תרגיל בית flex & bison (וגם flex & recursive descent parser) סמסטר 2025 ב
10 אפריל 2025

הגשה בזוגות דרך moodle

מה צריך להגיש? קובץ zip הכולל את הקוד שאתם כתבתם ודוגמא לקלט עם הפלט המתאים. אין צורך להגיש את הקבצים שיצרו flex & bison.

יש להגיש 2 גרסאות של התכנית : גרסה אחת תשתמש ב- flex וב-parser שנוצר ע"י bison. גרסה שניה תשתמש ב- recursive descent parser (וב- flex) . את הגרסה הראשונה הגישו בתיקיה שתקרא bottom_up. את הגרסה השנייה הגישו בתיקיה שתקרא top_down.

יש לכתוב תכנית שהקלט והפלט שלה הם כמפורט :
קלט

שם קובץ הקלט יהיה ה- command line argument היחיד של התוכנית.
אם אין command line argument אז בתור ברירת מחדל התכנית תקרא את הקלט מה- standard input. הערה : yyin (המשתנה של flex) הוא בתור ברירת מחדל ה- standard input.

הקלט כולל מידע על טיסות יוצאות (כמו בתרגיל הבית הראשון). תיאור מפורט של הקלט הופיע בתרגיל הבית הראשון.

הפלט של התכנית
התכנית תכתוב את הפלט ל- standard output.
בפלט יופיעו המספרים של טיסות המטען. בנוסף יופיע בפלט פרוט של מספר הטיסות שיוצאות "לפני הצהריים" ומספר הטיסות שיוצאות "אחרי הצהריים". בספירה של מספר הטיסות אין לקחת בחשבון את טיסות המטען.

כפי שתואר בתרגיל הבית הראשון, שעות עשויות להופיע בקלט בשני פורמטים שונים :
בפורמט של am/pm או בפורמט של 24 שעות.
שעה בפורמט של am/pm תחשב לשעה שהיא "לפני הצהריים" אם רשום אחריה am והיא תחשב לשעה שהיא "אחרי הצהריים" אם רשום אחריה pm.

שעה בפורמט של 24 שעות תחשב לשעה שהיא "לפני הצהריים" אם היא שעה בטווח [00:00 – 11:59] והיא תחשב לשעה שהיא "אחרי הצהריים" אם היא בטווח [12:00-23:59]. (הסוגריים המרובעות כאן מציינות שהטווח כולל את נקודות הקצה).

הנה דוגמא לקלט :

```
<departures>

LY1 00:10a.m. [JFK]

BA289 10:15p.m. [Heathrow]  cargo

AF1234  15:20        [Charles de Gaulle]

OP78 05:37 [Athens]  freight
```

```
AA17 11:40 [LAX]
```

בדוגמא זו הפלט צריך להיות:

```
BA289
0P78
number of flights before noon: 2
number of flights after noon: 1
```

הפלט כולל את מספרי טיסות המטען ולאחר מכן את מספר הטיסות מכל סוג (לפני או אחרי הצהריים). טיסות המטען לא נכללות בספירה. בדוגמא זאת הטיסה הראשונה והטיסה האחרונה הן לפני הצהריים. הטיסה השלישית היא אחרי הצהריים.

<u>דקדוק לתאור הקלט (בפורמט של bison)</u>

בהתאם למוסכמה של bison -- אסימונים כתובים כאן באותיות גדולות, ומשתנים כתובים באותיות קטנות. בכל אחד מכללי הגזירה מופיע נקודותיים במקום חץ (זה הפורמט של bison). למשל הכלל foo: A B C הוא הכלל foo -> A B C
שימו לב שבסוף כל כלל גזירה (או מספר כללי גזירה המופרדים ע"י |) מופיע נקודה פסיק בהתאם לפורמט של bison.

כשרשום למשל flights_list: %empty;
הכוונה היא ש- flights_list גוזר את המילה הריקה.

```
input: DEPARTURES flights_list;

flights_list: flights_list flight;

flights_list: %empty;

flight: FLIGHT_NUMBER TIME AIRPORT optional_cargo

optional_cargo:  CARGO | FREIGHT | %empty;
```

<u>לגבי recursive descent parser</u>:
הדקדוק הנתון אינו LL(1) בגלל שיש בו רקורסיה שמאלית בכלל של flights_list.
אבל למרות זאת לא קשה לכתוב recursive descent parser:
את הכלל עם רקורסיה שמאלית ניתן להחליף ברקורסיה ימנית. או לחילופין ראו בדוגמאות ל- recursive descent parser שיש במודל (ומוזכרות בהמשך) כיצד ניתן לטפל בסדרות בעזרת לולאת while.

הערות

עליכם להחליט באיזה ערכים סמנטיים להשתמש. אין להשתמש במשתנים גלובליים (כדי לתרגל את השמוש בערכים סמנטיים).

<u>תזכורת: הכנת תוכנית בעזרת flex & bison.</u>
(ההערות מתיחסות ל- Windows ול- Linux)

נניח שברשותנו קובצי קלט ל- flex ול- bison שהכנו בעזרת text editor (למשל Notepad++). נקרא לקבצים airport.lex ו- airport.y.

נריץ את הפקודות הבאות ב- command line:

1. מריצים את flex

```
flex airport.lex
```

נוצר קובץ lex.yy.c (שבו הפונקציה yylex).

2. מריצים את bison עם האופציה d–

```
bison -d airport.y
```

bison יצור שני קבצים: airport.tab.c ו- airport.tab.h. (את השני הוא יצור בגלל האופציה d–).

הערה: בקובץ airport.tab.c תמצא הפונקציה yyparse.

בקובץ airport.tab.h ימצאו הגדרות של האסימונים, של ה- union והכרזה של yylval. אנו יוצרים את הקובץ הזה כדי שניתן יהיה לעשות לו include במקום המתאים ב- airport.lex כדי שהפונקציה yylex תכיר את ההגדרות של האסימונים ושל yylval שכן היא משתמשת בהם.

הערה נוספת: אין חשיבות לסדר שבו מבצעים את שני הצעדים הראשונים כלומר ניתן להריץ קודם את bison ולאחר מכן את flex.

3. יש לקמפל את קובצי ה- C ש- flex & bison יצרו עבורנו.
(כמובן שאם התוכנית שלנו כוללת קבצים נוספים יש לקמפל גם אותם).
לצורך כך ניתן להשתמש בכל קומפיילר לשפת C.

אם נשתמש בקומפיילר gcc (קומפיילר פופולרי של GNU) הפקודה היא:

```
gcc -o airport.exe  lex.yy.c airport.tab.c
```

כאן האופציה o- מציינת את שם הקובץ שהוא התוצר של הקומפילציה. במקרה זה שם הקובץ הוא airport.exe. (על Linux נוותר על הסיומת exe).

4. נכין קובץ טקסט שנקרא לו test_airport.txt ובו נכתוב קלט לדוגמא למשל

```
<departures>

LY1 00:10a.m. [JFK]

BA289 10:15p.m. [Heathrow]  cargo

          . . .
```

נריץ את הפקודה

```
airport  test_airport.txt
```

(על Linux: ./airport ...)
והפלט יהיה:

```
BA289
0P78
number of flights before noon: 2
number of flights after noon: 1
```

הכנת תוכנית עם recursive descent parser (ועם flex כדי לייצר את המנתח הלקסיקלי):

נניח שקובץ הקלט ל- flex נקרא airport.lex וששאר הקוד שלנו
(כולל הקוד של ה- recursive descent parser) נמצא בקבצים airport.c
ו- airport.h. בקובץ האחרון נשים הגדרות והכרזות משותפות
ל- lexer (yylex) ול- parser: הגדרות של סוגי האסימונים,
הכרזה של משתנה גלובלי שבו yylex יכתוב את הערך הסמנטי של האסימון שהוא מחזיר
(משתנה בעל תפקיד דומה ל- yylval של bison) ואולי דברים נוספים.

1. מריצים את flex

```
flex airport.lex
```

נוצר קובץ lex.yy.c (שבו הפונקציה yylex).

2. יש לקמפל את קובצי ה- C שלנו (כולל הקובץ ש- flex כתב עבורנו).
לצורך כך ניתן להשתמש בכל קומפיילר לשפת C.

אם נשתמש בקומפיילר gcc הפקודה היא:

```
gcc -o airport.exe  lex.yy.c airport.c
```

כאן האופציה o- מציינת את שם הקובץ שהוא התוצר של הקומפילציה (במקרה זה שם הקובץ הוא airport.exe).

ההמשך (הרצת התוכנית על קובץ קלט) כמו בדוגמא (שלב 4) שמופיעה למעלה
בתיאור של הכנת תכנית בעזרת flex & bison.

בכל מקרה מומלץ להשתמש ב- makefile (ראו דוגמאות בתכניות לדוגמא שמוזכרות בהמשך)

התוכנות של flex & bison נמצאות ב- moodle בתיקיה על bison
(אלו מתאימות להרצה על Windows).

דוגמאות לתוכניות

בתיקיה של תרגילי הבית של סמסטר 2025 ב (ב- moodle) יש דוגמאות
לתכניות שהן בסגנון של תרגיל הבית.

יש שתי תכניות כאלו. לכל אחת מהן 2 גרסאות: אחת מהן נכתבה עם flex & bison
והשנייה עם recursive descent parser (ועם flex).

באחת התכניות הקלט כולל רשימת שירים ("מדונה"). בדוגמא השנייה הקלט מתאר נצחונות
של שחקני טניס.

בנוסף לכך אפשר להסתכל בפתרונות של בחינות (בתיקית הבחינות) מהשנים
האחרונות. השאלה הראשונה בכל בחינה עוסקת ב- flex & bison.

בהצלחה!