שפת C – תרגיל 3

הקצאת זיכרון דינמית, מצביעים לפונקציות, Makefiles, ספריות

<u>תאריך הגשה:</u> יום רביעי 12.8.15 עד שעה 23:55

<u>הגשה מאוחרת (בהפחתת 10 נקודות):</u> יום חמישי 13.8.15 עד שעה 23:55

<u>תאריך ההגשה של הבוחן:</u> יום רביעי 12.8.15 עדהשעה 23:55

1) <u>הנחיות חשובות:</u>

a) בכל התרגילים יש לעמוד בהנחיות הגשת התרגילים וסגנון כתיבת הקוד. שני המסמכים נמצאים באתר הקורס – הניקוד יכלול גם עמידה בדרישות אלו.

b) בכל התרגילים עליכם לכתוב קוד ברור. בכל מקרה בו הקוד שלכם אינו ברור מספיק עליכם להוסיף הערות הסבר בגוף הקוד.

c) במידה ואתם משתמשים בעיצוב מיוחד או משהו לא שגרתי, עליכם להוסיף הערות המסבירות את העיצוב שלכם ומדוע בחרתם בו.

d) אין להגיש קבצים נוספים על אלו שתדרשו.

i) בתרגיל זה יש צורך להגיש קובץ README.

ii) **עליכם לקמפל עם הדגלים <u>-Wall</u>** -Wextra -Wvla 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:

```
c99 –Wextra -Wvla -Wall ex1.c -o ex1
```

e) עליכם לוודא שהתרגילים שלכם תקינים ועומדים בכל דרישות הקימפול והריצה במחשבי בית הספר מבוססי מעבדי bit-64 (מחשבי האקוריום, לוי, השרת river). **<u>חובה להריץ את התרגיל במחשבי בית הספר לפני ההגשה</u>**. (ניתן לוודא שהמחשב עליו אתם עובדים הנו בתצורת bit-64 באמצעות הפקודה "uname -a" ווידוא כי הארכיטקטורה היא 64, למשל אם כתוב x86_64)

f) לאחר ההגשה, בדקו את הפלט המתקבל בקובץ ה-PDF שנוצר מהpresubmission script בזמן ההגשה. באם ישנן שגיאות, תקנו אותן על מנת שלא לאבד נקודות.

<u>שימו לב ! תרגיל שלא יעבור את ה presubmission script ציונו ירד משמעותית</u> (הציון יתחיל מ-50, ויוכל לרדת) **<u>ולא יהיה ניתן לערער על כך.</u>**

g) בדיקת הקוד לפני ההגשה, גם על ידי קריאתו וגם על ידי כתיבת בדיקות אוטומטיות (tests) עבורו היא אחריותכם. חישבו על מקרי קצה לבדיקת הקוד.

h) **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 הקורס.

2) מידע חשוב נוסף:

a) ניתן להתחבר באמצעות SSH למחשבי בית הספר (למשל לשם בדיקת הקוד לפני הגשה מהבית)
http://wiki.cs.huji.ac.il/wiki/Connecting_from_outside

b) עליכם להכיר את ספריית הקלט-פלט של שפת C ובייחוד את השימוש בפונקציות printf ו scanf-
http://www.cplusplus.com/reference/clibrary/cstdio

c) תזכורת: כדי להשתמש בספריות של שפת C יש להכליל את הספרייה באמצעות הוספת קובץ ה-header המתאים. לדוגמה, כדי להשתמש בספריית הקלט-פלט יש לכתוב בראש הקוד של התכנית

```
#include <stdio.h>
```

3) MyString:

1. רקע

a) ועדת התקן של C נהנתה כל כך מהפונקציות שכתבתם בתרגילים הקודמים שהיא הטילה עליכם ליצור מבנה חדש לייצוג של מחרוזות :) .

b) בתרגיל זה, עליכם לייצר ספרייה שמרכזה ב-struct בשם MyString המייצג מחרוזת המסוגלת לתמוך בכל הפונקציות הקיימות עבור C String. המודול הוא למעשה אוסף פונקציות המוגדרות בקובץ MyString.h אותן עליכם ליישם. ככותבי הספריה אתם מתחייבים לממשק הפונקציות הנתון, כאשר המימוש הפנימי אינו חשוף למשתמש בספריה.

c) הגדרת הממשק האמור נמצאת בקובץ MyString.h המצורף לתרגיל. קראו היטב את הגדרות הממשק ואת הפונקציות שעליכם לממש. כמו כן מצורף קובץ הרצה לדוגמא העושה שימוש במודול, בקובץ mystring_demo.c אותו תוכלו להריץ כדי לבדוק את המודול שלכם ולהבין מהו השימוש שיעשה בו .

d) שימו לב כי הממשק המוגדר בקובץ MyString.h מגדיר לכם מה לעשות ולא כיצד ליישם זאת. בפרט, בעת כתיבת מימוש הממשק, עליכם לשקול את הנקודות הבאות:

- אילו פרטים כדאי לייצג לגבי המבנה מלבד אחסון תוכנו.
- באילו אלגוריתמים תשתמשו לצורך יישום הפונקציות השונות בממשק.
- מתי יתרחשו הקצאות ושחרורי הזכרון ובאיזה אופן.

2. <u>יישום הממשק MyString</u>

a) עליכם להוסיף לקובץ MyString.h את החתימה של המתודות החסרות `myStringCustomEqual`, `myStringCustomCompare`, myStringSort, myStringCoustomSort

(בהתאם לתיעוד המופיע בקובץ).

b) עליכם לכתוב מימוש למודול `MyString` לפי הממשק המוגדר בקובץ MyString.h את המימוש עליכם להגיש בקובץ MyString.c

c) הדרכה והנחיות כלליות:

- **לכל מתודה המוגדרת ב-MyString.h עליכם להוסיף תיעוד בקובץ MyString.c המתאר את מידת הסיבוכיות של המימוש שלכם.**
- אתם מממשים מעטפת חכמה למחרוזת (מערך תווים) של C, ולכן אין צורך להשתמש ב-"`\0`". עליכם רק לוודא שהמימוש שלכם עקבי עם עצמו.
- גודל המחרוזות שיהיו בשימוש אינו ידוע מראש ויקבע בזמן ריצה ע"י המשתמש בספרייה שתממשו.
- הממשק מטפל בפעולות רבות הדורשות הקצאת זיכרון. שימו לב להקצאות הזיכרון ונסו שלא לבצע הקצאות שלא לצורך מאחר וזו פעולה יקרה.
- אל תתעלמו משגיאות בהקצאת זיכרון. קראו בקפידה את תיעוד הממשק ב-MyString.h ופעלו בהתאם.
- <u>אסור בשום מקום במימוש להשתמש בפונקציות מספריית ה-string של C</u> (כדוגמת : strcpy,strcmp).
- <u>יוצא מן הכלל הקודם</u> - מותר לכם להשתמש בפונקציות ששמותיהן מתחילות במילה "mem" מתוך ספריית ה-string (כגון `memcmp`, `memcpy`).
- <u>אסור להשתמש ב-atoi, itoa, strtod וכד'</u>.
- חלק מפונקציות ב-Interface דומות - הימנעו משכפול קוד.
- כמובן שניתן להגדיר פונקציות נוספות לשימושכם הפנימי.
- שימו לב שאתם משחררים את **כל** הזיכרון שהוקצה.

- פונקציות פנימיות (שאינן מופיעות ב-interface) צריכות להיות מוגדרות כ- static function.
- את המבנה MyString_ עליכם להגדיר בקובץ המימוש שלכם MyString.c . לדוגמא:

```
struct _MyString
{
   // My members
};
```

במבנה זה תשמרו את השדות להם אתם זקוקים בייצוג המחרוזת, כגון מערך ה-char.

3. <u>פונקצית main</u>

a) עליכם לכתוב בקובץ MyStringMain.c תוכנית המקבלת מהמשתמש שתי מחרוזות (A,B) ושומרת לקובץ בשם "test.out", את המשפט "A is smaller than B", אם A קודם או שווה בסדר אלפבתי ל B; ואת המשפט "B is smaller than A", אם B קודם בסדר אלפבתי ל A.
כאשר A ו-B הם בעצם המחרוזות שהתקבלו מהמשתמש שהושוו באמצעות הפונקציה MyString_compare.
למשל עבור הקלט "az" ו- "abc" (לא משנה באיזה סדר נקלטו מהמשתמש) תוכן הקובץ יהיה "abc is smaller than az\n"

b) מאחר והקלט הוא מחרוזת, כל קלט הוא תקין (אולם אתם יכולים להניח שהוא לא יכלול ירידת שורה, וכי אורך הקלט <=500 תוים).

c) אין הגבלה על איך אתם מבקשים את הקלט מהמשתמש, העיקר שיהיה אינפורמטיבי.

4. <u>דרישות יעילות</u>

a) המבנה החדש שלכם צריך להיות יעיל ככל הניתן בזיכרון ובזמני הביצוע באופן כללי.
למשל עבור השימוש בפונקציות:
   1. myStringEqual– חישבו על מקרים בהם הפונק' יכולה לרוץ מהר יותר מ-(o(n)) והכניסו אותם למימוש.
   2. myStringSetFromMyString - גם כאן, חישבו כיצד ניתן לייעל את פעולת הפונקציה.

b) בעקבות קריאות לחלק מהפונקציות עשוים לחול שינויים בכמות הזכרון שהמבנה שלכם משתמש בה. במקרים אלו עליכם לשחרר את הזכרון באופן מושכל (מצד אחד,

אין טעם לשחרר זכרון של byte בודד. ומצד שני ברור שיש לשחרר זכרון של MyString שאורכו ירד ל-0). בחרו פתרון סביר כלשהו ונמקו את בחירתכם בתיעוד.

c) תארו בתחילת הקובץ MyString.c את הפרטים הבאים בעיצוב התוכנה שלכם ואת היתרון שבבחירות שביצעתם על פני עיצובים אחרים:
   - המבנה שבחרתם והמשתנים שבו.
   - מקומות בהם הקצתם זיכרון והשיטות לחיסכון בהקצאות בהן השתמשתם.
   - אלגוריתמים מיוחדים, שיטות עיצוב וכל פרט אחר הרלוונטי ליעילות התכנון שלכם.

5. <u>ספרייה סטטית</u>

a) בתרגיל זה, עליכם ליצור מהמימוש שלכם MyString ספרייה סטטית. שם הקובץ של הספרייה צריך להיות libmyString.a.

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

6. <u>unit-testing</u>

a) בכתיבת תוכנה נהוג לכתוב unit-testing. אלו יחידות קוד קטנות, המבצעות בדיקות תקינות של הפונקציות השונות, בדרך כלל במבנה מורכב יחסית. למשל במקרה שלנו אנו נרצה לוודא שפונקציה מסוימת בעץ עובדת ונרצה לבצע בדיקה זו בנפרד ככל האפשר משאר פעולת העץ. בדיקות אלו לא נועדו לבדוק את פלט התכנית, אלא בדיקות ברמה הרבה יותר בסיסית ומפורטת. כל בדיקה שכזו, מריצה את אחת הפונקציות עם פרמטרים שונים, ומשווה את ערך ההחזרה של השיטה עם ערך ההחזרה הצפוי. במקרה של שגיאה יש להדפיס פלט מתאים הכולל את שם השיטה שנכשלה, מה הפרמטרים, מה הערך הצפוי ומה היה ערך ההחזרה.

b) בתרגיל זה אתם מתבקשים לכתוב unit-testing לקובץ MyString.c. הבדיקות הינן חלק מתהליך המימוש, ולכן עליהן להיות בתוך הקובץ MyString.c (ישנן גם דרכים אחרות לעשות זאת, אך בתרגיל זה נבקש לפעול בדרך זו). מכיוון שאנו לא מעוניינים לספק אותן ללקוח, על הבדיקות להיות עטופות ב`#ifndef NDEBUG`. כך, כאשר הקוד יקומפל עם דגל זה אז הבדיקות ימחקו מהקוד. על ה-unit testing לכלול פונקציית main ובה מתבצעות בדיקות תקינות של <u>כל</u> המתודות ב MyString.

c) הפקודה make tests תקמפל קובץ ריצה MyStringTests **ותריץ** את הבדיקות שכתבתם. פורמט הבדיקות והפלט הינו פורמט חופשי, אך הרצת make tests צריכה לתת פלט שמסביר בצורה טובה מה נבדק והאם זה עבר את הטסטים או לא.

7. <u>שאלות ב-README</u>

לאחר שתקראו את הקובץ MyString.h תשימו לב כי מוצהר בו המבנה _MyString באופן הבא:

```
Struct _MyString;
```

טכניקה זו מכונה בשם forward declaration והיא מורה למהדר כי _MyString הוא מבנה אשר יש להכיר וכי הוא יוגדר לאחר מכן.

במקרה שלכם, המבנה יוגדר בקובץ MyString.c אותו אתם כותבים.

קובץ זה יעבור הידור יחד עם קובץ ה-h.

ענו על השאלות הבאות בקובץ ה-README:

1. כתבו מהם החסרונות בהגדרת _MyString במלואו בקובץ MyString.h באופן הבא:

```
struct _MyString
{
// My members actually defined here
...
};
typedef struct _MyString MyString;.
```

2. כאשר מהדרים קובץ המשתמש ב-MyString.h, ההידור יצליח למרות שהמהדר אינו יכול לדעת את גודל ומבנה _MyString. הסבירו בהרחבה כיצד הדבר מתאפשר.

4) <u>עבודה עם valgring</u>

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  MyStringTests
```

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

```
> chmod 777 MyStringTests
```

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

5) <u>בונוס (עד 8 נקודות)</u>

1. למתעניינים ב-Design Patterns מתקדמים יותר המאפשרים לייעל את המימוש של המודול בכמה מונים, מוצעת האפשרות לממש את המודול בשיטה מתקדמת יותר המבוססת על.Reference/pointer Counting
https://en.wikipedia.org/wiki/Reference_counting

המטרה של reference counting היא לצמצם את סה"כ כמות הזכרון אותם תופסים האובייקטים בתוכנית. כך, אובייקטים שבוצעה ביניהם השמה, ישמרו כולם הפנייה לאותו העתק של המידע ורק כאשר נבקש לשנות אחד מהם, אותו אחד יפריד את העתק המידע שלו מהשאר וישנה רק את ההעתק שלו. אם מספר אובייקטים משתפים מידע, המידע הזה יוכל להימחק רק כאשר אחרון האובייקטים הללו נמחק.

2. הדרישות לקבלת הבונוס הן לממש את הפונקציות הבאות ב-O(1) :

- `MyString_clone`
- `MyString_setFromMyString`
- `MyString_compare(const MyString *str1, const MyString *str2)` // if str2 was created from str1 using “clone” or “setFromMyString” or vice versa.

- MyString_swap

3. במידה ואתם בוחרים לממש את הבונוס, עליכם לציין זאת בהערה בתחילת המתודה.

6) <u>חומר עזר:</u>

1. את הקבצים המסופקים לצורך התרגיל ניתן למצוא ב:

~slabc/www/ex3/ex3_files.tar

2. לתיעוד של הפונקציות malloc, free, realloc וכו' השתמשו ב-man או חפשו ב-Google (המון מקורות אפשריים). כל ספר לימוד בסיסי מכיל תיאור של פונקציות אלו, ובנוסף דיברנו עליהן בכיתה
3. מצביעים לפונקציות ב-C:
   http://www.newty.de/fpt/index.html.
4. ליצירת ספריה סטטית ניתן להסתכל בדוגמא הבאה:
   http://www.adp-gmbh.ch/cpp/gcc/create_lib.html
5. הדפסה ל standard error בעזרת הפונקציה fprintf:
   http://www.cplusplus.com/reference/cstdio/fprintf/
   /http://www.cplusplus.com/reference/cstdio/stderr

7) <u>הגשה</u>

1. עליכם להגיש קובץ tar בשם ex3.tar המכיל רק את הקבצים הבאים:
   - קבצי פלט של valgrind:
     - valdbg_MyStringMain
     - valdbg_myStringTests
   - קובץ makefile התומך בפקודות הבאות:
     - make myString - יצירת ספריה סטטית libmyString.a
     - make tests - קימפול, יצירת תוכנית והרצת unit tests
     - make main - קימפול, יצירת תוכנית והרצת MyStringMain
     - make clean - ניקוי כל הקבצים שנוצרו באמצעות פקודות ה-Makefile
   - MyString.h, MyString.c ו- MyStringMain.c

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:

~slabc/www/codingStyleCheck <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.

~slabc/www/ex3/presubmit_ex3

בהצלחה!