Τεχνολογικό Εκπαιδευτικό Ίδρυμα Μεσολογγίου

Θέμα Πτυχιακής Εργασίας: "Μελέτη των αντιλήψεων-αναπαραστάσεων των πρωτοετών σπουδαστών του τμήματος Ε.Π.Λ.Ο. για βασικές έννοιες του προγραμματισμού"

Επιβλέπων καθηγητής: Καζαντζής Χρήστος

Ακαδημαϊκό Έτος 2005-2006

Ο Σπουδαστής: Σωτήρης Λεονάρδος (Α.Μ.: 9252)

Abstract

Σκοπός της παρούσας πτυχιακής εργασίας είναι η διερεύνηση των αναπαραστάσεων των σπουδαστών του τμήματος Ε.Π.Δ.Ο. που παρακολουθούν εισαγωγικά μαθήματα προγραμματισμού, Ά εξαμήνου, σχετικά με την έννοια της μεταβλητής, της εντολής ανάθεσης τιμής σε μεταβλητή, των δομών ελέγχου και της έννοιας της επανάληψης. Η έρευνα έχει γίνει με την μέθοδο του ερωτηματολογίου (εμπειρική έρευνα) και ακολουθήθηκε στατιστική επεξεργασία των απαντήσεων προκειμένου να εξαχθούν συμπεράσματα για τις δυσκολίες και τις παρανοήσεις των σπουδαστών για τις παραπάνω βασικές έννοιες του προγραμματισμού. Το αντικείμενο της έρευνας μπορεί να φανεί χρήσιμο σε όσους ασχολούνται με την διδακτική της πληροφορικής αλλά και σε όσους ενδιαφέρονται να διευρύνουν τις γνώσεις τους στον τομέα της διδακτικής της πληροφορικής.

Ο σπουδαστής,
Λεονάρδος Σωτήριος

ΠΕΡΙΕΧΟΜΕΝΑ

1. Εισαγωγή

Η διδακτική σημασία της μελέτης των αναπαραστάσεων είναι πλέον αναμφισβήτητη στην εκπαιδευτική κοινότητα. Σαν έννοια είναι συνυφασμένη με την διδακτική έρευνα και αποτελεί απαραίτητο εργαλείο για τον εκπαιδευτικό που θέλει να κατανοήσει τις νοητικές λειτουργίες των σπουδαστών και πως αντιλαμβάνονται την πραγματικότητα. Οι αναπαραστάσεις των σπουδαστών με βάση την εμπειρία που έχουν με τις μηχανές και με τις γνωστικές διαδικασίες που χρησιμοποιούν δουλεύοντας με συστήματα πληροφορικής, αποτελούν ένα πεδίο έρευνας πολύ λίγο διευρυμένο. Η αλληλεπίδραση ανάμεσα στις αναπαραστάσεις και τις πραγματικές χρήσεις αποτελεί επίσης ενδιαφέρον θέμα για μελέτη. Η μελέτη των αναπαραστάσεων ενός τεχνολογικού μέσου που σχηματίζονται από τους σπουδαστές κατά την διάρκεια της μάθησης της χρήσης του έχει διττή όψη: αφ' ενός, η αναπαράσταση του μέσου σαν σύστημα αναφοράς μαζί με όλη την λειτουργία του και αφ' ετέρου, η αναπαράσταση του χειριστή-χρήστη. Όντας μη μοναδική, αφού κάθε χρήστης οικοδομεί την δική του, πρέπει να συγκριθεί με αυτή του «ειδικού» που υποτίθεται ότι ξέρει να χειρίζεται σωστά το τεχνολογικό μέσο. Η ανάλυση και η αναδιοργάνωση των αναπαραστάσεων αποκτά και ευρύτερες διαστάσεις που ξεπερνούν τα όρια της εκπαίδευσης και άπτονται ζητημάτων που αφορούν την ίδια την σύλληψη και δημιουργία λογισμικού, τη σχέση ανθρώπου-μηχανής, την εργονομία των διασυνδέσεων (interfaces), εγγίζει δηλαδή πολλούς σημαντικούς τομείς και τάσεις έρευνας της σύγχρονης πληροφορικής.

Κάτω από το πρίσμα αυτό, η διδασκαλία της πληροφορικής προϋποθέτει την κατανόηση του σχηματισμού των αναπαραστάσεων των σπουδαστών πάνω στον υπολογιστή και την πληροφορική. Στα πλαίσια της διδακτικής η αναπαράσταση θεωρείται εν σχέσει με την παρέκκλιση από τη συναφή της έννοια και προκαλεί δύο διαφορετικές αντιδράσεις εκ μέρους του εκπαιδευτικού. Είτε θεωρείται σαν ένα λάθος, ακόμα και σαν παράπτωμα, που πρέπει να εξαλειφθεί. Τότε η διδασκαλία δεν πρέπει να ενδιαφέρεται για τις αναπαραστάσεις. Είτε θεωρείται σαν ένα επεξηγηματικό σύστημα που πρέπει να κατανοηθεί και να αναλυθεί σαν εμπόδιο για ξεπέρασμα ή σαν σημείο στήριξης για να «προσεγγίσουμε» την έννοια.

2. Η εξέλιξη της εισαγωγής και της ένταξης της πληροφορικής στην εκπαίδευση

2.1. Το πλαίσιο ένταξης της πληροφορικής στην εκπαίδευση

Η Πληροφορική και οι Τεχνολογίες της Πληροφορικής και των Επικοινωνιών είναι πλέον παρούσες σε όλες τις πτυχές (οικονομικές, επιστημονικές, βιομηχανικές, πολιτιστικές, κοινωνικές) της ανθρώπινης ζωής. Για το λόγο αυτό, οι προβληματισμοί και οι προσπάθειες για την εισαγωγή και την ένταξη της Πληροφορικής στην εκπαίδευση ξεκίνησαν ήδη από την δεκαετία του 1970 και εντάθηκαν με την εμφάνιση των προσωπικών υπολογιστών (PCs). Η ραγδαία ανάπτυξη των δικτύων υπολογιστών και του διαδικτύου (Internet) καθώς και η σημαντική πτώση των τιμών των υπολογιστών τα τελευταία χρόνια καθιστά την εν λόγω συζήτηση εκ νέου επίκαιρη τοποθετώντας την συχνά σε νέα βάση και προβληματική (Κόμης, 2004).

Τα διάφορα εκπαιδευτικά συστήματα εισήγαγαν καταρχήν την Πληροφορική και αργότερα τις Τεχνολογίες της Πληροφορικής και των Επικοινωνιών (ΤΠΕ) μέσω διαδικασιών που εξελίχθηκαν σε διάφορες φάσεις και ακολουθώντας διαφορετικές παιδαγωγικές προσεγγίσεις (Baron, 1989). Οι διαφορετικές προσεγγίσεις που αφορούν την εισαγωγή, την ένταξη και την ενσωμάτωση της Πληροφορικής και των ΤΠΕ στο εκπαιδευτικό σύστημα, είναι, κάθε φορά, συνάντηση πολλών παραμέτρων που σχετίζονται (Κόμης & Μικρόπουλος, 2001 – Κόμης, 2004) :

- με τα αναλυτικά προγράμματα (δηλαδή με το τι διδάσκεται και πως διδάσκεται) και το πλαίσιο προγράμματος σπουδών (δηλαδή με τη γενικότερη προβληματική που προσδιορίζει το σκοπό και τη μεθοδολογία της διδασκαλίας),
- με την βαθμίδα εκπαίδευσης (στην περίπτωσή μας τριτοβάθμια) που αφορά η εισαγωγή και η ένταξη,
- με τους προς επίτευξη διδακτικούς και γνωστικούς στόχους (οι οποίοι κατά κανόνα διαφοροποιούνται ανάλογα με την βαθμίδα και μπορεί να αφορούν διάφορα γνωστικά αντικείμενα που χρησιμοποιούν την Πληροφορική ως εκπαιδευτικό μέσο ή εργαλείο),
- με τις οικονομικές (σχετίζονται κυρίως με το κόστος του εξοπλισμού), πολιτικές και κοινωνικές συγκυρίες κατά την περίοδο της ένταξης,
- με το επίπεδο της τεχνολογικής ανάπτυξης,
- με τις φιλοσοφικές και ιδεολογικές θεωρήσεις των πρωτεργατών της ένταξης.

Είναι απαραίτητο στο σημείο αυτό να γίνει η διάκριση ανάμεσα στους όρους *εισαγωγή, ένταξη* και *ενσωμάτωση* της Πληροφορικής και των ΤΠΕ στο εκπαιδευτικό σύστημα.

Με τον όρο *εισαγωγή* εννοούμε τόσο την ύπαρξη κάποιου ή κάποιων μαθημάτων Πληροφορικής και περιστασιακές χρήσεις της Πληροφορικής και των ΤΠΕ στα επιμέρους γνωστικά αντικείμενα.

Με τον όρο *ένταξη* εννοούμε τόσο την ύπαρξη ενός συνόλου μαθημάτων σχετικών με την Πληροφορική στο πρόγραμμα σπουδών, όσο και την αντιμετώπιση της χρήσης των ΤΠΕ ως εκπαιδευτικού μέσου σε μεγάλο μέρος του προγράμματος σπουδών στα διάφορα γνωστικά αντικείμενα. Στο πλαίσιο αυτό, ως σημείο εκκίνησης θεωρείται ότι οι σχολές αποκτούν την κατάλληλη τεχνολογική υποδομή, δημιουργείται κατάλληλο εκπαιδευτικό λογισμικό και γίνεται επιμόρφωση των εκπαιδευτικών στις παιδαγωγικές χρήσεις των ΤΠΕ.

Τέλος, με τον όρο *ενσωμάτωση* εννοούμε την τακτική χρήση των ΤΠΕ από την πλειονότητα των εκπαιδευτικών για την υλοποίηση του εκπαιδευτικού έργου με μετασχηματιστικό τρόπο (με τρόπο, δηλαδή, που οδηγεί σε ποιοτική βελτίωση της εκπαιδευτικής διαδικασίας). Υπό το πρίσμα αυτό, δεν υπάρχει κάποιο εκπαιδευτικό σύστημα που να έχει επιτύχει την ενσωμάτωση των ΤΠΕ στην καθημερινή του πρακτική.

Στο σημείο αυτό πρέπει, επίσης, να σημειωθεί ότι κάθε ανάλυση που αναφέρεται στην εισαγωγή, στην ένταξη και στην ενσωμάτωση της υπολογιστικής τεχνολογίας και των εφαρμογών της στο εκπαιδευτικό σύστημα οφείλει να λάβει υπόψη της ως συμπληρωματική παράμετρο και τον χρόνο. Είναι πλέον αδιαμφισβήτητο γεγονός ότι η τεχνολογική πραγματικότητα (δηλαδή το υλικό των υπολογιστών, το κόστος, η υπολογιστική τους ισχύς, η διάδοση της χρήσης τους στα διάφορα επαγγέλματα και σε πολλές καθημερινές ανθρώπινες δραστηριότητες) της Πληροφορικής, των τηλεπικοινωνιών και των οπτικοακουστικών μέσων εξελίσσεται ραγδαία τα τελευταία χρόνια. Ας θυμηθούμε τον γνωστό «Νόμο» του Gordon Moore για τα ηλεκτρονικά κυκλώματα, όπως διατυπώθηκε το 1964, που ισχύει κατά προσέγγιση και σήμερα : ο αριθμός κυκλωμάτων που μπορεί να τοποθετηθεί σε ένα τσιπ διπλασιάζεται κάθε χρόνο. Η εξέλιξη αυτή επηρεάζει καταλυτικά τις αντιλήψεις που σχηματίζονται για την θέση των ΤΠΕ στην εκπαιδευτική πράξη.

3. Ένα εννοιολογικό πλαίσιο για την Διδακτική της Πληροφορικής

3.1. Εννοιολογικό πλαίσιο και ορισμοί

3.1.1. Τι ορίζουμε ως Διδακτική της Πληροφορικής

Με τον όρο «Διδακτική της Πληροφορικής» εννοούμε τη μελέτη, τον προβληματισμό και την εξάσκηση πάνω στο πώς να διδαχτούν επιμέρους γνωστικά αντικείμενα της επιστήμης της Πληροφορικής.

Το **περιεχόμενο** των γνωστικών αντικειμένων της Πληροφορικής διαμορφώνεται μέσα από τις απαιτήσεις και εξελίξεις της ίδιας της επιστήμης και των τεχνολογιών που προκύπτουν από αυτή. Επίσης μέσα από τις ευρύτερες επιστημονικές, κοινωνικές, πολιτισμικές και οικονομικές ανάγκες και ενδιαφέροντα.

Η βαθειά γνώση αυτού του περιεχομένου είναι το σημαντικότερο εφόδιο ενός ανθρώπου που συμμετέχει ή επιθυμεί να συμμετάσχει στο σχεδιασμό, την ανάπτυξη και εφαρμογή διδακτικών δραστηριοτήτων με αντικείμενο την επιστήμη της Πληροφορικής. Η γνώση ωστόσο του περιεχομένου δεν είναι από μόνη της αρκετή για να κάνει κάποιον ικανό εκπαιδευτικό. Γι' αυτό απαιτείται η προσέγγιση μια σειράς πρόσθετων γνώσεων και δεξιοτήτων στις οποίες πρόκειται να εστιαστούμε κατά τη διάρκεια του μαθήματος.

Η Διδακτική σαν κλάδος της Παιδαγωγικής Επιστήμης ασχολείται με μια σειρά από ζητήματα:

- το ευρύτερο διδακτικό – μαθησιακό περιβάλλον στο οποίο εντάσσεται μια διδακτική δραστηριότητα,
- τις μεθόδους διδασκαλίας και τις παιδαγωγικές αρχές στις οποίες αυτές στηρίζονται,
- τους συμμετέχοντες σε μια διδακτική δραστηριότητα,
- τα διδακτικά μέσα τα οποία πρόκειται να χρησιμοποιηθούν.

Στα θέματα που προτείνονται στο επόμενο κεφάλαιο γίνεται μια προσπάθεια να εξεταστούν όλα τα παραπάνω ζητήματα σε σχέση με τη διδακτική της Πληροφορικής.

3.1.2. Διδακτική και διδακτικό τρίγωνο

Ένα από τα βασικά ερωτήματα που απασχολεί τις τελευταίες δεκαετίες την παιδαγωγική έρευνα, καθώς και την εκπαιδευτική κοινότητα γενικότερα, αφορά στο πώς ευνοείται η

οικοδόμηση των γνώσεων στο πλαίσιο ατομικών ή συλλογικών καταστάσεων διδασκαλίας. Με άλλα λόγια, πρόκειται για ένα ερώτημα το οποίο σχετίζεται άμεσα με το πρόβλημα της μάθησης και της Διδακτικής.

Σε αυτό το ερευνητικό πλαίσιο αναπτύχθηκε η ***Διδακτική των Επιστημών***, η οποία εξετάζει τις διαδικασίες μετάδοσης, πρόσκτησης και οικοδόμησης των γνώσεων με απώτερο στόχο την βελτίωση αυτών των διαδικασιών (Vergnaud, 1994). Μελετά, με άλλα λόγια, τις συνθήκες μέσα στις οποίες οι μαθητές μαθαίνουν ή δεν μαθαίνουν και εστιάζει την προσοχή της στα ιδιαίτερα προβλήματα που ανακύπτουν από το περιεχόμενο των γνώσεων όσο και των δεξιοτήτων που πρέπει να προσκτηθούν (Ραβάνης, 2003).

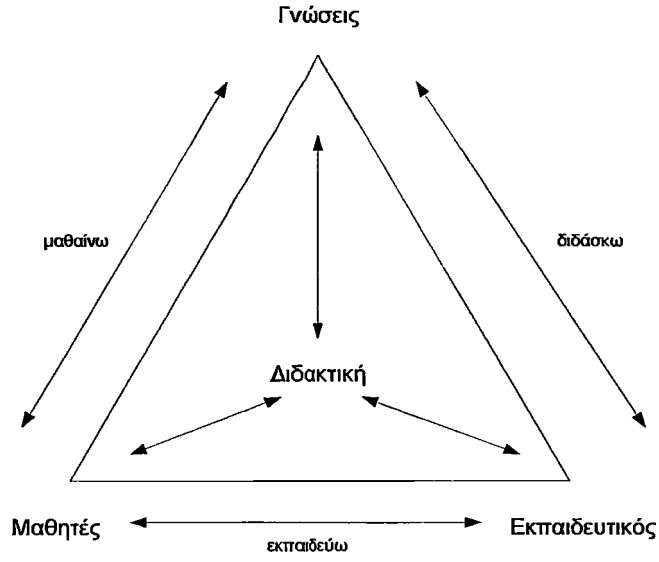
Υπό το πρίσμα αυτό, η ***Διδακτική*** ενδιαφέρεται για τους τρόπους με τους οποίους ευνοείται η οικοδόμηση των γνώσεων στο πλαίσιο ατομικών ή συλλογικών καταστάσεων διδασκαλίας. Ωστόσο, μέχρι σήμερα δεν υπάρχει μια θεωρία γενικής Διδακτικής και, συνακόλουθα, γίνεται χρήση των εννοιών και των θεωρητικών πλαισίων που αναπτύσσονται στις διδακτικές των διάφορων μαθημάτων και κυρίως αυτών που έχουν διαμορφωθεί στην Διδακτική των Επιστημών.

Η πρόοδος στη ***Διδακτική των Επιστημών*** είναι ραγδαία κατά την τελευταία εικοσαετία και συνίσταται κυρίως στη δημιουργία ενός θεωρητικού πλαισίου που βασίζεται στις *επιοικοδομιστικές* και *κοινονικοπολιτισμικές* ψυχολογικές θεωρίες. Οι πρώτες προσπάθειες δημιουργίας ενός θεωρητικού πλαισίου που να ξεφεύγει από τον εμπειρισμό ξεκίνησαν από το χώρο της ***Διδακτική των Μαθηματικών***. Το θεωρητικό αυτό πλαίσιο γνώρισε αρκετές διακυμάνσεις, αλλά τελικά κατάφερε να παγιώσει μια σειρά από βασικές έννοιες και μια μεθοδολογική προσέγγιση που είναι πλέον αποδεκτές στις μέρες μας από το σύνολο σχεδόν των ερευνητών στο χώρο της Διδακτικής των Επιστημών.

Μπορούμε να διακρίνουμε τρεις μεγάλους άξονες αυτού του θεωρητικού πλαισίου : την *επιστημολογία των γνώσεων* (η μελέτη του τρόπου με τον οποίο συγκροτούνται οι επιστημονικές γνώσεις), τη *γένεση και την πρόσκτηση των γνώσεων* από τα υποκείμενα που βρίσκονται σε κατάσταση μάθησης και την τοποθέτηση αυτής της γένεσης μέσα σε πραγματικές καταστάσεις (Vergnaud, 1994).

Τα τελευταία χρόνια έχουν γίνει σημαντικές προσπάθειες για μια διεπιστημονική και διαθεματική προσέγγιση που να επιτρέπει την ανάδυση κοινών εννοιών στις διδακτικές των διαφόρων μαθημάτων.

Η διδακτική συχνά αναπαρίσταται από ένα τρίγωνο (Houssaye, 1994), το οποίο συμβολίζει το σύστημα που συνδέει τις γνώσεις, το μαθητή και τον εκπαιδευτικό (Σχήμα 1).



Σχήμα 1 Το τρίγωνο της διδακτικής

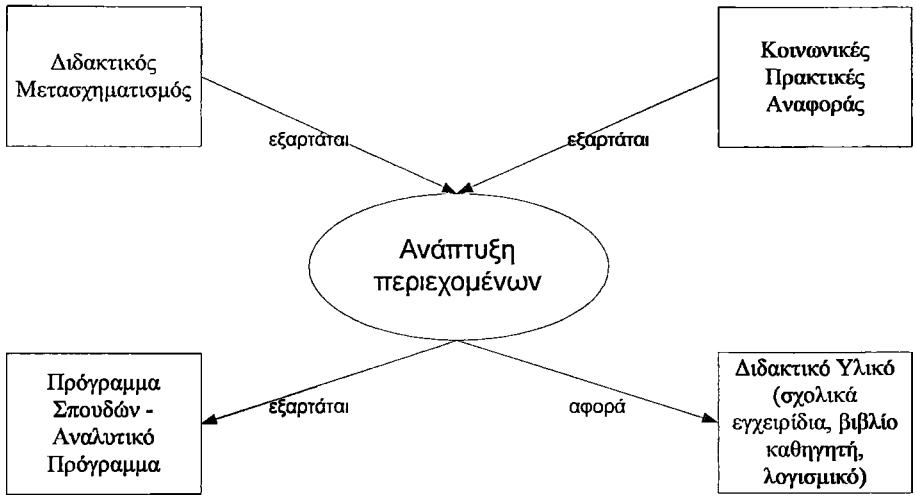
Καταρχήν, στο τρίγωνο αυτό πρέπει να ληφθούν υπόψη οι κορυφές παρότι κάθε μία από αυτές συνιστά ένα χώρο έρευνας που δεν αφορά αυτή καθαυτή την διδακτική. Στη συνέχεια, πρέπει επίσης να θεωρηθούν η επιστημολογική και εννοιολογική δομή του χώρου (που αφορούν συνεπώς στις ***γνώσεις*** και την προέλευσή τους), οι διάφορες ψυχολογίες της μάθησης (που μελετούν τον ***σπουδαστή*** και τη γνωστική του συμπεριφορά), τα μοντέλα διδασκαλίας και η κοινωνική ψυχολογία (που σχετίζονται με τον ***εκπαιδευτικό*** και την πρακτική του). Τέλος, και κατά κύριο λόγο, πρέπει να ληφθούν υπόψη οι αλληλεπιδράσεις ανάμεσα στις κορυφές του διδακτικού τριγώνου.

3.1.3. Κύριοι ερευνητικοί άξονες στη Διδακτική

Αυτό που ενδιαφέρει περισσότερο στο πλαίσιο της Διδακτικής είναι η μελέτη των διαφόρων τομέων που συνδέουν τις επιμέρους κορυφές του διδακτικού τριγώνου (Κόμης, 2002).

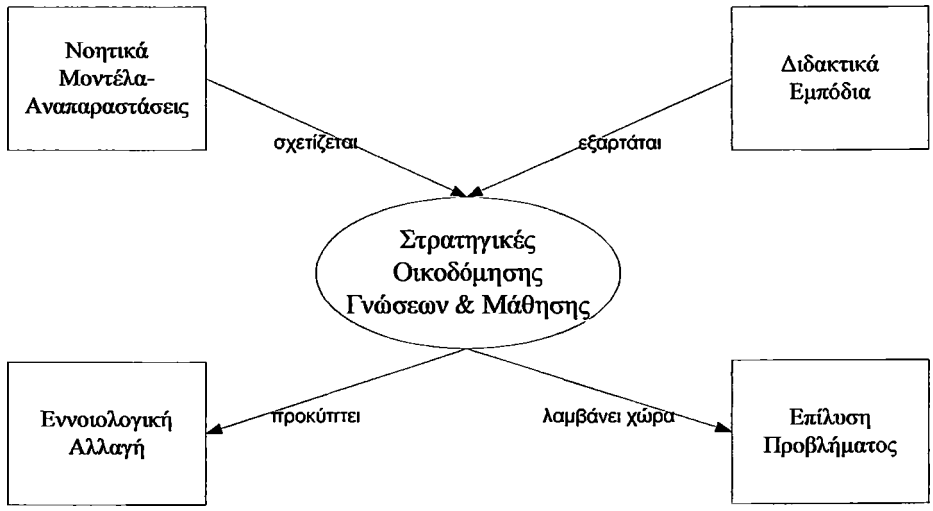
Βασικό πεδίο μελέτης της Διδακτικής συνιστά καταρχήν ο τομέας της ***ανάπτυξης των περιεχομένων*** (δηλαδή η διδακτέα ύλη και το εκπαιδευτικό υλικό). Ο τομέας αυτός μελετάται (Σχήμα 2) κάτω από ποικίλα πρίσματα. Αφενός κάτω από το πρίσμα του *διδακτικού μετασχηματισμού* και των *κοινωνικών πρακτικών αναφοράς* και, αφετέρου,

κάτω από το πρίσμα των διαδικασιών συγκρότησης σχετικών με το *Πλαίσιο Προγράμματος Σπουδών* και το *Αναλυτικό Πρόγραμμα*, όπως επίσης και με το παραγόμενο *διδακτικό υλικό* – δηλαδή, τα βιβλία και εγχειρίδια, καθώς και το *εκπαιδευτικό υλικό και λογισμικό.*



Σχήμα 2 Ανάπτυξη περιεχομένων

Στη συνέχεια, βασικό άξονα μελέτης αποτελεί ο τομέας των ***στρατηγικών της οικοδόμησης των γνώσεων και της κατανόησης των διαδικασιών της μάθησης*** (Σχήμα 3), ο οποίος κατά κανόνα αφορά το πλαίσιο των συνθηκών της σχολής. Τόσο η κατανόηση των στρατηγικών, όσο η μελέτη των διαδικασιών μάθησης, απαιτούν τω μελέτη των αναπαραστάσεων, των διδακτικών εμποδίων, της εννοιολογικής αλλαγής και των διαδικασιών επίλυσης προβλημάτων.

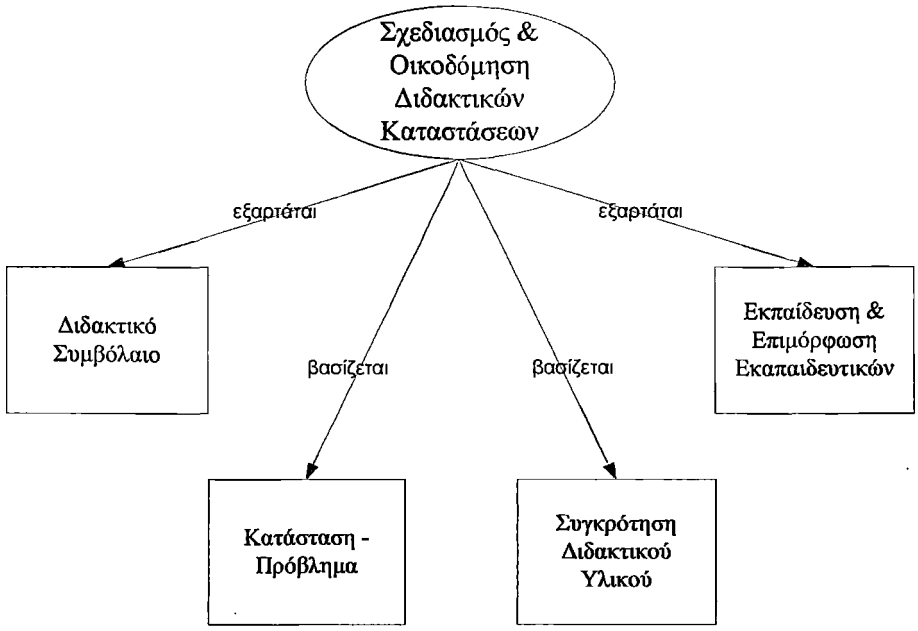


Σχήμα 3 Οικοδόμηση Γνώσεων & Μάθησης

Σημαντικός τομέας είναι, επίσης, αυτός της οικοδόμησης διδακτικών καταστάσεων (Σχήμα 4), που αφορά στο *διδακτικό συμβόλαιο* και στην *κατάσταση – πρόβλημα*, στην *επιμόρφωση των εκπαιδευτικών*, αλλά και στη συγκρότηση *διδακτικού υλικού*.

Ο τομέας των διδακτικών αλληλεπιδράσεων, όπως η *διδακτική βοήθεια*, η *γνωστική και η κοινονικογνωστική σύγκρουση*, καθώς και ο τομέας των χρησιμοποιούμενων μέσων (όπως γλώσσα και τεχνολογικά μέσα ή εργαλεία) που διαμεσολαβούν αυτές τις αλληλεπιδράσεις είναι άλλος ένας σημαντικός άξονας μελέτης της Διδακτικής. Κάτω από το πρίσμα αυτό, η ύπαρξη ενός τεχνολογικού μέσου όπως ο υπολογιστής φαίνεται να δρα καταλυτικά κατά τη διδασκαλία της Πληροφορικής ή κατά τη διδασκαλία με χρήση των Τεχνολογιών της Πληροφορικής και των Επικοινωνιών (Κόμης, 2001).

Ο τομέας αυτός, μάλιστα, είναι ιδιαίτερα σημαντικός στην περιοχή της Διδακτικής της Πληροφορικής, αφού είναι γνωστό ότι κύριο ρόλο στη διδακτική πράξη διαδραματίζει η αλληλεπίδραση του μαθητή με το τεχνικό μέσο και οι παρεμβάσεις που παρέχει ο εκπαιδευτικός.



Σχήμα 4 Οικοδόμηση διδακτικών καταστάσεων

Τέλος, δύο σημαντικοί άξονες μελέτης αποτελούν αφενός τα σύγχρονα προγράμματα σπουδών και αφετέρου οι στάσεις και αντιλήψεις των εκπαιδευτικών Πληροφορικής.

Ο τομέας ανάπτυξης σύγχρονου προγράμματος σπουδών και κατάλληλου εκπαιδευτικού υλικού αποτελεί βασικό πεδίο μελέτης της Διδακτικής. Τόσο ο σχεδιασμός των προγραμμάτων σπουδών, όσο και του εκπαιδευτικού υλικού, πρέπει να λαμβάνουν υπόψη τους και να βασίζονται στα αποτελέσματα των ερευνών που αφορούν όλες τις προηγούμενες κατηγορίες.

Ιδιαίτερα σημαντικός είναι επίσης ο τομέας των στάσεων και των αντιλήψεων των εκπαιδευτικών Πληροφορικής, αλλά και όλων των εκπαιδευτικών που χρησιμοποιούν τις ΤΠΕ στην εκπαιδευτική τους πρακτική, αφού άπτεται άμεσα της οργάνωσης των *διδακτικών παρεμβάσεων* και των *σχολικών πρακτικών*.

3.1.4. Διδακτική της Πληροφορικής: προς ένα πλαίσιο ορισμού

Εδώ και πολλά χρόνια, η θέση της Πληροφορικής στην εκπαίδευση γενικά και πολύ περισσότερο στο ελληνικό εκπαιδευτικό σύστημα καθορίζονταν από έναν εμπειρισμό και αναφορές από άλλους χώρους. Ωστόσο, την τελευταία δεκαετία, και κυρίως κάτω από την επιρροή των *εποικοδομιστικών προσεγγίσεων*, ο χώρος της Διδακτικής της Πληροφορικής τείνει να καθιερωθεί ως αυτόνομο γνωστικό πεδίο, παρά τις ιδιαιτερότητες που άπτονται της ραγδαίας εξέλιξης του λογισμικού και του υλικού αλλά και της θέσης που κατέχει (ή δεν κατέχει) ο προγραμματισμός στα προγράμματα

σπουδών (Κόμης, 2001). Η προβληματική εστιάζεται τόσο στην ανάλυση των περιεχομένων, όσο και στη διασαφήνιση των θεωρητικών αναφορών.

Η ***Διδακτική της Πληροφορικής*** φαίνεται να συγκροτεί το επιστημονικό της πεδίο γύρω από ένα σαφώς καθορισμένο αντικείμενο μελέτης: την παιδαγωγική αξιοποίηση εφαρμογών λογισμικού που έχουν ως κύριο χαρακτηριστικό την ένταξη και την ευρεία χρήση στοιχείων προγραμματισμού (Rouchier et al., 1988 – Bonar et al., 1989).

Η μελέτη αυτή πραγματοποιείται αφενός στο επίπεδο της *δομής* (αντικείμενα, πράξεις, έγκυρες εκφράσεις, κ.λπ.) και αφετέρου στο επίπεδο της *λειτουργίας* (τύπος προβλημάτων για την επίλυση των οποίων μπορεί να χρησιμοποιηθεί το συγκεκριμένο λογισμικό) (Green et al., 1990).

Ορισμένες από αυτές τις εφαρμογές λογισμικού είναι γλώσσες προγραμματισμού με την κλασσική έννοια του όρου όπως, για παράδειγμα, Logo, Pascal, Visual Basic, C++, κ.λπ. Άλλες εφαρμογές λογισμικού δεν ανήκουν στις γλώσσες προγραμματισμού αλλά είναι λογισμικό γενικής χρήσης χωρίς άμεσες εκπαιδευτικές λειτουργίες όπως, για παράδειγμα, επεξεργασία κειμένου, λογιστικό φύλλο, βάση δεδομένων, κ.λπ. (τα οποία ωστόσο εμπερικλείουν και δομικά στοιχεία προγραμματισμού με τη μορφή μακροεντολών).

Η ένταξη τέτοιου τύπου λογισμικού στην εκπαίδευση και στην κατάρτιση μπορεί να αιτιολογηθεί με πολλά επιχειρήματα :

- έχει δημιουργηθεί γι' αυτό το σκοπό (όπως για παράδειγμα οι γλώσσες προγραμματισμού Logo, MicroWorlds, Smalltalk),
- είναι αντικείμενα κατάρτισης (όπως για παράδειγμα το λογισμικό επεξεργασίας κειμένου, οι βάσεις δεδομένων, τα λογιστικά φύλλα, τα συστήματα δημιουργίας γραφικών κ.λπ.),
- συνιστούν *γνωστικά εργαλεία* που ενισχύουν τις ανθρώπινες γνωστικές δεξιότητες κατά τη σκέψη, τη μάθηση, και την επίλυση προβλημάτων (Jonassen, 1995).

Συνεπώς μπορούν να προσεγγίζονται ταυτόχρονα ως αντικείμενα μελέτης και ως τρόποι γνώσης και οικοδόμησης γνώσεων και δεξιοτήτων περισσότερο ή λιγότερο ειδικών.

Με την αποδοχή της παραπάνω προοπτικής οφείλουμε να αναγνωρίσουμε την διττή διάσταση αυτών των εφαρμογών λογισμικού :

- ***οικοδόμηση εννοιών της πληροφορικής*** από το ένα μέρος και

- ***δυνατότητα επεξεργασίας και επίλυσης προβλημάτων*** σχετικών με χώρους εξωτερικούς της Πληροφορικής από το άλλο μέρος.

Τα αποτελέσματα αυτής της γνώσης φαίνονται ιδιαίτερα ενδιαφέροντα τόσο από γνωστική, όσο και από διδακτική πλευρά. Καθένας από τους παραπάνω όρους χωρίζεται με τη σειρά του σε δύο κατευθύνσεις: αφενός αυτήν του αντικειμένου και αφετέρου αυτήν της παρέμβασης του αντικειμένου σε άλλους χώρους γνώσης.

Συνεπώς, βασικό αντικείμενο της Διδακτικής της Πληροφορικής είναι η μελέτη της οικοδόμησης των *γνώσεων* (κυρίως όσον αφορά στις διαχρονικές έννοιες) και της ανάπτυξης των τεχνικών και νοητικών *δεξιοτήτων* από τα υποκείμενα που χρησιμοποιούν υπολογιστές και ασχολούνται με την Πληροφορική (Κόμης, 2001). Οι δεξιότητες αυτές διαπιστώνονται κατά κύριο λόγο στο πλαίσιο *επίλυσης προβλημάτων* με τη χρήση υπολογιστών (Kahnay, 1993).

Με αυτή την έννοια, το αντικείμενο της Διδακτικής της Πληροφορικής ξεπερνά τα στενά όρια της μελέτης των υποκειμένων που βρίσκονται σε διαδικασία μάθησης στο γνωστικό αντικείμενο της Πληροφορικής και επεκτείνεται σε όλα τα πεδία μάθησης που κάνουν χρήση των εφαρμογών των Τεχνολογιών της Πληροφορικής και των Επικοινωνιών.

Κάτω από αυτό το πρίσμα, οι στάσεις και οι αξίες που διαμορφώνονται από τους σπουδαστές όταν χρησιμοποιούν υπολογιστές τόσο εντός σχολής όσο και εκτός σχολής συνιστά επίσης αντικείμενο μελέτης για τους ερευνητές στο χώρο της Διδακτικής της Πληροφορικής.

Η ανάπτυξη δεξιοτήτων πιστοποιείται όταν λαμβάνει χώρα συσχέτιση των γνώσεων πάνω σε μια διαδικασία ή σε ένα αντικείμενο που πρέπει να οικοδομηθεί με τις δυσχέρειες της χρήσης συγκεκριμένων ενεργειών και του προσδιορισμού συγκεκριμένων δομών δεδομένων και διακριτών κανόνων σύνταξης, όπως συμβαίνει κυρίως στον προγραμματισμό.

Όπως προκύπτει και από τη μελέτη της σχετικής βιβλιογραφίας και την μελέτη των προγραμμάτων σπουδών Πληροφορικής, ο χώρος αυτών των γνώσεων, των δεξιοτήτων και των στάσεων μπορεί να περιγραφεί συνοπτικά από τον επόμενο πίνακα (Πίνακας 1).

Γνώσεις	Γνώση των βασικών τμημάτων ενός υπολογιστή και των κύριων περιφερειακών συσκευών. Ικανή χρήση των διαχρονικών εννοιών Πληροφορικής (που αφορούν στο υλικό) και αποστασιοποίηση από τις αλλαγές που προκύπτουν από την τεχνολογική πρόοδο. Γνώση των βασικών στοιχείων επεξεργασίας της πληροφορίας (κωδικοποίηση, αποθήκευση, επεξεργασία) και των εγγενών περιορισμών. Γνώση του ρόλου του λογισμικού συστήματος. Γνώση των βασικών χαρακτηριστικών των αρχείων και δυνατότητα επεξήγησης των βασικών επεξεργασιών που πραγματοποιούνται με αυτά. Γνώση το τι είναι λογισμικό πρόγραμμα. Γνώση των βασικών προγραμματιστικών δομών
Δεξιότητες	Ικανότητα χρήσης εφαρμογών γενικού τύπου όπως λογισμικό συστήματος, επεξεργασία κειμένου, λογιστικό φύλλο, βάση δεδομένων, λογισμικό Διαδικτύου κ.λπ. Ικανότητα χρήσης εφαρμογών λογισμικού ειδικού τύπου που απαντώνται κατά τη διάρκεια των σπουδών, όπως εκπαιδευτικό λογισμικό, λογισμικό προσομοιώσεων, παιγνίδια, κ.λπ. Ικανότητα χρήσης προγραμματιστικών εφαρμογών (κάνω κάτι να κάνει κάτι). Ανάπτυξη δεξιοτήτων ορολογίας (ανάγνωση και κατανόηση καταλόγων πληροφορικής και εγχειριδίων χρήσης).
Στάσεις - Αξίες	Συνειδητοποίηση της ένταξης της Πληροφορικής στο σύγχρονο κόσμο. Αναγνώριση οικονομικών, κοινωνικών και πολιτικών συνθηκών που διαμορφώνει η Πληροφορική.

Πίνακας 1 Ο χώρος γνώσεων, δεξιοτήτων και στάσεων γύρω από την Πληροφορική στο επίπεδο της γενικής παιδείας

Σε αντίθεση με τις Διδακτικές άλλων Επιστημών (κυρίως των Μαθηματικών και των Φυσικών επιστημών), η έρευνα στη Διδακτική της Πληροφορικής έχει γνωρίσει περιορισμένη έκταση μέχρι σήμερα. Το γεγονός αυτό συνδέεται σε μεγάλο βαθμό με την θέση και την έκταση που έχει (ή δεν έχει) η Πληροφορική (και κυρίως ο προγραμματισμός) στα προγράμματα σπουδών. Ενώ κατά την δεκαετία του 1980 τα περισσότερα εκπαιδευτικά συστήματα στις ανεπτυγμένες χώρες εισήγαγαν μαθήματα Πληροφορικής, κυρίως στη δευτεροβάθμια εκπαίδευση, την επόμενη δεκαετία

παρατηρήθηκε σε αρκετές χώρες μια υποχώρηση στον τομέα αυτό προς όφελος της εισαγωγής των Τεχνολογιών της Πληροφορικής και των Επικοινωνιών (ΤΠΕ) ως μέσου διδασκαλίας και μάθησης σε όλα τα μαθήματα.

Στην Ελλάδα, όμως, εξελίσσεται πρακτικά μία αντίστροφη πορεία και η Πληροφορική, από μάθημα γενικής κουλτούρας στο γυμνάσιο κατά τις αρχές της δεκαετίας του 1990, επεκτείνεται και στο ενιαίο λύκειο, στο οποίο θεσμοθετείται και κλάδος «Πληροφορικής και Υπηρεσιών» στην τεχνολογική του κατεύθυνση ενώ στα Τ.Ε.Ε. δημιουργείται τομέας Πληροφορικής και Δικτύων Η/Υ με αρκετά μαθήματα. Παράλληλα εξελίσσεται ένα πρόγραμμα ευρείας κλίμακας αναφορικά με την ένταξη των ΤΠΕ στο πρόγραμμα σπουδών.

Στο πλαίσιο αυτό ανακύπτουν πολλαπλά ερωτήματα για το αναλυτικό πρόγραμμα και την επιμόρφωση των εκπαιδευτικών της Πληροφορικής, τις διδακτικές μεθοδολογίες και τις παιδαγωγικές στρατηγικές, το χρησιμοποιούμενο και το προς ανάπτυξη εκπαιδευτικό υλικό. Οι απαντήσεις στα ερωτήματα αυτά πρέπει να αναζητηθούν μέσα από τα πορίσματα της παιδαγωγικής και της διδακτικής έρευνας.

4. Νοητικά μοντέλα και αναπαραστάσεις της Πληροφορικής

Στόχος της ενότητας αυτής είναι η μελέτη της έννοιας των **αναπαραστάσεων** (representations), που συνιστά κεντρική έννοια στη Διδακτική, καθώς και μια πρώτη ανάλυση των αναπαραστάσεων που σχηματίζουν οι σπουδαστές για τις τεχνολογίες και βασικές έννοιες της Πληροφορικής.

Η έννοια της αναπαράστασης, αφού μελετήθηκε διεξοδικά στα πλαίσια της ψυχολογίας, αποτελεί πλέον πεδίο μελέτης για τη Διδακτική και την Παιδαγωγική. Το ενδιαφέρον των παιδαγωγών επικεντρώνεται βασικά πάνω στις αναπαραστάσεις των παιδιών και είναι το αποτέλεσμα της πρόσφατης ανάπτυξης της Διδακτικής των Επιστημών. Η βαρύτητα και η σημασία αυτών των αναπαραστάσεων, μάλιστα, γίνονται όλο και περισσότερο αισθητές όταν η διδασκαλία στοχεύει στις πιο χαμηλές βαθμίδες εκπαίδευσης.

4.1. Η αναπαράσταση στη γνωστική και την κοινωνική ψυχολογία

Τα τελευταία χρόνια παρακολουθούμε την διείσδυση της ***γνωστικής επιστήμης*** (cognitive science) τόσο στους χώρους της επιστημονικής έρευνας, όσο και στη διανοητική επικαιρότητα (Ausubel, 1968). Η προοδευτική συγκρότηση και ανάπτυξή της βρίσκονται σε στενή σχέση με την εξέλιξη τόσο των ανθρωπιστικών, όσο και των θετικών επιστημών. Πρωταρχικός της στόχος είναι η κατανόηση των λειτουργιών του εγκεφάλου και των διαδικασιών της μάθησης. Σε αυτό το πλαίσιο εγγράφεται και η μελέτη των ***αναπαραστάσεων*** του υποκειμένου που βρίσκεται σε διαδικασία μάθησης, καθώς και ο μετασχηματισμός τους (Vergnaud, 1996).

Η έννοια της αναπαράστασης είναι πρωταρχική στη γνωστική ψυχολογία, αλλά παραμένει αρκετά αμφιλεγόμενη και ασαφής. Στη γνωστική ψυχολογία, κάθε γνωστική διαδικασία συνίσταται από αναπαραστάσεις και από επεξεργασίες.

Κάθε υποκείμενο αναπτύσσει τις γνωστικές του στρατηγικές, οι οποίες εξαρτώνται από τις γνώσεις του, την αναπαράσταση που έχει για τις καταστάσεις, και τους τρόπους συλλογισμού τους οποίους δύναται να θέσει σε λειτουργία. Συνεπώς, οι γνωστικές αυτές στρατηγικές υπόκεινται στις δυσχέρειες αποθήκευσης και ανάκτησης της πληροφορίας (Κόμης & Μιχαϊλίδης, 1996).

Η κοινωνική ψυχολογία προσεγγίζει τις αναπαραστάσεις στο σημείο σύγκλισης του γνωστικού με το κοινωνιολογικό. Οι αναπαραστάσεις μπορούν να θεωρηθούν ως κοινωνικές γνώσεις και παίζουν βασικό ρόλο στην κοινωνικοποίηση του υποκειμένου. Γενικότερα, οι αναπαραστάσεις είναι προϊόντα και ταυτόχρονα διαδικασίες των διανοητικών μας δραστηριοτήτων και έχουν κατά κάποιο τρόπο ως στόχο να καταστήσουν παρόν αυτό που είναι απόν.

Οι αναπαραστάσεις είναι μια ανθρώπινη δραστηριότητα που συνίσταται στην παραγωγή συμβόλων με βασικό χαρακτηριστικό να αντικαθιστούν άλλες (απούσες κατά κανόνα) οντότητες. Το ανθρώπινο πνεύμα είναι επίσης υπόβαθρο αναπαραστάσεων ψυχολογικής υφής, γνωστικών προϊόντων που αντανακλούν αυτά που το άτομο συγκρατεί από την αλληλεπίδρασή του με τον κόσμο.

4.2. Μορφές αναπαραστάσεων

Οι ***αναπαραστάσεις ου σχετίζονται με την εκτέλεση δράσεων*** (enactive representations) ή ***έμπρακτες αναπαραστάσεις*** αποτελούν μια ειδική κατηγορία αναπαραστάσεων. Σε αυτή την περίπτωση, ο Bruner χρησιμοποιεί τον όρο «enactive» που αντιστοιχεί στην πρωταρχική μορφή των αναπαραστάσεων (Bruner, 1966).

Οι ***εικονικές αναπαραστάσεις*** (iconic representations) ή νοητικές εικόνες αντιστοιχούν στις δομές του χώρου, είναι σχετικά ανεξάρτητες της δράσης και σχετίζονται με την οπτική αντίληψη (visual perception). Σε ευρύτερο πλαίσιο, όταν το αυτό που αναπαριστά έχει σχέσει ομοιότητας και αντιστοιχίας με αυτό που αναπαρίσταται, διατηρώντας παράλληλα τα στοιχεία του μοντέλου (του αναπαριστώμενου) καθώς και τις σχέσεις που έχουν μεταξύ τους, μπορούμε να μιλήσουμε για αναλογικές αναπαραστάσεις.

Συνήθως χρησιμοποιείται και η έννοια του ***νοητικού μοντέλου*** (Βοσνιάδου, 1998), η οποία συνιστά αναλογική αναπαράσταση που θεωρείται ότι διατηρεί την δομή του αντικειμένου το οποίο αναπαριστά. Πρόκειται για υψηλού επιπέδου νοητικά κατασκευάσματα, τα οποία έχουν ως σκοπό να αναπαριστούν την δομή αντικειμένων, πεποιθήσεων, ή θεωριών που ίσως δεν έχουν γίνει ποτέ ορατά από το υποκείμενο.

Οι ***προτασιακές αναπαραστάσεις*** (propositional representations) αφορούν τις προτασιακές δομές της γλώσσας, οι οποίες βρίσκονται στη βάση των λειτουργιών επικοινωνίας. Ο J.Bruner αποκαλεί αυτές τις αναπαραστάσεις «συμβολικές» (symbolic). Οι προτασιακές αναπαραστάσεις συνιστούν ***συμβολικές αναπαραστάσεις*** που δεν έχουν εικονική (αναλογική) σχέση με αυτό που αναπαρίσταται.

Κάθε συμβολική αναπαράσταση οικοδομείται κατά κύριο λόγο πολιτισμικά και επιτρέπει στον εκπαιδευόμενο την ευρεία χρησιμοποίηση των αντιληπτικών χαρακτηριστικών του κόσμου, ώστε να αναπτύξει δραστηριότητες κατηγοριοποίησης και εννοιοποίησης (conceptualization) για την καλύτερη επίτευξη των πράξεών του.

4.3. Δομή, περιεχόμενο και ανασχηματισμός των αναπαραστάσεων

Ο ανασχηματισμός των αναπαραστάσεων αποτελείται από τρεις αναπαραστασιακές ομάδες – τρεις βασικοί πυρήνες – γύρω από τους οποίους αναδομούνται οι σχετικές με τις νέες τεχνολογίες της πληροφορικής αναπαραστάσεις.

Αυτές οι τρεις ομάδες είναι οι εξής:

- **η πληροφορία,** που παραπέμπει στο σύνολο των «γνώσεων» πάνω στο αντικείμενο, που του προσδίδουν συγκεκριμένο νόημα,
- **το αναπαραστασιακό πεδίο,** που καθορίζεται από κάθε βασικό πυρήνα, παραπέμπει στον τρόπο με τον οποίο οι σπουδαστές αντιλαμβάνονται το τεχνολογικό αντικείμενο. Το πεδίο αυτό μπορεί να επεξηγηθεί σαν η συνολική δομή που οργανώνει τις «γνώσεις» του υποκειμένου,
- **η στάση,** που εκφράζει το γενικό προσανατολισμό απέναντι στο αντικείμενο της αναπαράστασης. Αυτή προτρέπει σε συγκεκριμένες συμπεριφορές που επιδρούν πάνω στις λογικές χρήσης.

4.4. Παιδαγωγικές και διδακτικές πτυχές των αναπαραστάσεων

Η εκπαιδευτική ψυχολογία και η διδακτική δεν παρέλειψαν να χρησιμοποιήσουν την έννοια της αναπαράστασης, κυρίως λόγω των σοβαρών δυσκολιών που συναντούν οι εκπαιδευτικοί κατά τη διδασκαλία των επιστημονικών μοντέλων και εννοιών.

Ο εκπαιδευτικός οφείλει, εκ των πραγμάτων, να λάβει υπόψη του τις πρότερες γνώσεις των μαθητών του. Είναι όμως γνωστό ότι οι «προεπιστημονικές» πρότερες γνώσεις δεν εξαλείφονται εύκολα από το πνεύμα των υποκειμένων στα οποία απευθύνεται ο επιστημονικός λόγος αλλά, αντιθέτως, συνιστούν σημαντικά γνωστικά αλλά και επιστημολογικά εμπόδια (Bachelard, 1989).

Μέσα από αυτή την προοπτική, πολλοί ερευνητές υποστηρίζουν την άποψη ότι οι σπουδαστές που βρίσκονται σε διαδικασία μάθησης διαθέτουν ***προ-αναπαραστάσεις***, οι οποίες είναι γνώσεις που αποτελούν μέρος του γνωστικού τους υποβάθρου και δεν είναι αποτέλεσμα μιας συστηματικής διδακτικής πράξης.

Αν και ορισμένοι θεωρούν ότι κάποιες προ-αναπαραστάσεις τις οποίες διαθέτει ο σπουδαστής θα μπορούσαν να χρησιμοποιηθούν ως σημείο αγκίστρωσης για την οικοδόμηση νέων εννοιών, άλλοι υπογραμμίζουν τα καταστροφικά αποτελέσματα ορισμένων λανθασμένων προ-αναπαραστάσεων (misconceptions), που είναι ικανές να αντισταθούν σε μια συστηματική διδασκαλία και να οδηγήσουν σε σημαντικές παρανοήσεις.

4.5. Αναπαραστάσεις και τεχνολογίες: διδακτική προσέγγιση

Όπως έχει ήδη τονιστεί, η έννοια της αναπαράστασης είναι άρρηκτα συνυφασμένη με τη διδακτική έρευνα και εμφανίζεται όλο και περισσότερο ως ένα απαραίτητο εργαλείο για τον εκπαιδευτικό που θέλει να κατανοήσει τις πρότερες γνώσεις, τις νοητικές λειτουργίες των μαθητών του και τον τρόπο με τον οποίο αντιλαμβάνονται την τεχνολογική και τη φυσική πραγματικότητα.

Ωστόσο, ιδιαίτερο ρόλο στη μελέτη των αναπαραστάσεων παίζει το επιστημολογικό πλαίσιο αναφοράς που χαρακτηρίζει κάθε γνωστικό αντικείμενο. Οι διάφορες γνώσεις, οι τεχνικές και οι συνεπαγόμενες μέθοδοι στην κατανόηση των εννοιών είναι διαφορετικές σε κάθε γνωστικό αντικείμενο. Συνεπώς, όταν τίθεται το ερώτημα του διδακτικού ρόλου των αναπαραστάσεων μέσα σε ένα ***πλαίσιο διδασκαλίας των τεχνολογιών της Πληροφορικής*** ή ***διδασκαλίας υποβοηθούμενης από τις τεχνολογίες***, θα πρέπει να διασαφηνίσουν ορισμένες ιδιαίτερα σημαντικές ειδικές πτυχές του ζητήματος (Κόμης, 1993 / 2001).

Όσον αφορά τη διδασκαλία της Πληροφορικής, θα ήταν σκόπιμο να αναδυθούν οι αναπαραστάσεις που σχετίζονται με τις ιδιαίτερες έννοιες της Πληροφορικής ως επιστήμης (όπως, για παράδειγμα, η έννοια της μνήμης, η επεξεργασία της πληροφορίας, οι διάφορες προγραμματιστικές δομές, κ.λπ.). Αυτό το ευρύ, πολύ λίγο διερευνημένο μέχρι τώρα, πεδίο σχετίζεται με το πρότυπο της απομονωμένης τεχνικής προσέγγισης για την εισαγωγή των νέων τεχνολογιών και των υπολογιστών στην εκπαιδευτική διαδικασία.

Στα πλαίσια της διδασκαλίας της χρήσης των ΤΠΕ σε όλα τα γνωστικά αντικείμενα του αναλυτικού προγράμματος (πρότυπο της πραγματολογικής προσέγγισης για την ένταξη των ΤΠΕ στην εκπαίδευση) ή κατά τη διδασκαλία των τεχνολογιών κατανεμημένης στα επιμέρους γνωστικά αντικείμενα (πρότυπο της ολοκληρωμένης ή ολιστικής προσέγγισης), θα ήταν πιο χρησιμότερο να μελετηθεί η στάση των σπουδαστών απέναντι στις ίδιες έννοιες της *Πληροφορικής* και των *Τεχνολογιών της Πληροφορίας και των Επικοινωνιών*. Είναι σκόπιμο επίσης να μελετηθεί ο τρόπος με τον οποίο οι σπουδαστές αναπαριστούν τα τεχνολογικά αντικείμενα – τεχνικά μέσα κατασκευασμένα για συγκεκριμένο σκοπό άρα «τελεολογικά αντικείμενα» όπως ο υπολογιστής – με τα οποία βρίσκονται σε αλληλεπίδραση.

Μια τέτοια μελέτη είναι αδιαμφισβήτητα εξαιρετικά επίκαιρη λόγω των σύγχρονων τεχνολογικών αλλαγών, που εξελίσσονται ραγδαία. Είναι λοιπόν επείγον να κατανοήσουμε πως λειτουργούν και αναδομούνται οι αναπαραστάσεις των σπουδαστών πάνω στις τεχνολογίες της Πληροφορικής και πως διασυνδέονται με τις κοινωνικές πρακτικές. Η προσέγγιση αυτή θα μας επιτρέψει επιπλέον να σκεφτούμε με διαφορετικό τρόπο τη χρήση των τεχνολογικών μέσων και των συνακόλουθων παιδαγωγικών μεθόδων, καθώς και τις διαδικασίες της μετάδοσης και οικοδόμησης των γνώσεων.

Οι αναπαραστάσεις που σχηματίζουν οι σπουδαστές με βάση την εμπειρία που έχουν με τις μηχανές και τις διαδικασίες τις οποίες χρησιμοποιούν δουλεύοντας με συστήματα Πληροφορικής, αποτελούν ένα πολύ λίγο διευρυμένο πεδίο έρευνας. Ενδιαφέρον θέμα για μελέτη αποτελεί επίσης η αλληλεπίδραση ανάμεσα σε αυτές τις αναπαραστάσεις και τις πραγματικές χρήσεις που κάνουν οι σπουδαστές.

Σε αυτά τα πλαίσια, η μελέτη των αναπαραστάσεων ενός τεχνολογικού μέσου, οι οποίες σχηματίζονται από τους σπουδαστές κατά τη διάρκεια της μάθησης της χρήσης αυτού του μέσου, εμπεριέχει δύο όψεις: αφενός, την αναπαράσταση που προέρχεται από το ίδιο το μέσο ως ***σύστημα αναφοράς*** μαζί με όλη τη λειτουργία του και αφετέρου, την αναπαράσταση από το ***χειριστή – χρήστη*** (Lévy, 1995). Αυτή η αναπαράσταση, μη μοναδική αφού κάθε χρήστης οικοδομεί τη δική του, πρέπει να συγκριθεί με αυτήν του «ειδήμονα» που υποτίθεται ότι ξέρει να χειρίζεται σωστά το συγκεκριμένο τεχνολογικό μέσο.

Η ανάλυση και η αναδιοργάνωση των αναπαραστάσεων αποκτά και ευρύτερες διαστάσεις, οι οποίες ξεπερνούν τα όρια της εκπαίδευσης και άπτονται ζητημάτων που αφορούν την ίδια τη σύλληψη και τη δημιουργία λογισμικού, τη σχέση ανθρώπου-μηχανής και την εργονομία των διασυνδέσεων (interfaces) – δηλαδή αγγίζει πολλούς σημαντικούς τομείς και τάσεις έρευνας της σύγχρονης Πληροφορικής.

Οι αναπαραστάσεις που σχηματίζει ο σπουδαστής για ένα τεχνολογικό αντικείμενο δεν είναι ποτέ εντελώς «αυθόρμητες», γεγονός που γίνεται όλο και περισσότερο κατανοητό στις σύγχρονες τεχνολογικές κοινωνίες. Οι αναπαραστάσεις αυτές αποτελούν αντικείμενο επεξεργασίας που υλοποιείται κατά ένα μέρος σε άτυπες εκπαιδευτικές καταστάσεις και συνθήκες, μέσα στις οποίες ο σπουδαστής αποκτά εμπειρίες από πολύ νωρίς (Κόμης, 1993/1994/2001).

Η αναπαράσταση πρέπει λοιπόν να θεωρηθεί ως ένα προσωπικό *μοντέλο* ερμηνείας του κόσμου, ένας συνδυασμός επιχειρησιακής φύσης σχημάτων και των μεταβολών που

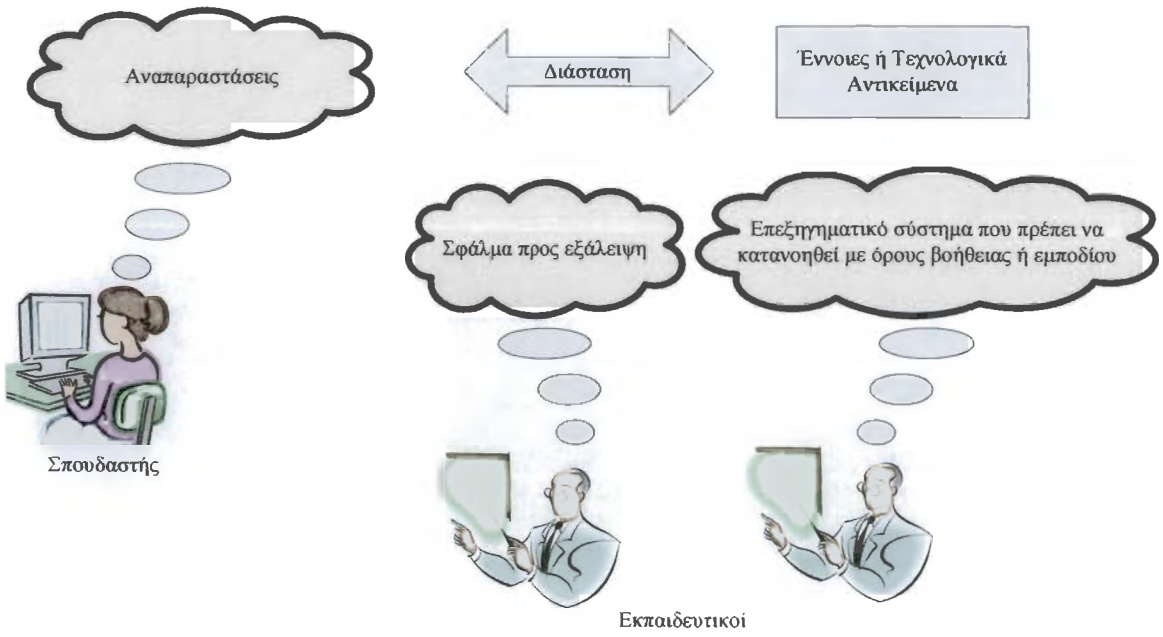
πραγματοποιούνται μέσα σε αυτήν. Πρέπει επίσης να τονιστεί ότι, για τους σπουδαστές, το μοντέλο αυτό βρίσκεται σε κατάσταση διαρκούς ανασχηματισμού.

Οι σπουδαστές, όπως και κάθε υποκείμενο που μαθαίνει, αναπτύσσουν ***σχήματα δράσης***, τα οποία συγκροτούν την συμπεριφορά τους για μια δεδομένη κλάση καταστάσεων.

Μέσα από αυτές τις συνθήκες, η διδασκαλία της Πληροφορικής και των ΤΠΕ (ή με την βοήθεια των ΤΠΕ) δεν μπορεί να μελετηθεί παρά μόνο όταν κατανοήσουμε πως σχηματίζονται οι αναπαραστάσεις τους. Αν και η αναπαράσταση αποτελεί πρωταρχική έννοια, έννοια «νομάδα» που «δανείζεται» από διάφορους επιστημονικούς κλάδους (ψυχολογία, παιδαγωγική, διδακτική, κοινωνιολογία, φιλοσοφία), δεν υπάρχει ομοφωνία σχετικά με τον ορισμό της. Αφού όμως παίζει βασικό ρόλο επικοινωνίας ανάμεσα στο αντικείμενο και στο υποκείμενο, γίνεται απαραίτητη για την κατανόηση της γνωστικής δραστηριότητας του ατόμου.

Είναι συνεπώς απαραίτητο να δοθεί ένας ορισμός της έννοιας της αναπαράστασης. Δεν θα δώσουμε ωστόσο έναν πλήρη ορισμό, περισσότερο θα σκιαγραφήσουμε ένα είδος «λειτουργικού» ορισμού, περιοριζόμενοι στα πλαίσια της Διδακτικής.

Μέσα από αυτό το πρίσμα, η αναπαράσταση εξετάζεται σε σχέση με την παρέκκλιση από την συναφή της έννοια και είναι δυνατόν να προκαλέσει δύο διαφορετικές στάσεις (Σχήμα 5) εκ μέρους του εκπαιδευτικού: Είτε θεωρείται ως ένα λάθος ακόμα και ως παράπτωμα, που πρέπει εξαλειφθεί οπότε, στην περίπτωση αυτή, η διδασκαλία δεν πρέπει να ενδιαφέρεται για τις αναπαραστάσεις, είτε θεωρείται ως ένα επεξηγηματικό σύστημα που οφείλει να γίνει κατανοητό και να αναλυθεί ως εμπόδιο προς υπέρβαση ή ως σημείο στήριξης για την «προσέγγιση» της έννοιας. Εφεξής θα υιοθετούμε την δεύτερη προσέγγιση.



Σχήμα 5 Εκπαιδευτικοί και έννοια της αναπαράστασης

5. Βασικά στοιχεία αλγοριθμικής και προγραμματισμού

5.1. Οι έννοιες του αλγορίθμου και του προγράμματος

Στην ενότητα αυτή θα προσεγγίσουμε με συντομία τις έννοιες του *αλγορίθμου* (algorithm) και του *προγράμματος* (program), δεδομένης της θεμελιώδους σημασίας που έχουν στη διδασκαλία και τη μάθηση του προγραμματισμού. Είναι σκόπιμο να τονίσουμε ότι στη σημερινή εποχή υπάρχουν παντού προγράμματα και αλγόριθμοι. Οι περισσότερες μηχανές, οι υπολογιστές και οι ρομποτικές συσκευές λειτουργούν με προγράμματα, κάποια από τα οποία είναι στοιχειώδη και κάποια άλλα ιδιαιτέρως πολύπλοκα.

Μια από τις πιο βασικές έννοιες του προγραμματισμού είναι η έννοια του ***αλγορίθμου***. Για πολλούς, η έννοια αυτή είναι συνυφασμένη με την επιστήμη της Πληροφορικής και, ειδικότερα, με τον προγραμματισμό. Σύμφωνα με τους Goldschlager και Lister (Goldschlager & Lister, 1996), η έννοια του αλγορίθμου είναι θεμελιώδης στην Πληροφορική και αποτελεί ενοποιό έννοια για όλες τις δραστηριότητες που αντιμετωπίζουν οι επιστήμονες των υπολογιστών.

Η διαδικασία ανάπτυξης αλγορίθμων, που συχνά χαρακτηρίζεται με τον όρο ***αλγοριθμική σκέψη***, αποτελεί θεμελιώδη ανθρώπινη δεξιότητα υψηλού επιπέδου, η οικοδόμηση της οποίας είναι ζητούμενο στα σύγχρονα εκπαιδευτικά συστήματα (Arsac, 1987). Στην εκπαιδευτική διαδικασία, η αλγοριθμική εμφανίζεται αρκετά νωρίς, κυρίως στη μαθηματική εκπαίδευση. Ένα κλασικό παράδειγμα είναι ο γνωστός αλγόριθμος του Ευκλείδη για την εύρεση του μέγιστου κοινού διαιρέτη δύο ακεραίων αριθμών.

Γενικά, με τον όρο ***αλγόριθμος*** εννοούμε μια πεπερασμένη σειρά βημάτων ή ενεργειών που απαιτούνται για την επίλυση ενός δεδομένου προβλήματος. Συνεπώς, ένας αλγόριθμος περιγράφει τη μέθοδο με την οποία μπορεί να διεκπεραιωθεί ένα έργο. Ειδικά στον προγραμματισμό, με τον όρο *αλγόριθμος* αναφερόμαστε σε ένα διατεταγμένο και πεπερασμένο σύνολο από καλώς ορισμένα βήματα για τη διενέργεια μιας διεργασίας, στο τέλος της οποίας – και δεδομένης μιας αρχικής κατάστασης – θα προκύψει μια αντίστοιχη τελική κατάσταση σε πεπερασμένο χρόνο.

Με άλλα λόγια, ένας αλγόριθμος πρέπει να περιγραφεί με ακρίβεια τα δεδομένα που ο ίδιος θα χρησιμοποιήσει και κάθε βήμα του να είναι τελείως καθορισμένο φανερώνοντας τι ακριβώς κάνει. Ο αλγόριθμος πρέπει επίσης να περιγράφει με ακρίβεια τα αποτελέσματα που παράγονται από το ίδιο και να είναι αποτελεσματικός, δηλαδή να δουλεύει και να περιγράφει τα επιθυμητά αποτελέσματα. Πρέπει επίσης πάντοτε η διαδικασία επίλυσης του προβλήματος να περατώνεται ύστερα από ένα πεπερασμένο αριθμό βημάτων.

Ο ρόλος του αλγορίθμου είναι θεμελιώδης για δύο τουλάχιστον λόγους. Αφενός, χωρίς αλγόριθμο δεν μπορεί να υπάρξει *πρόγραμμα* και συνεπώς δεν μπορεί να πραγματοποιηθεί μια εργασία μέσω υπολογιστή. Αφετέρου, οι αλγόριθμοι είναι ανεξάρτητοι τόσο από την γλώσσα προγραμματισμού στην οποία διατυπώνονται, όσο και από τον υπολογιστή ο οποίος τους εκτελεί (Goldschlager & Lister, 1996).

Η διδακτική προσέγγιση της έννοιας του αλγορίθμου συμπεριλαμβάνει απαραιτήτως την ανάλυση μιας σύνθετης εργασίας σε στοιχειώδεις εργασίες, την αναγνώριση των επαναλαμβανόμενων εργασιών, την εκτίμηση της διάρκειας μιας διαδικασίας και την επαλήθευσή του εάν η διαδοχή των στοιχειωδών πράξεων οδηγεί στο αναμενόμενο αποτέλεσμα (Satratzemi, 2001). Η προσέγγιση αυτή σχετίζεται άμεσα τόσο με την προσέγγιση του προγραμματιστή που γράφει ένα *πρόγραμμα*, όσο και γενικότερα με τον τρόπο συλλογισμού κάθε ορθολογικού όντος.

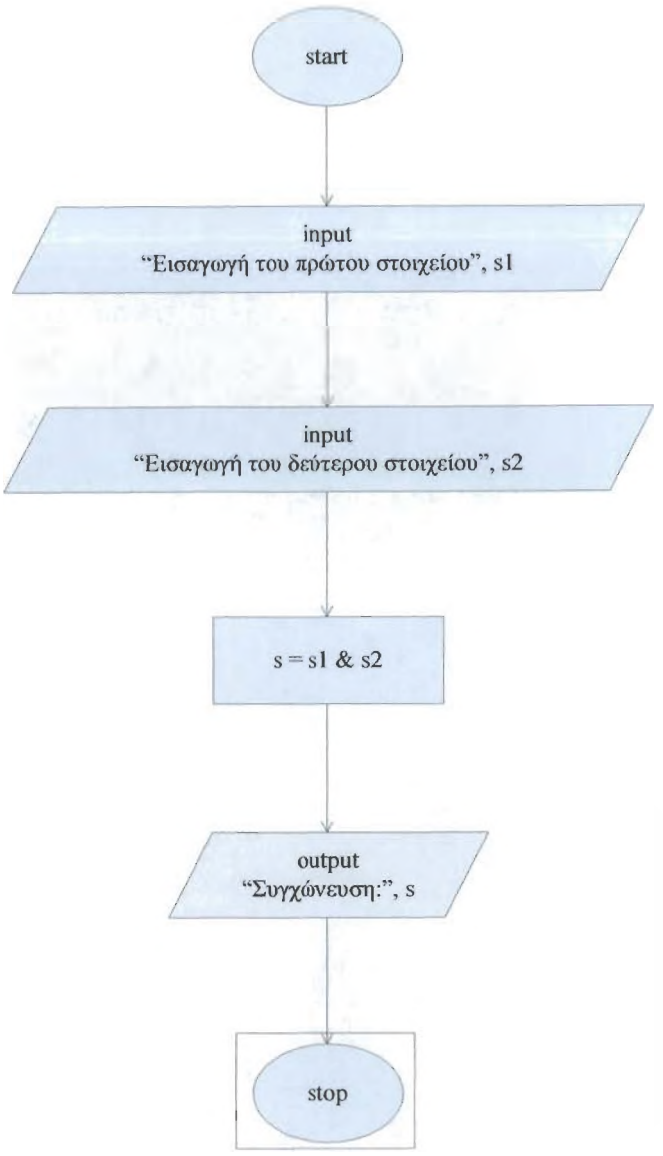
Μια άλλη βασική έννοια στον προγραμματισμό είναι η έννοια του ***προγράμματος***. Το ***πρόγραμμα*** (program), που γράφεται σε μια γλώσσα προγραμματισμού, αποτελείται από μια σειρά εντολών, κατάλληλων για την εκτέλεση ορισμένων λειτουργιών, οι οποίες χρησιμοποιούνται προκειμένου να εξαχθούν κάποια αποτελέσματα, ώστε να αυτοματοποιηθεί μια διαδικασία. Τα προγράμματα γράφονται σε γλώσσες προγραμματισμού (υπάρχουν εκατοντάδες διαφορετικές γλώσσες), όπως η PASCAL, η C++, η Visual Basic, κ.λπ., και χρησιμοποιούνται για την επίλυση διαφόρων προβλημάτων.

Τα προγράμματα που γράφονται σε μια γλώσσα προγραμματισμού, βασίζονται σε έναν ή περισσότερους *αλγόριθμους*. Στο πλαίσιο αυτό, οι αλγόριθμοι αποτελούνται από *ακολουθίες εντολών* που περιγράφουν βήμα προς βήμα μια διαδικασία που τερματίζει την εκτέλεσή της μετά από την ολοκλήρωση πεπερασμένου αριθμού βημάτων και αφορούν την επίλυση ενός συγκεκριμένου προβλήματος (Αβούρης κ.α. 2002). Συνεπώς, ένα πρόγραμμα αποτελείται από μια ακολουθία προτάσεων, κάθε μία από τις οποίες προσδιορίζει συγκεκριμένες λειτουργίες που πρόκειται να εκτελέσει ο υπολογιστής (Goldschlager & Lister, 1996).

Η προγραμματιστική επίλυση ενός προβλήματος συνίσταται στην εύρεση ενός ή περισσότερων αλγορίθμων, βάσει των οποίων θα συνταχθεί το τελικό πρόγραμμα σε μια γλώσσα προγραμματισμού. Συνήθως ένας αλγόριθμος διατυπώνεται αρχικά σε μορφή ανεξάρτητη (π.χ. σε φυσική γλώσσα) από τη σύνταξη μιας συγκεκριμένης γλώσσας προγραμματισμού.

Σήμερα έχουν αναπτυχθεί διάφοροι τρόποι περιγραφής και αναπαράστασης αλγορίθμων. Μια συνηθισμένη, και συγχρόνως πολύ απλή, μέθοδος είναι η περιγραφή του αλγορίθμου με συμβατική γλώσσα και, όπου απαιτείται, η χρησιμοποίηση αριθμητικών πράξεων απλής ή σύνθετης μορφής. Η μορφή αυτή μοιάζει αρκετά με το πρόγραμμα (κώδικα), γι' αυτό και ονομάζεται ***ψευδοκώδικας***.

Μια άλλη μορφή αναπαράστασης αλγορίθμων είναι το ***διάγραμμα ροής*** (που συχνά αποκαλείται – εσφαλμένως – λογικό διάγραμμα). Ένα διάγραμμα ροής απεικονίζει (χρησιμοποιώντας σύμβολα γραφικών με συγκεκριμένη σημασιολογία) τη φύση και τη ροή των βημάτων μιας διαδικασίας (Σχήμα 6).



Σχήμα 6 Διάγραμμα Ροής

Συνεπώς, ο βασικός σκοπός ενός διαγράμματος ροής είναι να αναπαραστήσει τη σχεδίαση ενός αλγορίθμου, απελευθερώνοντας τους προγραμματιστές από συντακτικές και άλλου είδους λεπτομέρειες της γλώσσας προγραμματισμού και επιτρέποντας τους να επικεντρωθούν στο πρόβλημα που πρέπει να επιλυθεί.

Το ισχυρό πλεονέκτημα της χρήσης των διαγραμμάτων ροής έγκειται στη σχηματική αναπαράσταση του αλγορίθμου, η οποία διαθέτει μια οπτική ισχύ που λείπει από τις άλλες μορφές αναπαράστασης (Forouzan, 2003), γι' αυτό το λόγο, είναι πολύ χρήσιμα για τη διδασκαλία της αλγοριθμικής σε σπουδαστές.

Τα διαγράμματα ροής χρησιμοποιήθηκαν για πολύ καιρό για την αλγοριθμική περιγραφή προβλημάτων και για τη διδασκαλία της αλγοριθμικής. Τελευταία δεν χρησιμοποιούνται

ιδιαίτερα, σε πανεπιστημιακό τουλάχιστον επίπεδο, διότι δεν πληρούν τη διατύπωση δομημένων προγραμμάτων (Dahl, Dijkstra, Hoare, 1972 / Μανωλόπουλος & Ζάχαρης, 2002).

Στο Σχήμα 6 παρουσιάζεται το διάγραμμα ροής ενός αλγορίθμου που συνενώνει δύο ομάδες από χαρακτήρες. Η ανάγκη κατανόησης της δομής ενός προγράμματος, είτε είναι γραμμένο σε μορφή ψευδοκώδικα, είτε σε μορφή διαγράμματος ροής, είτε σε μορφή τυπικών εντολών με μια γλώσσα προγραμματισμού, οδήγησε στη δημιουργία της τεχνικής του *δομημένου προγραμματισμού.*

5.2. Η έννοια του δομημένου προγραμματισμού

Ο ***δομημένος προγραμματισμός*** (Dijkstra, 1976) αποτελεί βασική έννοια στο πλαίσιο της διδασκαλίας του προγραμματισμού, κυρίως όσον αφορά το διαδικαστικό (procedural) προγραμματιστικό παράδειγμα. Η έννοια αυτή χρησιμοποιήθηκε για πρώτη φορά στα μέσα της δεκαετίας του 1960, αλλά καθιερώθηκε ως τεχνική προγραμματισμού προς το τέλος τα δεκαετίας του 1970.

Ένα από τα σημαντικά θέματα που προσπάθησε να μελετήσει η προσέγγιση του δομημένου προγραμματισμού ήταν η συγγραφή προγραμμάτων που δεν περιέχουν την εντολή **GOTO**, η οποία οδηγεί σε διακλαδώσεις του προγράμματος που είναι πολύ δύσκολο να ελεγχθούν στη συνέχεια.

Αυτό οδηγεί αναπόφευκτα σε προγραμματιστικά σφάλματα και προκαλεί μεγάλες δυσκολίες στη συντήρηση των προγραμμάτων. Από την αντίστοιχη έρευνα προέκυψε ότι εντολές τέτοιου τύπου μπορούν να αποφευχθούν με την χρήση κατάλληλων εντολών βρόχων (Dahl, Dijkstra, Hoare, 1972 / Μανωλόπουλος & Ζάχαρης, 2002).

Στο πλαίσιο αυτό, στόχος του δομημένου προγραμματισμού είναι να αλλάξει τη διαδικασία του προγραμματισμού από μια επίπονη λειτουργία *δοκιμής και λάθους* σε μια ποιοτική και ελεγχόμενη λειτουργία. Προϋποθέτει την δομημένη σχεδίαση και τον έλεγχο ενός δομημένου προγράμματος, το οποίο αποτελείται από ανεξάρτητα μέρη (άλλα μικρότερα προγράμματα που συνήθως αποκαλούνται διαδικασίες ή ρουτίνες), με βάση ένα προκαθορισμένο σχέδιο.

Με άλλα λόγια, δομημένος προγραμματισμός είναι η τεχνική όπου οι εντολές είναι οργανωμένες με τρόπο ώστε να ελαχιστοποιούνται τα λάθη και οι παρανοήσεις (Μανωλόπουλος & Ζάχαρης, 2002).

Η κύρια ιδέα στην προσέγγιση αυτή είναι η χρησιμοποίηση βασικών δομών για τη δημιουργία πολύπλοκων προγραμμάτων και η αποφυγή αλμάτων μέσα στον κώδικα με χρήση εντολών τύπου **GOTO**. Η κατανόηση ενός δομημένου προγράμματος είναι πολύ πιο εύκολη από την κατανόηση προγραμμάτων που είναι γραμμένα με διαφορετικό τρόπο.

Η κατανόηση αυτή διευκολύνει τον έλεγχο του κώδικα του προγράμματος και, συνεπώς, ελαττώνει το χρόνο για τις δοκιμές του προγράμματος και τον εντοπισμό των λαθών. Επιπρόσθετα, δεδομένου ότι ένα δομημένο πρόγραμμα διαβάζεται εύκολα και αποτελείται από σαφώς καθορισμένα τμήματα, έχει και εύκολη συντήρηση, ακόμα και από προγραμματιστές που δεν έχουν συμμετάσχει στη σύνταξή του.

5.3. Βασικές δομές δομημένου προγραμματισμού

Όπως περιγράψαμε στην προηγούμενη ενότητα, ο δομημένος προγραμματισμός είναι μία τεχνική προγραμματισμού στην οποία η δομή του προγράμματος, δηλαδή τα επιμέρους τμήματα από τα οποία αποτελείται και η σύνδεσή τους, γίνεται κατά το δυνατό ξεκάθαρη. Η τεχνική αυτή, όπως είδαμε, προκύπτει από την ανάγκη συγγραφής κώδικα κατανοητού από άλλους προγραμματιστές και την ευκολία κατά τη διαδικασία αποσφαλμάτωσης (debugging) του προγράμματος. Όταν πρόκειται για προγράμματα επαγγελματικής χρήσης, η δομημένη οργάνωσή τους καθιστά ευκολότερη τη διαδικασία συντήρησης ή τροποποίησης.

Πρόκειται για μια τεχνική προγραμματισμού που χρησιμοποιείται αφενός για την δημιουργία «καλών» (ή τουλάχιστον ευανάγνωστων) προγραμμάτων και αφετέρου για τη διδασκαλία και την κατανόηση του προγραμματισμού.

Ο στόχος αυτός, μάλιστα, είναι εφικτός με τον περιορισμό των λογικών δομών προγραμματισμού σε τρεις βασικούς τύπους, οι οποίοι είναι επαρκείς για την κατασκευή οποιουδήποτε αλγορίθμου:

- την ***ακολουθία*** ή ***διαδοχή*** (sequence),
- την ***επιλογή*** (selection) ή ***έλεγχο*** ή ***απόφαση*** (decision)

6. Απαραίτητες ικανότητες για τη μάθηση προγραμματισμού

6.1. Ο προγραμματισμός ως πρόσκτηση μιας ικανότητας

Στο πλαίσιο μιας εποικοδομητικής προσέγγισης της διδασκαλίας και της μάθησης του προγραμματισμού, το κύριο ζητούμενο στην τάξη της Πληροφορικής δεν είναι απλώς να διδαχθούν η σύνταξη και οι εντολές μιας γλώσσας προγραμματισμού. Αντίθετα, το ζητούμενο είναι η βοήθεια που πρέπει να δοθεί στους μαθητές ώστε να οικοδομήσουν τα απαραίτητα νοητικά πλαίσια για να εξασκήσουν προγραμματιστικές δραστηριότητες (Pair, 1988).

Κάτω από το πρίσμα αυτό, η διδακτέα ύλη (με τη μορφή εντολών και συντακτικών αρχών μιας γλώσσας προγραμματισμού) δεν μπορεί να συνιστά το μοναδικό σημείο εκκίνησης, αλλά πολύ σημαντικό ρόλο επίσης παίζουν τα *νοητικά μοντέλα* και οι *αναπαραστάσεις* που διαθέτει ο σπουδαστής, τα οποία και πρέπει να γνωρίζει ο εκπαιδευτικός ώστε να τα χρησιμοποιήσει ως σημείο εκκίνησης για τη διδασκαλία του.

Στην περίπτωση αυτή, κύριο μέλημα της διδασκαλίας συνιστά η εξέλιξη αυτών των νοητικών μοντέλων με στόχο να πλησιάσουν, όσο γίνεται, αυτά των ειδικών. Η αναπαράσταση που διαθέτει ο σπουδαστής για τη συγκεκριμένη δραστηριότητα που ασκεί πρέπει να έρθει σε αντιπαράθεση με εμπειρίες που πιθανόν να την επιβεβαιώσουν, να την εμπλουτίσουν ή – όπως γίνεται συνήθως – να την θέσουν σε αμφισβήτηση. Στα πλαίσια αυτά, η μάθηση είναι μια συνεχής διεργασία *εννοιολογικών αλλαγών* εκ μέρους του σπουδαστή. Συνεπώς, ο εκπαιδευτικός πρέπει, στα πλαίσια της διδακτικής πράξης, να προτείνει δραστηριότητες οι οποίες θα ευνοούν τις διαδικασίες ανάπτυξης *γνωστικών συγκρούσεων* και *εννοιολογικών αλλαγών.*

Η μάθηση του προγραμματισμού μπορεί να θεωρηθεί ότι αφορά την οικοδόμηση μιας ***ικανότητας*** (competence) υψηλού επιπέδου, δηλαδή, ενός συνόλου από γνώσεις και δεξιότητες με ιδιαίτερο πεδίο εφαρμογής, μιας ικανότητας με την οποία καμία άλλη ανθρώπινη ικανότητα πριν την εμφάνιση της Πληροφορικής δεν μπορεί να συγκριθεί. Η ανάπτυξη της ικανότητας αυτής έχει σαφή και ιδιαίτερα σημαντική παιδαγωγική διάσταση, κυρίως όσον αφορά τις διαδικασίες ανάλυσης προβλημάτων ή καταστάσεων που προηγούνται της συγγραφής ενός προγράμματος σε μια γλώσσα προγραμματισμού. Συνεπώς μπορεί να ενταχθεί σε μια παιδαγωγική προβληματική που ευνοεί την ανάπτυξη της λογικής σκέψης.

Στο πλαίσιο αυτό, η μάθηση του σχεδιασμού δράσεων και, στη συνέχεια, η υλοποίησή τους (μέσω μιας μηχανής) συνιστά μια νοητική δεξιότητα υψηλού επιπέδου – συνεπώς και δυνητικό αντικείμενο της εκπαιδευτικής διαδικασίας – που εντάσσεται στη μεγάλη κατηγορία έργων τα οποία οι ψυχολόγοι ονομάζουν ***επίλυση προβλημάτων*** (problem solving).

Οικοδόμηση ικανοτήτων

Για πάρα πολλά χρόνια, η οικοδόμηση ικανοτήτων (με έμφαση στις τεχνικές γνώσεις και δεξιότητες) ήταν μια όχι ιδιαίτερα αναγνωρισμένη πτυχή στα διάφορα εκπαιδευτικά συστήματα. Δεν είναι τυχαίο ότι η τεχνική εκπαίδευση θεωρείτο (και από πολλούς θεωρείται ακόμα και σήμερα) ως μια εκπαίδευση δεύτερης επιλογής. Η ανάπτυξη όμως των ικανοτήτων συνιστά βασικό χαρακτηριστικό της ανθρώπινης δραστηριότητας και επιτυγχάνεται εντός ή, κυρίως, εκτός σχολείου (Pair, 1988). Τέτοιου τύπου δραστηριότητες είναι για παράδειγμα η ανάγνωση, ο αλγεβρικός υπολογισμός, η απόδειξη θεωρημάτων, αλλά και η οδήγηση.

Στις ικανότητες αντιστοιχούν συνήθως γνώσεις όπως: η προφορά της λέξης «Αθήνα», η ανάλυση του διωνύμου $(\alpha - \beta)^2 = \alpha^2 - 2\alpha\beta + \beta^2$, το ότι δύο συμπληρωματικές γωνίες έχουν άθροισμα 90^o, ο οδικός κώδικας, κ.λπ. Η κατάκτηση των προηγούμενων γνώσεων, ωστόσο, δεν συνεπάγεται αυτόματα την άσκηση των αντίστοιχων ικανοτήτων (για παράδειγμα η γνώση ενός κανόνα δεν συνεπάγεται αυτόματα τη δυνατότητα εφαρμογής του σε διάφορα πλαίσια). Ιδιαιτέρως στον προγραμματισμό, μάλιστα, το ζητούμενο είναι η μάθηση της προγραμματιστικής διαδικασίας ή, με άλλα λόγια, η οικοδόμηση μιας προγραμματιστικής μεθόδου.

Πώς όμως αποκτάται μια ικανότητα; Πώς προσδιορίζεται μια ικανότητα στα πλαίσια του προγραμματισμού και πώς μαθαίνουν οι σπουδαστές να λειτουργούν σε προγραμματιστικά περιβάλλοντα; Σημαντικό ρόλο παίζει εδώ ή έννοια του νοητικού πλαισίου. Κατά την εκμάθηση του προγραμματισμού, ένα νοητικό πλαίσιο μπορεί να αντιστοιχιστεί σε ένα ***Σύστημα Αναπαράστασης και Επεξεργασίας***, το οποίο χαρακτηρίζει ένα ενδιάμεσο στάδιο από το οποίο περνά ο σπουδαστής κατά την απόκτηση μιας δεξιότητας (Hoc, 1987).

Για τη λειτουργία του νοητικού πλαισίου, φαίνεται απαραίτητη η ύπαρξη μιας αναπαράστασης, ενός μοντέλου, μιας έννοιας για το αποτέλεσμα που πρέπει να παραχθεί. Παράδειγμα αποτελεί ο κώδικας ενός προγράμματος σε μια γλώσσα

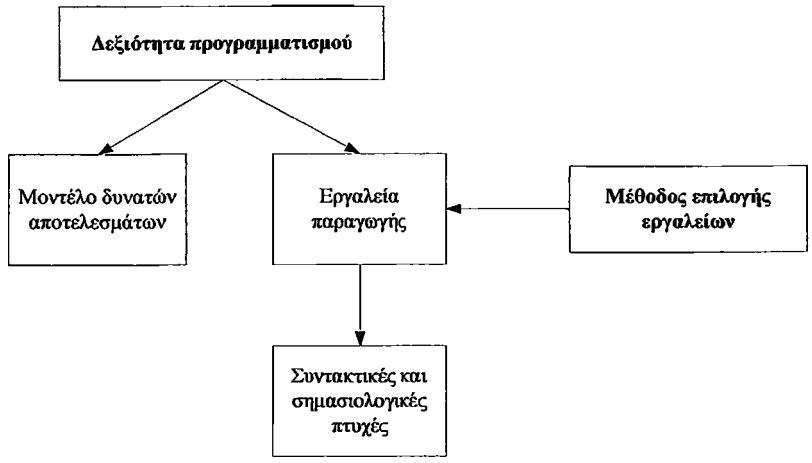
προγραμματισμού (για παράδειγμα σε Logo, Pascal, ή C++), ο οποίος γίνεται αντιληπτός ως μία ακολουθία από εντολές ή ως ένα «λογικό διάγραμμα» (στον προγραμματισμό χρησιμοποιούμε τα *διαγράμματα ροής*) ή ως μια ακολουθία από γλωσσικές οδηγίες (όπως έχουμε δει, η δραστηριότητα αυτή στον προγραμματισμό ονομάζεται *σύνταξη ψευδοκώδικα*).

Αναγκαία είναι επίσης η ύπαρξη και η χρήση εργαλείων (νοητικών ή υλικών) για την παραγωγή του αποτελέσματος μέσω διαδοχικών μετασχηματισμών. Το ζεύγος *νοητικό μοντέλο του αποτελέσματος – εργαλεία παραγωγής του* μπορεί να ονομαστεί ***ικανότητα*** (competence) και δεν διαφέρει πολύ από τη σημασία που αποδίδουν οι γλωσσολόγοι στην ειδική ανθρώπινη δεξιότητα που επιτρέπει την δημιουργία φράσεων σε μια γλώσσα. Σε αρχικό στάδιο, η μάθηση μιας γλώσσας προγραμματισμού μπορεί να θεωρηθεί ως μάθηση μιας ξένης γλώσσας και μπορεί συνεπώς να μελετηθεί με τις μεθόδους που χρησιμοποιούν στις περιπτώσεις αυτές οι γλωσσολόγοι. Στο πλαίσιο αυτό, η ***προγραμματιστική ικανότητα*** παρουσιάζει δύο πτυχές: μία υλική ή ***συντακτική*** (ένα πρόγραμμα σε Pascal ή Visual Basic θεωρούμενο ως μία ακολουθία από εντολές και οικοδομούμενο σταδιακά με προσθήκη νέων εντολών) και μία ***σημασιολογική***.

Η σημασιολογική πτυχή σχετίζεται με τη λειτουργία που πρέπει να εκπληρωθεί, με το νόημα του προς οικοδόμηση αποτελέσματος. Η πτυχή αυτή αιτιολογεί και τις διαδοχικές αλλαγές που απαιτούνται για την επίτευξη του αποτελέσματος. Σημαντικό ρόλο παίζει επίσης (από τη στιγμή που δεν πρόκειται για μια καθαρά αντανακλαστική δεξιότητα) η γνώση για το ποιο εργαλείο πρέπει να χρησιμοποιηθεί σε κάθε στάδιο.

Συμπερασματικά, τα νοητικά πλαίσια που συνιστούν τα διαδοχικά στάδια της μάθησης του προγραμματισμού (Pair, 1988) μπορούν να θεωρηθούν ως (Σχήμα 9):

- μια ***δεξιότητα*** που σχηματίζεται από ένα μοντέλο των δυνατών αποτελεσμάτων και των εργαλείων παραγωγής μαζί με τις συντακτικές και σημασιολογικές πτυχές τους,
- μια ***μέθοδος*** για την επιλογή των εργαλείων.



Σχήμα 9 Νοητικό πλαίσιο μάθησης προγραμματισμού

Μέσα στο πλαίσιο αυτό, ο προγραμματισμός μπορεί να αναλυθεί ή με όρους εξειδίκευσης του υποκειμένου που προγραμματίζει (Ehrlich and Soloway, 1984) ή με όρους των ιδιοτήτων του χώρου των γνώσεων (Perkins et al., 1988), ή και με τα δύο.

Στην πρώτη περίπτωση ορίζεται ένα μοντέλο της λειτουργίας του ειδικού, με το οποίο συγκρίνεται η δραστηριότητα των αρχαρίων. Συνεπώς, ακολουθούνται οι παραδοχές του ρεύματος της γνωστικής ψυχολογίας στο πλαίσιο του οποίου αναπτύχθηκε η *Τεχνητή Νοημοσύνη.*

Στη δεύτερη περίπτωση λαμβάνεται υπόψη το γεγονός ότι η γνώση είναι ιστορικά και κοινωνικά οικοδομημένη και ότι σχηματίζεται από συμβολικές αναπαραστάσεις, οι οποίες αφενός συνιστούν εργαλεία για την επίλυση προβλήματος και αφετέρου πληρούν μια επικοινωνιακή λειτουργία. Η προσέγγιση αυτή εγγράφεται στο ψυχολογικό πλαίσιο που προσδιορίζεται από τον εποικοδομισμό.

Κάποιοι ερευνητές χρησιμοποιούν μια μοντελοποίηση της εξειδίκευσης του ειδικού ως πλαίσιο αναφοράς για να αναλύσουν τα λάθη και τις λαθεμένες αντιλήψεις των σπουδαστών (du Boulay, 1986 / Spohrer & Soloway, 1989) και να καθορίσουν το πεδίο μάθησης του προγραμματισμού. Άλλες προσεγγίσεις θεωρούν τον προγραμματισμό ως έναν επιστημονικό χώρο με τις δικές του έννοιες, διαδικασίες, σύστημα συμβολισμού και ειδικά εργαλεία.

Σε κάθε περίπτωση, πρωτεύοντα ρόλο για την ορθή ανάπτυξη προγραμμάτων παίζει η οικοδόμηση και ο χειρισμός εκ μέρους των σπουδαστών βασικών προγραμματιστικών δομών και εννοιών όπως η μεταβλητή, η ακολουθιακή δομή, η δομή ελέγχου, η επανάληψη, η διαδικασία και η αναδρομική διαδικασία, με τις οποίες ασχολούμαστε σε αυτή την ενότητα. Για να γίνει όμως αντιληπτό πως οικοδομούν οι μαθητές τις έννοιες

αυτές απαιτείται ένα πλαίσιο ανάλυσης, το οποίο στηρίζεται πάνω στην εννοιολογική θεώρηση του προγραμματισμού.

Στο πλαίσιο αυτό, που εξετάζεται στη συνέχεια, αναλύεται η πρόσκτηση των γνώσεων συναρτήσει των προβλημάτων προγραμματισμού διαφόρων επιπέδων πολυπλοκότητας τα οποία μπορούν να προσεγγιστούν και να επιλυθούν από τον σπουδαστή (Pair, 1990).

6.2. Ένα μοντέλο μάθησης του προγραμματισμού

Στην ενότητα αυτή θα παρουσιαστεί ένα μοντέλο μάθησης του προγραμματισμού συμβατό με το προηγούμενο εννοιολογικό πλαίσιο. Το μοντέλο αυτό ανέπτυξε ο C. Pair (Pair, 1988 / Pair, 1990). Σύμφωνα με αυτό το μοντέλο, η μάθηση του προγραμματισμού εξελίσσεται μέσα από μια σειρά διαδοχικών νοητικών σταδίων (Πίνακας 2), από τα οποία περνά υποχρεωτικά ό σπουδαστής που μαθαίνει προγραμματισμό. Η διαδικασία που περιγράφει το μοντέλο είναι εξελικτική και θεωρεί απαραίτητη προϋπόθεση την οικοδόμηση ενός σταδίου ώστε να είναι εφικτό τα πέρασμα στο επόμενο στάδιο.

Πρόκειται για ένα μοντέλο που δανείζεται εμφανώς την ιδέα του Piaget για τα στάδια της νοητικής ανάπτυξης των παιδιών, χωρίς φυσικά να έχει άμεση σχέση με αυτό. Σημαντικό ρόλο στο πλαίσιο του μοντέλου παίζουν επίσης οι προτεινόμενες διδακτικές δραστηριότητες, που είναι σε θέση να βοηθήσουν στη γνωστική εξέλιξη του σπουδαστή για το πέρασμα από το ένα νοητικό πλαίσιο στο επόμενο.

1. Κάνω κάτι με μια μηχανή (προ-προγραμματιστικό στάδιο)
2. Κάνω να κάνει μια μηχανή κάτι (συγγραφή απλού κώδικα)
3. Περιγράφω ένα σύνολο υπολογισμών
4. Περιγράφω το πρόγραμμα
5. Αντιλαμβάνομαι και περιγράφω με δομημένο τρόπο
6. Εισάγω πληροφορικές μεταβλητές
7. Υπολογίζω πάνω σε «αφηρημένα» αντικείμενα
8. Εφευρίσκω αλγορίθμους

Πίνακας 2 Στάδια μάθησης του προγραμματισμού

Τα στάδια αυτά (Pair, 1988), τα οποία συνιστούν επιμέρους ικανότητες υψηλού επιπέδου με κλιμακωτή δυσκολία, περιγράφονται αναλυτικά στη συνέχεια. Παράλληλα, όπου κρίνεται απαραίτητο, προτείνονται κάποιες διδακτικές παρεμβάσεις για την επίτευξή τους.

Κάνω με μια μηχανή κάτι (προ-προγραμματιστικό στάδιο)

Συνήθως η προϋπάρχουσα εμπειρία ενός αρχάριου προγραμματιστή, η οποία και συνιστά ένα από τα βασικά προαπαιτούμενα για τη μάθηση του προγραμματισμού, είναι αυτή της χρήσης ενός υπολογιστή για την υλοποίηση κάποιου έργου με άμεσες εντολές: για παράδειγμα, η χρήση επεξεργασίας κειμένου ή η χρήση λογισμικού για γραφικά, αλλά και η εκτέλεση εντολών σε άμεσο προγραμματιστικό περιβάλλον, χωρίς μετάφραση (compilation) ή διερμηνεία (interpretation), όπως μπορεί να γίνει στο περιβάλλον του *commander* της γλώσσας προγραμματισμού Logo. Η γλώσσα Logo διαθέτει διερμηνευτή, αλλά επιτρέπει και την εκτέλεση – δοκιμή κάθε εντολής ξεχωριστά ώστε ο σπουδαστής να έχει άμεση γνώση για το αποτέλεσμά της. Πρόκειται για την επίτευξη ενός αποτελέσματος συγκεκριμένου τύπου, π.χ. ενός κειμένου, ενός γεωμετρικού σχήματος κ.λπ.

Η δεξιότητα είναι προσανατολισμένη προς αυτού του τύπου τα αποτελέσματα: έννοια της παραγράφου, της ενότητας, της μορφοποίησης κειμένου, εργαλεία για την δημιουργία μιας γραμμής, μετακίνηση ενός μολυβιού. Από τη στιγμή που υπάρχει μία ισχυρή αντιστοιχία ανάμεσα στα χρησιμοποιούμενα εργαλεία και τις εντολές που πρέπει να δοθούν στον υπολογιστή, η μέθοδος συνίσταται στη μεταφορά αυτών που γίνονταν με το χέρι (ή με τη γραφομηχανή ή με διάφορα άλλα μη υπολογιστικά εργαλεία).

Η ενίσχυση τέτοιου τύπου προ-προγραμματιστικών δραστηριοτήτων είναι ιδιαίτερα σημαντική. Χαρακτηριστικό παράδειγμα αποτελεί η γλώσσα Logo, η οποία προσφέρει επίσης και ένα κατάλληλο διδακτικό πλαίσιο για την οικοδόμηση αυτού του σταδίου – κυρίως με τη δυνατότητα που παρέχει για άμεση εκτέλεση των εντολών. Αντίστοιχα πλαίσια μπορούν να προσφέρουν και τα προγράμματα γενικής χρήσης όπως η επεξεργασία κειμένου ή το λογιστικό φύλλο.

Κάνω να κάνει μια μηχανή κάτι

Όπως αναφέρει ο Duchâteau (Duchâteau, 2002), προγραμματίζω σημαίνει περιγράφω τον τρόπο εκτέλεσης εντολών από κάποιον άλλο (συνήθως τη μηχανή). Το πέρασμα από

το «κάνω» στο «κάνω να κάνει» συνιστά την ουσία της αλγοριθμικής προσέγγισης καθώς και της προγραμματιστικής ικανότητας. Αυτό συνιστά και την ιδιαιτερότητα της προγραμματιστικής δραστηριότητας: όταν προγραμματίζει κάποιος το πρόβλημα δεν έγκειται στο ίδιο το έργο που αποτελεί το σημείο εκκίνησης του προγραμματισμού, αλλά στην αναγκαιότητα να σχεδιάσει την εκτέλεση του έργου σε μεταγενέστερο χρόνο από μια εκτελεστική μηχανή – στην περίπτωσή μας, από τον υπολογιστή (Duchâteau, 1992).

Στο πλαίσιο αυτό, το δεύτερο στάδιο ανάπτυξης προγραμματιστικών δεξιοτήτων αφορά την προετοιμασία και την υλοποίηση μιας εργασίας που δεν έχει άμεσα αποτελέσματα (συνήθως στο στάδιο αυτό ο προγραμματιστής δεν είναι απαραίτητο να διαθέτει υπολογιστή) αλλά στοχεύει στη διατήρηση των εντολών με τρόπο ώστε να επιτευχθεί αργότερα ένα αποτέλεσμα. Στο σημείο αυτό παράγεται ο κώδικας ενός προγράμματος, του οποίου η σημασιολογία αφορά έναν υπολογισμό. Το εργαλείο παραγωγής συνίσταται στη διαδοχική προσθήκη εντολών και η μέθοδος παραγωγής είναι πάντα η τοποθέτηση του προγραμματιστή στη θέση του υπολογιστή.

Κατά το στάδιο αυτό υπάρχει σημαντική εννοιολογική πρόοδος σε σχέση με το προηγούμενο στάδιο, η οποία αφορά κυρίως τον τύπο της προς επίτευξη παραγωγής: ένα κείμενο (ο κώδικας του προγράμματος) που περιγράφει έναν υπολογισμό, ο οποίος θα οδηγήσει (σε μεταγενέστερο χρόνο) στο επιθυμητό αποτέλεσμα. Στα εισαγωγικά εγχειρίδια προγραμματισμού, για να προσεγγιστεί η έννοια του προγράμματος, γίνεται συχνά αναφορά στη έννοια της «συνταγής». Η μέθοδος αυτή, που χρησιμοποιεί την έννοια της αναλογίας, δεν είναι πάντοτε αποτελεσματική και μπορεί να οδηγήσει σε παρανοήσεις (Pair, 1988).

Περιγράφω ένα σύνολο υπολογισμών

Στο προηγούμενο στάδιο, ένα πρόγραμμα περιγράφει ένα μοναδικό υπολογισμό. Στο παρόν στάδιο εξετάζεται η περίπτωση όπου το πρόγραμμα περιγράφει περισσότερους (ακόμα και άπειρους) υπολογισμούς ως συναρτήσεις των δεδομένων του. Η εισαγωγή αυτής της προσέγγισης μπορεί να γίνει βάζοντας τους σπουδαστές να διαβάσουν ήδη έτοιμα προγράμματα, τα οποία περιγράφουν ένα σύνολο υπολογισμών πάνω σε κάποια δεδομένα και χρησιμοποιούν *επαναληπτικές* και *υπό συνθήκη* δομές. Σε περίπτωση που ο σπουδαστής διαθέτει το αντίστοιχο μαθητικό υπόβαθρο μπορεί επίσης να γίνει η εισαγωγή της έννοιας της *παραμέτρου* μέσα σε *διαδικασίες* (procedures).

Στο στάδιο αυτό αλλάζει ο *τύπος παραγωγής*: η αλλαγή αυτή δεν αφορά τόσο το επίπεδο της σύνταξης αλλά το επίπεδο της σημασιολογίας. Το εργαλείο που επιτρέπει να προστεθεί μια αλληλεπίδραση έχει επεκταθεί, αφού διαθέτει πλέον στο εύρος του την *υπό συνθήκη* και την *επαναληπτική εντολή*. Σε πρώτο επίπεδο, η μέθοδος παραμένει η ίδια και συνίσταται στη διαδοχική (σειριακή) αντιμετώπιση του συνόλου των υπολογισμών, με μεταφορά στο πρόγραμμα αυτών που θα γίνονταν με το χέρι. Η προσέγγιση αυτή όμως είναι εφικτή μόνο σε απλές περιπτώσεις:

- Για την υπό συνθήκη εντολή, αρκεί μια μόνο εντολή σε κάθε περίπτωση ή ένας μικρός αριθμός εντολών με τρόπο ώστε να μην ξεφύγουμε από τη σειριακή προσέγγιση.
- Για τις επαναλήψεις, μια μορφή του τύπου

  ... να ξαναρχίσει αυτό το μέρος του προγράμματος που δεν διαφέρει πολύ από την εντολή

  ... ...

  Σε αυτό το στάδιο πρέπει επίσης να αποφεύγονται οι ένθετες (nested) επαναλήψεις.

Περιγράφω το πρόγραμμα

Στο στάδιο αυτό γίνεται η πλήρης περιγραφή του προγράμματος, αφού το κείμενο του δεν επαρκεί για μια πλήρη εκτέλεση. Το ζήτημα αυτό τροποποιεί επίσης τη συντακτική δεξιότητα του χρήστη. Εδώ μπορούμε να προτείνουμε ένα είδος τεκμηρίωσης (documentation) προγραμματισμού που περιλαμβάνει:

- Τις *προδιαγραφές* (specifications) του προγράμματος: ποια είναι τα δεδομένα, ποια είναι τα αποτελέσματα, πώς ορίζονται τα αποτελέσματα με βάση τα δεδομένα. Εδώ μπορούμε να παρατηρήσουμε ότι η εισαγωγή των προδιαγραφών επιτρέπει την εξέλιξη της έννοιας του προγράμματος, κάνοντας την να προσεγγίσει την έννοια της συνάρτησης (function).
- Ένα *λεξικό* από κωδικούς παράλληλο με την ακολουθία των εντολών (Πίνακας 3).

Παράδειγμα περιγραφής που υπολογίζει το μισθό ενός εργαζόμενου σε συνάρτηση των ωρών εργασίας του

Ακολουθία εντολών	**Λεξικό κωδικών**
[illegible] : όνομα	όνομα (char) : όνομα εργαζόμενου
[illegible] : αωε	αωε (integer) : αριθμός ωρών εργασίας
[illegible] : μω	μω (real) : μισθός ανά ώρα
μισθ : = αωε* μω	μισθ (real) : μισθός
[illegible] μισθ	

Πίνακας 3 Ακολουθία εντολών και περιγραφή μεταβλητών

Ένας τέτοιος πίνακας ενισχύει τη σημασιολογική πτυχή της περιγραφής του προγράμματος ως συνάρτησης και εδώ μπορούμε να δείξουμε στους σπουδαστές ότι δεν υπάρχουν παρά μόνο τρεις τύποι «αποτελεσματικών» εντολών (εκτός της επανάληψης και της υπό συνθήκη εντολής): *δίνω μια τιμή σε μια μεταβλητή με ανάγνωση* ή *με εκχώρηση, γράφω ένα αποτέλεσμα.*

Η έννοια της μεταβλητής μπορεί να έχει ήδη απαντηθεί και μεταβληθεί στα μαθηματικά ή να αποτελέσει πρότερη γνώση για τις μαθηματικές μεταβλητές. Συνιστά ωστόσο ένα ιδιαίτερο πρόβλημα, αφού δεν υπάρχει πλήρης αντιστοιχία ανάμεσα σε μαθηματική μεταβλητή και πληροφορική μεταβλητή.

Ο παραπάνω πίνακας (Πίνακας 3) επιτρέπει επίσης να γίνουν επαληθεύσεις και διορθώσεις: καμία μεταβλητή δεν μπορεί να χρησιμοποιηθεί στο δεξιό τμήμα μιας απόδοσης τιμών πριν λάβει κάποια τιμή και, συνεπώς, εάν δεν βρίσκεται ήδη μέσα στο λεξικό των κωδικών.

Αντιλαμβάνομαι και περιγράφω με δομημένο τρόπο

Οι περιορισμοί (restrictions) που έγιναν στο τρίτο στάδιο (περιγράφω ένα σύνολο εντολών) είχαν σκοπό την ελάττωση της πολυπλοκότητας και τη χρήση ενός απλού εργαλείου παραγωγής το οποίο επιτρέπει την πρόσθεση εντολών με σειριακό τρόπο και διατηρεί, συνεπώς, μια γραμμική σύλληψη και αντίληψη του προγράμματος. Όταν όμως μεταβαίνουμε σε πιο σύνθετα προβλήματα, η διατήρηση της ακολουθιακής δομής στη σύλληψη του προγράμματος δεν είναι πλέον εφικτή παρά μόνο εάν δεν μπούμε σε λεπτομέρειες και ονομάσουμε κάποια τμήματα, τα οποία πρέπει να διευκρινιστούν και να περιγραφούν αναλυτικά σε μεταγενέστερο χρόνο.

Επομένως, τροποποιείται η έννοια της περιγραφής του προς ανάπτυξη προγράμματος: η περιγραφή παίρνει πλέον τη μορφή δένδρου και το πρόγραμμα χάνει έτσι σε μεγάλο βαθμό τη σειριακότητα-γραμμικότητά του. Στο σημείο αυτό είναι εύλογο να εισαχθεί η έννοια της *διαδικασίας* (procedure), η οποία εμπλουτίζει την έννοια του προγράμματος. Ένα νέο εργαλείο προστίθεται στα ήδη υπάρχοντα: αποσύνθεση ενός μέρους που προηγουμένως είχε αντιμετωπιστεί (και προσδιοριστεί) συνολικά.

Εισάγω πληροφορικές μεταβλητές

Από τη στιγμή που μια επανάληψη εμπεριέχει μια *εκχώρηση τιμής* (affectation), είμαστε υποχρεωμένοι να βγούμε από τη μονοδιάστατη σχέση ανάμεσα σε *όνομα* και *τιμή*, δηλαδή από το νοητικό πλαίσιο που περιγράφηκε στο τέταρτο στάδιο.

Η μεταβλητή στον προγραμματισμό παρουσιάζει ιδιαίτερες δυσκολίες, επειδή δεν υπάρχει αντίστοιχη έννοια στις υπάρχουσες εμπειρίες και γνώσεις του σπουδαστή. Ο C. Pair (Pair, 1988), σημειώνει τρία στάδια αύξουσας πολυπλοκότητας στην οικοδόμηση της έννοιας της πληροφορικής μεταβλητής.

1. Δεν υπάρχει συνύπαρξη δύο τιμών της μεταβλητής μέσα σε μια εκτέλεση επανάληψης, όπως για παράδειγμα όταν υπολογίζουμε διαδοχικά το μισθό των εργαζομένων μιας επιχείρησης (Πίνακας 3). Για κάθε εργαζόμενο, διαβάζουμε το *όνομα*, τον αριθμό ωρών εργασίας του *αωε*, τον ωριαίο μισθό του *ωμ* και υπολογίζουμε το μισθό του *μισθ*, χωρίς καμία αναφορά στον προηγούμενο εργαζόμενο. Σε μια δομημένη περιγραφή, προγραμματίζουμε ξεχωριστά το μισθό ενός εργαζομένου και *όνομα, αωε, ωμ* και *μισθ* είναι *μαθηματικές μεταβλητές*.
2. Χρησιμοποιείται απευθείας η παλιά τιμή μια μεταβλητής για την εύρεση της νέας. Εάν στο προηγούμενο παράδειγμα θέλουμε να υπολογιστεί το άθροισμα των μισθών: s:= s+μισθ; στο μυαλό του προγραμματιστή δεν υπάρχει στιγμιαία (έστω) συνύπαρξη της παλιάς με τη νέα τιμή, αλλά απαντάται η αναγκαιότητα της *αρχικοποίησης* (initialization) της μεταβλητής.
3. Στο εσωτερικό της επανάληψης χρησιμοποιείται μέσα σε διαφορετικές εντολές παλιά και νέα τιμή. Εδώ υπάρχει μεγάλη πιθανότητα να υπάρξει σύγχυση εάν δεν γίνει διάκριση ανάμεσα στη μία και την άλλη τιμή και δεν σημειωθούν στον πίνακα εντολών.

Το νοητικό αυτό πλαίσιο οδηγεί στην τροποποίηση της έννοιας της μεταβλητής που συνοδεύεται με αλλαγή της μεθόδου. Εφόσον δεν υπάρχει η προαπαιτούμενη γνώση στις

εμπειρίες του σπουδαστή, η έννοια της πληροφορικής μεταβλητής δεν μπορεί να οικοδομηθεί με το να «βρεθούμε στη θέση του υπολογιστή» (Pair, 1990).

Υπολογίζω πάνω σε «αφηρημένα» αντικείμενα

Σε όλα τα προηγούμενα στάδια, η κύρια χρησιμοποιούμενη μέθοδος ήταν πάντα η ίδια: ο προγραμματιστής μπαίνει στη θέση του υπολογιστή και αυτό που άλλαζε μόνο ήταν το επίπεδο «δεξιότητας». Έτσι, η αντιμετώπιση προβλημάτων όπου η παραπάνω μέθοδος δεν έχει καμία εφαρμογή, συνιστά ένα ισχυρό γνωστικό εμπόδιο για τους σπουδαστές.

Μια πρώτη περίπτωση δυσκολίας αφορά την επεξεργασία αντικειμένων διαφορετικών από αυτά που χειρίζεται συνήθως η χρησιμοποιούμενη γλώσσα προγραμματισμού. Τα αντικείμενα αυτά είναι, κατά κανόνα, πιο σύνθετα (πρόκειται για τους τύπους δεδομένων που ορίζονται από το χρήστη). Το γεγονός αυτό οδηγεί τόσο στον εμπλουτισμό του προς επίτευξη αποτελέσματος όσο και των εργαλείων παραγωγής και της μεθόδου. Αναφορικά με το αποτέλεσμα, πρέπει μέσα στην περιγραφή του προγράμματος να αποδεχθούμε ενδιάμεσες φόρμες που αναφέρονται πάνω σε «αφηρημένα» αντικείμενα – με την έννοια του ότι η γλώσσα προγραμματισμού δεν τα αναγνωρίζει – και πάνω σε αντικείμενα που αναφέρονται σε συναρτήσεις οι οποίες θα οριστούν αργότερα. Στα νέα εργαλεία πρέπει να εισαχθεί η αναπαράσταση των αντικειμένων. Για τη μέθοδο τίθεται το ερώτημα, πότε και τι να αναπαρασταθεί. Η διδακτική προσέγγιση για να απαντηθεί αυτό το ερώτημα ακολουθείται όσο πιο αργά είναι δυνατόν, όταν πλέον καμιά αποσύνθεση δεν είναι εφικτή (Pair, 1988).

Επομένως, οι υπολογισμοί πάνω σε «αφηρημένα» αντικείμενα αλλάζουν εντελώς το πλαίσιο αναφοράς. Σε αυτό το στάδιο ο προγραμματιστής δεν μπαίνει πλέον στη θέση του υπολογιστή, αλλά στη θέση μιας *αφηρημένης μηχανής*, την οποία πρέπει να ανακαλύψει και η οποία στη συνέχεια, θα «υλοποιηθεί» αναπαριστώντας τα αντικείμενα και τις πράξεις της.

Εφευρίσκω αλγόριθμους

Υπάρχουν περιπτώσεις όπου ακόμα και όταν έχουμε φανταστεί την *αφηρημένη μηχανή* που χειρίζεται τα αντικείμενα τα οποία εμφανίζονται στην εκφώνηση του προβλήματος, δεν ξέρουμε πώς να ξεκινήσουμε την κατάτμηση του προβλήματος. Σε αυτή την περίπτωση, η μέθοδος πρέπει να αναθεωρηθεί ολοκληρωτικά. Δεν μπορούμε πλέον να βασιστούμε σε αυτό που θα κάναμε στη θέση του υπολογιστή (Pair, 1988). Ο οδηγός

είναι πλέον αυτό που ξέρουμε, οι προδιαγραφές (specifications) του προγράμματος, ενώ παράλληλα καθοδηγούμαστε από τον τύπο των δεδομένων ή του αποτελέσματος.
Ένα χαρακτηριστικό παράδειγμα είναι ο *πύργος του Ανόι*, όπου το δεδομένο του αριθμού των δίσκων, ένας ακέραιος n μας οδηγεί στην αναδρομή (περνώντας από το n στο n-1). Η μέθοδος αυτή οδηγεί στο συναρτησιακό προγραμματισμό (functional programming).

6.3. Στάδια μάθησης του προγραμματισμού

Τα στάδια που περιγράφονται στην παρούσα ενότητα (Pair, 1988) αντικατοπτρίζουν σε μεγάλο βαθμό την προσέγγιση επίλυσης προβλημάτων προγραμματισμού με αξιοποίηση του δομημένου-διαδικασιακού (ή προστακτικού) παραδείγματος, το οποίο υπήρξε κυρίαρχο στη διδασκαλία του προγραμματισμού.
Ωστόσο, τίθενται πολλά ερωτήματα εάν ξεφύγουμε από το παράδειγμα του δομημένου (διαδικαστικού) προγραμματισμού: Αν περάσουμε πιο γρήγορα στο συναρτησιακό προγραμματισμό, αποφεύγουμε τα προβλήματα με τις πληροφορικές μεταβλητές; Αλλάζει το νοητικό πλαίσιο που εξαρτάται από το σχήμα *περιγραφή δεδομένων – περιγραφή αποτελεσμάτων*;
Τα σύγχρονα προγραμματιστικά περιβάλλοντα (Dagdilelis et al., 2001) που έχουν κάνει την εμφάνισή τους τα τελευταία χρόνια, όπως οι μικρογλώσσες (mini-languages) (Brusilovsky et al., 1997), τα οπτικά (visual) (Green & Navarro, 1995) και τα καθοδηγούμενα από τα γεγονότα (event driven) (Lang & Saacks-Giguette, 1999) περιβάλλοντα παρέχουν νέες δυνατότητες για την οικοδόμηση και την ανάπτυξη προγραμματιστικών δεξιοτήτων, αλλά η έρευνα σε αυτόν το χώρο είναι στις απαρχές της (Deek & McHugh, 1998).
Ένα ενδιαφέρον πεδίο έρευνας σχετικά με την ανάπτυξη προγραμματιστικών δεξιοτήτων από αρχάριους προγραμματιστές φαίνεται επίσης να προσφέρεται από το προγραμματιστικό παράδειγμα του αντικειμενοστραφούς προγραμματισμού (object oriented programming) (Xinogalos et al, 2005).
Στη συνέχεια του θα ασχοληθούμε με το πώς οι άνθρωποι γίνονται ειδικοί σε μια συγκεκριμένη περιοχή γνώσης μέσω της εμπειρίας τους στην επίλυση προβλημάτων και, πιο συγκεκριμένα, στην επίλυση προβλημάτων σε προγραμματιστικό περιβάλλον με την ευρεία έννοια του όρου (δηλαδή με τη χρήση συμβατικών γλωσσών προγραμματισμού ή

σύγχρονων εφαρμογών λογισμικού γενικής χρήσης – όπως είναι τα λογιστικά φύλλα και τα συστήματα βάσεων δεδομένων).

Θα μελετήσουμε κυρίως τα επιμέρους διδακτικά εμπόδια που απαντώνται όταν προσεγγίζονται βασικές έννοιες προγραμματισμού όπως η μεταβλητή, οι δομές ελέγχου, η επανάληψη, η αναδρομή και η διαδικασία και θα προτείνουμε κατάλληλες διδακτικές παρεμβάσεις για την υπέρβαση αυτών των εμποδίων. Θα ασχοληθούμε συνεπώς με τους μηχανισμούς που διέπουν τη μετάβαση από την κατάσταση του αρχάριου στην κατάσταση του ειδικού (Smith, 1991). Στο πλαίσιο της μελέτης αυτής θα χρησιμοποιηθεί ως βασικό προγραμματιστικό παράδειγμα το παράδειγμα του διαδικασιακού ή προστακτικού προγραμματισμού.

7. Η έννοια της μεταβλητής

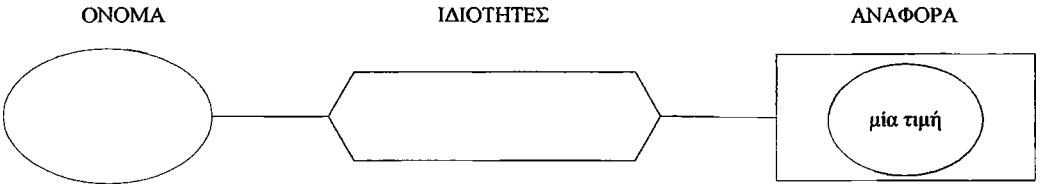
7.1. Η οικοδόμηση της έννοιας της μεταβλητής

Οι ***μεταβλητές*** (variables) είναι ονόματα για θέσεις μνήμης στο υπολογιστικό σύστημα. Κάθε θέση μνήμης διαθέτει μια διεύθυνση (που προσδιορίζεται με τη σειρά της από ένα μοναδικό αναγνωριστικό). Η επιλογή κατάλληλου ονόματος μεταβλητής στο πρόγραμμα διευκολύνει συχνά την κατανόηση της έννοιας της μεταβλητής ως ονομασίας μιας θέσης μνήμης. Το γεγονός όμως αυτό δεν επαρκεί για να κατανοήσουν οι αρχάριοι το λειτουργικό νόημα της μεταβλητής. Η κοινή αντίληψη των μαθητών για τη μεταβλητή (που συνήθως οφείλεται σε ανεπαρκείς διδακτικές προσεγγίσεις) βασίζεται στην *αναλογία του κουτιού* (το «περιέχον»), γεγονός που εισάγει διάφορες παρανοήσεις στη λειτουργία της (Τζιμογιάννης & Κόμης, 2000).

Με αυστηρό ορισμό, μια *μεταβλητή* (Σχήμα 10) στην Πληροφορική είναι μία τετράδα που αποτελείται από ένα *όνομα*, ένα *σύνολο ιδιοτήτων*, μια *αναφορά* και μία *τιμή* (Horowitz, 1984). Με τον τρόπο αυτό είναι δυνατόν να διασαφηνιστούν διάφορα θέματα που σχετίζονται με τις μεταβλητές και πολύ συχνά, όπως θα δούμε στη συνέχεια, δημιουργούν γνωστικά και διδακτικά προβλήματα. Ένα τέτοιο παράδειγμα είναι η απλή εντολή ανάθεσης:

$$X := X + 1$$

Στην περίπτωση αυτή μπορεί να γίνει κατανοητό ότι το Χ στη δεξιά πλευρά του συμβόλου := αναφέρεται στην τρέχουσα τιμή, ενώ το Χ στην αριστερή πλευρά αναφέρεται στην αναφορά.



Σχήμα 10 Τα τέσσερα συστατικά μέρη μιας μεταβλητής (Horowitz, 1984)

Η πρόσκτηση της έννοιας της *μεταβλητής στον προγραμματισμό* είναι ιδιαίτερα σημαντική και, παράλληλα, αρκετά δύσκολη. Όπως αναφέρει ο Dijkstra, όταν κάποιος καταλάβει που χρησιμοποιούνται οι μεταβλητές στον προγραμματισμό, έχει καταλάβει την πεμπτουσία του προγραμματισμού (Dijkstra, 1972). Η έννοια της μεταβλητής απαντάται ήδη από τα πρώτα βήματα της προγραμματιστικής δραστηριότητας. Παρά τις επιμέρους διαφοροποιήσεις, η πλειονότητα των γλωσσών προγραμματισμού βασίζεται στην ίδια θεμελιώδη τεχνική, αυτήν της «διαχείρισης τιμών που περιέχονται σε μεταβλητές» (Delannoy, 1996).

Η δραστηριότητα του προγραμματισμού απαιτεί, σε μεγάλο βαθμό, τη γνώση της χρήσης συμβόλων και αναπαραστάσεων για τη διαχείριση και την αποθήκευση δεδομένων. Ωστόσο, το ζήτημα της αποθήκευσης των δεδομένων με τη μορφή μεταβλητών – οι οποίες γίνονται αντιληπτές με τη χρήση συμβολικών κωδικών – συνιστά ένα δύσκολο πρόβλημα στον προγραμματισμό (Dufoyer, 1988). Το πρόβλημα γίνεται ακόμη πιο δύσκολο αφού η έννοια της αποθήκευσης εμπεριεκλείει τόσο την εκχώρηση (ανάθεση) τιμής, όσο και την εμφάνισή της – δηλαδή λειτουργικά χαρακτηριστικά της έννοιας, που δεν μπορούν εύκολα να διαχωριστούν.

7.2. Διδακτικά προβλήματα σχετικά με την έννοια της μεταβλητής

Τα βασικά διδακτικά προβλήματα που εμφανίζονται κατά τη διδασκαλία και τη μάθηση της έννοιας της μεταβλητής σχετίζονται με την ανάθεση τιμής, τον τύπο της μεταβλητής, τις ειδικές κατηγορίες μεταβλητών που χρησιμοποιούνται από τον προγραμματιστή αλλά δεν αντιστοιχούν στις απαιτήσεις του προβλήματος, και στην απόδοση αρχικών τιμών

(την «αρχικοποίηση») της μεταβλητής (Samurçay, 1989 / Du Boulay, 1989 / Τζιμογιάννης & Κόμης, 2000).

Από τις έρευνες φαίνεται ότι οι σπουδαστές οικοδομούν αναπαραστάσεις και νοητικά μοντέλα από το χώρο του προβλήματος και τις μεταφέρουν ως έχουν στο χώρο του προγραμματισμού, γεγονός που δημιουργεί διδακτικά και μαθησιακά προβλήματα (Lagrange, 1992). Διάφορα επίσης προβλήματα σχετίζονται με πρότερες γνώσεις και παρανοήσεις των σπουδαστών καθώς και με θέματα που αφορούν τις χρησιμοποιούμενες συμβάσεις από τις διάφορες γλώσσες.

Ένα σημαντικό μαθησιακό πρόβλημα προέρχεται από το γεγονός ότι η έννοια της μεταβλητής στον προγραμματισμό δεν ταυτίζεται με αυτήν των μαθηματικών. Ιδιαίτερα δύσκολος είναι, επίσης, ο έλεγχος από τους σπουδαστές οι οποίοι δεν έχουν φτάσει στο στάδιο της υποθετικής – επαγωγικής και φορμαλιστικής σκέψης σύμφωνα με τον Piaget.

Συντακτικό και συμβάσεις της γλώσσας προγραμματισμού

Σε πολλές γλώσσες προγραμματισμού απαντώνται κάποιες συμβάσεις που συνιστούν ένα επιπρόσθετο γνωστικό εμπόδιο, αφού «παραβιάζουν» ορισμένους συνηθισμένους κανόνες της λειτουργίας της σκέψης. Για παράδειγμα, στις γλώσσες υψηλού επιπέδου η προσωρινή αποθήκευση κάποιας τιμής γίνεται απευθείας από το σύστημα, οπότε ο προγραμματιστής δεν χρειάζεται να ασχοληθεί – ενώ δεν συμβαίνει το ίδιο στη γλώσσα Assembly.

Για παράδειγμα, όταν γράφουμε στην Pascal `Sum:= Sum+N;` ορισμένες αρχές της έννοιας της διατήρησης παραβιάζονται. Σε αυτό το παράδειγμα, η μεταβλητή `Sum` βρίσκεται και στις δύο πλευρές της σχέσης. Υπάρχει συνεπώς κάποια στιγμή όπου η τιμή `Sum+N` θα πρέπει να βρίσκεται κάπου πριν εισαχθεί στην `Sum`. Ο σπουδαστής θα κατανοούσε καλύτερα τη σύνταξη εάν γράφαμε:

Provisory := Sum+N;

Sum := Provisory

Η ***επιλογή κατάλληλου ονόματος μεταβλητής*** μπορεί να διευκολύνει τη κατανόηση της ίδιας της έννοιας στον προγραμματισμό ως ονομασίας μιας θέσης μνήμης. Η χρήση ενός διαφορετικού συστήματος κωδικοποίησης (όπου αντί για μια διεύθυνση μνήμης γίνεται χρήση ενός ονόματος) παρέχει μεγαλύτερη ελευθερία χειρισμού και διευκολύνει κυρίως την αναγνωσιμότητα των προγραμμάτων.

Ιδιαίτερη προσοχή απαιτείται όταν προσεγγίζεται διδακτικά η έννοια του ***τύπου μιας μεταβλητής***, αφού ο τύπος προσδιορίζει το είδος των πληροφοριών που αναπαρίστανται από τη μεταβλητή, την κωδικοποίηση, τους περιορισμούς που αφορούν στις τιμές που μπορούν να αναπαρασταθούν, και τις δυνατές πράξεις που μπορούν να πραγματοποιηθούν από τις αντίστοιχες μεταβλητές. Οι αρχάριοι προγραμματιστές χειρίζονται σχετικά εύκολα αριθμητικές μεταβλητές, αλλά συναντούν δυσκολίες όταν χειρίζονται δεδομένα τύπου string ή Boolean για τη σύνταξη λογικών συνθηκών (Τζιμογιάννης & Κόμης, 1999).

Ανάθεση τιμής σε μεταβλητή

Η ***ανάθεση*** (ή ***εκχώρηση***) ***τιμής μιας μεταβλητής*** συνιστά βασικό διδακτικό εμπόδιο αφού πολλές γλώσσες προγραμματισμού χρησιμοποιούν (για λόγους απλότητας) συμβολισμούς όπως `A=5`, με τους οποίους, και λόγω της χρήσης του συμβόλου = (όπως για παράδειγμα σε πολλές εκδόσεις της C ή της Basic) δημιουργούν σύγχυση με το αντίστοιχο σύμβολο στα μαθηματικά. Σε άλλες γλώσσες αντί για το = χρησιμοποιείται το := (π.χ. στην Pascal) ώστε να γίνεται εμφανής η διάκριση.

Πολλοί αρχάριοι προγραμματιστές δεν κατανοούν ότι η εντολή ανάθεσης καταχωρεί δεδομένα στη θέση της προϋπάρχουσας τιμής (βλέπε Σχήμα 10), η οποία χάνεται. Έτσι θεωρούν ότι το «κουτί» διατηρεί περισσότερες από μία τιμές. Συχνά επίσης θεωρούν ότι η μεταβλητή έχει τη δυνατότητα να «θυμάται» την ιστορία των αναθέσεων που έχουν προηγηθεί, σχηματίζοντας ένα νοητικό μοντέλο τύπου λίστας ή σωρού, από όπου μπορούν να ανακτηθούν οι τιμές αυτές (Τζιμογιάννης & Κόμης, 2000).

Μπορούμε να διακρίνουμε τέσσερις διαφορετικούς τύπους εκχώρησης (ή ανάθεσης) τιμής μιας μεταβλητής, οι οποίοι ενέχουν διαφορετικού επιπέδου διδακτικά προβλήματα (Samurçay, 1989 / Τζιμογιάννης & Κόμης, 2000):

1. **Εκχώρηση σταθερής τιμής**

Π.χ. a:=15, list:= "word", test:= true

Στην περίπτωση αυτή είναι απαραίτητο να εξηγείται στους σπουδαστές γιατί δεν μπορεί, παρότι φαίνεται μαθηματικά εύλογο να γραφεί

15:= x;

2. **Εκχώρηση τιμής μετά από υπολογισμό**

Π.χ. a:=15+4, y:= 4*x+5

Οι δύο αυτές περιπτώσεις (εκχώρηση σταθερής τιμής και εκχώρηση τιμής μετά από υπολογισμό) δεν εμφανίζουν ιδιαίτερα διδακτικά προβλήματα. Δεν συμβαίνει όμως το ίδιο για τις επόμενες περιπτώσεις (αντικατάσταση και συσσώρευση).

3. Αντικατάσταση (duplication)

Π.χ. x:= y

Η διδακτική ιδιαιτερότητα αυτής της περίπτωσης μελετάται συνέχεια. Το πρόβλημα της αντιμετάθεσης τιμών δύο μεταβλητών είναι ένα συχνό πρόβλημα στον προγραμματισμό. Στην περίπτωση αυτή μελετάται επιπρόσθετα επειδή απαιτεί τη χρήση βοηθητικής μεταβλητής. Ο πρώτος «αλγόριθμος» που συχνά χρησιμοποιούν οι σπουδαστές είναι: *βάλε όπου Α το Β και όπου Β το Α*. Σε γλώσσα Pascal (με A=10, B=20) το πρόγραμμα είναι το `Antimetathesh1` (Πίνακας 4).

Η εκτέλεση του προγράμματος `Antimetathesh1` θα δώσει A=20, B=20 γιατί «δεν έχει κρατηθεί» σε κάποια θέση μνήμης η αρχική τιμή του Α. Αντίθετα, σωστό πρόγραμμα είναι το `Antimetathesh2`, όπου χρησιμοποιείται η εσωτερική μεταβλητή S (Πίνακας 4).

```
Program Antimetathesh1;
Var
  A, B: Integer;
Begin
A:=10; B:=20;
A:=B; B:=A;
Writeln ('A= ', A, 'B= ', B)
End.
```

```
Program Antimetathesh2;
Var
Begin
A:=10; B:=20;
S:=A; A:=B; B:=S;
Writeln ('A= ', A, 'B= ', B)
End.
```

Πίνακας 4 Πρόγραμμα αντιμετάθεσης τιμών

4. Συσσώρευση (accumulation)

Π.χ. μετρητής, αθροιστής και πολλαπλασιαστής

```
X:= x+1, sum:= sum+n, expn:= expn*n
```

Η τέταρτη περίπτωση, όπου η μεταβλητή παρουσιάζεται στην πλήρη διάστασή της στην Πληροφορική, είναι και εκείνη που προκαλεί τις περισσότερες διδακτικές δυσκολίες. Η διάκριση ανάμεσα στην ***έννοια της μεταβλητής στα μαθηματικά και στην***

Πληροφορική συνιστά ένα ιδιαίτερο πρόβλημα, το οποίο αναδεικνύεται μέσα από όλες τις έρευνες που έχουν γίνει σε σπουδαστές.

Το πρόβλημα αυτό προκύπτει κυρίως σε αρχάριους προγραμματιστές, οι οποίοι χρησιμοποιούν έγκυρες αναπαραστάσεις της μεταβλητής που έχουν οικοδομήσει στο χώρο της άλγεβρας και τις μεταφέρουν στον προγραμματισμό, όπου όμως η εγκυρότητα τους δεν είναι πλέον βέβαιη. Για παράδειγμα, το πιο συνηθισμένο λάθος είναι η χρήση της *εντολής εκχώρησης* ως ισότητας.

Αντίθετα, η εντολή στην περίπτωση αυτή μετατρέπει τη μεταβλητή σε συνάρτηση (με τη μαθηματική έννοια του όρου). Για παράδειγμα, η λειτουργία αυτή είναι εμφανής στο πλαίσιο μιας επαναληπτικής δομής που υπολογίζει και εμφανίζει το άθροισμα των 100 πρώτων φυσικών αριθμών.

Στην περίπτωση της πρώτης δομής η δήλωση ν ← ν + 1 λειτουργεί ως συνάρτηση. Στην περίπτωση της δεύτερης δομής εμφανίζονται 2 μεταβλητές – συναρτήσεις, η μεταβλητή ν (ν + 1) και η μεταβλητή άθροισμα (άθροισμα ← άθροισμα + ν) (Πίνακας 5).

φυσικοί_1_100	*άθροισμα_100*
: ν	: ν, *άθροισμα*
ν ← 0	ν ← 0 *άθροισμα* ← 0
ν <> 100 ν ← ν + 1 ν	ν < 100 ν ← ν + 1 *άθροισμα* ← *άθροισμα* + ν
	άθροισμα

Πίνακας 5 Παράδειγμα κώδικα (Γλώσσα) με μεταβλητή ως συνάρτηση

Συνεπώς, η έννοια της μεταβλητής στην Πληροφορική μπορεί να οικοδομηθεί πάνω στη πιθανή πρότερη γνώση της μεταβλητής στα μαθηματικά. Ο στατικός όμως χαρακτήρας που έχει η μεταβλητή στα μαθηματικά (αφού αναπαριστά μια λειτουργική σχέση) αποτελεί συνήθως διδακτικό εμπόδιο στην αναπαράσταση της δυναμικής τροποποίησης της τιμής της μεταβλητής κατά την εκτέλεση του προγράμματος (Rogalski & Vergnaud, 1987).

Επομένως, οι οικείες διαδικασίες επίλυσης προβλημάτων σχετικών με μεταβλητές (αυτές που γίνονται «με το χέρι») άλλοτε παίζουν παραγωγικό ρόλο ενισχύοντας την οικοδόμηση εννοιών και άλλοτε ρόλο διδακτικού ή επιστημολογικού εμποδίου.

Πρέπει επίσης να τονιστεί ότι οι ακόλουθες εκφράσεις (που πιθανότατα οφείλεται σε νοητικό μοντέλο από τα μαθηματικά) δεν είναι ισοδύναμες:

$X \leftarrow \psi$ και $\psi \leftarrow X$

Ειδικές μεταβλητές

Στον προγραμματισμό μπορούμε να διακρίνουμε ***δύο τύπους μεταβλητών*** σε σχέση με τα νοητικά μοντέλα που χρησιμοποιούν οι σπουδαστές για να σχεδιάσουν τη λύση ενός προβλήματος (Τζιμογιάννης & Κόμης 2000 / Τζιμογιάννης & Γιούνης, 2003): τις *εξωγενείς* και τις *ενδογενείς* μεταβλητές.

Οι έρευνες έχουν δείξει ότι οι σπουδαστές χειρίζονται με διαφορετικό τρόπο τις μεταβλητές ανάλογα με τη λειτουργική τους θέση μέσα στο πρόγραμμα: οι μεταβλητές που εκφράζουν διασαφηνισμένα δεδομένα του προβλήματος (***εξωγενείς μεταβλητές***) είναι αντικείμενο ορθότερου χειρισμού από τις μεταβλητές που απαιτούνται από το πρόγραμμα αλλά δεν έχουν σημασιολογικό αντίστοιχο όταν το πρόβλημα επιλύεται με το χέρι (***ενδογενείς μεταβλητές***).

Οι *εξωγενείς* μεταβλητές αντιστοιχούν σε τιμές που ελέγχονται από τους χρήστες του προγράμματος, όπως για παράδειγμα οι μεταβλητές που αντιστοιχούν σε εισόδους (input) και εξόδους (output) του προβλήματος και κατά κανόνα παρουσιάζουν τα λιγότερα διδακτικά προβλήματα.

Οι *ενδογενείς* μεταβλητές αντιστοιχούν σε τιμές που ελέγχονται από τους προγραμματιστές και είναι απαραίτητες για την προγραμματιστική επίλυση του προβλήματος – όπως για παράδειγμα αυτές που απαιτούνται για την αλλαγή θέσης (αντιμετάθεση) δύο τιμών.

Οι μεταβλητές αυτές είναι πολύ πιο δύσκολο να αναπαρασταθούν από τους σπουδαστές, αφού η αναπαράστασή τους απαιτεί την κατανόηση της εσωτερικής λειτουργίας (καταστάσεις του συστήματος) του υπολογιστή.

Συνεπώς, η απουσία κατάλληλων αναπαραστάσεων του συστήματος «υπολογιστής – γλώσσα προγραμματισμού» αποτελεί μια από τις βασικές αδυναμίες των αρχάριων προγραμματιστών (Lagrange, 1992). Το πρόβλημα αυτό αντιμετωπίζεται με την οικοδόμηση των κατάλληλων *Συστημάτων Αναπαράστασης και Επεξεργασίας.*

Μια άλλη κατηγορία ειδικών μεταβλητών συνιστούν ο αθροιστής (S ← S + X), ο πολλαπλασιαστής (P ← P * X) και ο μετρητής (m ← m + 1). Πρόκειται για μεταβλητές του προγράμματος, οι οποίες επιτελούν ένα συγκεκριμένο υπολογιστικό ρόλο ανεξάρτητα από τον αλγόριθμο ή το πρόβλημα που αυτός επιλύει (Τζιμογιάννης & Γιούνης, 2003) και έχουμε ήδη δει κάποιες χρήσεις τους.

Απόδοση αρχικών τιμών μεταβλητής

Στο περιβάλλον των περισσότερων γλωσσών προγραμματισμού, η αρχική τιμή των μεταβλητών του προγράμματος είναι απροσδιόριστη, γεγονός που εισάγει σφάλματα στους υπολογισμούς, κυρίως στις περιπτώσεις των επαναληπτικών δομών.

Ένα σημαντικό διδακτικό πρόβλημα, που απαντάται κυρίως στους αρχάριους προγραμματιστές, είναι ότι ξεχνούν πολύ συχνά τις απαραίτητες ***αποδόσεις αρχικών τιμών στις μεταβλητές.***

Η απόδοση αρχικών τιμών (ή *αρχικοποίηση*) απαιτεί τη διατύπωση της υπόθεσης μιας αρχικής κατάστασης του συστήματος (που αποτελεί μια γνωστική δεξιότητα ευρύτερη του προγραμματισμού και σχετίζεται άμεσα με τις γνωστικές δεξιότητες επίλυσης προβλήματος) – δηλαδή την οικοδόμηση κατάλληλης αναπαράστασης.

Επομένως, συνιστά ένα πιο δύσκολο πρόβλημα από την ενημέρωση της τιμής μιας μεταβλητής ή τον έλεγχο της τιμής της, αφού δεν απαντάται στις διαδικασίες επίλυσης προβλήματος «με το χέρι» (Samurçay, 1989).

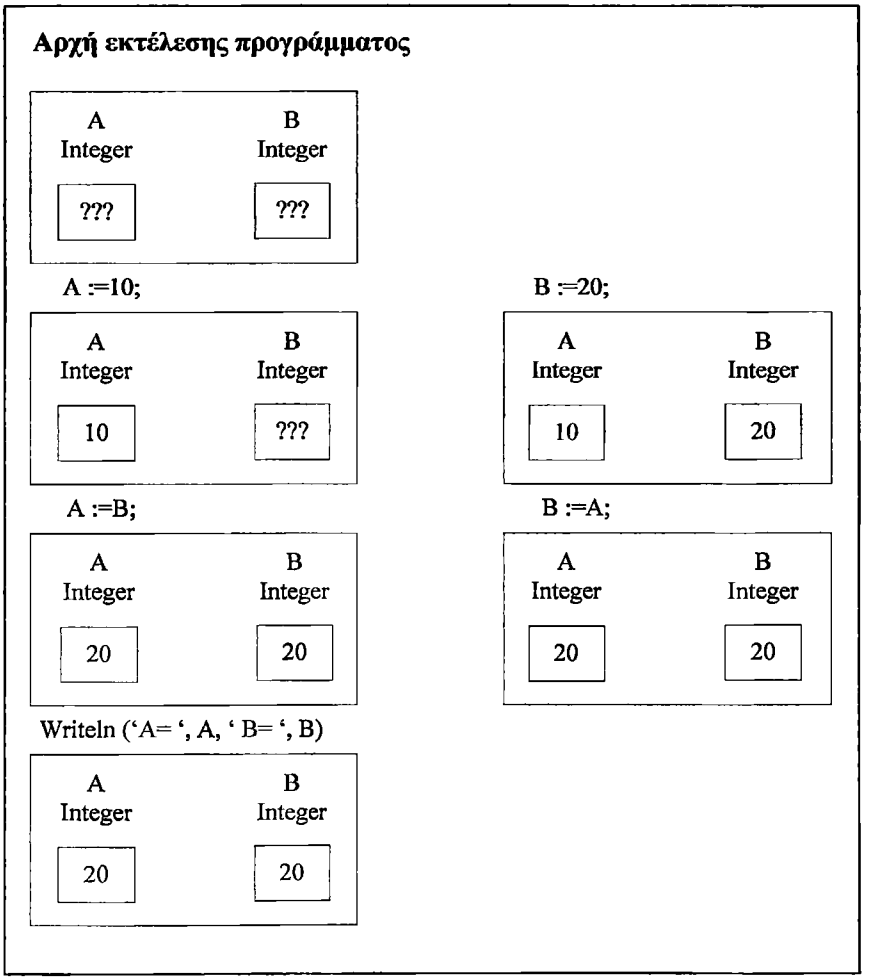
Οι αρχικοποιήσεις μεταβλητών που απαιτούνται κατά τη χρήση επαναληπτικών διαδικασιών συνιστούν ένα σύνθετο γνωστικό και, συνεπώς, διδακτικό πρόβλημα, αφού όπως είδαμε δεν είναι απαραίτητες σε προβλήματα που επιλύονται με μη προγραμματιστικό τρόπο. Στο πλαίσιο αυτό, κλασικό παράδειγμα προσφέρουν τα σφάλματα απόδοσης αρχικών τιμών σε μεταβλητές τύπου αθροιστή, πολλαπλασιαστή και μετρητή.

7.3. Διδακτικές προσεγγίσεις της έννοιας της μεταβλητής

Για την αντιμετώπιση όλων των προβλημάτων που αναλύθηκαν στις προηγούμενες ενότητες σχετικά με την οικοδόμηση και τη χρήση της έννοιας της μεταβλητής από αρχάριους προγραμματιστές απαιτούνται κατάλληλες διδακτικές παρεμβάσεις, οι οποίες

θα επιτρέπουν στους μαθητές που εμπλέκονται με προβλήματα προγραμματισμού να επιλύουν προβλήματα που τους υποχρεώνουν να συγκρούονται με γνωστές ή υποτιθέμενες δυσκολίες και γνωστικά εμπόδια.

Στο πλαίσιο αυτό, απαιτείται μια αλλαγή προοπτικής σχετικά με τις διδακτικές παρεμβάσεις που αφορούν στην οικοδόμηση της έννοιας της μεταβλητής: αντί οι παρεμβάσεις αυτές να είναι επικεντρωμένες στα θέματα που άπτονται των συντακτικών επιλογών μέσα στις διάφορες γλώσσες προγραμματισμού, πρέπει να ασχολούνται με την οικοδόμηση κατάλληλων νοητικών μοντέλων σχετικών με τις εμπλεκόμενες έννοιες.



Σχήμα 11 Γραφική αναπαράσταση εκτέλεσης προγράμματος (αντιμετάθεση τιμών)

Μια χρήσιμη διδακτική στρατηγική για την αποφυγή προβλημάτων που σχετίζονται με τις διάφορες μορφές ανάθεσης τιμής σε μεταβλητή, αλλά και σε θέματα που αφορούν στην αντιμετάθεση τιμών μεταβλητών, συνίστανται στην αναπαράσταση της μνήμης του υπολογιστή με τις εκάστοτε τιμές των μεταβλητών του προγράμματος (βλέπε Σχήμα 11) και στην εκτέλεση του αλγορίθμου επίλυσης του προβλήματος «με το χέρι» (Αλεβίζος & Καμπουρέλης, 1988).

Η διάκριση ανάμεσα στην έννοια της μεταβλητής στα μαθηματικά και την Πληροφορική είδαμε ότι συνιστά ιδιαίτερο πρόβλημα που πρέπει να αντιμετωπιστεί με κατάλληλες διδακτικές παρεμβάσεις, οι οποίες δεν θα περιορίζονται απλώς στην παράθεση παραδειγμάτων, αλλά θα οργανώνονται σε ένα μακροπρόθεσμο διδακτικό σχεδιασμό. Θα πρέπει επίσης να συμπεριλαμβάνουν κατάλληλα σχεδιασμένα έργα, τα οποία θα οδηγούν στην οικοδόμηση λειτουργικών νοητικών αναπαραστάσεων και νοητικών μοντέλων για την έννοια της μεταβλητής και τις πολλαπλές μορφές ανάθεσης τιμής (Τζιμογιάννης & Κόμης, 2000).

Μία διδακτική προσέγγιση της έννοιας της μεταβλητής με τη χρήση λογιστικού φύλλου

Όπως έχουμε ήδη συζητήσει, η έννοια της μεταβλητής αποτελεί σύνθετο διδακτικό πρόβλημα. Στο πλαίσιο του δομημένου προγραμματισμού δεν μπορεί να οικοδομηθεί χωρίς να έχουν οικοδομηθεί εκ των προτέρων οι έννοιες της καταχώρησης της τιμής, της απόδοσης της τιμής και της εκτύπωσης της τιμής (του αποτελέσματος). Επίσης, συνδέεται με τον τύπο, την τιμή αλλά και την ονοματολογία της. Μια εναλλακτική διδακτική προσέγγιση της έννοιας της μεταβλητής μπορεί να γίνει με τη χρήση λογιστικού φύλλου (Greff, 1996).

Καταρχήν από τη σκοπιά του χρήστη: Τα σύγχρονα λογιστικά φύλλα επιτρέπουν με τη λειτουργία μεταφοράς και απόθεσης (drag and drop) να χειριζόμαστε κελιά (που στην πραγματικότητα αναπαριστούν μεταβλητές) με εύχρηστο τρόπο, χωρίς προβλήματα σύνταξης. Η διεύθυνση του κελιού συνιστά ταυτόχρονα και το όνομα της μεταβλητής, ενώ ο τύπος της δεν είναι προκαθορισμένος αλλά προσδιορίζεται τη στιγμή της εκχώρησης.

Ο τύπος όμως της μεταβλητής παίζει ρόλο όταν αυτή χρησιμοποιείται για να ορίζονται άλλες μεταβλητές. Η δυνατότητα ονομασίας ενός κελιού (δηλαδή μιας μεταβλητής) επιτρέπει πλέον να δουλεύουμε πάνω σε ένα συμβολικό όνομα και όχι στην αντίστοιχη τιμή της. Επιπρόσθετα, έχουμε τη δυνατότητα να εισάγουμε την έννοια της διεύθυνσης (χρησιμοποιώντας τις έννοιες της απόλυτης και της σχετικής αναφοράς). Παράλληλα, η παρατήρηση της γραμμής εντολών του λογιστικού φύλλου επιτρέπει να αντιληφθούμε τον τρόπο με τον οποίο λειτουργεί εσωτερικά.

8. Οι δομές ελέγχου και η έννοια της επιλογής

8.1. Η οικοδόμηση της δομής ελέγχου

Η *δομή ελέγχου* είναι μία από τις τρεις βασικές λογικές δομές του προγραμματισμού και απαντάται σε όλα τα σύγχρονα προγραμματιστικά περιβάλλοντα. Η δομή αυτή επιτρέπει σε έναν αλγόριθμο να αλλάξει την εκτέλεσή του ανάλογα με τις περιστάσεις. Πρόκειται, με άλλα λόγια, για τη δυνατότητα του αλγορίθμου να επιλέγει ή να ελέγχει τις εντολές που πρόκειται να εκτελεστούν. Χρησιμοποιείται για τη λήψη απόφασης μεταξύ δύο διαφορετικών καταστάσεων, εκ των οποίων η μία είναι *αληθής* και η άλλη *ψευδής*.

Συνήθως, στα προβλήματα περιλαμβάνονται κάποιοι έλεγχοι δεδομένων. Ανάλογα με τα αποτελέσματά τους επιλέγονται οι επεξεργασίες (οι εντολές) που θα ακολουθήσουν. Η διατύπωση των ελέγχων γίνεται με τη χρήση λογικών προτάσεων που ονομάζονται *συνθήκες* (Τζιμογιάννης & Γιούνης, 2003).

Η δύναμη της επιλογής έγκειται στο ότι επιτρέπει στον επεξεργαστή να ακολουθήσει διαφορετικά μονοπάτια μέσα στον αλγόριθμο σύμφωνα με τις περιστάσεις. Χάρη στις ένθετες επιλογές επιτρέπεται απροσδιόριστος αριθμός μονοπατιών. Όπως παρατηρούν οι Goldschlager και Lister (Goldschlager & Lister, 1996), χωρίς επιλογή θα ήταν αδύνατο να γραφούν αλγόριθμοι με σημαντική πρακτική αξία. Όπως είδαμε στο προηγούμενο κεφάλαιο, οι βασικές μορφής της δομής αυτής είναι η *απλή*, η *διπλή*, η *σύνθετη* και η *πολλαπλή* επιλογή.

Οι διάφορες έρευνες δείχνουν ότι η δομή ελέγχου είναι μια δομή που οικοδομείται με ιδιαίτερη δυσκολία από τους σπουδαστές που μαθαίνουν προγραμματισμό (Rogalski, 1987 / Soloway & Spohrer, 1989). Στη μάθησή της παρεμβαίνουν διάφοροι παράγοντες, οι οποίοι και πρέπει να λαμβάνονται υπόψη στο πλαίσιο της διδασκαλίας του προγραμματισμού. Οι παράγοντες αυτοί έχουν σχέση τόσο με τις γενικότερες δυσκολίες που έχουν οι σπουδαστές στην οικοδόμηση των διαφόρων εννοιών του προγραμματισμού, όσο και με τις εγγενείς ιδιαιτερότητες που χαρακτηρίζουν τη συγκεκριμένη δομή. Οι ιδιαιτερότητες αυτές έχουν μελετηθεί στα πλαίσια διαφόρων εμπειρικών ερευνών που έχουν διεξαχθεί κάτω από το πρίσμα της διδακτικής και της ψυχολογίας του προγραμματισμού (Soloway & Spohrer, 1989 / Green, Hoc, Samurçay, Gilmore, 1990).

Μέσα σε ένα πρόγραμμα που είναι διατυπωμένο ως μια αυστηρή αλληλουχία εντολών, για κάθε είσοδο δεδομένων το σύνολο των περιγραφόμενων υπολογισμών πραγματοποιείται με τη σειρά γραφής των εντολών. Οι *δομές ελέγχου* διακόπτουν αυτό τον ισομορφισμό γραμμικής τάξης ανάμεσα στο κείμενο του προγράμματος και στην εκτέλεσή του (Rogalski, 1987). Στην περίπτωση των δομών ελέγχου, για μία είσοδο δεδομένων θα ακολουθείται κατά την εκτέλεση μόνο μία από τις δυνατές διαδρομές.

Ένα από τα σημαντικά εμπόδια για την οικοδόμηση της δομής ελέγχου άπτεται της αναπαράστασης που διαθέτουν οι σπουδαστές για τη ***σειριακή εκτέλεση του προγράμματος***. Μέσα σε ένα πρόγραμμα που είναι διατυπωμένο ως μία αυστηρή αλληλουχία εντολών, το σύνολο των περιγραφόμενων υπολογισμών πραγματοποιείται για κάθε είσοδο δεδομένων με τη σειρά εγγραφής των εντολών.

Συνεπώς, οι ιδιαιτερότητες που χαρακτηρίζουν την υλοποίηση ενός προγράμματος μέσω ενός πληροφορικού συστήματος θα παίξουν σημαντικό ρόλο μέσα στις διαδικασίες πρόσκτησης και οικοδόμησης των δομών ελέγχου. Ειδικότερα, στο πλαίσιο όπου μια δομή ελέγχου οφείλει να διαχειριστεί μια τροποποίηση των σχέσεων ανάμεσα στη γραμμική γραφή του προγράμματος και στο σειριακό τρόπο (μέσω της ακολουθιακής δομής) της εκτέλεσης, οι αναπαραστάσεις και τα νοητικά μοντέλα που έχει ο μαθητής πάνω στη σειριακή αυτή σχέση θα αλληλεπιδράσουν με τις σημασιολογικές και τις συντακτικές ιδιότητες του προγράμματος.

8.2. Διδακτικά προβλήματα σχετικά με την έννοια της δομής ελέγχου

Στο πλαίσιο αυτό, η χρήση της δομής ελέγχου προσθέτει ιδιαίτερες γνωστικές δυσκολίες στις γενικές γνωστικές δυσκολίες που αφορούν στην πρόσκτηση του προγραμματισμού. Οι ιδιαίτερες αυτές δυσκολίες συνδέονται, κατά κύριο λόγο, με τα ακόλουθα σημεία:

- λογικό περιεχόμενο των συνθηκών (συνδυασμός περιπτώσεων, αναλυτικότητα και αποκλειστικότητα, λογικές πράξεις σύζευξης, διάζευξης, άρνησης, κ.λπ.) (Hoc, 1989),
- συμβολικές αναπαραστάσεις αυτών των περιπτώσεων,
- σημασιολογικές και συντακτικές ιδιότητες της δομής ελέγχου μέσα στη χρησιμοποιούμενη γλώσσα προγραμματισμού (Du Boulay, 1989),

- αλληλεπιδράσεις με τις αναπαραστάσεις της ακολουθιακής μορφής της εκτέλεσης.

Από τη σχετική βιβλιογραφία φαίνεται ότι οι ***πρότερες μαθηματικές γνώσεις*** καθώς και οι ***γνώσεις λογικής*** παίζουν σημαντικό ρόλο στην οικοδόμηση της έννοιας της δομής ελέγχου. Σε γενικό επίπεδο, οι έρευνες συγκλίνουν στο ότι οι μαθητές που έχουν ένα ανεπτυγμένο μαθηματικό υπόβαθρο μαθαίνουν πιο γρήγορα τις δομές του τύπου αυτού (Rogalski, 1987). Η δεξιότητα της χρήσης λογικών τελεστών είναι απαραίτητη για την πρόσκτηση της υπό συνθήκη εντολής. Όμως, η δεξιότητα αυτή δεν είναι αρκετή, αφού σημαντικό ρόλο παίζει το «ακατάλληλο» μοντέλο της ακολουθιακής αντίληψης εκτέλεσης ενός προγράμματος από το πληροφορικό μέσο.

Οι γνωστικές δυσκολίες που αφορούν στη μάθηση των δομών ελέγχου περιλαμβάνουν πολλές ταυτόχρονες πτυχές (Samurçay, 1987): ***λογικό περιεχόμενο, σημασιολογικό περιεχόμενο*** και ***αναπαράσταση της συνθήκης***. Οι έρευνες επιτρέπουν επίσης τη διατύπωση κάποιων υποθέσεων πάνω στην επιρροή των πρότερων γνώσεων και στις ιδιαίτερες δυσκολίες των επεξεργασιών υπό συνθήκη.

Η ανάλυση των σχέσεων ανάμεσα στο εννοιολογικό πεδίο και το έργο του προγραμματισμού με τη χρήση δομής ελέγχου δίνει μια σειρά από διδακτικά προβλήματα, τα οποία και πρέπει να λαμβάνονται υπόψη κατά τη διδασκαλία του προγραμματισμού (Τζιμογιάννης & Κόμης, 1999).

Ο χώρος προβλημάτων που απαιτούν μια δομή ελέγχου μπορεί να οριστεί με πολύ γενικό τρόπο ως τα προβλήματα στα οποία μια επεξεργασία εξαρτάται από την τιμή μιας παραμέτρου (Rogalski, 1987). Η εξάρτηση μπορεί να έχει τη μορφή μιας ***απλής υπό συνθήκη εντολής***:

```
συνθήκη      επεξεργασία 1
```

Μπορεί επίσης να είναι εξάρτηση με τη μορφή μιας ***σύνθετης υπό συνθήκη εντολής***:

```
συνθήκη      επεξεργασία 1
             επεξεργασία 2
```

Ή μπορεί να είναι ***πολλαπλή με πολλές συνθήκες*** και πολλές αποφάσεις:

```
συνθήκη      επεξεργασία 1
συνθήκη      επεξεργασία 2

...

συνθήκη N    επεξεργασία N
             επεξεργασία N + 1
```

Τέλος, μπορεί να έχει ή να μην έχει μια δυαδική σχέση ανάμεσα στις συνθήκες ναι τις επεξεργασίες (μέσα στη γραφή του προγράμματος).

Τα πιο συνηθισμένα σφάλματα που γίνονται με την δομή επιλογής είναι (Putnam et al., 1989):

α) Όταν η συνθήκη είναι ψευδής, το πρόγραμμα σταματά.

β) Όταν η συνθήκη είναι ψευδής, ο έλεγχος περνά στην αρχή του προγράμματος.

α) Όταν η συνθήκη οδηγεί σε μία εκτύπωση ( ), τότε τυπώνονται τόσο η τιμή της μεταβλητής, όσο και η τιμή της συνθήκης.

Η ***φύση των συνθηκών*** από τις οποίες εξαρτάται ο έλεγχος οφείλει επίσης να ληφθεί υπόψη. Στο πλαίσιο αυτό, η συνθήκη μπορεί να είναι *ενδογενής*, ορισμένη από το αποτέλεσμα ενός υπολογισμού – με την ευρεία έννοια του όρου – ή *εξωγενής*, ορισμένη από το χρήστη μέσω μιας αλληλεπιδραστικής εισόδου. Οι έρευνες με σπουδαστές έχουν δείξει ότι ο ενδογενής ή εξωγενής χαρακτήρας των συνθηκών παράγει πάνω στην επεξεργασία από αρχάριους προγραμματιστές παρόμοια αποτελέσματα με αυτά της μεταβλητής. Με άλλα λόγια, οι αρχάριοι προγραμματιστές συναντούν μεγαλύτερες δυσκολίες στις ενδογενείς παρά στις εξωγενείς συνθήκες (Rogalski & Vergnaud, 1987).

Η ***φύση των «πληροφορικών αντικειμένων»*** που εμπλέκονται με σαφή τρόπο μέσα στη συνθήκη και οι αναπαραστάσεις των σπουδαστών για αυτά τα αντικείμενα (Lagrange, 1992) μπορεί να ποικίλλει. Για παράδειγμα, οι σχέσεις που αφορούν μεταβλητές όπως οι αριθμοί ή οι χαρακτήρες επιτρέπουν να οδηγηθούμε σε οικεία σχήματα, οπότε δεν υπάρχει ιδιαίτερο πρόβλημα στο χειρισμό. Αντίθετα, η χρήση λογικών (Boolean) μεταβλητών απαιτεί νέες και πιο σύνθετες αναπαραστάσεις, ενώ η παρέμβαση των συναρτήσεων οδηγεί σε αλλαγή γνωστικού επιπέδου, το οποίο είναι πολύ πιο σύνθετο στο νοητικό χειρισμό του από τους σπουδαστές (Rogalski, 1987).

Με άλλα λόγια, η ύπαρξη πρότερων *προ-προγραμματιστικών γνώσεων* (preprogramming knowledge) μέσα στην ***αναπαράσταση των συνθηκών*** τροποποιεί τη δυσκολία μιας δομής ελέγχου. Έτσι, έχει παρατηρηθεί ότι κατά τη συγγραφή ελέγχων (απλή υπό συνθήκη εντολή) από σπουδαστές σπάνια γίνονται λάθη τα οποία αφορούν απλούς αριθμητικούς ελέγχους (όπως M>0) αλλά γίνονται συγκριτικά περισσότερα λάθη με πιο σύνθετες αριθμητικές δομές (που αναπαριστούν διάρκεια) ή δεδομένα αλφαριθμητικού τύπου (string), για τα οποία οι σπουδαστές δεν διαθέτουν τις ίδιες πρότερες γνώσεις (Bonar & Soloway, 1985 / Τζιμογιάννης & Κόμης, 1999).

Η υλοποίηση κώδικα μέσα σε μια γλώσσα προγραμματισμού μπορεί να στηριχθεί σε μια σύνταξη περισσότερο ή λιγότερο πλούσια και να επιτρέψει να αντανακλαστεί περισσότερο ή λιγότερο άμεσα η σχέση ανάμεσα σε συνθήκες και επεξεργασίες. Η χρησιμοποιούμενη σύνταξη μπορεί να απομακρύνεται περισσότερο ή λιγότερο από τη ***φυσική γλώσσα***, γεγονός που οδηγεί σε ανάλογα διδακτικά εμπόδια.

Η εντολή **...** **...** **...** της Pascal (και όλων των προστατικών γλωσσών που υποστηρίζουν τέτοιου τύπου δομή) πρόσκειται στην επεξεργασία της φυσικής γλώσσας και είναι πιο κατανοητή από τους σπουδαστές. Αντίθετα η δομή ελέγχου σε ένα λογιστικό φύλλο (π.χ. Microsoft Excel) εκφράζεται με μια συνάρτηση με τρία ορίσματα, όπου το πρώτο αφορά στον έλεγχο και τα άλλα τις δυνατές τιμές ανάλογα με τον έλεγχο:

(logical_test, value_if_true, value_if_false)

(A1>=5, "προβιβάζεται", "απορρίπτεται")

Το ίδιο συμβαίνει αντίστοιχα και για τις μη προστακτικές (imperative) γλώσσες. Για παράδειγμα, στην Prolog δηλώσεις και δομές ελέγχου χρησιμοποιούν το ( ) της συνεπαγωγής, ενώ στη Lisp το ( ) είναι μια συνάρτηση επιλογής με πολλά ορίσματα.

Στο πλαίσιο αυτό μπορούμε να υποθέσουμε ότι η ύπαρξη ***πρότερων γλωσσολογικών γνώσεων*** διευκολύνει την πρόσβαση στη συντακτική φόρμα, ενώ η απουσία τέτοιου είδους γνώσεων αναγκάζει τους σπουδαστές να αφομοιώσουν εξολοκλήρου νέες παραστάσεις (εκφράσεις), οι οποίες συνδέονται με την έννοια της συνάρτησης – που είναι ιδιαίτερα περίπλοκη (Soloway & Spohrer, 1989).

Ωστόσο, πρέπει να υπογραμμιστεί ότι οι πρότερες γλωσσολογικές γνώσεις μπορούν επίσης να παίξουν ένα ρόλο *διδακτικού εμποδίου* μέσα στις προσκτήσεις, στο μέτρο που οι κανόνες λειτουργίας της φυσικής γλώσσας και αυτοί των γλωσσών προγραμματισμού δεν συνάδουν απόλυτα.

Στην παραπάνω ανάλυση της πολυπλοκότητας των δομών ελέγχου πρέπει επίσης να προστεθεί η σχέση μέσα στο πρόγραμμα ανάμεσα σε αυτές τις δομές και σε άλλες δομές ελέγχου: χωριστή δομή ελέγχου, διαδοχή πολλών δομών ελέγχου, ένθεση δομής ελέγχου μέσα σε άλλες, ένθεση μιας δομής ελέγχου σε μια επαναληπτική δομή ή το αντίστροφο (Rogalski & Vergnaud, 1987). Αναμφίβολα, αυτές οι διαστάσεις δεν είναι ανεξάρτητες του *πλαισίου προγραμματισμού*, «τοπικού» τύπου επικεντρωμένου στην αφομοίωση των σχημάτων (όπου γίνονται απλές επεξεργασίες, με περιορισμένες σειρές εντολών) ή «ολικού» τύπου επικεντρωμένες πάνω στο συγχρονισμό και τη δομή της επεξεργασίας.

Τέλος, φαίνεται απαραίτητο να ληφθούν υπόψη οι αλληλεπιδράσεις ανάμεσα στη λογική της επεξεργασίας υπό συνθήκη, της σημασιολογικής και συντακτικής αναπαράστασης μέσα σε μια γλώσσα προγραμματισμού και τις αναπαραστάσεις τις οποίες διαθέτει το υποκείμενο πάνω στις ιδιότητες εκτέλεσης από το πληροφορικό μέσο.

8.3. Τα στυλ προγραμματισμού: η εντολή GO TO και η εντολή IF THEN

Οι πρώτες έρευνες πάνω στη δομή ελέγχου γίνονται την εποχή όπου η επιστημονική κοινότητα συζητά πάνω στην εντολή και τοποθετείται για την ανάγκη ανάπτυξης νέων γλωσσών προγραμματισμού, οι οποίες δεν θα χρησιμοποιούν την παραπάνω εντολή (Dahl et al., 1972). Βασικό επιχείρημα αυτού του εγχειρήματος ήταν ότι με την εντολή αυτή δεν δημιουργούνται καλώς δομημένα προγράμματα, αυξάνεται η πολυπλοκότητά τους και δυσχεραίνεται η αποσφαλμάτωσή τους.

Στο πλαίσιο αυτό, έχουν γίνει διάφορες έρευνες σχετικά με δύο διαφορετικά στυλ προγραμματισμού: αυτό που χρησιμοποιεί το άλμα (*JUMP-style*) – δηλαδή την εντολή – και αυτό που χρησιμοποιεί την ένθεση (*NEST-style*) – δηλαδή αυτό που χρησιμοποιεί ένθετες δομές (Rogalski & Samurcay, 1990). Ενώ σε εκφραστικό επίπεδο (ως συνάρτηση γνώσεων) η δομή άλματος φαίνεται ευκολότερη από ένθετη δομή, ο έλεγχος της σε επίπεδο προγράμματος γίνεται πιο δύσκολος. Επιπρόσθετα, από τα αποτελέσματα των ερευνών φαίνεται ότι όταν χρησιμοποιούνται συντακτικά σύμβολα (...), που ορίζουν το χώρο αναφοράς των και , συντελούν στην ευκολότερη χρήση της δομής ελέγχου σε σχέση με τη δομή άλματος και διευκολύνουν στην κατανόηση των προγραμμάτων.

Πρόγραμμα με GOTO και `IF` ... `THEN` ... `ELSE`

Το επόμενο πρόγραμμα (Πίνακας 6) υπολογίζει τα αποτελέσματα μιας δημοσκόπησης (P: θετικές απαντήσεις, N: αρνητικές απαντήσεις, A: αποχές).

120 response="yes" 150 130 response="no" 160 140 A=A+1 : 170 150 P=P+1 : 170 160 N=N+1 170 (*συνέχεια προγράμματος*)	Με άλμα σε Basic
response='yes' P:=P+1 response='no' N:=N+1 A:=A+1; ("συνέχεια προγράμματος")	Με άλμα σε Pascal

Πίνακας 6 Προγράμματα με άλμα (GOTO) και ένθεση

8.4. Διδακτική προσέγγιση της δομής ελέγχου – επιλογής

Στη γενική προβληματική των διαφόρων ερευνών στη διδακτική του προγραμματισμού παίζουν σημαντικό ρόλο και οι σχέσεις ανάμεσα σε γλώσσες προγραμματισμού και προσκτήσεις προγραμματιστικών εννοιών. Συνεπώς, τίθεται ένα ερώτημα που αφορά το χώρο εγκυρότητας των αποτελεσμάτων: μια γλώσσα είναι καλύτερη όποιο και αν είναι το πλαίσιο ή εξαρτάται η κατανόηση από την προσαρμογή της γλωσσικής μορφής στις νοητικές πράξεις που απαιτούνται από το προς επίλυση έργο;

Έρευνες που αφορούσαν στη σχέση ανάμεσα σε συντακτικές φόρμες και τύπους προβλημάτων έδειξαν ότι ένα απόσπασμα κειμένου προγράμματος περιέχει δύο τύπους πληροφοριών: α) μία πληροφορία σειριακή (ποια πράξη ακολουθεί κάθε κατάσταση) και β) μία πληροφορία περιστασιακή (σε ποια συνθήκη εκτελείται μια πράξη). Οι έρευνες αυτές κατέστησαν προφανές ότι η σύνταξη των δομών ελέγχου αντανακλά αυτούς τους δύο τύπους πληροφορίας σε μεγαλύτερο ή μικρό βαθμό καταλληλότητας και είναι συνεπώς περισσότερο ή λιγότερο κατάλληλη για την επεξεργασία σειριακών ή περιστασιακών ερωτημάτων (Green, Hoc, Samurçay, Gilmore, 1990).

Πρότερες γνώσεις λογικής και αναπαραστάσεις του πληροφορικού συστήματος

Σημαντικό ρόλο στην πρόσκτηση της έννοιας της δομής ελέγχου διαδραματίζουν και οι πρότερες γνώσεις μαθηματικών και λογικής (Bonar & Soloway, 1985). Σε γενικό επίπεδο, οι έρευνες δείχνουν ότι οι σπουδαστές που έχουν ένα ανεπτυγμένο μαθηματικό υπόβαθρο, οικοδομούν πιο γρήγορα τις δομές αυτού του τύπου.

Η δεξιότητα της χρήσης λογικών τελεστών (and, or, not) είναι απαραίτητη για την πρόσκτηση της υπό συνθήκης εντολής, αλλά η δεξιότητα αυτή δεν είναι αρκετή αφού το ακατάλληλο μοντέλο της γραμμικής αντίληψης εκτέλεσης ενός προγράμματος από το πληροφορικό μέσο παίζει ανασταλτικό ρόλο.

Οι αναπαραστάσεις των σπουδαστών πάνω στη σειριακή εκτέλεση του προγράμματος από το πληροφορικό μέσο δεν φαίνεται να ανασχηματίζονται από τις απλές αλληλεπιδράσεις ανάμεσα στη χρήση από τα υποκείμενα των δομών ελέγχου. Για το λόγο αυτό απαιτούνται κατάλληλες διδακτικές παρεμβάσεις από τη μεριά του εκπαιδευτικού.

Αλγόριθμοι επίλυσης του προβλήματος σύγκρισης και εκτύπωσης του μεγαλύτερου από δύο πραγματικούς αριθμούς

Στην ενότητα αυτή δίνονται τρεις δυνατές λύσεις που κάνουν χρήση ενός [illegible], δύο [illegible] και ενός [illegible] (Αλεβίζος & Καμπουρέλης, 1988) και σχολιάζονται διδακτικά.

1η περίπτωση

```
[illegible] megalyteros1;

    X, Y, Meg: [illegible];

            ('Δώσε δύο αριθμούς');
           (X, Y);
  Meg:= Y;
     X > Y      Meg:=X;
            ('Μεγαλύτερος είναι ο αριθμός :' Meg);
```

Εδώ γίνεται χρήση μιας εντολής εκχώρησης `Meg:= Y;` Που κάνει την υπόθεση ότι ο μεγαλύτερος αριθμός είναι ο δεύτερος. Στην περίπτωση που η υπόθεση δεν αληθεύει, τότε εκτελείται η εσωτερική εντολή της [illegible], η οποία διορθώνει την αρχική τιμή της μεταβλητής Meg.

2η περίπτωση

```
megalyteros2;

X, Y, Meg:  ;

    ('Δώσε δύο αριθμούς');
   (X, Y);
X <= Y      Meg:=Y;
 X > Y      Meg:=X;
    ('Μεγαλύτερος είναι ο αριθμός :' Meg);
```

Στην περίπτωση αυτή δεν γίνεται αυθαίρετη χρήση της εντολής `Meg:=Y` αλλά χρησιμοποιείται ως εσωτερική της εντολής . Και στις δύο περιπτώσεις, οι χρησιμοποιούμενοι αλγόριθμοι δεν είναι κοντά στη φυσική γλώσσα, αλλά χρησιμοποιούν κυρίως τις ιδιότητες του πληροφορικού συστήματος: μία τιμή καταχωρίζεται στη μεταβλητή Meg και αλλάζει μόνο όταν δεν ισχύει η λογική συνθήκη (για την πρώτη περίπτωση).

Αντίθετα, ο επόμενος αλγόριθμος – που υλοποιείται με τη χρήση της εντολής – είναι πιο κοντά στη φυσική γλώσσα.

3η περίπτωση

```
megalyteros2;

X, Y, Meg:  ;

    ('Δώσε δύο αριθμούς');
   (X, Y);
 X > Y      Meg:=Y;
    ('Μεγαλύτερος είναι ο αριθμός :' Meg);
```

Ο αλγόριθμος αυτός δεν συνάδει μόνο με τη φυσική γλώσσα, αλλά είναι και πιο οικονομικός αφού απαιτεί ακριβώς μία σύγκριση και μία εκχώρηση.

Αντίθετα, στην πρώτη περίπτωση απαιτείται μία (1) σύγκριση και μιάμιση (1,5) εκχώρηση κατά μέσο όρο, αφού η πρώτη εκχώρηση θα γίνει οπωσδήποτε και η δεύτερη (εάν X > Y) έχει πιθανότητα 50% να πραγματοποιηθεί. Στην δεύτερη περίπτωση θα γίνουν δύο συγκρίσεις (X =< Y και X > Y) και μόνο μία εντολή εκχώρησης.

9. Η έννοια της επανάληψης

9.1. Η οικοδόμηση της επαναληπτικής δομής

Η ***επαναληπτική διαδικασία*** (iteration) ή ***βρόχος*** (loop) επιτρέπει την επανάληψη μιας επεξεργασίας για κάποιες φορές, ο αριθμός των οποίων δεν είναι κατανάγκη γνωστός εκ των προτέρων. Συνεπώς, η διαδικασία μπορεί να εκτελείται όσο μία συνθήκη είναι αληθής (conditional loop) ή για προκαθορισμένο αριθμό επαναλήψεων (counting loop) (Θραμπουλίδης, 2000).

Δύο συνηθισμένες μορφές της επαναληπτικής διαδικασίας (που χρησιμοποιούνται όταν δεν γνωρίζουμε εκ των προτέρων τον ακριβή αριθμό των επαναλήψεων) είναι οι ακόλουθες (Πίνακας 7).

«Προ»-ελεγχόμενη επανάληψη	**«Μετα» -ελεγχόμενη επανάληψη**
συνθήκη *εντολή_1* *εντολή_2* . . . *εντολή_ν*	*εντολή_1* *εντολή_2* . . . *εντολή_ν* *συνθήκη*

Πίνακας 7 Δομές ακαθόριστης επανάληψης (με άγνωστο αριθμό επαναλήψεων)

Στο πλαίσιο αυτό, η επανάληψη ελέγχεται από μία συνθήκη, η οποία καθορίζει την έξοδο από την επαναληπτική εκτέλεση των εντολών. Στην περίπτωση αυτή, η επανάληψη ονομάζεται *ακαθόριστη*. Η συνθήκη άλλοτε ελέγχεται εκ των προτέρων, δηλαδή πριν εκτελεστεί το σώμα της ανακύκλωσης, οπότε έχουμε *«προ»-ελεγχόμενη επανάληψη* και άλλοτε ελέγχεται εκ των υστέρων, μετά την εκτέλεση του σώματος της επανάληψης, οπότε έχουμε *«μετα»-ελεγχόμενη επανάληψη* (Goldschlager & Lister, 1996).

Όπως θα δούμε και στη συνέχεια, η αποτυχία να καθοριστεί επιτυχώς η συνθήκη τερματισμού αποτελεί ένα από τα πιο συχνά λάθη στη σχεδίαση αλγορίθμων. Όπως και στη δομή επιλογής, η επανάληψη διακόπτει την ακολουθιακή εκτέλεση του προγράμματος, με όλες τις γνωστικές συνέπειες που επιφέρει αυτό το γεγονός.

Μια Τρίτη μορφή επαναληπτικής διαδικασίας, η απλούστερη εννοιολογικά, είναι αυτή όπου ο αριθμός των επαναλήψεων του σώματος του βρόχου είναι γνωστός πριν ακόμα

εκτελεστεί η ανακύκλωση. Για το λόγο αυτό αποκαλείται *καθορισμένη επανάληψη*. Στην περίπτωση αυτή, ο αλγόριθμος έχει την ακόλουθη μορφή:

```
        Ν φορές
    σώμα ανακύκλωσης
```

και με ψευδοκώδικα γράφεται ως εξής:

```
i      1      m      n
εντολή_1
εντολή_2

εντολή_ν
```

Στην περίπτωση που το βήμα n του προηγούμενου αλγορίθμου είναι 1, η δομή αυτή είναι ισοδύναμη με τον ακόλουθο αλγόριθμο, εκφρασμένο ως δομή ακαθόριστης επανάληψης:

```
μετρητής ← 1 'θέσε μετρητή_επαναλήψεων ίσο με 1
    μετρητής < m
εντολή_1
εντολή_2

εντολή_ν

μετρητής ← μετρητής + 1 'πρόσθεσε 1 στο μετρητή_επαναλήψεων
```

Συνεπώς, ο αριθμός των επαναλήψεων μπορεί να είναι γνωστός εκ των προτέρων ή να εξαρτάται από μια κατάσταση που εμφανίζεται κατά τη διάρκεια της επεξεργασίας. Στην πρώτη περίπτωση, λοιπόν, ο έλεγχος τέλους είναι μια σταθερά, ενώ στη δεύτερη περίπτωση είναι η τιμή μιας μεταβλητής που υπολογίζεται κατά τη διαδικασία της επανάληψης. Στη δεύτερη περίπτωση φαίνεται και η ισχύς της επαναληπτικής δομής: ένας αλγόριθμος πεπερασμένου μήκους που περιγράφει μια διεργασία ακαθόριστης διάρκειας (Goldschlager & Lister, 1996).

Η οικοδόμηση της έννοια της επανάληψης (Laborde et al., 1985) εμπερικλείει αφενός τον προσδιορισμό των στοιχειωδών δράσεων ή κανόνων που πρέπει να επαναληφθούν και αφετέρου τη συνθήκη που προσδιορίζει το τέλος ή τη συνέχιση της επανάληψης (Rogalski, 1990).

Κάτω από αυτό το πρίσμα, οι βασικές παράμετροι που απαιτούνται για την οικοδόμηση της επαναληπτικής διαδικασίας είναι τρεις:

1. Οικοδόμηση και σχεδιασμός της επεξεργασίας, δηλαδή ανάπτυξη της διατύπωσης των εντολών που θα επαναληφθούν (και ονομάζονται *σώμα του βρόχου* ή της ανακύκλωσης).

2. Προσδιορισμός της *συνθήκης ελέγχου* για τη διακοπή της επανάληψης (έλεγχος), η οποία αφορά σε παραστάσεις που συναπαρτίζουν το σώμα του βρόχου (και συνεπώς πρέπει να προσδιοριστεί η θέση τους μέσα σε αυτό).
3. Προσδιορισμός της αρχικής κατάστασης των μεταβλητών του βρόχου (*αρχικοποίηση*).

Κάθε ένα από τα παραπάνω σημεία προκαλεί ιδιαίτερα διδακτικά προβλήματα, τα οποία και θα μελετηθούν στη συνέχεια.

Πριν όμως μελετήσουμε τα επιμέρους διδακτικά προβλήματα πρέπει να δούμε πως οικοδομείται η επαναληπτική προσέγγιση επίλυσης προβλημάτων. Μια επαναληπτική διαδικασία συνιστά τη μετάβαση από μια γνωστή αρχική κατάσταση σε μια τελική ζητούμενη κατάσταση και οικοδομείται σε τέσσερίς φάσεις (Arsac, 1980).

α) Καταρχήν διατυπώνεται μια επαγωγική υπόθεση, η οποία στην πραγματικότητα συνίσταται στην περιγραφή μιας ενδιάμεσης κατάστασης.

β) Στη συνέχεια αναζητείται ο τρόπος μετάβασης από τη μία κατάσταση στην επόμενη της – δηλαδή περιγράφεται όλη η ακολουθία των εντολών που επιτρέπουν αυτή τη μετάβαση.

γ) Στο επόμενο βήμα διατυπώνεται η συνθήκη ελέγχου για το τέλος της επανάληψης, δηλαδή το πότε η ενδιάμεση κατάσταση ταυτίζεται με την τελική κατάσταση.

δ) Τέλος, αναζητείται από πού αρχίζει η επαναληπτική διαδικασία – δηλαδή, εντοπίζεται μία περίπτωση όπου η επαγωγική υπόθεση προφανώς ισχύει. Η περίπτωση αυτή δεν είναι άλλη από την περίπτωση όπου γνωρίζουμε όλες τις τιμές των μεταβλητών που εμπλέκονται στην επανάληψη και, συνήθως, πρόκειται για τη γνωστική αρχική κατάσταση.

Συμπερασματικά, η επαναληπτική διαδικασία προϋποθέτει την οικοδόμηση γνωστικών δεξιοτήτων υψηλού επιπέδου και νοητικών στοιχείων ελέγχου ενός μη καθορισμένου συνόλου πράξεων. Στο μαθηματικό επίπεδο, το βασικό εργαλείο ελέγχου είναι η ***μαθηματική επαγωγή***, με τη βοήθεια της οποίας εγκαθίσταται η κληρονομικότητα της ιδιότητας $P(n)$ ή του συνόλου πράξεων $P(n)$ στη μετάβαση από το n στο $n+1$. Η κληρονομικότητα αυτή θα ισχύει όσες φορές είναι απαραίτητο.

9.2. Διδακτικά προβλήματα σχετικά με την έννοια της επανάληψης

Η επαναληπτική δομή συνιστά μία από τις βασικές προγραμματιστικές δομές και, όπως δείχνουν διάφορες έρευνες, παρουσιάζει ιδιαίτερα διδακτικά προβλήματα (Soloway et al., 1983 / Laborde et al., 1985 / Kessler & Anderson, 1986 / Soloway et al., 1989 / Dagdilelis et al., 1990 / Γρηγοριάδου κ.α., 2004).

Από τις έρευνες προκύπτει ότι οι αρχάριοι προγραμματιστές δεν χρησιμοποιούν αυθόρμητα την επαναληπτική διαδικασία για να λύσουν ένα πρόβλημα (Dagdilelis et al., 1990). Όπως είδαμε ήδη, η επαναληπτική διαδικασία της οποίας ο αριθμός επαναλήψεων δεν είναι εκ των προτέρων γνωστός (ακαθόριστη επανάληψη) μπορεί να διατυπωθεί με δύο διαφορετικές μεθόδους, καθεμία από τις οποίες επηρεάζει τη συμπεριφορά του (αρχάριου κυρίως) προγραμματιστή και τη σημασία ή την πολυπλοκότητα του προγράμματος:

1. Τη μέθοδο με **σώμα βρόχου – συνθήκη ελέγχου** (που διατυπώνεται π.χ. στην Pascal με την εντολή ...).
2. Τη μέθοδο με **συνθήκη ελέγχου – σώμα βρόχου** (που διατυπώνεται π.χ. στην Pascal με την εντολή ...).

Οι παραπάνω μέθοδοι αντιστοιχούν σε δύο διαφορετικούς σχεδιασμούς του αλγορίθμου με την επίτευξη του ίδιου στόχου:

1. Μεταβλητές διαδικασίας – μεταβλητή ελέγχου
2. Μεταβλητή ελέγχου – μεταβλητές διαδικασίας

Στην πρώτη περίπτωση πρέπει να περιγραφεί η δράση (action) και στη συνέχεια να γίνει ο έλεγχος (test), ενώ στη δεύτερη περίπτωση ισχύει το αντίθετο (Samurçay, 1987 / Rogalski & Vergnaud, 1987).

Κάθε ένας από τους παραπάνω τρόπους δεν παρουσιάζει τα ίδια διδακτικά προβλήματα, ούτε οικοδομείται ως έννοια με την ίδια ευκολία. Οι έρευνες που έχουν γίνει πάνω στα νοητικά μοντέλα των μαθητών δείχνουν ότι οι αυθόρμητοι συλλογισμοί τους και οι αναπαραστάσεις που διαθέτουν σχετικά με την επανάληψη έχουν κάποια συγκεκριμένη δομή και σταθερή σειρά έκφρασης: περιγραφή της δράσης, μετρητής επαναλήψεων, προσδιορισμός της επανάληψης και τέλος συνθήκη ελέγχου (Samurçay, 1987). Επιπρόσθετα, σημαντικό διδακτικό πρόβλημα φαίνεται ότι συνιστά η πρόβλεψη και διατύπωση της συνθήκης ελέγχου, η οποία και δεν εμφανίζεται αυθόρμητα αφού στην

«επεξεργασία με το χέρι» ή στις συνήθεις (μη προγραμματιστικές) δραστηριότητες των σπουδαστών υπονοείται.

Σύμφωνα λοιπόν με τα προηγούμενα, ο τύπος επαναληπτικής δομής που ταιριάζει περισσότερο με τις αρχικές ιδέες των σπουδαστών είναι ο ... («μετα»-ελεγχόμενη επανάληψη), ενώ αντιθέτως η δομή ... («προ»- ελεγχόμενη επανάληψη) ακολουθεί αντίθετη πορεία συλλογισμού και για το λόγο αυτό προκαλεί περισσότερα προβλήματα κατανόησης.

Επιπλέον, κατά την υλοποίηση της δομής ... η δράση μπορεί να μην εκτελεστεί ποτέ αφού είναι δυνατόν η συνθήκη ελέγχου να γίνει αληθής κατά την πρώτη επανάληψη. Το γεγονός αυτό έρχεται σε αντίθεση με την αναπαράσταση της επαναληπτικής διαδικασίας που αρχικώς διαθέτουν οι σπουδαστές. Αντίθετα, με την εντολή ... η δράση εκτελείται τουλάχιστον μία φορά.

Ένα άλλο σημαντικό διδακτικό πρόβλημα τίθεται από την ίδια την διατύπωση του σώματος του βρόχου. Ο μαθητής οφείλει να προσδιορίσει αφενός τις εμπλεκόμενες μεταβλητές και αφετέρου τη σχέση ανάμεσά τους, η οποία πρέπει να διατηρείται σε όλη τη ζωή του βρόχου (Rogalski, 1990). Επιπρόσθετα, οι πράξεις πάνω στις μεταβλητές της επανάληψης δεν είναι πάντα του ίδιου βαθμού δυσκολίας. Από τις έρευνες φαίνεται ότι τα λάθη απόδοσης αρχικών τιμών παρουσιάζονται πιο συχνά από τα λάθη ενημέρωσης ή ελέγχου (Samurçay, 1987 / Spohrer & Soloway, 1989).

Όπως είδαμε πιο πριν, όταν είναι εκ των προτέρων γνωστός ο αριθμός των επαναλήψεων, χρησιμοποιείται η εντολή (πρόγονος της εντολής αυτής υπήρξε η εντολή της Fortran), η οποία απαντάται πολύ συχνά στον προγραμματισμό αφού αποτελεί δομή όλων πρακτικά των γλωσσών. Αντίθετα, δεν ισχύει το ίδιο για την ..., η οποία χρησιμοποιείται σε περιορισμένο αριθμό γλωσσών.

Σε καθαρά συντακτικό επίπεδο, η είναι πιο απλή από την ..., αφού δεν ελέγχει καμία συνθήκη, ενώ παράλληλα είναι πιο εύκολο να δούμε τον αριθμό των επαναλήψεων που πραγματοποιεί. Παρότι έχει σχετικά απλό συντακτικό, το συντακτικό αυτό διαφοροποιείται από γλώσσα σε γλώσσα. Για παράδειγμα στην Basic ή στη Fortran μπορούμε να ελέγξουμε το βήμα της επανάληψης, ενώ αυτό δεν ισχύει σε πολλές εκδόσεις της Pascal. Η τυπική μορφή της εντολής αυτής σε συμβολισμό BNF είναι:

<μεταβλητή> := *<αρχική - τιμή>* *<βήμα>* *<τελική - τιμή>* *<εντολή>*

Υπάρχουν αρκετά προβλήματα που πρέπει να λύσουμε ώστε να γίνει κατανοητός ο τρόπος με τον οποίο δουλεύει η [illegible] σε κάθε γλώσσα (Horowitz, 1984): Ποιος είναι ο τύπος των τιμών που παίρνει η <μεταβλητή>, πόσο σύνθετες μπορεί να είναι οι παραστάσεις <τελική - τιμή>, <αρχική - τιμή>, <βήμα>, και ποιος είναι ο τύπος των αποτελεσμάτων τους.

Ακόμη, πόσο συχνά υπολογίζονται <τελική - τιμή> και <βήμα> όσο συνεχίζεται η επανάληψη, πότε ελέγχεται η <μεταβλητή> σε σχέση με την <τελική - τιμή>, εάν μπορεί η <μεταβλητή> να αλλάξει από μία ανάθεση τιμής μέσα στο βρόχο, ποια είναι η τιμή της <μεταβλητή> μετά τον τερματισμό του βρόχου και εάν επιτρέπεται η μεταφορά της μέσα στο βρόχο ή έξω από αυτόν.

9.3. Διδακτικές παρεμβάσεις για την επαναληπτική δομή

Για την αντιμετώπιση των διδακτικών προβλημάτων που θέτει η επαναληπτική διαδικασία έχουν προταθεί διάφοροι τύποι διδακτικών παρεμβάσεων, με χρήση ποικίλων διδακτικών στρατηγικών. Βασικός άξονας αυτών των παρεμβάσεων είναι η εμπλοκή των σπουδαστών στη διαδικασία επίλυσης προβλημάτων – τα οποία εμπεριέχουν διάφορους τύπους δυσχερειών (ή διδακτικών εμποδίων) – με στόχο να οικοδομήσουν κατάλληλες αναπαραστάσεις και νοητικά μοντέλα για τις διάφορες δομές επανάληψης.

Ωστόσο, έρευνες που έγιναν μετά από την εφαρμογή αυτών των παρεμβάσεων δείχνουν ότι οι αρχάριοι προγραμματιστές ξαναγυρνούν στις αρχικές αναπαραστάσεις τους, οι οποίες έχουν κύριο χαρακτηριστικό τη βήμα προς βήμα οικοδόμηση των επαναληπτικών διαδικασιών (Laborde et al., 1985).

Με άλλα λόγια, τα νοητικά μοντέλα των αρχάριων προγραμματιστών βασίζονται σε μια αναπαράσταση της διαδοχής των δράσεων και όχι σε μια αναπαράσταση της σχέσης ανάμεσα στις διαφορετικές καταστάσεις των μεταβλητών. Τυπικό παράδειγμα κώδικα αρχάριου προγραμματιστή είναι το παρακάτω:

```
......
sum:= 0+ number
counter:= 1
sum:= number1 + number2
counter:= 2
[illegible]
......
```

Αυτό το τμήμα κώδικα υποδεικνύει δυσκολίες δύο τύπων. Η πρώτη αφορά στην οικοδόμηση του σώματος του βρόχου (`sum:= sum + number` και `counter:= counter + 1`) και η δεύτερη αφορά στον προσδιορισμό των μεταβλητών. Ο αρχάριος τείνει να χρησιμοποιεί διαφορετικά ονόματα σε κάθε βήμα εκτέλεσης για να ονομάσει την ίδια λειτουργική μεταβλητή.

Συνεπώς, η οικοδόμηση της επαναληπτικής διαδικασίας απαιτεί έμφαση σε συλλογισμούς με όρους ***καταστάσεων*** και όχι με όρους ***εντολών*** (statements). Στην εντολή επανάληψης είναι απαραίτητη η σωστή αναπαράσταση τόσο της δράσης του προγράμματος, όσο και των αντίστοιχων μεταβλητών.

Για παράδειγμα, η δομή ... , απαιτεί τη νοητική διαχείριση της μεταβλητής ελέγχου (αριθμός επαναλήψεων), καθώς και τις μεταβλητές και τους λογικούς (Boolean) τελεστές (για αποτίμηση της παράστασης που ακολουθεί το).

Ένα άλλο σημαντικό εμπόδιο που εμφανίζεται κατά τη μάθηση της επανάληψης και των αντίστοιχων μεθόδων υλοποίησης προγραμμάτων με επανάληψη και αναδρομή (recursion) μπορεί να διατυπωθεί με δύο διαφορετικές μορφές:

Αφενός, με έλεγχο της σχέσης ανάμεσα στην εκτέλεση πράξεων (οι «υπολογισμοί») με το δυναμικό χαρακτήρα της και στη διαδοχή των καταστάσεων (οι «σχέσεις») με το στατικό της χαρακτήρα φαίνεται να είναι στο κέντρο της πρόσκτησης των επαναληπτικών και αναδρομικών διαδικασιών.

Αφετέρου, η μεταβλητή στον προγραμματισμό πρέπει να αντιμετωπιστεί ως μία συνάρτηση εκτέλεσης: μια τέτοια σχέση ανάμεσα σε συνάρτηση και μεταβλητή δεν είναι ίδιον του προγραμματισμού, αλλά δεν εμφανίζεται ποτέ στο «γνωστικό παρελθόν» των σπουδαστών.

Η έννοια της *σταθεράς* (invariant) που βρίσκεται στο επίκεντρο της παρουσίασης της επαναληπτικής διαδικασίας (Arsac, 1980) πολύ δύσκολα μπορεί να αποκτήσει νόημα χωρίς τον έλεγχο αυτής της διπλής προοπτικής.

Η συγγραφή της κατάστασης του συνόλου των μεταβλητών ενός προγράμματος με πυρήνα την επαναληπτική δομή στο τέλος κάθε εντολής και στο τέλος της εκτέλεσης του σώματος του βρόχου μπορεί να παράσχει τις βάσεις ώστε:

- να γίνει η διάκριση των μεταβλητών που κρατούν την τιμή τους σε όλη τη διάρκεια της εκτέλεσης και αυτών που αλλάζει η τιμή τους κατά την εκτέλεση του βρόχου,

- να αρχίσει μια εργασία αφενός πάνω στον εμπειρικό προσδιορισμό των αναλλοίωτων σχέσεων ανάμεσα σε τιμές και μεταβλητές και αφετέρου στην εγκυρότητα (validation) της παραδοχής μέσω ενός συλλογισμού που αφορά τους μετασχηματισμούς των μεταβλητών στο σώμα του βρόχου.

Ο αλγόριθμος για την πρόσθεση των πρώτων 100 ακεραίων αριθμών

Στην ενότητα αυτή προσεγγίζεται διδακτικά ο αλγόριθμος για την πρόσθεση των πρώτων 100 ακεραίων αριθμών. Μπορούμε να δούμε τρεις απλούς επαναληπτικούς αλγορίθμους επίλυσης αυτού του προβλήματος.

Ο πρώτος αλγόριθμος (Πίνακας 8) χρησιμοποιεί την εντολή FOR, ο δεύτερος (Πίνακας 9) την εντολή REPEAT ... UNTIL, και ο τρίτος (Πίνακας 11) την εντολή WHILE.

1η περίπτωση (υλοποίηση σε Basic και Pascal)

```
10 sum=0
20      i=1    100      1
30 sum=sum+i
40      i
50      sum
```

```
sum:=0;
    i:=1    100
 sum:=sum+I;
       (sum)
```

Πίνακας 8 Αλγόριθμος πρόσθεσης των 100 πρώτων ακεραίων (με For)

2η περίπτωση (Pascal, με δύο ελαφρώς διαφορετικούς αλγορίθμους)

```
sum:=0;
i:=0;

 i:=i+1;
 sum:=sum+i
    i=100;
       (sum)
```

```
sum:=0;
i:=1;

 sum:=sum+i;
 i:=i+1
    i>100;
       (sum)
```

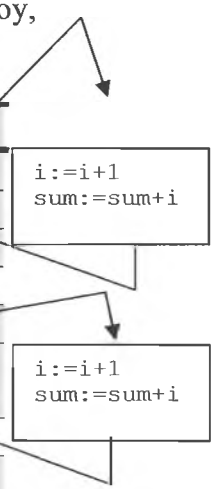
Πίνακας 9 Αλγόριθμος πρόσθεσης των 100 πρώτων ακεραίων (με Repeat Until)

Και στις δύο προηγούμενες περιπτώσεις, που υλοποιήθηκαν με τις εντολές FOR και REPEAT ... UNTIL, ο αλγόριθμος περιέχει τα ακόλουθα στοιχεία:

- απόδοση αρχικής τιμής (εδώ: μηδενισμός) της μεταβλητής sum (ΑΘΡΟΙΣΜΑ),
- επανάληψη 100 φορές των επόμενων πράξεων (αφού γίνει χρήση μιας μεταβλητής i που αυξάνεται από το 1 στο 100): πρόσθεση της sum με την i, απόδοση αυτής της τιμής στην sum, εμφάνιση ή εκτύπωση του αποτελέσματος.

Αυτή η διαδικασία μπορεί να αναπαρασταθεί σε έναν πίνακα (Πίνακας 10) που δείχνει την εξέλιξη του προγράμματος βήμα προς βήμα και την κατάσταση των μεταβλητών.
Μια σημαντική παρατήρηση που αφορά τον επόμενο Πίνακα 10 είναι ότι πρέπει να γίνει η διάκριση ανάμεσα στη μεταβλητή `i` που ορίζει μια μεταβλητή στην Πληροφορική και στη μεταβλητή k, που είναι μια μεταβλητή με τη μαθηματική έννοια. Επομένως, υπάρχει και γίνεται διάκριση ανάμεσα στη μεταβλητή `i` και το περιεχόμενό της (Delannoy, 1996).

Κατάσταση	**Τιμή της i**	**Τιμή της sum**
Πριν την είσοδο στο βρόχο	0	0
Μετά από 1 εκτέλεση του βρόχου	1	1
Μετά από 2 εκτελέσεις του βρόχου	2	1+2
Μετά από k εκτελέσεις του βρόχου	k	1+2+ … + k
Μετά από k+1 εκτελέσεις του βρόχου	k+1	1+ … + k +(k+1)
Μετά από 100 εκτελέσεις του βρόχου	100	1+2+ … +99+100



Πίνακας 10 Εξέλιξη προγράμματος και κατάσταση μεταβλητών

3η περίπτωση (με εντολή While … Do σε Pascal)

```
 sum:=0;
i:=0;
      i<=100 do

  sum:=sum+i;
  i:=i+1
    ;
        (sum)
```

Πίνακας 11 Αλγόριθμος πρόσθεσης των 100 πρώτων ακεραίων (με While Do)

10. Μεθοδολογία της έρευνας

10.1. Το αντικείμενο

Η επιλογή του αντικειμένου μελέτης έγινε επειδή η **έννοια της μεταβλητής**, της **εντολής ανάθεσης τιμής σε μεταβλητή**, των **δομών ελέγχου** και της **έννοιας της επανάληψης** αποτελούν πρωταρχικές έννοιες σε όλα τα προγραμματιστικά περιβάλλοντα. Οι αναπαραστάσεις των σπουδαστών σχετικά με αυτές τις έννοιες επηρεάζουν καθοριστικά τη λειτουργική εφαρμογή αυτών των εννοιών όπως και πιο σύνθετων δομών (πίνακες, συναρτήσεις, κλάσεις κ.λπ.) για την επίλυση προβλημάτων.

10.2. Το δείγμα

Η έρευνα έγινε στο Τ.Ε.Ι. Μεσολογγίου στο τμήμα της Ε.Π.Δ.Ο. (Εφαρμογές της Πληροφορικής στη Διοίκηση και στην Οικονομία) με δείγμα ομάδα σπουδαστών που παρακολουθεί το εργαστήριο του μαθήματος "Προγραμματισμός Υπολογιστών ΙΙ", στο οποίο διδάσκεται η γλώσσα C++. Η ομάδα περιλαμβάνει 42 σπουδαστές. Ο διδάσκων είχε προχωρήσει την ύλη του μαθήματος ως την κατανόηση των κλάσεων στην γλώσσα C++. Άρα οι έννοιες της **μεταβλητής**, της **εντολής ανάθεσης τιμής σε μεταβλητή**, των **δομών ελέγχου** και της **έννοιας της επανάληψης** είχαν διδαχθεί ήδη από το προηγούμενο εργαστήριο του "Προγραμματισμού Υπολογιστών Ι" όπως επίσης είχαν αναφερθεί και στο ξεκίνημα των μαθημάτων για το εργαστήριο "Προγραμματισμός Υπολογιστών ΙΙ".

10.3. Το ερωτηματολόγιο

Η έρευνα διεξήχθη με τη μορφή ανώνυμου γραπτού ερωτηματολογίου. Στους σπουδαστές δόθηκαν **δέκα έργα** σε μορφή προγραμμάτων κωδικοποιημένων σε γλώσσα C++. Κάθε έργο περιλάμβανε τόσο την κατανόηση του προβλήματος όσο και του προγράμματος που χρησιμοποιήθηκε για την επίλυσή του. Το πρόγραμμα του **1ου έργου** δεν δόθηκε ολοκληρωμένο. Οι σπουδαστές κλήθηκαν να συμπληρώσουν τις κενές

γραμμές που αφορούσαν στις δηλώσεις των μεταβλητών του προγράμματος. Επιχειρήθηκε έτσι να διερευνηθεί κατά πόσο οι σπουδαστές μπορούν να διακρίνουν τις μεταβλητές ενός προγράμματος, τον τύπο τους και το ρόλο τους στο πρόγραμμα. Με τα προγράμματα του **2ου** και **3ου έργου** ζητήθηκε από τους σπουδαστές να δώσουν το τελικό αποτέλεσμα που θα εμφανιστεί στην οθόνη του υπολογιστή μετά την εκτέλεση του προγράμματος. Επιχειρήθηκε έτσι να διερευνηθεί κατά πόσο οι σπουδαστές μπορούν να χειριστούν τις μεταβλητές ενός προγράμματος. Με τα προγράμματα **του 4ου και 5ου έργου** ζητήθηκε από τους σπουδαστές να συγκρίνουν αν τα προγράμματα που δίνονται είναι ή όχι ισοδύναμα. Επιχειρήθηκε έτσι να διερευνηθεί κατά πόσο οι σπουδαστές μπορούν να χειριστούν τις μεταβλητές ενός προγράμματος, κατά πόσο κατανοούν την έννοια της εντολής εκχώρησης και ποιες είναι οι αντιλήψεις και οι παρανοήσεις τους. Με το πρόγραμμα του **6ου έργου** ζητείται από τους σπουδαστές να μελετήσουν τα δύο κομμάτια του κώδικα ως προς την απόδοση τιμής σε σταθερά και την απόδοση τιμής σε παράσταση, με σκοπό να μελετηθούν οι αντιλήψεις τους ως προς τις μεταβλητές και της εντολής ανάθεσης τιμής σε μεταβλητή. Με το πρόγραμμα του **7ου έργου** ζητείται από τους σπουδαστές να διερευνήσουν αν το κομμάτι του κώδικα έχει κάποιο λάθος. Επιχειρήθηκε έτσι να διερευνηθεί κατά πόσο οι σπουδαστές μπορούν να διακρίνουν αν οι τελεστές σύγκρισης είναι συντακτικά σωστοί ή όχι. Το πρόγραμμα του **8ου έργου** δε δόθηκε ολοκληρωμένο. Οι σπουδαστές κλήθηκαν να το ολοκληρώσουν. Επιχειρήθηκε έτσι να διερευνηθεί κατά πόσο οι σπουδαστές αντιλαμβάνονται την δομή του ελέγχου. Με τα προγράμματα του **9ου** και **10ου έργου** ζητείται από τους σπουδαστές να εξηγήσουν τι θα συμβεί αν για το πρώτο πρόγραμμα δοθεί μηδενική ή αρνητική τιμή και για το δεύτερο πότε θα σταματήσει η επαναληπτική διαδικασία. Επιχειρήθηκε έτσι να διερευνηθεί κατά πόσο οι σπουδαστές αντιλαμβάνονται την επαναληπτική δομή και τους περιορισμούς που μπορεί να προκύψουν κατά την εκτέλεση των προγραμμάτων. Το ερωτηματολόγιο ήταν το εξής:

ΕΡΩΤΗΜΑΤΟΛΟΓΙΟ ΓΙΑ ΤΗΝ ΔΙΕΡΕΥΝΗΣΗ ΤΩΝ ΑΝΑΠΑΡΑΣΤΑΣΕΩΝ ΤΩΝ ΣΠΟΥΔΑΣΤΩΝ ΤΟΥ ΤΜΗΜΑΤΟΣ Ε.Π.Δ.Ο.

Έργο 1. Να συμπληρώσετε τις δηλώσεις μεταβλητών στο παρακάτω απόσπασμα προγράμματος.

```
# include <iostream.h>
void main (void)
{
    ____ i=32;
    ____ misthos = 25000.75;
    ____ epostasi_seliniis = 236857;
}
```

Έργο 2. Τι αναμένετε να τυπωθεί στην οθόνη κατά την εκτέλεση του παρακάτω προγράμματος και γιατί; Εξηγήστε εν συντομία.

```
# include <iostream.h>
void main (void)
{
    int x=5;
    int y=x;

    x=x+5;
    y=y+5;
    cout << "x:" << x <<endl;
    cout << "y:" << y <<endl;
}
```

Έργο 3. Τι αναμένετε να εμφανιστεί στην οθόνη και γιατί κατά την εκτέλεση του παρακάτω τμήματος προγράμματος;

```
# include <iostream.h>
void main (void)
{
    int x=5;
    int a=2;
    cout << "x=" << x << "a=" << a;

    y=2*x-a/2;
    z=x+y;
    cout << a << z << endl;
}
```

Έργο 4. Τα παρακάτω τμήματα προγράμματος είναι ισοδύναμα ή όχι και γιατί;

```
# include <iostream.h>
void main (void)
{
    int x;
    float y;
    char a [100];

    x=0;
    y=0.231;
    a="ΑΣΚΗΣΗ";
}
```

```
# include <iostream.h>
void main (void)
{
    int x=45;
    float y=0.231;
    char a="ΑΣΚΗΣΗ";
}
```

Έργο 5. Τα παρακάτω τμήματα του προγράμματος είναι ισοδύναμα ή όχι και γιατί;

```
# include <iostream.h>
void main (void)
{
    int a=10;
    int b=5;

    a=b;
    b=a;
    cout << "Τώρατο α είναι:" << a << "και
b:" << b <<endl;
}
```

```
# include <iostream.h>
void main (void)
{
    int a=10;
    int b=5;

    b=a;
    a=b;
    cout << "Τώρατο α είναι:" << a << "και
b:" << b <<endl;
}
```

Έργο 6. Έχουν λάθος τα επόμενα τμήματα του προγράμματος; Αν ναι, αιτιολογείστε σύντομα.

```
# include <iostream.h>
void main (void)
{
    int a=2;
    int b=3;

    5= a+b;
}
```

```
# include <iostream.h>
void main (void)
{
    int a=2;
    int b=3;

    a+b=5;
}
```

Έργο 7. Το παρακάτω κομμάτι κώδικα είναι σωστό ή όχι; Δικαιολογήστε σύντομα την απάντησή σας

```
# include <iostream.h>
void main (void)
{
    int x;
    float y;
    int a;
```

```
if (x><y)
    c=x+y;
else if (x<>y)
    c=x*y;
cout << "Το c έχει τιμή:" << c <<endl;
}
```

Έργο 8. Συνεχίστε το παρακάτω κομμάτι κώδικα ώστε όταν διαβαστεί ένας ακέραιος να εκτυπωθεί το αντίστοιχο γράμμα της αλφαβήτου. Ο ακέραιος έχει τιμή 1 ή 2 ή 3, σε κάθε άλλη περίπτωση να εκτυπώνεται η λέξη «άγνωστος».

```
# include <iostream>
void main (void)
{
    if (a=1)
        cout << "A" << endl;
    .
    .
    .
    .
```

Έργο 9. Το πρόγραμμα ζητά από το χρήστη έναν αριθμό στον οποίο θα τερματίσει ο βρόχος. Τι θα γίνει στο παρακάτω κομμάτι κώδικα αν δοθεί μηδενική ή αρνητική τιμή; Εξηγήστε σύντομα.

```
# include <iostream.h>
void main (void)
{
    int counter;
    int end_value;
    cout << "Δώσε την τελική τιμή και πάτησε Enter:";
    cin >> end_value;
    for (counter=1; counter<= end_value; counter++)
        cout << counter << " ";
}
```

Έργο 10. Στο παρακάτω κομμάτι κώδικα πότε τερματίζει η επαναληπτική διαδικασία; Εξηγήστε σύντομα.

```
#include <iostream>
void main (void)
{
  int n;
  while (n=0)
      cout << n;
      cin >> n;
}
```

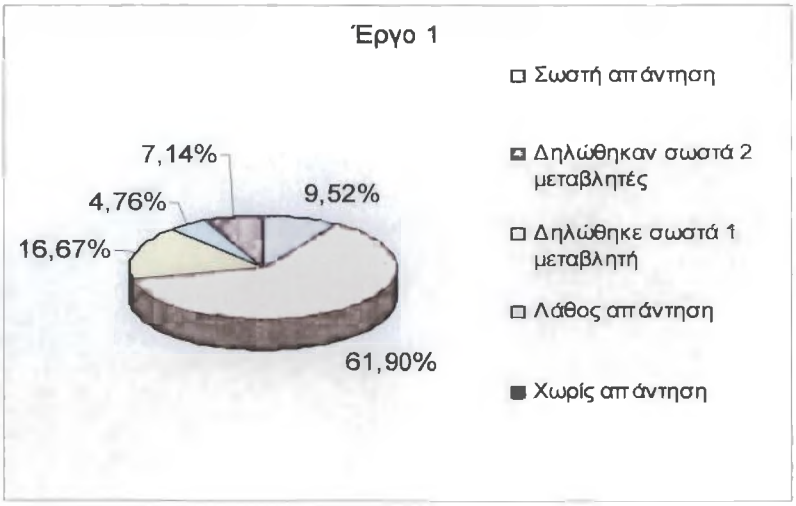
10.4. Περιγραφική ανάλυση

Έργο 1

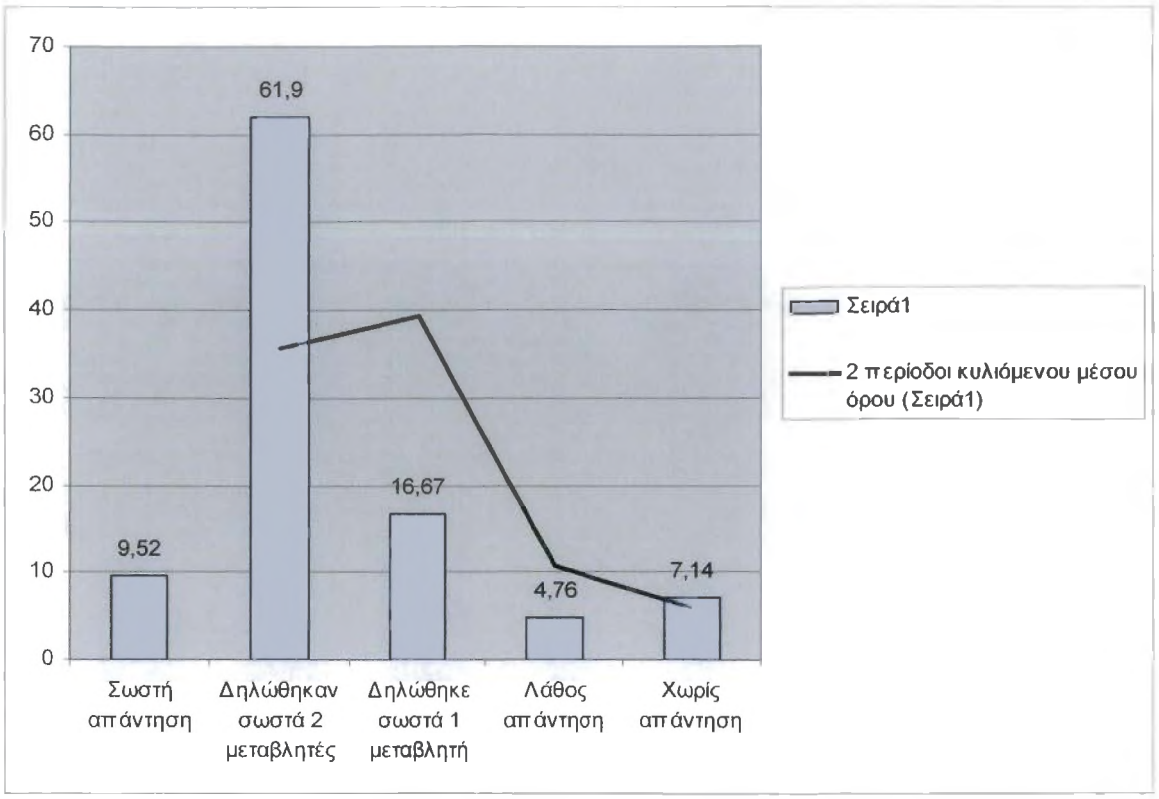
Οι απαντήσεις στο 1° έργο δίνονται στον Πίνακα 12. Σωστή απάντηση έδωσαν μόνο 4 σπουδαστές. Σε ένα μεγάλο ποσοστό (61,90%) οι σπουδαστές αντιμετώπισαν δυσκολία στο να δηλώσουν σωστά την μεταβλητή apostasy_selinis. Την δήλωναν είτε ως integer είτε ως float πράγμα που είναι λάθος. Επίσης ένα σημαντικό ποσοστό από αυτούς (16,67%) δήλωσαν σωστά μόνο την μία μεταβλητή που ήταν η ilikia ενώ τις άλλες δύο αντί να τις δηλώσουν σωστά τις δήλωναν ως integer. Τέλος 2 σπουδαστές δίνουν λανθασμένες απαντήσεις και 3 από αυτούς επιλέγουν να μην απαντήσουν καθόλου στο 1° έργο. Έτσι βλέποντας το Γράφημα 2, όπου χρησιμοποιείται η στατιστική μέθοδος της ανάλυσης της παλινδρόμησης που με αυτήν επεκτείνεται μια γραμμή τάσης στο γράφημα ώστε να γίνεται εύκολα η πρόληψη των μελλοντικών τιμών, βλέπουμε ξεκάθαρα σε ποιο κομμάτι οι σπουδαστές δυσκολεύονται να απαντήσουν.

α/α	Διαδικασία	Συχνότητα (N=42)	Ποσοστό %
1	Σωστή απάντηση	4	9,52
2	Δηλώθηκαν σωστά 2 μεταβλητές	26	61,9
3	Δηλώθηκε σωστά 1 μεταβλητή	7	16,67
4	Λάθος απάντηση	2	4,76
5	Χωρίς απάντηση	3	7,14

Πίνακας 12 Κατηγορίες απαντήσεων στο 1° έργο



Γράφημα 1 Ποσοστά απαντήσεων στο Έργο1



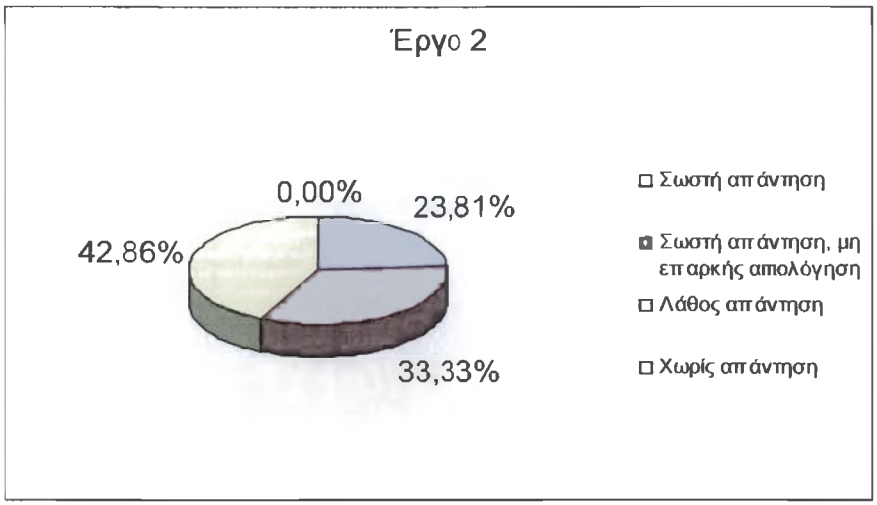
Γράφημα 2 Ραβδόγραμμα με τα ποσοστά των απαντήσεων στο 1ο έργο

Έργο 2

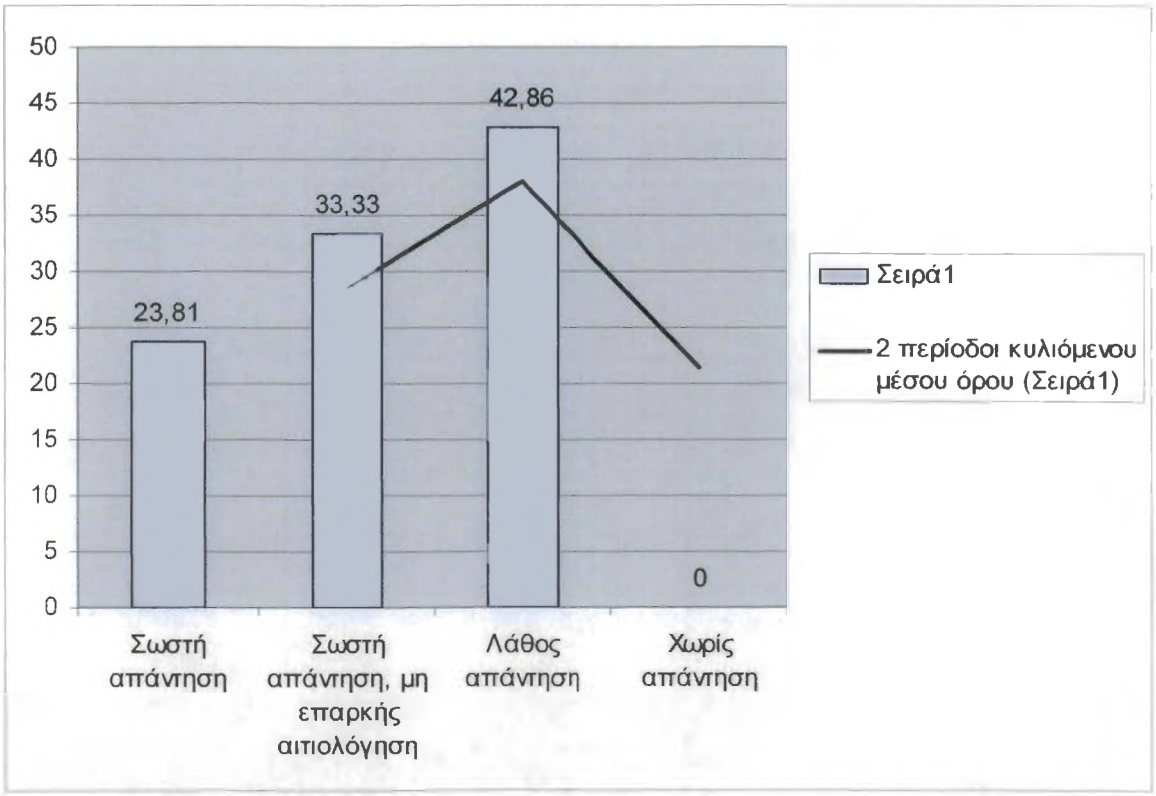
Στον πίνακα 13 ταξινομούνται οι απαντήσεις για το 2ο έργο. Σωστή απάντηση έδωσαν 10 σπουδαστές σε ποσοστό 23,81%. Ένα σημαντικό ποσοστό από αυτούς (33,33%) απάντησε μεν σωστά αλλά δεν έδωσε επαρκή ή και καθόλου εξήγηση για την διαδικασία που ακολούθησαν ώστε να καταλήξουν στα σωστά συμπεράσματα. Όμως ένα μεγάλο ποσοστό από αυτούς (42,86%) έδωσε λανθασμένη απάντηση ως προς το αποτέλεσμα της εκτέλεσης του κώδικα αλλά και ως προς την εξήγηση της διαδικασίας που ακολουθεί ο κώδικας για να καταλήξει σε αυτά τα αποτελέσματα. Σε αυτό το έργο απάντησε το σύνολο των σπουδαστών στους οποίους δόθηκε το ερωτηματολόγιο. Έτσι όπως βλέπουμε και στο Γράφημα 4 οι σπουδαστές τείνουν να δώσουν μια μη επαρκή αλλά και λανθασμένη απάντηση στο 2ο έργο.

α/α	Διαδικασία	Συχνότητα (N=42)	Ποσοστό %
1	Σωστή απάντηση	10	23,81
2	Σωστή απάντηση, μη επαρκής αιτιολόγηση	14	33,33
3	Λάθος απάντηση	18	42,86
4	Χωρίς απάντηση	0	0 ,00

Πίνακας 13 Κατηγορίες απαντήσεων στο 2ο έργο



Γράφημα 3 Ποσοστά απαντήσεων στο Έργο 2



Γράφημα 4 Ραβδόγραμμα με τα ποσοστά των απαντήσεων στο 2^{ο} έργο

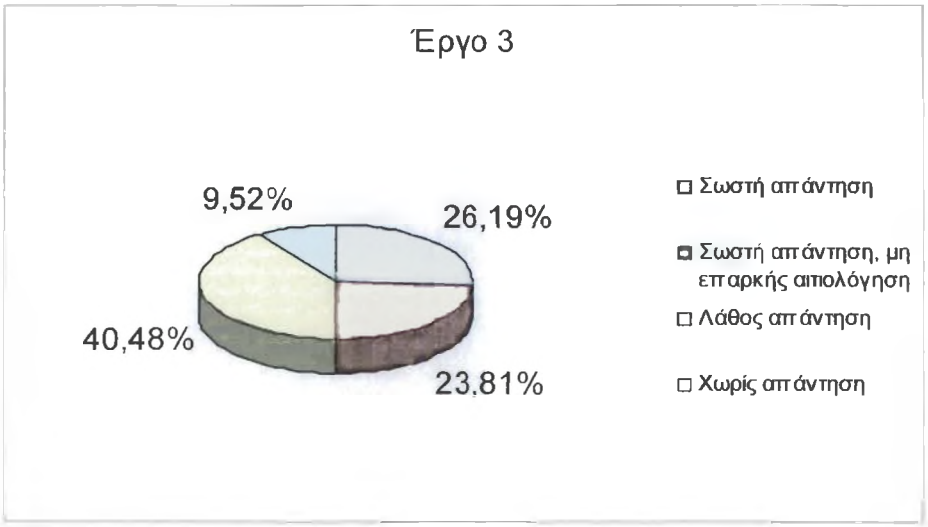
Έργο 3

Τα αποτελέσματα της έρευνας σχετικά με το 3^{ο} έργο ταξινομούνται στον Πίνακα 14. Σωστή απάντηση με επαρκή αιτιολόγηση για αυτό το έργο έδωσαν 11 σπουδαστές (26,19%). Σωστή απάντηση αλλά με ανεπαρκή αιτιολόγηση έδωσε το 23,81% από τους σπουδαστές. Από την μελέτη των απαντήσεων τους προκύπτει ότι αν και ήξεραν να

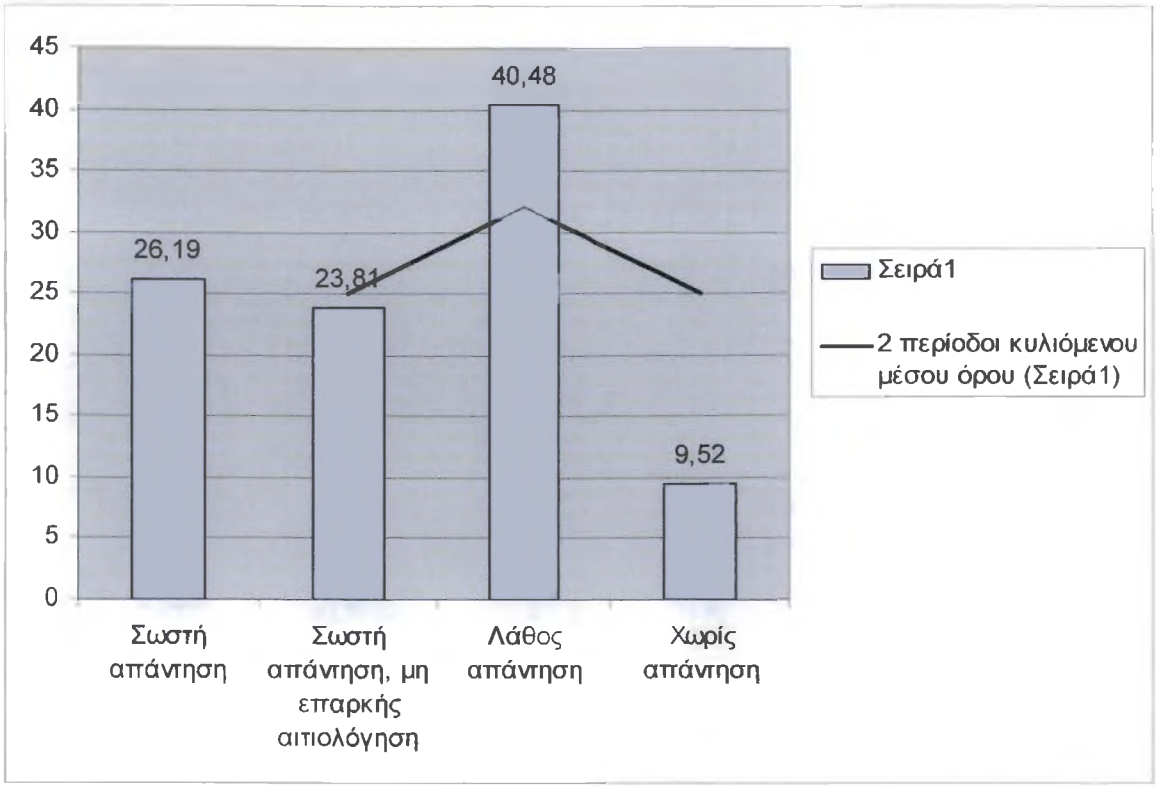
δώσουν την αιτιολόγηση προτίμησαν να μην τη δώσουν καθόλου. Αλλά όμως ένα μεγάλο ποσοστό, το 40,48%, έδωσε λανθασμένη απάντηση συγχέωντας για το τι πρέπει να εμφανιστεί στην οθόνη μετά την εκτέλεση του κώδικα δίνοντας απαντήσεις της μορφής: *«Θα τυπωθούν οι τιμές των y,z. Για y=3 και z=8.»*. Τέλος 4 σπουδαστές (9,52%) δε έδωσαν καμία απάντηση. Παρατηρώντας το Γράφημα 6 παρατηρούμε την τάση οι σπουδαστές να τείνουν προς την λανθασμένη απάντηση έχοντας προφανή δυσκολία στο να αντιληφθούν το τι τους ζητάει να βρουν το συγκεκριμένο έργο.

α/α	Διαδικασία	Συχνότητα (N=42)	Ποσοστό %
1	Σωστή απάντηση	11	26,19
2	Σωστή απάντηση, μη επαρκής αιτιολόγηση	10	23,81
3	Λάθος απάντηση	17	40,48
4	Χωρίς απάντηση	4	9,52

Πίνακας 14 Κατηγορίες απαντήσεων στο 3^{ο} έργο



Γράφημα 5 Ποσοστά απαντήσεων στο Έργο 3



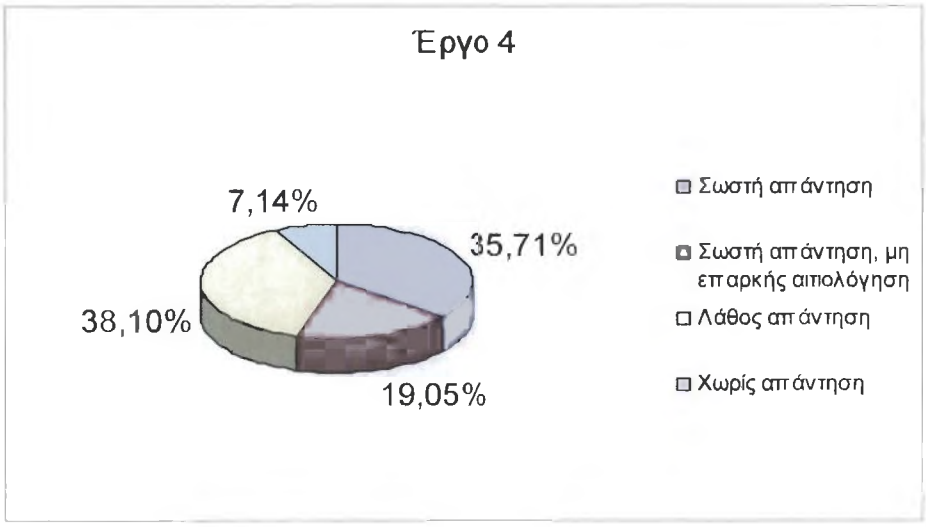
Γράφημα 6 Ραβδόγραμμα με τα ποσοστά των απαντήσεων στο 3ο έργο

Έργο 4

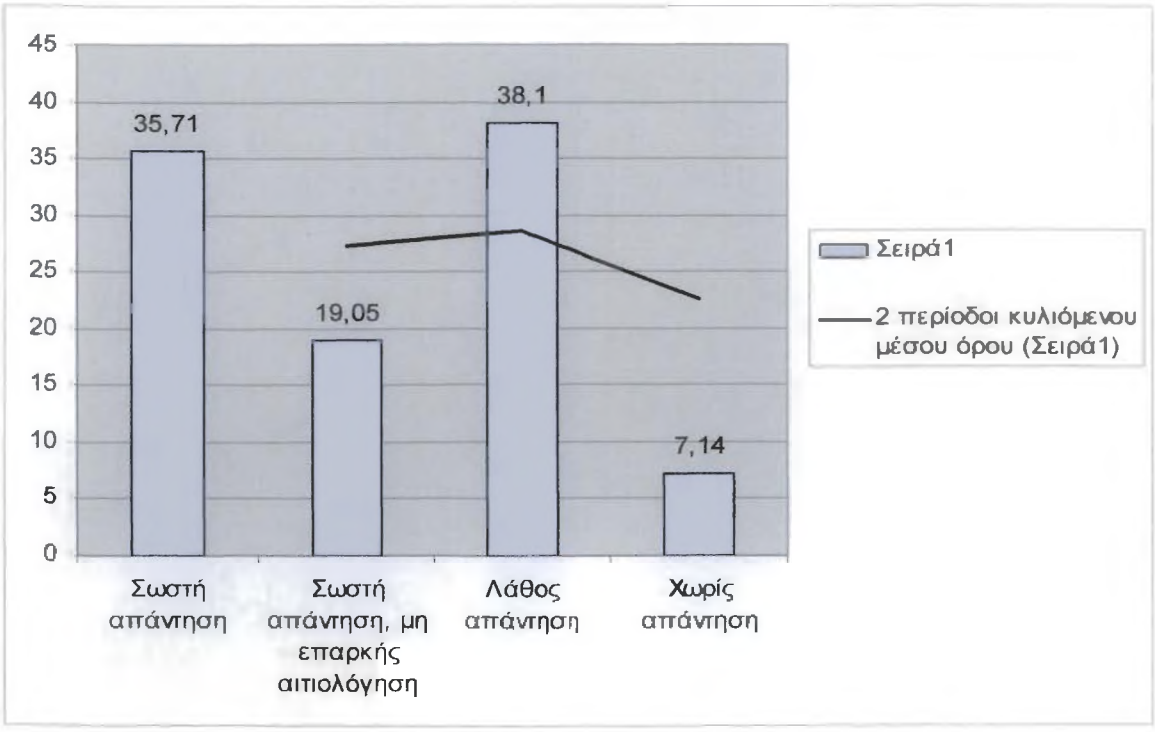
Στο 4ο έργο ζητείται από τους σπουδαστές να συγκρίνουν δύο κομμάτια κώδικα ώστε να βρουν αν είναι ισοδύναμα και να αιτιολογήσουν την απάντησή τους. 15 από αυτούς (35,71%) έδωσαν την σωστή απάντηση και επαρκή αιτιολόγηση. Ένα ποσοστό της τάξεως του 19,05% έδωσε μεν τη σωστή απάντηση αλλά δεν αιτιολόγησε επαρκώς. Προφανώς αυτοί οι σπουδαστές ήξεραν την αιτιολόγηση αλλά προτίμησαν να απαντήσουν μονολεκτικά ή με δυο λέξεις με απαντήσεις της μορφής: *«Ισοδύναμα»* ή *«Είναι ισοδύναμα»*. Οι 16 από τους σπουδαστές (38,1%) απάντησαν λανθασμένα. Οι λανθασμένες απαντήσεις χωρίζονταν σε δύο σκέλη, στις μονολεκτικές του τύπου: *«Όχι»* και *«Όχι δεν είναι»*, και σε αυτές που οι σπουδαστές έδιναν λάθος απάντηση αλλά και λάθος αιτιολόγηση όπως για παράδειγμα: *«Όχι, γιατί στο πρώτο κομμάτι κώδικα το z λαμβάνει μέχρι 10 χαρακτήρες»* ή *«Όχι γιατί το z="ΑΣΚΗΣΗ"; δεν είναι σωστό»*. Από το δεύτερο σκέλος των λανθασμένων απαντήσεων βγαίνει το συμπέρασμα ότι οι σπουδαστές έχουν πρόβλημα να κατανοήσουν ότι η εκχώρηση τιμών στις μεταβλητές μπορεί να γίνει και κατά την δήλωσή τους. Επίσης 3 σπουδαστές (7.14%) δεν απαντούν καθόλου σε αυτό το έργο. Παρατηρώντας το Γράφημα 8 βλέπουμε ότι η τάση μεταξύ της λανθασμένης και σωστής απάντησης τείνει να εξομαλυνθεί.

α/α	Διαδικασία	Συχνότητα (N=42)	Ποσοστό %
1	Σωστή απάντηση	15	35,71
2	Σωστή απάντηση, μη επαρκής αιτιολόγηση	8	19,05
3	Λάθος απάντηση	16	38,1
4	Χωρίς απάντηση	3	7,14

Πίνακας 15 Κατηγορίες απαντήσεων στο 4ο έργο



Γράφημα 7 Ποσοστά απαντήσεων στο Έργο 4



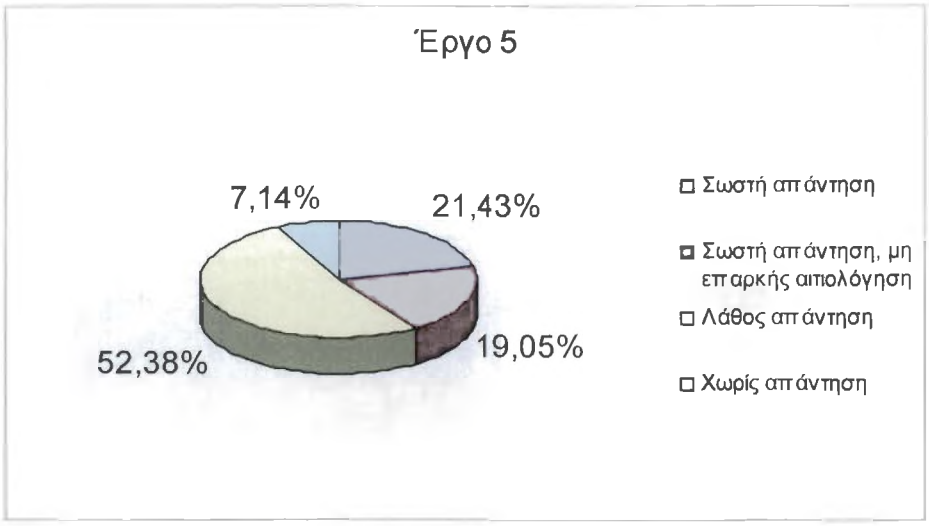
Γράφημα 8 Ραβδόγραμμα με τα ποσοστά των απαντήσεων στο 4ο έργο

Έργο 5

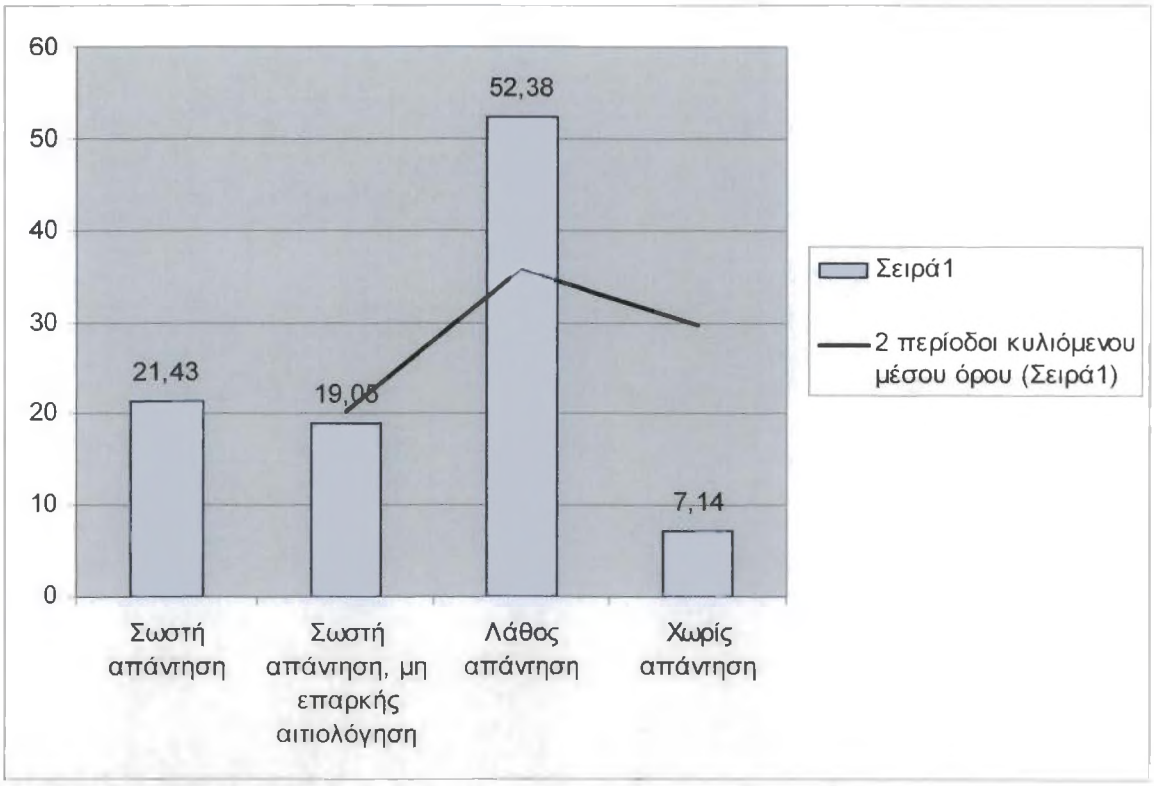
Στο 5^{ο} έργο ζητείται από τους σπουδαστές και πάλι να μελετήσουν δύο κομμάτια κώδικα ως προς την ισοδυναμία τους. Οι 9 (21,43%) από τους 42 σπουδαστές δίνουν τη σωστή απάντηση με μια επαρκή αιτιολόγηση. Οι 8 (19.05%) δίνουν τη σωστή απάντηση αλλά η αιτιολόγησή είναι ανεπαρκής έως και καμία. Οι σπουδαστές δίνουν απαντήσεις της μορφής: *«a=5, b=5»* για το πρώτο κομμάτι κώδικα και *«a=10, b=10»* για το δεύτερο ή απαντήσεις της μορφής: *«Ως προς το αποτέλεσμα δεν θα είναι ισοδύναμα γιατί αλλάζει η μεταβλητή»*. Παρατηρούμε ότι οι απαντήσεις αυτού του τύπου χωρίζονται σε δύο κατηγορίες. Στην πρώτη κατηγορία δίνουν το σωστό αποτέλεσμα από κάθε κομμάτι κώδικα αλλά επιλέγουν να μην δώσουν αιτιολόγηση γιατί βγαίνει αυτό το αποτέλεσμα. Στην δεύτερη κατηγορία αιτιολογούν σωστά την απάντηση τους αλλά δεν δίνουν τα αποτελέσματα του κώδικα. Λανθασμένη απάντηση δίνει η πλειοψηφία των σπουδαστών, το 52,38%. Οι απαντήσεις τους είναι της μορφής: *«Ναι είναι ισοδύναμα»* ή *«Είναι ισοδύναμα γιατί το b=a είναι ίδιο με το a=b»*. Προφανώς και εδώ έχουμε δύο τύπους απαντήσεων. Στην πρώτη περίπτωση οι σπουδαστές προτιμούν να δώσουν μια σύντομη απάντηση χωρίς αιτιολόγηση. Στην δεύτερη περίπτωση δίνουν μια αιτιολόγηση, λανθασμένη όμως. Από τον δεύτερο τύπο των απαντήσεων που δίνουν οι σπουδαστές καταλαβαίνουμε ότι έχουν δυσκολία να καταλάβουν την σειρά ακολουθίας των πράξεων συγχέωντας την εντολή εκχώρησης με την ισότητα στα μαθηματικά. Παρατηρώντας το ραβδόγραμμα συχνοτήτων στο Γράφημα 10 βλέπουμε ότι υπάρχει μια αυξανόμενη τάση προς την λανθασμένη απάντηση. Αυτή η τάση οφείλεται στο γεγονός ότι το μεγαλύτερο μέρος των σπουδαστών που συμμετείχε στην έρευνα αδυνατεί να κατανοήσει την εντολή εκχώρησης στον προγραμματισμό.

α/α	Διαδικασία	Συχνότητα (N=42)	Ποσοστό %
1	Σωστή απάντηση	9	21,43
2	Σωστή απάντηση, μη επαρκής αιτιολόγηση	8	19,05
3	Λάθος απάντηση	22	52,38
4	Χωρίς απάντηση	3	7,14

Πίνακας 16 Κατηγορίες απαντήσεων στο 5^{ο} έργο



Γράφημα 9 Ποσοστά απαντήσεων στο Έργο 5



Γράφημα 10 Ραβδόγραμμα με τα ποσοστά των απαντήσεων στο 5^{ο} έργο

Έργο 6

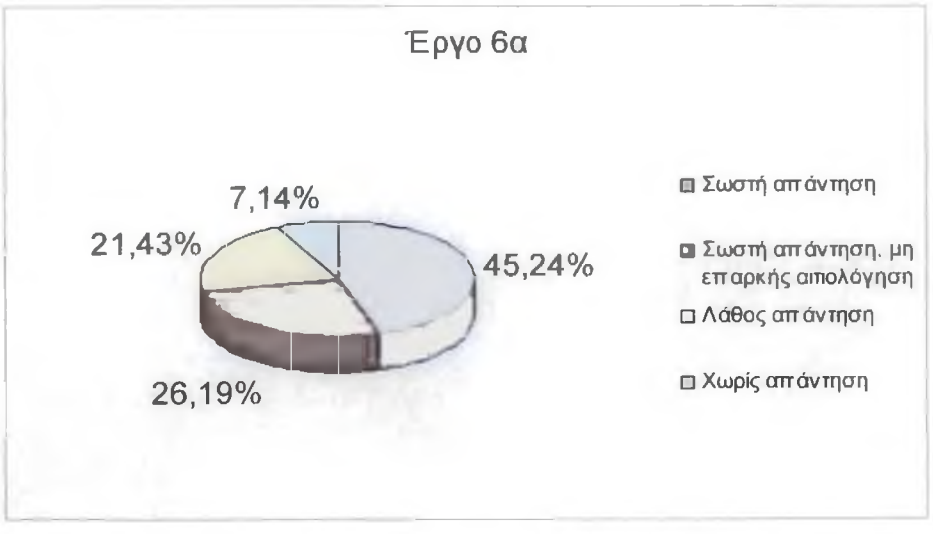
Στο 6^{ο} έργο ζητείται από τους σπουδαστές να βρουν το λάθος σε δύο κομμάτια κώδικα. Επειδή η στατιστική μελέτη ήταν δύσκολο να γίνει ταυτόχρονα και για τα δύο κομμάτια κώδικα αυτή η μελέτη χωρίστηκε σε δύο σκέλη, στο Έργο 6α και Έργο 6β.

Έργο 6α

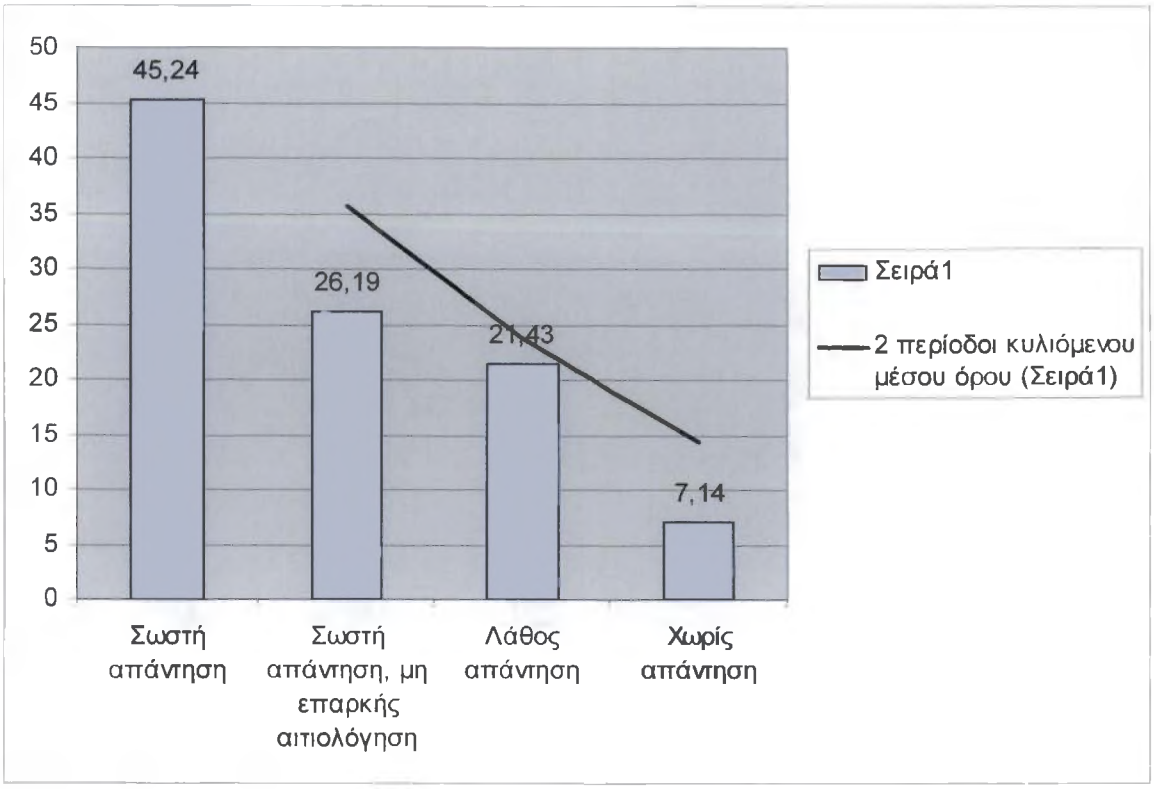
Σε αυτό το σκέλος του έκτου έργου απάντησαν και αιτιολόγησαν σωστά 19 σπουδαστές (45,24%). Οι 11 (26,19%) απάντησαν σωστά αλλά δεν έδωσαν επαρκή αιτιολόγηση. Οι απαντήσεις τους ήταν του τύπου: *«Το 5=a+b είναι λάθος»*. Από τις απαντήσεις τους καταλαβαίνουμε ότι αντιλήφθηκαν το λάθος αλλά προτίμησαν να μην αιτιολογήσουν γιατί το συγκεκριμένο κομμάτι κώδικα είχε λάθος. Οι 9 (21,43%) έδωσαν λανθασμένη απάντηση. Θεώρησαν ότι το κομμάτι κώδικα ήταν σωστό και οι απαντήσεις τους ήταν της μορφής: *«Είναι σωστό», «Όχι δεν έχει λάθος»*. Όλες οι λανθασμένες απαντήσεις ήταν χωρίς αιτιολόγηση. Τέλος 3 σπουδαστές (7,14%) προτίμησαν να μην απαντήσουν καθόλου στο συγκεκριμένο σκέλος. Όπως παρατηρούμε και στο Γράφημα 12 οι δύο κατηγορίες των σωστών απαντήσεων συγκεντρώνουν τα μεγαλύτερα ποσοστά που σημαίνει ότι η πλειοψηφία των σπουδαστών αντιλαμβάνεται το λάθος.

α/α	Διαδικασία	Συχνότητα (N=42)	Ποσοστό %
1	Σωστή απάντηση	19	45,24
2	Σωστή απάντηση, μη επαρκής αιτιολόγηση	11	26,19
3	Λάθος απάντηση	9	21,43
4	Χωρίς απάντηση	3	7,14

Πίνακας 17 Κατηγορίες απαντήσεων στο Έργο 6α



Γράφημα 11 Ποσοστά απαντήσεων στο Έργο 6α



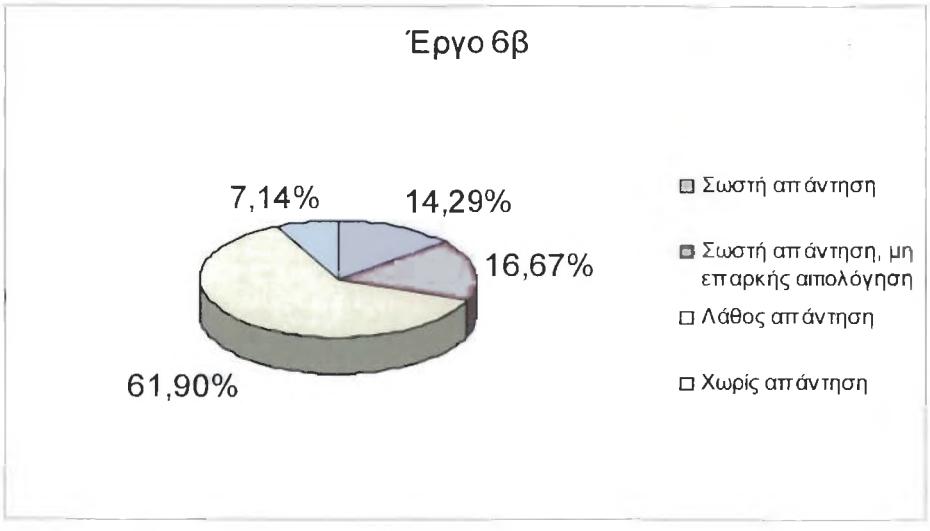
Γράφημα 12 Ραβδόγραμμα με τα ποσοστά των απαντήσεων στο Έργο 6α

Έργο 6β

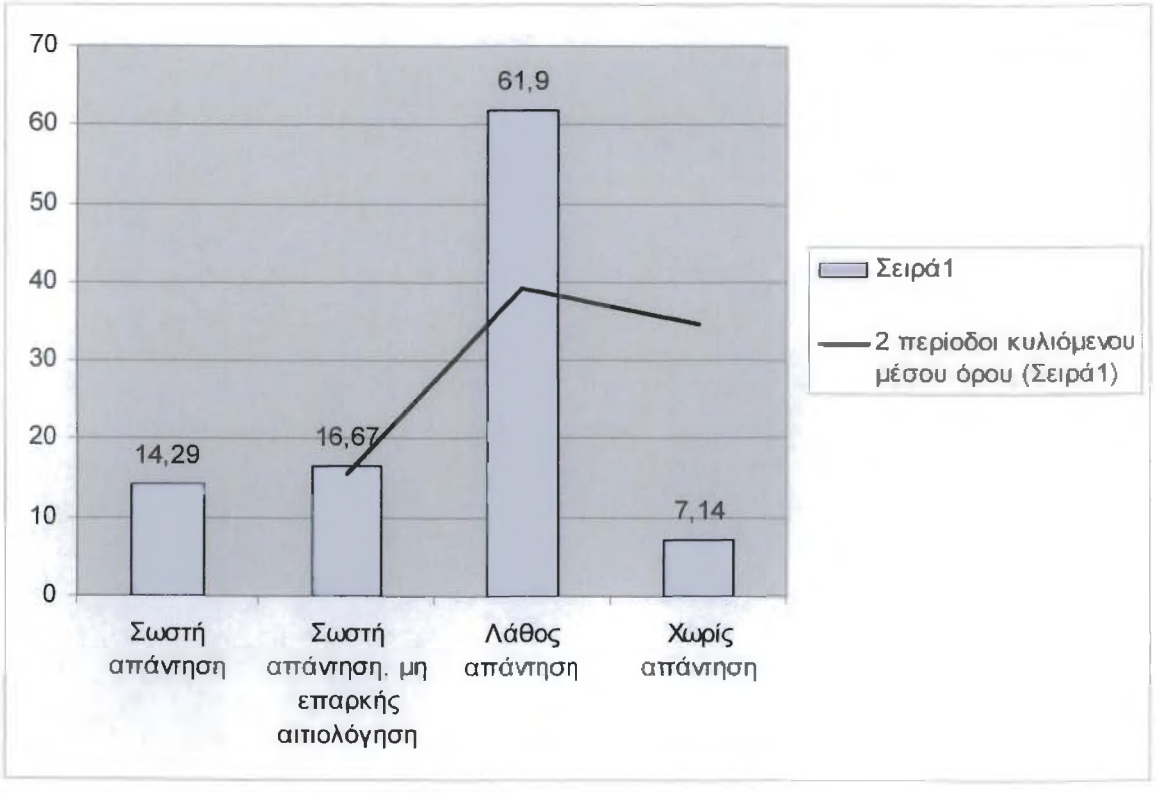
Στο δεύτερο σκέλος του 6ου έργου απάντησαν και αιτιολόγησαν σωστά τις απαντήσεις τους 6 μόνο σπουδαστές (14,29%). Επτά από τους 42 (16,67%) απάντησαν σωστά χωρίς όμως να δώσουν επαρκή αιτιολόγηση. Οι απαντήσεις τους όπως και στο πρώτο σκέλος του 6ου έργου ήταν της μορφής: *«Λάθος», «Είναι λάθος»*. Από τους 7 σπουδαστές κανένας του δεν έδωσε έστω και την παραμικρή εξήγηση γιατί το κομμάτι του κώδικα είχε λάθος. Η πλειοψηφία των σπουδαστών (61,9%) έδωσε λανθασμένη απάντηση. Όλοι οι σπουδαστές που έδωσαν λάθος απάντηση είχαν απαντήσεις της μορφής: *«Σωστό», «Είναι σωστό»*. Κανένας τους δεν αιτιολόγησε γιατί θεώρησε τον κώδικα σωστό. Παρατηρώντας την γραμμή τάσης στο Γράφημα 14 βλέπουμε ότι κορυφώνεται στην λανθασμένη απάντηση. Άρα οι σπουδαστές αντιμετωπίζουν πρόβλημα στην κατανόηση της συνάρτησης a+b=5 συγχέοντάς την με την μαθηματική εξίσωση. Τέλος τρεις από τους σπουδαστές (7,14%) δεν απάντησαν στο συγκεκριμένο ερώτημα.

α/α	Διαδικασία	Συχνότητα (N=42)	Ποσοστό %
1	Σωστή απάντηση	6	14,29
2	Σωστή απάντηση, μη επαρκής αιτιολόγηση	7	16,67
3	Λάθος απάντηση	26	61,9
4	Χωρίς απάντηση	3	7,14

Πίνακας 18 Κατηγορίες απαντήσεων στο Έργο 6β



Γράφημα 13 Ποσοστά απαντήσεων στο Έργο 6β



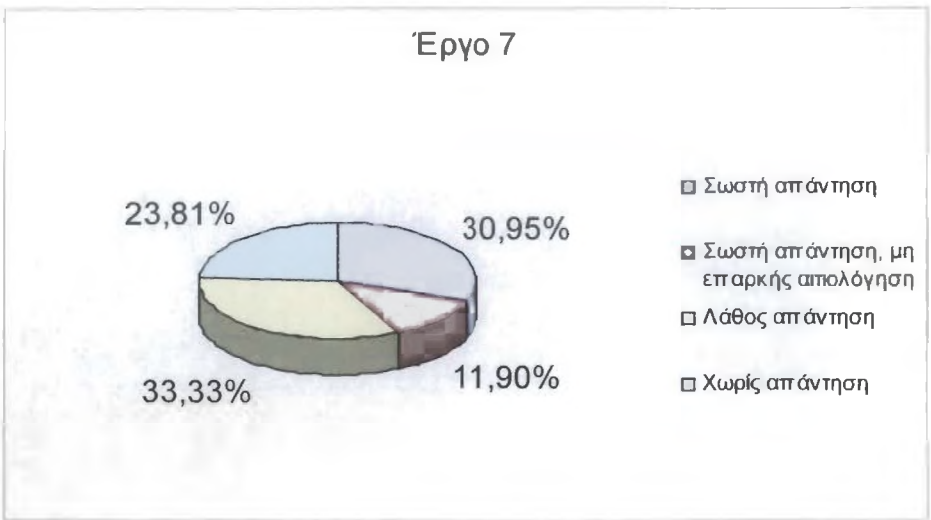
Γράφημα 14 Ραβδόγραμμα με τα ποσοστά των απαντήσεων στο Έργο 6β

Έργο 7

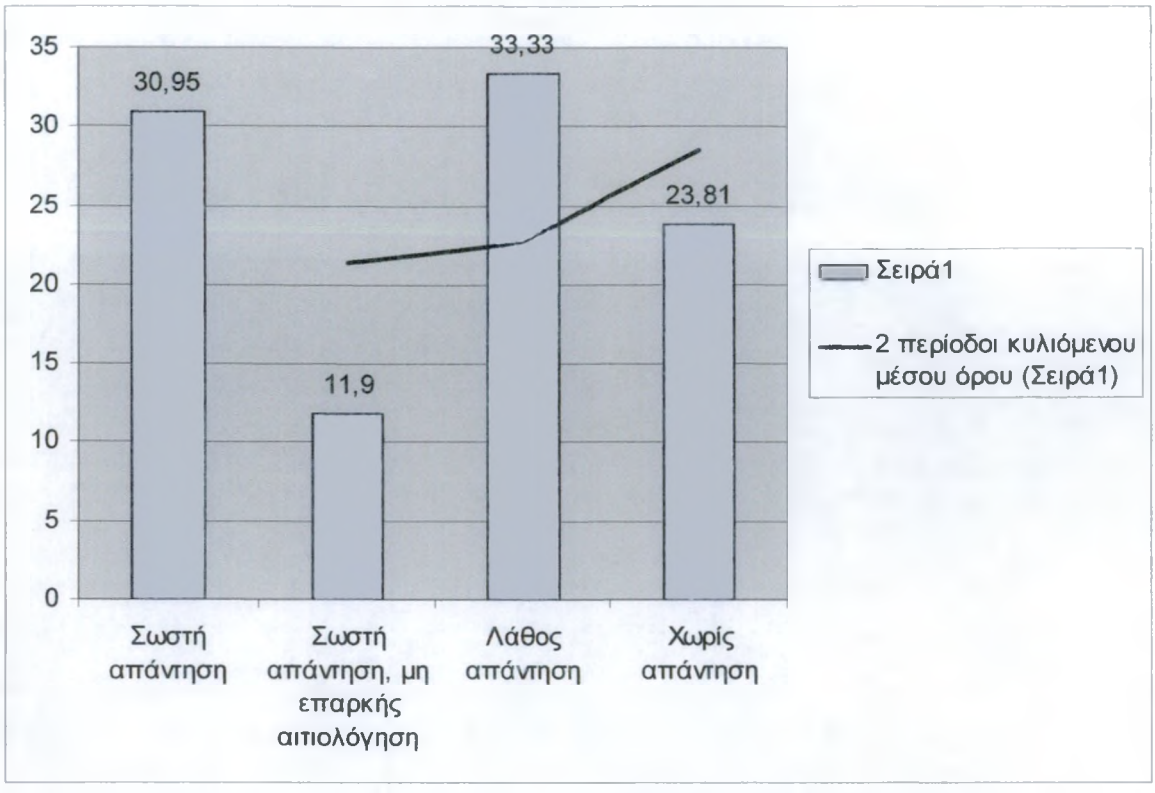
Σε αυτό το έργο ζητήθηκε από τους σπουδαστές να διερευνήσουν αν το κομμάτι κώδικα που τους δίνεται είναι σωστό ή όχι και γιατί. Οι 13 (30,95%) έδωσαν σωστή απάντηση αλλά και επαρκή αιτιολόγηση. Οι 5 (11,9%) έδωσαν σωστή απάντηση αλλά η αιτιολόγησή τους ήταν ανεπαρκής. Και οι πέντε εντόπισαν το λάθος και το επισήμαναν αλλά δεν εξήγησαν γιατί ήταν λάθος ο κώδικας. Οι 14 (33,33%) απάντησαν λάθος. Οι απαντήσει τους ήταν της μορφής: *«Είναι λάθος γιατί το if και το else if είναι το ίδιο πράγμα»* ή *«Δεν είναι σωστό γιατί στην if εφόσον δεν θα ισχύει η συνθήκη δεν χρειάζεται να ξαναβάλουμε if»*. Από τις απαντήσεις που έδωσαν είναι ξεκάθαρο ότι οι σπουδαστές δεν αντιλήφθηκαν ότι το λάθος βρισκόταν στους τελεστές σύγκρισης αλλά νόμιζαν ότι υπήρχε κάποιο πρόβλημα με την δομή ελέγχου. Δέκα (23,81%) προτίμησαν να μην απαντήσουν στο ερώτημα Παρατηρώντας και το Γράφημα 16 βλέπουμε την γραμμή τάσης να κλίνει προς τα λανθασμένα αποτελέσματα.

α/α	Διαδικασία	Συχνότητα (N=42)	Ποσοστό %
1	Σωστή απάντηση	13	30,95
2	Σωστή απάντηση, μη επαρκής αιτιολόγηση	5	11,9
3	Λάθος απάντηση	14	33,33
4	Χωρίς απάντηση	10	23,81

Πίνακας 19 Κατηγορίες απαντήσεων στο 7ο έργο



Γράφημα 15 Ποσοστά απαντήσεων στο Έργο 7



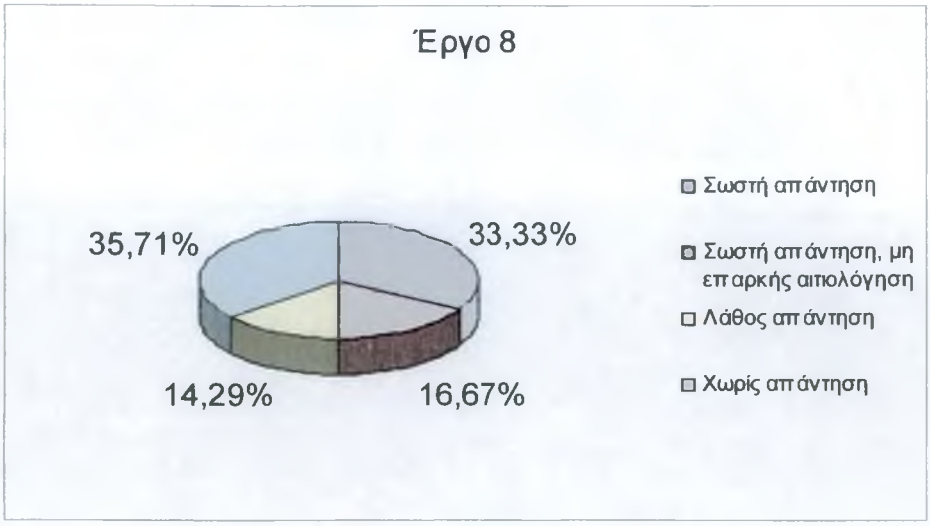
Γράφημα 16 Ραβδόγραμμα με τα ποσοστά των απαντήσεων στο 7^{ο} έργο

Έργο 8

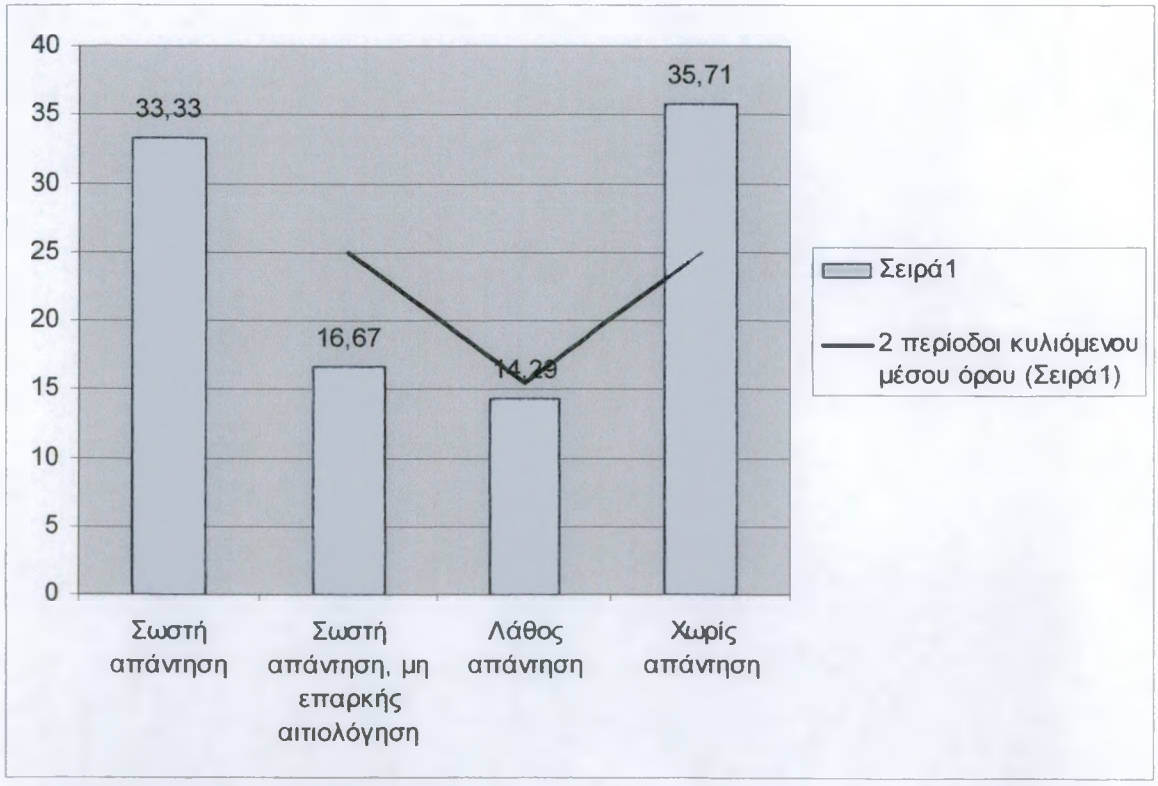
Αυτό το έργο στοχεύει να ανακαλύψει αν οι σπουδαστές έχουν καλή αντίληψη της δομής ελέγχου. Τους ζητείται να συνεχίσουν ένα κομμάτι κώδικα. Οι 14 από αυτούς (33,33%) συμπλήρωσαν σωστά το κομμάτι του κώδικα που έλειπε. Οι 7 (16,67%) που έδωσαν σωστή απάντηση χωρίς επαρκή αιτιολόγηση είχαν σωστή την γενική δομή της δομής ελέγχου αλλά είχαν μικρά συντακτικά λάθη στην δήλωση της else if. Οι 6 (14,29%) είχαν δώσει λανθασμένη απάντηση. Η σύνταξη της δομής ελέγχου ήταν τελείως λανθασμένη και σε πολλές περιπτώσεις και ημιτελής. Τέλος 15 σπουδαστές (35,71%) δεν απάντησαν. Όπως βλέπουμε και στο Γράφημα 18 η μεγαλύτερη συγκέντρωση τείνει προς τις λανθασμένες απαντήσεις με τους σπουδαστές να έχουν πρόβλημα στη δομή ελέγχου.

α/α	Διαδικασία	Συχνότητα (N=42)	Ποσοστό %
1	Σωστή απάντηση	14	33,33
2	Σωστή απάντηση, μη επαρκής αιτιολόγηση	7	16,67
3	Λάθος απάντηση	6	14,29
4	Χωρίς απάντηση	15	35,71

Πίνακας 20 Κατηγορίες απαντήσεων στο 8^{ο} έργο



Γράφημα 17 Ποσοστά απαντήσεων στο Έργο 8



Γράφημα 18 Ραβδόγραμμα με τα ποσοστά των απαντήσεων στο 8^{ο} έργο

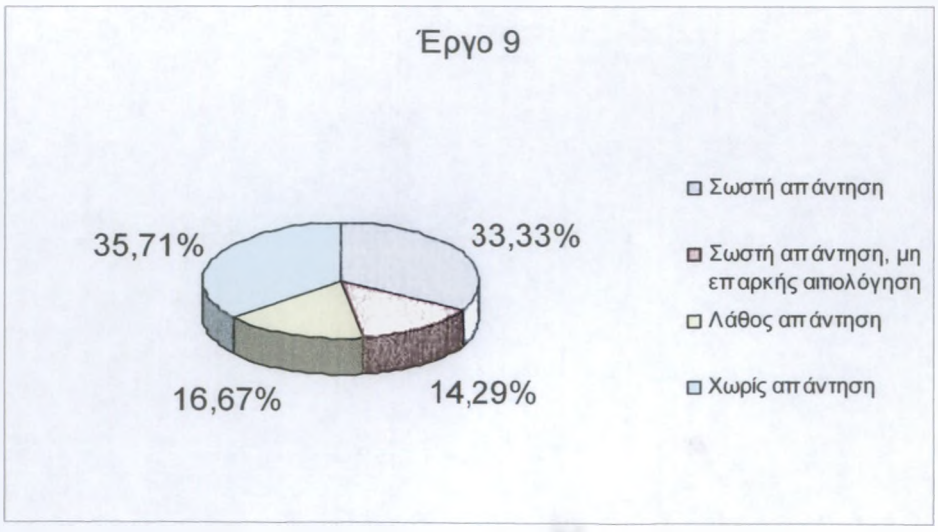
Έργο 9

Με αυτό το έργο ερευνάται η αντίληψη των σπουδαστών για τη δομή επανάληψης. Στο κομμάτι κώδικα που ακολουθεί ζητείται από τους σπουδαστές να βρουν τι θα γίνει αν δοθεί από τον χρήστη μηδενική ή αρνητική τιμή και να το εξηγήσουν εν συντομία. Οι 14

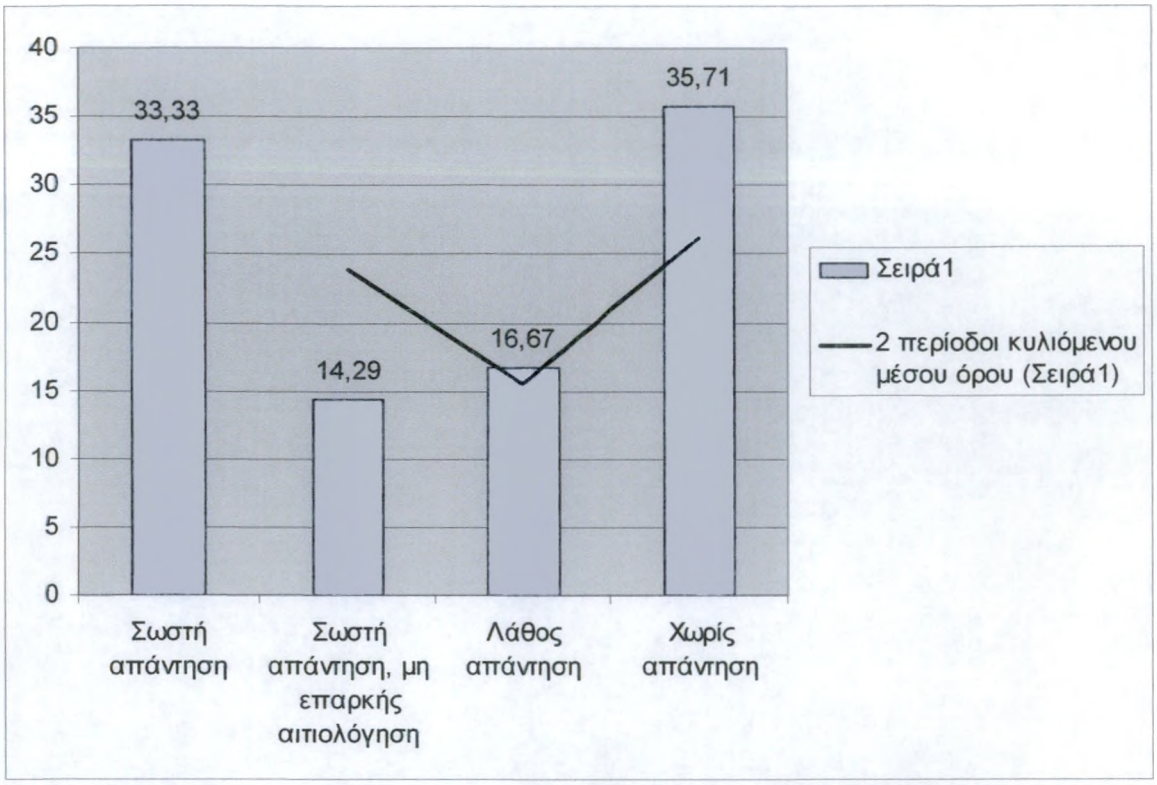
(33,33%) απαντούν και αιτιολογούν σωστά. Έξι από τους σπουδαστές (14,29%) απαντούν μεν σωστά αλλά οι απαντήσεις τους είναι ελλιπείς. Οι απαντήσεις τους είναι του τύπου: *«Αν δοθεί μηδενική ή αρνητική τιμή η εντολή for δεν θα εκτελεστεί καμία φορά»* ή *«Θα τερματίσει ο βρόχος»*. Από τις απαντήσεις τους βγάζουμε το συμπέρασμα ότι αν και είχαν καταλάβει τι έκανε το κομμάτι κώδικά που τους είχε δοθεί, παρέλειψαν να δώσουν μια επαρκή αιτιολόγηση γιατί δεν εκτελείται ο κώδικας. Λάθος απάντηση έδωσε το 16,67% των σπουδαστών. Οι απαντήσεις τους ήταν της μορφής: *«Εάν δοθεί μηδενική ή αρνητική τιμή τότε η for θα συνεχίσει κανονικά τις επαναλήψεις»*, *«Όποια τιμή και να δοθεί θα σταματήσει η εκτέλεση του βρόχου γιατί ο counter ξεκινάει από το 1»* ή *«Δεν θα τερματίσει ο βρόχος γιατί ο τερματισμός θα γίνει για τιμές μεγαλύτερες του 1»*. Από τις απαντήσεις του συμπεραίνουμε ότι δεν κατάλαβαν τι έκανε το πρόγραμμα έχοντας προφανή αδυναμία να αντιληφθούν σωστά την δομή επανάληψης. Τέλος 15 σπουδαστές (35,71%) δεν απάντησαν έχοντας φανερή δυσκολία στην αντίληψη της δομής επανάληψης. Παρατηρώντας το Γράφημα 20 βλέπουμε την τάση των απαντήσεων σε όλο το φάσμα των απαντήσεων.

α/α	Διαδικασία	Συχνότητα (N=42)	Ποσοστό %
1	Σωστή απάντηση	14	33,33
2	Σωστή απάντηση, μη επαρκής αιτιολόγηση	6	14,29
3	Λάθος απάντηση	7	16,67
4	Χωρίς απάντηση	15	35,71

Πίνακας 21 Κατηγορίες απαντήσεων στο 9^{ο} έργο



Γράφημα 19 Ποσοστά απαντήσεων στο Έργο 9



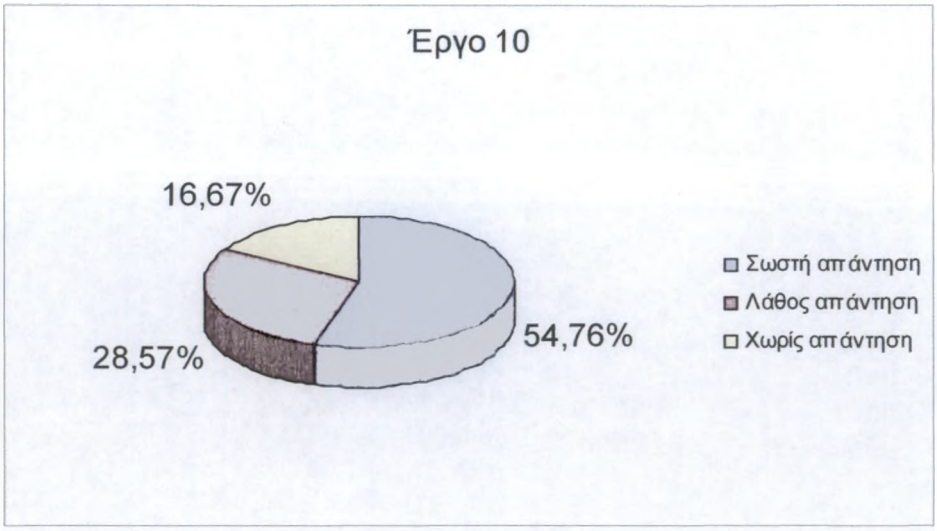
Γράφημα 20 Ραβδόγραμμα με τα ποσοστά των απαντήσεων στο 9ο έργο

Έργο 10

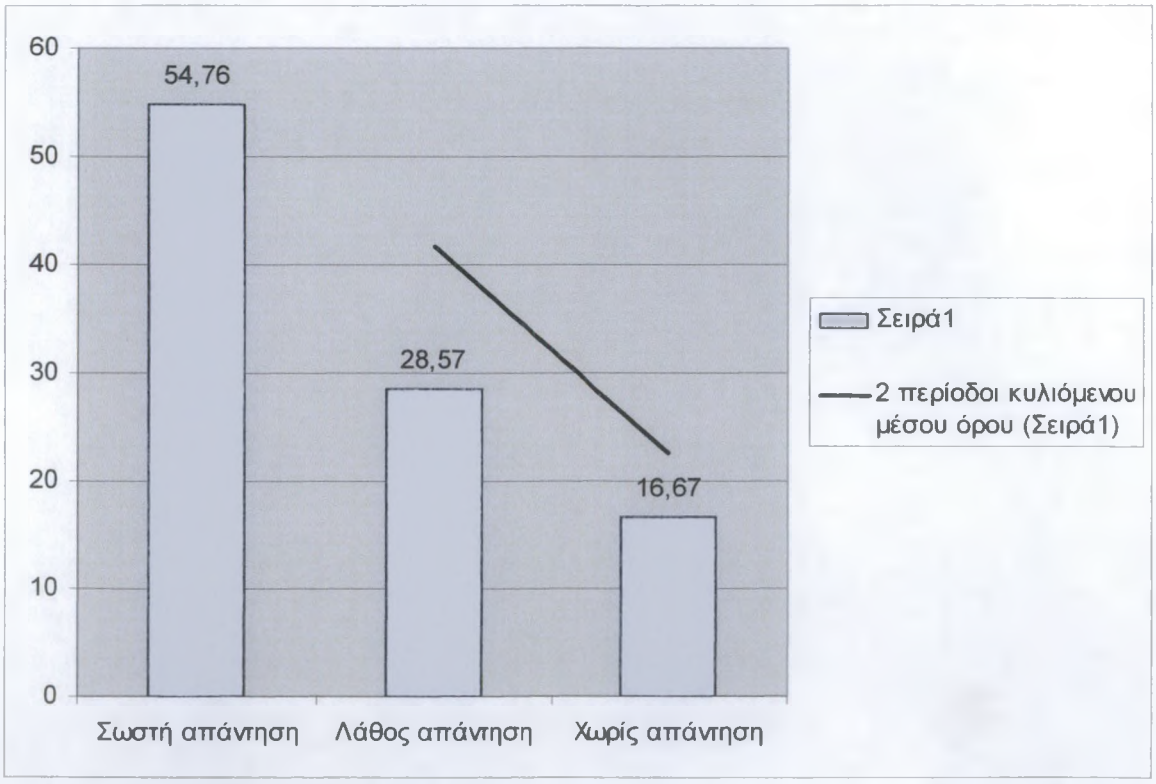
Σε αυτό το έργο ζητείται να βρεθεί πότε τερματίζει η επαναληπτική διαδικασία και γιατί. Σωστή απάντηση και αιτιολόγηση έδωσαν 23 σπουδαστές (54,76%), δηλαδή η πλειοψηφία των ερωτηθέντων σπουδαστών. Αυτό μας δίνει μια ένδειξη ότι στο συγκεκριμένο έργο οι σπουδαστές είχαν καλή αντίληψη της δομής επανάληψης. Βέβαια το έργο αυτό ήταν πιο «βατό» σε σχέση με το αντίστοιχο Έργο 9. Λάθος απάντηση έδωσαν 12 σπουδαστές (28,57%). Οι απαντήσεις τους ήταν της μορφής: *«Όταν το x=0»* ή *«Δεν τερματίζει ποτέ»*. Από τις απαντήσεις αυτών των σπουδαστών συμπεραίνουμε ότι έχουν δυσκολία να καταλάβουν την δομή επανάληψης. Τέλος 7 σπουδαστές (16,67%) δεν απαντούν. Βλέποντας το Γράφημα 22 η οι απαντήσεις τείνουν προς την σωστή.

α/α	Διαδικασία	Συχνότητα (N=42)	Ποσοστό %
1	Σωστή απάντηση	23	54,76
2	Λάθος απάντηση	12	28,57
3	Χωρίς απάντηση	7	16,67

Πίνακας 22 Κατηγορίες απαντήσεων στο 10ο έργο



Γράφημα 21 Ποσοστά απαντήσεων στο Έργο 10



Γράφημα 22 Ραβδόγραμμα με τα ποσοστά των απαντήσεων στο $10^{ο}$ έργο

10.5. Συμπεράσματα-Διδακτικές Προτάσεις

Από τα αποτελέσματα της έρευνας προκύπτει ότι, η πλειονότητα των σπουδαστών του δείγματος έχει σημαντικές δυσκολίες στην εφαρμογή της **μεταβλητής**, της **εντολής ανάθεσης τιμής σε μεταβλητή**, των **δομών ελέγχου** και της **έννοιας της επανάληψης**. Καταγράφεται σε μεγάλο βαθμό αδυναμία χειρισμού μεταβλητών long και float γεγονός που οφείλεται στην περιορισμένη χρήση τους στο μάθημα Προγραμματισμός Ι και ΙΙ. Επίσης καταγράφεται αδυναμία στη διαδικασία ανάθεσης τιμής σε μεταβλητή. Η πλειονότητα των σπουδαστών δεν μπορεί να αντιληφθεί την εντολή ανάθεσης τιμής σε μεταβλητή με την προγραμματιστική της έννοια αλλά ως ισότητα, όπως είναι στα μαθηματικά. Επίσης η πλειοψηφία των σπουδαστών του δείγματος ακολουθούν την μαθηματική αναπαράσταση για την έννοια της μεταβλητής. Αντιμετωπίζουν τις διαδοχικές εκχωρήσεις ως μαθηματικές σχέσεις. Δεν έχουν κατανοήσει τη διαδοχικότητα των εντολών εκχώρησης σε ένα πρόγραμμα και την δυναμική τροποποίηση των τιμών των εμπλεκόμενων μεταβλητών. Οι δυσκολίες των σπουδαστών στην κατανόηση της συνολικής μορφής των εντολών εκχώρησης ή η πρόσκτησή τους με λανθασμένη σειρά, οφείλεται στην έλλειψη της αναπαράστασης της διαδοχικής φύσης της εκτέλεσης τους. Οι σπουδαστές πιστεύουν ότι πρόκειται για ταυτόχρονες εντολές σχετικά με τις ιδιότητες των μεταβλητών παρά για μια διαδικασία που αφορά σε μια εσωτερική υπολογιστική κατάσταση.

Επίσης ένα μεγάλο ποσοστό των σπουδαστών του δείγματος δεν μπορεί να αντιληφθεί ότι στο Έργο 7 οι τελεστές σύγκρισης είναι λανθασμένοι και εντοπίζουν το λάθος στον κώδικα. Αυτή η αδυναμία τους οφείλεται στο γεγονός ότι δεν έχουν δώσει την ανάλογη προσοχή κατά τη διδασκαλία του προγραμματισμού, ιδιαίτερα στο πρώτο εξάμηνο των σπουδών τους, αλλά και στο γεγονός ότι συγχέουν τους τελεστές σύγκρισης στον προγραμματισμό με τη σύγκριση στα μαθηματικά.

Στην κατανόηση των δομών ελέγχου επίσης παρατηρείται πρόβλημα. Ένα μεγάλο ποσοστό του δείγματος των σπουδαστών που ερωτήθηκαν παρουσιάζει δυσκολία κατανόησης ως προς την δομή των δομών ελέγχου. Δεν μπορούν να αντιληφθούν ότι η πρόταση if της C++ επιτρέπει στο πρόγραμμα να ελέγχει μία συνθήκη και να εκτελεί μία ή περισσότερες προτάσεις αν η συνθήκη αυτή αληθεύει. Επίσης, δεν μπορούν να αντιληφθούν ότι η πρόταση else της C++ τους επιτρέπει να καθορίσουν στο πρόγραμμα

μία ή περισσότερες προτάσεις που θα εκτελούνται όταν κάποια συνθήκη που ελέγχθηκε από μια πρόταση if είναι ψευδής.

Στην δομή της επανάληψης σχεδόν οι μισοί σπουδαστές αντιμετώπισαν πρόβλημα και στα δύο έργα που δίνονταν. Στο Έργο 9 δεν μπόρεσαν να αντιληφθούν τι θα γινόταν αν έδιναν μηδενική ή αρνητική τιμή στο κομμάτι του κώδικα που τους είχε δοθεί. Πολλοί απάντησαν ότι δε θα γίνει τίποτα και ο βρόχος θα συνεχίσει να εκτελείται, πράγμα που είναι λάθος. Δεν μπόρεσαν να καταλάβουν την δομή της επανάληψης στο συγκεκριμένο έργο, δηλαδή ότι η εντολή for αποτελείται από τέσσερα μέρη: μία απόδοση αρχικής τιμής, μία συνθήκη ελέγχου, τις προτάσεις που θα επαναλαμβάνεται η εκτέλεσή τους και ένα βήμα αύξησης. Στο Έργο 10 η πλειονότητα των σπουδαστών απαντά σωστά, αλλά ένα σημαντικό ποσοστό από αυτούς, περίπου το 45% δίνει λανθασμένη ή και καθόλου απάντηση. Όσοι απαντούν λανθασμένα αιτιολογούν την απάντησή τους λέγοντας η επαναληπτική διαδικασία δε θα τελειώσει ποτέ ή ότι θα τελειώσει όταν ο χρήστης δώσει τιμή ίση με το μηδέν. Και οι δύο λανθασμένες απαντήσεις αντικατοπτρίζουν τις λανθασμένες αντιλήψεις των σπουδαστών πάνω στην δομή επανάληψης. Το πρόβλημα εντοπίζεται στην έλλειψη γνώσης που έπρεπε να έχουν οι σπουδαστές ήδη από το μάθημα του Προγραμματισμού Ι. Αυτή η έλλειψη γνώσης μπορεί να οφείλεται σε παράγοντες όπως μη σωστή διδασκαλία του προγραμματισμού στο πρώτο εξάμηνο σπουδών αλλά στην μη επαρκή προσοχή που έπρεπε να δείξουν οι σπουδαστές κατά τη διάρκεια της διδασκαλίας του προγραμματισμού.

Για την αντιμετώπιση όλων των προαναφερθέντων προβλημάτων απαιτούνται κατάλληλες διδακτικές καταστάσεις οι οποίες θα επιτρέπουν στους σπουδαστές να επιλύουν προβλήματα τα οποία τους υποχρεώνουν να συγκρούονται με γνωστές ή υποτιθέμενες δυσκολίες. Η διάκριση ανάμεσα στην **έννοια της μεταβλητής** στα μαθηματικά και στην πληροφορική συνιστά ένα ιδιαίτερο πρόβλημα, το οποίο αναδεικνύεται στους αρχάριους προγραμματιστές. Σε συνάρτηση με τα υπάρχοντα αποτελέσματα, που αφορούν στην οικοδόμηση της **εντολής εκχώρησης** και των λειτουργικών εφαρμογών της, οι διδακτικές αυτές καταστάσεις δεν πρέπει να περιορίζονται απλά στην παράθεση παραδειγμάτων αλλά να οργανώνονται σε ένα μακροπρόθεσμο διδακτικό σχεδιασμό. Θα πρέπει επίσης να συμπεριλαμβάνουν κατάλληλα σχεδιασμένα έργα, τα οποία θα στοχεύουν άμεσα στη δημιουργία λειτουργικών αναπαραστάσεων για την **έννοια της μεταβλητής** και τις πολλαπλές

μορφές της **εντολής εκχώρησης**. Οι παραπάνω παραδοχές ισχύουν βεβαίως και για την **δομή ελέγχου** όπως και για την **δομή επανάληψης**.

Παρατηρήθηκε επίσης ότι τα αποτελέσματα αυτής της έρευνας συμπίπτουν με τα αποτελέσματα άλλων παρόμοιων ερευνών που διεξήχθησαν από ειδήμονες του αντικειμένου της Διδακτικής της Πληροφορικής, όπως είναι οι καθηγητές Κόμης Βασίλειος του Τμήματος Επιστημών της Εκπαίδευσης και της Αγωγής του Πανεπιστημίου Πατρών και Τζιμογιάννης Αθανάσιος του Τμήματος Κοινωνικής και Εκπαιδευτικής Πολιτικής του Πανεπιστημίου Πελοποννήσου. Αυτές οι έρευνες έχουν δημοσιευθεί στα πρακτικά συνεδρίων, ημερίδων και σεμιναρίων σχετικών με τη Διδακτική της Πληροφορικής. Αυτές οι έρευνες διεξήχθησαν με την ίδια μεθοδολογία σε αρχάριους προγραμματιστές και τα αποτελέσματά τους κατά το μεγαλύτερο μέρος τους ήταν ίδια. Το γεγονός αυτό δείχνει ότι οι δυσκολίες που αντιμετωπίζουν οι αρχάριοι προγραμματιστές σε όλες τις εκπαιδευτικές βαθμίδες είναι ίδιες.

Συμπερασματικά, απαιτείται μια αλλαγή προοπτικής σχετικά με τις μελέτες που αφορούν στην οικοδόμηση των προγραμματιστικών δομών, πρέπει να ασχολούνται με το ζήτημα των κατάλληλων διδακτικών καταστάσεων οι οποίες θα μπορούσαν να ευνοήσουν την οικοδόμηση των εννοιών που εμπλέκονται στις δομές αυτές.

11. Γλωσσάρι

Αποσφαλμάτωση (debugging)

Η διαδικασία ανίχνευσης και διόρθωσης των λαθών (κυρίως τα λογικά λάθη) στο πλαίσιο του προγραμματισμού.

Γνωστική σύγκρουση

Η διαδικασία κατά την οποία στη σκέψη ενός ατόμου εμφανίζεται μια αντίφαση ή μια ασυμβατότητα ανάμεσα στις ιδέες του, τις αναπαραστάσεις του και τις πράξεις του.

Διδακτική

Το επιστημονικό πεδίο που μελετά τις διαδικασίες μετάδοσης και οικοδόμησης των γνώσεων με απώτερο στόχο τη βελτίωση αυτών των διαδικασιών.

Διδακτική της Πληροφορικής

Ο κλάδος της Διδακτικής που μελετά την παιδαγωγική χρήση λογισμικών διαφόρων τύπων (όπως προγραμματιστικά εργαλεία και λογισμικό ευρείας χρήσης) καθώς και τις διαδικασίες οικοδόμησης των βασικών εννοιών της πληροφορικής.

Διδακτικό συμβόλαιο

Οι υπονοούμενοι όροι που καθορίζουν τις προσδοκώμενες δραστηριότητες του εκπαιδευτικού αλλά και των σπουδαστών, τις αντίστοιχες θέσεις του καθένα απέναντι στην προς επεξεργασία γνώση καθώς επίσης και τις γενικές συνθήκες μέσα στις οποίες οι σχέσεις με τη γνώση εξελίσσονται κατά τη διάρκεια της διδασκαλίας.

Διδακτικό τρίγωνο

Το τρίγωνο που συμβολίζει το σύστημα που συνδέει τις γνώσεις, το σπουδαστή και τον εκπαιδευτικό.

Επίλυση προβλήματος

Η διαδικασία που συνίσταται στην ανάπτυξη στρατηγικών και στην εγκαθίδρυση διαδικασιών που επιτρέπουν την σύνδεση ανάμεσα στην αρχική κατάσταση (κατάσταση - στόχος) μιας κατάστασης προβλήματος.

Κοινωνικογνωστική σύγκρουση

Η γνωστική σύγκρουση που είναι προϊόν διαπροσωπικής αλληλεπίδρασης και συνεπώς αναπτύσσεται μέσω διαδικασιών κοινωνικής προέλευσης.

Πληροφορική και εκπαίδευση

Οι διαδικασίες, τα μέσα και οι προσεγγίσεις της εισαγωγής και της ένταξης των Τεχνολογιών της Πληροφορίας και των Επικοινωνιών στο σχολικό σύστημα.

Πληροφορική ως γνωστικό αντικείμενο

Η προσέγγιση που υποστηρίζει την ένταξη της πληροφορικής ως ανεξάρτητο γνωστικό αντικείμενο στο πρόγραμμα σπουδών.

Στάδια μάθησης του προγραμματισμού

Η μάθηση του προγραμματισμού εξελίσσεται μέσα από μια σειρά διαδοχικών σταδίων από τα οποία περνά υποχρεωτικά ο σπουδαστής που μαθαίνει προγραμματισμό.

Σύστημα αναπαράστασης και επεξεργασίας

Το προϊόν της εσωτερίκευσης ενός χώρου έργων, που συνδέει τις αναπαραστάσεις και τις επεξεργασίες που σχετίζονται στα αντικείμενα, στις ιδιότητες και στις πράξεις του χώρου.

12. Ακρωνύμια

PC : Personal Computer (Προσωπικός Υπολογιστής)

Ε.Π.Δ.Ο. : Εφαρμογές της Πληροφορικής στην Διοίκηση και στην Οικονομία

Η/Υ : Ηλεκτρονικός Υπολογιστής

Τ.Ε.Ε. : Τεχνολογικό Επαγγελματικό Εκπαιδευτήριο

ΤΠΕ : Τεχνολογίες της Πληροφορικής και των Επικοινωνιών

Βιβλιογραφία

1. **Bonar, J., & Soloway E.,** 1985, Programming Knowledge: a major source of misconceptions in novice programmers, εκδόσεις Human – Computer Interaction, USA
2. **Delannoy, C.,** 1996, Initiation à la programmation, Eyrolles, Paris
3. **Dijkstra, E.W.,** 1976, A Discipline of Programming, Englewood Clis, εκδόσεις NJ: Prentice – Hall
4. **Jamsa Kris,** 1994, Εισαγωγή στη C++, εκδόσεις Κλειδάριθμος, Αθήνα
5. **Piaget, J.,** 1969, Psychologie et Pédagogie, εκδόσεις Denöel, Paris
6. **Samurçay, R.,** 1987, Modèles cognitifs dans l' acquisition des concepts informatiques, Actes du premier colloque franco – allemande de didactique, εκδόσεις La Pensée Sauvage, Grenoble
7. **Κόμης Βασίλης Ι.,** 2005, Εισαγωγή στη Διδακτική της Πληροφορικής, εκδόσεις Κλειδάριθμος, Αθήνα
8. **Πανέτσος Σ.,** 2001, Οι Υπολογιστές στην Εκπαίδευση, εκδόσεις ΙΩΝ, Αθήνα
9. **Εφόπουλος Β., Ευαγγελίδης Γ., Δαγδιλέλης Β. & Κλεφτοδήμος Α.,** 2005, Οι δυσκολίες των αρχάριων προγραμματιστών, Πρακτικά 3ου Πανελληνίου Συνεδρίου: Διδακτική της Πληροφορικής, εκδόσεις Ελληνικά Γράμματα, Αθήνα
10. **Κόμης Βασίλης Ι.,** 1997, Αντιλήψεις και αναπαραστάσεις των μαθητών της Γενικής Παιδείας για τον υπολογιστή και την πληροφορική, Πρακτικά διημερίδας πληροφορικής «Η πληροφορική στη Δευτεροβάθμια Εκπαίδευση», Αθήνα
11. **Τζιμογιάννης Α. & Κόμης Β.,** 2000, Η έννοια της μεταβλητής στον Προγραμματισμό: δυσκολίες και παρανοήσεις μαθητών του Ενιαίου Λυκείου, Πρακτικά 2ου Πανελληνίου Συνεδρίου: Οι τεχνολογίες της Πληροφορίας και της Επικοινωνίας στην Εκπαίδευση, Πάτρα
12. **Τζιμογιάννης Α. & Κόμης Β.,** 2004, Μελέτη των αναπαραστάσεων μαθητών του Ενιαίου Λυκείου για τη ροή δεδομένων και το ρόλο των βασικών μονάδων του υπολογιστή, Πρακτικά 2ης Διημερίδας με διεθνή συμμετοχή «Διδακτική της Πληροφορικής», Βόλος
13. **Τζιμογιάννης Α. & Κόμης Β.,** 2005, Μελέτη των αναπαραστάσεων μαθητών του Ενιαίου Λυκείου για την έννοια της μεταβλητής, Πρακτικά 3ου Πανελληνίου Συνεδρίου: Διδακτική της Πληροφορικής, εκδόσεις Ελληνικά Γράμματα, Αθήνα

14. **Φεσάκης Γ. & Δημητρακοπούλου Α.,** 2005, Γνωστικές δυσκολίες μαθητών της Δευτεροβάθμιας Εκπαίδευσης σχετικά με την έννοια της προγραμματιστικής μεταβλητής και προτεινόμενες παρεμβάσεις, Πρακτικά 3ου Πανελληνίου Συνεδρίου: Διδακτική της Πληροφορικής, εκδόσεις Ελληνικά Γράμματα, Αθήνα