Η εύρεση μέγιστου στοιχείου ως ολοκληρωμένο παράδειγμα εφαρμογής υπολογιστικής σκέψης

Παναγιώτης Γροντάς

Καλλιτεχνικό Γυμνάσιο Γέρακα με Λυκειακές Τάξεις, pgrontas@gmail.com

Περίληψη

Η εργασία αυτή παρουσιάζει μια πρόταση διδασκαλίας για την εύρεση της ελάχιστης ή μέγιστης τιμής από ένα σύνολο στοιχείων υπό το πρίσμα αρχών της υπολογιστικής σκέψης. Το συγκεκριμένο πρόβλημα είναι αρκετά απλό και οικείο στους μαθητές με αποτέλεσμα να μπορεί να αναλυθεί χωρίς να χρειάζονται περισσότερες γνώσεις και αναπαραστάσεις που η αφομοίωση τους να απομακρύνει από τον στόχο. Επιπλέον είναι ένα παράδειγμα στο οποίο μπορεί να γίνουν κατανοητές με βιωματικό τρόπο αρχές της υπολογιστικής σκέψης σε όλα τα στάδια της επίλυσής του: από την αρχική ανάλυση μέχρι την κωδικοποίηση.

Λέξεις κλειδιά: Υπολογισμός Ελάχιστης\Μέγιστης τιμής, υπολογιστική σκέψη, ανάλυση προβλήματος, αναδρομική σκέψη, αναγνώριση προτύπων

1. Εισαγωγή

Η υπολογιστική σκέψη είναι ένα σύνολο δεξιοτήτων και στάσεων χρήσιμο σε κάθε μαθητή (Wing, 2006). Δραστηριότητες υπολογιστικής σκέψης έχουν προταθεί για διάφορες ηλικίες (Ζερβουδάκη & Παπαδάκης, 2018) και μαθήματα (Κοταρίνου, Κουλέτση, Πλιάκου κ.ά., 2018). Όμως, το διδακτικό αντικείμενο της πληροφορικής είναι ο κύριος αντιπρόσωπός της στην εκπαίδευση, αφού η υπολογιστική σκέψη αποτελεί το κυριότερο εργαλείο της, ως ένα χαρακτηριστικό που αποκτούν όλοι οι επιστήμονες της πληροφορικής, είτε το έχουν κατακτήσει συνειδητά είτε ασυνείδητα. Με αυτό το πρίσμα, έννοιες της υπολογιστικής σκέψης υπάρχουν σε όλα τα προγράμματα σπουδών πληροφορικής της δευτεροβάθμιας εκπαίδευσης, έστω και αν δεν γίνεται ρητή αναφορά σε αυτήν.

Από την άλλη, η εύρεση του ελάχιστου ή του μέγιστου από ένα σύνολο στοιχείων είναι ένα βασικό αλγοριθμικό πρόβλημα. Εμπειρικά έχει διαπιστωθεί πως όταν τα στοιχεία είναι πολλά έχει επικρατήσει ένας επαναληπτικός αλγόριθμος, τον οποίο οι περισσότεροι μαθητές κατανοούν και χρησιμοποιούν χωρίς δυσκολίες. Περιέργως όμως όταν τα στοιχεία προς εξέταση είναι λιγότερα (3 ή 4), οι μαθητές δεν χρησιμοποιούν μια εξειδίκευση του, αλλά επινοούν έναν διαφορετικό ο οποίος έχει αρκετές παγίδες, όπως θα φανεί στη συνέχεια, και δεν γενικεύεται εύκολα.

ISBN: 978-960-578-058-6

Η εργασία αυτή επεμβαίνει και στα δύο παραπάνω ζητήματα: Δίνεται ένα ολοκληρωμένο παράδειγμα εφαρμογής της υπολογιστικής σκέψης χρησιμοποιώντας την εύρεση του μέγιστου από κάποια στοιχεία. Η προσέγγιση έχει δύο πλεονεκτήματα: Πρώτα από όλα κάνει απτή την διδασκαλία της υπολογιστικής σκέψης, καθώς δεν αναφέρεται στις αρχές της μόνο θεωρητικά, αλλά τις εφαρμόζει σε ένα απλό και προσιτό στους μαθητές παράδειγμα. Επιπλέον οδηγεί σε έναν ολοκληρωμένο (και ορθό σε όλες τις περιπτώσεις) τρόπο για την εύρεση του μέγιστου από ένα σύνολο στοιχείων. Τέλος, τονίζουμε ότι η παρέμβαση που προτείνεται μπορεί να εφαρμοστεί για την εύρεση τόσο του μέγιστου όσο και του ελάχιστου. Για απλότητα, όμως, θα αναφερόμαστε από εδώ και στο εξής στο μέγιστο στοιχείο.

2. Υπολογιστική σκέψη

Αρχικά θα παρουσιαστούν κάποιες βασικές αρχές της υπολογιστικής σκέψης. Φυσικά δεν είναι ο στόχος μια εξαντλητική περιγραφή, αλλά το να τεθούν κάποιες βάσεις που θα αξιοποιηθούν στις επόμενες ενότητες.

2.1 Βασικά στοιχεία

Ο όρος «υπολογιστική σκέψη» πρωτοχρησιμοποιήθηκε στο (Wing, 2006) ως «μέθοδος επίλυσης προβλημάτων, σχεδιασμού συστημάτων και κατανόησης της ανθρώπινης συμπεριφοράς που χρησιμοποιεί έννοιες από την επιστήμη των υπολογιστών».

Κάποιες από αυτές τις έννοιες είναι (Wing, 2006), (Νείρος & Ζάχαρης, 2018):

- *Διάσπαση και σύνθεση:* Για να αντιμετωπιστεί ένα σύνθετο πρόβλημα μπορεί να χωριστεί σε απλούστερα, να επιλυθεί κάθε ένα από αυτά και στη συνέχεια να συνδυαστούν οι λύσεις (το γνωστό «Διαίρει και Βασίλευε»). Η ευκολία επίλυσης των υποπροβλημάτων αυτών οφείλεται είτε στο ότι εξετάζουν μία μόνο πτυχή του αρχικού, είτε στο ότι ασχολούνται με το ίδιο πρόβλημα σε ολοένα και μικρότερο πλήθος δεδομένων, με αποτέλεσμα να διευκολύνεται η εύρεση της λύσης. Ο τρόπος σκέψης της τελευταίας περίπτωσης είναι *αναδρομικός* και σύμφωνα με το (Wing, 2006) αποτελεί τμήμα της υπολογιστικής σκέψης. Η αναδρομή μπορεί να χρησιμοποιηθεί ως τρόπος ανάλυσης ενός προβλήματος, αλλά να οδηγήσει σε επαναληπτικό αλγόριθμο, όπως για παράδειγμα στο κλασικό παράδειγμα της δυαδικής αναζήτησης.
- *Γενίκευση:* Επιτρέπει την μετάβαση από την λύση ενός συγκεκριμένου στιγμιότυπου ενός προβλήματος στην γενική λύση. Για παράδειγμα, μπορεί να γίνει η μετάβαση στον γενικό τύπο επίλυσης πρωτοβάθμιας εξίσωσης, αφού έχει πρώτα λυθεί μία συγκεκριμένη εξίσωση. Όπως γίνεται σαφές, η

ISBN: 978-960-578-058-6

γενίκευση οδηγεί στην εξέταση περιπτώσεων που δεν εμπίπτουν στα χαρακτηριστικά του αρχικού στιγμιότυπου.

- *Δημιουργία μοντέλων:* Όταν δίνεται ένα πρόβλημα προς επίλυση, απομονώνονται τα ουσιαστικά χαρακτηριστικά του, φτιάχνοντας έτσι μια ιδεατή αναπαράσταση, για την οποία είναι πιο εύκολο να βρεθεί η λύση.
- *Αφαίρεση*: Επιτρέπει την χρήση ενός μοντέλου για κάποιο (υπο)πρόβλημα ως μαύρο κουτί, χωρίς δηλαδή να είναι επακριβώς γνωστές, συγκεκριμένες λεπτομέρειες. Βέβαια, στο τέλος πρέπει να ελεγχθεί αν η λύση ανταποκρίνεται στο πραγματικό στιγμιότυπο που αντιμετωπίσαμε αρχικά, ώστε να μην εμφανιστεί στην παγίδα των "leaky abstractions" (Spolsky, 2002).
- *Αναγνώριση προτύπων*: Στοχεύει στην ανάπτυξη της ικανότητας εύρεσης μοτίβων σε σύνολα από δεδομένα, δηλαδή σχέσεων και συγκεκριμένων εξαρτήσεων μεταξύ μεμονωμένων στοιχείων. Τα μοτίβα αυτά μπορούν να χρησιμοποιηθούν και για τη δημιουργία μοντέλων αλλά και για τον έλεγχο ότι οι λύσεις που δόθηκαν στα προβλήματα είναι ορθές.

2.2 Ένταξη στη διδασκαλία – παραδείγματα

Τα παραπάνω εργαλεία, αλλά και ο ορισμός στο (Wing, 2006) κάνουν φανερή την χρησιμότητα της υπολογιστικής σκέψης σε όλα τα μαθήματα, αλλά και στην καθημερινή ζωή των μαθητών. Όμως δεν αναιρούν τη σημασία της στην ίδια τη διδασκαλία της πληροφορικής, όπου οι αρχές της μπορούν να χρησιμοποιηθούν ώστε να οδηγηθούμε σε ορθές και κομψές λύσεις (π.χ. μέσω αναδρομικής σκέψης) ή να ελεγχθούν λύσεις που προκύπτουν ως στιγμιαία έμπνευση ή από λάθος κατανόηση του προβλήματος, χρησιμοποιώντας το εργαλείο της αναγνώρισης προτύπων.

Παρ' όλα αυτά δεν υπάρχει αντίστοιχο ολοκληρωμένο παράδειγμα στα διδακτικά πακέτα που χρησιμοποιούνται στη δευτεροβάθμια εκπαίδευση. Στο Γυμνάσιο (Αράπογλου, Μαβόγλου, Οικονομάκος κ.ά., 2006), δίνεται ιδιαίτερη έμφαση στο εργαλείο της Διάσπασης και Σύνθεσης προβλήματος, όπου η οργάνωση μιας εκπαιδευτικής εκδρομής επιλύεται με ανάλυση σε απλούστερα υποπροβλήματα. Το συγκεκριμένο, ως πρόβλημα της καθημερινής ζωής είναι εύκολο προς κατανόηση και συμβατό με τις εμπειρίες των μαθητών. Ως τέτοιο όμως δεν μπορεί να χρησιμοποιηθεί αυτούσιο στην συνέχεια, ώστε οι μαθητές να δουν τα πλεονεκτήματα της προσέγγισης με βάση την υπολογιστική σκέψη κατά την ανάπτυξη ενός αλγόριθμου.

Παρόμοια προσέγγιση ακολουθείται και στο μάθημα «Ανάπτυξη Εφαρμογών Σε Προγραμματιστικό Περιβάλλον» (Βακάλη, Γιαννόπουλος, Ιωαννίδης, κ.ά., 1999). Ιδιαίτερη έμφαση δίνεται και εδώ στην δομή προβλήματος. Χρησιμοποιείται ένα παράδειγμα από την καθημερινή ζωή – το πρόβλημα των ναρκωτικών, για να τονιστεί το κέρδος από την ανάλυση ενός προβλήματος σε απλούστερα. Το

ISBN: 978-960-578-058-6

ενδιαφέρον όμως είναι ότι στο συγκεκριμένο βιβλίο υπάρχει και ένα δεύτερο παράδειγμα που εμπίπτει στον χώρο της πληροφορικής, αυτό της επεξεργασίας των αποτελεσμάτων των πανελλαδικών εξετάσεων. Σε αυτό θα μπορούσαν να εφαρμοστούν τεχνικές της υπολογιστικής σκέψης, ώστε οι μαθητές να δουν πώς σχετίζονται η φάση της ανάλυσης με τον κώδικα που παράγεται. Κάτι τέτοιο όμως δεν γίνεται, ίσως λόγω του εύρους του προβλήματος αλλά και επειδή κάποιες από τις λειτουργίες που προτείνονται δεν είναι εφικτές στη ΓΛΩΣΣΑ (π.χ. δημιουργία γραφημάτων). Τέλος, τα ίδια ισχύουν και στο μάθημα «Εισαγωγή στις Αρχές της Επιστήμης των Η/Υ» (Δουκάκης, Δουληγέρης, Καρβουνίδης κ.ά., 2014). Η παρουσίαση κάποιων αρχών της υπολογιστικής σκέψης υπάρχει στα εισαγωγικά κεφάλαια, χωρίς όμως αυτές οι αρχές να εφαρμόζονται με κάποιο ολοκληρωμένο παράδειγμα στη συνέχεια.

Το κενό που περιεγράφηκε παραπάνω έρχεται να καλυφθεί από αρκετές παρεμβάσεις που έχουν σκοπό να αναδείξουν στην πράξη τις αρχές της υπολογιστικής σκέψης – ενδεικτικά (Κοτίνη & Τζελέπη, 2016), (Νείρος & Ζάχαρης, 2018) κ.ά.. Αν και αυτές βασίζονται σε εφαρμογές που έχουν στόχο να προσελκύσουν το ενδιαφέρον των μαθητών, πολλές φορές αφορούν τομείς με τους οποίους δεν είναι εξοικειωμένοι οι τελευταίοι με αποτέλεσμα να πρέπει να γίνουν επιπλέον βήματα προτού μπουν στην ουσία της υπολογιστικής σκέψης. Πολλές φορές ακόμα κινούνται στο επίπεδο των γρίφων, χωρίς να υπάρχει καμία καθοδήγηση στο πώς οι μαθητές θα οδηγηθούν οργανωμένα προς τη λύση. Τέλος δεν αφορούν κομμάτια της διδακτέας ύλης, ώστε να μπορούν να χρησιμοποιηθούν σε επόμενες ενότητες-τάξεις.

3. Εύρεση μέγιστου

Με βάση τα παραπάνω, θα εφαρμοστούν οι αρχές της υπολογιστικής σκέψης ώστε να μπορέσουν οι μαθητές να οδηγηθούν στην καλύτερη λύση σε ένα απλό πρόβλημα, το οποίο συναντάται πολύ συχνά. Με τον όρο καλύτερη λύση, εννοείται εκείνη η οποία είναι ορθή σε όλες τις περιπτώσεις και γενικεύεται ευκολότερα. Σε όλη την εργασία χρησιμοποιούμε για απλότητα την *ψευδογλώσσα* του (Βακάλη, Γιαννόπουλος, Ιωαννίδης κ.ά., 1999). Είναι αυτονόητο ότι η ίδια προσέγγιση μπορεί να προσαρμοστεί σε όποιο προγραμματιστικό περιβάλλον επιλέξουν οι διδάσκοντες.

3.1 Το πρόβλημα και η επιθυμητή λύση

Η βασική μορφή του προβλήματος εύρεσης μέγιστης τιμής είναι ότι δίνεται ένα σύνολο από συγκρίσιμα στοιχεία και ζητείται από τους μαθητές η δημιουργία ενός αλγόριθμου για την εύρεση της μεγαλύτερης από αυτές. Στο πρόβλημα αυτό υπάρχουν αρκετές παραλλαγές ανάλογα με το αν είναι γνωστό εκ των προτέρων το πλήθος των τιμών ή και οι ίδιες οι τιμές ή αν εισάγονται σταδιακά και για την εύρεση του πλήθους τους χρησιμοποιείται κάποια ειδική τιμή (τιμή – φρουρός σύμφωνα με την ‘φροντιστηριακή’ ορολογία του (Βακάλη, Γιαννόπουλος, Ιωαννίδης, κ.ά., 1999).

ISBN: 978-960-578-058-6

Για τον καλύτερο συνδυασμό της προσέγγισης μας με τις αρχές της υπολογιστικής σκέψης θα επικεντρωθούμε στην παραλλαγή όπου διαβάζονται σταδιακά τρεις αριθμούς και πρέπει να βρεθεί ο μεγαλύτερος.

Ο αλγόριθμος που πρέπει να βρουν οι μαθητές είναι ο παρακάτω:

```
1:  Διάβασε α
2:  μέγιστο ← α
3:  Διάβασε β
4:  Αν β > μέγιστο τότε
5: μέγιστο ← β
6:  Τέλος_Αν
7:  Διάβασε γ
8:  Αν γ > μέγιστο τότε
10:     μέγιστο ← γ
11: Τέλος_Αν
```

***Αλγόριθμος 1**. Εύρεση μέγιστου 3 αριθμών (επιθυμητός τρόπος)*

Ο παραπάνω αλγόριθμος έχει κάποια σημαντικά πλεονεκτήματα, όπως:

- Δεν έχει πρόβλημα κάποιες από τις τιμές είναι ίσες, καθώς σε αυτή την περίπτωση οι εντολές επιλογής μπορεί να μην εκτελεστούν, αλλά το αποτέλεσμα δεν θα αλλάξει.
- Επεκτείνεται εύκολα για περισσότερες από 3 τιμές, προσθέτοντας απλά μία εντολή εισόδου και μία σύγκριση:

```
Διάβασε δ
Αν δ > μέγιστο τότε
     μέγιστο ← δ
Τέλος_Αν
```

***Αλγόριθμος 2.** Επέκταση για τέταρτο αριθμό*

- Γενικεύεται εύκολα για περισσότερα στοιχεία με δομή επανάληψης:

```
Διάβασε α
μέγιστο ← α
Για ι από 2 μέχρι Ν
     Διάβασε α
     Αν α > μέγιστο τότε
          μέγιστο ← α
     Τέλος_αν
Τέλος_επανάληψης
```

***Αλγόριθμος 3.** Επέκταση για πολλά στοιχεία συγκεκριμένου πλήθους*

ISBN: 978-960-578-058-6

3.2 Προσεγγίσεις μαθητών

Το πρόβλημα το οποίο αποτέλεσε αφορμή για την παρούσα εργασία είναι η εμπειρική παρατήρηση, ότι όταν ζητηθεί από τους μαθητές να επιλύσουν το πρόβλημα της εύρεσης μέγιστου 3 αριθμών σπάνια δίνουν τις παραπάνω λύσεις. Για παράδειγμα, συναντούμε συχνά τον **Αλγόριθμος 4.** Τυπικός αλλά λανθασμένος τρόπος**Αλγόριθμος 4**:

```
Διάβασε α,β,γ
Αν α > β και α > γ τότε
     μέγιστο ← α
Αλλιώς_αν β > α και β > γ τότε
     μέγιστο ← β
Αλλιώς
     μέγιστο ← γ
Τέλος_Αν
```

Αλγόριθμος 4. *Τυπικός αλλά λανθασμένος τρόπος*

Επίσης είναι σύνηθες το γεγονός πολλοί μαθητές να παρουσιάσουν κάποια παραλλαγή, χρησιμοποιώντας εμφωλευμένη δομή επιλογής. Σε αυτή την περίπτωση δεν καταφέρνουν συνήθως να δώσουν μία ολοκληρωμένη απάντηση. Επίσης είναι αρκετές οι περιπτώσεις όπου χρησιμοποιούνται ανεξάρτητες απλές επιλογές αντί μίας πολλαπλής. Οι λύσεις αυτές δεν είναι ορθές καθώς δεν μπορούν να χειριστούν επιτυχώς την περίπτωση ισότητας κάποιων από τους τρεις αριθμούς: Για παράδειγμα, ενώ στην τριάδα (3,4,5) θα εμφανίσουν σωστά αποτελέσματα, στην (4,4,1) θα εμφανίσουν ως αποτέλεσμα το 1 ή δεν θα εμφανίσουν τίποτα. Φυσικά αυτό το λάθος μπορεί να διορθωθεί χρησιμοποιώντας τον τελεστή $\geq$ αντί για το $>$. Ακόμα όμως και σε αυτή την περίπτωση, παραμένει το πρόβλημα της μη εύκολης (εύλογης) γενίκευσης. Για παράδειγμα, ο αλγόριθμος για 4 αριθμούς περιλαμβάνει περισσότερες και πιο συνθέτες συνθήκες, ενώ δεν υπάρχει αντίστοιχος για μεταβλητό πλήθος αριθμών.

4. Διδακτική προσέγγιση

Η διδακτική παρέμβαση που προτείνεται στη συγκεκριμένη εργασία έχει διάρκεια 3 διδακτικών ωρών. Έχει εφαρμοστεί σε μαθητές της Γ Γυμνασίου και της Β Λυκείου του Καλλιτεχνικού Γυμνασίου Γέρακα με Λυκειακές τάξεις τα έτη 2016-2018. Στην περίπτωση της Γ Γυμνασίου απαιτήθηκαν 3 διδακτικές ώρες, ενώ στην Β Λυκείου, επειδή κάποιες έννοιες ήταν πιο οικείες η προσέγγιση συντομεύτηκε κατά μία ώρα.

Για την εφαρμογή της, πρέπει, στα πλαίσια μιας εισαγωγής στο μάθημα, να έχουν αναφερθεί κάποιες βασικές αρχές της υπολογιστικής σκέψης, όπως αυτές που περιγράψαμε στην ενότητα 2.1. Επίσης θεωρείται δεδομένο πώς οι μαθητές γνωρίζουν τη δομή επιλογής και τους λογικούς τελεστές, όχι σε βάθος κατ' ανάγκη.

ISBN: 978-960-578-058-6

Για να γίνει γενίκευση για περισσότερα στοιχεία θα πρέπει να είναι γνωστή και η δομή επανάληψης. Η παρέμβαση υποθέτει ότι το μάθημα διεξάγεται στο εργαστήριο πληροφορικής και οι μαθητές παρακολουθούν στον πίνακα χρησιμοποιώντας ταυτόχρονα τους υπολογιστές τους (σε ζευγάρια).

4.1 Εισαγωγή και απλούστερη περίπτωση

Αρχικά γίνεται παρουσίαση του προβλήματος στους μαθητές, χρησιμοποιώντας αναλογίες - παραδείγματα με τα οποία είναι πιθανόν να είναι εξοικειωμένοι (Αδαμόπουλος, 2005). Για παράδειγμα θα μπορούσαν να αναφερθούν περιπτώσεις από αθλητικές δραστηριότητες ή παιχνίδια στον υπολογιστή, όπου κερδίζει εκείνος που φέρει την μέγιστη τιμή σε κάποιο μέγεθος, όπως μέτρα ή σκορ.

Στη συνέχεια αφήνονται οι μαθητές για ελεγχόμενο χρονικό διάστημα να εκφράσουν τα βήματα που λύνουν το πρόβλημα, ώστε να πειστούν πώς η εύρεση του μέγιστου δεν είναι κάτι προφανές. Επειδή γνωρίζουν πώς να το λύσουν για τους εαυτούς τους, επισημαίνεται ότι πρέπει να προσπαθήσουν να εκφράσουν τις εσωτερικές τους σκέψεις με σαφώς καθορισμένο τρόπο, ώστε να προκύψει ο αλγόριθμος. Αφού περάσει ο χρόνος αυτός, τους υπενθυμίζονται τα εργαλεία της διάσπασης του προβλήματος σε απλούστερα ή μικρότερα. Έτσι γίνεται πρόκληση να αναζητήσουν ένα πιο εύκολο, αλλά σχετικό (υπό)πρόβλημα.

Εμπειρικά προκύπτει πως οι μαθητές θα απαντήσουν πως η εύρεση του μέγιστου δύο αριθμών είναι πιο εύκολη. Οι διδάσκοντες θα επανέρθουν ρωτώντας αν υπάρχει κάτι πιο εύκολο. Εμπειρικά, έχει διαπιστωθεί πως οι μαθητές διστακτικά απαντούν - ρωτώντας στην ουσία – πως ακόμα ευκολότερη είναι η εύρεση του μέγιστου ενός αριθμού. Κάποιοι μαθητές μπορεί να αναρωτηθούν τι νόημα έχει αυτό. Η απορία απαντάται με ένα παράδειγμα – σε κάποιον αγώνα που έχουν εγκαταλείψει όλοι εκτός από έναν πρέπει να ανακηρυχθεί νικητής. Αφού πειστούν όλοι οι μαθητές, η νέα γνώση κατοχυρώνεται γράφοντας στον πίνακα τις γραμμές 1, 2 από τον **Αλγόριθμος 1**.

Στη συνέχεια, για προετοιμασία των επόμενων σταδίων, επιβεβαιώνεται ότι οι μαθητές γνωρίζουν ότι η λύση βρίσκεται στη μεταβλητή `μέγιστο`. Ακολούθως καλούνται να επιλύσουν το αμέσως πιο δύσκολο πρόβλημα – δηλαδή την εύρεση του μέγιστου από δύο αριθμούς. Γίνεται σαφές ότι πρέπει να χρησιμοποιηθεί ό,τι έχει γίνει μέχρι τώρα.

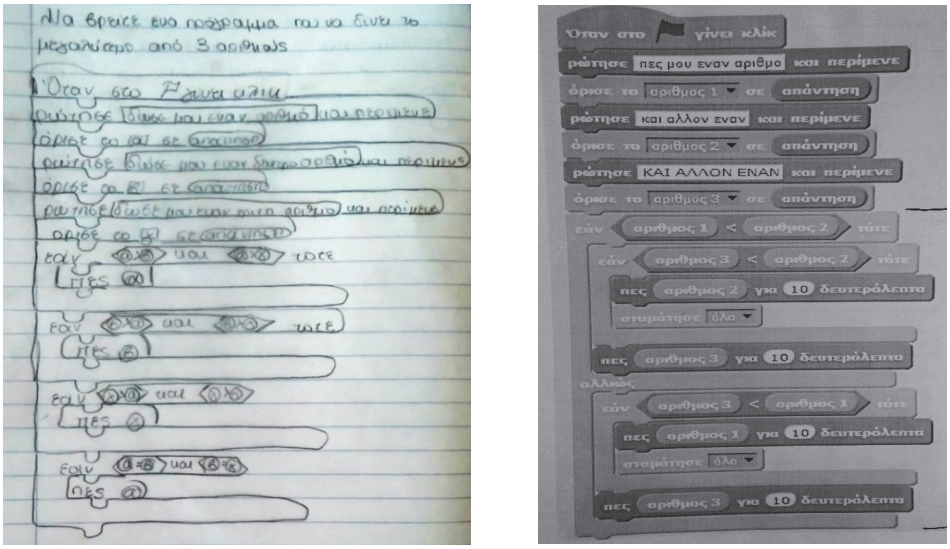
Στο στάδιο αυτό παρουσιάζεται συνήθως το φαινόμενο οι μαθητές να ξεχνούν τα μέχρι τώρα διδαχθέντα και να ξανασκέφτονται το πρόβλημα από την αρχή. Οι διδάσκοντες πρέπει να περιορίσουν κάτι τέτοιο τονίζοντας του ότι πρέπει μέσα στον αλγόριθμο να υπάρχει οπωσδήποτε η χρήση της μεταβλητής `μέγιστο`. Μετά από κάποια συζήτηση η τρέχουσα λύση αποκτά τις γραμμές 3-6 από τον **Αλγόριθμος 1**.

ISBN: 978-960-578-058-6

Σε αυτό το σημείο, οι διδάσκοντες πρέπει να σιγουρευτούν ότι έχει γίνει κατανοητό ότι και στις τρεις πιθανές περιπτώσεις ($\alpha < \beta$, $\alpha = \beta$, $\alpha > \beta$) οι εντολές που έχουν χρησιμοποιηθεί λύνουν το πρόβλημα σωστά. Αυτό μπορεί να γίνει με συζήτηση στον πίνακα αλλά και εκτέλεση των εντολών στο προγραμματιστικό περιβάλλον που έχει επιλεγεί. Η πρώτη διδακτική ώρα κλείνει, αναθέτοντας τον αλγόριθμο για την εύρεση του μέγιστου από τρεις αριθμούς ως άσκηση για το σπίτι, επισημαίνοντας και πάλι ότι πρέπει να χρησιμοποιηθεί η μέχρι τώρα λύση.

4.2 Έλεγχος λαθών με αναγνώριση προτύπων

Η δεύτερη διδακτική ώρα ξεκινά ζητώντας από τους μαθητές να φορτώσουν στο προγραμματιστικό τους περιβάλλον τις λύσεις τους, οι οποίες και ελέγχονται. Η εμπειρία δείχνει ότι οι περισσότεροι μαθητές δεν χρησιμοποιούν ό,τι έχει διδαχθεί στην 1η διδακτική ώρα και ουσιαστικά ξανασκέφτονται το πρόβλημα από την αρχή, δίνοντας ως λύση κάποια παραλλαγή από αυτές που φαίνονται στον **Αλγόριθμος 4**. Το συγκεκριμένο στάδιο είναι το πιο δύσκολο για τους διδάσκοντες, καθώς οι μαθητές έχουν την τάση να βρίσκουν αρκετά περίεργες λύσεις οι οποίες χρειάζονται χρόνο να ελεγχθούν, όπως φαίνεται στην **Εικόνα 1**:



Εικόνα 1. *Λύσεις μαθητών Γυμνασίου*

Εδώ εφαρμόζεται το εργαλείο της αναγνώρισης προτύπων το οποίο συμβαδίζει με αρχές από την ανακαλυπτική μάθηση (Shunk, 2010) και ζητείται από τους μαθητές να προσπαθήσουν να ελέγξουν την ορθότητα της λύσης τους, επισημαίνοντας ότι πρέπει να ψάξουν για ασυνήθιστες περιπτώσεις. Μετά από επαρκή χρόνο αναζήτησης και αν κανένα ζευγάρι δεν την έχει βρει μια περίπτωση, προκρίνεται η εξέταση της περίπτωσης της ισότητας δύο από τους τρεις αριθμούς. Στο ενδεχόμενο μιας

ISBN: 978-960-578-058-6

περίεργης, αλλά ορθής λύσης, αφού επαινεθούν οι μαθητές, τίθεται ο προβληματισμός της ευκολίας επέκτασης της προσέγγισής τους σε 4 και παραπάνω αριθμούς.

Στη συνέχεια, γίνεται ανάκληση της λύσης για την περίπτωση των δύο αριθμών που περιεγράφηκε την πρώτη διδακτική ώρα και προσπάθεια από τους μαθητές για επέκτασή της. Για να γίνει αυτό επισημαίνεται ότι ο τρόπος για να φθάσει ο μαθητής από την δεύτερη στην τρίτη τιμή (**Αλγόριθμος 1**, γραμμές 7-11) είναι παρόμοιος με τον τρόπο που έφθασε από την πρώτη στη δεύτερη (**Αλγόριθμος 1**, γραμμές 3-6).

Αφού οι μαθητές συντάξουν τις εντολές, γίνεται προτροπή να τις δοκιμάσουν πάλι, ελέγχοντας και τις τιμές που παρουσίασαν πρόβλημα νωρίτερα. Τέλος για να ελεγχθεί η κατανόηση, ζητείται ως μια γρήγορη άσκηση να τροποποιήσουν το πρόγραμμά τους για να βρίσκει το μέγιστο από τέσσερις αριθμούς.

4.3 Γενίκευση

Ανάλογα με το επίπεδο, τις γνώσεις και την ηλικία των μαθητών, οι διδάσκοντες μπορούν να τους κινήσουν το ενδιαφέρον ώστε να προσπαθήσουν να βρουν τον μέγιστο από περισσότερους αριθμούς. Έτσι μπορούν να επιλέξουν για πλήθος αρχικά μία τιμή που κάνει δύσκολη την αντιγραφή των εντολών, αλλά εύκολη την εισαγωγή των τιμών π.χ. το 10. Στο τέλος της δεύτερης διδακτικής ώρας ζητείται, ως άσκηση για το επόμενο μάθημα, να τροποποιηθεί ο **Αλγόριθμος 2**, χρησιμοποιώντας δομή επανάληψης. Αυτό θα αναγκάσει τους μαθητές να αναλογιστούν το πλήθος των μεταβλητών που χρειάζεται να χρησιμοποιήσουν. Ως πιθανή βοήθεια, μπορεί να προταθεί η ερώτηση πόσες μεταβλητές χρειάζονται σε κάθε σύγκριση που γίνεται.

Την τρίτη διδακτική ώρα γίνεται συζήτηση σχετικά με τις λύσεις των μαθητών. Σε προχωρημένα τμήματα γυμνασίου μπορεί να φθάσει κανείς μέχρι αυτό το σημείο. Σε τμήματα Λυκείου πρέπει να εξεταστούν και παραλλαγές με άγνωστο πλήθος αριθμών, να γίνει κατάλληλη συζήτηση για διαφορετικούς τρόπους αρχικοποίησης των μεταβλητών και τις συνέπειές τους, αλλά και για χρήση άλλων εντολών επανάληψης.

5. Συμπεράσματα

Στην εργασία αυτή χρησιμοποιήθηκαν αρχές της υπολογιστικής σκέψης στα πλαίσια διδασκαλίας ενός τυπικού αλγόριθμου που συναντάται στον προγραμματισμό και όχι για κάποιο άλλο αντικείμενο εκτός της πληροφορικής. Η μεθοδολογία αυτή επέτρεψε να αξιοποιηθεί ένα κομμάτι της ύλης, που συνήθως γίνεται αντιληπτό μόνο σε θεωρητικό επίπεδο, ώστε να κατασκευάσουν οι μαθητές κώδικα που λύνει κάποιο υπαρκτό πρόβλημα. Η αξία της συγκεκριμένης προσέγγισης φαίνεται όταν συγκρίνονται τα αποτελέσματά της με άλλες εναλλακτικές λύσεις που υπάρχουν για το ίδιο πρόβλημα. Η σύγκριση αυτή ενσωματώθηκε στην προσέγγισή της εργασίας

ISBN: 978-960-578-058-6

ώστε να γίνουν τα πλεονεκτήματά της σαφή και στους ίδιους τους μαθητές με βιωματικό τρόπο χωρίς να τους τα ‘επιβάλλει’ κάποιος θεωρητικά.

Αναφορές

Shunk, H. D. (2010). Θεωρίες Μάθησης. Μεταίχμιο.

Spolsky, J. (2002). The Law of Leaky Abstractions. Ανάκτηση από Joel on Software: www.joelonsoftware.com/2002/11/11/the-law-of-leaky-abstractions/

Wing, J. M. (2006, March). Computational Thinking. Communications Of The ACM, 49(2), 33-35.

Αδαμόπουλος, Ν. (2005). Χρήση Αναλογιών και Μεταφορών στη Διδασκαλία του Μαθήματος ‘Μετάδοση Δεδομένων & Δίκτυα Υπολογιστών’: Μια μελέτη Περίπτωσης. 3ο Πανελλήνιο Συνέδριο "Διδακτική της Πληροφορικής". Κόρινθος.

Αράπογλου, Α., Μαβόγλου, Χ., Οικονομάκος, Η., & Φύτρος, Κ. (2006). Πληροφορική Α’, Β’, Γ’ Γυμνασίου. Αθήνα: Ο.Ε.Δ.Β.

Βακάλη, Α., Γιαννόπουλος, Η., Ιωαννίδης, Ν., Κοίλιας, Χ., Μάλαμας, Κ., Μανωλόπουλος, Ι., & Πολίτης, Π. (1999). Ανάπτυξη Εφαρμογών σε Προγραμματιστικό Περιβάλλον. Αθήνα: ΟΕΔΒ - ΙΕΠ.

Δουκάκης, Σ., Δουληγέρης, Χ., Καρβουνίδης, Θ., Κοίλιας, Χ., & Πέρδος, Α. (2014). Εισαγωγή στις Αρχές της Επιστήμης των Η/Υ. Αθήνα: Ινστιτούτο Τεχνολογίας Υπολογιστών και Εκδόσεων «ΔΙΟΦΑΝΤΟΣ».

Ζερβουδάκη , Ε., & Παπαδάκης, Σ. (2019). Χρήση Εκπαιδευτικής Ρομποτικής και Ανάπτυξη Υπολογιστικής Σκέψης στην Προσχολική και Πρωτοσχολική Ηλικία. 10th Conference on Informatics in Education 2018 (σσ. 389-398). Εκδόσεις Νέων Τεχνολογιών.

Κοταρίνου, Π., Κουλέτση, Ε., Πλιάκου, Μ., Συριόπουλος, Σ., & Χούπη, Μ. (2018). Hobbits και Orcs: Διασχίζοντας έναν ποταμό με τους ήρωες του Tolkien. 10th Conference on Informatics in Education 2018 (σσ. 262-277). Εκδόσεις Νέων Τεχνολογιών.

ISBN: 978-960-578-058-6

Κοτίνη, Ι., & Τζελέπη, Σ. (2016). Η μεθοδολογία Agile και η εφαρμογή της στην μαθησιακή διαδικασία ενισχύουν την Υπολογιστική Σκέψη. 8th Conference on Informatics in Education (σσ. 212-222). ΕΠΥ.

Νείρος, Α., & Ζάχαρης, Κ. (2018). Δραστηριότητες Υπολογιστικής Σκέψης. 10th Conference on Informatics in Education 2018 (σσ. 399-409). Εκδόσεις Νέων Τεχνολογιών.

Abstract

This paper presents a teaching proposal for the computation of the maximum value of an element in a set using computational thinking concepts. The problem selected is simple enough so that it can be understood without the need for intricate examples that might divert students' attention. Furthermore, it can illustrate, in a straightforward way, important computational thinking principles, applicable in all phases ranging from problem analysis to coding.

Λέξεις κλειδιά: Minimum\Maximum value computation, computational thinking, problem analysis, recursive thinking, pattern matching

ISBN: 978-960-578-058-6