תרגיל 4–מחרוזות, מצביעים, הקצאה דינמית ורקורסיה

להגשה עד ה- 18.01 בשעה 23:40

הוראות ההגשה ודגשים מיוחדים **מופיעים בסוף התרגיל! חובה לקרוא!**

מי שלא יעבוד בדיוק לפי ההנחיות יפסיד נקודות רבות מציון התרגיל! ראו הוזהרתם!

חלק א' – תאורטי (מענה בקובץ טקסט מוקלד בלבד):

משימה 1 – מה עושה התוכנית?

בכל סעיף עליכם להסביר שורה שורה מה מתבצע בתוכנית וכמובן לפרט **ולהסביר** מהו הפלט הסופי של התוכנית. (תשובה ללא פירוט והסבר לא תתקבל).

א.

```
#include<stdio.h>
int main() {
    static char *str[] = {"black", "white", "pink", "violet"};
    char **ptS[] = {str+3, str+2, str+1, str}, ***p;
    p = ptS;
    ++p;
    printf("%s", **p+1);
    return 0;
}
```

ב.

```
#include<stdio.h>
int main() {
    int array[2][2][2] = {10, 2, 7, 4, 1, 30, 5, 8};
    int *p, *p2;
    p = &array[1][1][1];
    p2 = (int*) array;
    printf("%d, %d\n", *p, *p2);
    return 0;
}
```

ג. (בהנחה שהתא הראשון במערך נשמר בכתובת 1000 בזיכרון).

```
#include<stdio.h>
int main(){
    int a[3][4] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 };
    printf("%u, %u, %u\n", a[0]+1, *(a[0]+1), *(*(a+0)+1));
    return 0;
}
```

משימה 2 – מהי השגיאה?

בכל סעיף הסבירו איזה סוג שגיאה יש בקוד הנ"ל (קומפילציה, קישור, ריצה, לוגית או אין שגיאה) וכן הסבירו ממה בדיוק היא נגרמת ואיך (אם בכלל) ניתן לתקנה.

א.

```
#include<stdio.h>
int main() {
    int *p;
    *p=100;
    return 0;
}
```

ב.

```
#include<stdio.h>
#inclue<stdlib.h>
#define SIZE 5
int main() {
    int i;
    int *p = (int*)calloc(SIZE,sizeof(int));
    for(i=0; i<SIZE; ++i) {
         *p=scanf("%d",p);
         p++;
    for(i=0; i<SIZE; ++i)
         printf("%d\t",p[i]);
    free(p);
    return 0;
}
```

ג.

```
#include<stdio.h>
int main() {
    int array[] = {10, 20, 30, 40, 50};
    int j;
    for(j=0; j<5; j++) {
        printf("%d\n", array);
        array++;
    }
    return 0;
}
```

ד.

```
#include<stdio.h>
#define SIZE 3
int main() {
    int i;
    int arr[SIZE] = {1,2,3,4,5};
    for(i=0; i<SIZE; ++i)
         printf("%d".arr[i]);
    return 0;
}
```

ה.

```
#include<stdio.h>
int main() {
    char str1[] = "Hello World";
    char str2[] = "Bye Bye";
    char *ptr1 = str1;
    char *ptr2 = str2;
    if(ptr1>ptr2)
        printf("\"%s\" is bigger than \"%s\".\n", ptr1, ptr2);
    else
        printf("\"%s\" is bigger than \"%s\".\n", ptr2, ptr1);
    return 0;
}
```

חלק ב' – מעשי (מענה בקובץ c. בלבד!):

בתרגיל זה עליכם לממש את כל מה שכתוב בתרגיל. שימו לב כי מדובר בתוכנית אחת(!) - פונקציה main אחת – שמפעילה הרבה פונקציות אחרות! **חישבו היטב כיצד לחלק את התוכנית לפונקציות!**

המערכת:

בתרגיל זה אתם מתבקשים לבנות מילון.

מילון הוא מבנה נתונים ממוין (לפי סדר ה- A, B, C) המכיל מילים באנגלית ולכל מילה ישנן מספר שונה של הגדרות שונות (הגדרה היא משפט באנגלית המסביר את פירושה של המילה).

המילון עצמו ישמר כמצביע ברמה שלישית ל-char (***char). הרעיון הוא שהמצביע בעצם יוקצה דינמית למערך של "כניסות" במילון. וכל "כניסה" (מטיפוס **char) – תהיה בעצם מערך דינמי (המשתמש יודיע בכל כניסה בנפרד באיזה אורך) של מחרוזות שבתא הראשון שלו (מסוג *char) תהיה המילה עצמה ובשאר התאים ההגדרות של אותה מילה.

בתחילת התוכנית המערכת תקבל את הנתונים שיכיל המילון מהמשתמש, תבנה את המילון לפי המידע ואז תיכנס לשלב החיפוש. בשלב החיפוש המשתמש יכול לחפש מילים במילון ולקבל את המשמעות שלהם. כאשר המשתמש מחליט כי אינו רוצה לחפש יותר במילון הוא יקליד את בקשת היציאה ואז המערכת תנקה ותשחרר את הזיכרון בצורה מסודרת(!) לפני היציאה.

פירוט השלבים

שלב קבלת הנתונים:

בהתחלה לא ידוע למערכת כמה מילים יוכנסו למילון ולכן יש במערכת מצביע מסוג ***char.
שלב ראשוני יהיה לקבל מהמשתמש את מספר המילים שהוא מעוניין להכניס למילון כקלט של מספר שלם. בהתאם לקלט המערכת תיצור את המערך הראשוני (הקצאה דינמית!).

כעת יש לקלוט מהמשתמש כל "כניסה" במילון בפני עצמה.
בתחילת כל "כניסה" יזין המשתמש כמה הגדרות הולכות להיות למילה הנוכחית. בהתאם לכך יוקצה התא הנוכחי של המערך (שהוא מצביע מסוג **char) למערך של מחרוזות באורך המתאים (להחזקת מילה + כל ההגדרות שלה).

ואז בתא הראשון של המערך החדש (כל תא מסוג *char) תוכנס המילה ובכל שאר התאים יוכנסו ההגדרות.

המשתמש יזין את הנתונים כך:

- המילה עצמה תוזן כרצף של לא יותר מ-80 תווים שלא כולל בתוכו רווחים ולאחר מכן יקיש המשתמש אנטר (ENTER).
- כעת יקיש בזו אחר זאת את ההגדרות של המילה (רצף של לא יותר מ-200 תווים שכן עלול להכיל רווחים) ובסוף כל ההגדרה תהיה ירידת שורה (ENTER).

יש לוודא שאחסון המילים וההגדרות יהיה יעיל מבחינת מקום (זאת אומרת שיוקצה בדיוק הגודל הדרוש ולא יהיה בזבוז).

שימו לב: לכל מילה יכול להיות מספר אחר לגמרי של הגדרות. ולכן אתם חייבים לשמור לכם כמה הגדרות יש לכל מילה! (הן בשביל הבניה, העבודה עם מבנה הנתונים ושחרור הזיכרון). **חישבו היטב כיצד לשמור נתון זה עבור כל מילה!**

<u>שלב בניית המילון:</u>

לאחר שהמשתמש הכניס את כל המילים המערכת רוצה לבנות את המילון. לשם כך יש:

1. לתקן את הכתיב:

a. במילים – כל מילה תתוקן כך שהאות הראשונה שלה תהיה uppercase וכל שאר האותיות יהיו lowercase. **שימו לב כי חובה שהפונקציה המתקנת את המילה תהיה רקורסיבית!**

b. בהגדרות - כך שהאות הראשונה במשפט וכן אות ראשונה בכל מילה המופיעה לאחר נקודה (זאת אומרת שיש נקודה, רווח ואז את התו שיש לשנות ל-uppercase) תתחיל באות uppercase וכל שאר האותיות בהגדרה יהיו lowercase. **גם כאן חובה להשתמש בפונקציה רקורסיבית!**

שימו לב: מובטח לכם כי כל תו המתחיל מילה או משפט יהיה באמת תו שהוא אות באלף-בית האנגלי. לגבי שאר התווים ייתכן שיהיו אותיות (ואז יש לוודא שהם lowercase) וייתכן שיהיו תווים אחרים – ואז אין לבצע בהם דבר.

2. למיין את כל המילים בסדר לקסיקוגרפי (A, B, C…).

שימו לב למיין את המילון בצורה היעילה ביותר האפשרית ותוך שימוש באלגוריתם רקורסיבי.

3. להיפטר ממילים זהות המופיעות יותר מפעם אחת (אפילו אם ההגדרות שלהם שונות!) יש לקחת רק את המופע הראשון של מילה (הפעם הראשונה שבה הוכנסה).

לאחר המיון והצמצום של המילים הכפולות – יש לוודא שוב כי הקצאת המילון לא מבזבזת מקום במערכת (זאת אומרת שאם קודם הוקצו יותר תאים במערך הראשי משהיה צריך כי חלק מהמילים היו כפולות – עכשיו צריך לתקן זאת...)

<u>שלב החיפוש:</u>

בשלב זה המשתמש יכניס למערכת מילים שהוא מעניין לחפש במילון והמערכת תדפיס תוצאות בהתאם:

- למילה הנמצאת במילון – יודפס למסך כמה הגדרות יש למילה ואז יודפסו ההגדרות השונות בצורה ממוספרת.
- למילה שלא נמצאת במילון – יודפס למסך "Unknown word!"

המשתמש יוכל להמשיך להכניס מילים כרצונו, עד שיכניס את המילה "exit" שמסמנת כי המשתמש רוצה לצאת מהמערכת (מובטח כי exit לא תהיה אחת המילים במילון).

יש לוודא כי חיפוש המילה במילון הוא יחסית יעיל (זאת אומרת שמשום שהמילון ממוין בהכרח אין חובה לחפש בכל המילון, אלא ניתן לחפש בדילוגים תוך מעבר חלקי בלבד על המערך) **שימו לב לממש את החיפוש תוך שימוש באלגוריתם רקורסיבי.**

<u>שלב היציאה:</u>

חובה לוודא יציאה מסודרת תוך שחרור כל הזיכרון המוקצה דינמית!

<u>סעיף בונוס (מסלול סייבר)</u>

יש לממש את האלגוריתמי המיון והחיפוש במילון באופן ג'נרי (כללי) – זאת אומרת בלי להתחייב על סוג הנתונים במבנה הנתונים, ותוך שימוש במצביעים לפונקציות.

הערות:

1. שימו לב שכאשר הקלט מהמשתמש הוא מסוג מסוים או מבנה מסוים, אתם יכולים להניח שאנו אכן נכניס את הסוג\המבנה המתאים. אך לא בהכרח ערכים חוקיים (אינדקסים חוקיים) – באחריותכם לוודא את חוקיות הקלט!
2. אחרי כל הדפסה יש לבצע ירידת שורה ("\n" בסוף המחרוזת שנשלחת ל-prinf).
3. בתרגיל יש להשתמש בספריות stdio, stdlib ו-string **בלבד!**
4. **אין להשתמש בתרגיל בחומר שנלמד לאחר נושא התרגיל אלא אם נכתב במפורש בתרגיל שמותר! (אסור להשתמש במבנים!)**
5. **שימוש בפונקציות רגילות במקום בו נדרשתם להשתמש בפונקציות רקורסיביות יוביל לקבלת אפס על חלק זה בתרגיל – ראו הוזהרתם!**
6. יש להקפיד על תכנות נכון:
   a. כל הערכים שהם קבועים חייבים להיות מוגדרים כ: define, const או enum, בהתאם לצורך, אין זה נכון להשתמש בקבועים מספריים.
   b. יש לרשום הערות בשפע! **ובאנגלית בלבד**.
   c. יש לנסות ולייעל את הקוד והתוכנית ככל שניתן ולהשתמש באלגוריתמים יעילים כל הניתן!
   d. לפני כל בקשת קלט יש להדפיס למשתמש הוראה איזה קלט מבוקש.
   e. יש להקפיד לבצע חלוקה לפונקציות. אין לכתוב קוד כפול!
   f. בהקצאה דינמית – חובה לוודא כי ההקצאה הצליחה ולטפל כמו שצריך במידה ולא! (לשחרר את כל הזיכרון שהוקצה עד אותו הרגע ולצאת בצורה מסודרת)!
   g. **יש להקפיד על הזחות!!! וכיתוב נכון וקריא!**
   h. יש להקפיד על כל כללי התכנות הנכון כפי שנלמדו בכיתה.
7. בהצלחה ☺

הנחיות הגשה

- תרגילים הם ביחידים! כל עבודה משותפת (כולל העזרות בקוד של מתגבר או בקוד כתוב מהאינטרנט) אסורה ותיענש בחומרה!
- ההגשה היא של שני קבצים בלבד: קובץ ה**קוד** (כקובץ c.) וקובץ מוקלד (בלבד) לחלק התאורטי. הקבצים ישלחו **אך ורק** דרך המערכת!
  שליחת קבצים אחרים – תוביל לאיפוס ציון התרגיל. שליחת קבצים מיותרים תוביל להפחתת נקודות.
- כל הקוד נכתב בקובץ אחד, אך יחולק להרבה פונקציות. שימו לב לכתוב את הצהרות הפונקציות מעל ה-main ואת המימוש מתחת.

- ההגשה של קובץ הקוד היא הגשה של קובץ מוכן לקימפול והרצה! הבודק לא ישנה דבר בקובץ לפני בדיקתו. כל מי שיגיש קובץ שחלקו בהערה – החלק הנ"ל לא ייבדק לו. כל מי שיגיש קובץ עם שגיאות – ציונו יהיה בהתאם. דאגו לבדוק את הקבצים **לפני השליחה!!!**

- בתחילת **קובץ הקוד חובה** להוסיף את התיעוד הבא:

```
/* Assignment: 4
        Campus: Ashdod / Beer Sheva (תבחרו את המתאים)
        Author: Israel Israeli, ID: 01234567
*/
```

כמובן שיש לעדכן את השמות ומספרי תעודות הזהות שלכם.

- כמו כן **חובה** לציין את שמכם המלא ומספר ת"ז שלכם בתחילת הקובץ המוקלד.
מי שלא יכתוב את שמו ומס' ת"ז שלו בתרגיל – יופחתו 10 נקודות מציונו!

- כל שאלה ופניה בנוגע לתרגיל יש להפנות אך ורק למרצה האחראית על התרגיל – תמר שרוט באימייל tammarm@gmail.com. פניות בדרך אחרת לא יענו!

- ***הארכות יינתנו אך ורק במקרים חריגים (מילואים, אבל על קרובים ומחלה חריפה!) ובצרוף אישורים מתאימים! כמו כן חובה ליצור קשר עם המרצה האחראית על התרגיל לפחות יומיים לפני חלוף הדד-ליין!***

- ההגשה היא עד התאריך האחרון לתרגיל: 18.01 בשעה 23:40. הגשה מאוחרת אפילו בדקה – לא תתקבל (המערכת חוסמת את אפשרויות ההגשה!). קחו זאת בחשבון ותכננו את זמנכם בהתאם!