תרגיל מס' 4 בנושא רקורסיה

שימו לב להוראות שלהלן, איך לערוך את עבודתכם ולאן לשגר אותה, ונהגו על-פיהן. הקפידו על הגשת העבודה בזמן!

אופן ההגשה:

א. יש להגיש קובץ יחיד בפורמטzip (בלבד!) שיכיל:

- לכל בעיה תוגש תוכנית בקובץ מקור (לא פרויקט) בסיומת c (ולא cpp) ששמו יתחיל ב- q ובהמשכו מספר השאלה, למשל עבור תכנית הפותרת את שאלה 2 יהיה שם הקובץ q2.c. בתרגיל זה חמש בעיות.

- מסמך word שבראשו שמות שני מגישי המטלה ומספרי הזהות שלהם, ובהמשכו, הקוד של כל תרגיל ולאחריו דוגמאות פלט לכל בעיה על-פי הנדרש בה. רשמו מעל כל דוגמת פלט לאיזו בעיה היא שייכת.

ב. את העבודה יש להגיש דרך אתר הקורס במודל:

תיקיית תרגילי בית להגשה -> הגשת שיעורי בית 4.

ג. יש להגיש את העבודה בזוגות בלבד. עבודות שיוגשו על-ידי יחידים לא ייבדקו, אלא אם כן ניתן אישור מיוחד לכך מיד עם פרסום התרגיל.

ד. שימו לב להגיש את העבודה בזמן - אל תחכו לרגע האחרון. לא תינתן דחייה של מועד ההגשה, אלא במקרים חריגים ורק לאחר אישור מראש.

ה. ניתן להוסיף פונקציות עזר – בתנאי שגם הן תהיינה רקורסיביות.

ו. ניתן לפנות בשאלות הבהרה למרצי הקורס במהלך השיעורים, בשעות הקבלה ובדוא"ל.

ז. הקפידו על עבודת צוות שבה שני הסטודנטים הם שותפים פעילים! מומלץ ששני הסטודנטים יהיו (על-פי תחושתם) בעלי ידע דומה, כדי שלא ייווצר מצב שבו האחד מבצע את רוב העבודה והשני שותף פסיבי. זכרו שתרגיל הבית נועד בראש ובראשונה לקדם את הלמידה וההבנה שלכם.

מועד הגשה אחרון: יום ב' ה- 2 ביולי בשעה 23:00.

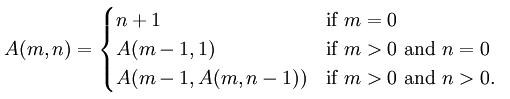
הערות כלליות

* הקפידו על כתיבה מסודרת והזחות בקוד היכן שצריך.
* בחרו שמות משמעותיים למשתנים.
* הוסיפו משפטי תיעוד למשתנים ולקטעי קוד המבצעים תת-משימות באנגלית.
* אין להשתמש בהוראה break, אלא בתוך משפט switch בלבד. אין להשתמש בהוראה return בפונקציה main() אלא בסופה בלבד.
* יש להקפיד לכתוב קוד על פי התחביר של C. למשל, לא ניתן להצהיר על משתנה בזמן אתחול משתנים בלולאת for.

אלו מהווים חלק מהקריטריונים להערכת עבודתכם.

**הערה כללית:**

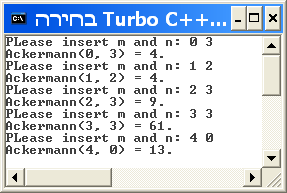
**אם רוצים להגדיר פונקציית עזר – גם היא צריכה להיות רקורסיבית.**

**תרגיל 1**:

[***פונקציית***](http://he.wikipedia.org/wiki/%D7%A4%D7%95%D7%A0%D7%A7%D7%A6%D7%99%D7%99%D7%AA_%D7%90%D7%A7%D7%A8%D7%9E%D7%9F) ***אקרמן*** מוגדרת באופן הבא:

כתבו פונקציה רקורסיבית בשם Ackermann המקבלת שני מספרים אי שליליים, ומחזירה את תוצאת A(m,n).

כתבו תכנית המבקשת וקולטת את m ו- n ומציגה את A(m,n)

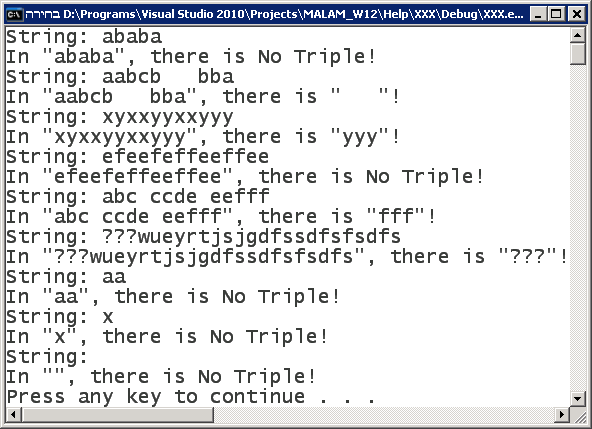


הציגו כפלטים את הדוגמאות שלהלן:

**תרגיל 2:**

כתבו תוכנית המבקשת וקולטת מחרוזת (אורך מכסימלי 30).  
התוכנית קוראת לפונקציה רקורסיבית הבודקת האם ישנו תו שמופיע במחרוזת לפחות 3 פעמים ברציפות.  
אם ישנו תו כזה הפונקציה מחזירה אותו, אחרת – הפונקציה מחזירה '\0'.

התוכנית הראשית צריכה להדפיס הודעה מתאימה – ראו דוגמא:

אין צורך לבדוק אם ישנם מספר תווים כאלו.  
בדקו עבור הקלטים שבטבלה. שימו לב, הם כוללים רווחים.

אב הטיפוס של הפונקציה char is\_tripple(char \*);

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| א. | "ababa" |  | ו. | "???wueyrtjsjgdfssdfsfsdfs" |
| ב. | "aabcb bba" |  | ז. | "aa" |
| ג. | "xyxxyyxxyyy" |  | ח. | "x" |
| ד. | "efeeefeffeeffee" |  | ט. | "" |
| ה. | "abc ccde eefff" |  | י. | "|||||||||||||||" |

הגישו כדוגמאות את הפלטים לדוגמאות ב', ג', ו'

**תרגיל 3:**

מערך של מספרים שלמים וחיוביים ממוין *oddLast* אם כל המספרים הזוגיים נמצאים בתחילת המערך מאינדקס 0 עד אינדקס m כלשהו וכל המספרים האי זוגיים ממוינים מקטן לגדול מאינדקס m+1 עד סוף המערך. הנחה: במערך שני מספרים לפחות, אחד לפחות הוא זוגי, ואחד לפחות הוא אי-זוגי.

למשל, המערך arr הבא ממוין *oddLast*:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 5 | 1 | 24 | 12 | 2 | 8 | arr |

הסבר: כל המספרים הזוגיים נמצאים בתחילת המערך מאינדקס 0 ועד m=3 , והאי-זוגיים מסודרים החל מאינדקס 4 עד סוף המערך, בצורה ממוינת מהקטן לגדול.

1. כתבו פונקציה **רקורסיבית** המקבלת מערך arr של מספרים שלמים חיוביים ממוין *oddLast* ואת אורכו n, ומחזירה את המספר האי- זוגי הקטן ביותר במערך.

כותרת הפונקציה: int minOddLast (int \*arr, int n)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 5 | 1 | 24 | 12 | 2 | 8 |

למשל, עבור המערך באורך 6 הבא: הפונקציה מחזירה 1 .

דרישות הסיבוכיות של הפונקציה minOddLast:

**סיבוכיות זמן : O(log n), סיבוכיות מקום: .O(log n)**

1. כתבו פונקציה **רקורסיבית** המקבלת מטריצה ריבועית של מספרים שלמים ואת מספר השורות / העמודות שלה, כאשר כל שורה ממוינת לפי *oddLast*. הפונקציה תחזיר את המספר האי זוגי הקטן ביותר במטריצה (הפונקציה משתמשת ב- minOddLast שמימשתם בסעיף א' ).  
   דרישות סיבוכיות של הפונקציה minMatrixOdd:

**סיבוכיות זמן : O(n\*log n), סיבוכיות מקום : .O(n)**

כותרת הפונקציה: int minMatrixOdd(int arr[][N], int n)

לדוגמא: אם הפונקציה מקבלת את המטריצה הבאה היא תחזיר 1.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 11 | 7 | 3 | 6 | 2 | 8 |
| 33 | 21 | 21 | 5 | 12 | 22 |
| 9 | 7 | 1 | 1 | 80 | 4 |
| 27 | 15 | 10 | 2 | 12 | 120 |
| 111 | 57 | 17 | 13 | 5 | 2 |
| 113 | 111 | 111 | 3 | 14 | 54 |

1. כתבו תכנית לבדיקת הפונקציות שכתבתם. בפונקציה הראשית יש להגדיר מטריצה ריבועית מסדר NxN של מספרים שלמים (הגדירו קבוע N בגודל 6), ולאתחל אותה בערכים על-פי הדוגמה. התכנית תשתמש בפונקציות שכתבתם בסעיפים א' ו-ב' למציאת הערך המינימלי.

יש להגיש פלט של הרצת התכנית עבור המטריצה שבדוגמא.

**חשיבה רקורסיבית פורייה!**