מבוא למדעי המחשב מ'/ח' (234114/7), סמסטר חורף 2020-21

תרגיל בית 4

מועד אחרון להגשה: 06.01.2021 23:59

המתרגלת האחראית על תרגיל זה: **קטרין חדאד-זקנון**

שעת קבלה לתרגול: שלישי 12:30-14:30
Zoom: https://technion.zoom.us/j/4966839889
E-mail: catherine@cs.technion.ac.il

הנחיות:

- הגשה ב**בודדים**. עליכם לכתוב את הפתרונות לבד ולהגיש ביחידים.
- קראו את השאלות בעיון לפני שתתחילו בפתרונן.
- הקפידו לתעד את הקוד שלכם בהערות באנגלית.
- מלבד מילואים, לא יתקבלו תרגילים אחרי מועד הגשה. הגשה באיחור לאחר מועד הגשה נחשבת כאי-הגשה.
- כל יום מילואים = יום דחייה. על מנת לקבל את הדחייה, עליכם לשלוח באי-מייל, עותק של האישור המראה שהייתם במילואים (טופס 3010). אם האישור יגיע אליכם בתאריך מאוחר, יש להודיע על כך למתרגל האחראי על התרגיל לפני תאריך הגשת התרגיל.
- ערעורים ניתן להגיש עד שבוע לאחר קבלת הציון.
- **לא ניתן לערער על תוצאות הבדיקה האוטומטית.**
- **שימו לב! הבדיקה הינה בחלקה אוטומטית, ולכן הקפידו להדפיס בדיוק בפורמט שהתבקשתם ובידקו עם DiffMerge את הפלט שלכם מול הפלט של הדוגמאות שקיבלתם.**
  - השתמשו ב-redirection כדי להפנות את הפלט לקובץ טקסט.
  - וודאו את האותיות הגדולות והקטנות לפי הדוגמאות וההסברים בתרגיל.
  - **אין להדפיס רווחים שלא התבקשתם להדפיס (בתחילת שורה או בסופה).**
  - **בכל סוף שורה יש להדפיס תו ירידת שורה, גם בשורה האחרונה.**
  - השתמשו באתר הבדיקה העצמית.

- בתרגיל זה מותר להשתמש בפונקציות מהספריות stdbool.h ,stdio.h, stdlib.h, string.h, שנלמדו בהרצאות ובתרגולים, אלא אם צוין אחרת.
- ההגשה הינה אלקטרונית ו**בבודדים** דרך אתר הקורס. קובץ ההגשה יהיה מסוג **zip** (ולא אף פורמט אחר) ויכיל בתוכו את הקבצים הבאים בלבד, ללא כל תיקיות:
  - קובץ **students.txt** עם מספר תעודת הזהות שלך וכתובת האי-מייל שלך.
  - קובץ פתרון **hw4q1.c** לשאלה 1
  - קובץ פתרון **hw4q2.c** לשאלה 2
  - קובץ פתרון **hw4q3.c** לשאלה 3
- **חובה לשמור את קוד אישור ההגשה שמקבלים מהמערכת לאחר שמגישים, עד לסיום הקורס.**
- יש להקפיד להגיש את כל הקבצים בדיוק עם השמות שמופיעים לעיל. הגשה שלא תעמוד בתנאי זה **לא תתקבל ע"י המערכת!** אם המערכת לא מקבלת את התרגיל שלכם, חפשו את הפתרון לבעיה באתר הקורס תחת הכפתור FAQ.

שאלה 1 : מצביעים והקצאות דינמיות

בתרגיל ה נרצה לממש מערכת אשר מנהלת נוכחות סטודנטים במבחנים מכוונים. לכל חדר בחינה וירטואלי נוכל לשבץ עד MAX_STUDENTS סטודנטים. MAX_STUDENTS מוגדר להיות 10.

- כמו בבחינות רגילות, לא כל הסטודנטים מגיעים להיבחן, כך שיתכן שיהיו פחות מ- MAX_STUDENTS סטודנטים בבחינה.
- כל סטודנט שהגיע לבחינה רשאי לצאת מהבחינה מבלי להגיש את הבחינה שלו.
- כל סטודנט מקבל מזהה ייחודי שהוא מספר מ- 0 עד MAX_STUDENTS-1 . מזהה זה הוא מספר פנימי של המערכת, כלומר לא מתקבל כקלט מהמשתמש.
- לכל סטודנט נרצה לשמור את הנתונים הבאים:
  - השם של הסטודנט – מחרוזת באורך עד MAX_NAME תווים.
  - תעודת הזהות של הסטודנט – מספר שלם חיובי.
  - הארכת זמן – מספר בדקות שמסמן כמה הארכת זמן מגיע לסטודנט.
- תיאור מסד הנתונים (database): הנתונים של הסטודנטים יישמרו בשני מערכים:
  - מערך ראשון הוא **מערך השמות** המכיל מחרוזות מחרוזות השמות של הסטודנטים.
  - **מערך מידע** הוא מערך דו-ממדי הוא המכיל עבור כל סטודנט את מספר ת״ז שלו וכמה הארכת זמן מגיע לו במקרה שהוא זכאי לכזו. במידה והסטודנט אינו זכאי להארכה, ישמר עבורו 0 בשדה המתאים. המערך מכיל MAX_STUDENTS שורות ו- INFO עמודות.
  - **ערכי הקבועים מוגדרים לכם בקובץ hw4q1main.c**
  - **מותר** להשתמש בפונקציות של <string.h>.

א. ממשו את הפונקציה הבאה:

```
void init_db(int students_info[][INFO], char* students_names[], int n);
```

- הפונקציה מקבלת את **מערך השמות** ואת **מערך המידע** של הסטודנטים ואת **אורכם** ומאתחלת את התוכן שלהם לערכים ריקים.
- ערך ריק במערך המידע יסומן ע״י הערך 1-.
- דרישות סיבוכיות: O(n) כאשר n הוא אורך המערכים.

ב. ממשו את הפונקציה הבאה:

```
int read_students_info(int students_info[][INFO], char* students_names[], int n);
```

- הפונקציה מקבלת את **מערך השמות** ואת **מערך המידע** של הסטודנטים ואת **אורכם** ואמורה למלא את הנתונים של הסטודנטים במערכים הנ״ל ע״פ הנתונים שהיא קוראת מהקלט לפי התיאור הבא:
  - בצעד הראשון הפונקציה אמורה לקרוא מהקלט את מספר הסטודנטים **M**. מספר הסטודנטים כתוב בהתחלת שורה ללא כל אינפורמציה אחרת באותה שורה.
  - בצעד השני הפונקציה אמורה לקרוא בזה אחר זה את הנתונים של הסטודנטים. ניתן להניח כי לאחר מספר הסטודנטים מופיעים M שורות קלט המתארים את הנתונים של הסטודנטים. כל שורה היא בפורמט הבא:

    <student name> <id> <minutes>

  - כאשר:
    - <student name>: הינו מחרוזת המכילה אותיות בשפה האנגלית (אותיות גדולות או קטנות ללא רווחים או כל תו מיוחד אחר). אורך המחרוזת הוא **עד** MAX_NAME תווים. האורך MAX_NAME **איננו** נחשב קבוע לצורך חישוב סיבוכיות זמן או מקום.
    - <id>: הוא מספר שלם חיובי **שאיננו** אחד מהמספרים: 100, 200, 300 (מספרים אלה יקבלו משמעות מיוחדת – הסבר בהמשך).

- <minutes>: מספר שלם חיובי או אפס המתאר כמה דקות של הארכת זמן מגיע לסטודנט.

- אם M>n: על התכנית להחזיר את הערך 1- ולהדפיס את ההודעה:

Too many students!! Aborting!!

- אם הפעולה הצליחה אזי על הפונקציה להדפיס את ההודעה:

First <M> students were added to the database!!

כאשר <M> הוא מספר הסטודנטים שהוכנסו.

- **דרישות סיבוכיות:**
  - סיבוכיות זמן: על הפונקציה לרוץ בזמן: $O(M+K)$ כאשר K זה מספר התווים הכולל של **כל השמות** של הסטודנטים.
  - סיבוכיות מקום: $O(K+\text{MAX\_NAME})$
- בסעיף זה יש להניח שמסד הנתונים הוא ריק בעת הקריאה לפונקציה זו.
- בסעיף זה יש להניח שהקלט תקין וכמו כן הוא לא מכיל שני סטודנטים עם אותו מזהה.

ג. ממשו את הפונקציה הבאה:

```
void add_student(int students_info[][INFO], char* students_names[],int n);
```

- הפונקציה מקבלת את **מערך השמות** ואת **מערך המידע** של הסטודנטים ואת **אורכם**
- לאחר בניית מסד הנתונים (database), יהיה אפשרי להוסיף סטודנטים ע״י הפונקציה הנ״ל.
- הפונקציה הנ״ל קוראת מהקלט נתונים על סטודנט להוספה ומכניסה אותו למערכת לפי הכללים הבאים:
  - אם יש סטודנט עם אותו id אזי הפונקציה לא מכניסה את הסטודנט החדש ומדפיסה את ההודעה:

    Student <id> already exists!!

    כאשר id הוא המזהה של הסטודנט שמנסים להכניס.

  - אם הכיתה כבר מלאה (כלומר יש בפועל MAX_STUDENTS) אזי הפונקציה לא מכניסה את הסטודנט החדש ומדפיסה הודעה מתאימה:

    Cannot add student <id>, class already full!!

    כאשר id הוא המזהה של הסטודנט שמנסים להכניס.

  - אם הצלחנו להכניס את הסטודנט, אזי על התכנית להדפיס למסך את ההודעה:

    Student <id> was successfully added to the system!!

    כאשר id הוא המזהה של הסטודנט שמנסים להכניס.
- **דרישות סיבוכיות:**
  - סיבוכיות זמן $O(n+s)$ כאשר s הוא אורך המחרוזת של שם הסטודנט שרוצים להוסיף.
  - סיבוכיות מקום: $O(s+\text{MAX\_NAME})$
- קריאת הקלט: אפשר להניח כי שורת הקלט היא בפורמט הבא:

<student name> <id> <minutes>

ד. **ממשו את הפונקציה הבאה:**

```
void delete_student(int students_info[][INFO], char* students_names[],int n);
```

- הפונקציה מקבלת את **מערך השמות** ואת **מערך המידע** של הסטודנטים ואת **אורכם**
- לאחר בניית מסד הנתונים (database), יהיה אפשרי למחוק נתונים של סטודנט ע״י הפונקציה הנ״ל.

- הפונקציה הנ"ל קוראת מהקלט מזהה של הסטודנט ומנסה למחוק את הנתונים שלו מהמערכת לפי הכללים הבאים:
  - אם אין סטודנט עם אותו id אזי על הפונקציה להדפיס את ההודעה:
    Student <id> does not exist!!!
    כאשר id הוא המזהה של הסטודנט שמנסים להסיר.

  - אם הצלחנו להסיר את הנתונים של הסטודנט, אזי על התכנית להדפיס למסך את ההודעה:
    Student <id> information were deleted from the system!!!
    כאשר id הוא המזהה של הסטודנט שמנסים להסיר.
- **דרישות סיבוכיות:**
  - סיבוכיות זמן $O(\text{MAX\_STUDENTS})$
  - סיבוכיות מקום $O(1)$

ה. **ממשו את הפונקציה הבאה:**

```
void clean_db( char* students_names[],int n);
```

הפונקציה משחררת מהזיכרון את כל המידע שהוקצה באופן דינמי וביציאה מדפיסה את ההודעה:
All <num> entries were deleted!!!
כאשר num הוא מספר הסטודנטים בפועל במסד הנתונים.

הסברים על קובץ הקלט ופונקצית ה- main:

1. בקובץ hw4q1main.c ממומשת לכם פונקצית main.
2. הקלט מכיל שלושה סוגים של פקודות: 100,200,300
   a. **פקודה 100** מופיעה בהתחלת קובץ הקלט. הפקודה 100 אומרת לתוכנית שברצוננו לבצע פקודת קריאת הנתונים של כל הסטודנטים (read_students_info) . לאחר הפקודה 100 בקובץ, יירשם מספר הסטודנטים שרוצים להכניס, ולאחר מכן הנתונים על הסטודנטים שורה לאחר שורה.
      i. **דוגמא:**

```
100
3
Sam 1000 0
Phil 2000 0
Roni 3000 20
```

      ii. יש להניח כי הפקודה 100 מופיעה בדיוק פעם אחת והיא תופיע ראשונה בקובץ. וכן כי מספר שורות המידע שווה למספר שרשום מיד לאחר הפקודה 100.

   b. **פקודה 200**: היא פקודת הוספת סטודנט. מיד לאחר השורה בקלט שבה מופיעה הפקודה 200, מופיעה שורה בודדת המתארת את נתוני הסטודנט שרוצים להוסיף.
      i. **דוגמא:**

```
100
3
Sam 1000 0
Phil 2000 0
Roni 3000 20
200
```

```
Tomer 4000 15
```

a. **פקודה 300**: היא פקודת הסרת סטודנט. מיד לאחר השורה בקלט שבה מופיעה הפקודה 300, מופיעה שורה בודדת המכילה את המזהה של הסטודנט שרוצים להסיר.

i. **דוגמא:**

```
100
3
Sam 1000 0
Phil 2000 0
Roni 3000 20
200
Tomer 4000 15
300
2000
```

b. לאחר הפקודה 100 יתכנו מספר פקודות 200 או 300.

c. ניתן להניח כי הקלט תקין.

דוגמת הרצה:

```
100
4
Adam 1111 25
Eve 2222 0
Kaien 3333 0
Habil 4444 15
200
Romeo 5555 15
200
Juliette 6666 15
300
1111
200
Juliette 6666 15
```

```
First 4 students were added to the database!!
Student 5555 was successfully added to the system!!
Student 6666 was successfully added to the system!!
Student 1111 information were deleted from the system!!!
Student 6666 already exists!!
All 5 entries were deleted!!!
```

שאלה 2: סיווג מילים ע"פ סופית

חברת האח הגדול בע"מ רוצה לבנות מערכת אשר תאסוף מידע על משתמשים בפייסבוק: מה עשו, לאן הלכו ומה הם חושבים על זה. לשם כך הם רוצים לבנות מערכת אשר מסוגלת לקרוא טקסט, להבין את התוכן שלו ולנתחו. אחד השלבים הבסיסיים במערכת הינו כתיבת פונקציה אשר מזהה סוגי מילים מיוחדים בטקסט: שמות עצם, פעלים ותארים מיוחדים. מאחר ובאנגלית (כמו בשפות אחרות) ניתן במקרים רבים לזהות מילה לפי הסיומת שלה (suffix), הוטל עליכם לבחון את השיטה הזו.
לצורך בחינת המערכת הוגדרו שלושה מילונים של סופיות, אחד לשמות עצם (nouns), אחד פעלים (verbs) ואחד לשמות תואר (adjectives):

Noun: ism, ics, er, ist, ment, ness
Verb: ate, ify, ize, ing, ed
Adjective: able, al, ful, ish, ive, ous

לנוחיותכם המילונים נשמרו בשלושה מערכים של מחרוזות, כאשר כל סיומת כתובה מהסוף להתחלה וכל מילון ממוין בסדר לקסיקוגרפי:

```
.
.

int main(){
  char *noun_suffixes[] = {"msi", "re", "sci", "ssen", "tnem", "tsi"};
  char *verb_suffixes[] = {"de", "eta", "ezi", "gni", "yfi"};
  char *adj_suffixes[] = {"elba", "evi", "hsi", "la", "luf", "suo"};
  .
  .
```

א. בשלב הראשון עליכם לממש פונקציה עם החתימה הבאה:

```
bool is_suffix_in_dict(char *str, char *dict[], int n);
```

הפונקציה מקבלת מילה (מחרוזת אשר מובטח כי אורכה אינו עולה על 32 תווים) ומקבלת מילון ומחזירה true אם המילה מסתיימת בסופית אשר נמצאת במילון. מאחר ועל המערכת לנתח כמויות גדולות של טקסט, על הפונקציה להיות יעילה ככל האפשר: בהינתן מילון באורך n עליה לרוץ בסיבוכיות זמן ריצה O(logn) וסיבוכיות זיכרון נוסף O(1). האורך המקסימלי של מילים וסיומות הוא 32 ולכן נחשב כקבוע לצורך חישובי סיבוכיות. להלן החוקים לפיהם יש להתאים מילים למילון:

a. במילון אין סופית שבעצמה מסתיימת בסופית אחרת מהמילון. אם מצאתם סופית אפשר לסיים ואין צורך לבדוק האם יש עוד סופית ארוכה יותר שגם מתאימה.
b. יש להתעלם מתווים שאינם אות אנגלית לצורך הבדיקה. כלומר המחרוזת "communism!" מתאימה לסופית "ism". כמו כן "communis,m" מתאים גם הוא לסופית "ism".
c. על ההשוואה להיות **case-insensitive**, כלומר אין חשיבות לאות גדולה לעומת אות קטנה: CoMmUnIsM, COMMUNISM ו communism כולם מתאימים לסיומת "ism".

ב. עליכם לכתוב פונקציה עם החתימה הבאה:

```
bool read_sentence(char *noun_suffixes[], int noun_suffixes_len,
                   char *verb_suffixes[], int verb_suffixes_len,
                   char *adj_suffixes[], int adj_suffixes_len,
                   int *num_of_nouns, int *num_of_verbs, int *num_of_adjs);
```

הפונקציה מקבלת שלושה מילונים של סופיות, קוראת משפט מהקלט הסטנדרטי ומחזירה כמה מילים היו מכל סוג על פי הכללים הבאים:

a. ערך ההחזרה של הפונקציה יהיה true אם הגענו לסוף הקלט ו false אחרת.
b. את התוצאה יש לשים במשתנים המוצבעים ע"י `num_of_nouns, num_of_verbs, num_of_adjs`
c. אורך כל מילה לא יעלה על 32 תווים.
d. משפט הוא רצף מילים מופרדות ביניהן ברווח כלשהו (רווח, טאב או ירידת שורה) המסתיים בתו נקודה ("I hate communism."). לא ניתן להניח מספר מקסימלי של מילים במשפט.
e. בהינתן משפט באורך k מילים, על הפונקציה לרוץ בסיבוכיות זמן O(klogn) וסיבוכיות מקום נוסף O(1).

ג. עליכם להשלים את מימוש פונקציית ה- main הנתונה לכם בקובץ **hw4q2main.c**. על הפונקציה לקרוא משפטים בזה אחר זה כל עוד לא מגיעים ל EOF ולאחר כל משפט מדפיסה את הנתונים שנאספו ע"י שימוש בפונקציית ההדפסה שגם היא ניתנת לכם בקובץ **hw4q2main.c** .

a. *ניתן להניח כי כל משפט מסתיים בנקודה, כולל האחרון.*
b. *ניתן להניח שכל משפט חדש מופיע בשורה חדשה, אך משפט כן יכול להיות נמשך על יותר משורה אחת.*
c. להלן מספר דוגמאות הרצה של התוכנית:

Example #1:
Input:
I was wasting my precious time waiting for your student.
I was wasting my p-r-e-c-i-o-u-s time waiting for your foolish friend.
.

Expected output:
Enter text to analyze:
The sentence had 0 special nouns, 2 special verbs and 1 special adjectives.
The sentence had 0 special nouns, 2 special verbs and 2 special adjectives.
The sentence had 0 special nouns, 0 special verbs and 0 special adjectives.

Example#2:
Input:
gooooooogle?.
I frie'd one toast.
Expected output:
Enter text to analyze:
The sentence had 0 special nouns, 0 special verbs and 0 special adjectives.
The sentence had 0 special nouns, 1 special verbs and 0 special adjectives.

שאלה מס' 3: סיבוכיות של קוד

בשאלות 1-6 בחרו את האות המייצגת את הסיבוכיות המתאימה מתוך טבלת הפונקציות להלן:

a. $\Theta(1)$
b. $\Theta(\log n)$
c. $\Theta(\log^2 n)$
d. $\Theta(\sqrt{n})$
e. $\Theta(\sqrt{n}\log n)$
f. $\Theta(\sqrt{n}\log^2 n)$
g. $\Theta(n)$
h. $\Theta(n\log n)$
i. $\Theta(n\log^2 n)$
j. $\Theta(n\cdot\sqrt{n})$
k. $\Theta(n^2)$
l. $\Theta(n^2\log n)$
m. $\Theta(n^2\log^2 n)$
n. $\Theta(n^2\cdot\sqrt{n})$
o. $\Theta(n^3)$
p. $\Theta(n^3\log n)$
q. $\Theta(n^3\log^2 n)$
r. $\Theta(2^n)$
s. $\Theta(n\cdot 2^n)$
t. $\Theta(n^2\cdot 2^n)$
u. $\Theta(n^3\cdot 2^n)$
v. $\Theta(3^n)$
w. $\Theta(n\cdot 3^n)$
x. $\Theta(n^2\cdot 3^n)$
y. $\Theta(n^3\cdot 3^n)$
z. *the correct answer doesn't appear*

יש להניח כי סיבוכיות הזמן של הפונקציות $malloc$ ו- $free$ היא $\Theta(1)$, וסיבוכיות המקום של הפונקציה $malloc(n)$ היא $\Theta(n)$.

1. מהי סיבוכיות הזמן של הפונקציה f1 כתלות ב-n?
2. מהי סיבוכיות המקום של הפונקציה f1 כתלות ב-n?

```
void f1(int n)
{
      int x = 0;
      for (int i = 0; i <= n / 2; i += 3)
            for (int j = i; j <= n / 4; j += 2)
                  x++;
}
```

3. מהי סיבוכיות הזמן של הפונקציה f2 כתלות ב-n?
4. מהי סיבוכיות המקום של הפונקציה f2 כתלות ב- n?

```
void f2(int n)
{
       int i, j, k = 0;
       for (i = n/2; i <= n; i++)
       {
            for (j = 2; j <= i; j *= 2)
             {
                  k+=n;
             }
      } free(malloc(k));
}
```

5. מהי סיבוכיות הזמן של הפונקציה f3 כתלות ב-n?
6. מהי סיבוכיות המקום של הפונקציה f3 כתלות ב-n?

```
void f3(int n)
 {
 int* temp;
int k = 0;
      for (int i = 0; i < n; i += k)
      {
      ++k;
      temp = malloc(k * k);
      free(temp);
      }
 }
```

אופן הגשת התשובות בשאלה 3:
באתר הקורס מסופק לכם קובץ טקסט בשם hw4q3_example.txt שזה תוכנו:

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

int main()
{
    int question;
    scanf("%d",&question);
    switch (question)
    {
    case 1 :
        printf("The answer to question 1 is a\n");
        break;
    case 2 :
        printf("The answer to question 2 is a\n");
        break;
    case 3 :
        printf("The answer to question 3 is a\n");
        break;
    case 4 :
        printf("The answer to question 4 is a\n");
        break;
    case 5 :
        printf("The answer to question 5 is a\n");
        break;
    case 6 :
        printf("The answer to question 6 is a\n");
        break;
    default :
        printf("Error!\n");
        break;
    }
    return 0;
}
```

הקובץ כמובן מתאים למקרה שבו בכל השאלות בחרתם בתשובה "a". כל שעליכם לעשות הוא להוריד את הקובץ מאתר הקורס, לשנות את האות "a" לאות המתאימה לתשובה שלכם בכל שאלה, ולשמור את הקובץ מחדש בשם hw4q3.c. את הקובץ hw4q3.c יש להגיש בתוך קובץ הזיפ יחד עם קבצי הקוד של שתי השאלות הקודמות והקובץ students.txt.

בהצלחה!