

מושגים בשפות תכנות

תרגיל 1

להגשה עד 13/04/2015

בתרגיל זה נממש את האלגוריתמים שלמדנו לניתוח דקדוק חסר הקשר, וכן נבנה parser מסוג LL(1) לשפת JSON (JavaScript Object Notation).

הנחיות כלליות:

- בתרגיל נעשה שימוש ב Python בגרסת 2.7. כל הפתרונות צריכים לרוץ עם גרסה זו.
- כל השינויים בקבצים צריכים להיות במקומות המסומנים בהם. אין לשנות בקבצים דבר מלבד במקומות אלה.
- בכל שאלות המימוש באחריותך לבדוק את הקוד שכתבת על דוגמאות נוספות ולוודא את נכונותו.

1. הקובץ `grammar.py` מכיל פונקציות לניתוח דקדוק לפי האלגוריתמים שלמדנו: מציאת הקבוצות, `NULLABLE`, `FIRST`, `FOLLOW`, `SELECT`, ובדיקה האם הדקדוק הוא LL(1). הפונקציה `calculate_nullable` והפונקציה `analyze_grammar` נתונות במלואן. השלימי את המימוש לפונקציות `calculate_first`, `calculate_follow`, `calculate_select` בהתאם לאלגוריתמים שלמדנו. לצורך בדיקה, הקובץ מכיל את הדקדוק שראינו בתרגול - לאחר השלמת הפונקציות וודאי שהתוצאה שמתקבלת זהה לתוצאה שמופיעה בשקפי התרגול. הערה: בדקדוק בקובץ מופיע הכלל שלא נרשם במפורש בתרגול:

`P -> S EOF`

2. להלן דקדוק לחלק משפת JSON (JavaScript Object Notation):

```
json -> obj EOF
obj -> { }
obj -> { members }
members -> keyvalue
members -> members , members
keyvalue -> string : value
value -> string
value -> int
value -> obj
```

האם המילים הבאות נמצאות בשפה של הדקדוק הנ"ל? אם כן, הראי עץ גזירה שמוכיח זאת, אם לא, הסבירי מדוע אין עץ גזירה כזה:

- `{"course":"concepts in PL", "ex":1, "grade":100}`
- `{"course":"concepts in PL", "ex":1, "grade":{"100}}`

הערה: הניחי שמחרוזות בגרשיים מתורגמות לטוקן `string`, ומספרים שלמים לטוקן `int`.

3. הראי שהדקדוק משאלה 2 הוא רב משמעי ע"י מציאת שני עצי גזירה שונים לאותה מילה.

4.

a. לאחר שהשלמת את שאלה 1, הוסיפי לקובץ `grammar.py` את הדקדוק משאלה 2, והריצי עליו את הפונקציה `analyze_grammar`. הדקדוק הוא רב משמעי ולכן אינו LL(1). הסבירי כיצד זה מתבטא בקבוצות ה `SELECT` ?

b. מצאי דקדוק חד משמעי שמתאר אותה שפה כמו הדקדוק משאלה 2, הוסיפי אותו לקובץ `grammar.py` (במידת הצורך הוסיפי non-terminals נוספים לקובץ `symbols.py`). נתחי את הדקדוק שמצאת ע"י הרצת `grammar.py`, האם הדקדוק שמצאת הוא LL(1)?

c. האם בדקדוק שיצרת יש רקורסיה שמאלית? האם בדקדוק שיצרת ניתן להפעיל LEFT FACTORING? אם כן, בטלי את שניהם עד שתקבלי דקדוק LL(1). הוסיפי את הדקדוק המתקבל לקובץ grammar.py ונתחי אותו בכדי להיווכח שהוא אכן LL(1).

הערה: את הדקדוקים מסעיפים a,b,c יש להוסיף לקובץ grammar.py במקומות המסומנים בקובץ ואין צורך להגישם בכתב. כן נדרש להגיש קובץ בשם grammar_output.txt עם הפלט של grammar.py לאחר הוספת כל הדקדוקים ו uncommenting של השורות המתאימות בפונקציה main.

5. בשאלה זו תבני parser מסוג LL(1) עבור שפת JSON תוך שימוש בדקדוק שמצאת בשאלה 4. הקובץ lexer.py מכיל lexer (נקרא גם scanner) עבור שפת JSON. הקובץ parser.py מכיל שלד של parser מסוג LL(1). השלימי את הפונקציות החסרות בקובץ parser.py: פונקציה אחת לכל non-terminal בדקדוק, לפי השיטה שלמדנו ולפי קבוצת ה SELECT שקיבלת בסעיף 4c.

כל פונקציה צריכה להחזיר עץ גזירה (parse tree), כך שבכל הפעלה של כלל מהצורה:

$$A \rightarrow S_1 \dots S_n$$

ערך החזרה יהיה:

$$(A, (X_1, \dots, X_n))$$

כאשר אם S_i הוא non-terminal, אז X_i הוא עץ שמתקבל מהקריאה לפונקציה `parse_Si`, ואם S_i הוא terminal, אז X_i הוא עלה שמוחזר מהקריאה לפונקציה `match`. בפרט, עבור הפעלת כלל מהצורה:

$$A \rightarrow \epsilon$$

ערך החזרה יהיה:

$$(A, ())$$

את עץ הגזירה ניתן להציג באופן גרפי. הקובץ `tree_to_dot.py` מכיל פונקציה שממירה עץ גזירה לפורמט `dot`, שניתן להציג ע"י התקנת Graphviz, או ע"י העתקת פלט בפורמט `dot` לאתר: <http://www.webgraphviz.com>. הקובץ `json_example.json` מכיל קלט לדוגמה - וודאי שה `parser` שבנית מקבל את הקלט הזה ובונה עבורו עץ גזירה נכון (בדקי ע"י הצגה גרפית של עץ הגזירה, שייכתב לקובץ `json_example.gv`). וודאי גם שה `parser` שבנית מדווח על שגיאת תחביר (`syntax error`) עבור קלטים שאינם חוקיים בשפה: בני לפחות קלט אחד כזה בקובץ `json_bad_example.json`. שימי לב: בהרצת הקובץ `parser.py` הפונקציה `main` תקרא את הקובץ `json_example.json`, תפעיל את ה `parser` ותכתוב את עץ הגזירה בפורמט `dot` לקובץ `json_example.gv`. יש להגיש קובץ פלט זה.

6. **בנוס 1**

בשפת JSON ניתן להביע מערכים ע"י שימוש בסוגריים מרובעים. לדוגמה:

```
{ "array": [1,
            "two",
            { "object inside array?": "yes",
              "empty array": []
            }
          ] }
```

הוסיפי ל `grammar.py` דקדוק LL(1) שיתמוך גם במערכים, והתאימי את ה `parser` שבנית לדקדוק המורחב. בדקי את ה `parser` שבנית על קלט JSON שמכיל מערכים ובדקי שהוא בונה עץ גזירה נכון (את הדוגמה שבנית שמרי בקובץ `json_array_example.json`).

7. **בנוס 2**

לכל שפה רגולרית קיים דקדוק חסר הקשר. הוכיחי זאת באינדוקציה מבנית על ביטויים רגולריים.