

**מבוא למדעי המחשב**  
**מטלה 6**  
**נושא: הקיימות דינמיות ומבנה**  
**תרגול אחראי: אביב שוקרנו**

**הוראות כלליות**

המטלה כוללת 3 קבצים:  
mainTrain.c – קובץ ה-*main* שלכם, הוא מרים בדיקות על הפונקציות שתממשו.  
Survivor.h – קובץ הגדרות עבור המבנה Survivor והפונקציות הנחוצות לו.  
Tribe.h – קובץ הגדרות עבור המבנה Tribe והפונקציות הנחוצות לו.

במטלה זו תבקשו למשתמש במבנה נתונים (struct) ע"י מימוש של 12 פונקציות.  
יש להוריד את המטלה מערכות ההגשות האוטומטיות ולהשתמש בקבצי ה-*H* המקוריים על מנת  
שלא יתגלו שגיאות קומpileציה מסיבות כמו שלא רשותם שימוש של פונקציות בצורה נכונה.

עליכם ליצור 2 קבצי *C*:  
Survivor.c – בו תימושו את הפונקציות שיש ב-*h.h*.  
Tribe.c – בו תימושו את הפונקציות שיש ב-*h.h*.

שים לב!

בכל שאלה אתם רשאים להשתמש בקוד שמייחתם מסעיפים קודמים. למשל אם הפונקציה  
משאלת מספר 2 יכולה להשתמש בפונקציה שמייחתם בשאלת 1 אתם רשאים לקרוא לפונקציה  
מ-1 בפונקציה של 2.

לאחר שתממשו, תוכלו לкопל ולהריץ את התכנית עם ה-*main* המקורי או לשנות את ה-*main* כראות  
עונייכם ולבצע עוד בדיקות משלכם.

חשוב!

יש לוודא הרצה של התכנית בסביבה שלכם לפני הגשה למערכות האוטומטיות. בזורה זו תוכלו  
ללמוד ולהתפתח בצורה הטובה ביותר.

בסיום המטלה הנכם נדרשים להגיש **את 2 קבצי ה-C שיצרתם - Tribe.c-1 Survivor.c** ובهم  
המימושים של השאלות.

לא לשכוח לרשום הערות לאורק הקוד (כל-3 שורות).  
כמו כן הנכם מתבקשים לרשום הערה ארוכה בתחילת הקובץ Tribe.c הכוללת שם, ת.ז. ותאריך. כמו  
כן הנכם מתבקשים לרשום לפני כל פונקציה הערה ובה מה הפונקציה עשו.

ברצוננו למשם מערכת לניהול תכנית הטלויזיה הישרדות.  
לשם כך יש לנו את ההגדירות הבאות:

```
typedef struct Survivor
{
    char* name;           //the name of the survivor
    float age;            //the age of the survivor
    int followers;        //how many followers he has on social networks
} Survivor;

typedef struct Tribe
{
    Survivor ** survivors; //a pointer to an array of pointers to survivors
    int num_of_survivors; //the number of survivors in the tribe
    char* name;           //the name of the tribe
    char* bandana_color; //the color of the tribe's bandana
} Tribe;
```

על מנת להחזיק את מערכת ניהול הנ"ל ברצוננו למשם את הפונקציות הבאות.

.1

```
Survivor* CreateSurvivor( char* _name, float _age, int _followers );
```

הfonקציה מקבלת כפרמטר שם, גיל ומספר עוקבים ומוחזירה שורט חדש (shmokcha DINMIA) כאשר השדות של השורט מאותחלים בהתאם לפרמטרים שקיבל מהfonקציה. לא לשוכח לבצע הקצאות לכל המשתנים שזוקקים לכך, על מנת למשל שם השורט יועתק ולא יציבו לאווטו מקום בזיכרון כמו המקורי. במידה ואחת הקצאות בfonקציה לא מצליחה יש לשחרר את הזיכרון שהוקצה ולא ניתן להשתמש בו (אם יש צזה) ולהחזיר NULL.

.2

```
Survivor* DuplicateSurvivor( Survivor* source );
```

הfonקציה מקבלת כפרמטר מצביע לשורט ומוחזירה שורט חדש (shmokcha DINMIA) בעל נתונים זמינים לשורט שהתקבל כפרמטר. לא לשוכח לבצע הקצאות לכל המשתנים שזוקקים לכך, על מנת למשל שם השורט יועתק ולא יציבו לאווטו מקום בזיכרון כמו המקורי. במידה ואחת הקצאות בfonקציה לא מצליחה יש לשחרר את הזיכרון שהוקצה ולא ניתן להשתמש בו (אם יש צזה) ולהחזיר NULL .

.3

```
Tribe* AddSurvivor( Tribe* t, Survivor* s );
```

הfonקציה מקבלת מצביע לשבט ושורט ומוחזירה מצביע לשבט מעודכן. במידה והמצביע t מצביע ל-NULL, יש ליצור שבט חדש עם השם "Colman", עם בנדנה במצב אדום ("Red") ועם שורט אחד שתאותחל על ידי העתקה מתוכנו של השורט s. לא לשוכח לבצע הקצאות לכל המשתנים שזוקקים לכך, על מנת למשל שם השורט יועתק ולא יציבו לאווטו מקום בזיכרון כמו המקורי. במידה ובשבט כבר יש שורט עם שם זהה, הfonקציה רק תעדכן את כמות העוקבים של השורט הקיים בשבט לפי הכמות שהועבירה ב-s. במידה ושבט אין שורט עם שם זהה, נגידיל את מערך המצביעים של השבט, נוסף שורט חדש ונעתיק את הנתונים מ-s לשורט החדש. במידה וההקצאה הצליחה, יש לעדכן את השדה num\_of\_survivors. במידה ואחת הקצאות בfonקציה לא מצליחה יש לשחרר את הזיכרון שהוקצה ולא ניתן להשתמש בו (אם יש צזה) ולהחזיר NULL .

.4

```
Tribe* DuplicateTribe( Tribe* source );
```

הfonקציה מקבלת מצביע לשבט קיים source ועליה ליצור שבט חדש מאותחל נתונים של source. יש לבצע הקצאה לכל המשתנים ולא העתקת מצביעים, כלומר להקצות לשורדים זיכרון חדש וגם לכל הנתונים הדורשים הקצאה. במידה ואחת הקצאות בfonקציה לא מצליחה יש לשחרר את הזיכרון שהוקצה ולא ניתן להשתמש בו (אם יש צזה) ולהחזיר NULL .

.5

void SortByAge( Tribe\* t );

הfonקציה מקבלת מצביע לשפט t וממיינת את השורדים בשפט לפי גיל מהקטן לגודל.

.6

void SortByName( Tribe\* t );

הfonקציה מקבלת מצביע לשפט t וממיינת את השורדים בשפט לקסיקוגרפית לפי שם השורד מהקטן לגודל.

.7

int TotalFollowers( Tribe\* t );

הfonקציה מקבלת מצביע לשפט t ומחזירה את סך כל העוקבים מכל השורדים שיש בשפט (עוברת על כל השורדים בשפט וסוכמת את העוקבים שלהם).

.8

int UpdateFollowers( Tribe \* t, char\* name, int toAdd );

הfonקציה מקבלת מצביע לשפט t ושם name. הfonקציה מחפשת את השורד עם השם זהה בשפט. במידה ומוצאת מוסיפה את to לכמה העוקבים הקיימת ומחזירה 1. במידה והשורד לא נמצא, הfonקציה לא תעדכן כלום ותחזיר 0.

.9

int UpdateAge( Tribe \* t, char\* name, float newAge );

הfonקציה מקבלת מצביע לשפט t ושם name. הfonקציה מחפשת את השורד עם השם זהה בשפט. במידה ומוצאת מעדכנת את גיל השורד ל-newAge ומחזירה 1. במידה והשורד לא נמצא, הfonקציה לא תעדכן כלום ותחזיר 0.

.10

int GetSurvivorFollowers( Tribe\* t, char\* name );

הfonקציה מקבלת מצביע לשפט t ושם name. הfonקציה מחפשת את השורד עם השם זהה בשפט. במידה ומוצאת מהזירה את כמה העוקבים של השורד, במידה והשורד לא נמצא, הfonקציה תחזיר -1.

.11

void FreeSurvivor( Survivor\* s );

הfonקציה מקבלת מצביע לשורד ומשחררת את הזיכרון שהשורד משתמש בו.

.12

void FreeTribe( Tribe\* t );

הfonקציה מקבלת מצביע לשפט ומשחררת את כל נתוני השפט בצורה عمוקה (יש להשתמש בfonקציה מסעיף 11).