כתיבה בשפת Assembly בסביבת Linux

מבוא

במדריך זה נראה כיצד לכתוב קוד באסמבלי, להפעיל אסמבלר, להריץ את התוצאה וכיצד להגיש את הקוד עבור תרגילי הבית השונים.
רוב הדברים במדריך זה הינם המלצה לעבודה (למעט צורת ההגשה). ניתן לעבוד בכל צורה שנוחה לכם.

ניתן לכתוב קוד אסמבלי בעורך טקסט פשוט. יש לכם אחד על המכונה, בשם Mousepad. ניתן להוריד אחרים בהתאם לנוחיות שלכם, כגון notepad++ או sublime.
<u>הערה:</u> סגל הקורס אינו תומך ב-SASM וכל שימוש בתוכנה הוא על אחריותכם.

איך בונים קובץ ריצה?

GNU Assembler

ראשית, מפעילים את האסמבלר (GNU Assembler), עם הפקודה **as**, כדי שיתרגם את שפת האסמבלי (קובץ טקסט) לפקודות מכונה - קבצי **o**bject (קבצים עם סיומת o.):

```
as helloworld.asm -o helloworld.o
as start.asm -o start.o
```

קבצי o מכילים גם עוד דברים מלבד פקודות המכונה, עליהם נלמד בחצי השני של הקורס (כשנלמד על קבצי ELF, קישור סטטי ועוד). הפעלה של האסמבלר נקראת assemble ובעברית לפעמים נכנה אותה "בנייה". שימו לב שלא מדובר בקומפילציה המוכרת לכם משפת C, שם הקומפיילר מתרגם את השפה העילית לפקודות בשפת מכונה. (עוד על כך בחלק "איך עובד gcc" שמופיע במסמך).

GNU Linker

לאחר מכן, מפעילים את הקשר הסטטי (static linker) כדי שימזג את שני הקבצים לקובץ ריצה (גם הוא ELF) (בשם helloworld):

```
ld helloworld.o start.o -o helloworld
```

על תפקידו של ה-static linker ועל קיומו של ה-dynamic linker נלמד בהמשך הקורס.

Loader

לבסוף אפשר להפעיל את הטען (loader), שטוען את קובץ הריצה אל הזיכרון שמערכת ההפעלה מקצה עבור התוכנית שאנו מריצים:

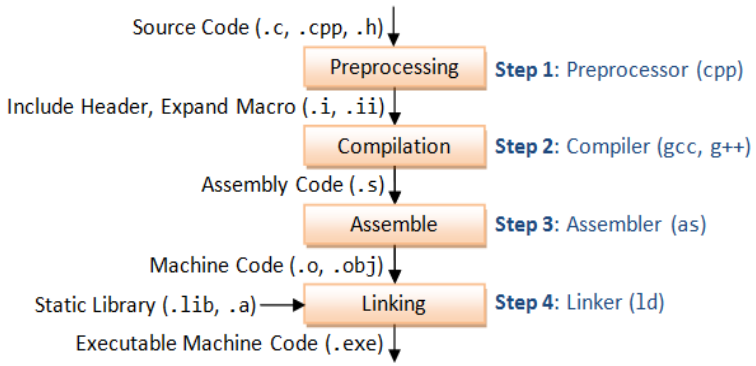
```
./helloworld
```

תהליך יצירת קובץ ריצה של GCC

קראו את הקטע הבא, המסביר כיצד ה-GCC הופך תוכנית (או תוכניות) בשפת c לקובץ ריצה:

GCC compiles a C/C++ program into executable in 4 steps as shown in the below diagram. For example, a "gcc -o hello.exe hello.c" is carried out as follows:

1. Pre-processing: via the GNU C Preprocessor (cpp.exe), which includes the headers (#include) and expands the macros (#define).
   > cpp hello.c > hello.i
   The resultant intermediate file "hello.i" contains the expanded source code.
2. Compilation: The compiler compiles the pre-processed source code into assembly code for a specific processor.
   > gcc -S hello.i
   The -S option specifies to produce assembly code, instead of object code. The resultant assembly file is "hello.s".
3. Assembly: The assembler (as.exe) converts the assembly code into machine code in the object file "hello.o".
   > as -o hello.o hello.s
4. Linker: Finally, the linker (ld.exe) links the object code with the library code to produce an executable file "hello.exe".
   > ld -o hello.exe hello.o ...libraries...



כאמור, בעזרת האפשרות:

```
gcc -S myfile.c
```

ניתן לחשוף את קוד האסמבלי שה-GCC מייצר כשלב ביניים.

תרגילי רשות (לא להגשה!)

קחו את הקבצים `helloworld.asm` ו-`start.asm` ועברו על הסעיפים הבאים:

1. מצאו את האזורים (sections) השונים בקוד.
2. מצאו מהי נקודת ההתחלה של התוכנית.
3. נסו להבין מה התוכנית עושה ואיך היא עושה את זה.
   (זה בסדר אם אתם לא מבינים הכל כרגע, בהמשך הקורס הכל יתבהר).
4. בנו קובץ ריצה בהתאם להוראות בתחילת הקובץ.
5. נסו לגרום לתוכנית להדפיס "ATAM Good" מבלי לשנות את ה-data
   (ולאחר מכן צרו מחדש את קובץ הריצה ובדקו אם זה עבד!)

וסעיף שאינו קשור לקבצים הנתונים:

6. כתבו קוד בשפת c, תרגמו אותו לאסמבלי ונתחו את הקוד.
   בפרט, נסו להבין כיצד פקודות המכונה מבצעות את מה שכתבתם ב-c.

בהצלחה!

צוות הקורס