מושגים בשפות תכנות

תרגיל 1

להגשה עד 23/11/2016

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.

שאלה 1

נתבונן בדקדוק הבא עבור ביטויים בוליאניים שמכיל את האופרטור and והאופרטור not:

```
B → B and B | not B | ( B ) | id
```

כאשר B הוא non-terminal, וה terminals הם:

```
and, not, "(", ")", id
```

א. הראי שהדקדוק אינו חד משמעי ע"י מתן שני עצי גזירה שונים לאותה מילה.

ב. כתבי דקדוק חד משמעי שקול (עבור אותה שפה), כך שלאופרטור not תהיה עדיפות על אופטור and, ולאופטור and תהיה אסוציאטיביות שמאלית. לדוגמה, המילה הבאה:

```
not x and not y and z
```

מתפרשת כ:

```
((not x) and (not y)) and z
```

ג. האם הדקדוק שרשמת בסעיף ב' הוא LL(1)? אם לא, כתבי דקדוק LL(1) שקול.

ד. ציירי עץ גזירה לפי הדקדוק שכתבת בסעיף ג', עבור המילה הבאה:

not not (x and y and z)

שאלה 2

הקובץ grammar.py מכיל פונקציות לניתוח דקדוק לפי האלגוריתמים שלמדנו: מציאת הקבוצות NULLABLE, FIRST, FOLLOW, SELECT, ובדיקה האם הדקוק 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.

שאלה 3

להלן דקדוק חסר הקשר לחלק משפת JSON (JavaScript Object Notation):

```
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:

A. `{"course":"concepts in PL", "ex":1, "grade":100}`
B. `{"course":"concepts in PL", "ex":1, "grade":{100}}`

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

שאלה 4

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

שאלה 5

A. לאחר שהשלמת את שאלה 2, הוסיפי לקובץ grammar.py את הדקדוק משאלה 3, 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 3, 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.

שאלה 6

בשאלה זו תבני parser מסוג LL(1) עבור שפת JSON תוך שימוש בדקדוק שמצאת בשאלה 5. 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 קבוצות ה SELECT שקיבלת בסעיף 5c.
כל פונקציה צריכה להחזיר עץ גזירה (parse tree), כך שבכל הפעלה של כלל מהצורה:

A -> S1 … Sn

ערך החזרה יהיה:

(A, (X1, …, Xn))

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

A -> ε

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.

שאלה 7 (בונוס)

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).

שאלה 8 (בונוס)

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

בהצלחה!