תרגיל מס׳ 4 - C - פונקציות ורקורסיה ללא מערכים (מתרגל אחראי: אייל)

התרגיל שווה 14% מציון התרגול

הוראות הגשה

שאלות בנוגע לתרגיל נא להפנות דרך פורום הקורס שנפתח במיוחד לשם כך:
https://piazza.com/biu.ac.il/fall2017/89110/.

אם לא נענתה תשובה תוך 24 שעות, נא לשלוח אלי (אייל) מייל עם לינק לדיון הרלוונטי ואענה. המייל הוא: eyal.dayan@live.biu.ac.il. בכל מייל יש לציין **שם, שם משתמש, מס' קורס, וקבוצת תרגול.**

- מועד פרסום: 26/11/17
- מועד אחרון להגשה: 23:50 10/12/17
- יש לשלוח את הקבצים באמצעות מערכת ההגשה לפני חלוף התאריך הנקוב לעיל.
- אין הארכות (למעט מילואים), אך ניתן להגיש באיחור עם קנס (עד יומיים).
  יום איחור גורר הורדה אוטומטית של 10 נקודות.
  יומיים איחור גוררים הורדה אוטומטית של 20 נקודות.

- שם ההגשה של התרגיל: ex4
- יש להקפיד מאוד על כל הוראות עיצוב הקלט והפלט, כמפורט בכל סעיף וסעיף. על הפלט להיראות בדיוק כמו בדוגמאות. **אין להוסיף או להשמיט רווחים או תווים אחרים ואין להחליף אותיות גדולות בקטנות או להיפך. אי-הקפדה על פרטים אלה עלולה לגרור ירידה משמעותית ביותר בציון התרגיל עד כדי 0. ראו הוזהרתם!**
  טיפ - אפשר להגיש קובץ ריק למערכת ההגשה ולהעתיק מהמייל החוזר את מחרוזות הפלט. אל תעתיקו מתוך מסמכי Pdf ומצגות.
- להזכירכם, העבודה היא אישית. "עבודה משותפת" דינה כהעתקה.
- אין להדפיס למסך שום דבר מעבר למה שנתבקש בתרגיל.
- יש לוודא שהתרגיל מתקמפל ורץ על השרתים באוניברסיטה (u2) ללא שגיאות/אזהרות.
- אתם יכולים לעבוד עם כל עורך טקסטואלי שאתם מעדיפים. להזכירכם – pico בשרתי linux שבמעבדות; notepad ב-windows ; או בסביבת פיתוח ייעודית.

הקפידו על כתיבה לפי קובץ ה-Coding-Style שבאתר הקורס!

- **הקובץ עודכן וכעת הוא כולל הנחיה לגבי השארת שורות ריקות. החל מהתרגיל הזה הבדיקה תתייחס לכך (בתרגילים הקודמים לא יורדו נקודות על זה, אם הורדנו לכם בטעות אז תגישו ערעור ונחזיר את הנקודות).**
- **קראו את המסמך בתשומת לב. אין הרבה שינויים אבל חבל ליפול על הנושא הזה.**

הקדמה

בתרגיל זה תגישו שלושה קבצים:
ex4.c - קובץ מקור שיכיל את המימושים של כל הפונקציות שלכם, למעט פונקציית הmain.
ex4.h - קובץ header שיכיל את ההצהרות של הפונקציות שאתם מעוניינים לחשוף כלפי חוץ (כלומר כל הפונקציות ששמותיהן מופיעים בסעיפים השונים של המסמך הזה, למעט הפונקציה בסעיף 2 שהיא למעשה פונקציית עזר עבור סעיף 3).
main.c - קובץ מקור שיכיל את פונקציית הmain שמריצה את התרגיל. הקובץ **מצורף לתרגיל** ואין צורך לשנות בו כלום, פשוט להגיש אותו יחד עם הקבצים הנוספים. אתם יכולים לשנות את הקובץ כדי לבדוק את עצמכם, אך כדי לעבור את הבדיקה האוטומטית הראשונית יש להגיש את הקובץ כפי שהוא מפורסם (אם תעיינו בקובץ תוכלו להבין מדוע אין קלט בבדיקה האוטומטית).

פקודת הקימפול **בדרך כלל** למי שבודק על השרת: gcc ex4.c ex4.h main.c -ansi -pedantic -lm
פקודת הקימפול **בתרגיל הזה** למי שבודק על השרת: gcc ex4.c ex4.h main.c -lm

כלומר, בתרגיל הזה ניתן (למשל) להצהיר על משתנה הלולאה בתוך ההצהרה [for (int i = 0; i < size; i++)], אין צורך להגדיר את כל המשתנים בתחילת הבלוק, ניתן להשתמש בהערות שורה וכו'. בתרגילים הבאים - עקבו אחרי ההוראות (נשתדל שזה יישאר כך).

הנחיות כלליות לתרגיל

הנחיות עבור סעיפים שבהם אתם נדרשים לממש פונקציות לא רקורסיביות:

- **אין להשתמש** ברקורסיה. ניתן להגדיר פונקציות עזר לא רקורסיביות ולהשתמש בהן (על פונקציות כאלה אין להצהיר בקובץ ex4.h מפני שהשימוש בהן הוא פנימי בקובץ ex4.c, והן לא נקראות מתוך פונקציית הmain).
- אם אתם מרגישים שאתם זקוקים למשתנה גלובלי או סטטי, יש להעדיף שימוש במשתנה סטטי על פני משתנה גלובלי **ולהסביר היטב בתיעוד מדוע יש צורך במשתנה כזה**. הסבר לא מספק יגרור הפחתת נקודות, לכן מומלץ להוריד למינימום את השימוש במשתנים כאלה.

הנחיות עבור סעיפים שבהם אתם נדרשים לממש פונקציות רקורסיביות:

- **אין להשתמש** בלולאות. ניתן להגדיר פונקציות עזר (רקורסיביות או לא רקורסיביות) אך גם בהן אין להשתמש בלולאות.
- אם אתם מרגישים שאתם זקוקים למשתנה גלובלי או סטטי, נסו להרגיש משהו אחר משום שהשימוש במשתנים כאלה בסעיפים האלו **אסור בהחלט.**
- טכניקה נפוצה עבור שאלות רקורסיביות היא להתאים את הפרמטרים של השאלה לקריאה רקורסיבית. במקרה כזה הפונקציה שמוגדרת לכם רק עושה את התרגום של הפרמטרים הראשוניים, וקוראת לפונקציית עזר רקורסיבית (כלומר – העובדה שפונקציה נקראת רקורסיבית בתרגיל הזה לא מחייבת אתכם לבצע בתוכה קריאה רקורסיבית לעצמה, אלא רק מחילה עליה את ההגבלות שבשני הסעיפים הקודמים).

הנחיות עבור כל הסעיפים:

- אין להשתמש בספריות חיצוניות, למעט בסעיף 4, שם הותר להשתמש בפונקציה pow.
- שימו לב להנחות ולאיסורים המפורטים בכל סעיף.

ההנחיות המופיעות בעמוד הראשון של המסמך הזה לא מופיעות שם סתם, אז זוהי תזכורת:

- שאלות **בנושא התרגיל** יש לשאול **בפיאצה** באופן **ציבורי בלבד.** אם השאלה שלכם נותרה ללא מענה לאחר 24 שעות (מה שלא סביר שיקרה), רק אז יש לשלוח אלי מייל עם לינק לדיון שלא קיבל מענה.
- שאלות אישיות ניתן לשאול בפיאצה גם כן ולפרסם אותן באופן **פרטי** רק למתרגלים. נא לא לשאול סתם שאלות באופן פרטי.
- אם כבר שלחתם מייל מסיבה כלשהי – נא לציין בסופו את שמכם, שם המשתמש שלכם, מספר קורס וקבוצת תרגול (אני למשל מתרגל בעוד מקומות ולכן זה מאוד מבלבל, ובסך הכל גורם לי לדחות את המענה למייל כזה).
- גם בפיאצה וגם במייל – הסבירו את השאלה/הבעיה שלכם **במילים.** אם אתם ממש ממש (ממש!) חייבים לצרף תמונה כי בלתי אפשרי להבין את הבעיה בלי זה, צרפו אותה כקובץ ולא כתמונה כדי שיהיה אפשר לקרוא את מה שכתבתם (ושהתמונה לא תהיה מסובבת הצידה...). **הקפידו ליישר את הטקסט** כמו שצריך: לימין (כשאתם כותבים בעברית) ולשמאל (כשאתם כותבים באנגלית, או מצרפים שורת קוד).

ההנחיות האלו תקפות מעכשיו ועד **סוף השנה** (גם עבור הקורס בתכנות מונחה עצמים), והן לא קיימות כדי להציק לכם אלא כדי להקל עלינו, ולעזור לנו לתת לכם מענה מהיר ומקצועי ככל האפשר. תודה.

בהצלחה!

1. ממשו את הפונקציה void Collatz(long int n) המקבלת מספר שלם מסוג long ומדפיסה את סדרת המספרים המתקבלת מהפעלת החוקיות של השערת קולץ המתוארת כאן. בנוסף הפונקציה מדפיסה את מספר הצעדים שנדרשו לה לעבוד בהרצה הנוכחית, וגם את מספר הצעדים שנדרשו לה מרגע ההרצה של התוכנית באופן כללי.
   - ניתן להניח שהפרמטר המועבר לפונקציה הזו יהיה קלט כזה שעבורו ההנחה מתקיימת.
   - ניתן להניח שהפרמטר המועבר לפונקציה יהיה חיובי וגדול ממש מאפס.

דוגמת הרצה עבור תוכנית main לדוגמה:

```
int main() {
   Collatz(7);
   Collatz(6);
   return 0;
}
```

```
7->22->11->34->17->52->26->13->40->20->10->5->16->8->4->2->1
num of steps: 16
total num of steps: 16
6->3->10->5->16->8->4->2->1
num of steps: 8
total num of steps: 24
```

2. ממשו את הפונקציה unsigned long int CollatzNoPrint(long int n) אשר מקבלת מספר שלם מסוג long ובודקת האם ההשערה מתקיימת עבורו באופן הבא – אם מספר הצעדים הנדרש קטן מהגבול העליון של unsigned long int (ניתן למצוא את הנתון הזה בגוגל) אז הפונקציה תחזיר את מספר הצעדים שנדרשו לה על מנת לבדוק את המספר. אם נדרשים יותר צעדים הפונקציה תעצור את הבדיקה ותחזיר 0.
שימו לב - הפונקציה לא מדפיסה דבר.

   - ניתן להניח שהפרמטר המועבר לפונקציה יהיה חיובי וגדול ממש מאפס.

3. ממשו את הפונקציה void ProofCollatzUntill(long int n) על מנת לבדוק (בעזרת הפונקציה מהסעיף הקודם) את כל המספרים מ1 ועד הקלט של הפונקציה (לא כולל).

דוגמת הרצה עבור תוכנית main לדוגמה:

```
int main() {
   ProofCollatzUntill(6);
   return 0;
}
```

```
passed: 1 (num of steps: 0)
passed: 2 (num of steps: 1)
passed: 3 (num of steps: 7)
passed: 4 (num of steps: 2)
passed: 5 (num of steps: 5)
```

במידה ומצאתם מספר (ניקח לדוגמה את המספר 3) שעבורו החלטתם לעצור את הפונקציה CollatzNoPrint אז יש להדפיס:

```
passed: 1 (num of steps: 0)
passed: 2 (num of steps: 1)
you might wanna check 3 and win 500$ :)
passed: 4 (num of steps: 2)
passed: 5 (num of steps: 5)
```

4. רקע לשאלה - בקישור כאן (מספיק לקרוא את פסקת הפתיחה).
ממשו את הפונקציה void Fermat(int n, int stop) המקבלת חזקה n וחסם stop.
הפונקציה מדפיסה את כל השלשות $a^n + b^n = c^n$ הנמצאות בטווח [1,stop] ממויינות מיון ראשי לפי a ומיון משני לפי b.

- לצורך הסעיף הזה מותר להשתמש בפונקציה pow הנמצאת בספרייה math.h
- השלשה היא בתוך הטווח אם כל שלושת המספרים a,b,c המרכיבים אותה נמצאים בתוך הטווח.
- ניתן להניח כי לא תוכנס חזקה שלילית.

דוגמת הרצה עבור תוכנית main לדוגמה:

```
int main() {
   Fermat(2, 20);
   return 0;
}

Eureka! 3^2 + 4^2 = 5^2
Eureka! 5^2 + 12^2 = 13^2
Eureka! 6^2 + 8^2 = 10^2
Eureka! 8^2 + 15^2 = 17^2
Eureka! 9^2 + 12^2 = 15^2
```

במידה ולא נמצאה אף שלשה בתוך הטווח, יש להדפיס: meh :(

הבהרה:
אין להשתמש בהוכחת המשפט התיאורטי על מנת לייעל את הקוד. מטרת התרגיל היא לאשש את ההוכחה המתמטית בעזרת בדיקה בפועל של השוויון עד גבול מסויים.

דוגמת הרצה עבור תוכנית main לדוגמה:

```
int main() {
   Fermat(3, 1000000);
   return 0;
}

meh :(
```

5. ממשו את הפונקציה <u>הרקורסיבית</u> int IsDividedBy3Rec(long int n) המקבלת מספר שלם מסוג long ומחזירה 1 אם הקלט מתחלק ב3 ומחזירה 0 אחרת.

- ניתן להניח כי המספר המתקבל כקלט מכיל אך ורק ספרות מהקבוצה {1,2,3} (ולא מכיל ספרות מהקבוצה {4,5,6,7,8,9,0}).
- **<u>ניתן להשתמש</u>** בפעולות חילוק ב10 ומודולו 10.
- מעבר לכך, **<u>אין להשתמש</u>** בפעולות חשבון כלל (חיבור,חיסור,כפל,חילוק,מודולו).

המלצה חמה עבור הסעיף הזה – **<u>השתמשו</u>** בפונקציית עזר (לא חובה למי שלא רוצה).

דוגמת הרצה עבור תוכנית main לדוגמה:

```
int main() {
   printf("%d\n", DivideBy3Rec(123232323231));
   printf("%d\n", DivideBy3Rec(123232323232));
   return 0;
}

1
0
```

6. ממשו בנוסף לסעיף הקודם גם את הפונקציה int IsDividedBy3Iter(long int n) המקבלת מספר שלם מסוג long ומחזירה 1 אם הקלט מתחלק ב3 ומחזירה 0 אחרת.
   - **<u>ניתן להשתמש</u>** בפעולת חילוק ב10 ומודולו 10 ובפעולת החיבור.
   - מעבר לכך, **<u>אין להשתמש</u>** בפעולות חשבון כלל (חיסור,כפל,חילוק,מודולו).

דוגמת הרצה עבור תוכנית main לדוגמה:

```
int main() {
   for (int i = 0; i < 100000000; i++) {
      if (DivideBy3Iter(i) != (i%3 == 0)) {
         printf("%d\n", i);
      }
   }
   return 0;
}

// No Output
```

- נסו לכתוב תוכניות בדיקה כאלו גם עבור הסעיפים האחרים במקום לבדוק את עצמכם עם מקרים פרטיים באופן סיזיפי. 😉
- ניתן להשתמש באותה התוכנית על מנת לבדוק גם את הסעיף הקודם. שאלה למחשבה: האם אתם חושבים שיהיה הבדל בזמן הריצה?

7. בסעיף הזה ניתן להניח כי המספרים המועברים כפרמטר חיוביים, גדולים מאפס, ומורכבים מספרות **ממויינות בסדר עולה משמאל לימין**.

ממשו את הפונקציה הרקורסיבית int FindCommonDigit(long int n1, long int n2) המקבלת שני מספרים שלמים מסוג long ומחזירה 1- אם אין להם אף ספרה משותפת, אחרת הפונקציה מחזירה את הספרה המשותפת הימנית ביותר.

דוגמת הרצה עבור תוכנית main לדוגמה:

```
int main() {
   printf("%d\n", FindCommondigit(1113355579999, 22222444466668888));
   printf("%d\n", FindCommondigit(1113555777777, 22222344446666899));
   printf("%d\n", FindCommonDigit(111222333444555, 111222333444555));
   return 0;
}
```

```
-1
3
5
```

8. ממשו את הפונקציה הרקורסיבית int CountDigit(long int n, int d) המקבלת מספר שלם מסוג long וספרה בודדת, ומחזירה את מספר הפעמים שהספרה מופיעה במספר.

- ניתן להניח כי המספר שיוכנס חיובי וגדול מאפס.

דוגמת הרצה עבור תוכנית main לדוגמה:

```
int main() {
   printf("%d\n", CountMyDigit(125827620123, 2));
   printf("%d\n", CountMyDigit(123432123432, 7));
   return 0;
}
```

```
4
0
```

9. ממשו את הפונקציה הרקורסיבית void PrintReverse(long int n) המקבלת מספר שלם מסוג long ומדפיסה אותו הפוך.
דוגמת הרצה עבור תוכנית main לדוגמה:

```
int main() {
   PrintReverse(123456789);
   PrintReverse(111222333);
   return 0;
}
```

```
987654321
333222111
```