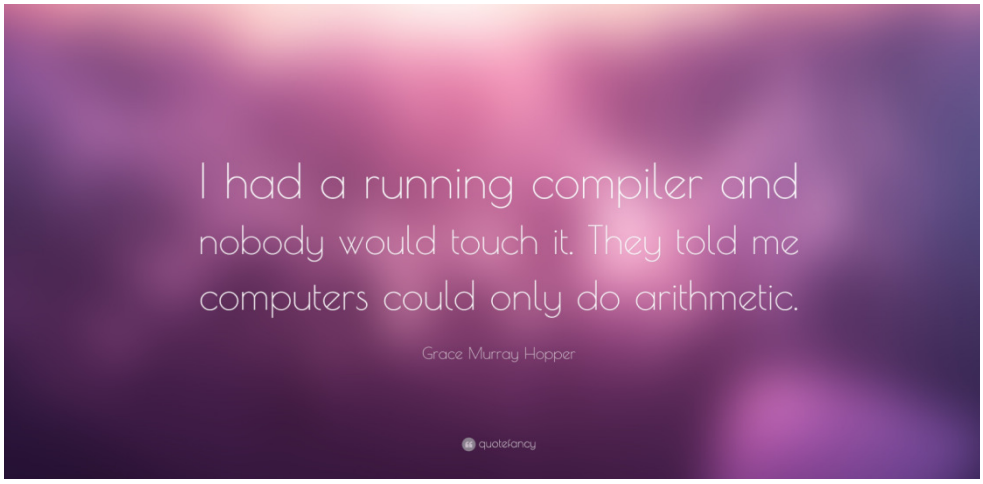
2019

Αλγόριθμος

ΚΕΦΑΛΑΙΟ 6

Algorithmos Computer Science Laboratory



[ΕΙΣΑΓΩΓΗ ΣΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ]

[Επιλεγμένα θέματα θεωρίας σύμφωνα με την ύλη 2018-2019
στην ενότητα Εισαγωγή στον Προγραμματισμό (Κεφάλαιο 6 Σχολικού βιβλίου).
Συγκεκριμένα αναλύονται: Φυσικές και Τεχνητές Γλώσσες, Μεθοδολογίες Προγραμματισμού & Προγραμματιστικά Περιβάλλοντα.]

Πίνακας περιεχομένων

Φυσικές και τεχνητές γλώσσες

1.Τι είναι γλώσσες προγραμματισμού και ποιος είναι ο σκοπός ανάπτυξης τους;

Οι γλώσσες προγραμματισμού αναπτύχθηκαν για να μπορεί ο προγραμματιστής να δίνει τις εντολές που πρέπει να εκτελέσει ο υπολογιστής. Χρησιμοποιούνται δηλαδή για την επικοινωνία του ανθρώπου και της μηχανής, όπως αντίστοιχα οι φυσικές γλώσσες χρησιμοποιούνται για την επικοινωνία μεταξύ των ανθρώπων. Οι γλώσσες προγραμματισμού, που είναι τεχνητές γλώσσες, ακολουθούν τις βασικές έννοιες και αρχές της Γλωσσολογίας, επιστήμη που μελετά τις φυσικές γλώσσες.

2. Από τι προσδιορίζεται μια γλώσσα; (Εξετάσεις 2017)

Μια γλώσσα, είτε πρόκειται για φυσική είτε για τεχνητή γλώσσα, προσδιορίζεται από:

- το αλφάβητο,
- το λεξιλόγιο,
- τη γραμματική και
- τη σημασιολογία.

3. α. Τι καλείται αλφάβητο μιας γλώσσας;
β. Από τι αποτελείται μια γλώσσα;
γ. Τι είναι το τυπικό μιας γλώσσας;
δ. Τι είναι το συντακτικό μιας γλώσσας; (Εξετάσεις 2018 & 2004)
ε. Τι είναι η σημασιολογία μιας γλώσσας

α. Το σύνολο των στοιχείων που χρησιμοποιεί μια γλώσσα.

β. Από το υποσύνολο των ακολουθιών των γραμμάτων του αλφαβήτου μιας γλώσσας που είναι αποδεκτά από αυτήν.

γ. Η γραμματική μιας γλώσσας αποτελείται από το τυπικό (ή τυπολογικό) και το συντακτικό. Το τυπικό είναι το σύνολο των κανόνων που ορίζει τις μορφές με τις οποίες μια λέξη της γλώσσας είναι αποδεκτή.

δ. Το σύνολο των κανόνων που καθορίζει τη νομιμότητα της διάταξης και της σύνδεσης των λέξεων για τη δημιουργία προτάσεων. Η γνώση του συντακτικού επιτρέπει τη δημιουργία σωστών προτάσεων.

ε. Το σύνολο των κανόνων που καθορίζει το νόημα των λέξεων και των εκφράσεων της γλώσσας. Ο δημιουργός της γλώσσας καθορίζει τη σημασιολογία της.

4. Τι είναι το αλφάβητο μιας γλώσσας;

Αλφάβητο μιας γλώσσας καλείται το σύνολο των στοιχείων που χρησιμοποιείται από τη γλώσσα. Για την ελληνική γλώσσα, για παράδειγμα, το αλφάβητο αποτελείται από όλα γράμματα, μικρά και κεφαλαία **(Α.. .Ω),** τα ψηφία **(0.. .9)** και όλα τα **σημεία στίξης**. Οι τεχνητές γλώσσες χρησιμοποιούν μόνο το αγγλικό αλφάβητο (A.. .Z), (a.. .z), ψηφία (0.. .9) και όλα τα σημεία στίξης.

5. Τι είναι το λεξιλόγιο μιας γλώσσας;

Το λεξιλόγιο αποτελείται από ένα υποσύνολο όλων των ακολουθιών που δημιουργούνται από τα στοιχεία του αλφαβήτου (τις λέξεις) που είναι δεκτές από τη γλώσσα. Για παράδειγμα, στην αγγλική γλώσσα η ακολουθία **print** είναι δεκτή, αφού αποτελεί λέξη, ενώ η ακολουθία **printf** δεν είναι δεκτή. Σε μια τεχνητή γλώσσα η ακολουθία **read** είναι δεκτή ως λέξη, ενώ η ακολουθία **reader** δεν είναι δεκτή.

6. Τι είναι η γραμματική μιας γλώσσας;

Η γραμματική αποτελείται από το τυπικό ή τυπολογικό και το συντακτικό.

- Τυπικό είναι το σύνολο των κανόνων που ορίζει τις μορφές με τις οποίες μια λέξη είναι αποδεκτή. Για παράδειγμα, στην αγγλική γλώσσα οι λέξεις **employ, employer** είναι δεκτές, ενώ η λέξη **emplyer** δεν είναι δεκτή.
- Συντακτικό είναι το σύνολο των κανόνων που καθορίζει τη νομιμότητα της διάταξης και της σύνθεσης των λέξεων της γλώσσας για τη δημιουργία προτάσεων στις φυσικές γλώσσες και εντολών στις τεχνητές γλώσσες. Για παράδειγμα, η πρόταση **όσο χ > 0 τότε** στη γλώσσα προγραμματισμού «ΓΛΩΣΣΑ» δεν είναι αποδεκτή.

7. Τι είναι η σημασιολογία μιας γλώσσας;

Η σημασιολογία είναι το σύνολο των κανόνων που καθορίζει το νόημα των λέξεων και κατ' επέκταση των εκφράσεων και προτάσεων που χρησιμοποιούνται σε μια γλώσσα. Στις γλώσσες προγραμματισμού, οι οποίες είναι τεχνητές γλώσσες, ο δημιουργός της γλώσσας αποφασίζει τη σημασιολογία των λέξεων της γλώσσας.

8. Σε τι διαφέρουν οι φυσικές από τις τεχνητές γλώσσες;

- **Φυσικές γλώσσες**

Οι φυσικές γλώσσες εξελίσσονται συνεχώς, νέες λέξεις δημιουργούνται, κανόνες γραμματικής και σύνταξης αλλάζουν με την πάροδο του χρόνου και αυτό γιατί η γλώσσα χρησιμοποιείται για την επικοινωνία μεταξύ ανθρώπων, που εξελίσσονται και αλλάζουν ανάλογα με τις εποχές και τον κοινωνικό περίγυρο.

- **Τεχνητές γλώσσες**

Αντίθετα οι τεχνητές γλώσσες χαρακτηρίζονται από στασιμότητα, αφού κατασκευάζονται για έναν συγκεκριμένο σκοπό. Κάθε αλλαγή γίνεται από τον δημιουργό τους για να διορθωθούν κάποιες αδυναμίες τους, για να μπορούν να καλύψουν μεγαλύτερο εύρος εφαρμογών και τέλος για να ακολουθήσουν τις νέες εξελίξεις. Οι γλώσσες προγραμματισμού αλλάζουν:

- σε επίπεδο διαλέκτου (για παράδειγμα GW-Basic και QuickBasic) ή
- σε επίπεδο επέκτασης (για παράδειγμα Basic και Visual Basic).

Τεχνικές σχεδίασης προγραμμάτων

9. Τι περιλαμβάνει η ιεραρχική σχεδίαση προγράμματος;

Η ιεραρχική σχεδίαση προγράμματος ή η διαδικασία σχεδίασης «από πάνω προς τα κάτω», όπως συχνά ονομάζεται, περιλαμβάνει τον καθορισμό των βασικών λειτουργιών ενός προγράμματος, σε ανώτερο επίπεδο, και στη συνέχεια τη διάσπαση των λειτουργιών αυτών σε όλο και μικρότερες λειτουργίες μέχρι το τελευταίο επίπεδο, όπου οι λειτουργίες είναι πια απλές και μπορούν να επιλυθούν εύκολα.

10.Ποιος είναι ο σκοπός της ιεραρχικής σχεδίασης ενός προγράμματος και πώς γίνεται η παράσταση της;

Ο σκοπός της ιεραρχικής σχεδίασης είναι η διάσπαση ενός προβλήματος σε μια σειρά από απλούστερα υποπροβλήματα, τα οποία είναι εύκολο να επιλυθούν οδηγώντας στην επίλυση του αρχικού προβλήματος. Η παράσταση της γίνεται με τη βοήθεια διαγραμματικών τεχνικών.

11. Τι είναι ο τμηματικός προγραμματισμός;

Η ιεραρχική σχεδίαση προγράμματος υλοποιείται με τον τμηματικό προγραμματισμό. Μετά την ανάλυση του προβλήματος σε αντίστοιχα υποπροβλήματα, κάθε υποπρόβλημα αποτελεί ανεξάρτητη ενότητα που γράφεται ξεχωριστά από τα υπόλοιπα τμήματα προγράμματος.

12. Ποια είναι τα πλεονεκτήματα του τμηματικού προγραμματισμού (τι προσφέρει); (Εξετάσεις 2003)

Ο τμηματικός προγραμματισμός διευκολύνει τη δημιουργία του προγράμματος, μειώνει τα λάθη και επιτρέπει την ευκολότερη παρακολούθηση, κατανόηση και συντήρηση του προγράμματος από τρίτους.

13. Τι είναι ο δομημένος προγραμματισμός;

Ο δομημένος προγραμματισμός δεν είναι απλώς ένα είδος προγραμματισμού, είναι μια **μεθοδολογία σύνταξης προγραμμάτων** που έχει σκοπό να βοηθήσει τον **προγραμματιστή** στην ανάπτυξη σύνθετων προγραμμάτων, να μειώσει τα λάθη, να εξασφαλίσει την εύκολη κατανόηση των προγραμμάτων και να διευκολύνει τις διορθώσεις και τις αλλαγές σε αυτά.

Σημείωση! Ο δομημένος προγραμματισμός ενθαρρύνει και βοηθάει την ανάλυση του προγράμματος σε επί μέρους τμήματα, ώστε σήμερα ο όρος δομημένος προγραμματισμός περιέχει τόσο την ιεραρχική σχεδίαση όσο και τον τμηματικό προγραμματισμό.

14. Ποιες είναι οι αρχές του δομημένου προγραμματισμού;

Ο δομημένος προγραμματισμός εμπεριέχει τόσο την ιεραρχική σχεδίαση όσο και τον τμηματικό προγραμματισμό και στηρίζεται σε δύο αρχές:

► Όλα τα προγράμματα μπορούν να γραφούν χρησιμοποιώντας μόνο τις **τρεις στοιχειώδεις λογικές δομές**, δηλαδή τη δομή ακολουθίας, τη δομή της επιλογής και τη δομή της επανάληψης. (Εξετάσεις 2003)

► Κάθε πρόγραμμα, όπως και κάθε ενότητα προγράμματος, έχει **μία είσοδο και μία έξοδο**.

15. Ποια είναι τα πλεονεκτήματα του δομημένου προγραμματισμού;

- Δημιουργία απλούστερων προγραμμάτων.
- Άμεση μεταφορά των αλγόριθμων σε πρόγραμμα.
- Διευκόλυνση ανάλυσης του προγράμματος σε τμήματα.
- Περιορισμός των λαθών κατά την ανάπτυξη του προγράμματος.
- Διευκόλυνση στην ανάγνωση και κατανόηση του προγράμματος από τρίτους.
- Ευκολότερη διόρθωση και συντήρηση.

(Εξετάσεις 2003)

Προγραμματιστικά περιβάλλοντα

16. Τι είναι ο μεταγλωττιστής και τι ο διερμηνευτής μιας γλώσσας προγραμματισμού;

Ο **μεταγλωττιστής** είναι ένα μεταφραστικό **πρόγραμμα** του υπολογιστή, το οποίο δέχεται σαν είσοδο ένα **πρόγραμμα** γραμμένο σε μια γλώσσα υψηλού επιπέδου και παράγει ένα ισοδύναμο **πρόγραμμα** σε γλώσσα μηχανής. Το τελευταίο μπορεί να εκτελείται από οποιονδήποτε υπολογιστή και είναι τελείως ανεξάρτητο από το αρχικό πρόγραμμα.

Ο **διερμηνευτής** είναι ένα μεταφραστικό **πρόγραμμα** που διαβάζει μία προς μία τις εντολές του αρχικού προγράμματος και για καθεμία από αυτές εκτελεί αμέσως μια ισοδύναμη ακολουθία εντολών μηχανής. Για να εκτελεστεί το πρόγραμμα, πρέπει να υπάρχει το πηγαίο πρόγραμμα και κάθε φορά να επαναλαμβάνεται η παραπάνω διαδικασία.

Σημείωση! Κάθε πρόγραμμα που γράφηκε σε οποιαδήποτε γλώσσα προγραμματισμού, πρέπει να μετατραπεί σε μορφή αναγνωρίσιμη και εκτελέσιμη από τον υπολογιστή, δηλαδή σε εντολές γλώσσες μηχανής. Η μετατροπή αυτή επιτυγχάνεται με τη χρήση ειδικών προγραμμάτων, του μεταγλωττιστή και του διερμηνευτή.

17. Ποια τα μειονεκτήματα και τα πλεονεκτήματα του μεταγλωττιστή και του διερμηνευτή (ομοιότητες - διαφορές); (Εξετάσεις 2002 & 2008)

- **Ομοιότητες**

Και ο μεταγλωττιστής και ο διερμηνευτής **μεταφράζουν** το πηγαίο πρόγραμμα (από γλώσσα υψηλού επιπέδου) σε γλώσσα μηχανής και επιπλέον και οι δύο **ανιχνεύουν** τα συντακτικά λάθη.

- **Διαφορές**

Ο μεταγλωττιστής μεταγλωττίζει όλο το πρόγραμμα και με την βοήθεια του **συνδέτη - φορτωτή** παράγεται το εκτελέσιμο, ενώ ο διερμηνευτής εκτελεί μία μία τις εντολές και δεν χρειάζεται συνδέτη. Επιπλέον, για να εκτελεστεί ένα πρόγραμμα με τον διερμηνευτή είναι απαραίτητη η παρουσία του πηγαίου προγράμματος ενώ με τον μεταγλωττιστή μόνο την πρώτη φορά.

Ο διερμηνευτής αφού εκτελεί τις εντολές μία μία έχει το **πλεονέκτημα** της άμεσης διόρθωσης των λαθών. Για τον λόγο αυτά χρησιμοποιείται συνήθως κατά την συγγραφή-διόρθωση ενός προγράμματος. Η εκτέλεση ενός προγράμματος με τον διερμηνευτή είναι **πιο αργή** (μειονέκτημα), γιατί για να εκτελεστεί το πρόγραμμα, πρέπει κάθε φορά να ξαναγίνεται η διερμηνεία από την αρχή.

Ο μεταγλωττιστής παράγει μια φορά το αντικείμενο πρόγραμμα και δεν χρειάζεται ξανά μεταγλώττιση αφού είναι σχεδόν εκτελέσιμο (με τη βοήθεια του συνδέτη) και έχει ως απόρροια των παραπάνω, το **πλεονέκτημα** της άμεσης εκτέλεσης του προγράμματος και το **μειονέκτημα** ότι, για να εκτελεστεί το πρόγραμμα, πρέπει να περάσει ολόκληρο από τη διαδικασία μεταγλώττισης και έτσι δεν προσφέρει άμεση διόρθωση των λαθών.

Σημείωση! Στα σύγχρονα προγραμματιστικά περιβάλλοντα ο μεταγλωττιστής και ο διερμηνευτής παρουσιάζονται συνήθως με μεικτές υλοποιήσεις. Έτσι ο διερμηνευτής χρησιμοποιείται κατά τη φάση δημιουργίας του προγράμματος και ο μεταγλωττιστής για την τελική έκδοση και εκμετάλλευση του προγράμματος.

18. Τι ονομάζεται πηγαίο και τι αντικείμενο πρόγραμμα;

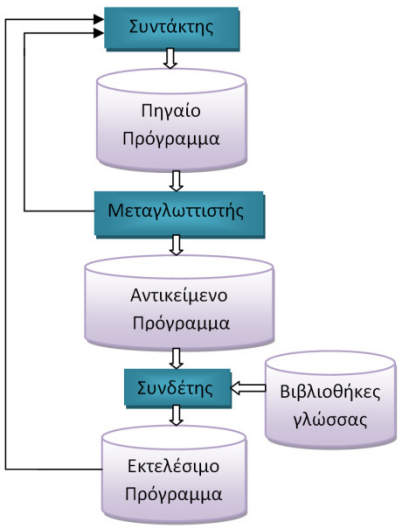
Το αρχικό πρόγραμμα το οποίο γράφεται από τον προγραμματιστή λέγεται πηγαίο πρόγραμμα, ενώ το πρόγραμμα που προκύπτει από τον μεταγλωττιστή ονομάζεται αντικείμενο πρόγραμμα.

19. Τι ονομάζεται συνδέτης-φορτωτής και ποιο είναι το αποτέλεσμα της χρήσης του;

Το αντικείμενο πρόγραμμα που προκύπτει από τον μεταγλωττιστή, παρόλο που είναι κατανοητό από τον υπολογιστή, συνήθως δεν μπορεί να εκτελεστεί, γιατί χρειάζεται να **συμπληρωθεί και να συνδεθεί** με άλλα τμήματα προγράμματος, απαραίτητα για την εκτέλεση του. Τα τμήματα αυτά μπορεί να τα έχει γράψει ο ίδιος ο προγραμματιστής του πηγαίου προγράμματος ή να βρίσκονται σε βιβλιοθήκες της γλώσσας, δηλαδή να είναι έτοιμα προγράμματα από τον κατασκευαστή της γλώσσας. Το πρόγραμμα που επιτρέπει τη σύνδεση αυτή ονομάζεται συνδέτης-φορτωτής. Το αποτέλεσμα του συνδέτη είναι η παραγωγή του εκτελέσιμου προγράμματος, το οποίο είναι το τελικό πρόγραμμα που εκτελείται από τον υπολογιστή.

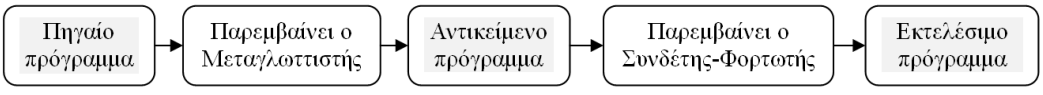
20. Να περιγράφετε τη διαδικασία για τη μετατροπή με μεταγλωττιστή ενός πηγαίου προγράμματος σε εκτελέσιμο πρόγραμμα, συμπεριλαμβανομένης της ανίχνευσης και διόρθωσης λαθών. (Εξετάσεις 2002)

Γράφουμε το *πηγαίο πρόγραμμα* στον **συντάκτη**, το οποίο αποτελεί είσοδο στον **μεταγλωττιστή** και παράγει το *αντικείμενο πρόγραμμα.*
Αν ο μεταγλωττιστής εντοπίσει συντακτικά λάθη στο *πηγαίο πρόγραμμα*, με κατάλληλα διαγνωστικά μηνύματα ζητάει από τον προγραμματιστή τη διόρθωσή τους **μέχρι να εξαλειφθούν** τα συντακτικά λάθη.
Το αποτέλεσμα της μεταγλώττισης είναι το **αντικείμενο πρόγραμμα**, εκφρασμένο σε μορφή κατανοητή από τον υπολογιστή (γλώσσα μηχανής) αλλά δεν μπορεί να εκτελεστεί.
Ο **συνδέτης** δέχεται το αντικείμενο πρόγραμμα και το συνδέει με τις βιβλιοθήκες της γλώσσας, δημιουργώντας το εκτελέσιμο πρόγραμμα που εκτελείται από τον υπολογιστή.
Τα **λογικά λάθη** εντοπίζονται με επανεκτέλεση και έλεγχο από τον προγραμματιστή, ώσπου να διορθωθούν όλα.



21. Να περιγραφεί σχηματικά η διαδικασία ενός προγράμματος μεταγλώττισης και σύνδεσης.

Η διαδικασία της μεταγλώττισης και σύνδεσης μπορεί να περιγραφεί σχηματικά ως εξής:



Εδώ πρέπει να σημειώσουμε ότι η δημιουργία του εκτελέσιμου προγράμματος γίνεται μόνο στην περίπτωση που το αρχικό πρόγραμμα **δεν περιέχει λάθη**. Τις περισσότερες φορές το πρόγραμμα αρχικά περιέχει λάθη.

22. Ποια είναι τα είδη λαθών ενός προγράμματος;

Τα λάθη ενός προγράμματος είναι γενικά δύο ειδών. Τα **λογικά λάθη** που εμφανίζονται μόνο κατά την εκτέλεση του προγράμματος και τα **συντακτικά λάθη** που εμφανίζονται κατά τη μεταγλώττιση του προγράμματος.

23. Πού οφείλονται - πώς αντιμετωπίζονται τα λάθη ενός προγράμματος;

Τα **συντακτικά λάθη** ανιχνεύονται από τον μεταγλωττιστή ή τον διερμηνευτή εμφανίζοντας κατάλληλα διαγνωστικά μηνύματα και οφείλονται σε αναγραμματισμούς ονομάτων εντολών και παραλείψεις στον κώδικα, όπως παράληψη δήλωσης μεταβλητών και πρέπει να διορθωθούν, ώστε να παραχθεί το τελικό εκτελέσιμο πρόγραμμα. (Εξετάσεις 2009)

Τα **λογικά λάθη** είναι τα πλέον σοβαρά και δύσκολα στη διόρθωση τους και οφείλονται σε σφάλματα κατά την υλοποίηση του αλγόριθμου. Ανιχνεύονται και αντιμετωπίζονται μόνο από τον προγραμματιστή αφού ελέγξει τα αποτελέσματα του προγράμματος του.

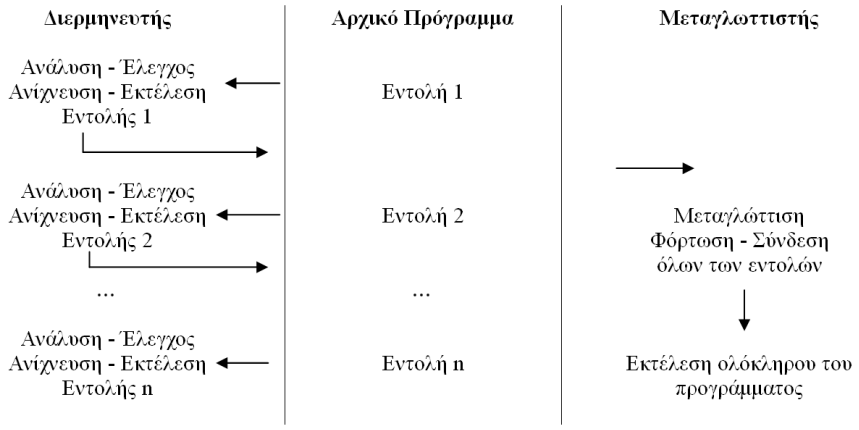
Η εντολή **Αν α>0** περιέχει συντακτικό λάθος (λείπει το **τότε**). Η εντολή **ΜΟ←α+β/2** περιέχει λογικό λάθος (λείπουν παρενθέσεις).

24. Τι περιλαμβάνει το στάδιο της διόρθωσης των λαθών προγράμματος;

Στην αρχή διορθώνονται όλα τα **συντακτικά λάθη**. Το πρόγραμμα που προκύπτει **επαναϋποβάλεται** για μεταγλώττιση και η διαδικασία αυτή επαναλαμβάνεται μέχρις ότου να εξαλειφτούν πλήρως όλα τα λάθη.

25. Ποια είναι σχηματικά η διαδικασία μετάφρασης και εκτέλεσης ενός προγράμματος με χρήση διερμηνευτή;

Ο διερμηνευτής δέχεται μια μια τις εντολές του πηγαίου προγράμματος. Αναλύει και ανιχνεύει τα λάθη σε κάθε εντολή εμφανίζοντας κατάλληλο διαγνωστικό μήνυμα σε περίπτωση λάθους. Η διαδικασία πραγματοποιείται μέχρι να μην υπάρχει λάθος στην εντολή αυτή, οπότε και εκτελείται. Η διαδικασία επαναλαμβάνεται για όλες τις εντολές του πηγαίου προγράμματος και σχηματικά απεικονίζεται παρακάτω:



26. Τι ονομάζεται συντάκτης προγραμμάτων;

Συντάκτης (editor) ονομάζεται το πρόγραμμα το οποίο χρησιμοποιείται για την αρχική σύνταξη των προγραμμάτων και τη διόρθωση τους. Είναι ουσιαστικά ένας **μικρός επεξεργαστής κειμένου** με δυνατότητες που διευκολύνουν τη γρήγορη γραφή των εντολών των προγραμμάτων.

27. Τι Ποιός είναι ο ρόλος του συντάκτη και του συνδέτη σε ένα προγραμματιστικό περιβάλλον; (Εξετάσεις 2014)

I. Ο συντάκτης είναι ένας επεξεργαστής κειμένου όπου συντάσσεται και υποβάλλεται σε επεξεργασία το πηγαίο πρόγραμμα.
II. Ο συνδέτης είναι πρόγραμμα που συνδέει το αντικείμενο πρόγραμμα με τις βιβλιοθήκες της γλώσσας και δημιουργεί το εκτελέσιμο πρόγραμμα.

28. Ποια είναι τα προγράμματα και τα εργαλεία που περιέχει ένα προγραμματιστικό περιβάλλον;

Ένα προγραμματιστικό περιβάλλον περιέχει τουλάχιστον τρία προγράμματα:

- τον συντάκτη,
- τον μεταγλωττιστή,
- τον συνδέτη.

Εκτός, όμως, από αυτά και ανάλογα με τις ιδιότητες και τα ιδιαίτερα χαρακτηριστικά του εκάστοτε προγραμματιστικού περιβάλλοντος μπορεί να περιέχει επιπλέον δυνατότητες και εργαλεία. Ένα σύγχρονο προγραμματιστικό περιβάλλον χρησιμοποιεί μεικτές υλοποιήσεις μεταγλωττιστή και διερμηνευτή. Διερμηνευτή κατά τη φάση της δημιουργίας του προγράμματος και μεταγλωττιστή για την τελική έκδοση και εκμετάλλευση του προγράμματος.

Η εντολή GOTO (το μαύρο πρόβατο του προγραμματισμού)

29. Τι γνωρίζετε για την εντολή GOTO;

Η εντολή **GOTO** έχει ως αποτέλεσμα την αλλαγή της ροής του προγράμματος σε μια άλλη εντολή του προγράμματος εκτός από την επόμενή της. Η εντολή αυτή χώρισε τους προγραμματιστές σε δυο αντιμαχόμενες ομάδες. Η μία αποτελείτο από τους φανατικούς υποστηρικτές της χρήσης της, που έλυναν εύκολα και αβασάνιστα προβλήματα της ανάπτυξης προγραμμάτων, και η δεύτερη με πολέμιους, που έβλεπαν ότι η εντολή αυτή ήταν υπεύθυνη για τη δυσκολία στην αρχική σχεδίαση της λύσης, στην παρακολούθηση και κατανόηση του προγράμματος και τέλος στη συντήρησή του.

30. Από πού προήλθε ο δομημένος προγραμματισμός;

Ο δομημένος προγραμματισμός προήλθε από την ανάγκη περιορισμού της ανεξέλεγκτης χρήσης της εντολής **GOTO**.

Σημείωση! Αρχικός στόχος ήταν η κατάργηση της εντολής GOTO (πήγαινε σε), που αλλάζει την ροή του προγράμματος σε μια άλλη εντολή του προγράμματος εκτός από την επόμενή της, αλλά κατέληξε να είναι βασική μεθοδολογία προγραμματισμού.