הנדסת תוכנה
שנקר
מהנד0ים טאלנטים בת1כנה

מבוא לתכנות מערכות

תרגיל בית מספר 1

מועד פרסום: 18.4.2024 בשעה 12:00

מועד הגשה: 16.5.2024 בשעה 23:59

מתרגל אחראי לתרגיל: **דוד עבדת** davidavdat@shenkar.ac.il

קבוצת דיסקורד לשאלות: https://discord.gg/3fWspvRvQu

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 1**
- מומלץ לקרוא ולהבין את כל ההנחיות לתרגיל לפני תחילת הפתרון.
- הגשת התרגיל תתבצע **בזוגות**.
- התרגיל מנוסח בלשון זכר אך מכוון לשני המינים.

2 נושא התרגיל – פיתוח מערכת ניהול משמרות

3 מטרת התרגיל

תכנון וכתיבת קוד המנצל ערוצי קלט ופלט שונים תוך חזרה על נושאים מקורס המבוא

4 חלק א (10%)

כחלק מפיתוח מערכת לניהול משאבי אנוש, נתבקשתם לכתוב קוד הנותן קידום לעובדים מסוימים. על הקוד לקבל מערך של עצמים מטיפוס employee המתואר בהמשך, לקדם את העובדים הזכאים ששכרם עולה על 20,000 ₪ לדרגת "מנהל" . על הפונקציה להחזיר את כמות המנהלים החדש בחברה (כלומר סה"כ מנהלים ישנים + חדשים)

לשם כך, עליכם לבדוק עבור כל עובד המועסק בחברה - האם השכר שלו עולה על 20,000 ש"ח והאם הוא זכאי לקידום (כלומר אחד מהבאים: "Tester", "Developer", "Analyst").

בנוסף, מועבר לפונקציה פרמטר מטיפוס מצביע ל- struct employee_t לתוכו הפונקציה תבצע העתקה של העובד בעל השכר הגבוה ביותר בחברה.

לא ניתן להניח דבר על הקלט מלבד שמערך הקלט הוא חוקי והגודל המסופק אתו (בפרמטר size) אכן מתאר את גודל המערך.

הסטודנט יונתן הציע את הפתרון הבא:

```
typedef struct employee_t
{
    char name[128];
    int id;
    char role[128];
    int salary;
} *employee;

int employees_promotions(employee arr[], size_t size, /*OUT*/ employee expensive)
{
    int mc = 0;
    int ms= 0;
    int i = 0;
    if (size == 0)
        return -1;

/* Promoting employees with high salary :) */
for (i = 0; i < size; i++)
{
employee c = arr[i];
if (c->salary > 20000 && strcmp(c->role, "Developer") == 0)
    strcpy(c->role, "Manager");
    else if (c->salary > 20000 && strcmp(c->role, "Tester") == 0)
        strcpy(c->role, "Manager");
else if (c->salary > 20000 && strcmp(c->role, "Analyst") == 0)
    strcpy(c->role, "Manager");
}

/* Counting managers */
for (i = 0; i < size; i++)
{
employee current = arr[i];
if (strcmp(current->role, "Manager") == 0)
mc++;
}
/* Searching the most expensive worker */
for (i = 0; i < size; i++)
{
employee c = arr[i];
if (c->salary > ms)
ms = c->salary;
*expensive = *c;
}
return mc;
}

// Yona-Tan ><>
```

לאחר מספר הרצות מוצלחות על מספר קלטים שונים, הגיע יונתן למסקנה כי הקוד שכתב עובד נכון משום שהתוצאות שקיבל תאמו לציפיותיו.

4.1 סעיף א
מצאו קלט עבורו הפונקציה נכשלת. הסבירו בקצרה מדוע היא נכשלת והציעו תיקון פשוט כך שהפונקציה תעבוד גם עם קלט זה. (אין צורך לכתוב את הפונקציה מחדש, אלא רק אילו שינויים יש לבצע והיכן).

4.2 סעיף ב
כתבו דוגמה לפונקציית main שמשתמשת בקוד שכתב יונתן **ושאינה** נכשלת.

4.3 סעיף ג
בתעשייה נהוג לבצע "ביקורת קוד" (**Code Review**). זהו תהליך בו עוברים מי שכתב את הקוד יחד עם מפתחים נוספים שטרם ראו את הקוד בסקירה על הקוד. יחד המפתחים המנוסים נותנים את חוות דעתם ומזהים תקלות או בעיות איכות בקוד שחמקו מעיניי המפתח.

בעיות אלו יכולות להיות באגים, תכנות לא איכותי ואף נראות וקריאות הקוד. בעיות אלו עלולות לגרום לעבודה מיותרת או באגים בהתממשקות לקוד אחר שעשוי לכלול הרחבות ושינויים.

בצעו ביקורת קוד לפתרון של יונתן. מצאו 3 בעיות לכל הפחות, פרט לאלו שמצאתם בסעיפים הקודמים, והסבירו בקצרה כיצד ניתן לפתור כל בעיה.

5 חלק ב (90%)

5.1 תיאור המערכת
בעקבות מעבר למערכת ממוחשבת, בעלי רשת הברים וחנויות המשקאות "המלך בבר" מעוניינים במערכת לניהול משמרות של העובדים ברשת. המערכת צריכה להיות מסוגלת לקלוט ולנהל משמרות עבור השבוע הקרוב (ימים א'-ש') כל עובד במערכת יכול לעבוד **משמרת אחת ביום לכל היותר**. כלומר בסה"כ 7 משמרות בשבוע המשמרות עשויות להיות בוקר, אחר הצהריים, ערב ולילה.
לשם כך עליכם לתכנן ולפתח מערכת שתאפשר שמירת נתונים של בעלי תפקידים שונים ומשמרותיהם לשבוע הקרוב.
בנוסף, המערכת תוכל להפיק דוחות על פי בקשה.

- המערכת תופעל באמצעות ממשק טקסטואלי

הפעלת התוכנית
התוכנית, אשר תקרא hr4you תופעל משורת הפקודה באופן הבא:

```
hr4you [-i <input_file>] [-o <output_file>]
```

- שימוש בפרמטר `-i` יגרום למערכת לקבל הוראות מהקובץ ששמו `<input_file>`. אחרת, המערכת תקבל הוראות מערוץ הקלט הסטנדרטי (`standard input`).
- שימוש בפרמטר `-o` יגרום למערכת לשלוח את הפלט לקובץ ששמו `<output_file>`. אחרת, המערכת תשלח את הפלט לערוץ הפלט הסטנדרטי (`standard output`)
- שימו לב ששני הפרמטרים האלו הם רשות. כלומר יכולים להיות 0, 2 או 4 פרמטרים לתוכנית שלכם.
- שימוש ב "`-i`" או ב "`-o`" ללא העברת שמות הקבצים, או העברת ארגומנטים אחרים או נוספים **אינו חוקי!** במקרים אלו, יש לדווח על השגיאה המתאימה באמצעות הפונקציה `prog2_report_error_message` המסופקת לכם.

- שינוי סדר הארגומנטים ("o-" ואחריו "i-") הינו חוקי
- במקרה בו מבנה הפרמטרים לתוכנית אינו תקין יש לדווח על שגיאה מסוג *INVALID_ARGUMENTS*. אם הפרמטרים תקינים אך פתיחת אחד הקבצים נכשלת, יש להחזיר *CANNOT_OPEN_FILE* . דווח על שגיאות מוסבר בפירוט בהמשך
- בכל מקרה אין לשלוח פלט שלא נתבקשתם. כל שליחת פלט תיעשה באמצעות הפונקציות המסופקת לכם.

5.2 הקלט לתוכנית

בין אם מהקלט הסטנדרטי ובין אם מקובץ, הקלט מורכב מסדרת שורות טקסט, כאשר בכל שורה, תהיה פקודה אחת. ייתכנו שורות ריקות שלא יכילו פקודות כלשהן או שורות המכילות הערות

במידה והקלט הוא מערוץ הקלט הסטנדרטי, נסיים את התוכנית על ידי **Ctrl+d** (EOF). אחרת, התוכנית תסתיים ברגע שתקרא את כל השורות מהקובץ.

ניתן להניח כי הקלט תקין מבחינה סינטקטית, כלומר, הפקודות מאויתות נכון, מספר הפרמטרים לכל פקודה תקין, אם הפרמטר הוא מחרוזת אז היא תמיד מהווה קלט חוקי, ואם הפרמטר הוא מספר, ניתן להניח שהוא מחרוזת הניתנת להמרה למספר.

יכולות להיות שורות שהתו הראשון בהן שאינו רווח, הוא '#', אלו הן שורות הערה ואין להתייחס אליהן. במידה והתוכנית קוראת שורה כזו, עליה לעבור לשורה הבאה.

כל שורה בקלט עשויה להיות אחת מהבאות:

1. מכילה פקודה חוקית
2. שורה ריקה
3. שורת הערה שהתו הראשון הוא סולמית - '#' ומיד לאחריו יש רווח

מבנה שורת פקודה:

```
<command> [arg1 arg2 …]
```

כאשר <command> הוא שם הפקודה ו- [arg1 arg2 …] היא רשימת הארגומנטים לפקודה.

כאשר אורך רשימת הארגומנטים יכול להיות שונה מפקודה לפקודה ועשויים להיות ארגומנטים אופציונליים שלא תמיד יופיעו.

בין כל זוג מילים יופיע רווח אחד לכל הפחות אך ייתכנו מספר רווחים. כמו כן ייתכנו מספר רווחים בתחילת השורה ובסופה.

5.3 טיפול בשגיאות

במקרה של שגיאה יש לדווח על סוג השגיאה המתאים בעזרת הפונקציה **prog2_report_error_message** ולהמשיך לפקודות הבאות.

אם מופיעות כמה שגיאות באותה הפקודה, יש לדווח על השגיאה החשובה ביותר. סדר החשיבות מוגדר בהגדרת הטיפוס hr_result הנמצא בקובץ prog2_ex1.h.

5.4 פקודות

על המערכת לממש את הפקודות הבאות

5.4.1 הוספת עובד למערכת

```
Add Worker <name> <id> <hourly wage> <role> <number of shifts>
```

תיאור הפעולה: פעולה זו מוסיפה עובד למערכת

פרמטרים:

- <name> שם העובד שנרצה להוסיף. מחרוזת אחת ללא רווחים
- <id> מספר הזהות של העובד (מספר חיובי גדול ממש מ – 0. ניתן להניח שאין 0 מוביל)
- <hourly wage> השכר השעתי של העובד (מספר עשרוני גדול מ-0. עד שתי ספרות אחרי הנקודה)
- <role> תפקיד העובד ("Bartender", "Waiter", "Manager", "Cashier", "Chef", "Dishwasher"). כל מחרוזת אחרת אינה חוקית!
- <number of shifts> מספר משמרות זמינות לשבוע הקרוב (מספר שלם בין 0 לבין 7 כולל)

שגיאות:

- מספר זהות אינו תקין – יש לדווח שגיאה מסוג `INVALID_WORKER_ID`
- שכר שעתי אינו תקין – יש לדווח שגיאה מסוג `INVALID_WAGE`
- תפקיד אינו תקין – יש לדווח שגיאה מסוג `INVALID_ROLE`
- מספר משמרות אינו תקין – יש לדווח שגיאה מסוג `INVALID_NUM_OF_SHIFTS`
- עובד בעל אותו מספר זהות כבר קיים במערכת – יש לדווח שגיאה מסוג `WORKER_ALREADY_EXISTS`
- מספר העובדים במערכת עובר את המקסימום האפשרי – יש לדווח שגיאה מסוג `WORKERS_OVERFLOW`

5.4.2 הסרת תקליט מהמערכת

```
Remove Worker <id>
```

תיאור הפעולה: פעולה זו מסירה עובד מהמערכת יחד עם כל המשמרות שלו.

פרמטרים:

- <id> מספר הזהות של העובד אותו יש להסיר. שימו לב שזהו ערך ייחודי לעובד במערכת.

שגיאות:

- מספר זהות אינו תקין – יש לדווח שגיאה מסוג `INVALID_WORKER_ID`
- עובד בעל מספר זהות כזה אינו נמצא במערכת – יש לדווח על שגיאה מסוג `WORKER_DOESNT_EXIST`

5.4.3 הוספת משמרת לעובד

```
Add Shift <worker id> <day> <shift>
```

תיאור הפעולה: הוספת משמרת לעובד מסוים בתפקיד מסוים.

פרמטרים:

- <worker id> מספר הזהות של העובד אליו יש להוסיף את המשמרת
- <day> היום בו המשמרת מתקיימת (מספר שלם בין 1 לבין 7 כולל)
- <shift> שם המשמרת אותה יש להוסיף (אחד מהבאים: "Morning", "Afternoon", "Evening", "Night"). כל מחרוזת אחרת אינה חוקית!

שגיאות:

- מספר זהות אינו תקין – יש לדווח שגיאה מסוג `INVALID_WORKER_ID`
- עובד בעל מספר זהות כזה אינו נמצא במערכת – יש לדווח על שגיאה מסוג `WORKER_DOESNT_EXIST`
- יום המשמרת אינו חוקי – יש לדווח על שגיאה מסוג `INVALID_SHIFT_DAY`

- סוג המשמרת אינו חוקי – יש לדווח על שגיאה מסוג *INVALID_SHIFT_TYPE*
- הוספת המשמרת תגרום למספר רב יותר של משמרות מאלו שהעובד הצהיר כשהוא התווסף למערכת – יש לדווח על שגיאה מסוג *SHIFTS_OVERFLOW*
- משמרת כבר קיימת אצל העובד הספציפי הזה ביום המבוקש – יש לדווח על שגיאה מסוג *SHIFT_ALREADY_EXISTS*

5.4.4 הדפסת העובדים בשבוע הקרוב

```
Report Workers [role]
```

תיאור הפעולה: פקודה זו מדפיסה לערוץ הפלט את כל העובדים הקיימים במערכת. העובדים יודפסו על פי סדר מספרי של תעודות הזהות שלהם. (עליכם למיין את המערך לפני ההדפסה). ההדפסה של כל עובד, בנוסף, תתבצע באמצעות הפונקציה `prog2_report_worker()` המוגדרת בקובץ prog2_ex1.h.
ההדפסות יישלחו לערוץ הפלט שנקבע על פי הפרמטרים שניתנו לתוכנית.

פרמטרים:

- <role> **אופציונלי.** במידה וקיים, המערכת תדפיס רק עובדים בעלי התפקיד המבוקש. אחרת, היא תדפיס את כל העובדים הקיימים במערכת.

שגיאות:

- התפקיד אינו חוקי – יש לדווח על שגיאה מסוג *INVALID_ROLE*
- לא קיימים עובדים במערכת (או לא קיימים עובדים בתפקיד המבוקש במידה והפרמטר קיים) – יש לדווח על שגיאה מסוג *NO_WORKERS*

5.4.5 הדפסת המשמרות עבור עובד מסוים

```
Report Shifts <worker id>
```

תיאור הפעולה: פקודה זו מדפיסה לערוץ הפלט את כל המשמרות הקיימות עבור עובד מסוים. ראשית יש להדפיס את פרטי העובד באמצעות הפונקציה `prog2_report_worker()` המוגדרת בקובץ prog2_ex1.h ורק אז את המשמרות. המשמרות יודפסו על פי סדר כרונולוגי באותו השבוע. ההדפסה של המשמרות תתבצע באמצעות הפונקציה `prog2_report_shift()` המוגדרת בקובץ prog2_ex1.h.
ההדפסות יישלחו לערוץ הפלט שנקבע על פי הפרמטרים שניתנו לתוכנית

פרמטרים:

- <id> מספר הזהות של העובד עבורו יש להדפיס את המשמרות

שגיאות:

- מספר זהות אינו תקין – יש לדווח שגיאה מסוג *INVALID_WORKER_ID*
- עובד בעל מספר זהות כזה אינו קיים במערכת – יש לדווח על שגיאה מסוג *WORKER_DOESNT_EXIST*
- לעובד הנ"ל אין משמרות – יש לדווח שגיאה מסוג *NO_SHIFTS*

5.4.6 הדפסת כל העובדים במשמרת מסוימת

```
Report Shift Details <day> <shift>
```

תיאור הפעולה: פקודה זו מדפיסה לערוץ הפלט את המידע עבור משמרת אחד במערכת. מידע זה מכיל את יום המשמרת, סוג המשמרת, כמות העובדים במשמרת ועלות כוללת של המשמרת. ההדפסה תתבצע באמצעות הפונקציה `prog2_report_shift_details()` המוגדרת בקובץ prog2_ex1.h.
ההדפסות יישלחו לערוץ הפלט שנקבע על פי הפרמטרים שניתנו לתוכנית

פרמטרים:

- <day> היום בו המשמרת מתקיימת (מספר שלם בין 1 לבין 7 כולל)
- <shift> שם המשמרת אותה יש להוסיף (אחד מהבאים : "Night", "Evening", "Afternoon", "Morning"). כל מחרוזת אחרת אינה חוקית!

שגיאות:

- יום המשמרת אינו חוקי – יש לדווח על שגיאה מסוג `INVALID_SHIFT_DAY`
- משמרת אינה חוקית – יש לדווח על שגיאה מסוג `INVALID_SHIFT`
- אין כלל עובדים במאגר או שאין עובדים במשמרת המבוקשת – יש לדווח על שגיאה מסוג – `NO_WORKERS`

5.5 הגבלות על המימוש

עליכם לממש את המערכת תוך התחשבות במגבלות הבאות:

- שימו לב שעליכם לכלול (`#include "prog2_ex1.h"`) את הקובץ הזה בקובץ המימוש שלכם על מנת שתוכלו להשתמש בהגדרות שבו.
- מספר העובדים המקסימלי לאחסון במערכת לא יעלה על `MAX_WORKERS` המוגדר ב prog2_ex1.h.
- מספר שעות למשמרת (עבור חישובי העלויות) הינו `HOURS_PER_SHIFT` ומוגדר אף הוא בקובץ prog2_ex1.h.
- **את כל ההדפסות לפלט יש לבצע באמצעות הפונקציות המתאימות המוגדרות ב prog2_ex1.h**. אין להדפיס דבר למסך או לכל ערוץ פלט אחר פרט למה שנתבקשתם!

5.6 הידור, קישור ובדיקה

על המימוש שלכם לעבור הידור בעזרת הפקודה הבאה (זוהי הפקודה בה בודקי התרגילים ישתמשו לבדיקת התרגיל):

```
gcc -o hr4you -ansi -Wall -pedantic-errors *.c prog2_ex1.o
```

משמעות הפרמטרים:

- -o hr4you זהו שם קובץ ההרצה המהודר
- -ansi 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 שאתם מימשתם
- prog2_ex1.o קובץ אובייקט בו ממומשות פונקציות ההדפסה בהן הנכם נדרשים להשתמש

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, מסופקים שני מבדקים כאלו עם הפלט המצופה עבור כל אחד (כל מבדק מכיל קובץ פקודות בעל הסיומת – in, קובץ פלט בעל הסיומת – out וקובץ שגיאות בעל הסיומת – err).
תוכלו לערוך השוואה בין הפלט שלכם לפלט המצופה באמצעות הפקודה diff.
לדוגמה: אם תריצו א התוכנית שלכם באופן הבא:

```
(hr4you -i test1.in -o my_test1.out ) >& my_test1.err
```

התוכנית תקרא את הקלט מתוך הקובץ test1.in ותדפיס את הפלט הסטנדרטי לתוך הקובץ my_test1.out ואת פלט השגיאות לקובץ my_test1.err.
עתה, תוכלו להשוות את הקבצים שלכם עם אלו שסופקו באמצעות פקודות הלינוקס הבאות:

```
diff  my_test1.out test1.out
diff my_test1.err test1.err
```

במידה והקבצים זהים, הפקודה לא תדפיס כלום למסך. אחרת יודפסו השורות השונות ביניהם.

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"

6 הגשת התרגיל

עליכם להגיש את פתרון חלק א בקובץ נפרד (word, pdf)
בחלק ב עליכם להגיש אך ורק את הקובץ שכתבתם כקבצי טקסט. בפרט אין להגיש את הקבצים המסופקים prog2_ex1.h ו-prog2_ex1.o.

7 דגשים ורמזים

- לשם ניתוח הקלט, ניתן להשתמש בפונקציית הספרייה strtok. מידע על הפונקציה ניתן למצוא כמובן בפקודת הלינוקס man שראינו בתרגול או באתרים כדוגמת http://www.cppreference.com.
  - על מנת להחשיב את סוף השורה כתו מפריד בנוסף על רווח, השתמשו ב
    `strtok(NULL, " \n\r");`
- לצורך המרת מחרוזת למספרים ניתן להשתמש בפונקציות atof() atoi() atol() וכד'
- לבדיקה האם נקרא התו EOF מהקלט, ניתן להשתמש בפונקציה feof
- שמות העובדים מורכבים ממחרוזת בעלת מילה אחת (ללא רווחים)
- ניתן להניח כי אורך שורה בקלט הוא לכל היותר `MAX_LEN` תווים. קבוע זה מוגדר בקובץ prog2_ex1.h
- 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 לפונקציות בהתאם למבנה הלוגי של התוכנית ולא לכתוב פונקציית main אחת גדולה.
- נצפה לראות הפרדה בין פונקציות הממשות לוגיקה מסוימת לאחרות.
- כדי למנוע בעיות עם הבודק האוטומטי, ערך החזרה של התוכנית (ביציאה מפונקציית main) צריך להיות 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'). ולשים לב לא להגיש את התרגיל עם ההדפסות האלו!
- לשם פתרון התרגיל, הגדירו טיפוס חדש מסוג מבנה עבור עובד שיכלול את פרטיו ובנוסף יכלול מערך מגודל 7 עבור משמרות בימות השבוע. השתמשו במערך של עובדים מגודל מקסימלי שאותו תגדירו מראש.

בהצלחה! 😊