Μεταφραστές-ReadMe

Εαρινό Εξάμηνο 2020-2021

Πηνελόπη Ελευθεριάδη 3221

Τσαούση Αθανασία Δανάη 3349

Περιεχόμενα

1η φάση : Λεκτικός και Συντακτικός αναλυτής

Λεκτικός αναλυτής :Έχουμε δημιουργήσει μια κλάση Token με πεδία το τύπο της λεκτικής μονάδας ,την ίδια τη λεκτική μονάδα καθώς και τον αριθμό της γραμμής στην οποία βρισκόμαστε .Το διάβασμα στο λεκτικό αναλυτή γίνεται ανά χαρακτήρα (π.χ. ch =InputFile.read(1)) και επιστρέφει αποτέλεσμα αναλόγως το τύπο της λεκτικής μονάδας σε κάθε περίπτωση. Ο λεκτικός αναλυτής(lex1()) είναι μια συνάρτηση και επιστρέφει κάθε φορά που καλείται την επόμενη λεκτική μονάδα ως ένα αντικείμενο Token.Επίσης, παραλείπονται οι λευκοί χαρακτήρες και αγνοούνται τα σχόλια ,που βρίσκονται ανάμεσα στα σύμβολα “#...#”.Σε περίπτωση που διαβαστεί παραπάνω χαρακτήρας που δεν ανήκει τελικά στην λεκτική μονάδα ,αυτό αναιρείται με την χρήση της εντολής inputFile.seek(inputFile.tell()-1).

Η κλήση του λεκτικού αναλυτή γινεται μεσα στις συναρτήσεις του συντακτικού αναλυτή σύμφωνα με την γραμματική της γλώσσας Cimple.

Τέλος, ελέγχουμε αν οι ακέραιες σταθερές εχούν τιμές απο -(2^32-1)εώς (2^32-1)και τα ονόματα των μεταβλητών να έχουν ορθή μορφή.

Συντακτικός αναλυτής:

Χρησιμοποιείται ώστε να διαπιστωθεί αν ένα πρόγραμμα ανήκει στην γλώσσα Cimple.

Συναρτήσεις συντακτικού :

(1) program() :Εδώ καθορίζεται η εκκίνηση του προγράμματος. Είναι υποχρεωτικό το πρόγραμμα να ξεκινά με την λέξη "**program".**

Έπειτα καλείται η ID() και στην συνέχεια η **block**().

(2) block():Σε αυτή καλούνται διαδοχικά οι συναρτήσεις

declarations(), **subprograms**() και **statements**().

(3) declarations():Είναι υπεύθυνη για τον έλεγχο των δηλώσεων των μεταβλητών,όπου ,όταν υπάρχουν, ξεκινούν με την λέξη **declare** και έπειτα καλείται η συνάρτηση varlist() και επιστρέφοντας ελέγχεται η ύπαρξη ερωτηματικού.

(4) varlist(): Κρατάει μια λίστα απο μεταβλητές που δηλώθηκαν και σε περίπτωση που είναι πάνω απο μία ελέγχει την ύπαρξη "," ανάμεσα .Στην περίπτωση που ο τύπος της λεκτικής μονάδας δεν είναι identifier επιστρέφεται το κενό.

(5) subprograms():Διαπιστώνει αν υπάρχουν υποπρογράμματα ή όχι με την κλήση της **subprogram**().

(6) subprogram(): Ελέγχει την δομή του υποπρογράμματος(function ή procedure).Παίρνει το όνομα του,ελέγχει τα ορίσματα καλώντας την formalparlist() και ξανακαλεί την block().

(7) formalparlist :Είναι υπέυθυνη για τις παραμέτρους των υποπρογραμμάτων .Κάθε παράμετρος χωρίζεται απο την επόμενη με την χρήση “,”.Καλεί την formalparitem() για την διαχείρηση κάθε παραμέτρου ξεχωριστά.

(8) formalparitem() :Διαβάζει τα ονόματα των παραμέτρων,εφόσον υπάρχουν οι λέξεις “in “ ή “inout”,που δηλώνουν πέρασμα με τιμή ή αναφορά.

(9) statements(): Ελέγχει την ύπαρξη αγκυλών και εσωτερικά καλεί την statement().Επίσης, ελέγχει οτι κάθε statement χωρίζεται με “;”.

(10) statement():Αναγνωρίζει το token και ανάλογα καλεί την κατάλληλη συνάρτηση.

(11) assignStat(): Είναι η εκχώρηση.Με την ύπαρξη του σύμβολου “:=” καλει την expression().

(12) ifStat():Ελέγχει τα ορίσματα μεσα στις “()” για την if καλώντας την condition() .Έπειτα, καλεί την statements() για να ελέγξει την δομή μέσα. Ακόμα καλεί την συνάρτηση elsepart() ,η οποία ενεργοποιείται με την ύπαρξη της λέξης “else”.

(13) whileStat() : Ελέγχει τα ορίσματα μεσα στις “()” για την if καλώντας την condition() .Μετά καλεί την statements().

(14) switchcaseStat(): Εφόσον υπάρχει η λέξη switch(άρα την έχει καλέσει η statement),περιμένει την λέξη case και παίρνει τα ορίσματα με την condition() και καλεί την statements .

(15) forcaseStat(): Περιμένει την λέξη case και παίρνει τα ορίσματα με την condition() και καλεί την statements .Επίσης ,ελέγχει το default και καλεί ξανά την statements.

(16) incaseStat(): Περιμένει την λέξη case και παίρνει τα ορίσματα με την condition() και καλεί την statements .

(17) callStat(): Περιμένει κάποιον identifier και μετα καλεί την actualparlist() για τις παραμέτρους.

(18) returnStat():Ελέγχει οτι υπάρχουν () και καλεί την expression().

(19) inputStat():Ελέγχει οτι υπάρχουν () και μεσα υπάρχει identifier.

(20) printStat():Καλεί την expression() εφόσον υπάρχουν ().

(21) actualparlist() : Ελέγχει αν είναι σωστοί οι παράμετροι σύμφωνα με τη γραμματική της Cimple.Για κάθε παράμετρο καλεί την actualparitem() και ελέγχει αν οι παράμετροι χωρίζονται με κόμμα .Αν δεν υπάρχουν παράμετροι επιστρέφει το κενό.

(22) actualparitem() : Ελέγχει αν υπάρχει ο τύπος της παραμέτρου (in/inout) και καλεί αντίστοιχα τις συναρτήσεις expression() /ID().

(23) condition() : Καλεί την boolterm() και κάθε φορά που συναντάει τη λεκτική μονάδα «or» ξανακαλεί την boolterm().

(24) boolterm() : Καλεί την boolfactor() και κάθε φορά που συναντάει τη λεκτική μονάδα «and» ξανακαλεί την boolfactor().

(25) boolfactor() : Ελέγχει 3 περιπτώσεις δομών. Η πρώτη είναι να περιέχει τη λεκτική μονάδα «not» ,αφού συναντήσει τη λεκτική μονάδα «[» κάνει κλήση τής condition() και έπειτα όταν επιστρέψει από τη condition() πρέπει υποχρεωτικά να συναντήσει τη λεκτική μονάδα «]». Η δεύτερη είναι αφού συναντήσει τη λεκτική μονάδα «[» κάνει κλήση τής condition() και έπειτα όταν επιστρέψει από τη condition() πρέπει υποχρεωτικά να συναντήσει τη λεκτική μονάδα «]». Η τρίτη είναι να κάνει διαδοχικά κλήση των συναρτήσεων expression(), REL_OP() και ξανά της expression().

(26) expression() : Ελέγχει αν η δομή των αριθμητικών εκφράσεων είναι ορθή. Καλεί διαδοχικά την optionalsign() , term() και κάθε φορά που ο τύπος της επόμενης λεκτικής μονάδας είναι add operator καλεί διαδοχικά την ADD_OP() και την term().

(27) term(): Καλεί την factor() και κάθε φορά ο τύπος της επόμενης λεκτικής μονάδας είναι mul operator (το ελέγχει καλώντας την MUL_OP()) ξανακαλεί την factor().

(28) factor() : Ελέγχει 3 περιπτώσεις δομών. Η πρώτη ελέγχει αν ο τύπος της λεκτικής μονάδας είναι αριθμός και αν είναι καλεί την INTEGER().«]». Η δεύτερη είναι αφού συναντήσει τη λεκτική μονάδα «(» κάνει κλήση τής expression() και έπειτα όταν επιστρέψει από τη expression() πρέπει υποχρεωτικά να συναντήσει τη λεκτική μονάδα «)».Η τρίτη είναι να κάνει διαδοχικά κλήση των συναρτήσεων ID() , idtail() .

(29) idtail() : Ελέγχει αν ο τύπος της λεκτικής μονάδας είναι add operator. Αν είναι καλεί την ADD_OP() αλλιώς επιστρέφει το κενό.

(30) REL_OP() : Ελέγχει αν η επόμενη λεκτική μονάδα είναι μεσα στη λίστα oper = ["=","<=",">=",">","<","<>"] και αν είναι επιστρέφει την λεκτική μονάδα.

(31) ADD_OP() : Ελέγχει αν ο τύπος της επόμενης λεκτικής μονάδας είναι add operator(“+” / ”-“) και αν είναι επιστρέφει την λεκτική μονάδα.

(32) MUL_OP() :Ελέγχει αν ο τύπος της επόμενης λεκτικής μονάδας είναι mul operator(“*” / “/”) και αν είναι επιστρέφει την λεκτική μονάδα.

(33) INTEGER() : Ελέγχει αν η επόμενη λεκτική μονάδα είναι integer και την επιστρέφει.

(34) ID () : Ελέγχει αν η επόμενη λεκτική μονάδα ακολουθεί τους κανόνες για ένα id σύμφωνα με τη γραμματική.

2η φάση : Ενδιάμεσος κώδικας

Ο Ενδιάμεσος κώδικας απαρτίζεται από αριθμημένες τετράδες. Εκτελούνται με βάση τον αριθμό τους (σε αύξουσα σειρά) εκτός εάν η τελευταία τετράδα που εκτελέστηκε παραπέμψει σε κάτι άλλο.

Συναρτήσεις ενδιάμεσου :

(1) nextQuad() : Επιστρέφει τον αριθμό της επόμενης τετράδας που πρόκειται να δημιουργηθεί, αυξάνοντας την γραμμή του αρχείου κατά ένα.

(2) genQ(op,x,y,z): Δημιουργεί μια τετράδα (op, x, y, z) και την προσθέτει στο τέλος της λίστας των τετράδων.

(3) newtemp(): Δημιουργεί και επιστρέφει μια καινούρια προσωρινή μεταβλητή (T_i). Το i εκφράζει τον αριθμό των προσωρινών μεταβλητών.

(4) emptylist(): Δημιουργεί μια κενή λίστα στην οποία θα προστεθούν ετικέτες(ετικέτα=αριθμός τετράδας) τετράδων.

(5) createlist(qLine): Δημιουργεί μια λίστα ετικετών τετράδων όπου μέσα περιλαμβάνει μόνο το χ πεδίο τους.

(6) mergelist(list1,list2): Δημιουργεί μία λίστα ετικετών τετράδων από τη συνένωση των λιστών list1 , list2.

(7) backpatch(list,ql) :Το πρώτο όρισμα είναι μια λίστα από αριθμημένες τετράδες ,τις οποίες όμως το πεδίο z δεν έχει ακόμα συμπληρωθεί. Η δουλειά της backpatch είναι να επεξεργάζεται μια προς μια τις τετράδες αυτές ώστε να συμπληρώνει το πεδίο z.

Προσθήκες στον συντακτικό αναλυτή :

(1)program():Δημιουργείται τετράδα για των τερματισμό του προγράμματος, αφότου γυρίσει από την block()

(2)block():Δημιουργείται τετράδα που δείχνει την αρχή του κυρίως προγράμματος.

(3)subprogram():Δημιουργείται τετράδα που δείχνει την αρχή ενός block υποπρογράμματος (function ή procedure),και όταν κλείνει το τέλος του .

(4)statement(): Δημιουργείται τετράδα στην περίπτωση που υπάρχει εκχώρηση.

(5) ifStat(): Αποθηκέυουμε την κλήση της condition() σε μια μεταβλητή,οπου δίνει 2 λίστες.Η μια είναι αληθής και η άλλη όχι.Η λίστα CTrue δείχνει την επόμενη τετράδα.Μετά δημιουργείται μια τετράδα jump και μια λίστα.

Συμπληρώνεται η δεύτερη λίστα και μετα το κομμάτι της else η επόμενη τετράδα είναι αυτή που παίρνουμε όταν βγαίνουμε από την if.

(6)whileStat():Αποθηκέυουμε την κλήση της condition() σε μια μεταβλητή,οπου δίνει 2 λίστες.Η μια είναι True και η άλλη False. Πριν καλέσουμε την statements() δημιουργούμε μια τετράδα .Η λίστα True δείχνει στην αρχή του block και η λίστα false στην επόμενη τετράδα.

(7)switchcaseStat(): Αποθηκέυουμε την κλήση της condition() σε μια μεταβλητή,οπου δίνει 2 λίστες.Η μια είναι True και η άλλη False. Πριν καλέσουμε την statements() δημιουργούμε μια τετράδα .Η λίστα True δείχνει στην αρχή του block. Μετά δημιουργείται μια τετράδα jump και μια λίστα.

Με την εμφάνιση του “default” χρησιμοποιούμε την exitlist ώστε να μπορούμε να γυρίσουμε στην αρχή της switch.

(8)forcaseStat():Αποθηκέυουμε σε μια μεταβλητη p1Quad την 1η τετράδα.Μετα αποθηκεύουμε την κλήση της condition() σε μια μεταβλητή,οπου δίνει 2 λίστες.Η μια είναι True και η άλλη False. Πριν καλέσουμε την statements() δινεται στην λίστα true η επομενη τετράδα . Δημιουργούμε μια τετράδα jump για να βγαίνει εκτος της for.

Με την εμφάνιση του “default” χρησιμοποιούμε την exitlist ώστε να μπορούμε να γυρίσουμε στην αρχή της switch.

(9)incaseStat():Δημιουργούμε προσωρινή μεταβλητή και αποθηκεύετε.Καλείται η condition και δημιουργούνται 2 λίστες True και False.Πριν την statements,γεμίζει η λίστα True με την επόμενη τετράδα και αλλάζει η προσωρινή μεταβλητή σε 0 .Μετά μπαίνει η επόμενη τετράδα στην False.Στο τέλος η προσωρινή επιστρέφει στην τιμη 0.

(10)callStat():Δημιουργούμε μια προσωρινή μεταβλητή και δημιουργούμε μια τετράδα για την παράμετρο και μία για την call.

(11)returnStat():Αποθηκέυουμε την κλήση της expression() σε μια μεταβλητή. Μετά δημιουργούμε μια νέα τετράδα δίνοντας αυτή την μεταβλητή.

(12)inputStat():Δημιουργούμε μια τετράδα για να δώσουμε το inp.

(13)printStat():Αποθηκεύουμε την κλήση της expression() σε μια μεταβλητή. Μετά δημιουργούμε μια νέα τετράδα δίνοντας αυτή την μεταβλητή.

(14)condition():Αποθηκεύουμε την κλήση της boolterm() σε έναν πίνακα BOOL και αποθηκεύουμε σε δυο προσωρινές Boolean μεταβλητές(CTrue = BOOL[0],CFalse = BOOL[1]). Καλεί την backpatch ώστε να συμπληρώσει το πεδίο z της επόμενης τετράδας για την CFalse. Αποθηκεύουμε την κλήση της boolterm() σε έναν πίνακα BOOL2 και αποθηκεύουμε σε δυο προσωρινές Boolean μεταβλητές(BOOL2True = BOOL2[0], BOOL2False = BOOL2[1]).Τέλος, συγχωνέυουμε τις λίστες με την mergelist(CTrue,BOOL2True).

(15)boolterm():():Αποθηκεύουμε την κλήση της boolfactor() σε έναν πίνακα B και αποθηκεύουμε σε δυο προσωρινές Boolean μεταβλητές (BOOLTrue = B[0],BOOLFalse = B[1]). Καλεί την backpatch ώστε να συμπληρώσει το πεδίο z της επόμενης τετράδας για BOOLTrue. Αποθηκεύουμε την κλήση της boolfactor() σε έναν πίνακα B2 και αποθηκεύουμε σε δυο προσωρινές Boolean μεταβλητές(B2True = B2[0], B2False = B2[1]). Τέλος, συγχωνέυουμε τις λίστες με την mergelist(BOOLFalse,B2False).

(16)boolfactor():Δημιουργεί 2 νέες τετράδες, σε περίπτωση που γίνεται διαδοχικά κλήση των συναρτήσεων expression(), REL_OP() και ξανά της expression().Μια με τις τιμές που επιστρέφει η expression() και μια για υποδειξη μεταπήδησης(jump).

(17)expression():Σε περίπτωση που εχουμε addoperator,δημιουργεί 1 νέα τετράδα με τις προσωρινές μεταβλητές T1,T2 (T1 = term(), T2 = term()). Ενώ, στην περίπτωση που έχουμε optional sign, δημιουργεί μία νέα τετράδα με το σύμβολο και την προσωρινή μεταβλητή.

(18)term():Δημιουργεί 1 νέα τετράδα με τις προσωρινές μεταβλητές F1,F2 (F1 = factor(),F2 = factor())

(19)idtail():Δημιουργεί 2 νέες τετράδες. Μια για την επιστροφή προσωρινής μεταβλητής και μια για τη κλήση της συνάρτησης με το όνομα της.

Τέλος, για τον έλεγχο αυτής της φάσης εκτυπώνουμε τα αποτελέσματα σε αρχείο. Υπάρχουν 2 περιπτώσεις, είτε το cimple πρόγραμμα έχει συνάρτηση ή διαδικασία και δημιουργούμε αρχείο με κατάληξη ".int" με την χρήση της intFile(infilename), είτε δεν έχει οπότε το μετατρέπουμε και δημιουργούμε αρχειο c με χρήση των συναρτήσεων newCFile(name) και makeCcode().

3η φάση : Πίνακας Συμβόλων, Σημασιολογική Ανάλυση και Τελικός Κώδικας

Πίνακας Συμβόλων:

Δημιουργείται για κάθε συνάρτηση που καλείται. Αποτελείται από τις κλάσεις: Scope,Entity,Argument, Function, varEn, Const, Parameter, TempVariable. Κάθε scope έχει λίστα από entities,τα οποία είναι μεταβλητή, συνάρτηση, σταθερά, παράμετρος και προσωρινή μεταβλητή.

Κάθε Entity έχει όνομα και τύπο. Οι συναρτήσεις έχουν όνομα, offset, startQ, argumementslist και framelength.

Οι σταθερές έχουν όνομα και τύπο. Οι παράμετροι έχουν όνομα, τρόπο περάσματος και offset.Τέλος, οι προσωρινές μεταβλητές έχουν όνομα και offset.

Σημασιολογική Ανάλυση:

Ελέγχουμε αν κάθε συνάρτηση ή μεταβλητή που έχει δηλωθεί ήδη μια φορά να μην ξαναδηλωθεί στο ίδιο ακριβώς βάθος φωλιάσματος(nestLevel) . Κάθε συνάρτηση ή μεταβλητή που έχει δηλωθεί ήδη μια φορά να μην ξαναδηλωθεί με τον ίδιο τρόπο που χρησιμοποιήθηκε πριν(var ή function).Επίσης, ελέγχουμε ότι οι συναρτήσεις χρησιμοποιούν παραμέτρους οι οποίες έχουν δηλωθεί και ακολουθούν την ορθή σειρά.

Τελικός κώδικας:

Για την παραγωγή του τελικού κώδικα θα αντιστοιχούμε κάθε εντολή του ενδιάμεσου κώδικα στον τελικό. Στόχος είναι η παραγωγή κώδικα για τον επεξεργαστή MIPS.

Βοηθητικές συναρτήσεις:

(1)gnvlcode: μεταφέρει στον $t0 την διεύθυνση μιας μη τοπικής μεταβλητής. Απο τις συναρτήσεις του πίνακα

συμβόλων παίρνουμε βρίσκουμε πόσα επίπεδα πάνω είναι η μη τοπική μεταβλητή και την εντοπίζει.

(2) loadvr(v, r):είναι υπέυθυνη για την μεταφορά δεδομένων στον καταχωρητή r .Εξετάζουμε τις περιπτώσεις το v να είναι σταθερά, καθολική μεταβλητή, τοπική μεταβλητή ,τυπική παράμετρος που περνάει με τιμή ,προσωρινή μεταβλητή, τυπική παράμετρος που περνά με αναφορά. Επίσης, αν v έχει δηλωθεί σε κάποιο πρόγονο και είναι τοπική μεταβλητή ,ή τυπική που περνάει με τιμή και αν η v έχει δηλωθεί σε κάποιο πρόγονο και είναι τοπική ή τυπική που περνάει με τιμή.

(3) storerv(r, v): είναι υπέυθυνη για την μεταφορά δεδομένων από τον καταχωρητή r στην μνήμη(δηλαδή την μεταβλητή v). Εξετάζουμε τις περιπτώσεις το v να είναι καθολική μεταβλητή, τοπική μεταβλητή ή τυπική παράμετρος που περνάει με τιμή και βάθος ίσο με το τρέχον ή προσωρινή, τυπική παράμετρος που περνάει με αναφορά και βάθος ίσο με το τρέχον, τοπική μεταβλητή ή τυπική παράμετρος που περνάει με τιμή και βάθος μικρότερο από το τρέχον .

(4) branch(x): σε αυτή την συνάρτηση αντιστοιχούμε τα σύμβολα συγκίσεων με τις αντίστοιχες εντολές της assembly(bne,beq…)

(5) arithOper(x): σε αυτή την συνάρτηση αντιστοιχούμε τα σύμβολα πράξεων με τις αντίστοιχες εντολές της assembly(add,sub…)

(6) makeAssembly(asname):Λαμβάνοντας εντολές από τον ενδιάμεσο κώδικα μέσω της λίστας quadsList ,δημιουργούμε το

αρχείο με τον κώδικα assembly με κατάληξη “.asm”, καλώντας την mips_code_file(qu, name);

(7) mips_code_file(qu, name): Για κάθε εντολή του ενδιάμεσου συντάσσει και γράφει αντίστοιχα στο αρχείο.