**פרויקט מסכם בקורס נושאים מתקדמים בשפת C**

**המחלקות להנדסת חשמל ומחשבים**

**סמסטר אביב, תשפ"ד**

## מבוא

עליכם לפתח מערכת לניהול מטופלים בחדר מיון בבית חולים מרכזי.

המערכת תפותח בשפת C ותשתמש במבני נתונים אבסטרקטיים כגון עצים, תור ומחסנית. המערכת תשמור נתוני חולים, את ההיסטוריה הרפואית שלהם ותנהל את התור לחדר המיון.

תיאור כללי של המערכת: המערכת תבצע טעינה של נתוני חולים קיימים מקובץ בתחילת התוכנית, תאפשר הוספת חולים חדשים, ביקורים חדשים, הפקת דוחות, ושמירת הנתונים ביציאה מהתוכנית.

## הנחיות כלליות

יש להקפיד על שימוש בשמות משתנים ושמות פונקציות כפי שהוגדרו בעבודה. במידה וישנם משתנים או פונקציות ששמם לא הוגדר, ניתן להגדירם כרצונכם. **אין להשתמש במשתנים גלובליים.**

o **שאלות לצוות הקורס** יש להפנות באמצעות קובץ השאלות **בלבד** ולא באופן פרטי – השאלות תתרכזנה בהבנת המשימה. שאלות הקשורות לאופן המימוש או יישום הסטודנט לא יקבלו מענה. יש לוודא כי לא קיימת שאלה דומה בקובץ לפני כתיבת השאלה חדשה.

https://docs.google.com/spreadsheets/d/1boRexFQxxrEURxmk0WZtsil4k7yVLtZDVmt9WFyyQ8U/edit?usp=sharing

# תיאור המערכת

**פעולות טעינה ושמירה בעת פתיחת המערכת וסגירתה**

בעת פתיחת המערכת יטענו נתונים מתוך הקבצים שסופקו לכם:

1. מאגר הרופאים **Doctors.txt** – **לרשימה מקושרת** חד כיוונית. מספר הרישיון הינו חד-חד ערכי.
2. מאגר המטופלים **Patients.txt** – יטען **לעץ חיפוש בינארי** (מסודר לפי מספר תעודת זהות). יש להניח שכל מטופל שטופל במיון בעבר, מטופל כעת או נמצא בתור לטיפול, יכנס למאגר המטופלים.
3. רשימת המטופלים שנמצאים כעת בתור לטיפול במיון **Line.txt** – **לתור queue** (המידע השמור בקובץ התור הינו רק תעודות הזהות של מטופלים, אבל התור עצמו יהיה של מצביעים למטופלים בהתאמה לסדר תעודות הזהות בקובץ כאשר המטופל הממוספר בקובץ ע"י 1 הינו המטופל הראשון בתור).

בעת סגירת המערכת יעודכנו הקבצים הבאים:

1. מאגר המטופלים **Patients.txt** – חולים עדכניים במערכת.
2. הרשימה העדכנית של תעודות הזהות של המטופלים שנמצאים בתור **Line.txt**.
3. מאגר הרופאים **Doctors.txt** עם מספר המטופלים העדכני של כל רופא.

**תיאור כללי של פעולת המערכת**

המערכת תתבסס על תפריט ממנו תיבחר הפעולה המתבקשת ע"י המשתמש.

במהלך האינטראקציה עם המשתמש, כל הקלטים יבדקו, ואם הקלט אינו תקין, תוצג הודעה מתאימה למשתמש שתאפשר לו להכניס את הקלט מחדש או תאפשר לחזור לתפריט הראשי לבחירת פעולה חדשה.

יש לדאוג לאתחל משתנים.

על הודעות השגיאה להיות אינפורמטיביות ולהנחות את המשתמש כיצד להזין קלט תקין.

# דרישות מימוש המערכת

## הגדרת מבנים

**מבנה של תאריך – Date** יכלול את השדות הבאים:

- **Year:** שנה - מספר שלם חיובי בן 4 ספרות - שנה תקינה = החל משנת 1900 ועד השנה הנוכחית
- **Month**: חודש - מספר שלם חיובי, בין 1-12
- **Day**: יום - (מספר שלם חיובי).
- **Hour**: שעה – בפורמט של 24 שעות, מספר שלם בין 0 ל-23
- **Min**: דקות – מספר שלם בין 0-59

**מבנה של רופא – Doc** יכלול את השדות הבאים:

- **Name:** שם ושם משפחה (מצביע למחרוזת דינאמית) – יש לאפשר קליטה של אותיות ורווחים בלבד.
- **nLicense:** מספר רישיון (מחרוזת בת 7 ספרות הכוללת '\0') – תכיל ספרות בלבד והמספר יהיה חד-חד ערכי
- **nPatients:** מספר מטופלים בהם מטפל הרופא במיון. המספר המקסימאלי שווה ל-4.

**מבנה ביקור – Visit** יכלול את השדות הבאים:

- **tArrival:** תאריך הגעה כולל שעת הגעה – מטיפוס Date (יש להשתמש ב- time.h לקליטת זמן נוכחי)
- **tDismissed:** תאריך שחרור מהמיון כולל שעת היציאה – מטיפוס Date. יתקבל בקלט מהמשתמש. אם עדיין שוהה במיון, כל השדות יאותחלו ל- 1-.
- **Duration:** משך השהייה במיון ביחידות של שעות ודקות (float) (ראו דוגמא בקובץ Patients.txt). אם המטופל עדיין שוהה במיון, כל השדות יאותחלו ל- 1-.
- **Doctor:** מצביע למבנה של הרופא המטפל מטיפוס Doc
- **vSummary:** סיכום ביקור – מצביע למחרוזת דינאמית. אם אין סיכום, המצביע מאותחל ל-NULL

**מבנה של מטופל – Patient** יכלול את השדות הבאים:

- **Name:** שם ושם משפחה– בזמן קבלת קלט מהמשתמש יש לאפשר קליטה של אותיות ורווחים בלבד
- **ID:** תעודת זהות (מחרוזת בת 10 תווים כולל '\0')
- **Allergies:** char המקודד bitwise 7 סוגי אלרגיות + NONE. הגדרת הייצוג המספרי של האלרגיות עם האפשרויות הבאות:

  NONE = 0000 0000 (אין אלרגיות למטופל)
  PENICILLIN = 0000 0001

SULFA = 0000 0010
OPIOIDS = 0000 0100
ANESTHETICS = 0000 1000
EGGS = 0001 0000
LATEX = 0010 0000
PRESERVATIVES = 0100 0000

לאדם מסוים **יכולות להיות מספר אלרגיות שונות**. למשל, עבור אדם עם אלרגיה ללטקס ולפניצילין יישמר בשדה Allergies המספר הבינארי הבא: 0001 0010

- **Visits:** מצביע **למחסנית דינאמית** המשמשת לשמירת מידע על ביקורי המטופל בעבר ובהווה
- **nVisits:** מספר הביקורים של החולה במיון

**מבנה של עלה בעץ החיפוש הבינארי המאחסן את נתוני המטופלים - pInTree:**

- **tpatient**: מטופל מטיפוס **Patient**
- מצביע למטופל **right**
- מצביע למטופל **left**

**מבנה של מטופל בתור למיון - pInLine:**

- **lpatient:** מצביע למבנה של מטופל מטיפוס **Patient**
- **next**: מצביע למטופל הבא בתור

**יש להתייחס אל תור המטופלים, ואל עץ החיפוש של המטופלים כאל ADTs.** לכן, יש להגדיר מבנה מנהל לעץ החיפוש בו ישמרו כלל המטופלים **(pTree)**, לתור המטופלים במיון **(pLine)** ולבנות קבצים ו-headers מתאימים לטיפול במבני נתונים אלה עם השמות הבאים בהתאמה: ptree.h, pline.h.

## פונקציות למימוש

יש לבנות תוכנית מודולרית וקריאה, לכן יש לפרק את הפעולות השונות לתתי פונקציות במידה וישנו צורך.

יש להשתמש בסביבה מרובת קבצים למימוש מבני הנתונים השונים.

## פונקציות עזר שהן חובה:

**loadPatients** – טעינה של המטופלים הקיימים מתוך קובץ Patients.txt **לעץ חיפוש בינארי** לפי מספר תעודת זהות. שימו לב כי הקובץ Patients.txt מכיל גם את הביקורים של המטופל אותם צריך גם לטעון.

**loadDoctors** – טעינה של הרופאים הקיימים מתוך קובץ Doctors.txt לתוך **רשימה מקושרת**.

**loadLine** – יצירת **תור** על בסיס הנתונים שנמצאים בקובץ Line.txt. התור יאפשר גישה ישירה לכל המידע של כל מטופל בתור ולא רק למספרי תעודות הזהות.

**updateFiles** – שמירת החולים לקובץ Patients.txt בעת הצורך או בזמן יציאה מהתוכנית. אם היו חולים בתור אז יש לעדכן גם את קובץ Line.txt ולעדכן את Doctors.txt בהם מופיע מספר המטופלים שיש לכל רופא באותו הרגע.

**searchPatient** – הפונקציה מקבלת תעודת זהות בעץ חיפוש בינארי ומחזירה מצביע למטופל עם תעודת זהות זו

**displayError** – פונקציה להדפסת הודעות שגיאה – מקבלת קוד שגיאה ומדפיסה הודעה מתאימה.

**assignDoctor2case** – הקצאת רופא למטופל. ממלאה את פרטי הרופא המטפל. יש לבחור רופא עם העומס הקטן ביותר מתוך רשימת הרופאים. אם ישנם מספר רופאים עם אותו עומס מינימלי, יש לבחור רופא לפי הסדר ברשימה המקושרת. יש לעדכן את מספר המטופלים אצל הרופא.

## פעולות תפריט:

יש לבנות פונקציית **Menu** המציגה תפריט למשתמש בהתאם לרשימת האפשרויות הבאה:

<u>פעולה 1 בתפריט</u>: **Admit patient** – הוספת מטופל

לפני קליטת מטופל, יש לבדוק אם ישנו רופא פנוי במיון. במידה ואין, אין לקלוט את נתוני המטופל.

במידה ויש רופא פנוי, עליכם לקלוט מהמשתמש תעודת זהות. לבדוק האם המטופל קיים במאגר הנתונים (ניתן להניח שתעודת הזהות הינה מאפיין ייחודי של המטופל). במידה ולא קיים, ואז לקלוט מהמשתמש את כל הפרטים האישיים של המטופל ולבדוק את תקינות הקלט. יש לבדוק את הקלט באופן יעיל, כך שהמשתמש לא יצטרך להקליד את הכל מחדש אם טעה בהקלדת אחד הנתונים. במידה ומדובר במטופל חדש, יש להכניס את המטופל ל-BST לפי מספר תעודת הזהות.

לאחר אימות/ יצירת מטופל חדש, עליכם לקלוט את נתוני הביקור לתוך מחסנית הביקורים של המטופל ולהכניס את המטופל לסוף תור. יש לוודא כי לא קיים ביקור פעיל טרם יצירת ביקור חדש.

יש לקלוט את שם הרופא המטפל בעזרת פונקציית עזר **assignDoctor2case** ולעדכן את מספר החולים בהם מטפל הרופא.

<u>פעולה 2 בתפריט:</u> **Check for patient's allergies** – יש להציג את רשימת המטופלים הנמצאים בתור (תעודת זהות ושם), לקלוט מהמשתמש את תעודת הזהות של המטופל ולהציג את רשימת האלרגיות שלו.

<u>פעולה 3 בתפריט:</u> **Display all patients** – יש להדפיס את הנתונים של כל המטופלים בעץ החיפוש הבינארי (תעודות זהות ושמות).

<u>פעולה 4 בתפריט:</u> **Display all patient's admissions** – יש לקלוט מהמשתמש תעודת זהות של מטופל ולהדפיס את כל ביקורי המטופל בסדר כרונולוגי הפוך – מהחדש ועד הישן ביותר.

<u>פעולה 5 בתפריט:</u> **Display all patients in line** – יש להדפיס את הנתונים האישיים ונתוני הביקור הנוכחי של המטופלים שנמצאים בתור.

<u>פעולה 6 בתפריט:</u> **Advance patient in line** – יש להדפיס את נתוני המטופלים שנמצאים בתור (תעודת זהות ושם), לאחר מכן לקלוט ממשתמש את תעודת הזהות של המטופל אותו רוצים לקדם ולהזיזו לראש התור. יש להדפיס את נתוני המטופלים בתור לאחר עדכון התור.

<u>פעולה 7 בתפריט:</u> **Display list of doctors** – להציג את כל הרופאים (שם של רופא) וכמה מטופלים משויכים לכל רופא.

<u>פעולה 8 בתפריט:</u> **Display all patients assigned to a doctor** – יש לקלוט מהמשתמש שם של רופא ולהציג את הנתונים של המטופלים המשויכים אליו (שם, תעודת זהות ומתי נכנסו למיון).

<u>פעולה 9 בתפריט:</u> **Discharge patient** – יש לקלוט מהמשתמש את תעודת הזהות של המטופל, לעדכן את נתוני הביקור של המטופל ששוחרר בכל מבנה נתונים רלוונטי ולהוציאו מהתור. לא לשכוח לעדכן את מספר המטופלים גם אצל הרופא המטפל. יש להדפיס את נתוני המטופלים בתור לאחר שחרור המטופל (שם ותעודת זהות). שימו לב כי אין למחוק מטופל מהמערכת ורק להשלים ביקור פעיל.

<u>פעולה 10 בתפריט:</u> **Remove visit** – יש לקלוט מהמשתמש תעודת זהות של מטופל ואת תאריך הביקור במיון, ולמחוק את הביקור מהמחסנית. יש להדפיס את נתוני הביקורים של המטופל לאחר המחיקה.

<u>פעולה 11 בתפריט:</u> **Remove patient** – יש לקלוט מהמשתמש תעודת זהות של מטופל, למחוק את המטופל ממאגר הנתונים ולעדכן את כל השדות בכל מאגרי הנתונים הרלוונטיים.

<u>פעולה 12 בתפריט:</u> **Close the hospital** – בית החולים נסגר ולכן עליכם למחוק את כל המידע ממערכת הנתונים (שחרור זיכרון), לברך לשלום ולצאת מהמערכת.

<u>פעולה 0 בתפריט:</u> **Exit Program –** יציאה מהמערכת. יש לבצע שמירה של כל המידע העדכני לקבצים הרלוונטיים, לשחרר זיכרון ולצאת מהמערכת.

### הנחיות כלליות ליישום

- לכל טיפוס של נתונים יש להתייחס כאל ADT ולכתוב סט של פונקציות לניהולם שישמרו בספרייה ייעודית: אתחול, הוספה, מחיקה, הדפסה.
- יש לעבוד בסביבה מרובת קבצים
- יש לדאוג לקוד יעיל ליישום הפונקציות
- יש להשתמש בהגדרת קבועים ופקודות מאקרו במידת הצורך
- יש לדאוג לממשק נוח וברור למשתמש העובד עם המערכת
- יש לרשום תיעוד מפורט של הקוד **באנגלית**
- יש לדאוג לשחרר כל זיכרון דינאמי שאינו בשימוש + שחרור מסודר לפני היציאה מהמערכת

דוגמא למעטפת כללית לקוד של התוכנית הראשית:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

// Include your libraries

// Structures and function prototypes go here

int main() {
    // Initialize data structures and other necessary variables

    // Display menu and handle user input
    return 0;
}

// definition of functions goes here
```

## מדדים להערכה של העבודה

ציון העבודה יקבע על פי המדדים הבאים:

**נכונות הקוד** – יש לוודא שהפעולות אותם אתם מבצעים על הנתונים לא פוגעים בנכונות הנתונים וישנו טיפול קפדני בכל השגיאות שעלולות להופיע מבלי שתהיה קריסה של המערכת. יש לוודא שכל פעולה מתבצעת באופן מלא ותקין. דו"ח בדיקות אותו אתם מגישים (ראו הנחיות בסוף) אמור לכסות את כל המקרים האפשריים.

**יעילות הקוד** – יש לכתוב קוד לטיפול יעיל בנתונים בהתאם לסוג מבנה הנתונים איתו אתם עובדים

**מודולריות הקוד** – יש לחלק את הקוד לפונקציות המותאמות לכל סוג של נתונים

**נוחות ממשק משתמש** – יש לספק אפשרויות ברורות למשתמש לאינטראקציה נוחה בניהול בסיס הנתונים

**עקביות המערכת** – יש לדאוג שהנתונים נשמרים ונטענים באופן תקין בין הרצה להרצה של המערכת ולוודא שהמערכת מטפלת בכל המקרים כך שלא תתרחש קריסה בגלל שגיאות מכל סוג שהוא

**תיעוד** – יש להוסיף הערות מלאות וברורות בקוד המסבירות את המטרה והפונקציונליות של כל פעולה ופונקציה באנגלית

**עמידה בכללי ההגשה** – יש להגיש את העבודה על פי כללי ההגשה המופיעים למטה

**קובץ בדיקות** – יש לכלול בתוך הקובץ להגשה בדיקות שבוצעו למערכת, שיכלול צילום של הקלט והפלט בעת הבדיקה

**שימוש ב- ADT וסביבת מרובת קבצים** - יש לעבוד בסביבת מרובת קבצים למימוש מבני הנתונים השונים.

## כללי ההגשה

במהלך התקופה שניתנה לביצוע העבודה, יהיה עליכם להגיש 2 גרסאות של עבודה חלקית בהתאם להנחיות המופיעות באתר הקורס ולהספק שלכם + גרסה סופית של עבודתכם בתאריכים ידועים מראש. הגשות אלה הן חובה. **סטודנטים אשר לא יגישו את קבצי הביניים, יקבלו ציון נכשל בעבודה גם אם גרסתם הסופית תהיה מושלמת.**

יש לשים לב להנחיות ההגשה של כל תיבה. תהיה הורדה בניקוד במידה ולא תוגש העבודה על פי ההנחיות ועם הקבצים המתאימים.

בכל גרסה של עבודה – חלקית או סופית (סה"כ 3 הגשות בתאריכים שונים), יש להגיש לשתי תיבות הגשה:

- **תיבת הגשה 1:** יש להגיש קובץ ZIP או RAR המכיל את התוכנית במלואה (ספריית הפרויקט ב-VS), כולל כל הקבצים והספריות שיצרתם + קבצי הטקסט עם הנתונים בהם השתמשתם (Patients.txt, Doctors.txt ו- Line.txt), ושידרשו להרצה תקינה של התוכנית. אנא מקמו את הקבצים בספריית הפרויקט על מנת לאפשר הרצה תקינה.
- **תיבת הגשה 2:** יש להגיש קובץ WORD בו יופיעו קבצי הקוד במלואם, באופן ברור וקריא (חשבו על הבודק/ת ונוחות הבדיקה. עבודה נוחה לבדיקה = בודק/ת נחמד/ה יותר). בסוף קובץ WORD, יש לצרף את דו"ח הבדיקות של הקוד. הדו"ח יכיל פירוט מילולי של הבדיקות שבוצעו + צילומי מסך של הקלט והפלט הרלוונטיים עבור כל בדיקה (גם פלט לקבצים). על הבדיקות להיות כמה שיותר מקיפות. בנוסף יש לצרף שרטוט סכמטי המתאר את מבני הנתונים שקיימים במערכת ואת ההקשרים ביניהם.

**הערות חשובות:**

אנו מעודדים אתכם להרחיב את העבודה בהתאם לסקרנות והצרכים שלכם.

אנו מעודדים אתכם להתייעץ עם חברים לכיתה, לחלוק ביניכם רעיונות ולהתייעץ בפתרון בעיות, אך כתיבת העבודה הינה אישית וקטעי קוד זהים אינם מקובלים.

**אין להעתיק עבודות או להיעזר באמצעים חיצוניים כגון תשלום לגורם אחר. שימו לב שייתכן מאוד שתתבקשו להגן על עבודתכם.**

**סטודנטים שעבודתם תיחשד כמועתקת או שלא בוצעה על ידם, גם אם באופן חלקי, יעמדו לדין משמעתי והקורס יפסל.**