Η Διδασκαλία της Κληρονομικότητας στον Προγραμματιστικό Μικρόκοσμο objectKarel*

Ξυνόγαλος Στέλιος[1], Σατρατζέμη Μάγια[1], Δαγδιλέλης Βασίλειος[2], Ευαγγελίδης Γεώργιος[1]

[1] Τμήμα Εφαρμοσμένης Πληροφορικής, Πανεπιστήμιο Μακεδονίας
[2] Τμήμα Εκπαιδευτικής και Κοινωνικής Πολιτικής, Πανεπιστήμιο Μακεδονίας
stelios@uom.gr, maya@uom.gr, dagdil@uom.gr, gevan@uom.gr

ΠΕΡΙΛΗΨΗ

Η διδασκαλία και εκμάθηση του αντικειμενοστραφούς προγραμματισμού παρουσιάζει αρκετές δυσκολίες. Για την αντιμετώπιση των δυσκολιών αυτών έχει προταθεί από ερευνητές και διδάσκοντες η χρήση εκπαιδευτικών προγραμματιστικών περιβαλλόντων και εναλλακτικών διδακτικών προσεγγίσεων. Στην παρούσα εργασία παρουσιάζεται μια πρόταση διδασκαλίας που βασίζεται στον διδακτικό μικρόκοσμο objectKarel. Η προβληματική που διέπει την προτεινόμενη διδακτική προσέγγιση, αλλά και τη σχεδίαση του περιβάλλοντος objectKarel, παρουσιάζεται μέσω της αναλυτικής περιγραφής της πορείας και των αποτελεσμάτων ενός από τα πέντε μαθήματα που υλοποιήθηκαν στα πλαίσια της πιλοτικής εφαρμογής του με φοιτητές ενός Τμήματος Πληροφορικής. Το μάθημα που παρουσιάζεται αναφέρεται σε μία από τις θεμελιώδεις έννοιες του αντικειμενοστραφούς προγραμματισμού, την έννοια της κληρονομικότητας, για την οποία έχουν καταγραφεί αρκετές δυσκολίες και παρανοήσεις. Συγκεκριμένα, όπως προκύπτει από σχετικές μελέτες, οι σπουδαστές δυσκολεύονται να χρησιμοποιήσουν, να σχεδιάσουν και να υλοποιήσουν κλάσεις, να κατανοήσουν τη χρησιμότητα των κλάσεων και της κληρονομικότητας, καθώς επίσης και τη λειτουργικότητα των μεθόδων.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: *Διδακτική Προγραμματισμού, αντικειμενοστραφής προγραμματισμός, προγραμματιστικός μικρόκοσμος*

ΕΙΣΑΓΩΓΗ

Η διδασκαλία και η εκμάθηση του αντικειμενοστραφούς προγραμματισμού, όπως αποδεικνύουν σχετικές έρευνες (Carter & Fowler 1998, Holland et al. 1997), συνοδεύεται από πολλές δυσκολίες για τους διδάσκοντες και τους σπουδαστές αντίστοιχα. Ένα από τα σημαντικότερα διδακτικά προβλήματα έγκειται στο γεγονός ότι η αντικειμενοστραφής τεχνική ανάπτυξης προγραμμάτων είναι δύσκολη για τους σπουδαστές, μιας και είναι πιο αφηρημένη, απαιτεί νέους τρόπους σκέψης και είναι πιο απαιτητική όσον αφορά στις διαδικασίες της ανάλυσης και της σχεδίασης σε σχέση με

* *Η εργασία αυτή χρηματοδοτείται από το ΥΠΕΠΘ και την Ευρωπαϊκή Ένωση στα πλαίσια του έργου «ΠΥΘΑΓΟΡΑΣ: «Ενίσχυση Ερευνητικών Ομάδων στα Πανεπιστήμια (ΕΕΟΠ)»*

την τεχνική του δομημένου προγραμματισμού (Handjerrouit 1999). Οι προτάσεις που έχουν γίνει από ερευνητές/διδάσκοντες για την αντιμετώπιση των δυσκολιών αφορούν κατά κύριο λόγο στη χρήση εκπαιδευτικών προγραμματιστικών περιβαλλόντων ή/και τη χρήση εναλλακτικών προσεγγίσεων διδασκαλίας. Η πρόταση που φαίνεται να έχει τη μεγαλύτερη αποδοχή είναι η πρόταση των (Kolling et al. 2003), όπου χρησιμοποιείται το περιβάλλον BlueJ και η γλώσσα Java.

Στην παρούσα εργασία γίνεται μια πρόταση για τη διδασκαλία των εννοιών του αντικειμενοστραφούς προγραμματισμού, η οποία βασίζεται στην προσέγγιση των μικρόκοσμων. Συγκεκριμένα, προτείνεται η χρήση του προγραμματιστικού μικρόκοσμου objectKarel (Ξυνόγαλος 2002) - που χρησιμοποιεί τη γλώσσα προγραμματισμού του ρομπότ Karel++ (Bergin et al. 1997) - και της σειράς των μαθημάτων που συνοδεύουν το λογισμικό. Τα παιδαγωγικά χαρακτηριστικά του objectKarel είναι: (1) ένας *εκδότης δομής* για την ευκολότερη ανάπτυξη των προγραμμάτων και την αποφυγή της επικέντρωσης στις συντακτικές λεπτομέρειες της γλώσσας προγραμματισμού, (2) η αναφορά φιλικών προς το χρήστη μηνυμάτων λάθους, (3) ένα *σύστημα δυναμικής προσομοίωσης εκτέλεσης των προγραμμάτων και οπτικοποίησης*, το οποίο περιλαμβάνει τη δυνατότητα της βήμα προς βήμα εκτέλεσης και της επεξηγηματικής οπτικοποίησης, της εμφάνισης δηλαδή επεξηγήσεων σε φυσική γλώσσα για τη σημασία της τρέχουσας εντολής, και (4) μια *σειρά μαθημάτων*. Τα μαθήματα που έχουν ενσωματωθεί στο περιβάλλον του objectKarel περιλαμβάνουν σύντομη και περιεκτική *θεωρία* και *δραστηριότητες* για την εξοικείωση των σπουδαστών με τις έννοιες του αντικειμενοστραφούς προγραμματισμού, πριν να τους ζητηθεί να τις χρησιμοποιήσουν για την ανάπτυξη προγραμμάτων. Η λογική της κατασκευής κάθε δραστηριότητας έχει ως στόχο είτε να παρουσιάσει κάθε διδασκόμενη έννοια με άμεσο τρόπο - συνήθως οπτικό, όπως για παράδειγμα δημιουργία αντικειμένου και αποστολή μηνυμάτων σε αντικείμενα με το πάτημα κουμπιών, ταυτόχρονη εμφάνιση του κώδικα και αναπαράσταση της λειτουργίας στην οθόνη, είτε να οδηγήσει τον αρχάριο σε αδιέξοδο ώστε να θεωρήσει ως φυσική συνέπεια την υιοθέτηση των νέων εννοιών, όπως για παράδειγμα την έννοια της κληρονομικότητας. Στη συνέχεια, λόγω της περιορισμένης έκτασης της εργασίας, παρουσιάζεται η διδακτική προβληματική που διέπει τη σχεδίαση του περιβάλλοντος objectKarel και του διδακτικού υλικού μέσω της περιγραφής της πορείας και των αποτελεσμάτων του 2ου από τα 5 μαθήματα που πραγματοποιήθηκαν στα πλαίσια της πιλοτικής εφαρμογής με φοιτητές ενός Τμήματος Πληροφορικής και είχε ως στόχο τη διδασκαλία της έννοιας της κληρονομικότητας.

ΤΟ ΣΧΕΔΙΟ ΤΟΥ ΜΑΘΗΜΑΤΟΣ «ΚΛΗΡΟΝΟΜΙΚΟΤΗΤΑ»

Ο γενικός **διδακτικός στόχος** του μαθήματος με τίτλο «Κληρονομικότητα» είναι: 1) να κατανοήσουν οι φοιτητές την έννοια της κληρονομικότητας, τα πλεονεκτήματα της δημιουργίας νέων κλάσεων και της επαναχρησιμοποίησης/τροποποίησης υπαρχουσών κλάσεων, και 2) να εξοικειωθούν με τη δημιουργία νέων κλάσεων και μεθόδων. Οι βασικές **έννοιες** στις οποίες αναφέρεται το μάθημα είναι: κληρονομικότητα (inheritance), γονική κλάση (parent class), δήλωση (declaration), ορισμός (definition),

εμβέλεια τελεστή (scope resolution operator) και λεξικό (dictionary). Στη συνέχεια περιγράφεται η **πορεία του μαθήματος**, διάρκειας δύο διδακτικών ωρών.

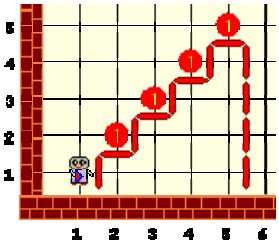
Αρχικά, γίνεται **σύνδεση με το προηγούμενο μάθημα**, στο οποίο διδάχθηκαν τις έννοιες: αντικείμενο (object), μήνυμα (message) – μέθοδος (method) και κλάση (class). Συγκεκριμένα, στο 1ο μάθημα παρουσιάστηκε στους φοιτητές ο μικρόκοσμος και το βασικό μοντέλο (κλάση) ρομπότ με όνομα Primitive_Robot, το οποίο μπορεί με την κατάλληλη εντολή να μας εφοδιάσει με ρομπότ που ανταποκρίνονται στα εξής μηνύματα: (1) στο μήνυμα move() μεταβαίνουν στην επόμενη διασταύρωση, (2) στο μήνυμα turnLeft() στρίβουν αριστερά κατά 90 μοίρες, (3) στο μήνυμα pickBeeper() σηκώνουν με το μηχανικό τους χέρι 1 beeper (μικρό πλαστικό κώνο που παράγει ένα ήχο – beep) και το τοποθετούν στην ηχομονωμένη τσάντα τους, (4) στο μήνυμα putBeeper() βγάζουν 1 beeper από την τσάντα τους και το τοποθετούν στην τρέχουσα διασταύρωση. Οι φοιτητές εξοικειώθηκαν με τις παραπάνω έννοιες και τον μικρόκοσμο με μια σειρά δραστηριοτήτων που περιλάμβαναν: τη δημιουργία ρομπότ και την αποστολή μηνυμάτων σε αυτά χρησιμοποιώντας τεχνικές άμεσης διαχείρισης (πατώντας κουμπιά) χωρίς να γράφουν κώδικα, τη βηματική εκτέλεση έτοιμων προγραμμάτων και την ανάπτυξη προγραμμάτων σε ομάδες.

Μετά από τη σύνδεση με το προηγούμενο μάθημα χρησιμοποιείται το **παράδειγμα του ρομπότ-ταξιδιώτη** που περιλαμβάνεται στην θεωρία της ενότητας «Κληρονομικότητα». Το πρόβλημα διατυπώθηκε ως εξής: *«Ένα ρομπότ πρέπει να προγραμματιστεί έτσι ώστε να ταξιδεύει διανύοντας μεγάλες αποστάσεις. Υποθέτουμε ότι το ρομπότ ξεκινάει από την 1η λεωφόρο και τον 2ο δρόμο και πρέπει να κινηθεί ανατολικά κατά μήκος του 2ου δρόμου για 10 χιλιόμετρα (1 χλμ=8 μπλοκ), να σηκώσει ένα beeper και στη συνέχεια να κινηθεί 5 χλμ βόρεια».* Μετά από συζήτηση με τους φοιτητές επισημαίνεται ότι η λύση στο πρόβλημα πρέπει να μεταφραστεί σε εντολές που μετακινούν το ρομπότ κατά ένα μπλοκ κάθε φορά, αφού τα ρομπότ της κλάσης Primitive_Robot δεν κατανοούν την έννοια του χιλιομέτρου. Αυτό σημαίνει ότι το πρόγραμμα θα περιλαμβάνει *120 εντολές* move() και θα έχει την εξής μορφή (στους φοιτητές παρουσιάζεται το πρόγραμμα στην πλήρη έκτασή του):

```
task
{
      Primitive_Robot Karel (1, 2, East, 0);
      Karel.move(); ... // 79 φορές η εντολή Karel.move()
      Karel.pickBeeper();
      Karel.turnLeft():
      Karel.move(); ... // 39 φορές η εντολή Karel.move();
}
```

Μέσα από το παράδειγμα αυτό οι φοιτητές κατανοούν ότι πολλές φορές, ακόμη και για τη λύση απλών προβλημάτων όπως το πρόβλημα του ρομπότ-ταξιδιώτη, απαιτείται η ανάπτυξη προγραμμάτων με πολύ μεγάλο μέγεθος με αποτέλεσμα να είναι δύσκολη η κατανόηση, η αποσφαλμάτωση και πολύ περισσότερο η τροποποίησή τους για την επίλυση παρόμοιων προβλημάτων. Στην ουσία με το παράδειγμα αυτό οι φοιτητές αναγνωρίζουν τη σημασία της επέκτασης των δυνατοτήτων του βασικού μοντέλου (κλάσης) ρομπότ έτσι ώστε τα ρομπότ να ανταποκρίνονται και σε άλλα μηνύματα. Αυτή

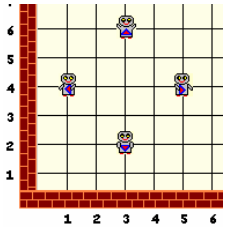
είναι η ιδανική στιγμή για να παρουσιαστεί στους φοιτητές η έννοια της κληρονομικότητας. Επισημαίνεται ότι το πρόβλημα του κώδικα που αναπτύχθηκε για το παράδειγμα του ρομπότ-ταξιδιώτη έγκειται στο γεγονός ότι η λύση πρέπει να κάνει χρήση των στοιχειωδών μεθόδων της κλάσης Primitive_Robot. Προτείνεται, λοιπόν ως λύση στο πρόβλημα αυτό η **δημιουργία νέων κλάσεων** ρομπότ που κληρονομούν τις δυνατότητες της κλάσης Primitive_Robot και τις επεκτείνουν - τροποποιούν. Για την παρουσίαση της δυνατότητας αυτής δίνεται 2η λύση για το πρόβλημα του ρομπότ-ταξιδιώτη που περιλαμβάνει τη δήλωση της νέας κλάσης Klm_Walker, η οποία έχει ως γονική κλάση την Primitive_Robot και περιλαμβάνει τη νέα μέθοδο moveKLm() που επιτρέπει στα ρομπότ να μετακινούνται κατά 1 χλμ. Χρησιμοποιώντας το πρόγραμμα αυτό εξηγούνται οι έννοιες κληρονομικότητα, γονική κλάση, λεξικό και ο τρόπος δήλωσης κλάσεων και ορισμού μεθόδων.



Στη συνέχεια χρησιμοποιείται η **δραστηριότητα** της ενότητας «Κληρονομικότητα», στα πλαίσια της οποίας παρουσιάζεται το πρόβλημα του «σκουπίσματος σκαλιών» (το ρομπότ πρέπει να καθαρίσει τα beepers) και γίνεται συζήτηση με τους φοιτητές για τη λύση του. Οι φοιτητές μελετούν και εκτελούν δύο προγράμματα για το παραπάνω πρόβλημα, από τα οποία στο 1ο χρησιμοποιείται ένα ρομπότ της κλάσης Primitive_Robot, ενώ στο 2ο ένα ρομπότ μιας νέας κλάσης. Επιπλέον, τους ζητείται να επιλέξουν ένα από τα 2 προγράμματα προκειμένου να κατευθύνουν ένα ρομπότ να σκουπίσει μια σκάλα με 15 σκαλιά και να αιτιολογήσουν την απάντησή τους.

Ο κώδικας της παραπάνω δραστηριότητας δίνεται σε έντυπη μορφή και οι φοιτητές καθοδηγούμενοι από τον διδάσκοντα χρησιμοποιούν τον εκδότη δομής του objectKarel για να μεταφέρουν τον κώδικα, ενώ στη συνέχεια κάνουν διορθώσεις στην κλάση. Στόχος μας είναι η εξοικείωση των φοιτητών με τον εκδότη δομής του objectKarel πριν ξεκινήσουν να αναπτύσσουν μόνοι τους προγράμματα με κλάσεις.

Μετά το πέρας των δραστηριοτήτων δίνεται στους φοιτητές το **φύλλο εργασίας** που περιλαμβάνει τις δύο ασκήσεις που περιγράφονται στη συνέχεια:



1η Άσκηση: *Να δημιουργήσετε μια νέα κλάση με όνομα AugmentedRobot, τα ρομπότ της οποίας θα είναι ικανά να ανταποκρίνονται στα μηνύματα: 1) turnRight: στρίβοντας δεξιά κατά 90 μοίρες, και 2) turnAround: κάνοντας στροφή 180 μοιρών. Στη συνέχεια να δημιουργήσετε 4 ρομπότ, όπως φαίνεται στην διπλανή Εικόνα, και να στείλετε σε κάθε ένα από αυτά το μήνυμα turnRight.*

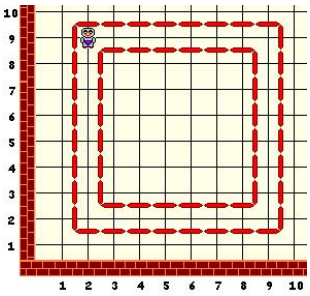
Ο **πηγαίος κώδικας** του προγράμματος που καλούνται να αναπτύξουν οι φοιτητές και οι **στόχοι** της 1ης άσκησης είναι οι εξής:

```
class AugmentedRobot:Primitive_Robot
{
  void turnRight();
  void turnAround();
};
void AugmentedRobot::turnRight()
{
  turnLeft();
  turnLeft();
  turnLeft();
}
void AugmentedRobot::turnAround()
{
  turnLeft();
  turnLeft();
}
```

```
task
{
  // Δημιουργία και αρχικοποίηση των
  // αντικειμένων (ρομπότ).
  AugmentedRobot Nik(4, 1, West, 0);
  AugmentedRobot Marta (2,3,South,0);
  AugmentedRobot Josef (4,5,East,0);
  AugmentedRobot Baso (6,3,North,0);
  //Αποστολή του μηνύματος turnRight
  //στα ρομπότ.
  Nik.turnRight();
  Marta.turnRight();
  Josef.turnRight();
  Baso.turnRight();
}
```

- Ο βασικότερος στόχος αυτής της άσκησης είναι να εξακριβωθεί αν επαληθεύεται το συμπέρασμα των (Holland et al., 1997) οι οποίοι αναφέρουν ότι μία από τις παρανοήσεις των φοιτητών είναι η **ταύτιση της κλάσης και του αντικειμένου** ή το συμπέρασμα των (Carter & Fowler, 1998) οι οποίοι αναφέρουν ότι **ο διαχωρισμός μεταξύ των αντικειμένων και των κλάσεων δεν δημιουργεί πρόβλημα** στους φοιτητές. Οι *ερωτήσεις* που ενδεχομένως θα διατυπωθούν και τα *προγράμματα* που θα αναπτυχθούν *θα μας επιτρέψουν να διαπιστώσουμε ποιο από τα δύο αντικρουόμενα συμπεράσματα επαληθεύεται*.
- *Η κλάση που θα δημιουργήσουν οι φοιτητές θα χρησιμοποιηθεί σε επόμενα μαθήματα* ως γονική νέων κλάσεων, προκειμένου να κατανοήσουν οι φοιτητές καλύτερα την έννοια της κληρονομικότητας και τα πλεονεκτήματα της δημιουργίας κώδικα (κλάσεων) που μπορεί να επαναχρησιμοποιηθεί.

2η Άσκηση: *Ο Karel αποφάσισε να δουλέψει σε μια εταιρία καθαριότητας. Το καθήκον του είναι να στρώνει χαλιά (φτιαγμένα από beepers) σε κτίρια που υπάρχουν στον κόσμο του. Γράψε ένα πρόγραμμα που καθοδηγεί τον Karel στο στρώσιμο του χαλιού στο κτίριο που φαίνεται στην διπλανή Εικόνα. Δεν πρέπει να υπάρχουν «εξογκώματα» στο χαλί, γεγονός που σημαίνει ότι ο Karel πρέπει να τοποθετεί μόνο ένα beeper σε κάθε γωνία του διαδρόμου. Ο Karel ξεκινάει την αποστολή του έχοντας στην τσάντα του όσα ακριβώς beepers χρειάζονται.*



Ο **πηγαίος κώδικας** του προγράμματος και οι **στόχοι** της 2ης άσκησης είναι οι εξής:

```
class
CarpetLayer:Primitive_Robot
{
       void carpetOneHall();
       void
carpetFourHalls();
};
void
CarpetLayer::carpetOneHall()
{
     putBeeper();
     move();
     putBeeper();
     move();
     putBeeper();
     move();
     putBeeper();
     move();
     putBeeper();
     move();
     putBeeper();
     move();
     putBeeper();
     move();
```

```
putBeeper();
     move();
}
void CarpetLayer::carpetFourHalls()
{
     carpetOneHall();
     turnLeft();
     carpetOneHall();
     turnLeft();
     carpetOneHall();
     turnLeft();
     carpetOneHall();
     turnLeft();
}
task
{
     CarpetLayer Karel (9, 2, South, 28);
     Karel.carpetFourHalls();
}
```

- Ο βασικότερος στόχος της 2ης άσκησης είναι η *εξοικείωση των φοιτητών με τη δήλωση κλάσεων και τον ορισμό μεθόδων*, καθώς επίσης και με τη διόρθωσή τους.
- *Ο ορισμός του προβλήματος πιστεύουμε ότι δεν πρέπει να αναφέρεται στον Karel* - και γενικότερα να χρησιμοποιούνται εκφράσεις όπως *«Γράψε ένα πρόγραμμα που καθοδηγεί τον Karel…»* που επαναλαμβάνονται συνεχώς στο βιβλίο "Karel++: A Gentle Introduction to the Art of Object-Oriented Programming" (Bergin et al., 1995) - *γιατί ενδεχομένως να δημιουργηθεί η παρανόηση ότι η κλάση και το αντικείμενο είναι έννοιες ταυτόσημες, ειδικά όταν στις περισσότερες ασκήσεις δημιουργείται ένα μόνο αντικείμενο μιας δεδομένης κλάσης*. Αντίθετα, πρέπει να τονίζεται ότι μια κλάση μπορεί να μας εφοδιάσει με όσα αντικείμενα θέλουμε εμείς, χρησιμοποιώντας εκφράσεις όπως *«Δημιούργησε μια νέα κλάση που θα μας εφοδιάσει με ρομπότ ικανά να….»*. Ωστόσο, στην άσκηση αυτή το πρόβλημα διατυπώθηκε όπως υπάρχει στο προαναφερθέν βιβλίο γιατί θέλαμε να διαπιστώσουμε αν οι φοιτητές κατανόησαν την έννοια και τη χρησιμότητα της κλάσης, οπότε θα δημιουργήσουν μια νέα κλάση, ή αν θα λύσουν το πρόβλημα χωρίς τη δημιουργία νέας κλάσης.

ΤΑ ΣΥΜΠΕΡΑΣΜΑΤΑ ΤΟΥ ΜΑΘΗΜΑΤΟΣ

Στη συνέχεια παρουσιάζονται τα συμπεράσματα που εξήχθησαν στο 2ο μάθημα της πιλοτικής εφαρμογής στην οποία συμμετείχαν 19 φοιτητές του Τμήματος Εφαρμοσμένης Πληροφορικής που χωρίστηκαν σε 10 ομάδες. Οι 5 πρώτες ομάδες (η 3η αποτελούνταν από 1 άτομο) αποτελούνταν από φοιτητές του Β' Εξαμήνου, οι 3 επόμενες από φοιτητές του Δ' εξαμήνου και οι 2 τελευταίες του ΣΤ' εξαμήνου. Όλοι οι φοιτητές είχαν αποτύχει στις εξετάσεις του μαθήματος «Προγραμματισμός Ι» που έχει ως στόχο την εισαγωγή των φοιτητών στο διαδικαστικό προγραμματισμό με την Pascal.

Επιπλέον, οι φοιτητές του Δ' και ΣΤ' εξαμήνου είχαν παρακολουθήσει και το μάθημα «Αντικειμενοστραφής Προγραμματισμός» και η πλειοψηφία τους (οι 6 από τους 10) είχε αποτύχει στις εξετάσεις και αυτού του μαθήματος.

Στους Πίνακες 1 και 2 παρουσιάζονται τα στοιχεία που προέκυψαν από την **ανάλυση των διαδοχικών εκδόσεων** (αποθηκεύονται αυτόματα στο περιβάλλον του objectKarel) των προγραμμάτων των φοιτητών. Τα συμπεράσματα που εξήχθησαν είναι τα εξής:

- Όλες οι ομάδες ανέπτυξαν *σωστά προγράμματα και για τις 2 ασκήσεις*, ενώ δημιούργησαν την αρχική κατάσταση για την $2^{η}$ άσκηση χωρίς προβλήματα.
- Όσον αφορά στη *στρατηγική επίλυσης προβλημάτων*, η πλειοψηφία των φοιτητών (8 ομάδες στην $1^{η}$ και 7 ομάδες στη $2^{η}$ άσκηση) αναπτύσσει ολόκληρο το πρόγραμμα και στη συνέχεια το ελέγχει.
- Τα μισά προγράμματα δεν περιλάμβαναν κανένα ή περιλάμβαναν ένα μόνο λάθος. Γενικά, στην πλειοψηφία των προγραμμάτων εντοπίστηκαν ελάχιστα λάθη, τα οποία και διορθώθηκαν άμεσα. Ακόμη και *τα λογικά λάθη που δεν οδηγούν σε λάθη εκτέλεσης εντοπίζονται άμεσα*. Τα αποτελέσματα αυτά βέβαια, τα οποία καταγράφηκαν και στο $1^{ο}$ μάθημα, ήταν αναμενόμενα λόγω των φιλικών μηνυμάτων λάθους και των τεχνικών οπτικοποίησης του objectKarel.
- Από τις ερωτήσεις που διατύπωσαν και τα προγράμματα που ανέπτυξαν οι φοιτητές για την $1^{η}$ άσκηση προκύπτει ότι **η πλειοψηφία των φοιτητών δεν συγχέει τις έννοιες κλάση και αντικείμενο**. Συγκεκριμένα, μία μόνο από τις ομάδες διατύπωσε, κατά την ανάπτυξη του προγράμματος, στον διδάσκοντα ερώτηση που υποδηλώνει την ύπαρξη της εν λόγω παρανόησης που αναφέρουν οι (Holland et al., 1997). Η ομάδα αυτή ρώτησε αν *χρειαζόταν να δημιουργήσει 4 κλάσεις,* όσα δηλαδή ήταν και τα αντικείμενα που έπρεπε να δημιουργήσει.
- Όλες οι ομάδες ξεκίνησαν τη $2^{η}$ άσκηση δηλώνοντας μια νέα κλάση, παρόλο που ο ορισμός του προβλήματος αναφέρεται συνεχώς σε ένα ρομπότ με όνομα Karel και δεν αναφέρεται καθόλου στη δημιουργία νέας κλάσης. Μπορούμε λοιπόν να συμπεράνουμε ότι *οι φοιτητές κατανόησαν την έννοια και τη χρησιμότητα της κλάσης.*
- Τα παραπάνω αποτελέσματα ήταν αναμενόμενα λόγω της έμφασης που δίνεται στις διδασκόμενες έννοιες στο περιβάλλον του objectKarel. Αναμενόμενη δεν ήταν η παρακάτω υλοποίηση (ομάδες 1, 3 και 8) της μεθόδου turnAround() της $1^{ης}$ άσκησης:

```
void AugmentedRobot : turnAround()
{
    turnRight();
     turnRight();
}
```

Παρόλο που ένα ρομπότ θα ανταποκριθεί σωστά στο μήνυμα turnAround() θα εκτελέσει έξι αντί για δύο φορές το μήνυμα turnLeft(). Η υλοποίηση αυτή επιβεβαιώνει το **πρόβλημα της προηγούμενης εμπειρίας** (Sprohrer & Soloway 1986) σύμφωνα με το οποίο η ανάπτυξη, ανακύκλωση και προσαρμογή σχεδίων βασίζεται σε προϋπάρχουσες εμπειρίες. Στη συγκεκριμένη περίπτωση είναι προφανές ότι βάσει της εμπειρίας από την καθημερινή ζωή δεν υπάρχει διαφορά μεταξύ αριστερόστροφης

και δεξιόστροφης αναστροφής. Ιδιαίτερο ενδιαφέρον θα είχε να δούμε ποια θα ήταν η αντίδραση των φοιτητών αν τους είχαμε ζητήσει να στείλουν το μήνυμα turnAround() στα ρομπότ και να εκτελέσουν το πρόγραμμά τους βηματικά.

Πίνακας 1: Στοιχεία από την ανάλυση των προγραμμάτων για την 1η άσκηση

		Λάθη					
Ομάδα	**Μεταγλ/σεις**	**Συντακτικά**	**Σημασ/ογικά**	**Λογικά**	**Χρόνος (λεπτά)**	**Στρατηγική Επίλυσης**	**Παρατηρήσεις**
1	3	-	-	1 *(1)*	4	1) Δημιουργία κλάσης, αντικειμένων και έλεγχος, 2) Στη συνέχεια αποστέλλουν τα μηνύματα στα ρομπότ	Υλοποίηση της turnAround χρησιμοποιώντας την turnRight
2	1	-	-	-	?	Ανάπτυξη προγράμματος και στη συνέχεια έλεγχος	
3	2	-	-	-	?	Ανάπτυξη προγράμματος και στη συνέχεια έλεγχος	1) Υλοποίηση της turnAround χρησιμοποιώντας την turnRight 2) Χρησιμοποίησε σχόλια
4	2	-	-	1 *(1)*	6	Ανάπτυξη προγράμματος και στη συνέχεια έλεγχος	
5	1	-	-	-	?	Ανάπτυξη προγράμματος και στη συνέχεια έλεγχος	
6	9	1 *(1)*	-	6 *(2,4, 5,6, 7,8)*	15	Ανάπτυξη προγράμματος και στη συνέχεια έλεγχος	1) Ρώτησαν αν χρειάζεται να δημιουργήσουν 4 κλάσεις (όσα είναι τα ρομπότ), 2) Από την 3η μεταγλώττιση είναι σωστό, αλλά στέλνουν τα 2 μηνύματα αρκετές φορές σε κάθε ρομπότ
7	3	1 *(1)*	-	-	5	1) Δημιουργία κλάσης & ενός ρομπότ, αποστολή μηνύματος & έλεγχος, 2) Δημιουργία υπόλοιπων ρομπότ & αποστολή μηνυμάτων	
8	1	-	-	-	?	Ανάπτυξη προγράμματος και στη συνέχεια έλεγχος	Υλοποίηση της turnAround χρησιμοποιώντας την turnRight
9	5	-	2 *(1, 3)*	1 *(2)*	8	Ανάπτυξη προγράμματος και στη συνέχεια έλεγχος	
10	2	-	-	1 *(1)*	2	Ανάπτυξη προγράμματος και στη συνέχεια έλεγχος	

Παρατήρηση: σε παρενθέσεις αναφέρεται ο αύξοντας αριθμός της έκδοσης του προγράμματος

Πίνακας 2: Στοιχεία από την ανάλυση των προγραμμάτων για την 2η άσκηση

Ομάδα	Μεταγλ/σεις	Λάθη: Συντακτικά	Λάθη: Σημασ/κά	Λάθη: Λογικά	Εκτέλεσης	Χρόνος (λεπτά)	Στρατηγική Επίλυσης	Παρατηρήσεις
1	10	-	6 *(4, 5,6,7 ,8,9)*	-	2 *(1, 2)*	23	Ανάπτυξη προγρ/τος & στη συνέχεια έλεγχος	Το πρόγ/μα είναι σωστό από την 3η μεταγλώττιση (16 λεπτά), αλλά κάνουν αλλαγές (όνομα κλάσης, μεθόδων) ίσως για να δοκιμάσουν τον εκδότη.
2	1	-	-	-	-	45		Τα 45 λεπτά είναι ο χρόνος από την αποθήκευση της 1ης άσκησης
3	4	2 *(2, 3)*	-	-	-	14	1) Δημιουργία κλάσης-ρομπότ, αποστολή μηνύ-ματος, έλεγχος, 2) ανάπ-τυξη υπόλοιπου προγρ/τος	
4	4	-	-	-	2 *(1,2)*	18	Ανάπτυξη προγρ/τος & στη συνέχεια έλεγχος	Κάνουν δεξιόστροφο στρώσιμο
5	4	1 *(3)*	-	1 *(1)*	-	40	Ανάπτυξη προγρ/τος & στη συνέχεια έλεγχος	Αρχικά τοποθετούν beepers μόνο στις γωνίες
6	2	-	-	-	1 *(1)*	31	Ανάπτυξη προγράμματος & στη συνέχεια έλεγχος	
7	2	-	-	-	1 *(2)*	12	Ανάπτυξη προγρ/τος & στη συνέχεια έλεγχος	
8	7	1 *(1)*	2 *(4,5)*	-	1 *(2)*	30	Ανάπτυξη προγρ/τος & στη συνέχεια έλεγχος	Αρχικά τοποθετούν beepers μόνο στις γωνίες
9	2	-	-	-	-	24	1) Δημιουργία κλάσης & ρομπότ, 2) Αποστολή μηνυμάτων & έλεγχος	Ορίζουν μέθοδο stripes που περιλαμβάνει μόνο την turnLeft
10	4	-	-	-	1 *(1)*	25	Ανάπτυξη προγρ/τος & στη συνέχεια έλεγχος	Το πρόγραμμα είναι σωστό από την 2η μεταγλώττιση

- Από τη σύγκριση των συγκεντρωτικών στοιχείων για τις δύο ασκήσεις προκύπτει ότι όλες οι ομάδες αφιέρωσαν πολύ περισσότερο χρόνο για τη 2η άσκηση, χωρίς ο αριθμός των μεταγλωττίσεων και των προβλημάτων που αντιμετώπισαν να είναι ιδιαίτερα μεγαλύτερος. Αν και θα μπορούσαμε να συμπεράνουμε ότι *η διαδικασία ανάπτυξης αλγορίθμων ακόμα και για απλά προβλήματα είναι ιδιαίτερα χρονοβόρα,* από την μελέτη των διαδοχικών εκδόσεων των 8 προγραμμάτων που η ανάπτυξή τους κράτησε περισσότερα από 15 λεπτά προκύπτει ότι: (1) το παραπάνω συμπέρασμα ισχύει για τρεις ομάδες (*2, 6, 9*), (2) δύο ομάδες (*1, 5*) δεν κατανόησαν σωστά τον ορισμό του προβλήματος, (3) μία ομάδα δοκίμασε τον εκδότη κάνοντας αλλαγές στο πρόγραμμα, το οποίο επανέφερε στην αρχική μορφή, (4) μία ομάδα (*10*) βελτίωσε το

πρόγραμμά της. Είναι προφανές λοιπόν, ότι η μελέτη των διαδοχικών εκδόσεων των προγραμμάτων μπορεί να μας αποτρέψει από την εξαγωγή λανθασμένων συμπερασμάτων και να μας βοηθήσει να εντοπίσουμε τις πραγματικές δυσκολίες, παρανοήσεις και προγραμματιστικές συνήθειες των φοιτητών.

Από την **παρατήρηση** των ομάδων θα θέλαμε να αναφέρουμε ότι έγιναν ελάχιστες ερωτήσεις από τους φοιτητές σχετικά με τον τρόπο δήλωσης των κλάσεων και τον ορισμό των μεθόδων στο προγραμματιστικό περιβάλλον, ενώ δεν παρατηρήθηκε καμία δυσκολία κατά την ανάπτυξη των προγραμμάτων. Όσον αφορά στη $2^{η}$ άσκηση, μία ομάδα διατύπωσε ερώτηση για το αν θα πρέπει το ρομπότ να τοποθετεί beeper μόνο στις γωνίες ή κατά μήκος του διαδρόμου. Επιπλέον, δύο ακόμη ομάδες ανέπτυξαν αρχικά πρόγραμμα που τοποθετούσε beepers μόνο στις γωνίες και στη συνέχεια το διόρθωσαν (ίσως μετά την ερώτηση που διατυπώθηκε). Καλό θα ήταν λοιπόν *στις ασκήσεις να δίνεται η εικόνα τόσο της αρχικής όσο και της τελικής κατάστασης* μιας και φαίνεται ότι η οπτική παρουσίαση των πληροφοριών γίνεται πολύ πιο εύκολα αντιληπτή.

Από την ανάλυση των δεδομένων του $2^{ου}$ μαθήματος έγινε εμφανές ότι θα πρέπει να γίνουν κάποιες **διορθώσεις στο διδακτικό υλικό**. Συγκεκριμένα, στην $1^{η}$ άσκηση πρέπει να ζητηθεί από τους σπουδαστές να στείλουν εκτός από το μήνυμα turnRight() και το μήνυμα turnAround() στα ρομπότ. Έτσι, στην περίπτωση που κάποιοι φοιτητές υλοποιήσουν τη μέθοδο turnAround() χρησιμοποιώντας την turnRight() και όχι την turnLeft(), θα μας δοθεί η δυνατότητα να διαπιστώσουμε αν η βηματική εκτέλεση θα τους βοηθήσει να εντοπίσουν το πρόβλημα. Επιπλέον, για όλες τις ασκήσεις: (1) ο ορισμός του προβλήματος πρέπει να αναφέρεται στη δημιουργία μιας κλάσης που μπορεί να μας εφοδιάσει με αντικείμενα και όχι στο ρομπότ Karel, γιατί ενδεχομένως να δημιουργηθεί η παρανόηση ότι κλάση και αντικείμενο είναι έννοιες ταυτόσημες, και (2) πρέπει να δίνεται η εικόνα τόσο της αρχικής όσο και της τελικής κατάστασης.

ΕΠΙΛΟΓΟΣ

Η διδασκαλία του αντικειμενοστραφούς προγραμματισμού παρουσιάζει αρκετές δυσκολίες, γεγονός που οδήγησε στη σχεδίαση ενός μεγάλου αριθμού εκπαιδευτικών προγραμματιστικών περιβαλλόντων, αλλά και ολοκληρωμένων προτάσεων διδασκαλίας. Στην παρούσα εργασία παρουσιάστηκε μια πρόταση διδασκαλίας βασισμένη στον προγραμματιστικό μικρόκοσμο objectKarel. Τα αποτελέσματα της πρώτης πιλοτικής εφαρμογής ήταν ιδιαίτερα ενθαρρυντικά. Ωστόσο, κρίνεται απαραίτητη η περαιτέρω έρευνα της προταθείσας προσέγγισης διδασκαλίας, η οποία πραγματοποιείται στα πλαίσια του προγράμματος ενίσχυσης ερευνητικών ομάδων στα Πανεπιστήμια «ΠΥΘΑΓΟΡΑΣ ΙΙ». Συγκεκριμένα, θα γίνουν οι απαραίτητες διορθώσεις στο διδακτικό υλικό, βάσει των αποτελεσμάτων της πιλοτικής εφαρμογής και θα πραγματοποιηθούν πειραματικές διδασκαλίες σε φοιτητές με χρήση του περιβάλλοντος objectKarel και στη συνέχεια του περιβάλλοντος BlueJ για τη διδασκαλία του αντικειμενοστραφούς προγραμματισμού. Στόχος της έρευνας αυτής είναι να διαπιστωθεί κατά πόσον τα εκπαιδευτικά προγραμματιστικά περιβάλλοντα προσφέρουν βοήθεια στη διδασκαλία του αντικειμενοστραφούς προγραμματισμού και επιπλέον κατά πόσον μια εισαγωγή στον

προγραμματισμό που χρησιμοποιεί κατ' αρχήν προγραμματιστικούς μικρόκοσμους είναι πλέον αποτελεσματικότερη από εκείνες που χρησιμοποιούν επαγγελματικά προγραμματιστικά περιβάλλοντα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Bergin J., Stehlik, M., Roberts, J., & Pattis, R. (1997), Karel++. A gentle introduction to the art of object-oriented programming, New York: John Wiley & Sons (2nd edition)

Carter J. & Fowler A. (1998), Object oriented students? *SIGCSE Bulletin*, 30(3), 271

Hadjerrouit S. (1999), A constructivist approach to object-oriented design and programming, *ACM SIGCSE Bulletin*, 31(3), 171-174

Holland S., Griffiths R., & Woodman M. (1997), Avoiding object misconceptions, *ACM SIGCSE Bulletin*, 29(1), 131-134

Kolling M., Quig B., Patterson A. & Rosenberg, J. (2003), The BlueJ system and its pedagogy, *Journal of Computer Science Education,* 13(4), 249-268

Spohrer J. C. & Soloway E. (1986), Novice mistakes: are the folk wisdoms correct?, *Communications of the ACM*, 29(7), 624-632

Ξυνόγαλος Σ. (2002), *Εκπαιδευτική Τεχνολογία: Ένας διδακτικός μικρόκοσμος για την εισαγωγή στον αντικειμενοστραφή προγραμματισμό*, Διδακτορική διατριβή, Τμήμα Εφ. Πληροφορικής, Πανεπιστήμιο Μακεδονίας