**שיעורי בית 10: רשימות מקושרות**

**הנחיות**

* יש להגיש את שיעורי הבית דרך מערכת הבדיקה האוטומטית ודרך מערכת ה-neo, כקובץ מכווץ הנושא את שמכם ומס' העבודה. לדוגמה: JohnEx10.zip.
* כל תרגיל צריך להופיע כפרויקט נפרד באותו קובץ .sln
* בכל ()main צריך להיות ()getchar בסוף התוכנית.
* במקרים בהם ישנה דוגמת ריצה – חקו אותה במדויק.
* **ודאו שכל הזיכרון שהקצתם משוחרר!**

**שאלות**

1. בשאלה זו נכיר את השימוש ברקורסיה ברשימות מקושרות - וזהו שימוש די נפוץ. מדוע?

כי את רוב הפונקציות שעוסקות ברשימות מקושרות ניתן לממש גם בעזרת רקורסיה:

* תנאי העצירה הוא שהרשימה נגמרה או שזו החולייה הרצויה – ואם כן לבצע את הפעולה.
* אם לא – לשלוח לפונקציה הרקורסיבית את החוליה הבאה (כאילו היא ראש הרשימה – כי בעצם היא ראש הרשימה שמתחילה בה).

היכנסו ל[קישור הזה](http://pythontutor.com/c.html#code=%23include%20%3Cstdio.h%3E%0A%23include%20%3Cstdlib.h%3E%0A%23include%20%3Cstring.h%3E%0A%0Atypedef%20struct%20IntNode%0A%7B%0A%20%20int%20value%3B%0A%20%20struct%20IntNode*%20next%3B%0A%7D%20IntNode%3B%0A%0AIntNode*%20createIntNode%28int%20value%29%3B%0Avoid%20insertAtEnd%28IntNode**%20head,%20IntNode*%20newNode%29%3B%0Avoid%20printList%28IntNode*%20head%29%3B%0Aint%20secret%28IntNode*%20head%29%3B%0A%0Aint%20main%28%29%0A%7B%0A%20%20IntNode*%20first%20%3D%20NULL%3B%0A%20%20IntNode*%20newNode%20%3D%20createIntNode%2813%29%3B%0A%20%20insertAtEnd%28%26first,%20newNode%29%3B%0A%20%20newNode%20%3D%20createIntNode%28422%29%3B%0A%20%20insertAtEnd%28%26first,%20newNode%29%3B%0A%20%20newNode%20%3D%20createIntNode%2851%29%3B%0A%20%20insertAtEnd%28%26first,%20newNode%29%3B%0A%20%20newNode%20%3D%20createIntNode%286%29%3B%0A%20%20insertAtEnd%28%26first,%20newNode%29%3B%0A%20%20newNode%20%3D%20createIntNode%28733%29%3B%0A%20%20insertAtEnd%28%26first,%20newNode%29%3B%0A%20%20%0A%20%20int%20mySecret%20%3D%20secret%28first%29%3B%0A%20%20printf%28%22My%20secret%20is%3A%20%25d%5Cn%22,%20mySecret%29%3B%0A%20%20return%200%3B%0A%7D%0A%0A/**%0AFunction%20will%20create%20a%20IntNode%20with%20the%20provided%20value%0Ainput%3A%0Avalue%0Aoutput%3A%0AIntNode%20with%20the%20provided%20value%0A*/%0AIntNode*%20createIntNode%28int%20value%29%0A%7B%0A%20%20IntNode*%20newIntNode%20%3D%20%28IntNode*%29malloc%28sizeof%28IntNode%29%29%3B%0A%20%20newIntNode-%3Evalue%20%3D%20value%3B%0A%20%20newIntNode-%3Enext%20%3D%20NULL%3B%0A%20%20return%20newIntNode%3B%0A%7D%0A%0A/**%0AFunction%20will%20add%20a%20number%20to%20the%20numbers%20list%0Ainput%3A%0Ahead%20-%20the%20first%20node%20of%20the%20list%0AnewNode%20-%20the%20new%20intNode%20to%20add%0Aoutput%3A%0Anone%0A*/%0Avoid%20insertAtEnd%28IntNode**%20head,%20IntNode*%20newNode%29%0A%7B%0A%20%20if%20%28!*head%29%0A%20%20%7B%0A%20%20%20%20*head%20%3D%20newNode%3B%0A%20%20%7D%0A%20%20else%0A%20%20%7B%0A%20%20%20%20IntNode*%20p%20%3D%20*head%3B%0A%20%20%20%20while%20%28p-%3Enext%29%20%0A%20%20%20%20%7B%0A%20%20%20%20%20%20p%20%3D%20p-%3Enext%3B%0A%20%20%20%20%7D%0A%20%20%20%20p-%3Enext%20%3D%20newNode%3B%0A%20%20%7D%0A%7D%0A%0A/**%0AA%20secret%20function%0Ainput%3A%20the%20list%20%28the%20first%20intNode%29%0Aoutput%3A%0Anone%0A*/%0Aint%20secret%28IntNode*%20curr%29%0A%7B%0A%20%20int%20ans%20%3D%200%3B%0A%20%20if%28curr%29%0A%20%20%7B%0A%20%20%20%20if%28curr-%3Evalue%20%25%202%20%3D%3D%201%29%0A%20%20%20%20%7B%0A%20%20%20%20%20ans%20%3D%201%20%2B%20secret%28curr-%3Enext%29%3B%0A%20%20%20%20%7D%0A%20%20%20%20else%20%0A%20%20%20%20%7B%0A%20%20%20%20%20ans%20%3D%200%20%2B%20secret%28curr-%3Enext%29%3B%0A%20%20%20%20%7D%0A%20%20%7D%0A%20%20return%20ans%3B%0A%7D&mode=edit&origin=opt-frontend.js&py=c&rawInputLstJSON=%5B%5D), חפשו את הפונקציה secret והריצו את הקוד.

ענו בקובץ word בקצרה על השאלות הבאות:

א) הפונקציה secret מונה כמה מספרים מסוג מסוים קיימים ברשימה המקושרת. מהו סוג זה של מספרים?

ב) כיצד נשנה את הפונקציה secret כך שתמנה כמה מספרים דו-ספרתיים קיימים ברשימה המקושרת?

ג) כיצד נשנה את הפונקציה secret כך שתמנה כמה מספרים כלשהם קיימים ברשימה המקושרת?

**2**. צרו מבנה (struct) בשם personNode, שיאפשר לכם ליצור רשימה מקושרת של אנשים.

למבנה זה 3 שדות: שם האדם (מחרוזת עד 20 תווים), גיל (מספר שלם), החולייה הבאה ברשימה (כפי שלמדנו בכיתה).

א) כתבו תוכנית אשר קוראת מהמשתמש את אורך הרשימה, ולאחר מכן מבקשת ממנו את השמות והגילאים של כל האנשים ברשימה. הזכרון של כל החוליות צריך להיות מוקצה בצורה דינאמית (לא לשכוח לשחרר! בפונקציה נפרדת כמובן). ניתן להניח כי כל הקלט חוקי.

ב) הוסיפו פונקציה המדפיסה את תוכן הרשימה (בכל מקרה תצטרכו כזאת על מנת לבדוק את הקוד שלכם!).

ג) הוסיפו פונקציה רקורסיבית בשם listLength המקבלת מצביע לראש הרשימה המקושרת ומחזירה את מספר האיברים ברשימה. שימו לב למקרי קצה כמו רשימה ריקה.

ד) הוסיפו פונקציה בשם copyList המקבלת מצביע לראש רשימה אחת, ומחזירה מצביע לראש רשימה חדשה שתוכנה יהיה זהה לתוכן הרשימה המקורית (אך היא תוקצה בנפרד!).

מהלך הריצה יהיה כזה:

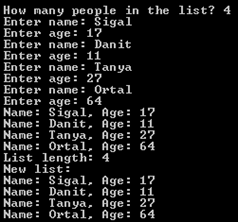
1. יצירת הרשימה המקורית.

2. הדפסת הרשימה.

3. הדפסת אורך הרשימה (בעזרת הפונקציה שלכם מסעיף ב'!).

4. יצירת העתק של הרשימה (בעזרת הפונקציה מסעיף ג') והדפסה של הרשימה המועתקת.

דוגמת ריצה:

****

3. **!MagshiParty – here we go again**

בשאלה זו נממש תור של אנשים שמחכים להיכנס למסיבה החמה ביותר בעיר ובכפר!

נרצה להשתמש במבנה Person שכתבנו בתרגול על מנת ליצור תור של בני אדם.

כתבו תוכנה המאפשרת לנהל תור למסיבה. למשתמש יינתנו האפשרויות הבאות:

א) הדפסת התור ואורכו (השתמשו לשם כך בקוד משאלה 2).

ב) הוספה של אדם חדש לתור. שימו לב – במקרה שלנו מדובר ב"תור ישראלי".

איך עובד תור ישראלי? כשאורח חדש מצטרף לתור, הוא מתווסף לסוף התור. אלא אם כן יש לו חברים שנמצאים כבר בתור, ואז הוא ייכנס מיד אחרי החבר שנמצא במקום הקדמי ביותר בתור. לכן, באפשרות זו, אפשרו לאורח להקליד את שמותיהם של 3 חברים – אם אחד מהם/ן נמצא/ת כבר בתור – האדם החדש ייכנס מיד מאחוריהם.

[הידעת?! יש מאמרים מדעיים במדעי המחשב שעושים שימוש בקונספט של "תור ישראלי", [הנה אחד מהם](https://rapidapi.com/blog/israeli-queues-exploring-a-bizarre-data-structure/)].

ג) הוצאה של אדם מהתור – אם מישהו בחר לעזוב את התור, מכל סיבה שהיא, אנחנו רוצים שתהיה אפשרות להסיר אותו מהתור לפי השם שלו. במידה ואין אדם בשם זה בתור – הפונקציה לא תעשה דבר ותעדכן את המשתמש. במידה והאדם נמצא – הפונקציה תעדכן את המשתמש שהוא הוסר.

ד) VIP – במידה ויבוא אח"מ כמו הרמטכ"ל, הילה מנהלת התוכנית, או גיבל, נרצה למקם אותם אוטומטית בראש התור. במידה ואופציה זו נבחרת, פרטיו של האורח החדש ייקלטו והוא יוצנח ישירות לראש התור.

ה) חיפוש בתור – הפונקציה תקרא שם מהמשתמש ותבדוק האם מישהו בשם זה נמצא בתור.

ו) "איך שגלגל מסתובב לו" – כתבו פונקציה שהופכת את סדר הרשימה. האיש בראש התור הופך להיות האחרון (ויצביע על NULL) והאחרון ברשימה יהפוך להיות הראשון. פתרו את השאלה עם מעבר אחד בלבד על הרשימה המקורית, ובלי ליצור רשימה חדשה!

ז) יציאה.

אל תשכחו להתחשב בכל מקרי הקצה (רשימה ריקה, הסרת אדם שלא נמצא או האדם בראש התור, וכו').

ניתן להניח שכל הקלט תקין.

דוגמת ריצה: ראו את הקובץ המצורף MagshiParty\_queue.exe.

**בונוס:** ממשו את הפונקציות רקורסיבית!

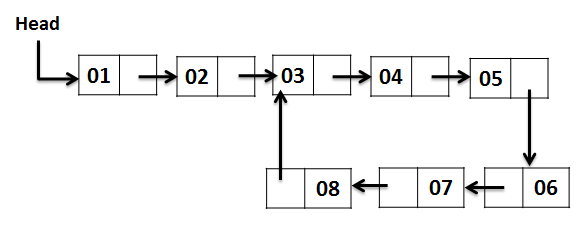
4. **בונוס:**

**This is such a great Bonus! It will give you so much cyber knowledge. Great!**

דניאל החניך המסוגל (ראיתם איזה רקורסיה הוא הפגיז?) לא בטוח אם הוא יצר לולאה ברשימה המקושרת שלו.

לולאה ברשימה מקושרת היא מצב שבו המצביע של אחת החוליות מצביע לחוליה קודמת בשרשרת. במצב בו אין לולאה, הרשימה מסתיימת כרגיל ב-NULL.

דוגמא ללולאה:



בהינתן רשימה מקושרת, יש לכתוב פונקציה הבודקת אם הרשימה המקושרת מכילה לולאה. על הפונקציה להשתמש בכמות קבועה של זכרון, שאינה תלויה באורך הרשימה המקושרת. זאת אומרת, לא משנה מהו אורך הרשימה, הפונקציה עדיין תקצה את אותה כמות זיכרון. השלימו את הקובץ LinkedListLoop.c.

**בהצלחה!**