Διδασκαλία δομών επανάληψης με τη χρήση του μεταγλωττιστή *Διερμηνευτής της ΓΛΩΣΣΑΣ* στα πλαίσια του μαθήματος «Ανάπτυξη Εφαρμογών σε Προγραμματιστικό Περιβάλλον»

Άλκης Γεωργόπουλος
Εκπαιδευτικός ΠΕ19
alkisg@sch.gr

Το μάθημα «Ανάπτυξη Εφαρμογών σε Προγραμματιστικό Περιβάλλον»

Το μάθημα «Ανάπτυξη Εφαρμογών σε Προγραμματιστικό Περιβάλλον» αποτελεί το βασικό μάθημα της Γ΄ τάξης του κύκλου Πληροφορικής και Υπηρεσιών της Τεχνολογικής Κατεύθυνσης του Ενιαίου Λυκείου. Ο γενικός σκοπός του μαθήματος είναι να αναπτύξουν οι μαθητές αναλυτική και συνθετική σκέψη, να αποκτήσουν ικανότητες μεθοδολογικού χαρακτήρα και να μπορούν να επιλύουν απλά προβλήματα σε προγραμματιστικό περιβάλλον (ΥΠΕΠΘ-ΠΙ, 1997).

Συνεπώς, το συγκεκριμένο μάθημα έχει ως πρωταρχικό στόχο την ανάπτυξη δεξιοτήτων και ικανοτήτων σχετικών με την αλγοριθμική και την ορθολογική χρήση των δεξιοτήτων αυτών στην καθημερινή ζωή. Πολλές βασικές έννοιες αλγοριθμικής (αλλά και προγραμματισμού), όπως συνθήκες ελέγχου, λογικές προτάσεις και συμπεράσματα, κ.α., συνιστούν αναπόσπαστο τμήμα των γενικών γνώσεων και δεξιοτήτων που πρέπει να αποκτήσει ο μαθητής στο πλαίσιο της γενικής του παιδείας, οι οποίες - στην πλειονότητά τους - δεν προσεγγίζονται από άλλα γνωστικά αντικείμενα (Πολίτης & Κόμης, 1999).

Βασικό ζητούμενο ενός σύγχρονου προγράμματος σπουδών είναι η καλλιέργεια δεξιοτήτων σχετικών με την κατανόηση, ανάλυση και επίλυση προβλημάτων. Το εν λόγω μάθημα έρχεται να καλύψει το κενό που υπάρχει στο πρόγραμμα σπουδών και αφορά στις δραστηριότητες επίλυσης προβλημάτων. Η διδασκαλία του περιλαμβάνει την ανάπτυξη τέτοιου είδους δεξιοτήτων με επικέντρωση στα ποικίλα - ως προς τη φύση τους - προβλήματα, στο ίδιο το πρόβλημα, την σχεδίαση της επίλυσής του και σε μικρότερο βαθμό στην υλοποίηση της επίλυσης.

Το μάθημα «Ανάπτυξη Εφαρμογών σε Προγραμματιστικό Περιβάλλον» δεν έχει ως στόχο τη διδασκαλία και την εκμάθηση κάποιου συγκεκριμένου προγραμματιστικού περιβάλλοντος. Δεν αποσκοπεί στη λεπτομερειακή εξέταση της δομής, του ρεπερτορίου και των συντακτικών κανόνων κάποιας γλώσσας προγραμματισμού. Με άλλα λόγια, δηλαδή, δεν προτίθεται να δημιουργήσει προγραμματιστές, γι' αυτό το λόγο δεν αναφέρεται στην εκμάθηση εξεζητημένων τεχνικών προγραμματισμού, αλλά εστιάζει στις προσεγγίσεις και στις τεχνικές επίλυσης προβλημάτων δίνοντας έμφαση στον τρόπο δόμησης της σκέψης (ΥΠΕΠΘ-ΠΙ, 1998).

Δομές επανάληψης και διδακτικά προβλήματα

Κατά τη σύνταξη μιας δομής επανάληψης προσδιορίζονται οι λειτουργίες που πρέπει να επαναληφθούν καθώς και η συνθήκη που θα προσδιορίζει την συνέχιση ή μη της επαναληπτικής διαδικασίας (Rogalski & Samurçay, 1990).

Δύο είναι οι δυνατοί τρόποι έκφρασης μιας επαναληπτικής διαδικασίας της οποίας ο αριθμός επαναλήψεων δεν είναι εξ αρχής γνωστός:

Επαναληπτικός βρόχος / συνθήκη ελέγχου

Συνθήκη ελέγχου / επαναληπτικός βρόχος

ΟΙ δύο αυτοί τρόποι δεν παρουσιάζουν τα ίδια διδακτικά προβλήματα από την πλευρά των μαθητών (Τζιμογιάννης & Γεωργίου, 1999). Ο τρόπος «προγραμματιστικής έκφρασης» των μαθητών επηρεάζεται από τις αναπαραστάσεις που έχουν σχετικά με την επανάληψη. Ο τύπος της επαναληπτικής δομής που ταιριάζει καλύτερα στις αρχικές ιδέες των μαθητών είναι ο **Επανάλαβε**…**Όσο**, ενώ αντίθετα ο τύπος **Όσο**… **Επανάλαβε**, εμφανίζει περισσότερα προβλήματα κατανόησης γιατί ακολουθεί αντίθετη πορεία συλλογισμού από αυτήν που έχουν συνηθίσει οι μαθητές (Κόμης, 2001).

Μελέτη περίπτωσης: Διδασκαλία της εντολής «Όσο» με το Διερμηνευτή

Ο διερμηνευτής της ΓΛΩΣΣΑΣ αποτελεί ένα ολοκληρωμένο περιβάλλον ανάπτυξης προγραμμάτων στη ΓΛΩΣΣΑ προγραμματισμού που ορίζεται στο βιβλίο του μαθήματος «Ανάπτυξη εφαρμογών σε προγραμματιστικό περιβάλλον» της Γ' Ενιαίου Λυκείου. Ο σχεδιασμός του περιβάλλοντος διεπαφής και των διαθέσιμων λειτουργιών του διερμηνευτή δεν έγινε με στόχο την απλή απόδοση στα ελληνικά ενός «επαγγελματικού» προγραμματιστικού περιβάλλοντος, αλλά με κύριο γνώμονα την υποβοήθηση του διδάσκοντα κατά την παρουσίαση των προγραμματιστικών εννοιών και την κατάλληλη καθοδήγηση των μαθητών στα πρώτα βήματα επίλυσης αλγοριθμικών προβλημάτων.

Ας εξετάσουμε τον συνήθη τρόπο παρουσίασης της εντολής «Όσο» στον πίνακα, ώστε να δούμε σε ποια σημεία μπορεί να βοηθήσει ο διερμηνευτής. Ο καθηγητής μπορεί να ξεκινήσει εξηγώντας την αναγκαιότητα για μια νέα εντολή, παρουσιάζοντας ένα πρόβλημα το οποίο δεν μπορεί να λυθεί με τις ήδη γνωστές εντολές.

Ένα απλούστατο παράδειγμα είναι η κατασκευή προγράμματος που να εμφανίζει 10 αριθμούς. Αυτό είναι πολύ εύκολο με τις γνωστές εντολές και η λύση του περιλαμβάνει απλά την επανάληψη της εντολής «Γράψε» 10 φορές (σχήμα 1).

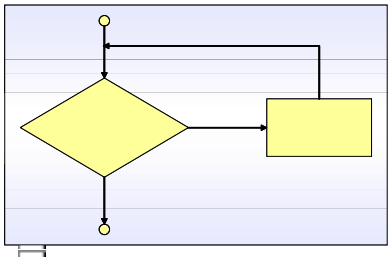
Τι γίνεται όμως αν θέλουμε να εμφανίσουμε χιλιάδες αριθμούς; Ή αν τη στιγμή που γράφουμε το πρόγραμμα δεν ξέρουμε πόσους αριθμούς θέλουμε να εμφανίσουμε αλλά αυτό είναι απόφαση του χρήστη ή εξαρτάται από τα δεδομένα του προβλήματος, όπως για παράδειγμα ο υπολογισμός των τόκων στο τέλος του έτους από μία τράπεζα; Ο αριθμός των λογαριασμών δεν είναι σταθερός. Επομένως η αντιμετώπιση του προβλήματος με τις ήδη γνωστές εντολές δεν είναι εφικτή. Η λύση είναι να χρησιμοποιηθεί η εντολή «Όσο».

```
Πρόγραμμα Αριθμοί
Αρχή
  Γράψε 1
  Γράψε 2
  Γράψε 3
  Γράψε 4
  Γράψε 5
  Γράψε 6
  Γράψε 7
  Γράψε 8
  Γράψε 9
  Γράψε 10
τέλος_προγράμματος
```

Σχήμα 1
Ακολουθιακή λύση

Η σύνταξη της εντολής είναι: όσο ισχύει κάποια συνθήκη επαναλαμβάνονται κάποιες εντολές (σχήμα 2).

Το διάγραμμα ροής (εικόνα 1) επιδεικνύεται στον πίνακα για την καλύτερη αναπαράσταση της ροής εκτέλεσης. Το πρόγραμμα καθώς εκτελείται συναντά μία συνθήκη. Αν αυτή η συνθήκη δεν ισχύει (Ψευδής), τότε το πρόγραμμα συνεχίζει με την επόμενη εντολή. Αν όμως ισχύει (Αληθής), τότε εκτελούνται οι εντολές που βρίσκονται μέσα στην «Όσο». Στη συνέχεια το



Εικόνα 1 Διάγραμμα ροής της εντολής «Όσο»

πρόγραμμα μπαίνει σε κύκλο, αποτιμώντας την συνθήκη και εκτελώντας τις εντολές. Οι εντολές αυτές επηρεάζουν τα στοιχεία της συνθήκης, οπότε αυτή κάποια στιγμή παύει να ισχύει. Τότε η ανακύκλωση σταματάει και το πρόγραμμα συνεχίζει με την επόμενη εντολή.

Ας δούμε πώς υλοποιείται το προηγούμενο πρόγραμμα χρησιμοποιώντας αυτήν την εντολή (σχήμα 2).

Όσο συνθήκη **επανάλαβε**
  εντολές
τέλος_επανάληψης

Σχήμα 2: Η σύνταξη της εντολής «Όσο»

Πλέον μόνο μία εντολή «Γράψε» αρκεί, όμως για να γράψουμε όλους αυτούς τους αριθμούς χρειαζόμαστε και έναν μετρητή, την μεταβλητή i. Ξεκινάμε από την τιμή 1 και θα πρέπει να συνεχίσουμε μέχρι το i να φτάσει την τιμή 100. Έτσι μέσα στην επανάληψη θα πρέπει εκτός από την εντολή «Γράψε» να υπάρχει και η αύξηση του μετρητή. Όταν τελικά το i ξεπεράσει το 100 η συνθήκη γίνεται ψευδής, οπότε η εκτέλεση του προγράμματος συνεχίζει με τις εντολές που βρίσκονται κάτω από το «τέλος_επανάληψης».

Ας δούμε τώρα αν μπορεί να βοηθήσει ο διερμηνευτής στην παρουσίαση της νέας εντολής.

Οι μαθητές καλούνται να ανοίξουν το παράδειγμα που έχει ετοιμάσει ο καθηγητής. Θα πρέπει να πιέζουν τακτικά το πλήκτρο βήμα προς βήμα εκτέλεσης, να παρακολουθούν τις τιμές του μετρητή i αλλά και της συνθήκης της Όσο, ώστε να βλέπουν αν θα γίνει διακλάδωση ή όχι. Τέλος τα αποτελέσματα της εντολής «Γράψε» εμφανίζονται στην οθόνη χρήστη.

Στην αρχή της εκτέλεσης η μεταβλητή i δεν έχει τιμή, και αυτό φαίνεται από την χαρακτηριστική παύλα (-). Συνθήκη στην τρέχουσα εντολή δεν υπάρχει και έτσι και σ' αυτήν υπάρχει παύλα. Φτάνουμε στην εντολή όσο, οπότε βλέπουμε ότι το i έχει γίνει 1 και αφού είναι μικρότερο από 10 η συνθήκη είναι αληθής. Συνεχίζουμε και παρατηρούμε ότι το i γράφεται στην οθόνη χρήστη και στη συνέχεια αυξάνεται. Εδώ βλέπουμε καθαρά ότι η επόμενη εντολή μετά το i ← i + 1 δεν είναι το «τέλος_επανάληψης», αλλά πάλι η εντολή όσο. Η συνθήκη ισχύει ακόμα, οπότε γίνεται και δεύτερη ανακύκλωση. Το i γράφεται, αυξάνεται και πάλι από την αρχή. Εδώ μπορεί να ενεργοποιηθεί η δυνατότητα «αργής εκτέλεσης» του διερμηνευτή ώστε να βλέπουμε την εκτέλεση χωρίς να πατάμε συνέχεια το πλήκτρο βήμα προς βήμα εκτέλεσης.

Τελικά το i θα γίνει 11, οπότε η συνθήκη θα είναι ψευδής. Επομένως το πρόγραμμα δεν θα εκτελέσει την εντολή «Γράψε i», αλλά θα συνεχίσει με τις εντολές που βρίσκονται από το «τέλος_επανάληψης» και κάτω.

Δεν μπορούμε να είμαστε σίγουροι ότι το πρόγραμμα που γράψαμε είναι σωστό αν δεν το «τρέξουμε» στον υπολογιστή ώστε να επιβεβαιώσουμε ότι εκτελείται κανονικά και αποδίδει τα σωστά αποτελέσματα. Τα προγραμματιστικά λάθη χωρίζονται σε δύο βασικές κατηγορίες, τα συντακτικά λάθη, τα οποία είναι λάθη στην σύνταξη των εντολών και τα λάθη εκτέλεσης ή αλλιώς λογικά λάθη. Τα πρώτα μπορούν να λυθούν εύκολα όταν γράφουμε σε πραγματικό προγραμματιστικό περιβάλλον, ενώ τα δεύτερα είναι πιο δύσκολο να εντοπιστούν.

Παράδειγμα χρήσης

Ας δούμε ένα παράδειγμα χρήσης και τα πιθανά λάθη που μπορεί να κάνει ο μαθητής. Η εκφώνηση είναι και πάλι απλή και λέει να φτιαχτεί ένα πρόγραμμα το οποίο θα διαβάζει τους αριθμούς που δίνει ο χρήστης, μέχρι ο χρήστης να δώσει τον αριθμό μηδέν, ο οποίος ορίζεται σαν συνθηματικό για τον τερματισμό των επαναλήψεων. Στο τέλος θέλουμε να εμφανιστεί το πλήθος των αριθμών που εισήγαγε ο χρήστης.

Μία πρώτη λύση που μπορεί να δώσει ο μαθητής είναι η ακόλουθη:

- Μία εντολή «Διάβασε», αφού το πρόγραμμα πρέπει να πάρει δεδομένα από τον χρήστη,
- Μία επανάληψη «Όσο» που η συνθήκη της θα είναι η είσοδος να μην είναι μηδέν,
- Μία αύξηση κατά ένα σε κάθε επανάληψη ώστε να μετράει πόσοι αριθμοί πέρασαν,
- Μια εντολή «Γράψε» για να εμφανιστεί το πλήθος των αριθμών.

Το σημαντικότερο πρόβλημα είναι ότι αν ο μαθητής λύνει την άσκηση στο χαρτί, δεν έχει κανέναν απολύτως τρόπο να επιβεβαιώσει την ορθότητά της. Σ' αυτό το σημείο είναι απαραίτητη η χρήση ενός προγραμματιστικού περιβάλλοντος, το οποίο θα δείξει στον μαθητή ότι η λύση του έχει και δύο συντακτικά αλλά και πολλά λογικά λάθη.

«Δεν είναι δυνατή η πράξη «όχι» στην ακέραια έκφραση 0». Αυτό είναι ένα συντακτικό λάθος και σημαίνει ότι η συνθήκη που έγραψε ο μαθητής δεν είναι σωστά διατυπωμένη. Με λίγη σκέψη ο μαθητής μπορεί να βρει ότι ο σωστός τρόπος να την γράψει είναι «όχι χ = 0», η πιο απλά «χ <> 0».

Ξαναπροσπαθεί να εκτελέσει το πρόγραμμα και βλέπει ένα ακόμα μήνυμα συντακτικού λάθους, το «άγνωστο αναγνωριστικό: επανέλαβε». Άγνωστα αναγνωριστικά ονομάζονται μεταβλητές που δεν έχουν δηλωθεί στο τμήμα δήλωσης μεταβλητών του προγράμματος. Όμως ο μαθητής δεν ήθελε να γράψει κάποια μεταβλητή, αλλά μία εντολή της ΓΛΩΣΣΑΣ, δηλαδή δεσμευμένη λέξη. Κοιτάζοντας το χρώμα της λέξης μπορεί να παρατηρήσει ότι αυτό είναι μαύρο αντί για μπλε, όπως είναι όλες οι άλλες εντολές, και επομένως ο διερμηνευτής δεν την αναγνωρίζει σαν δεσμευμένη λέξη. Αφού την διορθώσει, επαναρχίζει η εκτέλεση του προγράμματος.

Περιπτώσεις προγραμματιστικών λαθών παραδείγματος χρήσης

Ατέρμονη επανάληψη. Ο μαθητής εισάγει τον πρώτο αριθμό και παρατηρεί ότι το πρόγραμμα εκτελείται, χωρίς να του ζητάει κάτι άλλο. Μετά από λίγα δευτερόλεπτα ο διερμηνευτής εμφανίζει το μήνυμα «Αυτό το πρόγραμμα έχει ήδη εκτελέσει 1.000.000 εντολές. Αν νομίζετε ότι βρίσκεται σε ατέρμονα βρόχο, μετά το κλείσιμο του διαλόγου πατήστε [Pause] ή «Παύση εκτέλεσης» για να διακόψετε την εκτέλεσή του». Ο μαθητής μπορεί να πατήσει το πλήκτρο τη παύσης και να διαπιστώσει ότι το πρόγραμμα ανακυκλώνει τις εντολές που βρίσκονται μέσα στην «Όσο», χωρίς να διαβάζει άλλα δεδομένα. Μπορεί λοιπόν να συμπεράνει ότι πρέπει να βάλει την εντολή «Διάβασε» μέσα στην εντολή «Όσο».

Μη αρχικοποιημένη μεταβλητή. Ένα άλλο δύσκολο σημείο είναι η κατανόηση της έννοιας της μεταβλητής (Τζιμογιάννης & Κόμης, 2000). Ο μαθητής ξεκινά την εκτέλεση του προγράμματος και αυτή τη φορά η εκτέλεση σταματάει αμέσως και εμφανίζεται το μήνυμα λάθους «Το πρόγραμμα προσπάθησε να χρησιμοποιήσει την τιμή της μεταβλητής «χ» χωρίς αυτή να έχει αρχικοποιηθεί». Πράγματι, δεν μπορούμε να γίνει έλεγχος αν το χ είναι διαφορετικό από το μηδέν αν δεν έχει αποδοθεί μία συγκεκριμένη αρχική τιμή. Έτσι ο μαθητής μπορεί να γράψει την εντολή «χ ← 0» πάνω από την εντολή «Όσο».

Άπειρες εντολές «Διάβασε». Το πρόγραμμα φαίνεται να εκτελείται κανονικά, ο μαθητής εισάγει κάποιους αριθμούς, ενώ κανένα μήνυμα λάθους δεν εμφανίζεται. Όταν όμως ο μαθητής δώσει την τιμή «0» για τερματισμό, παρατηρεί ότι το πρόγραμμα δεν σταματάει αλλά συνεχίζει να ζητάει κι άλλα δεδομένα. Στο σημείο αυτό ο μαθητής μπορεί να κάνει βηματική εκτέλεση και να παρατηρήσει ότι δίνοντας μία τιμή στην εντολή «Διάβασε», αυτή καταστρέφει τον μετρητή. Έτσι μπορεί να καταλάβει ότι χρειάζονται δύο μεταβλητές, μία ως μετρητής του πλήθους των αριθμών και μία για να «κρατάει» τον αριθμό που διαβάστηκε.

Συμπεράσματα

Συμπερασματικά μπορούμε να πούμε ότι ο μαθητής χρησιμοποιώντας τον Διερμηνευτή έχει την δυνατότητα να ελέγξει την ορθότητα ενός προγράμματος, και μάλιστα να κατορθώσει να διορθώσει όλα τα συντακτικά λάθη και τουλάχιστον κάποια από τα λογικά λάθη. Σε αυτό το σημείο μπορούμε να παρατηρήσουμε ότι η βοήθεια που προσφέρει ένα τέτοιο περιβάλλον βρίσκεται σε άμεση συνάρτηση με τα μηνύματα λάθους. Θα μπορούσε να υλοποιηθεί ένα «έξυπνο» σύστημα βοήθειας (context - sensitive help) το οποίο να βοηθά αποτελεσματικά το μαθητή στον εντοπισμό των λογικών του λαθών.

Βιβλιογραφία

Rogalski, J. & Samurçay, R. (1990) Acquisition of programming knowledge and skills. In J-M. Hoc, T.R.G. Green, D.J. Gilmore and R. Samurçay (Eds.) Psychology of Programming, pp 157-174. London: Academic Press.

Κόμης Β., Διδακτική της Πληροφορικής, Εκδόσεις Ελληνικό Ανοικτό Πανεπιστήμιο, Πάτρα 2001.

Πολίτης Π., Κόμης Β., Η Πληροφορική ως βασικό μάθημα της Γ' τάξης Τεχνολογικής Κατεύθυνσης του Ενιαίου Λυκείου: αλγοριθμική έναντι προγραμματιστικής προσέγγισης, Βάση, Τεύχος 2.

Τζιμογιάννης Α. & Γεωργίου Β. (1999), Οι δυσκολίες μαθητών δευτεροβάθμιας εκπαίδευσης στην εφαρμογή της δομής ελέγχου για την ανάπτυξη αλγορίθμων. Μία μελέτη περίπτωσης, Στο Α. Τζιμογιάννης (επιμ.). Πρακτικά Πανελλήνιου Συνεδρίου «Πληροφορική και Εκπαίδευση», Σύλλογος Καθηγητών Πληροφορικής Ηπείρου, 183-192.

Τζιμογιάννης Α., Κόμης Β., Η έννοια της μεταβλητής στον Προγραμματισμό: δυσκολίες και παρανοήσεις μαθητών του Ενιαίου Λυκείου, Κόμης Β. (επιμέλεια), 2ο Πανελλήνιο Συνέδριο με Διεθνή Συμμετοχή «Οι Τεχνολογίες της Πληροφορίας και της Επικοινωνίας στην Εκπαίδευση», Πανεπιστήμιο Πατρών, Πάτρα, Οκτώβριος 2000.

ΥΠΕΠΘ, Παιδαγωγικό Ινστιτούτο (1997), Ενιαίο Πλαίσιο Προγράμματος Σπουδών Πληροφορικής, Αθήνα.

ΥΠΕΠΘ, Παιδαγωγικό Ινστιτούτο (1998), Η Πληροφορική στο σχολείο, Αθήνα.