

מבוא למדעי המחשב

מטלה 6

נושא: הקצאות דינמיות ומבנים

מתרגל אחראי: אביב שוקרון

הוראות כלליות

המטלה כוללת 3 קבצים:

mainTrain.c – קובץ ה-main שלכם, הוא מריץ 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 תתבקשו לממש מבנה נתונים (struct) ע"י מימוש של 12 פונקציות.
יש להוריד את המטלה ממערכת ההגשות האוטומטית ולהשתמש בקבצי ה-H המקוריים על מנת שלא יתגלו שגיאות קומפילציה מסיבות כמו שלא רשמתם שמות של פונקציות בצורה נכונה.

עליכם לייצר 2 קבצי C:

Survivor.c – בו תממשו את הפונקציות שיש ב-Survivor.h
Tribe.c – בו תממשו את הפונקציות שיש ב-Tribe.h

שימו לב!

בכל שאלה אתם רשאים להשתמש בקוד שמישתם מסעיפים קודמים. למשל אם הפונקציה משאלה מספר 2 יכולה להשתמש בפונקציה שמישתם בשאלה 1 אתם רשאים לקרוא לפונקציה מ-1 בפונקציה של 2.

לאחר שתממשו, תוכלו לקמפל ולהריץ את התכנית עם ה-main המוכן או לשנות את ה-main כראות עיניכם ולבצע עוד בדיקות משלכם.

חשוב!

יש לוודא הרצה של התכנית בסביבה שלכם לפני הגשה למערכת האוטומטית. בצורה זו תוכלו ללמוד ולהתפתח בצורה הטובה ביותר.

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

לא לשכוח לרשום הערות לאורך הקוד (כל 2-3 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 עושה.

ברצוננו לממש מערכת לניהול תכנית הטלוויזיה הישרדות.
לשם כך יש לנו את ההגדרות 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 הניהול הנ"ל ברצוננו לממש את הפונקציות הבאות.

1.

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

הפונקציה מקבלת כפרמטר שם, גיל ומספר עוקבים ומחזירה שורד חדש (שמוקצה דינמית) כאשר השדות של השורד מאותחלים בהתאמה בפרמטרים שקיבל מהפונקציה. לא לשכוח לבצע הקצאות לכל המשתנים שזקוקים לכך, על מנת למשל ששם השורד יועתק ולא יצביע לאותו מקום בזיכרון כמו המקורי. במידה ואחת ההקצאות בפונקציה לא מצליחה יש לשחרר את הזיכרון שהוקצה ולא ניתן להשתמש בו (אם יש כזה) ולהחזיר NULL.

2.

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

הפונקציה מקבלת כפרמטר מצביע לשורד ומחזירה שורד חדש (שמוקצה דינמית) בעל נתונים זהים לשורד שהתקבל כפרמטר. לא לשכוח לבצע הקצאות לכל המשתנים שזקוקים לכך, על מנת למשל ששם השורד יועתק ולא יצביע לאותו מקום בזיכרון כמו המקורי. במידה ואחת ההקצאות בפונקציה לא מצליחה יש לשחרר את הזיכרון שהוקצה ולא ניתן להשתמש בו (אם יש כזה) ולהחזיר NULL.

3.

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

הפונקציה מקבלת מצביע לשבט ושורד ומחזירה מצביע לשבט מעודכן. במידה והמצביע t מצביע ל-NULL, יש ליצור שבט חדש עם השם "Colman", עם בנדנה בצבע אדום ("Red") ועם שורד אחת שתאוחל על ידי העתקה מתוכנו של השורד s. לא לשכוח לבצע הקצאות לכל המשתנים שזקוקים לכך, על מנת למשל ששם השורד יועתק ולא יצביע לאותו מקום בזיכרון כמו המקורי. במידה ובשבט כבר יש שורד עם שם זהה, הפונקציה רק תעדכן את כמות העוקבים של השורד הקיים בשבט לפי הכמות שהועברה ב-s. במידה ובשבט אין שורד עם שם כזה, נגדיל את מערך המצביעים של השבט, נוסיף שורד חדש ונעתיק את הנתונים מ-s לשורד החדש. במידה וההקצאה הצליחה, יש לעדכן את השדה num_of_survivors. במידה ואחת ההקצאות בפונקציה לא מצליחה יש לשחרר את הזיכרון שהוקצה ולא ניתן להשתמש בו (אם יש כזה) ולהחזיר NULL.

4.

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

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

5. `void SortByAge(Tribe* t);`
הפונקציה מקבלת מצביע לשבט t וממיינת את השורדים בשבט לפי הגיל מהקטן לגדול.
6. `void SortByName(Tribe * t);`
הפונקציה מקבלת מצביע לשבט t וממיינת את השורדים בשבט לקסיקוגרפית לפי שם השורד מהקטן לגדול.
7. `int TotalFollowers(Tribe* t);`
הפונקציה מקבלת מצביע לשבט t ומחזירה את סך כל העוקבים מכל השורדים שיש בשבט (עוברת על כל השורדים בשבט וסוכמת את העוקבים שלהם).
8. `int UpdateFollowers(Tribe * t, char* name, int toAdd);`
הפונקציה מקבלת מצביע לשבט t ושם name. הפונקציה מחפשת את השורד עם השם הזה בשבט. במידה ומוצאת מוסיפה את toAdd לכמות העוקבים הקיימת ומחזירה 1. במידה והשורד לא נמצא, הפונקציה לא תעדכן כלום ותחזיר 0.
9. `int UpdateAge(Tribe * t, char* name, float newAge);`
הפונקציה מקבלת מצביע לשבט t ושם name. הפונקציה מחפשת את השורד עם השם הזה בשבט. במידה ומוצאת מעדכנת את גיל השורד ל-newAge ומחזירה 1. במידה והשורד לא נמצא, הפונקציה לא תעדכן כלום ותחזיר 0.
10. `int GetSurvivorFollowers(Tribe* t, char* name);`
הפונקציה מקבלת מצביע לשבט t ושם name. הפונקציה מחפשת את השורד עם השם הזה בשבט. במידה ומוצאת מחזירה את כמות העוקבים של השורד, במידה והשורד לא נמצא, הפונקציה תחזיר -1.
11. `void FreeSurvivor(Survivor* s);`
הפונקציה מקבלת מצביע לשורד ומשחררת את הזיכרון שהשורד משתמש בו.
12. `void FreeTribe(Tribe* t);`
הפונקציה מקבלת מצביע לשבט ומשחררת את כל נתוני השבט בצורה עמוקה (יש להשתמש בפונקציה מסעיף 11).