

COMPILERS

1Η ΦΑΣΗ 2017

Βασίλειος Δρέττας – 1115201300042

Δημήτριος Κωνσταντάκης – 1115201300079

Η εργασία έχει υλοποιηθεί σε Java.

Αρχεία:

Parser.grammar :

Το αρχείο αυτό χρησιμοποιείται για την λεκτική και συντακτική ανάλυση ενός κώδικα σε γλώσσα Grace. Πιο αναλυτικά έχουν υλοποιηθεί τα :

Helpers :

Για τα comment έχουν υλοποιηθεί δύο είδη, το παραδοσιακό (traditional) το οποίο μοιάζει με το `/* */` , με τη διαφορά ότι θέλει υποχρεωτικά δύο δολάρια στην αρχή και δύο δολάρια στο τέλος. Οι ενδιάμεσοι χαρακτήρες μπορεί να είναι οτιδήποτε , με τον περιορισμό όμως ότι αν υπάρχει ένα δολάριο εσωτερικά ο επόμενος χαρακτήρας πρέπει να είναι μη δολάριο(έτσι αποφεύγεται η χρήση εμφωλευμένων σχολίων). Το άλλο είδος comment είναι το `end_of_line_comment` , το οποίο έχει ένα δολάριο στην αρχή και μετά παίρνει έναν οποιοδήποτε χαρακτήρα εκτός από `eol`. Οι περιορισμοί είναι ότι επόμενως χαρακτήρας από το δολάριο δεν πρέπει να είναι δολάριο και ο τελευταίος χαρακτήρας είναι υποχρεωτικά `eol`. Πχ. Αν δοθεί `$$comment$$$` αυτό διαβάζεται ως traditional comment `$$comment$$` και απλό comment `$`.

Το helper `input_string` μπορεί να έχει οποιοδήποτε χαρακτήρα εκτός από `"` , `\` καθώς αυτά τα σύμβολα χρησιμοποιούνται με τη βοήθεια των escape sequence. Το helper `input_char` ορίζεται με τον ίδιο τρόπο όπως και το `input_string` με τη διαφορά ότι δεν δέχεται επιπλέον τον χαρακτήρα `'` .

Tokens :

Έχουν οριστεί όλες οι λέξεις κλειδιά και οι συμβολικοί τελεστές. Επίσης ορίζονται όλες οι σταθερές , τα white spaces και το όνομα μιας μεταβλητής (identifier) . Τα comments και τα white spaces γίνονται **ignore**. Οι `const char` σταθερές μπορούν να πάρουν οποιοδήποτε κοινό χαρακτήρα εκτός από `"` , `'` , `\` ή ένα escape sequence, μέσα σε `' '`.

Productions :

Τα productions έχουν υλοποιηθεί όπως περιγράφονται στην εκφώνηση της άσκησης (σελίδα -11-). Έχουν χρησιμοποιηθεί και ενδιάμεσες καταστάσεις καθώς δεν επιτρέπονται οι παραγωγές που περιλαμβάνουν εκφράσεις με παρενθέσεις που έχουν παραπάνω από ένα στοιχεία. Ειδικότερα για την αποφυγή των Shift/Reduce Conflicts χρησιμοποιούνται production για `if without else` και `if with else`. Επίσης, για το `condition` και για τις απλές πράξεις, η σειρά με την οποία γίνονται οι παραγωγές είναι ίδια με την προτεραιότητα των (λογικών και μαθηματικών) τελεστών αντίστοιχα.

Print.java :

Χρησιμοποιεί το concrete syntax tree για την εκτύπωση των σημαντικότερων κόμβων του δέντρου. Από το αρχείο Analysis.java δημιουργήθηκαν οι συναρτήσεις οι οποίες :

- Εκτυπώνουν το όνομα, τις παραμέτρους και τον τύπο μιας συνάρτησης όταν αυτή δηλώνεται. Αν δεν υπάρχουν παράμετροι εκτυπώνει null. Αν η συνάρτηση δηλώνεται εσωτερικά σε μία άλλη εμφανίζεται μήνυμα "local function definition".
- Εκτυπώνουν το όνομα και τον τύπο μιας μεταβλητής όταν αυτή δηλώνεται (variable definition).
- Όταν μπαίνουμε ή βγαίνουμε από ένα block τότε εκτυπώνεται αντίστοιχο μήνυμα (block in / block out).
- Όταν έχουμε ανάθεση εκτυπώνεται το όνομα της μεταβλητής και η τιμή που ανατίθεται στην μεταβλητή.
- Όταν έχουμε κάποιο statement εμφανίζεται το όνομα του και συγκεκριμένα ποια περίπτωση έχουμε (while, if with else, if without else) . Ακολούθως εκτυπώνεται το condition και το μήνυμα "—Enter "... Statement. Όταν βγει από το statement εμφανίζεται μήνυμα "_Exit "... Statement.
- Όταν γίνεται κλήση μιας συνάρτησης τότε εκτυπώνεται το όνομα της και οι παράμετροι που παίρνει

Main.java :

Παίρνει ως όρισμα ένα αρχείο , φτιάχνει το δέντρο και το εκτυπώνει μέσω της Print.

Παρατηρήσεις: Τρέξαμε όλα τα παραδείγματα (hello.grace , bsort.grace, ...) και η μεταγλώττιση γινόταν σωστά. Μόνο στο bsort μας πέταγε error , καθώς έλειπε ένα semi , αλλά το προσθέσαμε εκεί που μας έλεγε ότι το περίμενε και μετά λειτουργούσε κανονικά.

Εντολές : mvn clean package

java -cp target/compiler-1.0-SNAPSHOT.jar compiler.Main fileName