הנחיות הגשה – עבור כל שלושת שלבי הפרויקט

את הפרויקט יש לפתח באחת מהסביבות הבאות **בלבד**:
- Windows Visual Studio גרסה 17 או 19
- סביבת Linux GNU compiler

פרויקטים אשר יפותחו בסביבות אחרות (כגון: Unix, Apple, Eclipse, Cloud9) **לא יתקבלו**, זאת עקב בעיות העולות במהלך הפיתוח בסביבות אחרות, והקושי בבדיקה.

בדקו את הנחיות ההגשה המתאימות לכם בעמודים הבאים

1. עליכם למלא ולהגיש, בצמוד לקבציי הפרויקט, גם קובץ טקסט (information.txt) המכיל את **המידע** הבא:

   א. שמות הסטודנטים ות"ז
   ב. מערכת ההפעלה בה פותח הפרויקט, כגון:

   Windows:
   7-32Bit, 7-64Bit, 8-64Bit, 10-64Bit
   Linux:
   Ubuntu, Debian, RedHat, etc… + version of the operating system.

   ג. סביבת הפיתוח בה השתמשתם:
   Visual Studio 17, 19 / GCC compiler version (for Linux)

   ד. הערות חשובת במידה ויש.

דוגמא לקובץ information.txt נמצאת באתר המודל של הקורס. יש להוריד את הקובץ מהאתר המודל ולמלא אותו בהתאם. (אין להמיר את הקובץ לפורמט word או pdf)

2. יש להעלות את הפרויקט לאתר המודל של הקורס כקובץ מסוג ZIP או RAR.
   שם הקובץ חייב להכיל את כל מספרי ת"ז של הסטודנטים המגישים. למשל:

   032648915_045198127.zip

3. הסביבה של Flex מייצאת קובץ הכתוב ב-C.
   סטודנטים הרוצים לעבוד ב-++C מוזמנים, קחו בחשבון שעלולות לצוץ סוגיות שונות בשלב הפיתוח של הפרויקט הקשורות לשילוב של שתי השפות – באחריותכם.

4. יש לדאוג לכתוב הערות, שמות משמעותיים, חלוקה נכונה לפונקציות, עם קבצי כותרת. ניהול זיכרון נכון וכו'.

5. העתקות יטופלו בחומרה! ראו הוזהרתם!

6. פורמט של הודעות פלט של מנתח לקסיקלי (שלב ראשון של הפרויקט)

קובץ הפלט של המנתח הלקסיקלי הוא למעשה הדפסה של כל ה – Tokens שזוהו ונשמרו במהלך המעבר על קובץ קלט.

עבור כל Token יש לייצא את האינפורמציה באופן הבא:

Token of type '{token.kind}', lexeme: '{token.lexeme}', found in line: {token.line}

במקרה של שגיאה לקסיקלית יש להוציא הודעת שגיאה זו:

Character 'character' in line: {line} does not begin any legal token in the language.

7. פורמט של הודעות פלט של מנתח תחבירי (שלב שני של הפרויקט)

קובץ הפלט של המנתח התחבירי מכיל בתוכו שני חלקים:

א. רצף של כללי הגזירה אשר היו בשימוש במהלך הניתוח התחבירי של הקלט.
כלומר, כל שימוש בכלל גזירה מלווה בהדפסה בפורמט הבא:

Rule {derivation rule}

לדוגמא:

Rule {BLOCK -> block DECLARATIONS begin STATEMENTS end}
Rule {SIMPLE_TYPE -> integer}

ב. הודעות על שגיאות תחביר.
בכל פעם שבו מנתח תחבירי נתקל בשגיאה, יש להוציא הודעה בפורמט הבא:

Expected token of type '{expectedToken.type}' at line: {token.line},
Actual token of type '{token.type}', lexeme: '{token.lexeme}'.

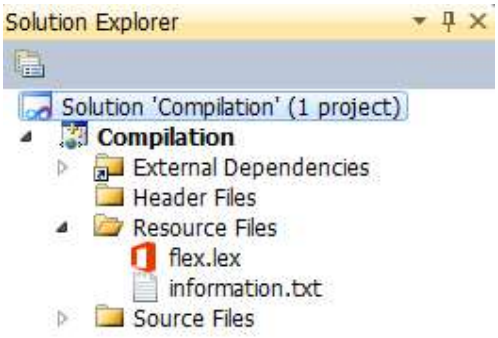
הגשה שלא לפי ההנחיות תגרום להורדת הציון!!!

הנחיות הגשה עבור Windows Visual Studio

1. הפרויקט ייבדק תחת Visual Studio 17 או Visual Studio 19. נא להתכוונן בהתאם.

2. עליכם לצרף את כל קבצי הפרויקט (קבצי c, h, flex file, lex.yy.c, project files) באופן בו ניתן יהיה להרים את כל הסביבה במחשב של הבודק (שזה כולל קובץ sln, vcxproj וכו'...). **יש גם לצרף קובץ exe מוכן להרצה בתיקייה נפרדת.**

3. קובץ ה-Flex (אין חשיבות לשם הקובץ) וקובץ ה-information.txt צריכים לשבת בתיקיית ה-Resource Files באופן הבא:



4. הקומפיילר שאתם מפתחים צריך לבדוק שני קבצי הרצה הממוקמים אצל הבודק בנתיב הבא:

C:\\temp\\test1.txt
C:\\temp\\test2.txt

אלו הם קבצי הטסטים שעליהם תיבדק תקינות הפרויקט שלכם.
התוכנית צריכה לפנות באופן ישיר (hard-coded) אל הנתיבים האלה.
ניתן לשלב עם תפריט קטן השואל את המשתמש איזה מהטסטים הוא רוצה להריץ, או לבצע איפוס בין שני הטסטים באמצעות פונקציית yyrestart()

5. בכל אחד מ-3 שלבי הפרויקט, יש לבצע בדיקות של הקומפיילר על שני קבצי הטסט. לכל סוג של ניתוח יש ליצור שני קבצי פלט. לדוגמא:

c:\\temp\\test1_032648915_984204912_lex.txt
c:\\temp\\test2_032648915_984204912_lex.txt
c:\\temp\\test1_032648915_984204912_syntactic.txt
c:\\temp\\test2_032648915_984204912_syntactic.txt
c:\\temp\\test1_032648915_984204912_semantic.txt
c:\\temp\\test2_032648915_984204912_semantic.txt

קבצי הפלט צריכים להופיע באותה תיקייה בסמוך לטסטים עצמם; השם של קובץ הפלט חייב לכלול את שם הטסט (test1 / test2) + מספרי ת"ז שלכם + סוג הניתוח שבוצע.

שימו לב:

- בשלב א' יש ליצור **2** קבצי output שונים (אחד ל- test2 ואחד ל-test1) עבור ניתוח לקסיקלי
- בשלב ב' יש ליצור **4** קבצי output שונים (2 ל-test2 ו-2 ל-test1) עבור ניתוח לקסיקלי ותחבירי
- בשלב ג' יש ליצור **6** קבצי output שונים (3 ל-test2 ו-3 ל-test1) עבור ניתוח לקסיקלי, תחבירי וסמנטי

הנחיות הגשה עבור סביבת Linux

1. פרויקט בסביבת Linux ייבדק תחת: Ubuntu 19.10 Eoan Ermine **64bit**
פרטים כיצד להתקין gcc ניתן למצוא בלינק הבא:
https://askubuntu.com/questions/987677/how-to-install-gcc-6-3-on-17-10-64bit-desktop

ניתן לעבוד בכל סביבת Linux המתאימה לכם, אך עליכם לוודא שהיא רצה ללא בעיות קומפילציה במערכת ה- Linux הרשומה למעלה.
ניתן להוריד את הסביבה עבור Virtual Machine בכתובת הבאה:
https://www.osboxes.org/ubuntu/#ubuntu-19-10-vbox

2. עליכם לצרף את כל קבצי הפרויקט (קבצי c, h, flex file, lex.yy.c, project files) באופן בו ניתן יהיה להרים את כל הסביבה במחשב של הבודק.
עליכם גם לצרף קובץ מוכן להרצה בתיקייה נפרדת.

3. עליכם ליצור קובץ make (אשר דואג להדר את הפרויקט על כל קבציו ולייצא קובץ הרצה)
פרויקט ללא קובץ make מתאים לא ייבדק!

ניתן להיעזר בלינק הבא המסביר כיצד לבנות קובץ make: (הסבר שלב א' אמור להספיק)
http://www.cs.colby.edu/maxwell/courses/tutorials/maketutor/
המטרה של קובץ ה-make היא אך ורק בשביל לקרוא ל-gcc, ולא מעבר לכך.
שימו לב: שם הקובץ חייב להיות **"Makefile"** או **"makefile"** ללא סיומת

4. קובץ ה-Flex והקובץ information.txt צריכים לשבת באותה התיקייה יחד עם קבצי ה-c ו-h.

5. הקומפיילר שאתם מפתחים צריך לבדוק שני קבצי הרצה הממוקמים אצל הבודק בנתיב הבא:

//home//osboxes//Documents//test1.txt	אלו הם קבצי הטסטים שעליהם תיבדק תקינות הפרויקט שלכם.
//home//osboxes//Documents//test2.txt	

התוכנית צריכה לפנות באופן ישיר (hard-coded) אל הנתיבים האלה.
ניתן לשלב עם תפריט קטן השואל את המשתמש איזה מהטסטים הוא רוצה להריץ, או לבצע איפוס בין שני הטסטים באמצעות פונקציית ()yyrestart

שימו לב: Linux היא מערכת case-sensitive, הנתיב חייב להיות מדויק כפי שהוא כתוב מעלה (כל האותיות הן קטנות, חוץ מ-'D' של Documents שחייב להיות באות גדולה)
שימו לב 2: אין אות כונן ב-Linux.
שימו לב 3: סימן ההפרדה בין תיקיות הוא הפוך ממה שנהוג ב-Windows (// במקום \\)

6. בכל אחד מ-3 שלבי הפרויקט, יש לבצע בדיקות של הקומפיילר על שני קבצי הטסט. לכל סוג של ניתוח יש ליצור שני קבצי פלט. לדוגמא:

//home//ubuntu//Documents//test1_032648915_984204912_lex.txt
//home//ubuntu//Documents//test2_032648915_984204912_lex.txt
//home//ubuntu//Documents//test1_032648915_984204912_syntactic.txt
//home//ubuntu//Documents//test2_032648915_984204912_syntactic.txt
//home//ubuntu//Documents//test1_032648915_984204912_semantic.txt
//home//ubuntu//Documents//test2_032648915_984204912_semantic.txt

קבצי הפלט צריכים להופיע באותה תיקייה בסמוך לטסטים עצמם; השם של קובץ הפלט חייב לכלול את שם הטסט (test1 / test2) + מספרי ת"ז שלכם + סוג הניתוח שבוצע.

שימו לב:
- בשלב א' יש ליצור **2** קבצי output שונים (אחד ל- test2 ואחד ל-test1) עבור ניתוח לקסיקלי
- בשלב ב' יש ליצור **4** קבצי output שונים (2 ל-test2 ו-2 ל-test1) עבור ניתוח לקסיקלי ותחבירי
- בשלב ג' יש ליצור **6** קבצי output שונים (3 ל-test2 ו-3 ל-test1) עבור ניתוח לקסיקלי, תחבירי וסמנטי