיות (שאינן מופיעות ב-interface) צריכות להיות מוגדרות כ- static function.
- שימו לב שאתם משחררים את **כל** הזיכרון שהוקצה.
- הגדרת חבורה אבלית:

חבורה אבלית הינה קבוצה סגורה עם פעולה בינארית (נסמנה ב- *) ואיבר יחידה (נסמנו ב- 1) שמקיימת את התנאים הבאים:

  - סגירות - לכל a,b בקבוצה => a*b גם בקבוצה.
  - קיבוצים - לכל a,b,c בקבוצה => a*(b*c)==(a*b)*c
  - איבר יחידה - לכל a בקבוצה => a*1==1*a==a.
  - קיום איבר הופכי - לכל a בקבוצה => קיים איבר b בקבוצה שמקיים a*b==b*a==1
  - חילופיות - לכל a,b בקבוצה => a*b==b*a.

לפירוט נוסף ראו ויקפדיה.

- סיבוכיות המתודה isAbelianGroup - צריכה להיות O(n^3).

6. פונקצית <u>main</u>

a) עליכם לכתוב בקובץ MyGroupMain.c תוכנית שמדגימה 4 דוגמאות של קריאות למתודה isAbelianGroup.

b) 2 הדוגמאות הראשונות הן על קבוצה של מספרים מורכבים (Complex).

(1) הגדירו פונקציה כלשהיא שפועלת על 2 מספרים מורכבים ומחזירה מספר מורכב.

(2) עליכם להגדיר 2 קבוצות (שמכילות לפחות 2 אברים) של מספרים מורכבים ולהגדיר איבר יחידה בכל אחת מהן.

(3) על אחת מהקבוצות להיות חבורה אבלית (עם הפונקציה שהגדרתם) ועל השניה לא.

c) באופן דומה 2 הדוגמאות השניות הן על קבוצה של מספרים מסוג int/double (בחרו טיפוס כרצונכם).

(1) הגדירו פונקציה כלשהיא שפועלת על 2 מספרים מהטיפוס שבחרתם ומחזירה מספר מאותו טיפוס.

(2) עליכם להגדיר 2 קבוצות (שמכילות לפחות 2 אברים) של מספרים מהטיפוס שבחרתם ולהגדיר איבר יחידה בכל אחת מהן.

(3) על אחת מהקבוצות להיות חבורה אבלית(עם הפונקציה שהגדרתם) ועל השניה לא.

d) על התוכנית שלכם להדפיס בצורה אינפורמטבית מה עושה הפוקנציה שבחרתם ואת כל אחת מהקבוצות והאם מדובר בחבורה אבלית. <u>אין הגבלה על איך אתם מדפיסים, העיקר שיהיה אינפורמטיבי</u>.

e) להלן דוגמא להדפסה של קבוצה כזו:

```
/////////////////
Complex Group with regular Complex add function:
Identity number: 0,0i
Group members: 0,0i ; 1,0i ; 3,0i
This is not Abelian group
//////////////////////
```

2) <u>ספרייה סטטית</u>

a) עליכם ליצור מהמימוש שלכם GenGroup ספרייה סטטית. שם הקובץ של הספרייה צריך להיות libgenGroup.a.

b) כאשר אתם באים להשתמש בספריה זו (לדוגמא בדרייבר MyGroupMain.c), עליכם לעשות linkage לספריה הסטטית שיצרתם.

7. **עבודה עם valgring:**

1. ניהול זיכרון ב-C הוא נושא רגיש ומועד לפורענות – יש הרבה אפשרויות לטעות (לא להקצות מספיק זיכרון, לשכוח לשחרר זיכרון, להשתמש במצביעים שמצביעים לזבל וכו'). כמובן שהקומפיילר לא ידווח על שגיאה בכל המקרים הללו. יתכן שתגלו את השגיאות הללו בזמן ריצה, אך יתכן גם כי התוכנה תעבוד אצלכם "במקרה" והבעיות יתגלו דווקא בביתו של הלקוח.
2. ישנו מבחר די גדול של תוכנות בשוק שמטרתם לסייע באיתור בעיות זיכרון בקוד לפני שחרורו אל הלקוח. אנו נשתמש בתוכנת valgrind, שיחסית לתוכנה חינמית, נותנת תוצאות מעולות. בתרגיל זה אנו מבקשים מכם להריץ את valgrind עם התוכנה שלכם. את הפלט שלה יש להגיש בקובץ בשם valdbg.out.
3. כדי להריץ את valgrind עליכם לבצע קומפילציה ו-linkage לקוד שלכם עם הדגל '-g' (הן בשורת הקומפילציה והן בשורת ה-linkage). לאחר מכן הריצו valgrind:

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

4. אם קיבלתם הודעת שגיאה, יתכן שתצטרכו לבצע שינוי הרשאות:

```
> chmod 777 ProgramName
```

5. כמובן שאם valgrind דיווח על בעיות עם הקוד שלכם, עליכם לתקן אותן.
6. היעזרו ב-tutorial הקצרצר של valgrind שבאתר הקורס.

8. **הערות למשימות התכנות:**

1. התכניות יבדקו גם על סגנון כתיבת הקוד וגם על פונקציונאליות, באמצעות קבצי קלט שונים (תרחישים שונים להרצת התכניות). הפלט של פתרונותיכם יושווה (השוואת טקסט) לפלט של פתרון בית הספר (בתרגיל זה רק הפלט של Complex.o יושווה לפתרון בית הספר). לכן עליכם להקפיד על פורמט הדפסה מדויק, כדי למנוע שגיאות מיותרות והורדת נקודות.
2. לרשותכם כמה קבצי קלט לדוגמה וקבצי הפלט המתאימים להם (אלו מהווים רק חלק קטן מקבצי הקלט-פלט שנשתמש בהם, כתבו לעצמכם בדיקות נוספות). עליכם לוודא שהתכנית שלכם נותנת את אותו הפלט בדיוק.
3. על מנת לעשות זאת הריצו את תכניתכם עם הקלט לדוגמה על ידי ניתוב ה standard input להקרא מקובץ (באמצעות האופרטור ">" בשורת ההרצה ב terminal), ונתבו את הפלט של תכניתכם, שהוא ה standard output,לתוך קובץ (באמצעות האופרטור "<") באופן הבא:

```
ProgramName < inputFile > myOutputFile
```

4. השוו את קובץ הפלט שנוצר לכם עם קובץ הפלט המתאים של פתרון בית הספר, באמצעות הפקודה diff

diff הנה תוכנה להשוואה טקסטואלית של שני טקסטים שונים. בהינתן שני קבצי טקסט להשוואה
(1.txt ,2.txt) הפקודה הבאה תדפיס את השורות אשר אינן זהות בשני הקבצים:

```
diff 1.txt 2.txt
```

במידה והקבצים זהים לחלוטין, לא יודפס דבר.

קראו על אפשרויות נוספות של diff בעזרת הפקודה man diff. לחלופין אתם יכולים גם להשתמש בתוכנה tkdiff אשר מראה גם את השינויים ויזואלית.

כמו כן, אתם יכולים גם להשוות ישירות באופן הבא:

```
ProgramName < inputFile | diff expected.out
```

5. אם ישנם מקרים שהוראות התרגיל לא מציינות בבירור כיצד התכנית צריכה להתנהג, הביטו בקבצי הקלט וקבצי הפלט לדוגמה שניתנים לכם ובדקו אם התשובה לשאלתכם נמצאת שם. כמו כן, היעזרו בפתרון בית הספר, הריצו עליו את הטסטים שלכם והשוו להתנהגות תוכניתכם.

9. חומר עזר:

1. את הקבצים הנדרשים לצורך התרגיל ניתן למצוא ב:

```
~labc/www/ex3/ex3_files.tar
```

2. את פתרון בית הספר ניתן למצוא ב:

```
~labc/www/ex3/school_sol.tar
```

3. מצביעים לפונקציות ב-C:

```
http://www.newty.de/fpt/index.html
```

10. הגשה

1. עליכם להגיש קובץ tar בשם ex3.tar המכיל לפחות את הקבצים הבאים:

- קובץ פלט של valgrind:
  - valdbg.out - פלט הריצה עם valgrind של MyGroupMain שלכם.
- קובץ Makefile התומך לפחות בפקודות הבאות:
  - make GenGroup - יצירת ספריה סטטית libgenGroup.a (ללא בדיקות debug).
  - make main - קימפול,ויצירת תוכנית MyGroupMain (ללא בדיקות debug).
  - make Complex.o - קימפול, ויצירת Complex.o (ללא בדיקות debug).
  - make - קימפול, יצירת תוכנית והרצת MyGroupMain (ללא בדיקות debug).

- make clean - ניקוי כל הקבצים שנוצרו באמצעות פקודות ה-Makefile (וניתן לשחזר באמצעות קריאה מחודשת לפקודות ה- make המתאימות)

- GenGroup.c, GenGroup .h, Complex.c, Complex.h, MyGroupMain .c.
- **<u>אין להגיש את Epsilon.h.</u>**
- שימו לב! - אל אף שאתם יכולים להוסיף קבצים נוספים כרצונכם, המנעו מהוספת קבצים לא רלוונטים (גם בכדי להקל על הבודקים, וגם בכדי שציונכם לא יפגע מכך).

2. לפני ההגשה, פתחו את הקובץ ex3.tar בתיקיה נפרדת וודאו שהקבצים מתקמפלים ללא שגיאות וללא אזהרות.
3. מומלץ מאוד גם להריץ בדיקות אוטומטיות וטסטרים שכתבתם על הקוד אותו אתם עומדים להגיש.
4. אתם יכולים להריץ בעצמכם בדיקה אוטומטית עבור סגנון קידוד בעזרת הפקודה:

~labc/www/codingStyleCheck <code file or directory>

כאשר <directory or file> מוחלף בשם הקובץ אותו אתם רוצים לבדוק או תיקייה שיבדקו כל הקבצים הנמצאים בה (שימו לב שבדיקה אוטומטית זו הינה רק חלק מבדיקות ה codingStyle)

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

~labc/www/ex3/presubmit_ex3

בהצלחה!