בס"ד

הנחיות כלליות

- יש לשלוח את הקבצים באמצעות מערכת ההגשה עד למועד ההגשה.
- ניתן להגיש את התרגיל באיחור עם קנס אוטומטי
  - 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 יהיה ניתן להגיש את התרגיל (ציון 0).

- שאלות בנוגע לתרגיל יש לפרסם באופן ציבורי בפורום הקורס.
- בקשות להארכה (מסיבות מוצדקות) יש לשלוח מייל פרטי למייל שפורסם לכם.
  כדי שאוכל לטפל בהארכה במידה והיא מאושרת, אנא ציינו:
  1. שם מלא.
  2. שם משתמש במערכת ההגשה.
  3. תעודת זהות.

- יש להקפיד מאוד על הוראות עיצוב הקלט והפלט, בדיוק על פי הדוגמאות המצורפות. הבדיקה האוטומטית בודקת שהפלט זהה לפלט הצפוי ולפיכך על הפלט להיות בדיוק באותו מבנה של הדוגמאות.
  בנוסף שימו לב להנחיות במסמך ה-Coding Style המפורסם באתר הקורס.

עליכם לכתוב קוד על פי ההנחיות ולוודא שקיבלתם 100 בבדיקה האוטומטית הראשונית, וכן שהתרגיל מתקמפל ורץ על השרת המחלקתי (planet) ללא שגיאות או אזהרות.

תרגיל שלא עומד בסטנדרטים הללו יגרור ירידה משמעותית בציון התרגיל, בשל הטרחה שהוא מייצר בתהליך הבדיקה שלו, עד כדי ציון 0.

להזכירכם העבודה היא אישית. "עבודה משותפת" דינה כהעתקה. התרגיל נבדק על ידי מערכת ההגשה האוטומטית גם מהבחינה הזו, ותרגיל שהועתק יגרור ציון 0 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.

שימו לב שאתם מגישים אך ורק את הקבצים המכילים את הקוד שלכם, ולא קבצים מיותרים שנוצרו על ידי סביבת העבודה. כמו כן הימנעו מהגשת קבצים/תיקיות עם שמות המכילים תווים בעברית.
שימו לב שאי אפשר להעתיק קובץ עם נתיב המכיל תווים בעברית לשרתי האוניברסיטה.
אנחנו משתמשים בc99 בתרגילים.

בס"ד

תרגיל 2

בתרגיל זה עליכם ליצור ולהגיש תוכנית בקובץ יחיד בשם ex_2.c
בתרגיל זה מותר להשתמש בספריות stdio.h math.h ובנוסף בכל החומר שנלמד בתרגולים.

פקודת הקימפול בתרגיל זה היא:

```
gcc ex_2.c -lm -std=c99 -DNDEBUG
```

בתרגיל זה נכתוב תכנית המאפשרת למשתמש לבחור ולבצע מספר פעולות.

בתחילת התוכנית יודפס למסך התפריט הבא:

```
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
```

לאחר מכן, על פי בחירת המשתמש, תתבצע המשימה המבוקשת.

הבחירה מתבצעת על ידי הקשה של המספר המתאים.

בסיום ביצוע משימה, יש להדפיס מחדש את התפריט ולפעול על פי בחירת המשתמש.

במידה והמשתמש מקיש 7, יש להדפיס Bye! ולסיים את התוכנית.

במידה והמשתמש מקיש אופציה שאינה מופיעה בתפריט (כלומר מספר שלא מ1 עד 7), יש להדפיס:

Invalid option!

ולהדפיס מחדש את התפריט.

שימו לב: לאחר הדפסת התפריט יש לרדת שורה.

ניתן להניח שלא יבדקו הכנסת קלט לתפריט שהוא לא מספר (כלומר לא נכניס אות למשל במקום מספר מ1 עד 7).

משימה 1: המרה מבסיס octal לבסיס hex

במשימה זו עליכם לקלוט מהמשתמש מספר בבסיס octal ולהמירו לבסיס hex.

לאחר בחירה במספר 1, תכתבו למסך:

Please enter number in octal base:

עם רווח לאחר הנקודתיים.

המשתמש מקליד מספר בבסיס octal **בסדר כתיבה הפוך**, כלומר – הספרה הראשונה שהוא מקליד תהיה ספרת "האחדות". המטרה כאן היא להקל עליכם בהמרת המספר, נסו להבין למה זה עוזר.

במידה והקלט תקין, עליכם להדפיס את ערכו של המספר בבסיס hex.

קלט חוקי הוא קלט המכיל מספרים אי-שליליים בבסיס octal בלבד.

שימו לב שבפלט שלכם שבבסיס hex , הספרות שערכן גדול מ-9 צריכות להיכתב באות גדולה ולא באות קטנה.

במידה והקלט לא חוקי יש להדפיס הודעת שגיאה של

Invalid input!

לדוגמה:

```
Please enter number in octal base: 275
17A
```

המספר האוקטלי הוא 572, מכניסים בסדר הפוך כאמור ולכן הוכנס 275, וייצוגו ב hex זה 17A.

שימו לב: ניתן להניח שהקלט יורכב ממספרים שלמים בלבד.

<u>משימה 2 :חיבור בינארי בין שני מספרים</u>

במשימה זו עליכם לקלוט שני מספרים מטיפוס unsigned long long אשר יכילו את הספרות 1/0 בלבד.

לאחר בחירה במספר 2, יש להדפיס:

Please enter two binary numbers:

גם כאן רווח לאחר הנקודתיים.

אם הקלט תקין, יש להדפיס את שני המספרים עם '+' בניהם, לאחר מכן את התו '=' ולבסוף את תוצאת החיבור.

שימו לב שישנו רווח לפני הפלוס ולאחריו. כנל עם ה=.

אחרת, יש להדפיס הודעת שגיאה של

Invalid input!

אם המספר הקטן לא תואם באורכו את המספר הגדול יש לרפד באפסים כך שיהיו באורך זהה.

יש להדפיס את המספר הקטן שרופד באפסים ראשון ולאחריו את המספר השני שהוזן. אם המספרים שהוזנו באותו אורך, אזי יש להדפיסם לפי סדר ההכנסה.

לדוגמה עבור הקלטים 1,101 יש להדפיס:

```
Please enter two binary numbers: 1
101
001 + 101 = 110
```

בנוסף, עבור קלט של אפסים פותחים אין צורך להתייחס שזה חלק מהאורך של המספר. למשל עבור:

00111 , 00011

אזי 00111 זה בעצם 111 ונחשיב את האורך של הקלט הראשון כ 3 ולא כ 5 ולכן יודפס:

011 + 111 = 1010

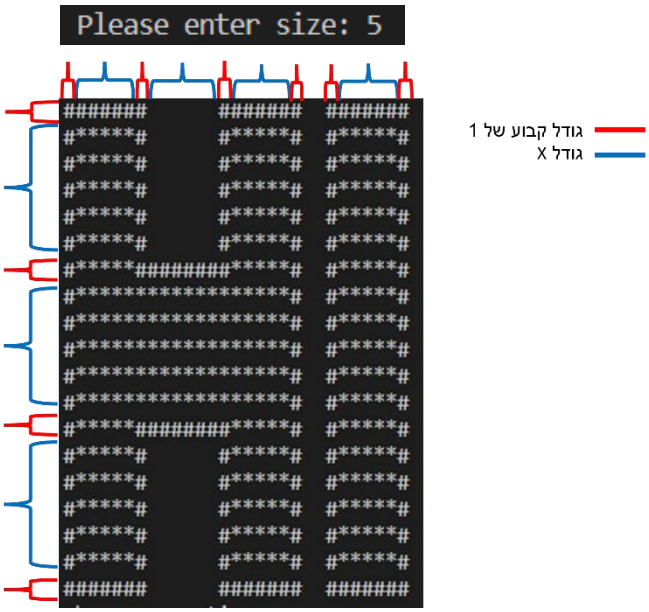
לכן, עבור קלט של אפסים בלבד, משמע שני המספרים 0, אין צורך לרפד עבור האפס "הארוך יותר" אלא לכתוב רק

0 + 0 = 0

<u>משימה 3: הדפסת HI</u>

במשימה זו עליכם לקבל מהמשתמש מספר שלם מטיפוס unsigned int (נסמן אותו בתור x) ולהדפיס את הצורה של המילה HI כך שהיא מורכבת מכוכביות (כאלו *) והיא עטופה במסגרת של סולמיות (כאלו #). x הוא פרמטר המגדיר את כמות הכוכביות שבתוך HI.

למשל עבור קלט באורך x=5 :



אם הקלט לא תקין, יש להדפיס הודעת שגיאה של

Invalid input!

לאחר הבחירה ב3, יש להדפיס:

Please enter size:

גם כאן רווח לאחר הנקודתיים.

<u>משימה 4 :ספירת הביטים שערכם 1 במספר</u>

במשימה זו עליכם לקלוט מספר בינארי ולהדפיס את מספר הפעמים שמופיע הביט 1.

לאחר הבחירה ב4 יש להדפיס:

Please enter a binary number:

גם כאן רווח לאחר הנקודתיים.

לדוגמה אם הקלט היה 111 יודפס למסך 3.

```
Please enter a binary number: 111
3
```

אם הקלט לא תקין, יש להדפיס הודעת שגיאה של

Invalid input!

<u>משימה 5 : המרה מבסיס דצימלי לבינארי</u>

במשימה זו עליכם לקלוט מספר דצימלי אי שלילי ולהפוך אותו לייצוג הבינארי שלו.

עליכם להדפיס את המספר הדצימלי שקלטתם עם חץ ואז את ייצוגו הבינארי. כלומר כך:

למשל אם קלטתם את המספר 7 יודפס למסך:

```
Enter a non negative decimal number: 7
7 => 111
```

שימו לב שיש רווח לפני תחילת החץ ואחריו.

(בשביל להקל עליכם בטסטים לא תקבלו קלט דצימלי שגדול מ 1023).

קלט חוקי הוא קלט המכיל מספרים אי-שליליים. אם הקלט לא תקין, יש להדפיס הודעת שגיאה של

Invalid input!

שימו לב: ניתן להניח שהקלט יורכב ממספרים שלמים בלבד.

לאחר הבחירה ב5 יש להדפיס:

Enter a non negative decimal number:

גם כאן רווח לאחר הנקודתיים.

בס"ד

משימה 6 : זיגזג

במשימה זו עליכם לקלוט מספר דצימלי אי שלילי ולקבוע האם הייצוג הבינארי שלו מייצג "זיגזג" כלומר האם הביטים הפוכים לאורך כל הייצוג. דוגמאות: 010101 מייצג זיגזג, לעומת 1110100 לא מייצג זיגזג.

תדפיסו למסך את הכרעתכם, כביטוי בוליאני, כלומר:

true אם זיגזג, false אחרת. שימו לב יש לכתוב זאת באותיות קטנות בלבד.

מספר המיוצג על ידי ביט אחד נחשב כזיג זג.

לאחר הבחירה ב6 יש להדפיס:

Enter a non negative decimal number:

גם כאן רווח לאחר הנקודתיים.

דוגמה:

```
Enter a non negative decimal number: 7
false
```

משימה 7 : exit

במשימה זו עליכם להדפיס Bye! למסך ולצאת מהתכנית.

```
Bye!
```

הערה כללית:
בכל מקום שאתם צריכים להדפיס
Invalid input!
וישנם מספר קלטים, תכתבו את הודעת השגיאה אחרי שקלטתם את כל הקלט, ולא ברגע שזיהיתים משהו לא תקין.

מלא בהצלחה!
☺ רונלי

מצורפת דוגמת הרצה:

```
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
```

```
5. decimal to binary
6. Zig-Zag bits
7. exit
1
Please enter number in octal base: 275
17A
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
1
Please enter number in octal base: 91
Invalid input!
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
2
Please enter two binary numbers: 1001
11
0011 + 1001 = 1100
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
2
Please enter two binary numbers: 25
100
Invalid input!
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
```

```
2
Please enter two binary numbers: 001
0011
01 + 11 = 100
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
3
Please enter size: 0
## ##  ##
#####  ##
#####  ##
## ##  ##
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
3
Please enter size: 3
#####    #####  #####
#***#    #***#  #***#
#***#    #***#  #***#
#***#    #***#  #***#
#***********#  #***#
#************#  #***#
#************#  #***#
#************#  #***#
#***********#  #***#
#***#    #***#  #***#
#***#    #***#  #***#
#***#    #***#  #***#
#####    #####  #####
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
```

```
4
Please enter a binary number: 11100101011
7
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
5
Enter a non negative decimal number: 7
7 => 111
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
5
Enter a non negative decimal number: 16
16 => 10000
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
6
Enter a non negative decimal number: 7
false
Choose an option:
1. octal to hex
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
6
Enter a non negative decimal number: 5
true
Choose an option:
1. octal to hex
```

```
2. binary addition
3. print HI
4. count bits
5. decimal to binary
6. Zig-Zag bits
7. exit
7
Bye!
```