Τεχνικές σχεδίασης προγραμμάτων, Προγραμματιστικά Περιβάλλοντα

Ενότητες βιβλίου: 6.4, 6.7
Ώρες διδασκαλίας: **1**

Τεχνικές σχεδίασης προγραμμάτων

Στο βιβλίο γίνεται αναφορά σε μία τεχνική για την ανάπτυξη προγραμμάτων που είναι γνωστή ως **ιεραρχική σχεδίαση και επίλυση** ή αλλιώς **διαδικασία σχεδίασης «από πάνω προς τα κάτω»** (Top-Down Design). Πρόκειται για μια τεχνική που λίγο πολύ όλοι μας χρησιμοποιούμαι σε καθημερινή βάση αλλά δεν το συνειδητοποιούμε πάντα.

Σκοπός της ιεραρχικής σχεδίασης είναι η διάσπαση ενός προβλήματος σε μικρότερα (και άρα απλούστερα) υποπροβλήματα. Λύνοντας τα υποπροβλήματα και ενώνοντας τις λύσεις τους έχουμε την λύση του αρχικού προβλήματος.

Δομημένος λέγεται ο προγραμματισμός που κάνει χρήση μόνο των δομών **ακολουθίας** , **επιλογής** και **επανάληψης**. Υπάρχουν και άλλοι τρόποι προγραμματισμού (π.χ. αντικειμενοστραφής) οι οποίοι όμως είναι εκτός ύλης. Τα πλεονεκτήματα που αναφέρει το βιβλίο στην ενότητα 6.4.3 είναι του δομημένου προγραμματισμού σε σχέση με τον προγραμματισμό που κάνει χρήση της εντολής **GOTO**. Εδώ και πολλά χρόνια η χρήση της εντολής **GO TO** έχει σχεδόν εξαλειφθεί.

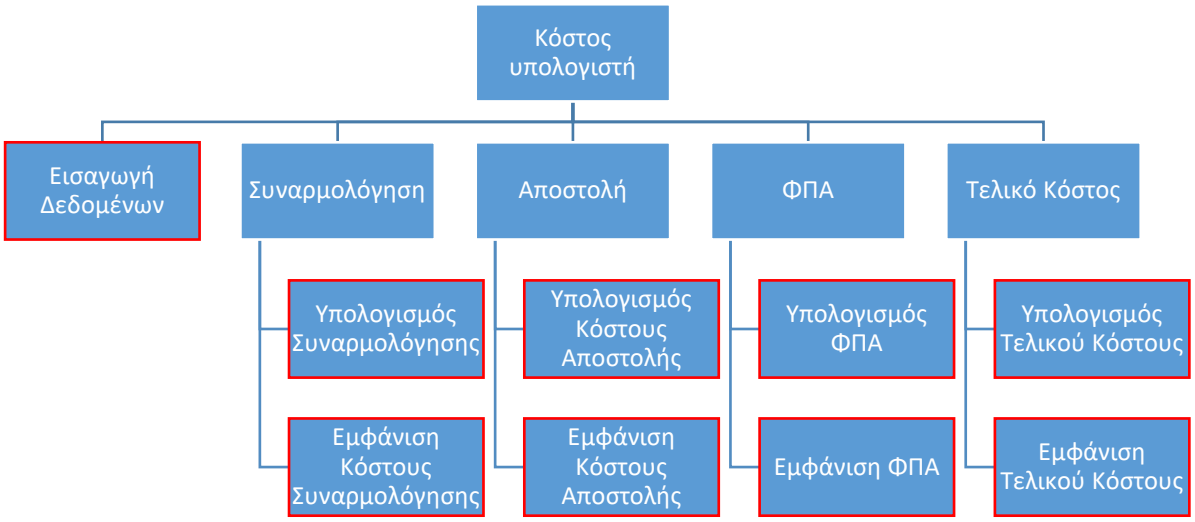
Παράδειγμα

Έστω ένα ηλεκτρονικό κατάστημα που πουλάει υπολογιστές. Οι πελάτες παραγγέλνουν τα εξαρτήματα που επιθυμούν και το κατάστημα αφού τα συναρμολογήσει , τα αποστέλλει στον πελάτη με εταιρία ταχυμεταφορών (courier).
Να αναπτύξετε πρόγραμμα που αφού διαβάσει την τιμή των εξαρτημάτων (χωρίς ΦΠΑ) , να υπολογίζει και να εμφανίζει:

- το κόστος συναρμολόγησης του υπολογιστή (10% επί της τιμής),
- το κόστος αποστολής (5% επί του κόστους εξαρτημάτων και συναρμολόγησης),
- τον ΦΠΑ που αντιστοιχεί (24% επί κόστους εξαρτημάτων, συναρμολόγησης και αποστολής)
- Την τελική τιμή

Λύση

Διασπάμε το αρχικό πρόβλημα σε υποπροβλήματα. Αν ένα υποπρόβλημα διασπάτε σε μικρότερα τότε το διασπάμε και αυτό κοκ. Στο σχήμα που ακολουθεί φαίνεται η ιεραρχική σχεδίαση του προβλήματος.



Στην συνέχεια γράφουμε τις εντολές για κάθε μικρό πρόβλημα που έχει προκύψει από την ιεραρχική σχεδίαση. Αυτό ονομάζεται **Τμηματικός Προγραμματισμός.** Στο σχήμα φαίνονται με κόκκινο περίγραμμα τα υποπροβλήματα για τα οποία θα γράψουμε εντολές.

```
ΔΙΑΒΑΣΕ κόστος                          ! Εισαγωγή Δεδομένων
συν ← (10*κόστος)/100                   ! Υπολογισμός Συναρμολόγησης
ΓΡΑΨΕ συν                               ! Εμφάνιση Κόστους Συναρμολόγησης
αποστολή ← (5*(κόστος+συν))/100         ! Υπολογισμός  Κόστους Αποστολής
ΓΡΑΨΕ αποστολή                          ! Εμφάνιση Κόστους Αποστολής
φπα ← (24*(κόστος+συν+αποστολή))/100    ! Υπολογισμός  ΦΠΑ
ΓΡΑΨΕ φπα                               ! Εμφάνιση ΦΠΑ
τελικό ← κόστος+συν+αποστολή+φπα        ! Υπολογισμός  τελικού κόστους
ΓΡΑΨΕ τελικό                            ! Εμφάνιση τελικού κόστους
```

Προγραμματιστικά Περιβάλλοντα

Ένα πρόγραμμα γραμμένο σε μια γλώσσα προγραμματισμού (όπως π.χ. Java, C , Python) δεν είναι κατανοητό από τον υπολογιστή. Χρειάζεται να μετατραπεί σε γλώσσα μηχανής (η μόνη γλώσσα που καταλαβαίνει ο υπολογιστής). Αυτή την δουλειά την κάνει ο **μεταγλωττιστής** ή ο **διερμηνευτής**. Ποια είναι όμως η διαφορά τους;

Ο **μεταγλωττιστής** παίρνει όλο το πρόγραμμα και το μετατρέπει σε γλώσσα μηχανής. Ο **διερμηνευτής** παίρνει μία μία τις εντολές, τις μετατρέπει σε εντολές γλώσσας μηχανής και τις δίνει

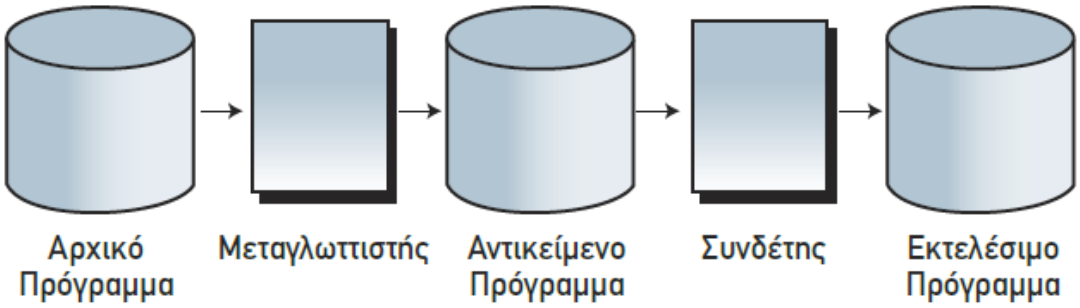
στον υπολογιστή για να την εκτελέσει. Ο **διερμηνευτής** λειτουργεί όπως και ο διερμηνέας μεταξύ δύο πολιτικών που μιλούν διαφορετικές γλώσσες. Κάθε φράση του ενός μεταφράζεται για την κατανόηση της από τον άλλο πολιτικό.

Το παρακάτω σχήμα είναι από το βιβλίο και δείχνει τα στάδια που ακολουθούνται προκειμένου να καταλήξει ένα πρόγραμμα σε μορφή κατάλληλη για εκτέλεση από τον υπολογιστή.

Το **αρχικό** ή **πηγαίο** πρόγραμμα είναι αυτό που γράφουμε χρησιμοποιώντας μια γλώσσα προγραμματισμού (π.χ. Python). Το **αντικείμενο** πρόγραμμα είναι το αρχικό όταν μετατραπεί σε γλώσσα μηχανής από τον μεταγλωττιστή. Όμως ένα πρόγραμμα μπορεί να κάνει χρήση κάποιας έτοιμης συνάρτησης. Στην ΓΛΩΣΣΑ έχουμε οκτώ τέτοιες συναρτήσεις (ΗΜ, ΣΥΝ, ΛΟΓ κλπ). Κάθε συνάρτηση είναι ένα πρόγραμμα που κάποιοι άλλοι προγραμματιστές έχουν φτιάξει για εμάς. Δουλειά του **συνδέτη-φορτωτή** είναι να βρει τις συναρτήσεις (σε γλώσσα μηχανής) που χρησιμοποιεί το πρόγραμμα μας και να τις ενσωματώσει στο **αντικείμενο** πρόγραμμα. Το τελικό πρόγραμμα (**εκτελέσιμο πρόγραμμα**) είναι και αυτό που μπορεί να διανεμηθεί σε όποιον θέλει να το τρέξει στον υπολογιστή του.

Εκτός από τις συναρτήσεις που έχει η κάθε γλώσσα προγραμματισμού μπορούμε αν θέλουμε να φτιάξουμε και δικές μας. Στις τελευταίες ενότητες του βιβλίου θα δούμε πως κατασκευάζονται οι συναρτήσεις και ένα άλλο είδος παρόμοιων «μικρών προγραμμάτων» που ονομάζονται **διαδικασίες**. Οι σύγχρονες γλώσσες προγραμματισμού συνοδεύονται από πληθώρα συναρτήσεων και διαδικασιών γι' αυτό και είναι χωρισμένες σε ομάδες τις οποίες και ονομάζουμε **βιβλιοθήκες**.

Η συγγραφή προγραμμάτων είναι μία επίπονη και χρονοβόρα εργασία γι' αυτό και όλοι οι προγραμματιστές σήμερα χρησιμοποιούν κατάλληλους επεξεργαστές κειμένου (**συντάκτες**) που διευκολύνουν κατά πολύ την δακτυλογράφηση. Ας υποθέσουμε ότι σε ένα πρόγραμμα έχουμε μία μεταβλητή με όνομα **η_επίδοση_του_αθλητή** την οποία και έχουμε ξαναγράψει. Δουλειά του συντάκτη είναι μόλις ξεκινήσουμε να γράφουμε την μεταβλητή να την συμπληρώσει αυτόματα για εμάς.



Λογικά και Συντακτικά λάθη

Συντακτικά είναι τα λάθη που εντοπίζει ο διερμηνευτής ή μεταγλωττιστής σε ένα πρόγραμμα όταν προσπαθεί να το μετατρέψει σε γλώσσα μηχανής. Τέτοια λάθη είναι εύκολο να εντοπιστούν και διορθωθούν.

Παραδείγματα συντακτικών λαθών:

- 5:6 (ο τελεστής της διαίρεσης είναι /)
- _x (δεν επιτρέπεται μια μεταβλητή να αρχίζει με _)
- Χ→Χ+1 (το σύμβολο της εκχώρησης είναι ←)

Τα λογικά λάθη είναι δυσκολότερο να εντοπιστούν διότι δεν μπορεί ούτε ο διερμηνευτής ούτε ο μεταγλωττιστής να τα εντοπίσουν. Τέτοια λάθη παρουσιάζονται όταν η εντολή που γράφουμε δεν υπολογίζει αυτό που θέλαμε αλλά κάτι άλλο. Έστω ότι θέλουμε να φτιάξουμε τμήμα προγράμματος που διαβάζει δύο αριθμούς και υπολογίζει τον μέσο όρο τους.

```
ΔΙΑΒΑΣΕ x,y
ΜΟ ← (x+y)/3
ΓΡΑΨΕ ΜΟ
```

Το λάθος σε αυτή την περίπτωση είναι ότι ο τύπος του μέσου όρου είναι (x+y)/2 και όχι (x+y)/3.

Σωστό - Λάθος

1. Ένα πρόγραμμα σε γλώσσα μηχανής είναι μια ακολουθία δυαδικών ψηφίων.
2. Ένα πρόγραμμα σε γλώσσα μηχανής χρειάζεται μεταγλώττιση. (2004-Θ1Α)
3. Η εντολή GOTO που αλλάζει τη ροή εκτέλεσης ενός προγράμματος είναι απαραίτητη στο δομημένο προγραμματισμό. (Ε2004-Θ1Γ3)
4. Ο μεταγλωττιστής δέχεται στην είσοδό του ένα πρόγραμμα γραμμένο σε μια γλώσσα υψηλού επιπέδου και παράγει ένα ισοδύναμο πρόγραμμα σε γλώσσα μηχανής.
5. Το πηγαίο πρόγραμμα εκτελείται από τον υπολογιστή χωρίς μεταγλώττιση.
6. Ο διερμηνευτής διαβάζει μία προς μία τις εντολές του πηγαίου προγράμματος και για κάθε μια εκτελεί αμέσως μια ισοδύναμη ακολουθία εντολών μηχανής. (2004-Θ1Α)
7. Τα λογικά λάθη είναι συνήθως λάθη σχεδιασμού και δεν προκαλούν τη διακοπή της εκτέλεσης του προγράμματος. (Ε2002-Θ1Α4)
8. Τα συντακτικά λάθη στον πηγαίο κώδικα εμφανίζονται κατά το στάδιο της μεταγλώττισής του. (Ε2004-Θ1Γ4)
9. Το πρόγραμμα που παράγεται από το μεταγλωττιστή λέγεται εκτελέσιμο. (Ε2006-Θ1Α3)

Ερωτήσεις

10. Σε ποιες στοιχειώδεις λογικές δομές στηρίζεται ο δομημένος προγραμματισμός; Να αναφέρετε τέσσερα πλεονεκτήματα του δομημένου προγραμματισμού. (Ε2003-Θ1Δ)
11. Ποια είναι τα πλεονεκτήματα του δομημένου προγραμματισμού; (Β2001-Θ1Δ1)
12. Ποια η διαφορά μεταξύ διερμηνευτή και μεταγλωττιστή; (2008-Θ1Β2)
13. Να περιγράψετε τη διαδικασία για τη μετατροπή με μεταγλωττιστή ενός πηγαίου προγράμματος σε εκτελέσιμο πρόγραμμα, συμπεριλαμβανομένης της ανίχνευσης και διόρθωσης λαθών. (2002-Θ1Γ)
14. Ποιες είναι οι διαφορές μεταξύ μεταγλωττιστή (compiler) και διερμηνευτή (interpreter). (Ε2002-Θ1Β)
15. Πότε εμφανίζονται τα συντακτικά λάθη ενός προγράμματος και πότε τα λογικά; (2009-Θ1Γ2α)
16. Δίνονται οι παρακάτω λανθασμένες εντολές για τον υπολογισμό του μέσου όρου δύο αριθμών:
    1. Γ ← Α+Β/2
    2. Γ ← (Α+Β/2
    3. Γ ← (Α+Β/2)

4. Γ ← (Α+Β):2

Να γράψετε τον αριθμό της κάθε εντολής (1, 2, 3, 4) και δίπλα τη λέξη συντακτικό ή τη λέξη λογικό, ανάλογα με το είδος του λάθους. (2009-Θ1Γ2β)

17. Δίνονται οι παρακάτω προτάσεις:
    Π1. Ο συνδέτης-φορτωτής μετατρέπει το ____1____ πρόγραμμα σε ____2____πρόγραμμα
    Π2. Ο συντάκτης χρησιμοποιείται για να δημιουργηθεί το ____3____ πρόγραμμα
    Π3. Ο μεταγλωττιστής μετατρέπει το ____4____ πρόγραμμα σε ____5____ πρόγραμμα
    και οι παρακάτω λέξεις:
    α. αντικείμενο β. εκτελέσιμο γ. πηγαίο.
    i) Να γράψετε τους αριθμούς (1–5) των κενών διαστημάτων των προτάσεων και δίπλα το γράμμα της λέξης (α, β, γ) που αντιστοιχεί σωστά. ΣΗΜΕΙΩΣΗ: Κάποιες από τις λέξεις χρησιμοποιούνται περισσότερες φορές από μία.
    ii) Κατά την ανάπτυξη ενός προγράμματος σε ένα προγραμματιστικό περιβάλλον, με ποια χρονική σειρά πραγματοποιούνται τα βήματα που περιγράφουν οι παραπάνω προτάσεις; Να απαντήσετε γράφοντας τα Π1, Π2, Π3 με τη σωστή σειρά. (2007-Θ1Δ)

Λύσεις

1. Σωστή
2. Λάθος
3. Λάθος
4. Σωστή
5. Λάθος
6. Σωστή
7. Σωστή
8. Σωστή
9. Λάθος
10. Ακολουθίας, επιλογής και επανάληψης. Σελ 136 του σχολικού βιβλίου.
11. Σελ 136 του σχολικού βιβλίου.
12. Ο μεταγλωττιστής διαβάζει ολόκληρο το πρόγραμμα (που είναι γραμμένο σε κάποια γλώσσα υψηλού επιπέδου) και το μετατρέπει σε γλώσσα μηχανής. Ο διερμηνευτής διαβάζει μία προς μία τις εντολές, τις μετατρέπει σε γλώσσα μηχανής και τις εκτελεί.
13. Χρησιμοποιώντας τον συντάκτη μιας γλώσσας υψηλού επιπέδου γράφουμε το (πηγαίο) πρόγραμμα. Ο μεταγλωττιστής το μετατρέπει σε γλώσσα μηχανής (αντικείμενο). Αν ο μεταγλωττιστής εντοπίσει συντακτικά λάθη μας ενημερώνει με κατάλληλα μηνύματα προκειμένου να τα διορθώσουμε και να ξαναυποβάλουμε το πηγαίο πρόγραμμα σε μεταγλώττιση. Η διαδικασία επαναλαμβάνεται έως ότου εξαλειφθούν όλα τα συντακτικά λάθη. Ο συνδέτης δέχεται το αντικείμενο πρόγραμμα και το συνδέει με τις βιβλιοθήκες της γλώσσας με αποτέλεσμα να παράγεται το εκτελέσιμο πρόγραμμα. Από τα αποτελέσματα του εκτελέσιμου αρχείου μπορούν να εντοπιστούν πιθανά λογικά λάθη τα οποία και διορθώνονται στο πηγαίο πρόγραμμα. Η όλη διαδικασία επαναλαμβάνεται έως ότου εξαλειφθούν όλα τα λογικά λάθη.
14. Δείτε ερώτηση 25.
15. Δείτε ερώτηση 26.
16. Λογικό, Συντακτικό, Λογικό, Συντακτικό.
17. 30. i)1. α , 2. β, 3. γ , 4. γ, 5.α ii) Π2 , Π3, Π1