האוניברסיטה העברית בירושלים

בית הספר להנדסה ומדעי המחשב ע"ש רחל וסלים בנין

סדנת תכנות בשפת C ו-++C - קורס קיץ (67320)
תרגיל בית 3

תאריך הגשה: 26/08/2020 בשעה 23:25

נושאי התרגיל: מערכים, structs, הקצאות זיכרון דינמיות, ניהול זיכרון

1 רקע

חברת נת"ע (נתיבי תחבורה עירוניים) האחראית על הקמת המסילה לרכבת הקלה בגוש דן, פוטרה מהמכרז עקב חריגה משמעותית מהתקציב. כעת, עיריות גוש דן מחפשות חברה שתחליף אותה. עקב ההפסדים הרבים, תכנון תוואי המסילה נאלץ להיעשות בעלות מינימלית. בתרגיל זה, תכתבו תוכנה אשר תסייע, בהינתן קלט מתאים, למצוא את מחיר המסילה המינימלי אותו החברה החדשה תיאלץ להשקיע בפרויקט.

2 בעיית מסילת הרכבת

בתרגיל זה ננסה לסייע למחליפה של חברת נת"ע. החברה החדשה תספק לנו קובץ קלט שיכלול את אורך המסילה אותה ירצו לבנות ואת סוגי חלקי המסילה הקיימים במאגר. התוכנה תחשב את המחיר המינימלי שיעלה לבנות מסילה באורך ומהחלקים המפורטים בקובץ הקלט.
באופן פורמלי, בהינתן הקלט:

1. אורך המסילה הרצוי.
2. פרטים לגבי חלקים מהם ניתן להרכיב מסילת רכבת (וביניהם מחיר כל חלק).

התוכנית תחשב ותחזיר (הפלט) את מחיר המסילה המינימלי שניתן להרכיב מהחלקים הנ"ל.

הדרך היעילה לפתרון בעיה זו היא עקרון הנקרא "תכנות דינמי" (Dynamic Programming), פתרון בעיות בשיטת "הפרד ומשול". כלומר, בשיטה זו אנו בוחנים תתי בעיות המרכיבות יחד את הבעיה הגדולה אותה אנו רוצים לפתור (פירוט בהמשך). על מנת לפתור בעיות מסוג זה מסתכלים על הצעד האחרון בפתרון אופטימלי ("מיטבי") כלשהו, ומנסים להבין האם כשמורידים צעד זה נשארים עם פתרון אופטימלי לבעיה אחרת, קטנה יותר. כלומר, אנחנו מתחילים בפתרון בעיה אותה אנו יודעים כיצד לפתור ומשתמשים בפתרון זה בכדי לחשב את הפתרון לבעיה מורכבת יותר. כך ממשיכים עד שמגיעים לפתרון הבעיה המקורית.
אם כן, המשימות האלו הן אוסף תתי הבעיות לבעיה הכללית, וכאשר נפתור אותן, נפתור גם את הבעיה המרכזית.

2.1 דוגמא לבעיית תכנות דינמי – סדרת פיבונאצ'י

סדרת פיבונאצ'י היא סדרת מספרים המוגדרת ע"י נוסחת הנסיגה (נוסחת רקורסיה) הבאה:

$$F_n = F_{n-1} + F_{n-2} \quad \forall\, n \in N \; s.t \; n > 2, F_1 = 0, F_2 = 1$$

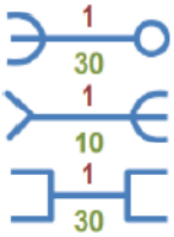
נוכל לפתור את בעיית "חישוב האיבר הn-י" באמצעות טכניקה של תכנות דינמי, נבצע זאת באופן הבא:
על מנת לחשב את F_n, עלינו לדעת את F_{n-1}, F_{n-2} - אלו תתי הבעיות אותן עלינו לפתור (אלה תתי בעיות מאחר שנדרשים פחות חישובים כדי לפתור אותן).
פתרון נאיבי היה לחשב בצורה רקורסיבית את F_{n-1}, F_{n-2} ולחבר אותם בכדי לקבל את F_n. פתרון נאיבי היה בעל זמן ריצה של $O(2^n)$. אבל, אם נתבונן היטב - נראה כי בפתרון הנאיבי אנו מחשבים את אותם ערכים מספר פעמים (למשל, את F_{n-2} נחשב גם עבור F_n וגם עבור F_{n-1}). לכן, אנחנו יכולים לייעל את התהליך - אם נבנה טבלה של הערכים אותם אנו מחשבים, ונסתמך על ערכי התאים שכבר מילאנו כדי לבצע את חישוב התאים הבאים בטבלה.
בתכנות דינמי, אנו מתחילים למלא את טבלת הערכים מהתאים שמייצגים את מקרי הבסיס של נוסחת הנסיגה - עבור פיבונאצ'י, אלה תאים 0,1 המכילים את הערכים של F_1, F_2. מכאן, אנו יכולים למלא את תא 2 בטבלה (המכיל את ערכו של F_3) בזמן $O(1)$. כך ניתן להמשיך ולמלא את הטבלה עד התא ה- n-1 שיכיל את פתרון הבעיה שלנו - ערכו של F_n.
כך, בשימוש בתכנות דינאמי, חישבנו את F_n בזמן $O(n)$, במקום $O(2^n)$ שהיה לנו במימוש הנאיבי.
*** <u>ראו סרטון מצורף לקובץ התרגיל במודל.</u>

2.2 בעיית מסילת הרכבת

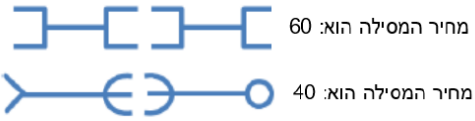
בבעיה זו אנו מרכיבים מסילת רכבת באורך כלשהו (נסמנו L) בעזרת סוגי חלקים נתונים, כאשר כל חלק כולל את הפרטים הבאים:

- מחיר (נסמנו p, מהמילה price)
- אורך (נסמנו d, מהמילה distance)
- חיבור ימני (נסמנו e, מהמילה end)
- חיבור שמאלי (נסמנו s, מהמילה start)

החיבורים הימני והשמאלי מגדירים לאיזה חלק ניתן להתחבר בכל צד (כלומר, חלק עם חיבור שמאלי x יכול להתחבר מצד שמאל לחלק עם חיבור ימני מסוג x).
מטרתנו, כפי שצוין, היא להרכיב מסילה באורך המבוקש בעזרת החלקים הנתונים שמחירה יהיה הנמוך ביותר.
דוגמא:
נניח כי החלקים הנתונים לנו לצורך המשימה הם:



(הספרה האדומה זה האורך, המספר הירוק זה המחיר)
אם ברצוננו להרכיב מסילה באורך 2 תחת אילוצי החיבורים הללו, עומדות בפנינו שתי אפשרויות כיצד לעשות זאת:



מובן שנעדיף את האפשרות השנייה שכן היא זולה יותר.

<u>הנחה</u>: המלאי מכיל רק סוגי חלקים מסוימים הנתונים לכם כקלט, אך עבור כל חלק כזה קיים מאגר בלתי מוגבל של חלקים מכל סוג נתון.

<u>הקלט</u> :

- $L \in N$ - אורך המסילה הרצוייה.
- קבוצת תווים באורך $K \in N$ המייצגת את סוגי החיבורים האפשריים.
- $M \in N$ שורות המייצגות חלקים.

<u>הפלט</u>:

- המחיר המינימלי עבור מסילה חוקית באורך L.

תזכורת: מסילה חוקית היא מסילה בה החלק ה- i מופיע מימין לחלק ה- j ומתקיים $e_j = s_i$.

<u>דוגמא</u>:

נניח שנרצה לבנות מסילה באורך 3 (L=3) עם 4 סוגי חיבורים {@,$,%,&}, ו5 סוגי חלקים:
חלק מספר 1 - (1,30,$,@)
חלק מספר 2 - (1,10,@,%)
חלק מספר 3 - (1,30,&,&)
חלק מספר 4 - (2,40,$,$)
חלק מספר 5 - (3,100,@,&)

<u>נבנה טבלה באופן הבא</u>: ציר ה-y מייצג את אורך המסילה, ציר ה-x מייצג את החיבור הימני עד שלב מסוים.
ממלאים את הטבלה מלמטה (מסילה באורך 0) למעלה (עד למסילה באורך הרצוי). כלומר, נאתחל את הטבלה כך:

3				
2				
1				
0	0	0	0	0
	@	$	%	&

בכל שורה נמלא את המחיר המינימלי למסילה באורך המתאים (לשורה עליה אנו מתבוננים) המסתיימת בחיבור מהסוג המצוין בעמודה
למשל, כך נתחיל למלא את הטבלה:
בתא (1,@) נשים את מחיר המסילה המינימלית באורך 1 המסתיימת בחיבור מסוג @. מסילה זו מתקבלת מחלק מספר 2.
כך נמשיך עבור כל סוגי החיבורים הימניים (כלומר - העמודות) בשורה 1 - כאשר אם אין מסילה מתאימה העונה למגבלות, נסמן ערך של תא זה ב-∞ כך, נקבל אחרי מילוי השורה הראשונה את הטבלה הבאה:

3				
2				
1	10	30	∞	30
0	0	0	0	0
	@	$	%	&

בשלב הבא, נמלא את שורה 2 - במיקום ה-(@, 2) נמלא את המחיר המינימלי למסילת רכבת באורך 2 המסתיימת בחיבור ימני מסוג @, וכפי שניתן להבחין אין קומבינציה של חלקים המתאימה לאילוץ זה, לכן נשים אינסוף בתא. נמשיך ונחשב את תא ($,2) - כלומר, מסילה באורך 2 המסתיימת בחיבור ימני מסוג $. שתי האפשרויות שלנו הן מסילה המסתיימת בחלק 1 או 4. אם נשים את חלק 4 בלבד, נענה על הדרישות. אם נשים את חלק 1, עלינו לשים חלק באורך 1 המסתיים ב-@, שזהו רק חלק 2. כלומר - המחיר המינימלי הוא המינימום בין שתי האופציות שיש לנו. כך נמשיך ונמלא את הטבלה, שורה אחרי שורה - ונקבל את הטבלה הבאה:

3	100	**70**	∞	90
2	∞	40	∞	60
1	10	30	∞	30
0	0	0	0	0
	@	$	%	&

*** שימו לב :

(1) במידה ולא ניתן למצוא אורך טבעי המחיר יהיה אינסוף שמסמל כי לא ניתן להרכיב מסילה באורך המבוקש בעלת חיבור ימני מאותו סוג.

(2) רמז עבה מאוד - מומלץ לבנות טבלה דומה.

(3) רמז עבה נוסף - זכרו כי גודל הטבלה דינמי, כלומר יכולים להשתנות בהינתן קלטים שונים.

(4) שימו לב! על מנת למלא את הטבלה, התחלנו בתתי הבעיות הפשוטות יותר (מסילה באורך 0, לאחר מכן אורך 1 וכו') והסתמכנו על התאים שחישבנו כבר כדי להמשיך ולמלא את הטבלה

ודאו שאתם מבינים היטב את הדוגמא ואיך מורכב כל תא ותא בטבלה!

<u>האלגוריתם:</u>

<u>אוסף תתי הבעיות:</u>

מציאת המחיר המינימלי עבור מסילה באורך l הנגמרת בחיבור ימני מסוג k, עבור $l \in [0, L]$ ו- $k \in A$ (A – מאגר סוגי החיבורים).

<u>נוסחת הרקורסיה - הנוסחה לפתרון כל תת בעיה:</u>

נסמן $f(l,k)$- המחיר המינימלי עבור מסילה באורך **l** הנגמרת בחיבור ימני מסוג **k**:

$$f(l,k) = \begin{cases} 0 & l = 0 \\ \min\limits_{1 \le i \le N : e_i = k \wedge l - d_i \ge 0} \{p_i + f(l - d_i, s_i)\} & l > 0 \end{cases}$$

נגדיר לשם התרגיל min על קבוצה ריקה להיות אינסוף בשביל לכסות את המקרה בו אין חלק שמסתיים בסוג חיבור מסוים.
שימו לב - קיים קובץ header כחלק מהספרייה הסטנדרטית ושמו *limits.h*, ובו נמצא המקרו INT_MAX - תעזרו בו כדי לדמות אינסוף.
המin בנוסחת הרקורסיה בודק מה היא האפשרות הטובה ביותר עבור הצעד האחרון על ידי כך שעבור כל צעד אפשרי (כל חלק שמסתיים בחיבור מסוג k), הוא בודק מה הוא המחיר שיתקבל במידה ובנינו את המסילה עד לחלק זה בצורה אופטימלית.
את האינפורמציה על המחיר המינימלי לכל תת בעיה (לכל $1 \leq l \leq L$ ולכל $k \in A$), נשמור בטבלה, בגודל (L+1) x K) , שנסמנה T. כך ש- $T(l,k) = f(l,k)$.

<u>מילוי הטבלה:</u>
נתחיל ממילוי המקרה בו $l = 0$ (מקרה הבסיס של הרקורסיה), ומשם נמלא את השורות לפי הגדילה של l. מאחר ובכל חישוב משתמשים רק בשורות שמתחת, אין משמעות לסדר המילוי בתוך השורה. בסוף נחזיר את המינימום על השורה העליונה, כלומר $\{T(L,K)\}$.

RailwayPlanner 3

בתרגיל זה נכתוב את התוכנית RailwayPlanner. מטרת התוכנית היא לחשב עבור קלט לבעיית מסילת הרכבת את העלות המינימלית. התוכנית תוודא את תקינות הקלטת, תתמודד עם שגיאות , ותדפיס את המחיר המינימלי לבניית מסילת הרכבת המתאים לקלט.

3.1 הקלט
התוכנית תקבל מהמשתמש בעת הרצתה דרך ה- Command Line Interface (cli, ידוע גם כטרמינל או shell) נתיב לקובץ המכיל את המידע על הקלט לבעיה, לכן פורמט ההרצה יהיה:

```
$ ./RailWayPlanner <InputFilePath>
```

מספר הערות:
- לא ניתן להניח כי יתקבל מספר ארגומנטים תקין.
- לא ניתן להניח כי הקובץ קיים.
-לא ניתן להניח כי הערכים שהתקבלו כקלט תקינים - יפורט בסעיף הבא.

3.2 מבנה קובץ הקלט
מבנה קובץ הקלט יהיה בפורמט הבא (דוגמא לקובץ קלט בהמשך - סעיף 5 - "דוגמת הרצה"(

- שורה ראשונה - המפרטת את אורך המסילה הרצוי (L).
- שורה שניה - המפרטת את סוגי החיבורים- רשימת התווים המהווים חיבורים (ימניים ושמאליים), מופרדים בפסיק.
- משורה שלישית והלאה - N שורות (מספר זה לא נתון) המפרטות כל אחת על סוג חלק אחר, כל שורה מכילה:
  1. חיבור התחלתי (תו מתוך רשימת התווים).
  2. חיבור סיום (תו מתוך רשימת התווים).
  3. אורך החלק.
  4. מחיר החלק.

  מופרדים בפסיקים.

כחלק מבדיקת הקלט עליכם לוודא:
- אורך ומחיר כל חלק הינם מספרים טבעי חיובי ממש. למען הסר ספק, מותר לכם להשתמש בפונקציות שממירות מחרוזות למספרים, כגון strtol ו strod, ורק בהן. כתיבת קוד שתחקה את התנהגות פונקציות אלו תחשב לשגיאה ותגרור הורדת נקודות.
- רשימת התווים מכילה רק תווים בודדים מופרדים בפסיקים.
- כל סוג חיבור המופיע בשורות החלקים נמצא ברשימת התווים המקורית, ובפרט הינו תו תקין.

הערות:

- ניתן להניח כי מבנה כל שורה תקין (לדוגמא, בשורה המפרטת חלק, ישנן 4 מחרוזות באורך גדול מ-0 המופרדות ע"י פסיקים). עוד ניתן להניח כי לא קיימים רווחים לפני או אחרי כל קלט.
- ניתן להניח כי במסמך הקלט קיימות לפחות 3 שורות (שורה המפרטת את אורך המסילה את פירוט החיבורים התקינים, ולפחות שורה אחת המפרטת על חלק). עם זאת, לא מובטח כי תוכן השורות תקין.
חוץ ממצב בו קובץ הקלט ריק לחלוטין (יפורט ב-3.4.3) אתם לא תבחנו על קבצים בהם פחות מ-3 שורות.
- ניתן להניח שאורך כל שורה אינו עולה על 1024 תווים. שימו לב – אין לעשות שימוש בהנחה זו בעת פרסור סוגי החיבורים (שורה 2), כלומר – לא ניתן להקצות מערך סטטי בגודל מחצית השורה, עליכם לנהל זאת באופן דינמי.
- ניתן להניח כי בסוף כל שורה מופיע התו \n והוא בלבד, כלומר השורות לא יסתיימו כ- \r\n.
- ניתן להניח שהתווים הבאים: \n\, r\, 0\, ופסיק לא יופיעו כסוגי חיבורים אפשריים.
- ניתן להניח שאף אחד מהמספרים בקובץ הקלט לא יעלה על MAX_INT.
- ניתן להניח שאם קיים פתרון חוקי לקובץ הקלט הוא יהיה קטן ממש מ MAX_INT.
- לא ניתן לקרוא את קובץ הקלט יותר מפעם אחת (ניתן לבצע מעבר יחיד על כל שורה בלבד).

3.3 המרת הקלט למבנה הנתונים המתאים

לאחר קריאת הקלט וכחלק מפתרון הבעיה עליכם לבנות מבנה נתונים מתאים לאחסון המידע.
הנכם חופשיים לעצב מבנה נתונים אחד או יותר שישרתו אתכם בפתרון התרגיל כראות עיניכם, ובלבד שתעמדו בהנחיות הבאות:

- **עליכם להשתמש לכל הפחות במבנה נתונים אחד (כלומר לכל הפחות בסוג struct אחד).**
- **עליכם לעשות לכל הפחות שימוש אחד ב־typedef.**
- **עליכם לעשות לכל הפחות בהקצאת זיכרון דינמית בהקשר אחד .**

שימו לב, כי כל שימוש לא רלוונטי, לא חכם, לא יעיל בכל אחד מהכתובים לעיל יגרור הורדה בציון ללא אפשרות לערעור, לכן חשבו מהי הדרך המתאימה והנכונה לעשות בהם שימוש!
*** דוגמא לשימוש לא חכם:

```
struct bool{
int value;
};
```

3.4 טיפול בשגיאות ובשגיאות קלט

עליכם לטפל במקרים בהם התקבל קלט לא תקין:
(1) אם מספר הארגומנטים שנשלחו לתוכנה אינו תקין, עליכם להדפיס לקובץ הפלט (יפורט בפרק הבא) את השורה הבאה כפי שהיא מוצגת כאן:

Usage: RailwayPlanner <InputFile>\n

(2) אם נתקלתם בקובץ שאינו קיים או במקרה בו יש תקלה בפתיחת קובץ הקלט, עליכם להדפיס לקובץ הפלט את הפלט:
File does not exist.\n

(3) אם נתקלתם בקובץ ריק, עליכם להדפיס לקובץ הפלט את הפלט:

File is empty.\n

(4) אם נתקלתם בקובץ קלט לא תקין מסיבה אחרת, עליכם להדפיס לקובץ הפלט את הפלט (שימו לב – line-number הינו ממלא מקום למספר השורה האמיתית):

Invalid input in line: <line-number>.\n

(5) במקרה בו ישנה שגיאה בפתיחת קובץ הפלט, אין צורך להדפיס דבר ולצאת מהתוכנית.

(6) במקרה בו אחת מפונקציות הספרייה הסטנדרטית נכשלה, אין צורך להדפיס דבר וצריך לצאת עם קוד שגיאה מתאים.

בסעיף 4 ציינו את השורה בקובץ הקלט בה התגלתה השגיאה המוקדמת ביותר. שימו לב שמדובר במספר השורה ולא באינדקס השורה. כלומר - השורה הראשונה בקוד היא השורה מספר 1 ולא השורה ה-0.

בארבעת המקרים "\n" מסמן ירידת שורה ואינו אמור להיות מודפס לקובץ הפלט.
לאחר הדפסת הפלט לקובץ, בשני המקרים עליכם, לוודא שחרור כל הזיכרון ולסגור באופן מידי את התוכנה עם קוד סיום EXIT_FAILURE.

בכל מצב של יציאה מן התכנית עליכם לוודא שכל הזיכרון שוחרר, למעט המקרים בהם נכשלות אחת מפונקציות הספרייה הסטנדרטית. דוגמא – במידה והפונקציה **malloc** נכשלה, נהגו לפי סעיף 6 והנכם פטורים משחרור כלל הזיכרון.

3.5 פלט התוכנית

אם הקלט תקין, על התוכנית שיצרתם לפתוח מסמך חדש בשם **rwp_output.out**, באותה התיקיה בה התוכנית מופעלת, ולהדפיס לתוכו את המחיר המינימלי שנמצא, בפורמט הבא:

The minimal price is: <minimal_price>.\n

למען הסר ספק, **minimal_price** הוא המחיר המינימלי שנמצא ע"י התכנית.

<u>הערה-</u> במקרה בו המחיר המינימלי הוא אין-סופי ,כלומר אין דרך לבנות מסילה באורך המבוקש מהחלקים שהוכנסו כקלט, יש להדפיס כמחיר מינימלי 1-. כלומר ההודעה תהיה:

The minimal price is: -1.\n

4 דרישות זמן ריצה

על האלגוריתם אותו תממשו במהלך התרגיל לפתור את הבעיה בזמן ריצה: <u>O(N*L*|A|)</u>
*** חשבו על כך היטב טרם אתם מתחילים לכתוב את התוכנית!
צפו בסרטון שהועלה למודל העוסק בתכנות דינמי לעומת תכנות רקורסיבי.
*** שימו לב – זמן הריצה המוצג כאן הינו מחייב וכל פתרון שיעלה עליו (גם אם יהיה מנומק היטב) יגרור אבדן נקודות משמעותי וסביר כי לא יצליח בבדיקה האוטומטית.

5 דוגמת הרצה

נציג דוגמה מתאימה לסיטואציה שתוארה בסעיף 2: נניח שיש לנו קובץ קלט בשם data.in שנראה כך:

```
3
@,$,%,&
@,$,1,30
%,@,1,10
&,&,1,30
$,$,2,40
&,@,3,100
```

אזי נוכל להריץ את הפקודה הבאה מהטרמינל:

```
$ ./RailwayPlanner ./data.in
```

המסמך rwp_output.out יהיה: (היצמדו לפתרון בית הספר!)

```
The minimal price is: 70.
```

6 הערות וטיפים לפתרון התרגיל

1. וודאו כי אתם מבינים היטב את הקלט ואת מבנהו. הבנה טובה של זה יכולה לחסוך זמן רב ושאלות מיותרות.
2. תכננו את התרגיל - ניתן לחלק את התרגיל לחלקים מאוד ברורים. תכנות של כל חלק לחוד יכול לעזור מאוד כאן. יש המון דרכים לפתור את התרגיל. הרבה מהן לא טובות. השתמשו בכל הכלים שלמדתם (רמז - עמ' 1, נושאי התרגיל)
3. סדר וניקיון הקוד. יש בתרגיל לא מעט עבודת פירסור (parsing) של מחרוזות. ביצוע פעולות אלו בC מורכב לעתים. וודאו שאתם שומרים על עבודה מסודרת ומתכננים מספר צעדים קדימה כדי שהקוד שלכם לא יהפוך לבלגן, שיפגע גם בעבודה שלכם וגם בקלות של הצוות לבדוק את הקוד. זכרו שמחרוזות חייבות להסתיים בתו ה-NULL (כלומר '\0').
5. זכרו לקפמל (compile) את התרגיל על מחשבי האקוריום , ישנם הבדלים משמעותיים בין מערכות הפעלה שונות. זכרו שהתרגילים נבדקים במחשבי בית הספר, ולכן ציון סטודנט שתרגילו לא רץ כהלכה במחשבי בית הספר עלול להיפגע משמעותית, אף אם במחשבו האישי התרגיל רץ כנדרש. הפלט הקובע הוא זה שניתן במחשבי בית הספר.
6. לאלו הנעזרים ב IDE CLion, שימו לב שלא כמו פייתון, הדיבאגר כאן הרבה פחות ידידותי. כלומר, אם נרצה לבדוק בעזרת הדיבאגר (debugger) את התוכן של מצביע, לעיתים לא נראה את כל התוכן. מומלץ להיעזר בפונקציות הדפסה על מנת לוודא שטיפוסים שהגדרתם בקוד מכילים את התוכן אותו אתם רוצים שיכילו.
7. שימו לב לשימוש בINT_MAX בפעולות אריתמטיות! רמז: overflow.
8. ישנם הרבה ניואנסים של מה ניתן\לא ניתן להניח על הקלט. כל אלה ניתנו על מנת שלא תצטרכו לפרסר את הקובץ ללא הנחות מקדימות, מה שבא לטובתכם. לכן שימו לב שקראתם היטב את מה שמותר לכם להניח ומה שלא, ומה מוגדר תקין ומה לא. (טיפ - כתבו בצורה מסודרת מה ניתן להניח על כל אלמנט בקלט).
9. **שימו לב כי פתרון ביה"ס מדפיס למסך לנוחות הסטודנטים בלבד!** על פתרון הסטודנטים לשמור את הפלט בקובץ ולא לבצע הדפסה למסך. (אם תרצו לבצע diff מול פתרון בית ספר: בצעו redirection לפתרון בית הספר לקובץ ורק אז בצעו השוואה).
10. היזהרו משימוש במערכי VLA!!! שימוש בהם יגרור הורדת נקודות משמעותית.
11. שימו לב – אין לקרוא את קובץ הקלט יותר מפעם אחת! קריאת קובץ הקלט פעמיים על מנת להימנע מהקצאות זיכרון דינמיות הינה שגיאה.

7 ציון התרגיל

ציון התרגיל יתבסס על מספר חלקים:

1. בדיקת התוכנית שלכם בעזרת טסטים אוטומטיים שיבחנו קלטים תקינים בגדלים שונים, ברמות מורכבות שונות וגם בעזרת קלטים לא תקינים. השוו את תרגילכם מול פתרון בית ספר בעזרת פקודת diff. תוצאת diff לא תקינה תחשב כאי מעבר של טסט ללא אפשרות לערעור על כך.
2. בדיקה ידנית (כמו בכל תרגיל).
3. בדיקות דליפת זיכרון – בתרגיל זה יינתן <u>משקל רב</u> לדליפות זיכרון. ודאו בעזרת תוכנת הvalgrind כי הקוד שלכם משחרר את כל הזיכרון עבור סוגי קלטים שונים. לנוחיותכם, מצורף <u>מדריך</u> מפורט במודל <u>וסרטון</u> בו אנו מראים כיצד לעשות זאת.

8 ספריות עזר

ניתן להשתמש בכל אחת מן הספריות הבאות ובתנאי שהשימוש איננו מייתר חלקים מהתרגיל:
string.h, assert.h, ctype.h, stdlib.h, stdio.h, stdbool.h, math.h

לא ניתן להשתמש בספריות:
regrx.h, unistd.h

לגבי כל ספרייה אחרת בה תרצו להשתמש:
- ודאו כי היא עומדת בסטנדרט c99.
- ודאו שזו איננה מייתרת חלקים מהתרגיל.
- ודאו עם הסגל דרך הפורום.

9 נהלי הגשה

- שימו לב כי על הקוד שלכם לעבוד על מחשבי אקווריום - קוד שלא ירוץ באופן תקין על מחשבי אקווריום לא יזכה בניקוד.

- קראו בקפידה את הוראות תרגיל זה ואת ההנחיות להגשת תרגילים שבאתר הקורס (שימו לב לפונקציות שקיים איסור בקורס על שימוש בהן).

- כתבו את כל ההודעות שבהוראות התרגיל בעצמכם. העתקת ההודעות מהקובץ יכולה להוסיף תווים מיותרים ולפגוע בבדיקה האוטומטית, המנקדת את עבודתכם.

- על מנת להריץ את פתרון בית הספר:
  ~proglab/www/ex3/school_solution <input_file>

- על מנת להריץ את בדיקת הcoding style:
  ~proglab/www/codingStyleCheck <code files>

- עליכם ליצור קובץ tar הכולל אך ורק את הקובץ RailwayPlanner.c (בפורמט הנדרש בהנחיות להגשת תרגילים). ניתן ליצור tar כנדרש על ידי:
  $ tar -cvf ex3.tar RailwayPlanner.c

- שימו לב:RailwayPlanner.c יכלול את מלוא התרגיל לרבות ה-main. על הקובץ להתקמפל כהלכה עם C 99, כנדרש בהוראות להגשת תרגילים שפורסמו באתר הקורס.
  gcc -Wextra -Wall -Werror -Wvla -std=c99 RailwayPlanner.c -o RailwayPlanner

- אנא וודאו כי התרגיל שלכם עובר את ה-Pre-submission Script ללא שגיאות או אזהרות. קובץ ה-Pre-submission Script זמין בנתיב:
  ~proglab/www/ex3/presubmit_ex3 <path to ex3.tar>

<u>בהצלחה.</u>