# הנחיות כלליות

יש לשלוח את הקבצים באמצעות מערכת ההגשה לפני חלוף השעה **23:50** בתאריך **29.11.20**.

ניתן להגיש את התרגיל באיחור עם קנס אוטומטי על פי הפירוט הבא:

- יום איחור – קנס של **10 נקודות** (ציון מקסימלי – 90).
- יומיים איחור – קנס של **20 נקודות** (ציון מקסימלי – 80).
- שלושה ימי איחור – קנס של **30 נקודות** (ציון מקסימלי – 70).

לאחר מכן לא ניתן יהיה להגיש את התרגיל (ציון 0).

המתרגלת האחראית על התרגיל היא רעות.

שאלות בנוגע לתרגיל יש לפרסם **באופן ציבורי** בפורום הקורס בלבד.

בקשות להארכה מסיבות מוצדקות (מילואים, לידה, אשפוז וכו׳) יש לפרסם **באופן פרטי** בפורום הקורס בלבד (יש למען את הפוסט ל-Instructors). בכל בקשה יש לציין: שם מלא, שם משתמש במערכת ההגשה, מספר תעודת זהות, האם אתם ממדעי המחשב או ממתמטיקה.

יש להקפיד מאוד על הוראות עיצוב הקלט והפלט, בדיוק על פי הדוגמאות המצורפות. אין להוסיף או להשמיט רווחים או תווים אחרים, ואין להחליף אותיות גדולות בקטנות או להיפך. חוסר הקפדה על פרטים אלו עלול לגרור הורדה משמעותית ביותר בציון התרגיל עד כדי 0. ראו עצמכם הוזהרתם!

**שימו** ❤️ שאתם עוקבים במדויק אחרי ההנחיות במסמך ה-Coding Style המפורסם בפורום הקורס.

עליכם לכתוב קוד על פי הוראות התרגיל ולוודא שקיבלתם 100 בבדיקה האוטומטית הראשונית, וכן שהתרגיל מתקמפל ורץ על שרתי המחלקה (u2) **ללא שגיאות וללא אזהרות**.
תרגיל שלא עומד בסטנדרטים הבסיסיים הללו יגרור, בשל הטרחה שהוא מייצר בתהליך הבדיקה שלו, הורדת נקודות משמעותית בציון שלו.

להזכירם העבודה היא אישית. ״עבודה משותפת״ דינה כהעתקה. העתקות נבדקות על ידי מערכת ההגשה האוטומטית, ותרגיל שהועתק יגרור ציון 0 ופגיעה בציוני התרגול הסופיים **לכל הגורמים** השותפים בהעתקה. אתם יכולים לדון בגישות לפתרון התרגיל באופן תיאורטי, אך אין לשתף קוד בשום צורה.

בפיתוח הקוד ניתן להשתמש בכל סביבת עבודה, העיקר הוא שתדעו איך לקחת את קבצי הקוד מתוך הסביבה הזו, לבדוק אותם על שרתי האוניברסיטה, ולהגיש אותם באמצעות מערכת ההגשה.

דוגמאות לחלק מסביבות העבודה האפשריות:

IDEs (Integrated Development Environment):

- Visual Studio
- Clion
- Eclipse
- Xcode

Text Editors:

- Sublime Text
- Atom
- Notepad++
- Vim

בהצלחה!

# תרגיל 3 – Ex3

משקל התרגיל מתוך ציון התרגול: **10%**.

בתרגיל זה עליכם לממש פונקציות בקובץ יחיד בשם ass3.c.

לתרגיל זה מצורף קובץ בשם ass3.h המכיל הצהרות. בנוסף מצורף קובץ בשם main.c המכיל פונקציית main ופונקציות נוספות, אין להגדיר פונקציות בשמות האלו בקובץ ass3.c שאתם מגישים. יתרה מכך, אין צורך להגיש את הקבצים המצורפים.

**הנחיות כלליות לתרגיל**

סעיפים שבהם אתם נדרשים לממש פונקציות לא רקורסיביות:

- אין להשתמש ברקורסיה. ניתן להגדיר פונקציות עזר לא רקורסיביות ולהשתמש בהן. (על פונקציות כאלו אין סיבה להצהיר בקובץ ex3.h מפני שהשימוש בהן הוא פנימי, רק בתוך הקובץ ex3.c.
- אם אתם מרגישים שאתם זקוקים למשתמש גלובלי או סטטי, יש להעדיף שימוש במשתנה סטטי על פני משתנה גלובלי, ולהסביר היטב בתיעוד מדוע יש צורך במשתנה כזה. הסבר לא מספק או לא מוצדק יגרור הפחתת נקודות, ולכן מומלץ להימנע לגמרי מהשימוש במשתנים אלו.

סעיפים שבהם אתם נדרשים לממש פונקציות כן 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. השימוש במשתנים כאלו בסעיפים האלו אסור בהחלט (חוץ משאלה 2 בה מותר להשתמש במשתנה גלובלי אחד, לא סטטי!).
- טכניקה נפוצה עבור שאלות רקורסיביות היא להתאים את הפרמטרים של השאלה לקריאה רקורסיבית. במקרה כזה הפונקציה שמוגדרת לכם יכולה לא להיות רקורסיבית, בתנאי שהיא קוראת לפונקציית עזר רקורסיבית ופשוט עושה רק את התרגום של הפרמטרים שהיא מקבלת לפרמטרים של פונקציית העזר שהגדרתם.

הנחיות עבור כל הסעיפים:

- אין להשתמש בספריות חיצוניות גם לא מה שנלמד בתרגול כולל get buffer, getchar, string.h, מערכים, הקצאות דינמיות, פוינטרים וכל מה שלא נלמד (חוץ מ stdio.h ).
- שימו לב להנחיות ולאיסורים המפורטים בכל סעיף.

בהצלחה!

שאלת פתיחה-בונוס :

מומלץ להיעזר בפסאודו קוד מויקיפדיה ולעשות אותה לפני שאר השאלות של הרקורסיה כמעין נחיתה רכה לקראת שאלות החובה.

מגדלי האנוי :

במקדש בעיר בנארס בהודו יש 64 דיסקים. הדיסקים העגולים משתנים בגודלם והם עולים לפי הסדר מהקטן ביותר לגדול ביותר (כלומר הכי גדול נמצא בתחתית).

מספר נזירים קדושים קיבלו את משימת האל להעביר את המגדל למקום חדש במקדש. כאשר הם עושים זאת עליהם לציית לכמה כללים קדושים.

1. ניתן להעביר את הדיסקים אך ורק ל-3 מקומות מסומנים במקדש. המקום הראשון הוא המקום בו היה המגדל לפני שהתחילו להזיזו, המקום השני הוא היעד הסופי, והמקום השלישי הוא באמצע בין המקום ההתחלתי ליעד הסופי.

2. הדיסקים כה קדושים כך שמתוך זהירות יש להעביר דיסק אחד בכל פעם.

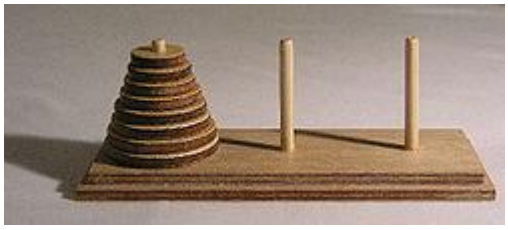
3. בשום נקודת זמן דיסק לא יכול להיות מונח על דיסק הקטן ממנו, אבל ניתן למקם כל דיסק בכל אחד משלושת המקומות.
כאשר הנזירים יצליחו האגדה מספרת שיגיע סוף העולם.

קישור לעוד הסברים: לחצו כאן

עליכם לממש את הפונקציה :

void towerOfHanoi(int numDisks, char fromRod, char toRod, char auxRod)

המקבלת את כמות הדיסקים ושלושה תווים המייצגים 3 מוטות, ומדפיסה את הדרך בה יש להזיז את הדיסקים בין המוטות



דוגמת הרצה :

```
int main()
{
    int n = 2; // Number of disks
    towerOfHanoi(n, 'A', 'C', 'B');  // A, B and C are the names of the rods
    return 0;
}
```

פלט :

```
Move disk 1 from rod A to rod B.
Move disk 2 from rod A to rod C.
Move disk 1 from rod B to rod C.
```

**מכאן כל השאלות הן חובה!**

1. ממשו את הפונקציה הרקורסיבית void isPalindrome(char str[],int len).

הפונקציה מקבלת מערך של תווים ומספר שלם מסוג int (אורך המחרוזת) ומדפיסה האם המחרוזת היא פולינדרום (תוכלו למצוא הסבר מה הוא פולינדרום כאן).

דוגמת הרצה:

```
int main()
{
    char str1[] = "malayalam";
    isPalindrome(str1,9);
    char str2[] = "sun";
    isPalindrome(str2,3);
    return 0;
}
```

והפלט:

```
The reverse of malayalam is also malayalam.
The reverse of sun is not sun.
```

2. ממשו את הפונקציה הרקוסיבית void printAllCombinations(char pattern[],int len).

הפונקציה מקבלת מספר שלם מסוג int (אורך המחרוזת-len) ומחרוזת המורכבת מהא״ב הבא: {0,1,2} ומהתו הפראי ״?״(pattern), ומדפיסה למסך את כל הקומבינציות האפשריות של המחרוזת ע״י החלפה של התו הפראי באחת מאותיות הא״ב, ואת כמות הקומבינציות שהיא מצאה.
הקומבינציות האפשריות יודפסו מהמספר הקטן לגדול, שימו לב!

**לא יבדקו מקרים שלא מורכבים מהא״ב הנתון!**

דוגמת הרצה:

```
int main()
{
    char pattern[] = "1?12?";
    printAllCombinations(pattern,5);
    return 0;
}
```

והפלט:

```
10120
10121
10122
11120
11121
11122
12120
12121
12122
Number of combinations is: 9
```

3.ממשו את הפונקציה הרקורסיבית void powRec (long int firsrtNum, long int secondNum).

הפונקציה מקבלת 2 מספרים מסוג long int ומדפיסה למסך את firsrtNum בחזקת secondNum .
ההדפסה תהיה בדיוק של 6 ספרות אחרי הנקודה.

דוגמת הרצה:

```
int main()
{
    powRec(-2, 3);
    return 0;
}
```

והפלט:

```
The result is -8.000000.
```

4. ממשו את הפונקציה הרקורסיבית void isDivisibleBy3(long long n).

הפונקציה מקבלת מספר מסוג long long ומדפיסה האם הוא מתחלק ב-3.

עבור תרגיל זה ניתן להניח כי המספר המתקבל מורכב מספרות {1,2,3} בלבד. אין להשתמש בפעולות חשבון כלל (לדוגמא חיבור, חיסור, כפל, חילוק ,מודלו וכו) **למעט חילוק ב10 ומודלו 10.**
(כולל, גם ++ אסור)

דוגמת הרצה:

```
int main()
{
    isDivisibleBy3(123232323231);
    isDivisibleBy3(123232323232);
    return 0;
}
```

והפלט:

```
The number 123232323231 is divisible by 3.
The number 123232323232 is not divisible by 3.
```

5. ממשו את הפונקציה הרקורסיבית void gcd(long int n1, long int n2).

הפונקציה מקבלת שני מספרים שלמים מסוג long int ומדפיסה למסך את המחלק המשותף הגדול ביותר של המספרים (תוכלו למצוא הסבר על gcd ושיטות למציאתו כאן).

בנוסף הפונקציה מדפיסה את הדרך בה חישבתם את התוצאה.

דוגמת הרצה:

```
int main()
{
    gcd(105, 51);
    return 0;
}
```

פלט:

```
51*2+3 = 105 (a=105, b=51)
3*17+0 = 51 (a=51, b=3)
GCD = 3
```

6. ממשו את הפונקציה הרקורסיבית void countDigit(long long n, int d).

הפונקציה מקבלת מספר שלם מסוג long long וספרה בודדת ומדפיסה למסך את מספר הפעמים שהספרה מופיעה במספר (ניתן להניח כי המספר שיוכנס חיובי).

דוגמת הרצה:

```
int main()
{
    countDigit(1234567891,1);
    return 0;
}
```

הפלט:

```
1234567891 has 2 times 1.
```

7. עליכם לחזור על השאלות הבאות ולממש אותן ע״י שימוש בלולאות. השמות החדשים של הפונקציות מצוינים כאן:

- שאלה 1 - void isPalindromeIter(char str[],int len)
- שאלה 4- void IsDividedBy3Iter(long long num); הפעם אין הגבלה על המספרים שיוכנסו והמספר המתקבל יוכל להיות מורכב מספרות {1,2,3,4,5,6,7,8,9,0} -כלומר גם המספר 3459 תקין. בנוסף מותר להשתמש בפעולת החיבור, כל שאר ההגבלות מתרגיל 4 קיימות.

בהצלחה!