4.1 ΑΝΑΛΥΣΗ ΠΡΟΒΛΗΜΑΤΩΝ

Η ανάλυση ενός προβλήματος σε ένα σύγχρονο υπολογιστικό περιβάλλον περιλαμβάνει:

- *την καταγραφή της υπάρχουσας πληροφορίας για το πρόβλημα,*
- *την αναγνώριση των ιδιαιτεροτήτων του προβλήματος,*
- *την αποτύπωση των συνθηκών και προϋποθέσεων υλοποίησής του*

και στη συνέχεια:

- *την πρόταση επίλυσης με χρήση κάποιας μεθόδου, και*
- *την τελική επίλυση με χρήση υπολογιστικών συστημάτων.*

Έτσι, κατά την ανάλυση ενός προβλήματος που πρόκειται να επιλυθεί με χρήση υπολογιστικού συστήματος, θα πρέπει να δοθεί απάντηση σε καθεμία από τις επόμενες ερωτήσεις:

1. *Ποια είναι τα δεδομένα και το μέγεθος του προβλήματος,*
2. *Ποιες είναι οι συνθήκες (προϋποθέσεις) που πρέπει να πληρούνται για την επίλυση του προβλήματος,*
3. *Ποια είναι η πλέον αποδοτική μέθοδος επίλυσής του (σχεδίαση αποδοτικότερου αλγορίθμου),*
4. *Πώς θα καταγραφεί η λύση του (δηλαδή πως θα περιγραφεί ο αλγόριθμος, π.χ. με φυσική γλώσσα κατά βήματα, με ψευδογλώσσα κλπ.), και*
5. *Ποιος είναι ο τρόπος υλοποίησης στο συγκεκριμένο υπολογιστικό σύστημα (π.χ. επιλογή γλώσσας προγραμματισμού).*

6.1 Η ΈΝΝΟΙΑ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ

Η επίλυση ενός προβλήματος με τον υπολογιστή περιλαμβάνει τα παρακάτω **τρία στάδια**:

1. *Τον ακριβή προσδιορισμό του προβλήματος.*
2. *Την ανάπτυξη (σχεδίαση) του αντίστοιχου αλγορίθμου.*
3. *Τη διατύπωση του αλγορίθμου σε κατανοητή μορφή από τον υπολογιστή.*

Ο προγραμματισμός ασχολείται με το τρίτο στάδιο, δηλαδή τη δημιουργία του προγράμματος.

Πρόγραμμα είναι το σύνολο των εντολών που πρέπει να δοθούν στον υπολογιστή, ώστε να υλοποιηθεί ο αλγόριθμος για την επίλυση του προβλήματος.

Βασικά στοιχεία ενός προγράμματος είναι **τα δεδομένα** και **οι δομές δεδομένων** επί των οποίων ενεργεί.
Οι αλγόριθμοι και οι δομές δεδομένων είναι μια αδιάσπαστη ενότητα κατά τη δημιουργία προγραμμάτων (Αλγόριθμοι + Δομές δεδομένων = Προγράμματα).

6.4 ΤΕΧΝΙΚΕΣ ΣΧΕΔΙΑΣΗΣ ΠΡΟΓΡΑΜΜΑΤΩΝ

Από την αρχή της εμφάνισης των υπολογιστών γίνονται συνεχείς προσπάθειες ανάπτυξης μεθοδολογιών και τεχνικών προγραμματισμού, που θα εξασφαλίζουν τη δημιουργία απλών και κομψών προγραμμάτων, την εύκολη γραφή τους όσο και την κατανόησή τους.

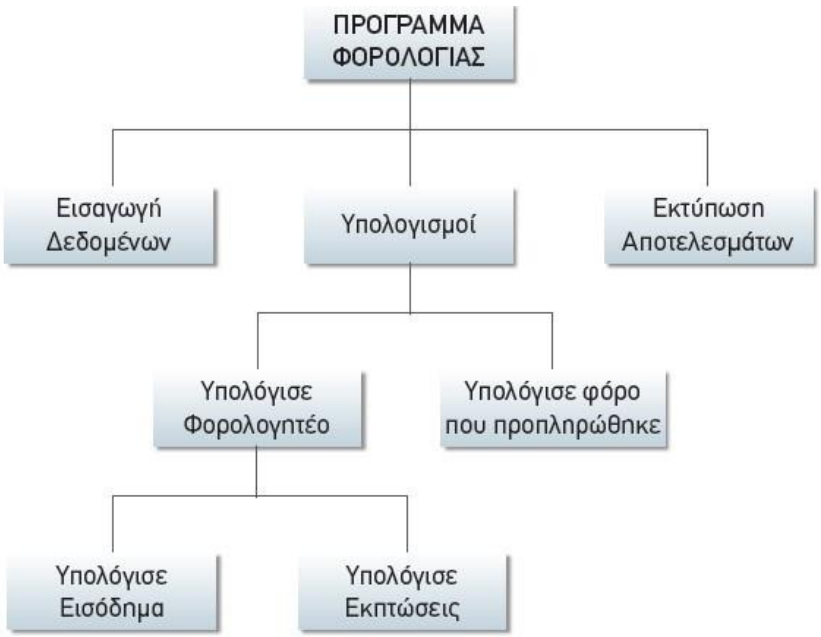
6.4.1 Ιεραρχική σχεδίαση προγράμματος

Η ιεραρχική σχεδίαση ή ιεραρχικός προγραμματισμός χρησιμοποιεί τη στρατηγική της συνεχούς διαίρεσης του προβλήματος σε υποπροβλήματα.

Η τεχνική της ιεραρχικής σχεδίασης ή της σχεδίασης «από επάνω προς τα κάτω» όπως συχνά ονομάζεται (top-down program design) περιλαμβάνει τον καθορισμό των λειτουργιών ενός προγράμματος σε επίπεδα, όπου οι βασικές λειτουργίες θα βρίσκονται σε ένα ανώτερο επίπεδο, στη συνέχεια οι λειτουργίες αυτές θα διασπώνται σε μικρότερες σε ένα επίπεδο ακριβώς πιο κάτω, μέχρι το τελευταίο επίπεδο όπου οι λειτουργίες είναι πολύ απλές, ώστε να μπορούν να επιλυθούν εύκολα.

Σκοπός, επομένως, της ιεραρχικής σχεδίασης είναι η διάσπαση του προβλήματος σε μια σειρά από απλούστερα υποπροβλήματα, τα οποία να είναι εύκολο να επιλυθούν οδηγώντας στην επίλυση του αρχικού προβλήματος.

Για την υποβοήθηση της ιεραρχικής σχεδίασης χρησιμοποιούνται διάφορες διαγραμματικές τεχνικές. Ένα παράδειγμα δίνει το παρακάτω σχήμα:



Ιεραρχική σχεδίαση υπολογισμού του φόρου εισοδήματος

6.4.2 Τμηματικός προγραμματισμός

Η ιεραρχική σχεδίαση προγράμματος υλοποιείται με τον τμηματικό προγραμματισμό. Μετά τη διάσπαση του προβλήματος σε υποπροβλήματα, **κάθε υποπρόβλημα αποτελεί μια ανεξάρτητη ενότητα (module).**

Η σωστή διαίρεση του αρχικού προβλήματος σε υποπροβλήματα και κατά συνέπεια του αρχικού προγράμματος σε τμήματα προγράμματος είναι μία αρκετά πολύπλοκη διαδικασία.
Ο τμηματικός προγραμματισμός:

- *διευκολύνει τη δημιουργία του προγράμματος,*
- *μειώνει τα λάθη και*
- *επιτρέπει την ευκολότερη παρακολούθηση, κατανόηση και συντήρηση του προγράμματος από τρίτους.*

6.4.3 Δομημένος προγραμματισμός

Ο δομημένος προγραμματισμός αναπτύχθηκε από την ανάγκη να υπάρχει μία κοινή μεθοδολογία στην ανάπτυξη των προγραμμάτων και **τη μείωση των εντολών GOTO** που χρησιμοποιούνται στο πρόγραμμα. Έχει επικρατήσει απόλυτα και σήμερα αποτελεί τη βασική μεθοδολογία προγραμματισμού σε όλες σχεδόν τις γλώσσες προγραμματισμού.

Ο δομημένος προγραμματισμός δεν είναι απλώς ένα είδος προγραμματισμού, είναι μία 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.

Ο δομημένος προγραμματισμός στηρίζεται στη χρήση τριών και μόνο στοιχειωδών λογικών δομών:
- *τη* ***δομή της ακολουθίας***
- *τη* ***δομή της επιλογής***
- *τη* ***δομή της επανάληψης***

Όλα τα προγράμματα μπορούν να γραφούν χρησιμοποιώντας μόνο αυτές τις τρεις δομές καθώς και συνδυασμό τους.
Κάθε πρόγραμμα όπως και κάθε ενότητα προγράμματος έχει μόνο μία είσοδο και μόνο μία έξοδο.
Ο όρος δομημένος προγραμματισμός περιέχει τόσο την ιεραρχική σχεδίαση όσο και τον τμηματικό προγραμματισμό.

<u>Πλεονεκτήματα του δομημένου προγραμματισμού:</u>

1. *Δημιουργία απλούστερων προγραμμάτων.*
2. *Άμεση μεταφορά των αλγορίθμων σε προγράμματα.*
3. *Διευκόλυνση ανάλυσης του προγράμματος σε τμήματα.*
4. *Περιορισμός των λαθών κατά την ανάπτυξη του προγράμματος.*
5. *Διευκόλυνση στην ανάγνωση και κατανόηση του προγράμματος από τρίτους.*
6. *Ευκολότερη διόρθωση και συντήρηση.*