הנדסת תוכנה
שנקר
מהנד0ים טאלנטים בת1כנה

מבוא לתכנות מערכות
תרגיל בית מספר 4

מועד פרסום: 28.5.2023
מועד הגשה: 11.6.2023
מתרגל אחראי לתרגיל: **דוד עבדת** davidavdat@shenkar.ac.il
קבוצת דיסקורד: https://discord.gg/bynatp2k

1 הערות כלליות

- **שימו לב! לא יינתנו דחיות במועד הגשת התרגיל. תכננו את זמנכם בהתאם**
- בכל שאלה בנוגע לתרגיל הבית יש לפנות לדוד (ולא למתרגל אחר). מומלץ ורצוי לפנות בקבוצת הדיסקורד. **בפניות באמצעות דוא"ל יש לכתוב בשורת הנושא (Subject): תכנות 2 תרגיל 4**
- מומלץ לקרוא ולהבין את כל ההנחיות לתרגיל לפני תחילת הפתרון. שאלות אשר התשובה עליהן מופיעה בגיליון התרגיל לא תיענינה!
- הגשת התרגיל תתבצע **בזוגות.**
- התרגיל מנוסח בלשון זכר אך מכוון לכל המינים.

2 נושא התרגיל – שדרוג מערכת אחסון נתוני התקליטים שפיתחתם

3 מטרת התרגיל

תכנון, כתיבה ומימוש של ADT. עבודה עם ממשקי ADT אשר מימושם לא נתון. תכנון וכתיבת ADT המשתמשים ב – ADTs אחרים (Nested ADTs). שימוש ב – ADTs גנריים. שימוש במצביעים לפונקציות.

4 חלק א (10%)

4.1 סעיף א

ראינו שקיימים טיפוסי נתונים מופשטים ADTs שונים המאפשרים אחסון של נתונים מסוגים שונים למשל:

1. מחסנית –מאפשר הכנסת נתונים והוצאתם בסדר הפוך לזה שהוכנסו
2. רשימה מקושרת – מאפשר נתונים על פי סדר הכנסתם
3. קבוצה – שמירה של נתונים ללא חשיבות לסדר וללא כפילויות

הנכם נדרשים להציע מבנה נתונים מתאים עבור כל אחת מהבעיות המופיעות מטה. אנא נמקו תשובתכם בקצרה

א. רשת החנויות סי-מודול מעוניינת לעקוב אחר הרכישות שביצעו לקוחותיה. לשם כך היא זקוקה למבנה נתונים בו תוכל לשמור פרטיהם של הלקוחות. כל לקוח יזוהה על פי מספר הזהות.
ב. חברת העיתונים ידיעות-שיא מחשבת את הכנסותיה על פי דירוג מחיר של כל כתב עת. לשם חישובי הסטטיסטיקה (ממוצע, חציון וסטיית תקן) מעוניינים כלכלני החברה במבנה נתונים שיאפשר דירוג של כתבי העת על פי מחיר

ג. חברת שקרניקים בע"מ מפתחת רובוט שיעזור במטלות הבית. החברה מעוניינת שהרובוט יבצע את המטלות בצורה פשוטה אחת-אחת וידווח על הצלחה או כשלון לתוך מבנה נתונים.
המפתחים דורשים הצגת תוצאות המטלות מהאחרונה ועד לראשונה (בסדר הפוך לביצוען)

ד. ענקית הקמעונאות המקוונת סופר-סי מעוניינת לנתח את פעילויות הלקוחות. החברה מעוניינת לעקוב אחר הרכישות שביצעו הלקוחות ולהתאים לכל לקוח פרופיל צרכני באמצעות AI ובאופן שיוכל להציע ללקוח לרכוש מוצרים שאולי הוא שכח בהתאם לסדר הקניות שלו בחודש האחרון. לקוחות יכולים ועשויים לרכוש את אותו המוצר מספר פעמים. לרשת חשוב לדעת עובדה זו בחישוביה.

4.2 סעיף ב

בתרגילי הבית אתם נדרשים לעיתי לכתוב תפריט על מנת שהמשתמש יוכל לבחור את האפשרויות השונות בתוכנית שלכם.
דניאלה הציעה לכתוב ADT גנרי עבור תפריט שיכול להכיל פעולות שונות עבור תרגילי בית שונים.
היא מיד ניגשה למימוש ADT מסוג **Menu**. להלן קטע מהקוד שלה.

file: **menu.h**

```
typedef struct menu_s *Menu;

// ...
MenuResult menu_add_action(Menu m, const char *name, MenuAction action);
MenuResult menu_remove_action(Menu m, const char *name);
LinkedList menu_get_actions(Menu m);
void menu_print(Menu m);

// ...
```

file: **menu.c**

```
struct menu_s {
  LinkedList actions;
  // ...
};

LinkedList menu_get_actions(Menu m) {
  return m->actions;
}

// Dany-Ella <3
```

איזה עקרון הנלמד בקורס הפרה דניאלה במימוש של **menu_get_actions()**? עזרו לה לממש את הפונקציה תוך שמירה על עקרון זה!

5 חלק ב (90%)

5.1 תיאור התרגיל

עליכם לשפר ולהרחיב את מערכת אחסון התרגילים שמימשתם בתרגיל בית מספר 1. השינויים יהיו בעיקר באיכות הקוד כגון:

- יציבות המערכת – הקפידו על דווח וטיפול בשגיאות מוקדם ככל האפשר על מנת למזער את הנזקים שייגרמו
- קריאות הקוד – על מנת להקל על ההבנה של התוכנית שלכם, הקפידו על מתן שמות משמעותיים למשתנים ופונקציות, המנעו מכתיבת פונקציות ארוכות, המנעו משימוש בקבועים ללא שמות והמנעו משימוש במשתנים גלובליים
- מודולריות – הקפידו על חלוקת הקוד למודולים במבנה הגיוני, כך שפונקציות ומבני נתונים קשורים יהיו באותו מודול.
- שימוש חוזר בקוד – בצעו שימוש חוזר בקוד ככל הניתן והמנעו משכפול קוד. מסופקים לכם עם תרגיל הבית שני מודולים בהם עליכם להשתמש במהלך הפיתוח. המודולים מממשים את ה-**ADTs** הבאים: רשימה מקושרת (**LinkedList**) וקבוצה (**Set**). המודולים כוללים קבצי כותרת (**.h**) וקובץ ספרייה **libprog2.a** בלבד. ללא קבצי "**.c**". כלומר – המימוש של המודולים הללו אינו נתון לכם.
- עבור ה- **LinkedList ADT**, קיימת דוגמה לשימוש בתיקייה הנקראת **people_example**. ניתן להתרשם כיצד ניתן להשתמש בצורה נכונה ב **ADT** הזה.
- שימו לב! אין לממש רשימות מקושרות בתרגיל זה. עליכם להשתמש במודולים המסופקים בלבד.

5.2 תיאור המערכת

- עליכם לשכתב את המערכת שכתבתם בתרגיל בית מספר 1. הפעם לא תספקו את פונקציית main, פונקציה זו תתווסף בזמן הבדיקה על ידי הבודקים. עם זאת, לשם בדיקה עצמית של התוכנית שלכם, תוכלו לכתוב פונקציית main משלכם בקובץ נפרד ולא להגיש אותה.
- **בניגוד לתרגיל בית מספר 1, כאן אין הגבלה על כמות התקליטים ו/או כמות הרצועות בכל תקליט**
- מכיוון שהבדיקה תתבצע באמצעות קוד, אין לנתח או לקבל קלט מהמשתמש.
- על המערכת להיות ממומשת בתצורת ADT ולממש את כל הפונקציות המופיעות בממשק records_db.h המסופק לכם. תיאור הפונקציות מופיע בסעיף 5.4 "פקודות"

5.3 טיפול בשגיאות

בניגוד לתרגיל בית מספר 1, בתרגיל זה אינכם מדפיסים הודעות שגיאה אלא נדרשים להחזיר את השגיאה המתאימה בערך החזרה של הפונקציות במקרה והתרחשה.

אם מופיעות כמה שגיאות יש לדווח על השגיאה החשובה ביותר. סדר החשיבות מוגדר בהגדרת הטיפוס recordsResult הנמצא בקובץ pr2ex4.h.

השגיאות הבאות אינן מפורטות מטה בכל פקודה אך רלוונטיות לכל הפקודות:

- במקרה שאחד או יותר מהפרמטרים המתקבלים הוא *NULL*, יש לדווח על השגיאה *RDB_NULL_ARGUMENT*.
- במקרה של כשלון בהקצאת זיכרון, יש לדווח על השגיאה *RDB_OUT_OF_MEMORY*, בניגוד לתרגיל בית מספר 1, **אין לשחרר משאבים שהוקצו במהלך ריצת התוכנית ואין לצאת מהתוכנית**.

5.4 פקודות

על המערכת לממש את הפונקציות הבאות (מוגדרות גם ב recods_db.h)

5.5.1 יצירת מערכת אחסון תקליטים חדשה

```
RecordsDB recordsDbCreate();
```

תיאור הפעולה: פעולה זו יוצרת מערכת אחסון נתונים.

פרמטרים:

- אין

ערכי החזרה:

- records_db מצביע למבנה נתונים חדש ומאותחל במקרה של הצלחה
- *NULL* במקרה של כשלון

5.5.2 שחרור מערכת אחסון התקליטים

```
void recordsDbDestroy(RecordsDB records_db);
```

תיאור הפעולה: שחרור כל משאבי המערכת שהוקצו בזמן הריצה ולא שוחררו עדיין.

פרמטרים:

- records_db מצביע למבנה אותו יש לשחרר

ערכי החזרה:

- אין

5.5.3 הוספת תקליט למערכת

```
RecordsResult recordsDbAddRecord(RecordsDB records_db, const char *name, unsigned
int year, RecordsCategory category);
```

תיאור הפעולה: פעולה זו מוסיפה תקליט למאגר הנתונים. שימו לב שאין הגבלה על כמות התקליטים והרצועות בכל תקליט!

פרמטרים:

- records_db מצביע למבנה הנתונים
- name שם התקליט שנרצה להוסיף
- year שנת ההוצאה של התקליט – גדולה או שווה 1900
- category לאיזו קטגוריה משתייך התקליט. יכולה להכיל ערך מתוך אלו המופיעים בהגדרת הטיפוס `recordsCategory` המופיע בקובץ pr2ex4.h בלבד.

ערכי החזרה:

- *RDB_INVALID_YEAR* - שנת ההוצאה אינה תקינה
- *RDB_INVALID_CATEGORY* - הקטגוריה אינה תקינה
- *RDB_RECORD_ALREADY_EXISTS* - תקליט בעל אותו שם כבר קיים במערכת
- *RDB_SUCCESS* – במקרה של הצלחה

5.5.4 הסרת תקליט מהמערכת

```
RecordsResult recordsDbRemoveRecord (RecordsDB records_db, const char *name);
```

תיאור הפעולה: פעולה זו מסירה תקליט מהמערכת יחד עם כל הרצועות השמורות בו.

פרמטרים:

- name - שם התקליט אותו יש להסיר. שימו לב שזהו ערך ייחודי לתקליט במערכת.

ערכי החזרה:

- *RDB_RECORD_DOESNT_EXIST* - תקליט בעל שם כזה אינו נמצא במערכת
- *RDB_SUCCESS* – במקרה של הצלחה

5.5.5 הוספת רצועה לתקליט

```
RecordsResult recordsDbAddTrackToRecord (RecordsDB records_db, const char
*recordName, const char *trackName, unsigned int trackLength);
```

תיאור הפעולה: הוספת רצועה לתקליט **בסוף רשימת הרצועות.**

פרמטרים:

- records_db מצביע למבנה הנתונים
- recordName - שם התקליט אליו יש להוסיף את הרצועה
- trackName - שם הרצועה אותה יש להוסיף
- trackLength - אורך הרצועה בשניות שלמות – גדול ממש מ-0

ערכי החזרה:

- *RDB_RECORD_DOESNT_EXIST* - תקליט בעל שם כזה אינו נמצא במערכת
- *RDB_INVALID_TRACK_LENGTH* - אורך הרצועה אינו חוקי
- *RDB_TRACK_ALREADY_EXISTS* - רצועה בעלת שם כזה כבר קיימת בתקליט הספציפי הזה
- *RDB_SUCCESS* – במקרה של הצלחה

5.5.6 הסרת רצועה מתקליט

```
RecordsResult recordsDbRemoveTrackFromRecord (RecordsDB records_db, const char
*recordName, const char *trackName);
```

תיאור הפעולה: הסרת רצועה מתקליט אם קיימת.

פרמטרים:

- records_db מצביע למבנה הנתונים
- recordName - שם התקליט אליו יש להוסיף את הרצועה
- trackName - שם הרצועה אותה יש להוסיף

ערכי החזרה:

- *RDB_RECORD_DOESNT_EXIST* - תקליט בעל שם כזה אינו נמצא במערכת
- *RDB_TRACK_DOESNT_EXISTS* - רצועה בעלת שם כזה אינה קיימת בתקליט הספציפי הזה
- *RDB_SUCCESS* – במקרה של הצלחה

5.5.7 דווח על תקליטים

```
RecordsResult recordsDbReportRecords (RecordsDB records_db, RecordsCategory
category);
```

תיאור הפעולה: פקודה זו מדפיסה לערוץ הפלט את כל התקליטים הקיימים במערכת. התקליטים יודפסו על פי סדר לקסיקוגרפי. (יתכן ויהיה עליכם למיין את מבנה הנתונים לפני ההדפסה). ההדפסה של כל תקליט תתבצע באמצעות הפונקציה prog2_report_record המוגדרת בקובץ pr2ex4.h. עבור כל תקליט תודפס רשימת הרצועות שבו באמצעות הפונקציה ()prog2_report_track שגם היא מוגדרת בקובץ pr2ex4.h

פרמטרים:

- records_db מצביע למבנה הנתונים
- category - המערכת תדפיס רק תקליטים מהקטגוריה המבוקשת. על מנת להדפיס את כל התקליטים, יש להשתמש בערך הקבוע *ALL_CATEGORIES* המוגדר בקובץ ההגדרות pr2ex4.h

ערכי החזרה:

- *RDB_INVALID_CATEGORY* - הקטגוריה אינה חוקית
- *RDB_NO_RECORDS* - לא קיימים תקליטים במערכת (או לא קיימים תקליטים מהקטגוריה המבוקשת במידה והפרמטר קיים)
- *RDB_SUCCESS* – במקרה של הצלחה

5.5.8 דווח על הרצועות עבור תקליט מסוים

```
RecordsResult recordsDbReportTracksOfRecord (RecordsDB records_db, const char
*name);
```

תיאור הפעולה: פקודה זו מדפיסה לערוץ הפלט את כל הרצועות הקיימות בתקליט מסוים. ראשית יש להדפיס את פרטי התקליט באמצעות הפונקציה prog2_report_record המוגדרת בקובץ pr2ex4.h ורק אז הרצועות יודפסו על פי סדר ההכנסה שלהן בתקליט. ההדפסה של הרצועות תתבצע באמצעות הפונקציה prog2_report_track המוגדרת בקובץ pr2ex4.h.

פרמטרים:

- records_db מצביע למבנה הנתונים
- name - שם התקליט עבורו יש להדפיס את הרצועות

ערכי החזרה:

- *RDB_RECORD_DOESNT_EXIST* -תקליט בעל שם כזה לא קיים במערכת
- *RDB_NO_TRACKS* - לתקליט הנל אין רצועות
- *RDB_SUCCESS* – במקרה של הצלחה

5.5.9 דווח על כל התקליטים המכילים רצועה מסויימת

```
RecordsResult recordsDbReportContainingRecords (RecordsDB records_db, const char
*name);
```

תיאור הפעולה: פקודה זו מדפיסה את פרטי התקליטים המכילים את הרצועה שבפרמטר (ייתכנו יותר מתקליט אחד- **על הפקודה להדפיס את כולם**). ההדפסה תתבצע באמצעות הפונקציה prog2_report_ record המוגדרת בקובץ pr2ex4.h.

פרמטרים:

- records_db מצביע למבנה הנתונים
- name שם הרצועה לחיפוש

ערכי החזרה:

- *RDB_TRACK_DOESNT_EXIST* - הרצועה אינה נמצאת באף תקליט
- *RDB_NO_RECORDS* - אין כלל תקליטים במאגר
- *RDB_SUCCESS* – במקרה של הצלחה

5.6 הגבלות על המימוש

עליכם לממש את המערכת תוך התחשבות במגבלות הבאות:

- בניגוד לתרגיל בית מספר 1, מספר התקליטים לאחסון במערכת **אינו מוגבל**. שימו לב שעליכם לכלול (#include "pr2ex4.h") את הקובץ הזה בקובץ המימוש שלכם על מנת שתוכלו להשתמש בהגדרות שבו.
- **את כל ההדפסות לפלט יש לבצע באמצעות הפונקציות המתאימות המוגדרות** ב pr2ex4.h
- עליכם להקצות את גודל הזיכרון המתאים עבור על מחרוזת שנשמרת בזיכרון. אין להקצות זיכרון מיותר
- בכל המקרים, ביציאה מהתוכנית, יש לשחרר באופן מפורש את כל המשאבים שהוקצו למערכת.

5.7 הידור, קישור ובדיקה

על המימוש שלכם לעבור הידור בעזרת הפקודה הבאה:

```
gcc -o records_db -std=c99 -Wall -L. -lprog2 -pedantic-errors *.c
```

משמעות הפרמטרים:

- -o records_db זהו שם קובץ ההרצה המהודר
- -std=c99 קביעת תקן לשפה
- -Wall דווח על כל האזהרות בזמן קומפילציה
- -pedantic-errors התייחס לאזהרות כאל שגיאות – משמעות הדגל, שהקוד חייב לעבור הידור ללא אזהרות
- *.c קבצי הקוד אותם נרצה להדר – אלו קבצי הקוד של התרגיל
- -L. -lprog2 קישור עם הספרייה libprog2.a המכילה את מימושי ה ADTs המסופקים. על ספרייה זו להיות באותה התיקייה עם קבצי הקוד שלכם.

מומלץ מאוד! (אך לא חובה) לכתוב גם Makefile שיבנה את התכנית.

התרגיל ייבדק בשני אופנים.
בדיקה אחת כוללת מעבר על הקוד ובודקת את איכות הקוד (אי קיום שכפולי קוד, קוד מבולגן ולא קריא "ספגטי", שימוש בטכניקות קידוד "רעות" וכו').
הבדיקה השנייה כוללת את הידור התוכנית המוגשת והרצתה במגוון בדיקות אוטומטיות. על התוכנית לעבור הידור, לרוץ את הבדיקות בשלמותן ולעבור השוואה של קבצי הפלט עם קבצי פלט מצופים.

על מנת לבדוק את תוכניתכם, **קיימת תיקייה בשם tests ובה מסופקים שני מבדקים כאלו עם הפלט הסטנדרטי והשגיאות המצופים עבור כל אחד**. תוכלו להוריד מהאינטרנט תוכנות השוואה בין קבצים (לדוגמה התוכנה meld או התכנה diff) על מנת להשוות את הפלט שלכם עם הפלט המצופה.
תוכלו להריץ את אותן הבדיקות עבור הקוד שלכם ע"י מיקום התיקייה בתוך תיקיית העבודה שלכם והרצת make מתוך תיקיית הבדיקות. ייווצרו בדיוק אותם הקבצים עם הפלט שלכם.
שימו לב! התוכנית שלכם תיבדק בדיקות נוספות. בדיקות אלו מכסות רק חלק בסיסי מהמקרים האפשריים.

בדיקת התרגיל בעצמכם מהווה חלק מהתרגיל!
- כתבו בדיקות נוספות בעצמכם בהתאם למפרט התרגיל
- וודאו את נכונות התוכנית שלכם במקרים כלליים ובמקרי קצה
- מומלץ לחשוב על הבדיקות כבר בזמן כתיבת הפתרון
- העדיפו ליצור מספר רב של בדיקות על פני בדיקה אחת גדולה שיהיה לכם קשה להתמצא היכן הטעויות
- וודאו נכונות הקוד לפני ההגשה גם לאחר שינויים קטנים ולכאורה "לא מזיקים"
- שימו לב לדליפות זיכרון! הם ייבדקו באופן אוטומטי על ידי תוכנה מתאימה (valgrind)

6 הגשת התרגיל

עליכם להגיש את פתרון חלק א בקובץ נפרד (word, pdf)
בחלק ב עליכם להגיש אך ורק את הקובץ/קבצים שכתבתם כקבציי טקסט. **בפרט אין להגיש את הקבצים המסופקים pr2ex4.h, linked_list.h, set.h, records_db.h, libprog2.a. כאמור, אין להגיש גם את פונקציית main שלכם**

7 דגשים ורמזים

- מומלץ, עוד לפני תחילת כתיבת הקוד, לצייר סכמתית את מבני הנתונים ואת תיאור המערכת כפי שראיתם בהרצאות ובתרגולים.
- אורך רצועה נמדד בשניות שלמות (ולא בחלקי שניות)
- סדר לקסיקוגרפי (מילוני) הוא הסדר המוגדר לפי ערכו המספרי של התו ב ascii ולכן המחרוזת "Aa" קטנה ממש מהמחרוזת "aa". לצורך השוואת מחרוזות בסדר כזה, ניתן להשתמש בפונקציה strcmp
- יש להקפיד על ניהול זיכרון נכון. אין להקצות זיכרון מיותר (למשל, זיכרון באורך `MAX_LEN` עבור כל מחרוזת). כמו כן, יש להקפיד על שחרור של זיכרון כאשר אין בו צורך יותר
- יש להקפיד על תכנון נכון של התוכנית וכתיבה נכונה בשפת C. בשלב זה של לימודיכם הנכם מסוגלים ונדרשים לתכנן ולכתוב תכנית בסדר גודל הנתון בעצמכם. הקפידו לבצע חלוקה נכונה למודולים ולפונקציות בהתאם למבנה הלוגי של התוכנית ולא לכתוב פונקציית main אחת גדולה.
- נצפה לראות הפרדה בין פונקציות הממשות לוגיקה מסוימת לאחרות וכן את המודולים record ו-records_db לכל הפחות. הנכם יכולים לכתוב מודולים נוספים כראות עינכם.
- כדי למנוע בעיות עם הבודק האוטומטי, ערך החזרה של התוכנית (ביציאה מפונקציית main) צריך להיות תמיד 0.
- **יש להקפיד לא להוציא פלט אלא על ידי הפונקציות שסופקו לכם.**
- במקרה של הסתבכות עם באג קשה:
  - בודדו את הבאג ושחזרו אותו באמצעות מספר מינימלי ככל הניתן של פעולות. ניתן לעשות זאת על ידי מחיקת חלק מהשורות בקוד ובדיקה אם הוא עדיין מתרחש.
  - ניתן להשתמש בהדפסות בריצת התוכנית שמציגות את מצב הריצה (ערכי משתנים, מצביעים וכו).
- **חשבו היטב באיזה טיפוס נתונים ראוי להשתמש על מנת לממש כל מודול. בחירה גרועה של טיפוסי נתונים (למשל שימוש ב – Set במקום שבו היה ראוי להשתמש ב – LinkedList) תוביל להורדת נקודות ותקשה עליכם את המימוש.**
- בתרגיל בית זה אין להשתמש במערכים ואין לממש רשימה מקושרת בעצמכם. הנכם נדרשים להשתמש המודולים המסופקים לכם.

בהצלחה! 😊