פרויקט – חלק 1

כפי שבוודאי כבר סיפרו לכם, עד סוף הסמסטר תממשו קומפיילר בסיסי. בתרגיל זה נתחיל בבניית היסודות עליהם יעמוד הקומפיילר.

שפת הקלט לקומפיילר שלכם היא שפת --C הידועה ☺, ובתרגיל זה עליכם לבנות מנתח לקסיקלי אשר מקבל קלט בסיסי בשפה, מחלק אותו לאסימונים, ומדפיס כל אסימון שזיהה.

אסימוני השפה --C:

מילים שמורות :

```
real integer write read while do if then else
main var call return defstruct extern
```

סימנים :

```
( ) [ ] { }
, : ;
```

אסימונים מורכבים:

`id`: מתחיל באות (לטינית גדולה או קטנה) וממשיך באותיות או ספרות

`num`: מספר שלם או בעל נקודה עשרונית

* ניתן להניח שאין תמיכה במספרים שליליים.

`str`: מחרוזת מתחילה בגרשיים ומסתיימת בגרשיים ומכילה את כל התווים שבינהם.

* מחרוזת לא יכולה להתפצל בין מספר שורות. אם מסתיימת השורה והמחרוזת לא הסתיימה (חסר גרשיים סוגרים) יש לטפל בשגיאה כמפורט בעמוד הבא.

פעולות :

(הסימן "|" מסמן "או", למעט עבור האסימון "or")

```
relop: == | <> |< | <= | > | >=
addop: +|-
mulop: * | /
assign: =
and: &&
or: ||
not: !
```

הערות:

הערות בשפה הן בסגנון הערות שורה ב-++C, כלומר מתחילות ב-// (בכל מקום בשורה) ומסתיימות בסוף השורה. ההערה **אינה** כוללת את סימן מעבר שורה החדשה.

רווחים:

בין אסימונים <u>יכולים</u> להופיע תווי רווח - whitespace (רווח רגיל, טאב, שורה חדשה) – אבל לא חייבים להיות.

שימו-לב, שורה חדשה יכולה להיות בסגנון UNIX (LF) או בסגנון DOS (CR+LF).

המשימה:

- כתבו מנתח לקסיקוגרפי (קובץ Lex עבור הכלי Flex) אשר מקבל תוכנית בשפת --C, מזהה את האסימונים שבה ומדפיס אותם בצורה הבאה (בהתאם לחלוקה לעיל של מבנה השפה לקבוצות):
  1. עבור מילה שמורה יודפס <*reserved_word*> (כלומר, המילה השמורה בסוגרים)
  2. עבור **num** יודפס <num,*num*_lexeme>
  3. עבור **id** יודפס <id,*id_lexeme*>
  4. עבור **str** יודפס <str,*string lexeme*> כאשר תוכן המחרוזת ללא הגרשיים.
  5. עבור פעולות יודפס שם הקבוצה של הפעולה ושם הפעולה: <op type,*op*_lexeme>, למשל: <relop,==>

כלומר: <token_type[,*value*]> כאשר token_type הוא סוג האסימון הנגזר ו-*value* הוא הלקסמה של האסימון הנגזר, בסוגי האסימון הרלוונטיים. **שימו לב:** אין להוסיף בפלט רווחים בין סימני '<', '**,**', '>' ובין הערכים המודפסים.

- הוראות נוספות:
  1. יש להגדיר בקובץ ה-Lex מקרואים עבור הביטויים הרגולריים של id, str ו- num בחלק ה-Definitions ולהשתמש בהם בחלק ה-Rules כנ"ל. שמות המקרואים יהיו כשמות האסימונים (id, num, str). **הגדרות אלו ייבדקו בנפרד בנוסף לבדיקת תפקוד המנתח**.
  2. יש להתעלם מהערות (כלומר, הן לא תופענה בפלט).
  3. כל אסימון מקבוצת ה"*סימנים*" יועבר לפלט ללא שינוי.
  4. כל תו מסוג רווח (רווח רגיל, טאב, שורה חדשה) יועבר לפלט ללא שינוי.

- מצורפת דוגמה לפלט מצופה עבור תכנית דוגמה: example.cmm → example.tokens
- הקלט של המנתח מגיע מהקלט הסטנדרטי (stdin) והפלט צריך להיכתב לפלט הסטנדרטי. אין צורך לפתוח קבצים לקלט או לפלט. ניתן להעביר למנתח קובץ באמצעות redirection, לדוגמה:

```
$ ./part1 < example.cmm
```

טיפול בשגיאות:

אם זוהה סימן שאיננו אסימון חוקי בשפה, יש להדפיס <u>בשורה חדשה</u> הודעת שגיאה בתבנית הבאה ולצאת מיידית מהתוכנית עם קוד שגיאה 1:

```
Lexical error: '<lexeme>' in line number <line_number>
```

כאשר <lexeme> הינה הלקסמה (הסימן) הלא חוקית שזוהתה ו-<line_number> הוא מספר השורה של אותה לקסמה. לדוגמא:

```
Lexical error: '@' in line number 2
```

כלומר, בפלט יופיע הניתוח של כל מה שהופיע לפני השגיאה ואז תופיע הודעת השגיאה בשורה נפרדת, והניתוח ייפסק.

* מצורפת דוגמה לפלט מצופה עבור תכנית עם שגיאה:

example2-err.cmm → example2-err.tokes

הוראות הגשה:

- מועד אחרון להגשה: יום ה' 10/11/2016 בשעה 23:55 .
- ההגשה בזוגות. הגשה בבודדים תתקבל רק באישור מראש מצוות הקורס.
- יש להגיש בצורה מקוונת באמצעות ה-Moodle מחשבונו של אחד הסטודנטים.
- יש להגיש קובץ ארכיב מסוג Bzipped2-TAR בשם מהצורה (שרשור מספרי ת.ז – 9 ספרות):
  `proj-part1-<student1_id>_<student2_id>.tar.bz2`
- בארכיב יש לכלול את הקבצים הבאים:
  - את קובץ ה LEX בשם `part1.lex`
  - קובץ `makefile` היוצר את המנתח הלקסיקלי - שם קובץ ההרצה של המנתח הנוצר צריך להיות `part1`
- קובץ הארכיב צריך להיות "שטוח" (כלומר, שלא ייצור ספריות משנה בעת הפתיחה אלא הקבצים ייווצרו בספריה הנוכחית).
- דוגמה לפקודה ליצירת הארכיב (בלינוקס) – נקראת מהספריה בה נמצאים קבצי ההגשה:

```
$ tar cjf proj-part1-123456789_232323239.tar.bz2 part1.lex makefile
```

- על התרגיל להתקמפל ולרוץ בהצלחה במכונה הוירטואלית של לינוקס המסופקת לכם.
- **<u>תרגיל שלא יצליח להתקמפל יקבל 0.</u>**

כללי:

- התרגיל נבדק באופן חצי אוטומטי. יש להקפיד על מבנה הפלט כמפורט ולא להוסיף או לגרוע רווחים כדי שיתאים לפלט המצופה במדויק.
- סביבת הבדיקה הרישמית הינה המכונה הוירטואלית של לינוקס המסופקת לכם. ניתן להוריד את קובץ המכונה ליבוא לסביבת **VirtualBox** באמצעות הקישור המסופק באתר הקורס. באותו מקום ניתנים גם מספר קישורים המתעדים התקנת **VirtualBox** במחשבכם האישי. יש להשתמש בגירסה עדכנית של **VirtualBox** לתפקוד מיטבי של המכונה (רצוי גירסה 5.1.x ומעלה).
- ניתן לפתח במחשב אחר מהמכונה הוירטואלית, אולם חובה לוודא שהתרגיל המוגש נבנה ורץ היטב במכונה הוירטואלית, לפני ההגשה.
- לא מומלץ לפתח בסביבות מבוססות ווינדוס. כלי ה**Flex** עלול להתנהג באופן שונה בסביבת ווינדוס לעומת סביבת המטרה (לינוקס). מעבר בין הסביבות עלול לחשוף שגיאות שקשה לאתר בגלל מבנה קבצים שונה וחוסר תאימות מלא לגירסאות התוכנה בשרתים.
- משאבים נוספים לגבי השימוש בכלי ה-Flex (תיעוד ודוגמאות) תוכלו למצוא במצגת התרגול הראשון ובאתר הקורס (תחת "חומר עזר לקורס ולפרויקט").
- שימו-לב למדיניות בנוגע לאיחורים בהגשה המפורסמת באתר הקורס. במקרה של נסיבות המצדיקות איחור, יש לפנות **מראש** לצוות הקורס לתיאום דחיית מועד ההגשה.
- הקפידו לוודא כי העלתם את הגירסה של ההגשה אותה התכוונתם להגיש. לא יתקבלו טענות על אי התאמה בין הקובץ שנמצא ב-Moodle לבין הגירסה ש"התכוונתם" להגיש ולא יתקבלו הגשות מאוחרות במקרים כאלו.

בהצלחה!