ΒΑΣΙΚΕΣ ΕΝΝΟΙΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΥΠΟΛΟΓΙΣΤΩΝ

Εισαγωγή

Όπως για όλες τις επιστήμες, έτσι και για την επιστήμη της Πληροφορικής, ο τελικός στόχος της είναι η επίλυση προβλημάτων. Λύνονται όμως όλα τα προβλήματα; Υπάρχουν προβλήματα επιλύσιμα για τα οποία έχει βρεθεί και διατυπωθεί η λύση τους, προβλήματα άλυτα για τα οποία έχει αποδειχθεί ότι δεν επιδέχονται λύση (π.χ. ο τετραγωνισμός του κύκλου) και προβλήματα ανοικτά για τα οποία δεν έχει μεν βρεθεί λύση αλλά ταυτόχρονα δεν έχει αποδειχθεί ότι δεν επιδέχονται λύση.
Ας αφήσουμε τα τελευταία δύο είδη προβλημάτων και ας μιλήσουμε για τα επιλύσιμα: Οι λύσεις όλων αυτών των προβλημάτων δεν προέρχονται πάντα από μια αυτοματοποιημένη διαδικασία ή από κάποιο τύπο δηλ. η λύση τους δεν είναι τυποποιημένη ή δεν μπορεί να τυποποιηθεί. Για παράδειγμα η επίλυση μιας δευτεροβάθμιας εξίσωσης (ενός τριωνύμου) είναι μια τυποποιημένη διαδικασία δηλ. η διαδικασία αυτή με τους τύπους της διακρίνουσας Δ και των ριζών μας «λέει» α) αν υπάρχουν λύσεις β) όταν υπάρχουν, ποιες είναι αυτές οι λύσεις.
Το συμπέρασμα λοιπόν που προκύπτει από τα παραπάνω είναι το εξής: Ένα πρόβλημα που δεν μπορεί να το επιλύσει ο άνθρωπος δεν μπορεί να το επιλύσει ο υπολογιστής. Ο άνθρωπος επιλύει ένα πρόβλημα δηλ. σχεδιάζει τη λύση και ο υπολογιστής βρίσκει το αποτέλεσμα.
Γιατί να αναθέσουμε την επίλυση ενός προβλήματος στον υπολογιστή; Οι λόγοι είναι οι επόμενοι: α) Πολυπλοκότητα υπολογισμών β) Επαναληπτικότητα διαδικασιών γ) Η ταχύτητα εκτέλεσης πράξεων δ) Το μεγάλο πλήθος δεδομένων.

Ποιες λειτουργίες όμως μπορεί να κάνει ένας υπολογιστής από τη φύση του δηλ. από τη κατασκευή του ως υλικό;
Α) Πρόσθεση. Όλες οι άλλες πράξεις είναι παράγωγες της πρόσθεσης και ο υπολογιστής περιέχει προγράμματα που τον καθοδηγούν πως θα τις κάνει με βάση τη πρόσθεση.
Β) Σύγκριση δεδομένων.
Γ) Μεταφορά δεδομένων δηλ. ανάκτηση δεδομένων από μια θέση μνήμης και αποθήκευση σε κάποια άλλη θέση μνήμης.

Προκύπτουν από τα παραπάνω τα εξής ερωτήματα:
Α) Πρέπει λοιπόν ο άνθρωπος όταν επιλύει ένα πρόβλημα να το διατυπώνει έτσι ώστε να χρησιμοποιεί μόνο αυτές τις λειτουργίες που μπορεί να κάνει ο υπολογιστής;
Β) Αφού η γλώσσα του υπολογιστή αποτελείται από 0 και 1 πρέπει η διατύπωση των οδηγιών του ανθρώπου προς τον υπολογιστή να γίνει στη γλώσσα του υπολογιστή ή ο υπολογιστής να μάθει και να καταλαβαίνει την ανθρώπινη γλώσσα;
Τα ερωτήματα αυτά θα απαντηθούν σε επόμενη ενότητα.

Επίλυση προβλήματος από τον άνθρωπο

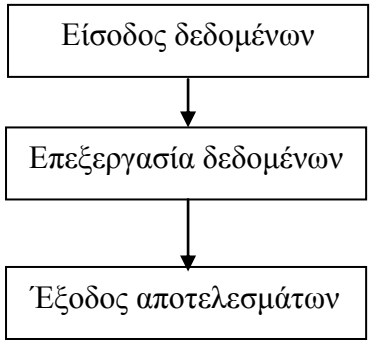
Σας δίνουν το εξής πρόβλημα: Να υπολογίσετε το τελικό βαθμό ενός μαθητή στα Μαθηματικά.
Πριν προχωρήσετε στην επίλυση ενός προβλήματος θα πρέπει να το διαβάσετε αρκετές φορές μέχρι να το κατανοήσετε. Η κατανόηση ενός προβλήματος αποτελεί συνάρτηση δύο παραγόντων: Της σωστής διατύπωσης από αυτόν που το θέτει και της σωστής ερμηνείας από αυτόν που καλείται να το λύσει.
Σωστή διατύπωση προβλήματος σημαίνει: Εύστοχη χρήση ορολογίας, σωστή σύνταξη φράσεων, σαφήνεια, πληρότητα. Πρέπει επίσης να είναι διατυπωμένο έτσι το πρόβλημα ώστε να μπορούν να προσδιορισθούν ακριβώς ποια είναι τα δεδομένα και ποια τα ζητούμενα. Η επεξεργασία των δεδομένων (υπολογισμοί, συγκρίσεις, έλεγχοι) οδηγεί στα ζητούμενα. Τον ακριβή προσδιορισμό των δεδομένων και των ζητούμενων πρέπει να τον κάνει αυτός που θα λύσει το πρόβλημα αλλά και αυτός που το διατυπώνει θα πρέπει να είναι σαφής και ακριβής. Την επεξεργασία επίσης θα πρέπει να τη κάνει αυτός που θα λύσει το πρόβλημα. Τα δεδομένα εκφράζουν που βρίσκομαι και τα ζητούμενα που πρέπει να πάω. Υπάρχει ένας δρόμος για να πάω

από τα δεδομένα στα ζητούμενα; Όχι, υπάρχουν γενικά πολλοί, έτσι υπάρχουν γενικά και πολλοί τρόποι επεξεργασίας (επίλυσης).
Ας επανέλθουμε στο πρόβλημα που σας τέθηκε: «Να υπολογίσετε το τελικό βαθμό ενός μαθητή στα Μαθηματικά». Το πρόβλημα είναι σωστά διατυπωμένο; Εννοεί να βρείτε το τελικό βαθμό ενός τριμήνου ή την ετήσια επίδοση (βαθμό) στο μάθημα; Όταν λέει τελικό βαθμό τι εννοεί; Μήπως το μέσο όρο των βαθμών των τριών τριμήνων και του γραπτού του Ιουνίου ή μήπως το πηλίκο του αθροίσματος των τριών βαθμών και του διπλασίου του γραπτού βαθμού με το 5; Ο μαθητής μήπως είναι της Γ Λυκείου οπότε ο προφορικός του μέσος όρος των δύο τετραμήνων αναπροσαρμόζεται σε σχέση με το γραπτό των πανελληνίων ώστε να μη διαφέρει πάνω από δύο μονάδες; Τέλος πάντων ποια είναι τα δεδομένα; Είναι μαθητής Γυμνασίου ή Λυκείου;
Η σωστή διατύπωση του παραπάνω προβλήματος είναι η εξής: «Να υπολογίσετε το τελικό βαθμό ενός μαθητή Γυμνασίου στα μαθηματικά με βάση τους βαθμούς των τριών τριμήνων και του γραπτού βαθμού. Ο τελικός βαθμός είναι ο μέσος όρος των τεσσάρων αυτών βαθμών στρογγυλοποιημένος προς το πλησιέστερο ακέραιο».
Πλέον η διατύπωση είναι ακριβής: Η φράση «μέσος όρος» είναι μια γνωστή ορολογία καθώς και η έννοια της στρογγυλοποίησης από τα μαθηματικά.

Συμπέρασμα: Όταν μας δίνουν ένα σωστά διατυπωμένο πρόβλημα το πρώτο πράγμα που αναζητάμε να προσδιορίσουμε είναι: Ποια είναι τα δεδομένα και ποια τα ζητούμενα του προβλήματος; Στη συνέχεια σχεδιάζουμε την επεξεργασία που από τα δεδομένα θα οδηγηθούμε στα ζητούμενα.

Αλγόριθμος

Για οποιοδήποτε πρόβλημα που σας καλούν να το λύσετε θα πρέπει: Να ζητήσετε τις συγκεκριμένες τιμές των δεδομένων (π.χ. οι τέσσερις βαθμοί), να επεξεργαστείτε αυτές τις τιμές κάνοντας διάφορες πράξεις και να εμφανίσετε τα αποτελέσματα. Αυτές οι τρεις ενέργειες αποτελούν το γενικό πλάνο επίλυσης ενός προβλήματος: α) Είσοδος δεδομένων β) Επεξεργασία δεδομένων γ) Έξοδος αποτελεσμάτων.



Αλγόριθμος είναι μια πεπερασμένη σειρά (δηλ. όχι άπειρου πλήθους) ενεργειών, αυστηρά καθορισμένων και εκτελέσιμων σε πεπερασμένο χρόνο, που επιλύουν ένα πρόβλημα.

Εξήγηση του ορισμού του αλγόριθμου

Α) «Πεπερασμένη σειρά ενεργειών» σημαίνει ότι οι ενέργειες (εντολές, βήματα) του αλγόριθμου θα είναι μετρήσιμου πλήθους, όχι άπειρου. Δεν μπορεί να είναι άπειρου πλήθους διότι ο αλγόριθμος δεν τελειώνει άρα δεν δίνει λύση στο πρόβλημα.

Β) «Αυστηρά καθορισμένων» σημαίνει ότι κάθε ενέργεια (εντολή) του αλγόριθμου πρέπει να καθορίζεται ώστε να μη χωρά καμία αμφιβολία για τον τρόπο εκτέλεσής της. Για παράδειγμα, ως γνωστό η διαίρεση με το μηδέν είναι μη επιτρεπτή (αδύνατη) πράξη. Αν μια ενέργεια είναι η πράξη α / β υπάρχει περίπτωση αν το β έχει τιμή 0 η πράξη να μην μπορεί να γίνει και το αντίστοιχο πρόγραμμα στον υπολογιστή θα τερματιστεί ανώμαλα και αναπάντεχα. Αν θέλαμε να υπάρχει η ενέργεια της διαίρεσης α / β θα έπρεπε η ενέργεια να έχει το εξής νόημα: «Αν το β ≠ 0 τότε κάνε τη πράξη α / β».

Γ) «Εκτελέσιμων» σημαίνει ότι κάθε ενέργεια (εντολή) του αλγόριθμου πρέπει να ανήκει στο ρεπερτόριο εντολών που μπορεί να εκτελέσει ο υπολογιστής. Για παράδειγμα δεν μπορεί μια ενέργεια να είναι κάτι σαν «βρες το Μέγιστο Κοινό Διαιρέτη των αριθμών 28 και 36» ή «βρες το μεγαλύτερο από τους αριθμούς 5 και 8»: Τέτοιες εντολές δεν μπορεί άμεσα να εκτελέσει ο υπολογιστής, πρέπει να φτιάξουμε πρόγραμμα για να λύνει καθένα από τα παραπάνω προβλήματα.
Δ) «Σε πεπερασμένο χρόνο» σημαίνει ότι ο αλγόριθμος πρέπει κάποια στιγμή χρονικά να τελειώνει, να μην εκτελείται έπ' άπειρο διότι αυτό σημαίνει ότι πάλι δε δίνει λύση. Για παράδειγμα, του ζητούμε να βρει τη λύση

της εξίσωσης $x^2 - 2 = 0$. Η λύση αυτής είναι ο αριθμός $\sqrt{2}$ δηλ. ο αριθμός 1,414213562373095048801688724209 7 κτλ. με άπειρα δεκαδικά ψηφία. Άρα ο αλγόριθμος αυτός δεν θα τελείωνε ποτέ! Θα μπορούσαμε να του πούμε όμως να μας βρει μέχρι 15 δεκαδικά ψηφία οπότε θα τελείωνε.

Από τα παραπάνω συμπεραίνουμε τα ακόλουθα:

Κριτήρια που πρέπει να ικανοποιεί ένας αλγόριθμος

1) Να περιλαμβάνει εντολές εισόδου τιμών δεδομένων
2) Να περιλαμβάνει εντολές για έξοδο αποτελεσμάτων προς το χρήστη ή σε ένα άλλο αλγόριθμο
3) Οι εντολές του να είναι αυστηρά καθορισμένες
4) Να έχει περατότητα δηλ. κάποια στιγμή να τελειώνει
5) Να είναι πραγματοποιήσιμος δηλ. κάθε εντολή του να μπορεί να εκτελεστεί.

Τρόποι αναπαράστασης αλγορίθμων

Η διατύπωση ενός αλγόριθμου γίνεται συνήθως με δύο από τους παρακάτω τρόπους:

Α) με διάγραμμα ροής που χρησιμοποιούμε γεωμετρικά σχήματα για να παραστήσουμε τις εντολές του (δες σελ. 243 βιβλίου)

Β) Με ψευδοκώδικα δηλ. κείμενο που έχει τη δομή μιας κανονικής γλώσσας προγραμματισμού υπολογιστών χωρίς όμως την αυστηρότητα των κανόνων σύνταξης της γλώσσας προγραμματισμού (για παραδείγματα ψευδοκώδικα, δες τις ασκήσεις παρακάτω).

Γλώσσες προγραμματισμού

Ένας αλγόριθμος επιλύει ένα πρόβλημα. Τον αλγόριθμο τον δημιουργεί ο προγραμματιστής. Στη συνέχεια, με βάση τον αλγόριθμο γράφει ένα πρόγραμμα σε μια γλώσσα προγραμματισμού.

Ποιος είναι ο ρόλος μιας γλώσσας προγραμματισμού;

Η γλώσσα του προγραμματιστή είναι η ανθρώπινη. Η γλώσσα του υπολογιστή είναι «προτάσεις» που αποτελούνται από τα δυαδικά ψηφία 0 και 1. Αυτή η γλώσσα λέγεται γλώσσα μηχανής**.** Ο προγραμματιστής πρέπει να δώσει οδηγίες επίλυσης του προβλήματος στον υπολογιστή δηλ. να γράψει ένα πρόγραμμα με εντολές προς αυτόν. Επειδή ο υπολογιστής δεν μπορεί να «μιλήσει» και να κατανοήσει την ανθρώπινη γλώσσα άρα θα πρέπει ο άνθρωπος να δώσει τις εντολές σε γλώσσα μηχανής. Ε! αυτό γινόταν αρχικά όταν εμφανίστηκαν οι υπολογιστές, οι προγραμματιστές έγραφαν προγράμματα σε γλώσσα μηχανής χρησιμοποιώντας τα σύμβολα 0 και 1. Αυτό όμως ήταν πολύ δύσκολο. Σημειώστε επίσης ότι κάθε μοντέλο υπολογιστή έχει τη δική του γλώσσα μηχανής. Αυτό σημαίνει ότι ένα πρόγραμμα γραμμένο για ένα τύπο υπολογιστή δε λειτουργεί σε άλλο τύπο. Για παράδειγμα, ένα πρόγραμμα που εκτελείται σε Pc δεν εκτελείται σε υπολογιστή της εταιρείας Apple (π.χ. Macintosh) Ως εξέλιξη της γλώσσας μηχανής εφευρέθηκε η συμβολική γλώσσα (Assembly) που ήταν σαν τη γλώσσα μηχανής με τη διαφορά πως κάποιες «προτάσεις» της αντικαταστάθηκαν από συμβολικά ονόματα όπως π.χ. με τη λέξη ADD που ήταν πιο ευκολομνημόνευτες σε σχέση με μια σειρά δυαδικών ψηφίων. Οι γλώσσες μηχανής και Assembly ονομάζονται γλώσσες χαμηλού επιπέδου διότι βρίσκονται πιο κοντά στη μηχανή παρά στον άνθρωπο. Αλλά και η συμβολική γλώσσα ήταν δύσκολη στη χρήση της και έτσι εφευρέθηκαν οι γλώσσες υψηλού επιπέδου που σαν δομή ήταν πιο κοντά στην ανθρώπινη γλώσσα παρά στη γλώσσα της μηχανής. Τελικά ο ρόλος μιας γλώσσας προγραμματισμού είναι ο ίδιος με το ρόλο του μεταφραστή ανάμεσα σε δύο ανθρώπους που ο ένας μιλάει μόνο ελληνικά και ο άλλος μόνο αγγλικά: Λες κάτι στα ελληνικά προς το μεταφραστή και αυτός το λέει στα αγγλικά στον άλλο.

Έτσι και ο προγραμματιστής: Γράφει ένα πρόγραμμα σε μια γλώσσα προγραμματισμού και αυτή το μεταφράζει σε γλώσσα μηχανής δηλ. στη γλώσσα του υπολογιστή. Πρέπει όμως ο μεταφραστής να καταλαβαίνει όλες τις λέξεις που του λένε στα ελληνικά ώστε να τις μεταφράσει στα αγγλικά, έτσι δεν είναι; Διαφορετικά δεν προχωρά στη μετάφραση. Για παράδειγμα, μεταφράστε τη φράση «Φέρε μου το ματσακόνι!» Έτσι και η γλώσσα προγραμματισμού. Πρέπει ο προγραμματιστής να γράφει προγράμματα χρησιμοποιώντας το λεξιλόγιό

της, τους κανόνες ορθογραφίας της και τους κανόνες σύνταξης των προτάσεών της διαφορετικά δεν μεταφράζει το πρόγραμμα σε γλώσσα μηχανής μέχρι ο προγραμματιστής να κάνει τις απαραίτητες διορθώσεις.
Παραδείγματα γλωσσών υψηλού επιπέδου είναι οι Basic, Pascal, Cobol, C που ονομάστηκαν γλώσσες τρίτης γενιάς και οι αντικειμενοστραφείς γλώσσες C++, Java, Visual Basic, Delphi κτλ.

Η έννοια της μεταβλητής στο προγραμματισμό

Είπαμε ότι ένας αλγόριθμος πρέπει να επιλύει ένα πρόβλημα κατά γενικό τρόπο και όχι για συγκεκριμένες τιμές δεδομένων. Στα Μαθηματικά, τη Φυσική κτλ. έχουμε διάφορους τύπους π.χ. Ε=β*υ/2 οπότε όταν μας δίνουν συγκεκριμένες τιμές για τα β και υ υπολογίζουμε το Ε. Χρησιμοποιούμε δηλ. σύμβολα Ε, β, υ τα οποία παριστάνουν δεδομένα και ζητούμενα. Αυτά τα σύμβολα ονομάζονται μεταβλητές διότι μπορούν να πάρουν διάφορες τιμές. Έτσι λοιπόν και στους αλγόριθμους τα δεδομένα και τα ζητούμενα τα παριστάνουμε με μεταβλητές. Για παράδειγμα τους τρεις βαθμούς τριμήνων σε κάποιο μάθημα ενός μαθητή μπορούμε να τους συμβολίσουμε π.χ. με β1, β2, β3, το γραπτό του Ιουνίου με Γρ και το ζητούμενο τελικό βαθμό με ΤΒ ή όπως αλλιώς θέλουμε. Αυτά τα σύμβολα είναι τα ονόματα των μεταβλητών που παριστάνουν τα δεδομένα και τα ζητούμενα. Επίσης σχεδόν πάντα σε ένα αλγόριθμο μέχρι να φτάσουμε από τα δεδομένα στα ζητούμενα πρέπει να υπολογίσουμε και κάποια ενδιάμεσα αποτελέσματα οπότε και αυτά τα παριστάνουμε με μεταβλητές. Για κάθε μεταβλητή ο υπολογιστής της αντιστοιχεί μια θέση στη μνήμη του ώστε σε αυτή να αποθηκεύεται η τιμή της μεταβλητής. Υπάρχουν διάφορες μορφές δεδομένων όπως αριθμοί, κείμενο κτλ. Ο υπολογιστής με άλλο τρόπο παριστάνει και αποθηκεύει τους ακέραιους, με άλλο τρόπο τους δεκαδικούς και με άλλο τρόπο το κείμενο γι' αυτό πρέπει να «πούμε» στον υπολογιστή και το τύπο μιας μεταβλητής ώστε για κάθε μία να καθορίσει πόσος θα είναι ο απαιτούμενος χώρος μνήμης που θα της αντιστοιχίσει.
Μια μεταβλητή λοιπόν έχει: α) Ένα όνομα β) Ένα τύπο δεδομένων (πραγματική, ακέραια, χαρακτήρες, λογική) γ) μια τιμή. ΠΡΟΣΟΧΗ! Αν μια μεταβλητή αλλάξει τιμή η προηγούμενη τιμή της χάνεται! Σε μια θέση μνήμης ο υπολογιστής περιέχει μια μόνο τιμή μιας μεταβλητής. Η καινούργια αντικαθιστά τη παλιά.

Χρήσιμοι ορισμοί και έννοιες

Ένα **πρόγραμμα** είναι η αναπαράσταση ενός αλγορίθμου γραμμένη σε μια γλώσσα κατανοητή από τον υπολογιστή. Ένα πρόγραμμα αποτελείται από **εντολές** που πρέπει να εκτελέσει ο υπολογιστής ώστε να παραχθεί το επιθυμητό αποτέλεσμα. Η εργασία σύνταξης των προγραμμάτων ονομάζεται **προγραμματισμός.**

Γλώσσα προγραμματισμού είναι μια τεχνητή γλώσσα που «καταλαβαίνει» ο υπολογιστής. Μια τέτοια γλώσσα πρέπει να τη γνωρίζει και ένας προγραμματιστής ώστε να γράφει προγράμματα που τελικά να «κατανοούνται» από τον υπολογιστή και να εκτελούνται από αυτόν. Μια γλώσσα προγραμματισμού χρησιμοποιείται από ένα προγραμματιστή για να επικοινωνήσει με τον υπολογιστή.

Τα χαρακτηριστικά των γλωσσών προγραμματισμού – όπως και μιας ανθρώπινης γλώσσας – είναι: **α)** Το αλφάβητό της **β)** Το λεξιλόγιό της **γ)** Το συντακτικό της.
Το **αλφάβητό** της είναι το σύνολο των χαρακτήρων που χρησιμοποιούνται από τη γλώσσα.
Το **λεξιλόγιό** της είναι το σύνολο των λέξεων που αναγνωρίζει η γλώσσα και έχουν μια μοναδική σημασία.
Το **συντακτικό** της είναι το σύνολο των κανόνων που πρέπει να τηρούμε ώστε να συνδέσουμε λέξεις σε προτάσεις.

Συντακτικά λάθη σε ένα πρόγραμμα που έχουμε γράψει λέγονται τα λάθη που οφείλονται σε άστοχη χρήση του αλφάβητου, του λεξιλογίου και του συντακτικού. Τα συντακτικά λάθη εντοπίζονται από τα **μεταφραστικά** προγράμματα.
Λέμε ότι ένα πρόγραμμα έχει **λογικά λάθη** όταν το αποτέλεσμα που προκύπτει δεν είναι το αναμενόμενο, παρότι το πρόγραμμα δεν έχει συντακτικά λάθη. Τα λογικά λάθη δεν εντοπίζονται από τα μεταφραστικά προγράμματα. Για παράδειγμα, στην εύρεση του μέσου όρου των τριμήνων για ένα μάθημα, λογικό λάθος είναι

να πολλαπλασιάζονται οι τρεις βαθμοί αντί να προστίθενται όπως είναι το σωστό. Τα λογικά λάθη τα εντοπίζει και τα διορθώνει ο προγραμματιστής.

Όπως έχουμε πει, η φυσική γλώσσα του υπολογιστή είναι η γλώσσα μηχανής με αλφάβητο το 0 και 1. Μια γλώσσα προγραμματισμού όπως π.χ. η Logo δεν είναι άμεσα κατανοητή από τον υπολογιστή. Όταν ένας προγραμματιστής γράψει ένα πρόγραμμα σε μια γλώσσα προγραμματισμού αυτό θα πρέπει να μεταφραστεί σε γλώσσα μηχανής ώστε να μπορεί να εκτελεστεί από τον υπολογιστή. Υπάρχουν δύο είδη τέτοιων **μεταφραστικών** προγραμμάτων:

α) Οι μεταγλωττιστές (compilers)
β) Οι διερμηνείς (interpreters)

Η διαφορά τους είναι ότι οι μεταγλωττιστές ελέγχουν όλο το πρόγραμμα για συντακτικά λάθη και αν δεν έχει το μετατρέπουν όλο σε μια σειρά από 0 και 1 δημιουργώντας ένα νέο αρχείο. Αν έχει συντακτικά λάθη τότε δεν κάνουν καμία μετατροπή αλλά εντοπίζουν αυτά τα λάθη και καλούν το προγραμματιστή να τα διορθώσει. Μετά τη διόρθωση υποβάλει ο προγραμματιστής το πρόγραμμα πάλι στη διαδικασία μεταγλώττισης.
Αντίθετα, οι διερμηνείς ελέγχουν συντακτικά μια εντολή κάθε φορά και αν δεν έχει λάθη τότε την εκτελούν και μετά ελέγχουν την επόμενη εντολή κ.ο.κ Η Logo χρησιμοποιεί διερμηνέα.