תרגיל 5, מבוא לתכנות מערכות, אביב 2022

הגשה **בזוגות או ביחידים** דרך המודל
התרגיל הוא להגשה עד ליום רביעי, 31/5/2022 בשעה 23:59

חלק א: הדפסת hex dump של קובץ בינארי

כתבו תוכנית הדומה לפקודת המערכת hexdump בלינוקס. על התוכנית לקרוא קובץ בינארי ולהציגו בפורמט הקסאדצימלי נוח לקריאה, לדוגמא:

```
0000000 2041 6568 2078 7564 706d 6920 2073 2061
0000010 6568 6178 6564 6963 616d 206c 6976 7765
0000020 6f20 2066 6f63 706d 7475 7265 6420 7461
0000030 2c61 6620 6f72 206d 656d 6f6d 7972 6f20
0000040 2072 7266 6d6f 6120 6320 6d6f 7570 6574
0000050 2072 6966 656c 202e 000a
0000059
```

הפלט הנ"ל הינו עבור הקובץ הדוגמא הבא (הדוגמא הינה קובץ טקסט על מנת לאפשר הצגתה כאן):

```
A hex dump is a hexadecimal view of computer
data, from memory or from a computer file.
```

על התוכנית לקרוא את המידע הבינארי מקובץ הקלט, אשר שמו ניתן בשורת הפקודה, ולכתוב את הפלט ל-stdout. פורמט הפלט הינו:

- offset התחלת השורה בקובץ כמספר הקסאדצימלי בן 7 ספרות (עם תוספת leading zeros אם צריך).
- תו אחד של רווח
- 8 מספרים הקסדצימאליים בני 4 ספרות כל אחד, כל אחד מייצג את 2 הבתים הבאים של תוכן הקובץ. בכל זוג בתים, ה-byte שמופיע ראשון בקובץ מודפס מימין, והשני מודפס משמאל (בהתאם לייצוג little endian).
- בסוף הקובץ, השורה האחרונה יכולה לייצג פחות מ-8 בתים, אם אורך הקובץ לא מתחלק ב-8. במקרה זה, יוצגו כמה מספרים שיש (מספר עבור כל זוג בתים). אם מספר הבתים אי זוגי, אז המספר האחרון מחושב כאילו יש בייט אפס נוסף בקובץ.
- לאחר השורה האחרונה שכוללת data, יש שורה עם offset בלבד של התו אחרי האחרון בקובץ, כלומר ערך ה-offset כאורכו של הקובץ בבתים.

שם התוכנית צריך להיות hex_dump.c, ושורת ההפעלה שלה:
hex_dump <binary_input_file>

הערה:

- למי שעובדת עם cygwin, הפקודה hexdump הינה חלק מהחבילה xdd ב-setup של cygwin ואפשר להשתמש בה על מנת לוודא שהתוכנית עובדת כנדרש. ב-Linux הפקודה סטנדרטית ובדרך כלל קיימת כחלק מההתקנה.

חלק ב: דחיסת קובץ טקסט – compress8to7

בקובץ טקטס ASCII מספיקים למעשה 7 ביטים על מנת לייצג כל תו, היות וביט ה-MSB הינו אפס בכל הבתים, כלומר אין תווים מעבר ל 127. ניתן לנצל עובדה זו על מנת לדחוס קובץ טקסט ביחס של 7/8, כלומר חיסכון של 12.5% בגודל הקובץ.

בהנחה שקובץ הקלט מכיל את הבתים a, b, c, d, … לפי הסדר, כאשר כל בייט מורכב מ-7 ביטים מעניינים a6,a6,a5,a4,a3,a2,a1,a0 עבור a, כאשר a0 הינו ה-Least Significant Bit (LSB), וכו.

```
0 a6 a5 a4 a3 a2 a1 a0
0 b6 b5 b4 b3 b2 b1 b0
0 c6 c5 c4 c3 c2 c1 c0
0 d6 d5 d4 d3 d2 d1 d0
…
```

אז הדחיסה תיתן את הבתים הבאים (כל שורה זה 8 ביטים המרכיבים בייט אחד, כאשר ה-LSB מימין):

```
b0 a6 a5 a4 a3 a2 a1 a0
c1 c0 b6 b5 b4 b3 b2 b1
d2 d1 d0 c6 c5 c4 c3 c2
e3 e2 e1 e0 d6 d5 d4 d3
…
```

כיתבו תוכנית ש:

1. מקבלת שני שמות קבצים משורת הפקודה, ומופעלת על ידי:
compress8to7 <input_file.txt> <output_file.8to7>
2. פותחת את שני הקבצים (את הקלט ל-"r" ואת הפלט ל-"wb")
3. כותבת header בקובץ הפלט המזהה את פורמט הקובץ, ואת אורך הקובץ. ה-header כולל שני מספרים 64 ביט:
- הראשון הינו MAGIC וערכו הקבוע 0x0087008700870087ll
- השני הינו מספר הבתים בקובץ הפלט
4. כותבים את תוכן הקובץ הדחוס לפי הפורמט 8to7 שתואר למעלה

הערות:

- מומלץ לכתוב את ה-header תוך שימוש בטיפוס uint64_t אשר הינו סטנדרטי ב-C99 ומוגדר ב-stdint.h.
- הקבוע של MAGIC מסתיים בפעמיים האות L קטנה. זה סימון לכך שמדובר בקבוע של 64 ביט.
- הפעולה של כתיבת מספר הבתים בקובץ הקלט ל-header מונעת ממימוש כל הקוד ב- one pass. שתי אסטרטגיות מומלצות:
- כתיבה של אפס בגודל בתחילת התוכנית, וביצוע פעולת fseek ועדכון של הגודל בסוף.
- ביצוע fseek לסוף ו-ftell לקובץ הקלט על מנת לברר את גודלו בהתחלת הריצה לפני כתיבת ה-header.
- ממולץ לקרוא את הקלט בייט בייט על ידי fgetc, ולכתוב בייט בייט על ידי fputc.
- יש להוציא הודעת שגיאה מתאימה אם יש בייט בקובץ הקלט מעבר לערך של 127, כלומר עם MSB דלוק.

חלק ג: פתיחת דחיסת קובץ טקסט – uncompress8to7

תוכנית זו מבצעת את הפעולה ההפוכה ל-compress8to7.
היא מופעלת על ידי:

uncompress8to7 <input_file.8to7 > <output_file.txt>

הערות:

- יש לוודא שערך ה-MAGIC ב-header נכון (שקובץ הקלט בפורמט הנכון)
- יש להוציא הודעת שגיאה אם יש אי התאמה של גודל קובץ הטקסט כפי שמדווח ב-header לבין אורך הקלט שנקרא מהקובץ 8to7. כלומר אם הקלט נגמר לפני הזמן, או אם נשאר קלט אחרי שסיימנו.

- מומלץ להפעיל את שתי התוכניות בשרשרת ולוודא שאכן קובץ טקסט חוזר למצבו המקורי.

הערות כלליות

- הקפידו על הנדסת תוכנה טובה. העדיפו הנדסת תוכנה טובה על פני יעילות.
- הקפידו להשתמש בטיפוס המתאים בכל מצב.
- הקפידו על שמות טובים למשתנים, על const correctness, ושכל משתנה יהיה מוגדר ב-scope הכי קטן בו הוא נדרש.
- העדיפו תמיד לעשות שימוש בפונקציות קיימות לצורך ביצוע משימות והמנעו ככל הניתן משכפול קוד.
- בעקרון, יש לתת תיעוד קצר לפני כל פונצייה, מלבד main. מלבד זאת, יש לתת תעוד רק בנקודות בהן יש משהו שדורש הסבר ולא מובן מייד מהקוד.
  דוגמא לתיעוד מיותר (שיכול להוריד נקודות) הינה: // define an integer variable
  דוגמא לתיעוד נחוץ הינה: // reset write offset and write the correct size to header
- הקפידו לבדוק שפתיחת הקבצים הצליחה, וכן לסגור את כל הקבצים בסוף.
- הקפידו לבדוק שהקצאות זכרון הצליחו, ולשחרר את הזיכרון בסוף, אם נדרשת בתוכנית הקצאה דינמית של זיכרון.
- ככלל, במקרה של שגיאה, יש להדפיס את הודעה ל-stderr ולסיים את התוכנית מיד.

הגשה

הגישו את הקבצים 3 הקבצים hex_dump.c, compress8to7.c, ו-uncompress8to7.c כארכיב zip.
אין להגיש קבצים אחרים. כתבו את שמות המגישים ומספרי תעודות הזהות בראש הקבצים.

בהצלחה!