**2η φάση: Παραγωγή ενδιάμεσου κώδικα, Πίνακας Συμβόλων και Σημασιολογική Ανάλυση**

Παραγωγή ενδιάμεσου κώδικα:

Για την παραγωγή του ενδιάμεσου κώδικα, χρησιμοποιήσαμε τις βοηθητικές συναρτήσεις και μεταβλητές που ακολουθούν:

*1α. Χρήση μεταβλητών*

* **quad\_program\_list :** Δομή λεξικού, εδώ διατηρούμε τις παραγόμενες τετράδες του προγράμματος.
* **total\_quads** **:** Διατηρεί το συνολικό πλήθος των τετράδων , λειτουργεί σαν μετρητής.
* **temp\_value :** Βοηθητική μεταβλητή (λειτουργεί επίσης σαν μετρητής), διατηρεί το πλήθος των προσωρινών μεταβλητών (*Τ\_1, Τ\_2, ...*).
* **programName :** Εδώ αποθηκεύεται το όνομα του προγράμματος.
* **findLoop :** Λίστα που υποδεικνύει την πρώτη τετράδα κάθε loop statement (ώστε να επιστρέφει το πρόγραμμα μετά από τον έλεγχο της κατάλληλης συνθήκης).
* **exitLoop() :** Λίστα που υποδεικνύει την ετικέτα κάθε exit statement. Κάθε φορά που ένα loop statement τερματίζει η τελευταία θέση της λίστας διαγράφεται και γίνεται backpatch με την επόμενη τετράδα.

*1β. Χρήση συναρτήσεων*

* **next\_quad() :** Επιστρέφει τον αριθμό της ετικέτας της επομένης τετράδας που θα παραχθεί.
* **gen\_quad() :** Δημιουργεί την επόμενη τετράδα του προγράμματος.
* **newTemp() :** Δημιουργεί την προσωρινή μεταβλητή (Τ\_1, Τ\_2, ...).
* **emptylist() :** Επιστρέφει μία καινούρια κενή λίστα.
* **makelist() :** Δημιουργεί μία καινούρια λίστα που περιέχει μόνο μία ετικέτα, αυτή που δίνεται ως όρισμα στη συνάρτηση.
* **merge() :** Συνενώνει τις δύο συναρτήσεις που δίνονται ως όρισμα στη συνάρτηση. Πιο συγκεκριμένα, προσθέτει στην πρώτη λίστα τη δεύτερη.
* **backpatch() :** Αυτή η συνάρτηση αναθέτει τιμές στα τελούμενα του προγράμματος που δεν έχουν ακόμα λάβει κάποια τιμή.Το πρώτο όρισμα συγκεκριμένα είναι η λίστα στην οποία θέλουμε να συμπληρώσουμε το τελευταίο τελούμενο, και το δεύτερο όρισμα είναι η τιμή του τελούμενου που θέλουμε να δώσουμε. Η λειτουργία είναι η εξής: ανατρέχοντας τις ετικέτες του προγράμματος μία προς μία, αν η τρέχουσα τιμή υπάρχει στη λίστα-όρισμα της συνάρτησης, τότε συμπληρώνουμε την ετικέτα αυτή με το δεύτερο όρισμα της συνάρτησης.

*Λειτουργία ενδιάμεσου κώδικα:*

Για να εκτελεστεί ορθά ο μεταγλωττιστής μας, τροποποιήσαμε τις κατάλληλες συναρτήσεις της Starlet ώστε να μπορέσει να παραχθεί ο ενδιάμεσος κώδικας. Πιο συγκεκριμένα οι συναρτήσεις αυτές είναι οι ακόλουθες:

* **block() :** Με τη χρήση της gen\_quad(), δημιουργούμε τις τετράδες begin\_block και end\_block, οριοθετώντας έτσι αντίστοιχα την αρχή και το τέλος του block. Επιπλέον , εφόσον το τρέχον block αντιστοιχεί σε αυτό του κύριου προγράμματος, δημιουργούμε και την τετράδα halt, που σηματοδοτεί το τέλος του προγράμματος.
* **assignment\_stat() :** Εδώ λαμβάνεται η τιμή της έκφρασης (που επιστρέφει η expression) και στη συνέχεια με την gen\_quad() δημιουργείται η τετράδα που δείχνει την ανάθεση αυτή.
* **if\_stat() :** Αφού λάβουμε τις δύο λίστες που επιστρέφει η condition, τις αναθέτουμε ως ορίσματα στην backpatch έτσι ώστε να οδηγήσουμε το πρόγραμμα στην κατάλληλη τετράδα ανάλογα με το αν αληθεύει ή οχι η συνθήκη ελέγχου.Στην περίπτωση που αληθεύει, το πρόγραμμα συνεχίζει στην επόμενη τετράδα, ενώ στην αντίθετη περίπτωση δημιουργεί την εντολή jump για έξοδο μετά την εκτέλεση και με την backpatch οδηγεί το πρόγραμμα στην ετικέτα του elsepart.
* **while\_stat() :** Η while\_quad διατηρεί την ετικέτα της τετράδας στην οποία πραγματοποιείται ο έλεγχος της συνθήκης. Εάν η συνθήκη αληθεύει, εκτελείται η συνάρτηση statements και ξαναελέγχεται η συνθήκη, διαφορετικά ο έλεγχος μεταβαίνει εκτός while block. (με κατάλληλα backpatch)
* **do\_while\_stat() :** Η μεταβλητή do\_while\_quad έχει την ίδια λειτουργία με τη μεταβλητή while\_quad της while\_stat (δηλαδή οδηγεί στην αρχή του loop για να γίνει ο έλεγχος της συνθήκης) και μετά από τον έλεγχο η backpatch επιστρέφει το πρόγραμμα στην ετικέτα της do\_while\_quad αν η συνθήκη είναι λανθασμένη ή συνεχίζει στην επόμενη τετράδα αν αληθεύει.
* **loop\_stat() :** H προσωρινή μεταβλητή t, αρχικοποιείται στο 0 μέχρι να βρούμε το exit statement που αντιστοιχεί στο loop που βρισκόμαστε. Η πρώτη τετράδα του loop προστίθεται στη λίστα findLoop και μετά την εκτέλεση του endloop, βρεθεί το αντίστοιχο exit statement, επιστρέφουμε τον έλεγχο στην αρχή του τελευταίου loop, διαγράφωντας από τις αντίστοιχες λίστες τα πιο πρόσφατα loop & exit statements.
* **exit\_stat():** Αν στην κλήση της συνάρτησης βρεθεί exit statementπου αντιστοιχεί στο καλόν loop statement, δημιουργείται η τετράδα jump που οδηγεί σε έξοδο από το loop.
* **forcase\_stat() :** Η μεταβλητή forcase\_quad δείχνει στην πρώτη τετράδα του forcase μπλοκ και η λίστα exitList χρησιμοποιείται για να οδηγήσει την ροή του προγράμματος εκτός block μετά την εκτέλεση των statements της πρώτης αληθούς συνθήκης. Για κάθε συνθήκη, αν αληθεύει εκτελούνται τα statements της αλλιώς ελέγχουμε την επόμενη διαθέσιμη συνθήκη.Με βάση τα παραπάνω, μέσω της backpatch συμπληρώνουμε κατάλληλα τις λίστες που προέρχονται από την condition και εάν δεν αληθεύει καμία συνθήκη, εκτελείται το default statement. Σε αυτή την περίπτωση, επιστρέφουμε στην τετράδα με ετικέτα forcase\_quad και επανεξεταζουμε τις συνθήκες.
* **incase\_stat() :** Σε αυτή τη συνάρτηση δημιουργώ μία προσωρινή μεταβλητή t που αρχικοποιείται με την τιμή 0. Έπειτα, μετά από κάθε condition που αληθεύει αλλάζω την τιμή του t σε 1 και κάνω backpatch με την cond\_true ώστε να εκτελεστούν τα statements. Στην περίπτωση που το condition είναι ψευδές, κάνω backpatch με την cond\_false για να εκτελεστεί το επόμενο when. Στο τέλος, μετά τη δεσμευμένη λέξη endincase, γίνεται έλεγχος για την τιμή του t. Αν η τιμή είναι ίση με 1, σημαινει ότι τουλάχιστον μία when είναι αληθής και ο έλεχος μεταβαίνει στην αρχή της incase, διαφορετικά γίνεται έξοδος από τη συνάρτηση.
* **return\_stat() :** Επιστρέφουμε την τιμή της expression σε μία μεταβλητή Ε και στη συνέχεια με την gen\_quad δημιουργούμε την τετράδα που επιστρέφει την τιμή της μεταβλητής Ε (αντιστοιχεί στη δεσμευμένη λέξη “retv”).
* **print\_stat() :** Επιστρέφουμε την τιμή της expression σε μία μεταβλητή Ε και στη συνέχεια με την gen\_quad δημιουργούμε την τετράδα που τυπώνει την τιμή της μεταβλητής Ε (αντιστοιχεί στη δεσμευμένη λέξη “out”).
* **input\_stat() :** Εδώ με την gen\_quad δημιουργούμε την τετράδα με τη δεσμευμένη λέξη “inp” και την τιμή που δώθηκε από το χρήστη στην επιθυμητή μεταβλητή.
* **actualparitem() :** Επιστρέφουμε την τιμή της expression στη μεταβλητή actualparitem\_result και δημιουργούμε την επόμενη τετράδα που υποδεικνύει τη δημιουργία παραμέτρου με την τιμή του actualparitem\_result και τον τύπο της παραμέτρου (“CV”, “REF” ή “CP” για in, inout και inandout αντίστοιχα).
* **condition() :** Η συνάρτηση αυτή επιστρέφει το αποτέλεσμα της λογικής παράστασης or (true, false) που δίνεται μέσω της αποτίμησης των συναρτήσεων boolterm που περιέχονται σε αυτή. Το αποτέλεσμα αποθηκεύεται σε δύο λίστες (Βtrue, Βfalse) οι οποίες περιέχουν τις τετράδες που είναι αληθείς και ψευδείς (η κάθε boolterm αντίστοιχα περιέχει δύο δικές της λίστες, τις Q1true, Q1false). Η λογική είναι η εξής: όταν αποτιμηθεί η πρώτη boolterm, η αληθής λίστα της εξισώνεται με την αληθή λίστα της condition και αντίστοιχα η λανθασμένη λίστα της με τη λανθασμένη της condition. Στη συνέχεια μετά από κάθε or και πριν από κάθε επόμενο boolterm , γίνεται συμπλήρωση όσων τετράδων μπορούν να συμπληρωθούν μέσα στη συνάρτηση (δηλαδή οι λανθασμένες τετράδες) και μετά την εκτέλεση του επόμενου boolterm, συγχωνεύουμε τις δύο αληθείς λίστες και εξισώνουμε την τελική λανθασμένη λίστα με αυτή του τελευταίου boolterm.
* **boolterm() :** Η συνάρτηση αυτή επιστρέφει το αποτέλεσμα της λογικής παράστασης and και ακολουθεί την ίδια εκτέλεση με την condition, με τη διαφορά ότι εδώ οι τετράδες που μπορούν να συμπληρωθούν είναι οι αληθεις, και μετά την εκτέλεση του επόμενου boolfactor, συγχωνεύουμε τις δύο λανθασμένες λίστες και εξισώνουμε την τελική αληθή λίστα με αυτή του τελευταίου boolfactor.
* **boolfactor() :** Εδώ καλείται αναδρομικά η condition.Έπειτα, η αληθής λίστα της condition εξισώνεται με την αντίστοιχη της boolfactor (το ανάλογο συμβαίνει με τις λανθασμένες λίστες). Στην περίπτωση που συναντήσουμε τη δεσμευμένη λέξη “not” συμβαίνει το ανάποδο (η αληθής λίστα της condition εξισώνεται με τη λανθασμένη της boolfactor κ.ο.κ.). Αν έχουμε αριθμητική παράσταση επιστρέφουμε την τιμή των expression σε δύο μεταβλητές (Ε1 και Ε2 ), θέτουμε την αληθή λίστα της συνάρτησης ίση με την τετράδα που εκτελεί την πράξη ανάμεσα στις Ε1 και Ε2 και την ψευδή λίστα ίση με την παραγωγή της τετράδας jump.
* **expression() :** Αυτή η συνάρτηση δημιουργεί και επιστρέφει την προσωρινή μεταβλητή στην οποία αποθηκεύεται το αποτέλεσμα της αριθμητικής πράξης (με κλήση της newTemp) και η παράγει την τετράδα που μας δείχνει την πράξη αυτή.
* **term() :** Με την ίδια λογική εδώ δημιουργούνται και επιστρέφονται αναδρομικά οι πράγοντες που αποτιμούνται στη συνάρτηση expression.
* **factor() :** Στη μεταβλητή factor\_result της συνάρτησης επιστρέφεται η τιμή της σταθεράς, της έκφρασης ή του id που αποτελεί τους όρους που της συνάρτησης term.