

מעבדה מתקדמת ב-C

תרגיל 1

נהלי הגשה יפורסמו לקראת הדדליין.

חובה להגיש בזוגות למעט מקרים שתואמו פרטנית מול סגל הקורס.

תזכורת: התרגילים נבדקים אוטומטית בעזרת מערכת לזיהוי העתקות, כולל מול הגשות מסמטרים קודמים.

בתרגיל זה, נתרגל את עקרונות התכנות הנכון שלמדנו עם פרוייקט פשוט. הפרוייקט מתחלק לשני חלקים (ולכן כנראה שהקוד יתחלק לשני מודולים, אולי עם מודול main בנוסף):

1. מימוש פשוט של רשימה מקושרת
2. קוד שקורא קלט מהמשתמש ומתרגם אותו לפעולה על הרשימה המקושרת

הקפידו למלא אחר ההנחיות לאיכות קוד גבוהה, כפי שניתנו בהרצאה ובמסמך איכות הקוד.

שימו לב: יכול להיות שצריך מודול נפרד עבור main, ויכול להיות שלא. זאת החלטה שלכם, וכל החלטה היא סבירה פה, כשהמבחן הוא האם יש משהו נוסף מלבד מה שרשום בנקודה 2 למעלה – אם יש, אז הוא יכול להיות בקובץ main, ואם אין, אז כנראה שלא צריך קובץ main. בכל מקרה, אין טעם לכתוב קובץ main שכל מה שיש בו זאת פונקציה שפשוט קוראת לפונקציה מקובץ אחר וזהו.

ולהלן פירוט:

1. הרשימה המקושרת: עליכם לממש רשימה מקושרת, המחזיקה איברים מסוג int. אתם יכולים לבחור בין רשימה מקושרת חד-כיוונית או דו-כיוונית. המימוש צריך להיות נקי, כאשר ברור משמות הפונקציות ומהארגומנטים שלהן מה הפונקציות עושות.
2. קוד הקלט: הקוד יקבל קלט המחולק לשורות. אין צורך להתגונן משורות ארוכות מדי או "מתוחכמות" – הקצו זיכרון של 100 תווים (בעזרת קבוע ניתן לשינוי!), ויהיה בסדר (זה כמובן לא נכון בכל סיטואציה בה אתם מתכנתים, רק הפעם...). כל שורה מכילה אחת מהפקודות הבאות, שמופעלות על הרשימה בזו אחר זו (מתחילים מרשימה ריקה כמובן):

Add_end i – insert element I at end of list

Add_start i – insert element I at start of list

Add_after i j – insert element I after the first occurrence of element j. If j is not found, print an error and exit the program

Index i – print the first index where element I is found, -1 if not found. The first element is index 0, the second is index 1, etc.

Del index – remove the element at the specified index. If index is too large, print an error and exit the program

Print – print the entire list. An empty list is printed as “[]”. A list with one element whose value is 1 is printed as “[1]”. A list with 3 elements, 1-3, is printed as “[1, 2, 3]”, with a space after each comma

Exit – free the list, and exit the program

שימו לב שהפקודות יכולות להופיע בכל capitalization שהוא, כלומר כל הפקודות הבאות חוקיות וזהות:

Add_start 1

ADD_START 1

add_start 1

aDd_StArT 1

לדוגמה, הנה מה שיקרה ברצף הפקודות הבאות:

Add_end 1 // list is now [1]

Add_start 2 // [2, 1]

Add_after 3 2 // [2, 3, 1]

Index 3 // list is unchanged, result is 1 (remember – indices start at 0)

Del 2 // [2, 3]

Print // list is unchanged, prints [2, 3]

Exit // Hooray, we're done!

ניתן להניח שהקלט תקין, ולא צריך לבדוק את תקינותו. ניתן להניח שהקלט יסתיים בפקודה Exit. כמו כן, ניתן להניח שהקצאות זכרון לא נכשלות, כפי שיקרה בכל סיטואציה סבירה בה תריצו תוכנית כ"כ קטנה על מחשב מודרני.

ניתן להשתמש בכל התכונות של שפת C שמוגדרות בסטנדרט C99, ורק בהן. בפרט, אי אפשר להשתמש בפונק' strtok_s.

הערה לגבי exit: זה כמובן הגיוני שמימוש של רשימה מקושרת יכול ל פונקציה שמשחררת את הרשימה. אבל, זה לא הגיוני שאותה פונקציה גם תצא מהתוכנית. אם הפונקציה שמשחררת גם יוצאת, מה נעשה אם יש לנו שתי רשימות, ואנחנו רוצים לשחרר אחת ולהמשיך את ריצת התוכנית? הדרך ההגיונית לטפל בפקודה exit היא לקרוא לפונקציה של

הרשימה שמשחררת אותה, ושהקוד שמטפל בפקודה יקרא בנוסף לפונקציית המערכת exit.
שימו לב שאין שום סיבה שפונקציה שיוצאת מהתוכנית תימצא במודול של הרשימה
המקושרת – זאת לא פונקציות שמודול של רשימה מקושרת אמור לספק.

בהצלחה!