Blue text on a black background

Description automatically generated

***Σχολή Μηχανικών***

***Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών***

***Εργαστήριο «Μεταγλωττιστές»***

***Μέρος Β-2: Σύνδεση κώδικα flex με κώδικα bison***

***Μέρος Β-3: Διαχείριση λεκτικών και συντακτικών προειδοποιητικών λαθών***

***Ημερομηνία Αποστολής: 18/5/2024***

***Τμήμα Β2 - Ομάδα 2***

*ΚΟΝΤΟΥΛΗΣ ΔΗΜΗΤΡΙΟΣ 21390095*

*ΜΕΝΤΖΕΛΟΣ ΑΓΓΕΛΟΣ ΚΩΝΣΤΑΝΤΙΝΟΣ 21390132*

*ΒΑΡΣΟΥ ΕΥΦΡΟΣΥΝΗ 21390021*

*ΑΛΕΞΟΠΟΥΛΟΣ ΛΕΩΝΙΔΑΣ 21390006*

Περιεχόμενα

[1. Εισαγωγή 3](#_Toc169361753)

[2. Τεκμηρίωση Γραμματικής 3](#_Toc169361754)

[2.1 Λεκτική Ανάλυση 3](#_Toc169361755)

[2.2 Συντακτική Ανάλυση 3](#_Toc169361756)

[2.3 Σημασιολογική Ανάλυση 3](#_Toc169361757)

[3. Παρουσίαση Εξαντλητικών ελέγχων 3](#_Toc169361758)

[4. Ανάλυση προβλημάτων/ελλείψεων 3](#_Toc169361759)

[4.1 Μέρος Β-2 3](#_Toc169361760)

[4.2 Μέρος Β-3 3](#_Toc169361761)

[5. Επίλογος 3](#_Toc169361762)

[5.1 Τελικά Σχόλια: 3](#_Toc169361763)

[Υποσημείωση/Διόρθωση για Μέρος Α-3 3](#_Toc169361764)

[6. Ανάλυση Αρμοδιοτήτων 4](#_Toc169361765)

[3.1 Γενικές Αρμοδιότητες 4](#_Toc169361766)

[3.2 Υλοποίηση Word 4](#_Toc169361767)

# 1. Εισαγωγή

Το παρών έγγραφο αποτελεί η κωδικοποίηση και σχεδιασμού ενός Συντακτικού αναλυτή με την γεννήτρια Bison. Ειδικότερα, επεκτείναμε την λογική που υλοποιήσαμε για το Μέρος Α-3 της λεκτικής ανάλυσης με την γεννήτρια Flex (αναγνώριση λεκτικών μονάδων) για να μπορούμε πλέον να αναγνωρίζουμε εκφράσεις με τον συνδυασμό των λεκτικών μονάδων. Δηλαδή, ο κώδικας Bison που υλοποιήσαμε διαπιστώνει την σωστή γραμματική της γλώσσας με είσοδο ένα πηγαίο κώδικα. Επιπλέον, ο Bison διαπιστώνει και την σημασιολογική ανάλυση δηλαδή εκτός αν γραμματικά είναι σωστή η έκφραση αλλά και επιπλέον αν μπορεί να οριστεί π.χ. απροσδιόριστες μεταβλητές, απροσδιόριστοι τύποι. Τέλος, ο κώδικας Bison κάνει και διαχείριση λεκτικών και συντακτικών προειδοποιητικών λαθών που ενημερώνει τον χρήστη που έγινε το λάθος.

Η δομή του ακόλουθου εγγράφου αποτελείται από τα κεφάλαια τεκμηρίωση της γραμματικής που υλοποιήσαμε στον κώδικα Bison, της παρουσίασης εξαντλητικών ελέγχων με διαφορετικά αρχεία εισόδου-εξόδου, της ανάλυσης προβλημάτων ελλείψεων που αντιμετωπίσαμε και υπάρχουν στον κώδικα και του επίλογου που κλείνουμε με κάποια τελικά γενικά σχόλια σχετικά με την υλοποίηση της εργασίας.

# 2. Τεκμηρίωση

## 2.1 Γραμματική (Μέρος Β-2)

### 2.1.1 Λεκτική Ανάλυση

ΑΓΓΕΛΟΣ

### 2.1.2 Συντακτική Ανάλυση

Γενικά, η συντακτική ανάλυση έχει αναλυθεί σε πολλές γραμματικές που αφορούν την δήλωση μεταβλητών, τις Built-in συναρτήσεις της Uni-C, τις δηλώσεις συναρτήσεων χρήστη, την δήλωση απλών εκφράσεων και τις σύνθετες δηλώσεις (if, while, for). Αυτές τις γραμματικές τις έχουμε ενσωματώσει στην γενική γραμματική valid που είναι η γραμματική που τελειώνει το πρόγραμμα (program valid). Είναι δηλαδή οι γραμματικές declaration (δήλωση μεταβλητών), func\_call (bulit\_in συναρτήσεις και κλήση συνάρτησης χρήστη), assignment (Αναθέσεις τιμών σε αναγνωριστικά), func\_decl (δήλωση συνάρτηση χρήστη), if\_while\_grammar και for\_grammar (σύνθετες δηλώσεις). Επιπλέον, η valid έχει και την λεκτική μονάδα EOP που συμβολίζει το τέλος του αρχείου, και οπότε καλεί την συνάρτηση print\_report() για το τύπωμα τον σωστών ή λανθασμένων λεκτικών και συντακτικών λέξεων στο τέλος της συντακτικής ανάλυσης.

Στις γραμματικές keyword, operator, num, var και str ουσιαστικά δεχόμαστε τα tokens για το κάθε ένα αντίστοιχα και αντιγράφουμε το κείμενο που δόθηκε και το αποδίδουμε στη σημασιολογική τιμή του κανόνα ($$). Το κάνουμε αυτό γιατί π.χ. για τους operators στον γραμματικό κανόνα operator\_val κάνουμε την αντιστοίχιση σε κωδικούς (id) τον κάθε operator (για την σημασιολογική ανάλυση που χρειάζεται στους έπειτα κανόνες).

Σε γενικές γραμμές στον κώδικα, έχουμε και βοηθητικούς κανόνες (όπως π.χ help\_int, help\_float κτλ) που χρησιμεύουν στην παράθεση ίδιων τύπων λέξεων να παρατίθενται είτε μόνες τους είτε με κόμματα (DELIMITER) όταν είναι δύο και άνω. Αυτούς τους τύπους γραμματικών κανόνων τους θέλουμε είτε στους πίνακες είτε στην ομαδική ανάθεση τιμών κτλ. Οπότε γενικά χρησιμοποιούνται σε πολλές γραμματικές της Uni-C.

Τέλος, H γραμματική expr ορίζει τις πράξεις και τις συγκρίσεις μεταξύ αριθμών, η οποία στην συνέχεια θεωρείται και γραμματική num (που περιέχει όλους τους αριθμούς). Αυτό το κάνουμε για να δέχεται π.χ. η δομή if και while και αριθμούς (που στην C αν είναι μεγαλύτερες του 1 σημαίνουν true) και πράξεις και συγκρίσεις.

Ο κάθε γραμματικός κανόνας αναλύεται και μέσα σε σχόλια στον κώδικα αλλά γενικά για τον σχεδιασμό και η λειτουργεία των υπόλοιπων βασίστηκε στους πάνω πολύ βασικούς κανόνες της γραμματικής μας που τις χρησιμοποιούμε σε πολλά σημεία στο πρόγραμμά μας.

### 2.1.3 Σημασιολογική Ανάλυση

ΚΟΝΤΟΥΛΗΣ

## 2.2 Διαχείριση Προειδοποιητικών λαθών (Μέρος Β-3)

ΑΓΓΕΛΟΣ

# 3. Παρουσίαση Εξαντλητικών ελέγχων

ΑΓΓΕΛΟΣ Β-3 και θα μας πει άμα θέλει βοήθεια

ΚΟΝΤΟΥΛΗΣ 3-4 αρχεία από Β-2 και ότι πει ο άγγελος για β-3

ΦΡΟΣΩ 3-4 αρχεία από Β-2 και ότι πει ο άγγελος για β-3

# 4. Ανάλυση προβλημάτων/ελλείψεων

Γενικά, το πρόγραμμα μεταγλωττίζεται και τρέχει με τα αρχεία εισόδου που φτιάξαμε κανονικά. Τρέχει επίσης και από το stdin αν δεν δοθεί όνομα αρχείο στην γραμμή εντολών. Στην μεταγλώττιση απλώς, υπάρχουν προειδοποιητικά σφάλματα (warnings) που αφορούν σε shift/reduce conflicts. Η shift/reduce είναι μια κατάσταση όπου ο αναλυτής μπορεί είτε να μετατοπίσει (shift) το επόμενο σύμβολο εισόδου είτε να μειώσει τη στοίβα (reduce) χρησιμοποιώντας έναν κανόνα παραγωγής. Αυτό γίνεται δηλαδή σε κάποιους γραμματικούς κανόνες μας που είναι λογικά είναι ίδιοι μεταξύ τους σε κάποια σημεία που δεν καταφέραμε να τις υλοποιήσουμε χωρίς να εμφανίζονται αυτά τα warnings.

## 4.1 Μέρος Β-2

Γενικά, έχουμε υλοποιήσει τις δομές που περιγράφονται από το κεφάλαιο 2.1 του έγγραφου που αναλύει την συντακτική ανάλυση της Uni-C. Δεν έχουμε πραγματοποιήσει τους κανόνες σύνδεσης φυσικών γραμμών (κεφάλαιο 2.1.2.1). Ειδικότερα, στην γραμματική declaration (ανάθεση τιμών) για την διατύπωση π.χ int a = a + 1; δεν κάνουμε έλεγχο αν είναι του ίδιου τύπου οπότε μπορεί να γραφτεί int a = a + 1;. Ειδικότερα, π.χ για την κλήση συνάρτησης δεν έχουμε κάνει έλεγχο αν έχει δηλωθεί πιο πριν. Και γενικά σε πολλές γραμματικές δεν έχουμε κάνει τόσο εξαντλητικό έλεγχο για τον τύπο και γενικά για την σημασιολογική ανάλυση, αλλά έχουμε επικεντρωθεί στην γραμματική. Στον περισσότερο βαθμό έχουμε επικεντρωθεί στα παραδείγματα που έχουν δοθεί στο έγγραφο της Uni-C και δεν έχουμε επεκτείνει τις γραμματικές π.χ για όλα τα keywords όπως return στο τέλος της συνάρτησης, struct, case κτλ. Επίσης, στο σώμα της συνάρτησης έχουμε καταφέρει μόνο να γράφονται οι γραμματικές declaration, assignment, func\_call και οι δομές if while και for μπορούν μόνο μια φορά να γράφονται μέσα σε brackets. Αυτό πιθανόν οφείλεται στον τρόπο που έχουμε ορίσει στους γραμματικούς κανόνες για την λεκτική μονάδα END (\n) στην γραμματική all και δεν υπάρχει ο γραμματικός κανόνας για να είναι σωστό αλλά δεν καταφέραμε να τα κάνουμε να δουλεύει. Οι δομές if, while, for δουλεύουν όμως από μόνες τους όταν δεν είναι μέσα σε body από bracket.

Επομένως, ότι άλλο λάθος υπάρχει είναι από απερισκεψία μας και λόγω της πολυπλοκότητας όλων των γραμματικών στο αρχείο. Στους εξαντλητικούς ελέγχους, ουσιαστικά αναλύσαμε τι βγαίνει σωστό αλλά γενικά όπως είναι ορισμένες οι γραμματικές μπορεί να υπάρχουν και συντακτικά σωστά που δεν προβλέψαμε λόγω των συνδυασμών και πάλι στην πολυπλοκότητα των γραμματικων.

## 4.2 Μέρος Β-3

ΑΓΓΕΛΟΣ

# 5. Επίλογος

## 5.1 Τελικά Σχόλια

ΚΟΝΤΟΥΛΗΣ

Υποσημείωση/Διόρθωση για Μέρος Α-3: Στο μέρος Α-3 στον κώδικα Flex, έγινε ένα συντακτικό λάθος στην διατύπωση της κανονικής έκφρασης για την λεκτική μονάδα που αφορά τα Strings. Στο flex αρχείο στην γραμμή 19 είναι διατυπωμένη ως εξής:

STRINGS "(\\[\\n\"]|[^\n\\\"])\*"

Αν και σημασιολογικά αυτή είναι η κανονική έκφραση για τα String (που περιγράψαμε και ελέγξαμε στο μέρος Α-2) στον Flex έπρεπε να διατυπωθεί ως εξής:

STRINGS \"(\\[\\n\"]|[^\n\\\"])\*\"

Δηλαδή, η μόνο διαφορά είναι στα εξωτερικά “ που έπρεπε να γραφτούν \” για να αναγνωρίζει τον χαρακτήρα. Κάνοντας αυτήν την αλλαγή τα Strings αναγνωρίζονται κανονικά από τον flex και αυτήν την αλλαγή την έχουμε κάνει στην μορφή του lexical\_analyzer.l για το μέρος Β.

# 6. Ανάλυση Αρμοδιοτήτων

Το μέλος της ομάδας «Εντερίσα Γκιόζι» που βοήθησε στην δημιουργία Α μέρους (Α-2, Α-3) δεν συμμετείχε καθόλου σε όλη την διάρκεια υλοποίησης της εργασίας μέρους Β (Β-2,Β-3) και αποχώρησε οικειοθελώς πριν την έναρξη της υλοποίησης. Για αυτόν τον λόγο, παρακάτω αναγράφονται οι αρμοδιότητες για τα υπόλοιπα τέσσερα μέλη καθώς και στο εξώφυλλο.

## 3.1 Γενικές Αρμοδιότητες

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **Κοντούλης Δημήτριος (21390095)** | **Μεντζέλος Άγγελος Κωνσταντίνος**  **(21390132)** | **Βάρσου**  **Ευφροσύνη (21390021)** | **Αλεξόπουλος Λεωνίδας (2139006)** |
| Υλοποίηση κώδικα (Γραμματικών) | Δηλώσεις Μεταβλητών |  |  | √ Πίνακες, Ιδίου τύπου αναθέσεις | √ Απλή δήλωση |
| Built-in συναρτήσεις | √ cmp, scan |  | √ len, print |  |
| Δήλωση συναρτήσεων χρήστη | √ Δήλωση συνάρτησης | √ Κλήση συνάρτησης |  |  |
| Δήλωση απλών εκφράσεων |  | √ Αριθμητικές Εκφράσεις, Συγκρίσεις, Συνένωση πινάκων | √ Αναθέσεις τιμών σε αναγνωριστικά (και ομαδοποιήση) |  |
| Σύνθετες δηλώσεις | √ For δομή | √ While δομή | √ If δομή |  |
| Καταμέτρηση λέξεων |  |  |  | √ Σωστών και Λανθασμένων |
| Καταμέτρηση εκφράσεων |  | √ Λανθασμένων | √ Σωστών |  |
| Τύπωμα ανάλυσης |  |  | √ |  |
| Είσοδος από αρχείο |  | √ Διόρθωση | √ Υλοποίηση |  |
| Διαχείριση Προειδοποιητικών λαθών | | √ | √ |  |  |
| Δημιουργία αρχείων εισόδου | | √ | √ | √ | √ |
| Έλεγχος εξόδου | | √ | √ |  |  |
| Σχολιασμός κώδικα | | √ Αναλυτικός |  | √ Επιμέρους |  |

## 3.2 Υλοποίηση Word

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **Κοντούλης Δημήτριος (21390095)** | **Μεντζέλος Άγγελος Κωνσταντίνος**  **(21390132)** | **Βάρσου**  **Ευφροσύνη (21390021)** | **Αλεξόπουλος Λεωνίδας (2139006)** |
| 1. Εισαγωγή | |  |  | **√** |  |
| 2. Τεκμηρίωση Γραμματικής | 2.1 Λεκτική Ανάλυση |  | **√** |  |  |
| 2.2 Συντακτική Ανάλυση |  |  | **√** |  |
| 2.3 Σημασιολογική Ανάλυση | **√** |  |  |  |
| 3. Παρουσίαση Εξαντλητικών Ελέγχων | | **√** | **√** | **√** | **√** |
| 4. Ανάλυση προβλημάτων /ελλείψεων | | **√** |  |  |  |
| 5. Επίλογος | |  | **√** |  |  |
| 6. Ανάλυση αρμοδιοτήτων | |  |  | **√** |  |