Πολυτεχνείο Κρήτης

Τμήμα Ηλεκτρονικών Μηχανικών και Μηχανικών Υπολογιστών

Εργαστήριο Διανεμομένων Πληροφοριακών Συστημάτων και Εφαρμογών Πολυμέσων (MUSIC)

Διπλωματική Εργασία: Ανάπτυξη γραφικού κέλυφους της γλώσσας προγραμματισμού C για την υποστήριξη μαθημάτων εκμάθησης προγραμματισμού.

Κυφωνίδης Χαράλαμπος

Επιβλέπων καθηγητής:

Δρ. Χριστοδουλάκης Σταύρος

Η παρούσα διπλωματική εργασία αφιερώνεται στους γονείς μου οι οποίοι με στήριξαν καθ’ όλη την διάρκεια της φοίτησης μου αλλά και γενικότερα της ζωής μου.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τους

-Καθηγητή Σταύρο Χριστοδουλάκη

-Καθηγητή Λαγουδάκη Μιχαήλ

-Καθηγήτρια Μανιά Αικατερίνη

-Νεκτάριο Μουμουτζή

-Καθηγητή Δουλάμη Αναστάσιο

-Γεωργία Κυριακάκη

-Πρωτοπαπαδάκη Ευτύχη

-Ορέστη Ξενή

-Ασλανίδη Χρήστο

-Καρεκλά Γεώργιο

-Φελλά Κωνσταντίνο

-Ψιλάκη Γεώργιο

-Ρουσόπουλο Χρήστο

-Μελιόπουλο Αριστείδη

-Κουκουλή Ανδρέα

Πίνακας Περιεχομένων

Κεφάλαιο Σελίδα

Ευχαριστίες **3**

Πίνακας Περιεχομένων **4**

Κατάλογος Πινάκων, Διαγραμμάτων και Εικόνων **6**

Περίληψη **10**

Κεφάλαιο 1 Εισαγωγή και στόχος παρούσας διπλωματικής **11**
Περίληψη 11
Εισαγωγή 11
Ερώτημα που απασχολεί την παρούσα εργασία 12
Αναγκαιότητα της παρούσας διπλωματικής εργασίας 13
Γιατί το εργαλείο πετυχαίνει τον στόχο του 13
Δομή της παρόντος κειμένου 13

Κεφάλαιο 2 *Σχετικές εργασίες και εργαλεία που χρησιμοποιήθηκαν* **14**
Περίληψη 14
Θεωρία που διέπει την υποβοηθούμενη μάθηση 14
Θεωρία Δραστηριοτήτων (Activity Theory) *14*
Αλληλεπίδραση Ανθρώπου-Υπολογιστή(HCI) και Θεωρία Δραστηριοτήτων(Activity Theory) *15*
Υποκείμενο-Αντικείμενο-Εργαλείο *15*
Ζώνη Επικείμενης Ανάπτυξης (Zone of Proximal Development) *16*
Αφομοίωση και Εξωτερίκευση (Internalization & Externalization) *18*

Εκμάθηση προγραμματισμού σε αρχάριους **18**
Ιστορική αναδρομή 18
Σύγχρονα εργαλεία και γλώσσες προγραμματισμού για αρχάριους 19
Προγραμματισμός με γραφικά τουβλάκια (blocks) 23
Openblocks 23

Κεφάλαιο 3 *Σχεδιασμός, Υλοποίηση και Περιγραφή του Εργαλείου* **23**
Περίληψη 23
Σχεδιαστικές Αποφάσεις 24
Βασική περιγραφή του Block-C 24
Ομαδοποίηση των Block 28
Αρχείο lan_def.dtd που ορίζει τους κανόνες που ακολουθεί το XML αρχείο περιγραφής της γλώσσας 29
Επέκταση του Openblocks 34
Αντιγραφή blocks 36
Zebra Coding 38
Ήδη υπάρχουσα λειτουργικότητα που αφορά την διαχείριση του project του χρήστη 39
Εξαγωγή Κώδικα σε C 40
Κατηγορία «User Defined» 49

Κεφάλαιο 4 *Διαδικασία Αξιολόγησης της γραφικής διεπαφής του Εργαλείου* **50**
Περίληψη 50
Εισαγωγή 50

Διαδικασία 50
Αποτελέσματα 55
Συμπεράσματα 60

Κεφάλαιο 5 Διαδικασία Αξιολόγησης ευχρηστίας της γραφικής διεπαφής και της πρόληψης συντακτικών λαθών. 60
Περίληψη 60
Εισαγωγή 60
Διαδικασία 61
Αποτελέσματα 62
Συμπεράσματα 65

Κεφάλαιο 6 *Διαδικασία Αξιολόγησης της αποτελεσματικότητας του εργαλείου* .. 65
Περίληψη 65
Εισαγωγή 65
Διαδικασία 66
1η ενότητα – εισαγωγή στο πείραμα 66
2η ενότητα – εισαγωγή στο πείραμα 69

Κεφάλαιο 7 Αποτελέσματα 3ής αξιολόγησης 83
Περίληψη 83
Εισαγωγή 83
Αποτελέσματα πρώτης «εισαγωγικής» ενότητας 84
Συμπεράσματα 94
Αποτελέσματα δεύτερης ενότητας 95
Αποτελέσματα ερωτηματολογίων 98

Κεφάλαιο 8 Συμπεράσματα και μελλοντική εργασία 103
Περίληψη 103
Εισαγωγή 103
Σύγκριση 2 μεθόδων 103
Αποτελέσματα 104
Ερωτηματολόγια 105
Συμπεράσματα 107
Μελλοντική δουλειά 107

References / Βιβλιογραφία 109

Κατάλογος Πινάκων, Διαγραμμάτων και Εικόνων

ΔΙΑΓΡΑΜΜΑ 2.1: Υ = ΥΠΟΚΕΙΜΕΝΟ , Ε = ΕΡΓΑΛΕΙΟ , Α = ΑΝΤΙΚΕΙΜΕΝΟ. ΕΝΑ ΥΠΟΚΕΙΜΕΝΟ ΔΡΑ ΜΕΣΩ ΜΙΑΣ ΕΝΕΡΓΕΙΑΣ ΓΙΑ ΝΑ ΚΑΤΑΛΗΞΕΙ ΣΤΟ ΣΤΟΧΟ (ΑΝΤΙΚΕΙΜΕΝΟ). ΣΤΗΝ ΕΝΕΡΓΕΙΑ ΑΥΤΗ ΜΕΣΟΛΑΒΕΙ ΤΟ ΕΡΓΑΛΕΙΟ ΤΟ ΟΠΟΙΟ ΧΡΗΣΙΜΟΠΟΙΕΙΤΑΙ ΑΠΟ ΤΟΝ ΧΡΗΣΤΗ (ΥΠΟΚΕΙΜΕΝΟ). 16
ΔΙΑΓΡΑΜΜΑ 1.3: ΖΩΝΗ ΕΠΙΚΕΙΜΕΝΗΣ ΑΝΑΠΤΥΞΗΣ 17
ΕΙΚΟΝΑ 2.4: ΟΠΤΙΚΗ ΕΠΙΣΗΜΑΝΣΗ ΚΕΙΜΕΝΟΥ (TEXT HIGHLIGHTING): ΜΕΘΟΔΟΣ ΓΙΑ ΤΗΝ ΔΙΑΚΡΙΤΟΠΟΙΗΣΗ ΤΩΝ ΔΙΑΦΟΡΕΤΙΚΩΝ ΣΥΝΤΑΚΤΙΚΩΝ ΜΟΝΑΔΩΝ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ. 20
ΕΙΚΟΝΑ 2.5: ΑΥΤΟΜΑΤΗ ΣΥΜΠΛΗΡΩΣΗ ΚΩΔΙΚΑ (CODE COMPLETION): ΜΕΘΟΔΟΣ ΓΙΑ ΝΑ ΠΡΟΤΕΙΝΕΙ ΤΟ ΣΥΣΤΗΜΑ ΠΙΘΑΝΑ ΚΟΜΜΑΤΙΑ ΚΩΔΙΚΑ ΠΟΥ ΙΣΩΣ ΕΠΟΝΤΑΙ ΑΥΤΩΝ ΠΟΥ ΓΡΑΦΕΙ Ο ΧΡΗΣΤΗΣ. 20
ΕΙΚΟΝΑ 2.6: Η ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ STARLOGO TNG. ΑΝΑΠΤΥΧΘΗΚΕ ΩΣ Η ΕΞΕΛΙΞΗ ΤΗΣ ΓΛΩΣΣΑΣ LOGO. 21
ΕΙΚΟΝΑ 2.7: Η ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ E-TOYS 21
ΕΙΚΟΝΑ 2.8: ΤΟ ΕΡΓΑΛΕΙΑ ALICE, ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΕΙΤΑΙ ΓΙΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ ΓΡΑΦΙΚΩΝ ΣΕ JAVA 22
ΕΙΚΟΝΑ 2.9: Η ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ SCRATCH 22
ΕΙΚΟΝΑ 3.1: Η ΓΡΑΦΙΚΗ ΔΙΕΠΑΦΗ ΤΟΥ ΕΡΓΑΛΕΙΟΥ ΧΩΡΙΖΕΤΑΙ ΣΕ 3 ΔΙΑΚΡΙΤΕΣ ΠΕΡΙΟΧΕΣ: 1) Η ΠΕΡΙΟΧΗ ΜΕ ΤΑ BLOCKS ΟΜΑΔΟΠΟΙΗΜΕΝΑ ΣΕ ΚΑΤΗΓΟΡΙΕΣ. 2) Η ΠΕΡΙΟΧΗ ΜΕ ΤΑ ΚΟΥΜΠΙΑ ΛΕΙΤΟΥΡΓΙΚΟΤΗΤΑΣ. 3) Ο ΧΩΡΟΣ ΕΡΓΑΣΙΑΣ ΤΟΥ ΧΡΗΣΤΗ 25
ΕΙΚΟΝΑ 3.2: Η ΚΑΤΗΓΟΡΙΑ "VARIABLES" ΑΦΟΥ ΤΗΝ ΕΠΙΛΕΞΕΙ ΜΕ ΤΟ ΠΟΝΤΙΚΙ ΤΟΥ Ο ΧΡΗΣΤΗΣ 26
ΠΙΝΑΚΑΣ 3.3: ΠΙΝΑΚΑΣ ΜΕ ΤΗΝ ΓΡΑΦΙΚΗ ΑΝΑΠΑΡΑΣΤΑΣΗ Ω ΣΥΝΔΕΣΜΩΝ ΚΑΙ ΤΙΣ ΛΕΠΤΟΜΕΡΕΙΕΣ ΠΟΥ ΤΟΥΣ ΑΦΟΡΟΥΝ 26
ΕΙΚΟΝΑ 3.4: Η ΔΙΑΔΙΚΑΣΙΑ ΜΙΑ ΕΠΙΤΥΧΟΥΣ ΣΥΝΔΕΣΕΩΣ. ΑΡΙΣΤΕΡΑ ΕΙΝΑΙ ΤΟ BLOCK ”INT” ΤΟ ΟΠΟΙΟ ΜΕΤΑΦΕΡΕΤΑΙ ΑΠΟ ΤΟ ΠΟΝΤΙΚΙ ΠΡΙΝ ΑΠΕΛΕΥΘΕΡΩΘΕΙ ΚΑΙ ΔΕΞΙΑ ΑΦΟΥ ΑΠΕΛΕΥΘΕΡΩΘΕΙ ΤΟ ΑΡΙΣΤΕΡΟ ΚΛΙΚ ΤΟΥ ΠΟΝΤΙΚΙΟΥ. ΠΑΡΑΤΗΡΟΥΜΕ ΟΤΙ ΤΟ BLOCK ΤΗΣ MAIN, ΜΕΤΑ ΤΗΝ ΣΥΝΔΕΣΗ, ΔΙΕΥΡΥΝΘΗΚΕ ΚΑΙ ΔΗΜΙΟΥΡΓΗΘΗΚΕ ΑΚΟΜΑ ΜΙΑ ΘΕΣΗ ΓΙΑ ΤΟ “BODY” 27
ΕΙΚΟΝΑ 3.5: Η ΔΙΑΔΙΚΑΣΙΑ ΜΙΑ ΑΝΕΠΙΤΥΧΟΥΣ ΣΥΝΔΕΣΕΩΣ. ΑΡΙΣΤΕΡΑ ΕΙΝΑΙ ΤΟ BLOCK “VALUE” ΤΟ ΟΠΟΙΟ ΜΕΤΑΦΕΡΕΤΑΙ ΠΡΙΝ ΑΠΕΛΕΥΘΕΡΩΘΕΙ ΚΑΙ ΔΕΞΙΑ ΑΦΟΥ ΑΠΕΛΕΥΘΕΡΩΘΕΙ ΤΟ ΑΡΙΣΤΕΡΟ ΚΛΙΚ ΤΟΥ ΠΟΝΤΙΚΙΟΥ. ΠΑΡΑΤΗΡΟΥΜΕ ΟΤΙ ΤΟ BLOCK ΤΗΣ MAIN, ΜΕΤΑ ΤΗΝ ΑΠΟΤΥΧΙΑ ΣΥΝΔΕΣΗΣ, ΔΕΝ ΔΙΕΥΡΥΝΘΗΚΕ. 27
ΕΙΚΟΝΑ 3.6: Ο ΤΡΟΠΟΣ ΜΕ ΤΟΝ ΟΠΟΙΟ Ο ΧΡΗΣΤΗΣ ΔΙΑΓΡΑΦΕΙ ΕΝΑ BLOCK. ΠΑΡΑΤΗΡΟΥΜΕ ΟΤΙ Ο ΚΑΔΟΣ ΑΝΟΙΓΕΙ ΣΕ ΕΝΔΕΙΞΗ ΟΤΙ ΑΝΤΑΠΟΚΡΙΝΕΤΑΙ ΣΤΗΝ ΚΙΝΗΣΗ ΤΟΥ ΧΡΗΣΤΗ 28
ΕΙΚΟΝΑ 3.7: ΟΙ ΠΙΘΑΝΕΣ ΤΙΜΕΣ ΠΟΥ ΜΠΟΡΕΙ ΝΑ ΠΑΡΕΙ ΤΟ BLOCK "%D", ΟΠΩΣ ΑΥΤΕΣ ΕΜΦΑΝΙΖΟΝΤΑΙ ΣΤΗΝ DROPDOWN ΛΙΣΤΑ ΤΩΝ ΕΠΙΛΟΓΩΝ 28
ΕΙΚΟΝΑ 3.8 : ΟΙ ΚΑΤΗΓΟΡΙΕΣ ΤΩΝ BLOCKS 29
ΔΙΑΓΡΑΜΜΑ 3.9: Η ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΤΟΥ OPENBLOCKS ΒΑΣΗ ΤΩΝ ΠΑΚΕΤΩΝ ΠΟΥ ΠΕΡΙΕΧΟΥΝ ΑΡΧΕΙΑ JAVA ΚΑΙ Η ΜΕΤΑΞΥ ΤΟΥΣ ΣΧΕΣΕΙΣ. 36
ΕΙΚΟΝΑ 3.10: ΠΡΙΝ ΤΗΝ ΑΝΤΙΓΡΑΦΗ 37
ΕΙΚΟΝΑ 3.11: ΜΕΤΑ ΤΗΝ ΑΝΤΙΓΡΑΦΗ 38
ΕΙΚΟΝΑ 3.12: ΠΑΡΑΔΕΙΓΜΑ ΤΗΣ ΧΡΩΜΑΤΙΚΗΣ ΚΩΔΙΚΟΠΟΙΗΣΗΣ ZEBRA-CODING 39
ΕΙΚΟΝΑ 3.13: ΤΑ ΚΟΥΜΠΙΑ ΚΑΙ ΤΟ ΠΕΔΙΟ ΑΝΑΖΗΤΗΣΗΣ ΠΟΥ ΠΑΡΕΧΟΥΝ ΛΕΙΤΟΥΡΓΙΚΟΤΗΤΑ ΓΙΑ UNDO/REDO, ΑΝΑΖΗΤΗΣΗ, ΑΠΟΘΗΚΕΥΣΗ, ΑΝΟΙΓΜΑ ΚΑΙ ΕΞΑΓΩΓΗ ΚΩΔΙΚΑ. 39
ΕΙΚΟΝΑ 3.14: ΑΝΑΠΑΡΑΣΤΑΣΗ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ ΕΞΑΓΩΓΗΣ ΚΩΔΙΚΑ C ΜΕ ΤΗΝ ΜΟΡΦΗ FLOW CHART. 43
ΕΙΚΟΝΑ 3.15: ΕΝΑ ΑΠΛΟ ΠΑΡΑΔΕΙΓΜΑ ΠΡΟΓΡΑΜΜΑΤΟΣ ΕΚΤΥΠΩΣΗΣ ΤΩΝ ΑΡΙΘΜΩΝ ΑΠΟ ΤΟ 0 ΕΩΣ ΤΟ 9. ΣΕ ΑΥΤΟ ΤΟ ΠΑΡΑΔΕΙΓΜΑ ΧΡΗΣΙΜΟΠΟΙΟΥΝΤΑΙ ΜΙΑ ΔΟΜΗ ΕΠΑΝΑΛΗΨΗΣ (FOR LOOP) ΚΑΙ ΜΙΑ ΣΥΝΑΡΤΗΣΗ ΕΞΟΔΟΥ (PRINTF). 43

ΔΙΑΓΡΑΜΜΑ 3.16 : ΤΟ ΔΕΝΤΡΟ ΠΟΥ ΠΑΡΑΧΘΗΚΕ ΑΠΟ ΤΗΝ ΣΥΝΑΡΤΗΣΗ GETCODEOFINNERBLOCKS ΓΙΑ ΤΗΝ ΕΞΑΓΩΓΗ ΤΟΥ ΚΩΔΙΚΑ ΤΗΣ ΣΥΝΑΡΤΗΣΗΣ ΤΗΣ ΕΙΚΟΝΑΣ 3.13 44
ΕΙΚΟΝΑ 3.17: ΕΝΑ ΣΥΝΘΕΤΟ ΠΡΟΓΡΑΜΜΑ ΣΤΟ BLOCK-C 48
ΕΙΚΟΝΑ 3.18: Ο ΕΞΑΧΘΕΙΣ ΚΩΔΙΚΑΣ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ ΣΤΗΝ ΕΙΚΟΝΑ 3.15 48
ΕΙΚΟΝΑ 3.19: ΤΑ BLOCK ΠΟΥ ΔΗΜΙΟΥΡΓΗΣΕ ΤΟ ΕΡΓΑΛΕΙΟ ΜΕ ΒΑΣΕΙ ΤΙΣ ΜΕΤΑΒΛΗΤΕΣ ΚΑΙ ΤΗΝ ΣΥΝΑΡΤΗΣΗ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ ΤΗΣ ΕΙΚΟΝΑΣ 3.15 49
ΠΙΝΑΚΑΣ 4.1: ΠΙΝΑΚΑΣ ΜΕ ΜΙΑ ΣΥΝΤΟΜΗ ΚΑΤΑΓΡΑΦΗ ΤΩΝ ΠΡΟΦΙΛ ΤΩΝ ΧΡΗΣΤΩΝ ΤΗΣ ΠΡΩΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ 51
ΕΙΚΟΝΑ 4.2: ΟΙ ΕΡΓΑΣΙΕΣ ΠΟΥ ΚΛΗΘΗΚΑΝ ΝΑ ΥΛΟΠΟΙΗΣΟΥΝ ΟΙ ΧΡΗΣΤΕΣ ΣΤΗΝ ΠΡΩΤΗ ΑΞΙΟΛΟΓΗΣΗ 52
ΕΙΚΟΝΑ 4.3: ΠΡΩΤΗ ΣΕΛΙΔΑ ΕΡΩΤΗΜΑΤΟΛΟΓΙΟΥ ΠΟΥ ΚΛΗΘΗΚΑΝ ΝΑ ΣΥΜΠΛΗΡΩΣΟΥΝ ΟΙ ΧΡΗΣΤΕΣ ΜΕΤΑ ΤΟ ΠΕΡΑΣ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ ΑΞΙΟΛΟΓΗΣΗΣ 53
ΕΙΚΟΝΑ 4.4: ΔΕΥΤΕΡΗ ΣΕΛΙΔΑ ΕΡΩΤΗΜΑΤΟΛΟΓΙΟΥ ΠΟΥ ΚΛΗΘΗΚΑΝ ΝΑ ΣΥΜΠΛΗΡΩΣΟΥΝ ΟΙ ΧΡΗΣΤΕΣ ΜΕΤΑ ΤΟ ΠΕΡΑΣ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ ΑΞΙΟΛΟΓΗΣΗΣ 54
ΠΙΝΑΚΑΣ 4.5: ΑΠΟΤΕΛΕΣΜΑΤΑ 1ΟΥ ΧΡΗΣΤΗ 55
ΠΙΝΑΚΑΣ 4.6: ΑΠΟΤΕΛΕΣΜΑΤΑ 2ΟΥ ΧΡΗΣΤΗ 56
ΠΙΝΑΚΑΣ 4.7: ΑΠΟΤΕΛΕΣΜΑΤΑ 3ΟΥ ΧΡΗΣΤΗ 56
ΠΙΝΑΚΑΣ 4.8: ΠΙΝΑΚΑΣ ΣΥΓΚΕΝΤΡΩΤΙΚΩΝ ΑΠΟΤΕΛΕΣΜΑΤΩΝ 1ΗΣ ΑΞΙΟΛΟΓΗΣΗΣ 57
ΠΙΝΑΚΑΣ 4.9: ΠΙΝΑΚΑΣ ΑΠΟΤΕΛΕΣΜΑΤΩΝ ΕΡΩΤΗΜΑΤΟΛΟΓΙΩΝ 1ΗΣ ΑΞΙΟΛΟΓΗΣΗΣ ΓΙΑ ΤΙΣ ΕΡΩΤΗΣΕΙΣ ΠΟΥ ΑΦΟΡΟΥΝ ΤΗΝ ΕΜΠΕΙΡΙΑ ΤΩΝ ΧΡΗΣΤΩΝ ΣΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ.. 57
ΠΙΝΑΚΑΣ 4.10: ΠΙΝΑΚΑΣ ΑΠΟΤΕΛΕΣΜΑΤΩΝ ΕΡΩΤΗΜΑΤΟΛΟΓΙΩΝ 1ΗΣ ΑΞΙΟΛΟΓΗΣΗΣ ΓΙΑ ΤΙΣ ΕΡΩΤΗΣΕΙΣ ΠΟΥ ΑΦΟΡΟΥΝ ΤΗΝ ΓΝΩΜΗ ΤΩΝ ΧΡΗΣΤΩΝ ΓΙΑ ΤΟ ΕΡΓΑΛΕΙΟ 58
ΠΙΝΑΚΑΣ 5.1: ΠΙΝΑΚΑΣ ΜΕ ΜΙΑ ΣΥΝΤΟΜΗ ΚΑΤΑΓΡΑΦΗ ΤΩΝ ΠΡΟΦΙΛ ΤΩΝ ΧΡΗΣΤΩΝ ΤΗΣ ΔΕΥΤΕΡΗΣ ΑΞΙΟΛΟΓΗΣΗΣ 61
ΕΙΚΟΝΑ 5.2: ΟΙ ΕΡΩΤΗΣΕΙΣ ΠΟΥ ΚΛΗΘΗΚΑΝ ΝΑ ΑΠΑΝΤΗΣΟΥΝ ΟΙ ΧΡΗΣΤΕΣ ΠΟΥ ΣΥΜΜΕΤΕΙΧΑΝ ΣΤΗΝ ΔΕΥΤΕΡΗ ΑΞΙΟΛΟΓΗΣΗ ΚΑΤΑ ΤΗΝ ΔΙΑΡΚΕΙΑ ΤΗΣ ΣΥΝΕΝΤΕΥΞΗΣ 62
ΠΙΝΑΚΑΣ 5.3: ΠΙΝΑΚΑΣ ΜΕ ΤΟΝ ΑΡΙΘΜΟ ΤΩΝ ΣΦΑΛΜΑΤΩΝ ΣΤΑ ΟΠΟΙΑ ΟΔΗΓΗΘΗΚΕ Ο ΚΑΘΕ ΧΡΗΣΤΗΣ ΚΑΤΑ ΤΗΝ ΔΙΑΡΚΕΙΑ ΤΗΣ ΧΡΗΣΗΣ ΤΟΥ ΕΡΓΑΛΕΙΟΥ, ΛΟΓΩ ΔΥΣΧΡΗΣΤΙΑΣ ΤΗΣ ΓΡΑΦΙΚΗΣ ΔΙΕΠΑΦΗΣ ΤΟΥ. 63
ΠΙΝΑΚΑΣ 6.1: ΠΙΝΑΚΑΣ ΜΕ ΜΙΑ ΣΥΝΤΟΜΗ ΚΑΤΑΓΡΑΦΗ ΤΩΝ ΠΡΟΦΙΛ ΤΩΝ ΧΡΗΣΤΩΝ ΤΗΣ ΤΡΙΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. «ΚΑΘΟΛΟΥ» ΔΗΛΩΣΑΝ ΟΙ ΦΟΙΤΗΤΕΣ ΠΟΥ ΔΕΝ ΕΙΧΑΝ ΠΡΟΤΕΡΗ ΕΠΑΦΗ ΜΕ ΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ. «ΛΙΓΟ» ΔΗΛΩΣΑΝ ΟΙ ΦΟΙΤΗΤΕΣ ΠΟΥ ΕΙΧΑΝ ΛΙΓΗ ΕΜΠΕΙΡΙΑ ΜΕ ΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ (ΔΕΥΤΕΡΟΒΑΘΜΙΑ ΕΚΠΑΙΔΕΥΣΗ) ΑΛΛΑ ΚΑΘΟΛΟΥ ΜΕ ΤΗ ΓΛΩΣΣΑ C. ΤΕΛΟΣ «ΑΡΚΕΤΗ» ΔΗΛΩΣΕ ΕΝΑΣ ΦΟΙΤΗΤΗΣ ΠΟΥ ΕΙΧΕ ΕΜΠΕΙΡΙΑ ΚΑΙ ΜΕ ΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ ΑΛΛΑ ΚΑΙ ΜΕ ΤΗΝ ΓΛΩΣΣΑ C. 66
ΕΙΚΟΝΑ 6.2: ΤΟ ΕΝΤΥΠΟ ΠΕΡΙΓΡΑΦΗΣ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ ΚΑΙ ΤΩΝ ΕΡΓΑΣΙΩΝ ΤΟΥ ΠΡΩΤΟΥ ΜΕΡΟΥΣ ΤΗΣ ΤΡΙΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ 68
ΕΙΚΟΝΑ 6.3: ΤΟ ΕΝΤΥΠΟ ΠΕΡΙΓΡΑΦΗΣ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ ΚΑΙ ΤΩΝ ΕΡΓΑΣΙΩΝ ΤΟΥ ΔΕΥΤΕΡΟΥ ΜΕΡΟΥΣ ΤΗΣ ΤΡΙΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ 70
ΕΙΚΟΝΑ 6.4: Η ΛΥΣΗ ΤΗΣ ΠΡΩΤΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ ΣΕ BLOCK-C 71
ΠΙΝΑΚΑΣ 6.5: Η ΛΥΣΗ ΤΗΣ ΠΡΩΤΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ ΣΕ C 72
ΕΙΚΟΝΑ 6.6: Η ΛΥΣΗ ΤΗΣ ΔΕΥΤΕΡΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ ΣΕ BLOCK-C 73
ΠΙΝΑΚΑΣ 6.7: Η ΛΥΣΗ ΤΗΣ ΔΕΥΤΕΡΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ ΣΕ C 74
ΕΙΚΟΝΑ 6.8: ΤΟ ΠΡΩΤΟ ΚΟΜΜΑΤΙ ΤΗΣ ΛΥΣΗΣ ΤΗΣ ΤΡΙΤΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ ΣΕ BLOCK-C 75
ΕΙΚΟΝΑ 6.9: ΤΟ ΔΕΥΤΕΡΟ ΚΟΜΜΑΤΙ ΤΗΣ ΛΥΣΗΣ ΤΗΣ ΤΡΙΤΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ ΣΕ BLOCK-C 76
ΠΙΝΑΚΑΣ 6.10: Η ΛΥΣΗ ΤΗΣ ΤΡΙΤΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ ΣΕ C 77
ΕΙΚΟΝΑ 6.11: ΠΡΩΤΗ ΣΕΛΙΔΑ ΕΡΩΤΗΜΑΤΟΛΟΓΙΟΥ ΠΟΥ ΚΛΗΘΗΚΑΝ ΝΑ ΣΥΜΠΛΗΡΩΣΟΥΝ ΟΙ ΧΡΗΣΤΕΣ ΜΕΤΑ ΤΟ ΠΕΡΑΣ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ ΤΗΣ ΤΡΙΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ 78
ΕΙΚΟΝΑ 6.12: ΔΕΥΤΕΡΗ ΣΕΛΙΔΑ ΕΡΩΤΗΜΑΤΟΛΟΓΙΟΥ ΠΟΥ ΚΛΗΘΗΚΑΝ ΝΑ ΣΥΜΠΛΗΡΩΣΟΥΝ ΟΙ ΧΡΗΣΤΕΣ ΜΕΤΑ ΤΟ ΠΕΡΑΣ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ ΤΗΣ ΤΡΙΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ 79
ΕΙΚΟΝΑ 6.13: ΤΡΙΤΗ ΣΕΛΙΔΑ ΕΡΩΤΗΜΑΤΟΛΟΓΙΟΥ ΠΟΥ ΚΛΗΘΗΚΑΝ ΝΑ ΣΥΜΠΛΗΡΩΣΟΥΝ ΟΙ ΧΡΗΣΤΕΣ ΜΕΤΑ ΤΟ ΠΕΡΑΣ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ ΤΗΣ ΤΡΙΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ 80
ΕΙΚΟΝΑ 6.14: ΤΡΙΤΗ ΣΕΛΙΔΑ ΕΡΩΤΗΜΑΤΟΛΟΓΙΟΥ, ΠΟΥ ΠΕΡΙΕΙΧΕ ΤΗΝ ΕΡΩΤΗΣΗ ΓΙΑ ΤΟ ΣΥΝΑΙΣΘΗΜΑ ΠΟΥ ΠΡΟΚΑΛΕΣΕ ΤΟ BLOCK-C ΚΑΤΑ ΤΗΝ ΧΡΗΣΗ ΤΟΥ, ΠΟΥ ΚΛΗΘΗΚΑΝ ΝΑ

ΣΥΜΠΛΗΡΩΣΟΥΝ ΟΙ ΧΡΗΣΤΕΣ ΜΕΤΑ ΤΟ ΠΕΡΑΣ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ ΤΗΣ ΤΡΙΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ [36] 81
ΕΙΚΟΝΑ 6.15: ΤΕΤΑΡΤΗ ΚΑΙ ΤΕΛΕΥΤΑΙΑ ΣΕΛΙΔΑ ΕΡΩΤΗΜΑΤΟΛΟΓΙΟΥ, ΜΕ ΕΡΩΤΗΣΕΙΣ ΕΛΕΥΘΕΡΗΣ ΑΠΑΝΤΗΣΗΣ, ΠΟΥ ΚΛΗΘΗΚΑΝ ΝΑ ΣΥΜΠΛΗΡΩΣΟΥΝ ΟΙ ΧΡΗΣΤΕΣ ΜΕΤΑ ΤΟ ΠΕΡΑΣ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ ΤΗΣ ΤΡΙΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ 82
ΔΙΑΓΡΑΜΜΑ 7.1: ΑΠΟΤΕΛΕΣΜΑΤΑ 1ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΠΡΩΤΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 84
ΔΙΑΓΡΑΜΜΑ 7.2: ΑΠΟΤΕΛΕΣΜΑΤΑ 2ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΠΡΩΤΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 85
ΔΙΑΓΡΑΜΜΑ 7.3: ΑΠΟΤΕΛΕΣΜΑΤΑ 3ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΠΡΩΤΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 86
ΔΙΑΓΡΑΜΜΑ 7.4: ΑΠΟΤΕΛΕΣΜΑΤΑ 4ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΠΡΩΤΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 87
ΔΙΑΓΡΑΜΜΑ 7.5: ΑΠΟΤΕΛΕΣΜΑΤΑ 5ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΠΡΩΤΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 88
ΔΙΑΓΡΑΜΜΑ 7.6: ΑΠΟΤΕΛΕΣΜΑΤΑ 6ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΠΡΩΤΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 89
ΔΙΑΓΡΑΜΜΑ 7.7: ΑΠΟΤΕΛΕΣΜΑΤΑ 7ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΠΡΩΤΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 90
ΔΙΑΓΡΑΜΜΑ 7.8: ΑΠΟΤΕΛΕΣΜΑΤΑ 8ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΠΡΩΤΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 91
ΔΙΑΓΡΑΜΜΑ 7.9: ΑΠΟΤΕΛΕΣΜΑΤΑ 9ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΠΡΩΤΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 92
ΔΙΑΓΡΑΜΜΑ 7.10: ΑΠΟΤΕΛΕΣΜΑΤΑ 10ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΠΡΩΤΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 93
ΔΙΑΓΡΑΜΜΑ 7.11: ΑΘΡΟΙΣΤΙΚΟΙ ΜΕΣΟΙ ΟΡΟΙ ΟΛΟΚΛΗΡΩΣΗΣ ΜΙΑΣ, ΔΥΟ ... ΕΩΣ ΔΕΚΑ ΕΡΓΑΣΙΩΝ. ΓΙΑ ΠΑΡΑΔΕΙΓΜΑ Ο ΜΕΣΟΣ ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ 3ΗΣ ΕΡΓΑΣΙΑΣ ΕΙΝΑΙ ΤΟ ΑΘΡΟΙΣΜΑ ΤΩΝ ΜΕΣΩΝ ΤΗΣ 1ΗΣ ΤΗΣ 2ΗΣ ΚΑΙ ΤΗΣ 3ΗΣ ΕΡΓΑΣΙΑΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ

ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΑΘΡΟΙΣΤΙΚΟ ΜΕΣΟΣ ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Ο ΑΡΙΘΜΟΣ ΤΩΝ ΦΟΙΤΗΤΩΝ ΠΟΥ ΟΛΟΚΛΗΡΩΣΑΝ ΤΟΝ ΣΥΓΚΕΚΡΙΜΕΝΟ ΑΡΙΘΜΟ ΕΡΓΑΣΙΩΝ. ΤΟ ΤΕΡΜΑ ΑΡΙΣΤΕΡΑ ΚΑΙ ΕΠΑΝΩ ΣΗΜΕΙΟ ΔΗΛΩΝΕΙ ΤΗΝ ΟΛΟΚΛΗΡΩΣΗ ΜΙΑΣ ΕΡΓΑΣΙΑΣ· ΠΡΟΧΩΡΩΝΤΑΣ ΕΥΘΕΙΑ ΚΑΙ ΕΠΕΙΤΑ ΜΕ ΒΑΣΗ ΤΗ ΦΟΡΑ ΤΟΥ ΡΟΛΟΓΙΟΥ ΕΙΝΑΙ ΤΑ ΣΤΟΙΧΕΙΑ ΓΙΑ ΤΗΝ ΟΛΟΚΛΗΡΩΣΗ ΔΥΟ … ΕΩΣ ΔΕΚΑ ΕΡΓΑΣΙΩΝ. 94
ΔΙΑΓΡΑΜΜΑ 7.12: ΑΠΟΤΕΛΕΣΜΑΤΑ 1ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΔΕΥΤΕΡΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 95
ΔΙΑΓΡΑΜΜΑ 7.13: ΑΠΟΤΕΛΕΣΜΑΤΑ 2ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΔΕΥΤΕΡΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 96
ΔΙΑΓΡΑΜΜΑ 7.14: ΑΠΟΤΕΛΕΣΜΑΤΑ 3ΗΣ ΕΡΓΑΣΙΑΣ ΤΗΣ ΔΕΥΤΕΡΗΣ ΕΝΟΤΗΤΑΣ ΤΗΣ ΑΞΙΟΛΟΓΗΣΗΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Η ΚΑΤΑΤΑΞΗ ΤΩΝ ΦΟΙΤΗΤΩΝ ΜΕ ΒΑΣΗ ΤΗΝ ΣΕΙΡΑ ΠΟΥ ΤΕΛΕΙΩΣΑΝ. 97
ΔΙΑΓΡΑΜΜΑ 7.15: ΑΘΡΟΙΣΤΙΚΟΙ ΜΕΣΟΙ ΟΡΟΙ ΟΛΟΚΛΗΡΩΣΗΣ ΜΙΑΣ, ΔΥΟ ΚΑΙ ΤΡΙΩΝ ΕΡΓΑΣΙΩΝ. ΓΙΑ ΠΑΡΑΔΕΙΓΜΑ Ο ΜΕΣΟΣ ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ 3ΗΣ ΕΡΓΑΣΙΑΣ ΕΙΝΑΙ ΤΟ ΑΘΡΟΙΣΜΑ ΤΩΝ ΜΕΣΩΝ ΤΗΣ 1ΗΣ ΤΗΣ 2ΗΣ ΚΑΙ ΤΗΣ 3ΗΣ ΕΡΓΑΣΙΑΣ. ΜΕ ΜΠΛΕ ΕΙΝΑΙ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΟ BLOCK-C ΚΑΙ ΜΕ ΚΟΚΚΙΝΟ Η ΟΜΑΔΑ ΠΟΥ ΔΟΥΛΕΨΕ ΜΕ ΤΗΝ C. ΣΤΟΝ ΟΡΙΖΟΝΤΙΟ ΑΞΟΝΑ ΠΑΡΑΤΙΘΕΤΑΙ Ο ΑΘΡΟΙΣΤΙΚΟ ΜΕΣΟΣ ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΗΣ ΕΡΓΑΣΙΑΣ ΚΑΙ ΣΤΟΝ ΚΑΘΕΤΟ ΑΞΟΝΑ Ο ΑΡΙΘΜΟΣ ΤΩΝ ΦΟΙΤΗΤΩΝ ΠΟΥ ΟΛΟΚΛΗΡΩΣΑΝ ΤΟΝ ΣΥΓΚΕΚΡΙΜΕΝΟ ΑΡΙΘΜΟ ΕΡΓΑΣΙΩΝ. ΤΟ ΤΕΡΜΑ ΑΡΙΣΤΕΡΑ ΚΑΙ ΕΠΑΝΩ ΣΗΜΕΙΟ ΔΗΛΩΝΕΙ ΤΗΝ ΟΛΟΚΛΗΡΩΣΗ ΜΙΑΣ ΕΡΓΑΣΙΑΣ· ΠΡΟΧΩΡΩΝΤΑΣ ΕΥΘΕΙΑ ΚΑΙ ΕΠΕΙΤΑ ΜΕ ΒΑΣΗ ΤΗ ΦΟΡΑ ΤΟΥ ΡΟΛΟΓΙΟΥ ΕΙΝΑΙ ΤΑ ΣΤΟΙΧΕΙΑ ΓΙΑ ΤΗΝ ΟΛΟΚΛΗΡΩΣΗ ΔΥΟ ΚΑΙ ΤΡΙΩΝ ΕΡΓΑΣΙΩΝ. 98
ΠΙΝΑΚΑΣ 7.16: ΒΑΣΙΚΗ ΠΕΡΙΓΡΑΦΗ ΤΟΥ ΠΡΟΦΙΛ ΤΩΝ ΧΡΗΣΤΩΝ ΤΗΣ ΟΜΑΔΑΣ ΠΕΙΡΑΜΑΤΙΣΜΟΥ ΚΑΙ ΤΟΥ ΜΕΣΟΥ ΟΡΟΥ ΒΑΘΜΟΛΟΓΙΑΣ ΤΟΥΣ ΣΤΟ ΣΥΝΟΛΟ ΤΩΝ ΕΡΩΤΗΣΕΩΝ ΤΟΥ ΕΡΩΤΗΜΑΤΟΛΟΓΙΟΥ. 99
ΠΙΝΑΚΑΣ 7.17: ΜΕΣΟΣ ΟΡΟΣ ΒΑΘΜΟΛΟΓΙΑΣ ΠΟΥ ΣΥΓΚΕΝΤΡΩΣΕ Η ΚΑΘΕ ΕΡΩΤΗΣΗ. 99
ΔΙΑΓΡΑΜΜΑ 7.18: ΓΡΑΦΙΚΗ ΑΝΑΠΑΡΑΣΤΑΣΗ ΑΠΟΤΕΛΕΣΜΑΤΩΝ ΣΤΗΝ ΕΡΩΤΗΣΗ «ΣΥΝΑΙΣΘΗΜΑΤΑ ΠΟΥ ΣΑΣ ΠΡΟΚΑΛΕΣΕ ΤΟ BLOCK-C». 102
ΠΙΝΑΚΑΣ 8.1: ΧΡΗΣΤΕΣ ΠΟΥ ΟΛΟΚΛΗΡΩΣΑΝ ΚΑΘΕ ΕΡΓΑΣΙΑ ΚΑΙ Ο ΜΕΣΟΣ ΧΡΟΝΟΣ ΓΙΑ ΤΟΥΣ K_I ΠΡΩΤΟΥΣ (K_I = Ο ΑΡΙΘΜΟΣ ΤΩΝ ΧΡΗΣΤΩΝ ΠΟΥ ΤΕΛΕΙΩΣΑΝ ΤΗΝ ΕΡΓΑΣΙΑ Ι ΧΡΗΣΙΜΟΠΟΙΩΝΤΑΣ ΤΗΝ C) 104
ΠΙΝΑΚΑΣ 8.2: ΒΕΛΤΙΩΣΗ ΜΑΘΗΣΙΑΚΗΣ ΑΠΟΔΟΤΙΚΟΤΗΤΑΣ ΤΟΥ BLOCK-C ΕΝΑΝΤΙ ΤΗΣ C (ΒΑΣΙΣΜΕΝΗ ΣΤΟΝ ΜΕΣΟ ΧΡΟΝΟ) 105
ΠΙΝΑΚΑΣ 8.3: ΒΕΛΤΙΩΣΗ ΑΡΙΘΜΟΥ ΠΟΥ ΟΛΟΚΛΗΡΩΣΑΝ ΕΡΓΑΣΙΕΣ ΜΕ BLOCK-C ΕΝΑΝΤΙ ΤΗΣ C (ΒΑΣΙΣΜΕΝΗ ΣΤΟΝ ΑΡΙΘΜΟ ΑΤΟΜΩΝ ΠΟΥ ΟΛΟΚΛΗΡΩΣΑΝ ΜΙΑ, ΔΥΟ ΚΑΙ ΤΡΕΙΣ ΕΡΓΑΣΙΕΣ) 105
ΠΙΝΑΚΑΣ 8.4: ΑΠΟΤΕΛΕΣΜΑΤΑ ΓΙΑ ΤΙΣ ΤΡΕΙΣ ΕΡΩΤΗΣΕΙΣ ΜΕ ΧΑΜΗΛΟΤΕΡΗ ΒΑΘΜΟΛΟΓΙΑ ΚΑΙ ΤΙΣ ΤΡΕΙΣ ΜΕ ΤΗΝ ΥΨΗΛΟΤΕΡΗ. ΠΑΡΟΥΣΙΑΖΟΝΤΑΙ Ο ΑΡΙΘΜΟΣ ΤΗΣ ΕΡΩΤΗΣΗΣ, Η ΕΡΩΤΗΣΗ ΚΑΙ Η ΒΑΘΜΟΛΟΓΙΑ ΠΟΥ ΠΗΡΕ. 105
ΠΙΝΑΚΑΣ 8.5: ΑΠΟΤΕΛΕΣΜΑΤΑ ΣΤΗΝ ΕΡΩΤΗΣΗ «ΣΥΝΑΙΣΘΗΜΑΤΑ ΠΟΥ ΣΑΣ ΠΡΟΚΑΛΕΣΕ ΤΟ BLOCK-C». ΠΑΡΟΥΣΙΑΖΟΝΤΑΙ ΤΟ ΣΥΝΑΙΣΘΗΜΑ ΚΑΙ ΤΟ ΠΛΗΘΟΣ ΤΩΝ ΧΡΗΣΤΩΝ ΠΟΥ ΤΟ ΔΙΑΛΕΞΑΝ (ΠΟΣΟΤΙΚΑ ΚΑΙ ΠΟΣΟΣΤΙΑΙΑ). 106

Περίληψη

Στις ημέρες μας θεωρείται απαραίτητη η εκμάθηση προγραμματισμού στους νεοεισαγόμενους φοιτητές των πανεπιστημιακών τμημάτων, κυρίως θετικών και τεχνολογικών σχολών. Μία ευρέως διαδεδομένη γλώσσα για την εισαγωγή τους στον προγραμματισμό είναι η γλώσσα C.

Έχει αποδειχθεί μετά από έρευνες [24] ότι η σύνταξη είναι ένα από τα πιο σημαντικά προβλήματα που αντιμετωπίζουν οι αρχάριοι στον προγραμματισμό. Συνήθως τα εισαγωγικά μαθήματα στον προγραμματισμό επικεντρώνονται στην λογική και στους αλγόριθμους και αφήνουν τους φοιτητές να μάθουν σύνταξη μέσω της τριβής τους με την συγγραφή προγραμμάτων.

Οι δυσκολίες που αντιμετωπίζουν οι αρχάριοι διογκώνονται όταν η πρώτη τους επαφή με τον προγραμματισμό είναι με μια συντακτικά απαιτητική γλώσσα όπως η C. Πέραν της βιβλιογραφικής μελέτης, η παρατήρηση των ίδιων των φοιτητών και η εμπειρία των διδασκόντων εισαγωγικών μαθημάτων μας έδειξε ότι οι φοιτητές δυσκολεύονται πολύ να προσαρμοστούν σε αυτή. Έτσι αποφασίστηκε, στα πλαίσια της παρούσας διπλωματικής, να δημιουργηθεί ένα εργαλείο το οποίο θα τους εισήγαγε πιο ομαλά στη συγκεκριμένη γλώσσα προγραμματισμού.

Το εργαλείο το οποίο αναπτύχθηκε, ονομάζεται Block-C και μετατρέπει τον προγραμματισμό σε C από κειμενικό σε γραφικό. Σχεδιάστηκε ώστε να κάνει τον προγραμματισμό σε C να μοιάζει με παιχνίδι, αξιοποιώντας γραφικά πλακίδια (blocks) τα οποία χρησιμοποιούνται ευρέως από γραφικές γλώσσες προγραμματισμού για αρχάριους. Μέσω της χρήσης των blocks (τα οποία συνδέονται μέσω στοιχείων που μοιάζουν με puzzle), οι χρήστες προγραμματίζουν σε C δίνοντας βαρύτητα στη λογική παρά στη σύνταξη. Το εργαλείο μετατρέπει το γραφικό πρόγραμμα σε «κλασσική» C (plaintext C). Οι σύνδεσμοι των block δεν επιτρέπουν την σύνδεση οποιονδήποτε block μεταξύ τους, παρά μόνο αυτών που έχουν ίδιου τύπου συνδέσμους (αρσενικό με θηλυκό). Με αυτόν τον τρόπο οι βασικοί συντακτικοί κανόνες της γλώσσας μετατρέπονται σε γραφικούς. Ο χρήστης, έτσι, προστατεύεται από τα συντακτικά λάθη. Όλα τα block παρέχονται κατηγοριοποιημένα, με βάση την λειτουργικότητα του κώδικα που αναπαριστούν, και έτσι ο χρήστης δεν αναγκάζεται να απομνημονεύσει αυτά που χρειάζεται αλλά απλά να τα αναγνωρίσει (recognition over recall)[1].

Το εργαλείο πέρασε από διάφορα στάδια ανάπτυξης (σχεδιασμός, αξιολόγηση γραφικού περιβάλλοντος, εκσφαλματοποίηση, αξιολόγηση αποτελεσματικότητας) πριν ολοκληρωθεί και βγει ως πρωτότυπο. Μετά και από το τελευταίο στάδιο της αξιολόγησης του, που αφορούσε την αποτελεσματικότητα, παρατηρήθηκε ότι όντως μπορεί να βοηθήσει τους αρχάριους να προσαρμοστούν πιο εύκολα και γρήγορα στη C.

Αποτέλεσμα της παρούσας διπλωματικής είναι ένα δωρεάν εργαλείο ανεξάρτητου πλατφόρμας (platform independent) του οποίου ο κώδικας διατίθεται προς οποιονδήποτε ενδιαφερόμενο για περαιτέρω ανάπτυξη (open source).

Κεφάλαιο 1
Εισαγωγή και στόχος παρούσας διπλωματικής

Περίληψη

Σε αυτό το κεφάλαιο γίνεται μια αναφορά για το γενικότερο πλαίσιο (χρονικό και θεματικό) στο οποίο αναπτύχθηκε η παρούσα διπλωματική εργασία. Περιγράφεται η φύση και η ουσία του προβλήματος που προσπαθεί να επιλύσει η εργασία και αναλύονται ο σκοπός, οι ειδικότεροι στόχοι και η αναγκαιότητά της

Εισαγωγή

Τα τελευταία χρόνια έχει δοθεί ιδιαίτερη έμφαση στην εκμάθηση προγραμματισμού υπολογιστών στην δευτεροβάθμια αλλά κυρίως στην τριτοβάθμια εκπαίδευση. Αυτό οφείλεται κυρίως στην καθολική χρήση των υπολογιστών, στην ραγδαία εξάπλωση της λεγομένης *κουλτούρας της συμμετοχής*[2] και στην τάση για ανάπτυξη[3], προγραμματισμό[4] και ενοποίηση τεχνολογιών[5] από τον τελικό χρήστη. Αυτή η τάση κατευθύνει προς εργαλεία λογισμικού τα οποία παρέχουν ισχυρές υψηλού επιπέδου γλώσσες, έτσι ώστε να επιτρέψουν την ευέλικτη προσαρμογή και την πλούσια ανάπτυξη αλληλεπιδραστικού περιεχομένου από τους τελικούς χρήστες. Από αυτή την άποψη, η γνώση προγραμματιστικών εννοιών είναι, στις ημέρες μας, απαραίτητη για τους περισσότερους εξειδικευμένους εργαζομένους συμπεριλαμβανομένων των επιστημόνων, μηχανικών και τεχνολόγων. Κατά συνέπεια, πολλά ιδρύματα τριτοβάθμιας εκπαίδευσης έχουν συμπεριλάβει εισαγωγικά μαθήματα προγραμματισμού στα προγράμματα σπουδών τους. Περαιτέρω, πολλές χώρες, εμπλουτίζουν τα προγράμματα της δευτεροβάθμιας ή ακόμα και της πρωτοβάθμιας εκπαίδευσης με σκοπό την ανάπτυξη βασικών δεξιοτήτων στον προγραμματισμό.

Αυτή η τάση διευκολύνεται αρκετά από την γρήγορη εξάπλωση των εκπαιδευτικών γλωσσών προγραμματισμού οι οποίες κατά κανόνα συνοδεύονται από ελκυστικά γραφικά περιβάλλοντα. Αυτά τα περιβάλλοντα προωθούν την μάθηση στην πράξη (situated learning)[41] ενεργοποιώντας την ανάπτυξη ηλεκτρονικών παιχνιδιών, γραφικών κινουμένων σχεδίων και άλλων ψηφιακών εργαλείων, στα οποία οι μαθητευόμενοι μαθαίνουν δημιουργώντας. Αρκετές τέτοιες γλώσσες είναι βασισμένες σε γραφικές δομικές μονάδες, οι οποίες θυμίζουν τουβλάκια Lego[6]. Ο κώδικας συναρμολογείται από τουβλάκια, τα οποία αναπαριστούν εντολές, και χειρίζονται γραφικά όπως τα τουβλάκια της Lego. Η στροφή της επιστήμης που μελετάει την μάθηση (προγραμματισμού) προς τέτοιες προσεγγίσεις καταδεικνύει ότι υπάρχουν αδυναμίες στην χρήση των γενικού

σκοπού γλωσσών προγραμματισμού για την εισαγωγή αρχαρίων στις έννοιες του προγραμματισμού[7].

Παρά την αποτελεσματικότητα τους, οι εκπαιδευτικές γλώσσες προγραμματισμού δεν μπορούν σε όλες τις περιπτώσεις να χρησιμοποιηθούν σε εισαγωγικά μαθήματα προγραμματισμού. Αυτό συμβαίνει σε πολλά τμήματα μηχανικών λόγω της κατεύθυνσης των προγραμμάτων σπουδών τους προς το υλικό (hardware), την ρομποτική, τους αυτοματισμούς, τον προγραμματισμό συστημάτων και άλλα πεδία τα οποία χρειάζονται δεξιότητες σε πιο «χαμηλού επιπέδου» (low level) γλώσσες όπως η C.

Η παρούσα εργασία στοχεύει να καλύψει αυτές τις περιπτώσεις όπου η χρήση της γλώσσας C θεωρείται αναγκαία. Αποσκοπεί στην εκμετάλλευση της εμπειρίας, που προέρχεται από τις εκπαιδευτικές γλώσσες προγραμματισμού βασισμένες σε τουβλάκια, για να προσφέρει ένα γραφικό κέλυφος «πάνω» από την C. Αυτό θα παρέχει καθοδήγηση και θα προλαμβάνει λάθη μέσω γραφικής κωδικοποίησης, όπως συντακτικά λάθη. Σκοπός είναι να μπορέσουν οι αρχάριοι να εκπαιδευτούν στην πράξη και να έχουν την καθοδήγηση και την βοήθεια που θα είχαν αν υποβοηθούνταν από ανθρώπους. Το αποτέλεσμα της προαναλυθείσας διαδικασίας θα οδηγήσει στην αφομοίωση των εννοιών και των δομών της γλώσσας C [8] και στην δημιουργία κατάλληλων νοητικών μοντέλων για την σωστή χρήση της.

Ερώτημα που απασχολεί την παρούσα εργασία

Το ερώτημα του οποίου η απάντηση θα μπορούσε συνοπτικά να περιγράψει τον σκοπό της παρούσας διπλωματικής και τις συνθήκες κάτω από τις οποίες αυτή διεξήχθη είναι το εξής:

Πώς θα μπορούσε ένας αρχάριος, στον προγραμματισμό, να αντιλαμβάνεται καλύτερα τον τρόπο σύνταξης και την συνδυασμένη χρήση εντολών στη γλώσσα C, ώστε να εκπαιδευτεί αποτελεσματικότερα στις βασικές αρχές του προγραμματισμού οικοδομώντας παράλληλα –με τρόπο ελεγχόμενο και σταδιακό- τεχνικές δεξιότητες αποτελεσματική χρήσης της C;

Έχει παρατηρηθεί, ότι οι αρχάριοι που προσπαθούν να μάθουν C, αντιμετωπίζουν μια σειρά, βασικών, εμποδίων και δυσκολιών [40].

Τα παραπάνω προβλήματα είναι κοινά για τους περισσότερους αρχάριους. Οφείλονται στη δυσκολία της γλώσσας C (λόγω πολυπλοκότητας της γραμματικής της [24]) και στο ότι αρκετοί από αυτούς δεν έχουν πρότερη εμπειρία με τον προγραμματισμό.

Όλες οι σύγχρονες προσεγγίσεις εκμάθησης προγραμματισμού σε αρχάριους βασίζονται στην χρήση γραφικών «μεταφορών» (metaphors) αντί για κώδικα. Ορμώμενοι από το γεγονός αυτό, αποφασίσαμε να προσπαθήσουμε να δημιουργήσουμε ένα γραφικό κέλυφος για την γλώσσα C.

Ο στόχος που τέθηκε ήταν να μπορούν οι χρήστες να αντιληφθούν διαισθητικά το πώς δομείται ένα πρόγραμμα. Επίσης, να μπορούν να αφιερωθούν

στην ανάπτυξη εφαρμογών χωρίς να αποθαρρύνονται από τα συντακτικά λάθη (όπως να ξεχνάει τα απαραίτητα σημεία στίξης της γλώσσας) και να καταναλώνουν άσκοπα χρόνο προσπαθώντας να τα επιλύσουν.

Μια ακόμα διάσταση ήταν αυτή της αίσθησης ευχαρίστησης και ηρεμίας που έπρεπε προκαλεί το εργαλείο στους χρήστες του. Το εργαλείο έπρεπε να έχει την δυνατότητα να προσελκύει τους χρήστες και να τους «παρακινεί» να το χρησιμοποιήσουν. Έτσι πάρθηκε η απόφαση η διεπαφή του εργαλείου να θυμίζει παιχνίδι στους χρήστες.

Ως εργαλείο εισαγωγής στον προγραμματισμό, έπρεπε να είναι προσιτό και ευκατάληπτο. Τέλος έπρεπε να μπορεί να εισάγει τους χρήστες άμεσα στο κλασσικό τρόπο του να γράφεις προγράμματα στην C (plain Text Programming)

Αναγκαιότητα της παρούσας διπλωματικής εργασίας

Η αναγκαιότητα αυτής της διπλωματικής είναι συνυφασμένη με την αναγκαιότητα για βελτίωση των τρόπων εκμάθησης προγραμματισμού. Όλες οι προαναφερθείσες δυσκολίες μπορούν να ξεπεραστούν όταν οι φοιτητές υποβοηθούνται από διδακτικό/εργαστηριακό προσωπικό. Συνήθως το εργαστηριακό προσωπικό και οι βοηθοί του μαθήματος δεν μπορούν να διαθέσουν πάρα πολλές ώρες λόγω αυξημένου φόρτου εργασίας, και του μεγάλου αριθμού φοιτητών. Επίσης ο προγραμματισμός απαιτεί πρακτική εμπειρία ώστε να εμπεδωθεί και αφομοιωθεί από τον αρχάριο.

Έτσι έπρεπε το εργαλείο να καθοδηγεί τους φοιτητές κατάλληλα και σωστά χωρίς να έχουν την ανάγκη του προσωπικού (τουλάχιστον για τα αρχικά τους βήματα στον προγραμματισμό). Άρα το εργαλείο ήρθε να ενισχύσει την εκμάθηση προγραμματισμού και να δώσει λύση στο πρόβλημα έλλειψης ανθρωπίνων πόρων.

Γιατί το εργαλείο πετυχαίνει τον στόχο του

Το εργαλείο καθ' όλη την διάρκεια της ανάπτυξης του πέρασε από τρεις αξιολογήσεις. Στις αξιολογήσεις αυτές πήραν μέρος χρήστες τα προφίλ των οποίων ήταν όμοια με αυτά των πραγματικών χρηστών για τους οποίους δημιουργήθηκε το εργαλείο.

Οι δύο πρώτες αξιολογήσεις αφορούσαν την ευχρηστία της γραφικής διεπαφής [9] και η τρίτη αφορούσε την αποτελεσματικότητα του εργαλείου στο να υποβοηθά την μάθηση της C.

Τα αποτελέσματα αυτών των τριών αξιολογήσεων έδειξαν ότι το εργαλείο αποτελεί αξιόλογο βοήθημα και πετυχαίνει τον στόχο για τον οποίο δημιουργήθηκε. Τα εν λόγω αποτελέσματα και τα συμπεράσματα των τριών αξιολογήσεων παρατίθενται στο παρόν κείμενο σε κεφάλαια που ακολουθούν.

Δομή της παρόντος κειμένου

Η παρούσα εργασία δομείται στα επιμέρους κεφάλαια ως εξής:

- Το κεφάλαιο 2 παρουσιάζει κάποιες από τις σχετικέςεργασίες και τις τεχνολογίες που χρησιμοποιήθηκαν.

- Το κεφάλαιο 3 αναλύει τα θέματα που άπτονται του σχεδιασμού και της υλοποίησης και κάνει μια σύντομη παρουσίαση του Block-C.
- Το κεφάλαιο 4 ασχολείται με την διαδικασία και τα αποτελέσματα της πρώτης αξιολόγησης του Block-C, η οποία αφορούσε την ευχρηστία της γραφικής διεπαφής
- Στο κεφάλαιο 5 γίνεται μια παρουσίαση της διαδικασίας και των αποτελεσμάτων της δεύτερης αξιολόγησης που αφορούσε την περαιτέρω αξιολόγηση της διεπαφής και της δυνατότητάς της να προλαμβάνει τα λάθη σύνταξης
- Το κεφάλαιο 6 ασχολείται με την παρουσίαση της διαδικασίας της τρίτης, και πιο σημαντικής, αξιολόγησης που αφορά την αποτελεσματικότητα του Block-C.
- Στο 7^{ο} κεφάλαιο υπάρχει μια εκτενής παρουσίαση και ανάλυση των αποτελεσμάτων της τρίτης αξιολόγησης
- Τέλος, στο 8^{ο} και τελευταίο κεφάλαιο, αναλύονται τα συμπεράσματα της παρούσας εργασίας και προτείνονται μελλοντικές επεκτάσεις.

Κεφάλαιο 2
Σχετικές εργασίες και εργαλεία που χρησιμοποιήθηκαν

Περίληψη

Σε αυτό το κεφάλαιο αναλύονται τα εργαλεία τα οποία προσπαθούν να αντιμετωπίσουν παρόμοια προβλήματα όσον αφορά τη διδασκαλία του προγραμματισμού. Επίσης αναλύεται η γενικότερη θεωρία στην οποία στηρίζονται τέτοιου είδους προσεγγίσεις εκμάθησης προγραμματισμού. Τέλος παρουσιάζονται οι τεχνολογίες που χρησιμοποιήθηκαν στην παρούσα εργασία.

Θεωρία που διέπει την υποβοηθούμενη μάθηση

Θεωρία Δραστηριοτήτων (Activity Theory)

Η Θεωρία Δραστηριοτήτων είναι βασισμένη στην επιστήμη της ψυχολογίας και στοχεύει στην κατανόηση των μεμονωμένων ανθρωπίνων όντων, όπως και των κοινωνικών οντοτήτων των οποίων αυτοί συνθέτουν. Για να επιτευχθεί αυτός ο στόχος αναλύει την γένεση, τη δομή και τις διαδικασίες των δραστηριοτήτων τους, παρατηρώντας τους στις καθημερινές τους περιστάσεις. Η βασική ιδέα της *δραστηριότητας (activity)* είναι έτσι το πιο θεμελιώδης ιδέα σε αυτή την θεωρία.

Η δραστηριότητα γενικότερα, όχι μόνο η ανθρώπινη, αλλά η δραστηριότητα οποιουδήποτε *υποκειμένου (subject),* είναι αντιληπτή ως μια σκόπιμη αλληλεπίδραση του υποκειμένου με τον κόσμο. Μέσω αυτής της διαδικασίας επιτυγχάνονται αμοιβαίοι μετασχηματισμοί μεταξύ των πόλων «υποκείμενο-αντικείμενο» (Leontiev 1978).

Αλληλεπίδραση Ανθρώπου-Υπολογιστή(HCI) και Θεωρία Δραστηριοτήτων(Activity Theory)

Τα μοντέλα της αλληλεπίδρασης ανθρώπου-υπολογιστή τα οποία ήταν δημοφιλή στο πρώτο «κύμα» του HCI (Nielsen κλπ), φαινόταν ότι επικεντρωνόταν στις ίδιες μονάδες όπως και η θεωρία δραστηριοτήτων (activity theory), δηλαδή στην αλληλεπίδραση μεταξύ ανθρώπων (χρήστες) και αντικειμένων (διαδραστικά συστήματα).

Παρόλα αυτά, τα παραδοσιακά μοντέλα του HCI επικεντρώνονται σε πιο χαμηλού επιπέδου αλληλεπίδραση η οποία περιορίζεται σε εργασίες (tasks). Πέρα όμως από την λειτουργικότητα του εργαλείου μας ενδιαφέρει και ο ρόλος του. Ως εργαλείο που έχει ως σκοπό του την υποστήριξη της μάθησης πρέπει να μελετηθεί με βάση τις υπάρχουσες θεωρίες που διέπουν την μάθηση.

Η διαμεσολάβηση ενός εργαλείου μεταξύ υποκειμένου και αντικειμένου, η οποία αναπτύχθηκε μέσα σε μια πολιτισμική-ιστορική προσέγγιση, η οποία εισήχθηκε στο HCI από τον Bødker (1989,1991), οδήγησε στην ανάπτυξη των εννοιών των νοητικών τεχνουργημάτων (cognitive artifacts – Norman 1991) και του «person-plus» (Perkins 1993), και πιο πρόσφατα χρησιμοποιήθηκε για να αναθεωρήσει το παράδειγμα του άμεσου χειρισμού (direct manipulation paradigm-Beaudouin-Lafon 2000).

Έτσι το 1991 ο Bødker πρότεινε την Θεωρία Δραστηριοτήτων, όπως αυτή εισήχθηκε από τον Leontiev και τους συνεχιστές του, ως την θεωρητική θεμελίωση για το HCI [38].

Υποκείμενο-Αντικείμενο-Εργαλείο

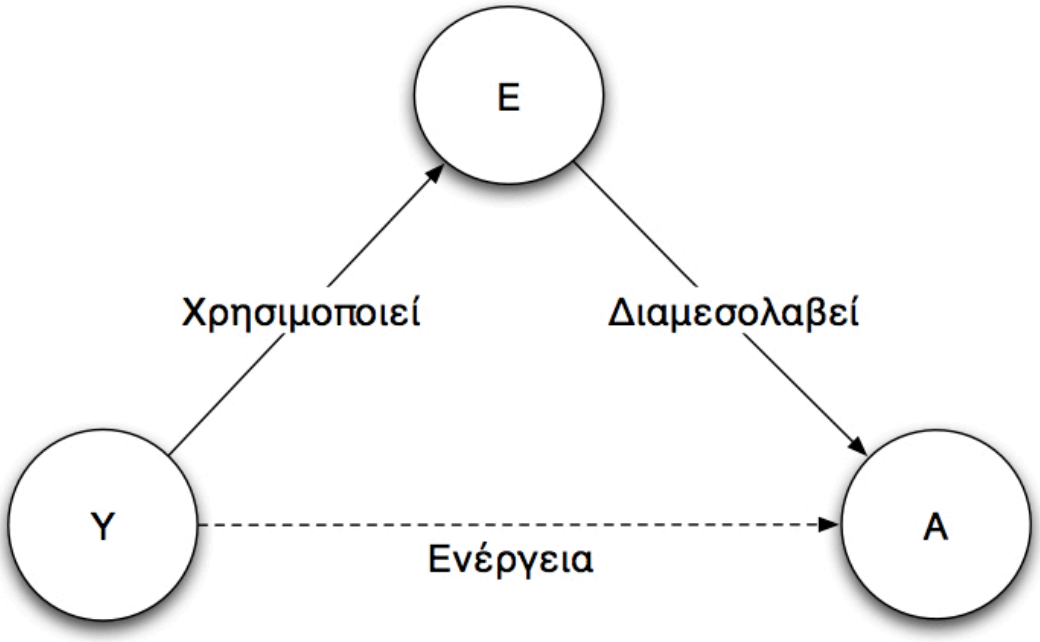
Η Θεωρία Δραστηριοτήτων αποτελεί μια από τις βασικές θεωρίες για την υποβοηθούμενη μάθηση. Χωρίζει την αλληλεπίδραση του ανθρώπου με το περιβάλλον, μέσω της τεχνολογίας, σε τρεις διακριτές (και όχι ομότιμες) οντότητες:

1. Το υποκείμενο

2. Το αντικείμενο

3. Το εργαλείο

Το υποκείμενο προσπαθεί να εκτελέσει μια ενέργεια. Η ενέργεια αυτή οδηγεί σε ένα αποτέλεσμα (το αντικείμενο). Το υποκείμενο, για να επιτύχει τον στόχο του χρησιμοποιεί ένα εργαλείο. Το εργαλείο «μεσολαβεί» για να επιτευχθεί ο στόχος. Η παραπάνω διαδικασία περιγράφεται σχηματικά ως εξής:

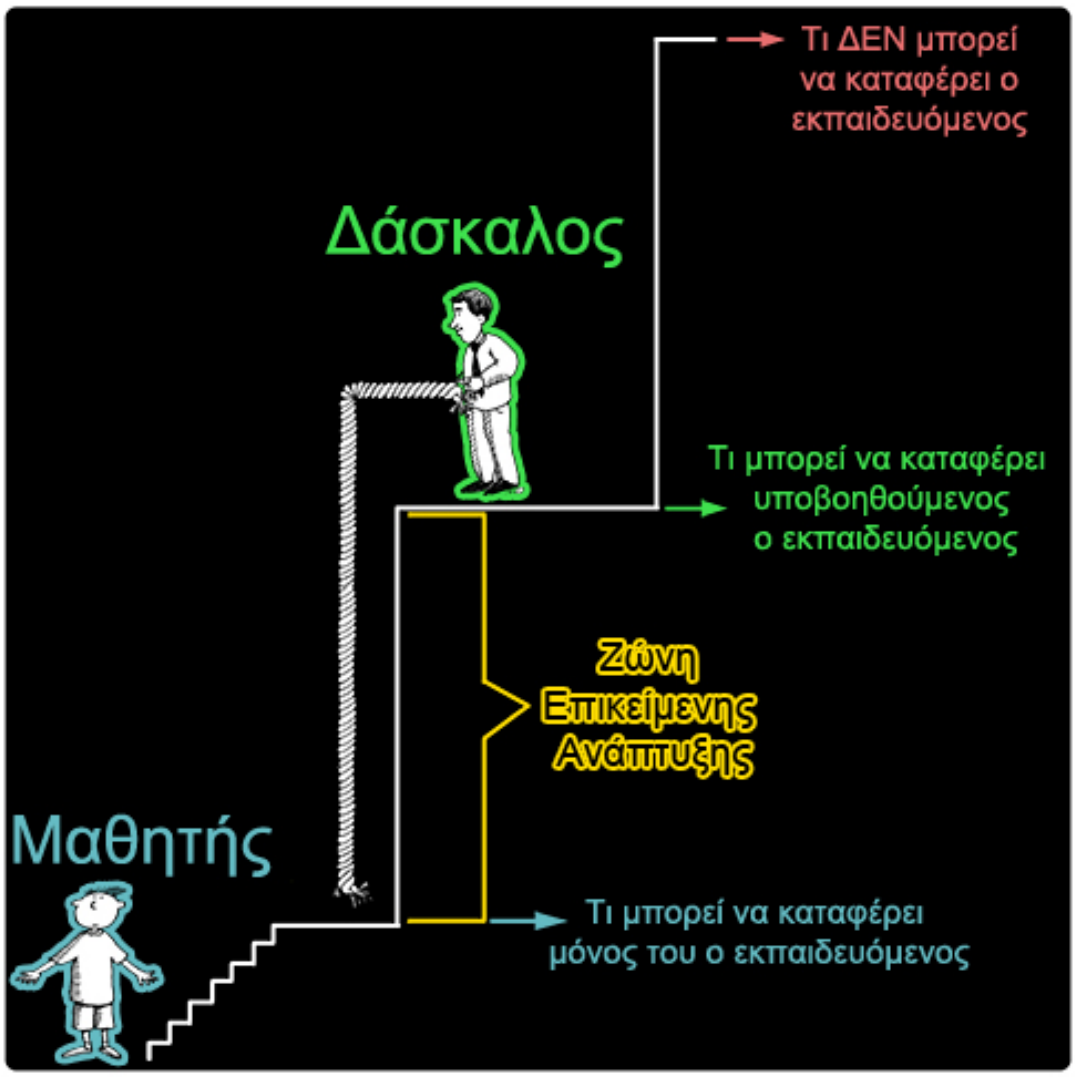


Διάγραμμα 2.1: Υ = Υποκείμενο , Ε = Εργαλείο , Α = Αντικείμενο. Ένα υποκείμενο δρα μέσω μιας ενέργειας για να καταλήξει στο στόχο (αντικείμενο). Στην ενέργεια αυτή μεσολαβεί το εργαλείο το οποίο χρησιμοποιείται από τον χρήστη (υποκείμενο).

Ο Vygotsky ανακάλυψε ότι τα υποκείμενα χρησιμοποιούν διαμεσολαβητικά εργαλεία για να λύσουν τα προβλήματα τους, αλλά μετά από την πάροδο του χρόνου, σταματούν να τα χρησιμοποιούν διατηρώντας όμως υψηλά επίπεδα απόδοσης.

Ζώνη Επικείμενης Ανάπτυξης (Zone of Proximal Development)

Στα πλαίσια της Θεωρία Δραστηριοτήτων αναπτύχθηκε και η έννοια της «Ζώνης Επικείμενης Ανάπτυξης» (Zone of Proximal Development ή ZPD) [8] . Η Ζώνης Επικείμενης Ανάπτυξης ερμηνεύεται ως η διαφορά μεταξύ του τι μπορεί να πετύχει ο μαθητευόμενος μόνος του και τι όταν υποβοηθείται.



Διάγραμμα 1.3: Ζώνη Επικείμενης Ανάπτυξης

Σκοπός της εργασίας μας είναι να εισέλθει ο χρήστης στο ZPD, μέσω του εργαλείου που κατασκευάζουμε, χωρίς την υποβοήθηση από κάποιο πρόσωπο. Επίσης το ZPD μας προσφέρει ένα τρόπο μέτρησης της αποδοτικότητας του εργαλείου μας πέρα από το παραδοσιακές μετρικές που χρησιμοποιούνται στην Επικοινωνία Ανθρώπου – Υπολογιστή και αφορούν κατά κανόνα την ευχρηστία των διεπαφών.

Πιο συγκεκριμένα ο Vygotsky (θεμελιωτής της ιδέας του ZPD) προτείνει την μέτρηση της αποδοτικότητας του μαθητευόμενου μέσω της διαφοράς της λύσης του ίδιου προβλήματος:

(1) από έναν μη υποβοηθούμενο και (2) έναν υποβοηθούμενο (από εργαλείο ή άνθρωπο).

Η πρόταση του Vygotsky για την μέτρηση του ZPD δεν αναφέρει τον χρόνο στον οποίο θα διεξαχθεί η μέτρηση (πχ στο ίδιο χρονικό όριο και για τους δύο ή όχι).

Έτσι αποφασίσαμε να θέσουμε ένα μέγιστο χρονικό όριο, ίδιο και για τα δύο είδη μαθητευόμενων (υποβοηθούμενο και μή), ώστε να εστιάσουμε στην αποτελεσματικότητα των δύο σε συγκεκριμένο χρόνο. Αυτή η απόφαση συνάδει και με τον τρόπο που εκτελείται η διαδικασία της μάθησης. Οι μαθητευόμενοι πρέπει να εκπαιδευτούν στο αντικείμενο σε ένα περιορισμένο χρονικό διάστημα (πχ ένα εξάμηνο). Άρα, θα υπολογίσουμε το ZPD με βάση το ποσοστό ολοκλήρωσης των εργασιών που θα τεθούν.

Το ZPD παρόλα αυτά, είναι προσεγγιστικό. Τα ποσοστά επιτυχίας εξαρτώνται από το ενδιαφέρον του εκπαιδευόμενου και από την πρότερη του γνώση[48].

Αφομοίωση και Εξωτερίκευση (Internalization & Externalization)

Η θεωρία δραστηριοτήτων εισάγει δύο ακόμα έννοιες που μας βοηθούν να κατανοήσουμε το πώς υιοθετείται - αφομοιώνεται (internalization) και το πώς αναλύεται για να εξωτερικευτεί (externalization) η πληροφορία από τον εγκέφαλο.

Internalization: Η μετάφραση-μετασχηματισμός εξωτερικών δραστηριοτήτων σε εσωτερικές.

Παρέχει ένα μέσο για τους ανθρώπους να εξετάσουν πιθανές αλληλεπιδράσεις με την πραγματικότητα μέσα από «νοητικές προσομοιώσεις ή αναπαραστάσεις» χωρίς να εκτελέσουν οποιονδήποτε χειρισμό με πραγματικά αντικείμενα. Οι νοητικές πλέον διαδικασίες έχουν την μορφή του αυτοματισμού και τα χαρακτηριστικά των αντανακλαστικών.

Externalization: Η μετατροπή εσωτερικευμένων διαδικασιών σε εξωτερικές δραστηριότητες

Πρόκειται ουσιαστικά για την ανάλυση των νοητικών μοντέλων και την εξωτερίκευση τους σε συγκεκριμένες διαδικασίες.

Εκμάθηση προγραμματισμού σε αρχάριους

Ιστορική αναδρομή

Η πρώτη γλώσσα προγραμματισμού για εκπαιδευτικούς σκοπούς ήταν η Logo. Το 1967 οι Wally Feurzeig και Seymour Papert εμπνεύστηκαν και υλοποίησαν την Logo. Το όνομα της προήλθε από το ελληνικό «Λόγος» προσπαθώντας να δώσει έμφαση στην διαφορά μεταξύ αυτής και των τότε γλωσσών προγραμματισμού που χειρίζονταν αριθμούς[10].

Το 1969 στο Xerox PARC ξεκίνησε μια προσπάθεια από τον Alan Kay και την ομάδα του για να δημιουργηθεί μια γλώσσα προγραμματισμού για εκπαιδευτική χρήση. Σκοπός ήταν να υποστυλωθεί η «νέα εποχή» των υπολογιστών όπου θα

υπήρχαν εύκολα αντιληπτοί –νοητικοί- συμβολισμοί από τους ανθρώπους στα metaphor των υπολογιστών.

Ως αποτέλεσμα ,το 1972, προέκυψε η Smalltalk [11]. Η Smalltalk, βασισμένη στην οντοκεντρική λογική που εισήγαγε το 1967 η Simula [12], ήταν ουσιαστικά η πρώτη γλώσσα οντοκεντρικού προγραμματισμού.

Σύγχρονα εργαλεία και γλώσσες προγραμματισμού για αρχάριους

Κάπως έτσι ένας καινούριος κλάδος, στην επιστήμη των υπολογιστών, ξεκίνησε να αναπτύσσεται. Σκοπός του είναι να κάνει να κάνει πιο κατανοητές και διαισθητικές τις έννοιες του προγραμματισμού στους αρχάριους, όπως επίσης και να τους καθοδηγεί ώστε μαθαίνουν γρήγορα και αποτελεσματικά.

Ένα κομμάτι αυτού του κλάδου είναι και το «*visual computing*». Το visual computing χωρίζεται σε δύο κλάδους[13] που έχουν σχέση με την δικιά μας δουλειά. Ο πρώτος είναι οι Γραφικές Γλώσσες Προγραμματισμού (Visual Programming Languages, γνωστές και ως VPLs) και τα Γραφικά Εργαλεία για Κειμενικές Γλώσσες (Visual Tools for Textual Languages, γνωστά και ως VETL).

Μερικά από τα πιο γνωστά παραδείγματα, παιδαγωγικού χαρακτήρα, Γραφικών Γλωσσών Προγραμματισμού (VPLs) είναι το Scratch [42] (Εικόνα 2.9), το Etoys [43] (Εικόνα 2.7) και η StarLogo [44] (Εικόνα 2.6). Το Alice [45] (Εικόνα 2.8), το Greenfoot [46], η Visual Basic [47], η Java2Sequence[14] και πολλά άλλα[15] είναι παραδείγματα Γραφικών Εργαλείων για Κειμενικές Γλώσσες (VETL). Τα εργαλεία αυτά επιτρέπουν την ανάπτυξη περίπλοκων εφαρμογών και είναι κατάλληλες για προσεγγίσεις RAD[16] (Rapid Application Development)

Επίσης υπάρχουν τα σύγχρονα περιβάλλοντα ανάπτυξης εφαρμογών γνωστά ως IDE[17] (Integrated Development Environments) τς οποία χρησιμοποιούν κάποιες γραφικές διευκολύνσεις οι οποίες μπορούν να βοηθήσουν τους αρχάριους.

Συγκεκριμένα παρέχουν λειτουργίες κατά την διάρκεια που ο χρήστης συντάσσει τον κώδικα (όπως Text Highlighting Εικόνα 2.4 ή Code Completion Εικόνα 2.5). Τα παραπάνω έχουν ως σκοπό τους την αποφυγή συντακτικών λαθών, την οπτική δόμηση του κώδικα (διακριτοποίηση κομματιών του) και την αυτόματη παραγωγή κώδικα μέσω προτάσεων (που κάνει το σύστημα στον χρήστη Εικόνα 2.5).

Τα προαναφερθέντα βοηθήματα δεν βελτιώνουν την ικανότητα αντίληψης του αρχαρίου χρήστη αλλά λειτουργούν ως καθοδηγητικές μέθοδοι για τους εξοικειωμένους ,κυρίως, προγραμματιστές.

```
public void changeGenusTo(String genusNam
    System.out.println("changing genus");
    this.genusName = genusName;
    label = BlockGenus.getGenusWithName(g
    //return new Block(genusName);
}
```

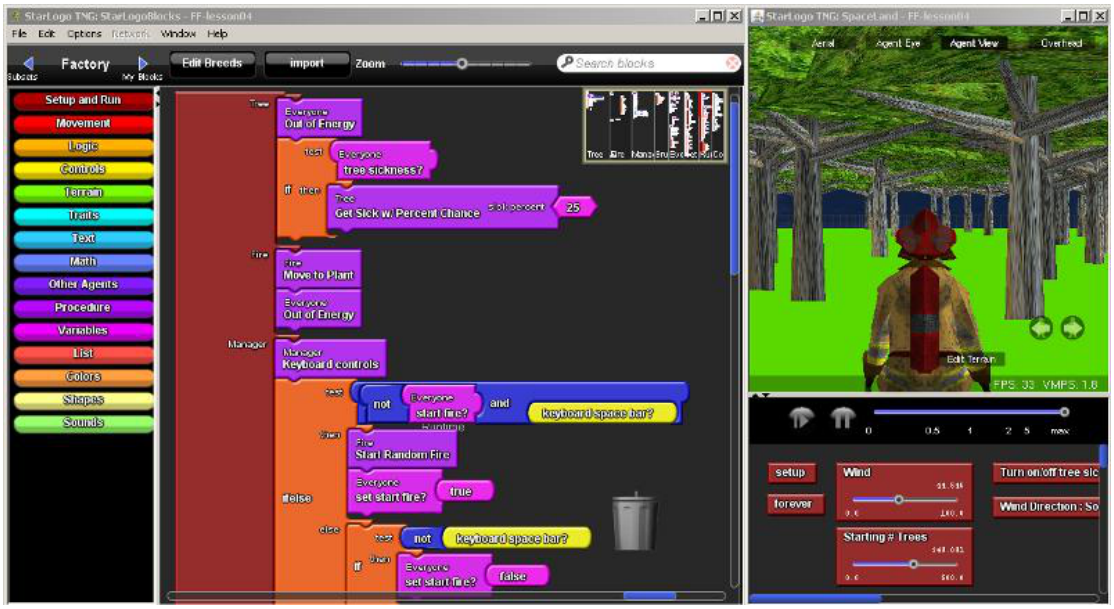
Εικόνα 2.4: Οπτική Επισήμανση Κειμένου (Text highlighting): Μέθοδος για την διακριτοποίηση των διαφορετικών συντακτικών μονάδων του προγράμματος.

Εικόνα 2.5: Αυτόματη Συμπλήρωση Κώδικα (Code Completion): Μέθοδος για να προτείνει το σύστημα πιθανά κομμάτια κώδικα που ίσως έπονται αυτών που γράφει ο χρήστης.

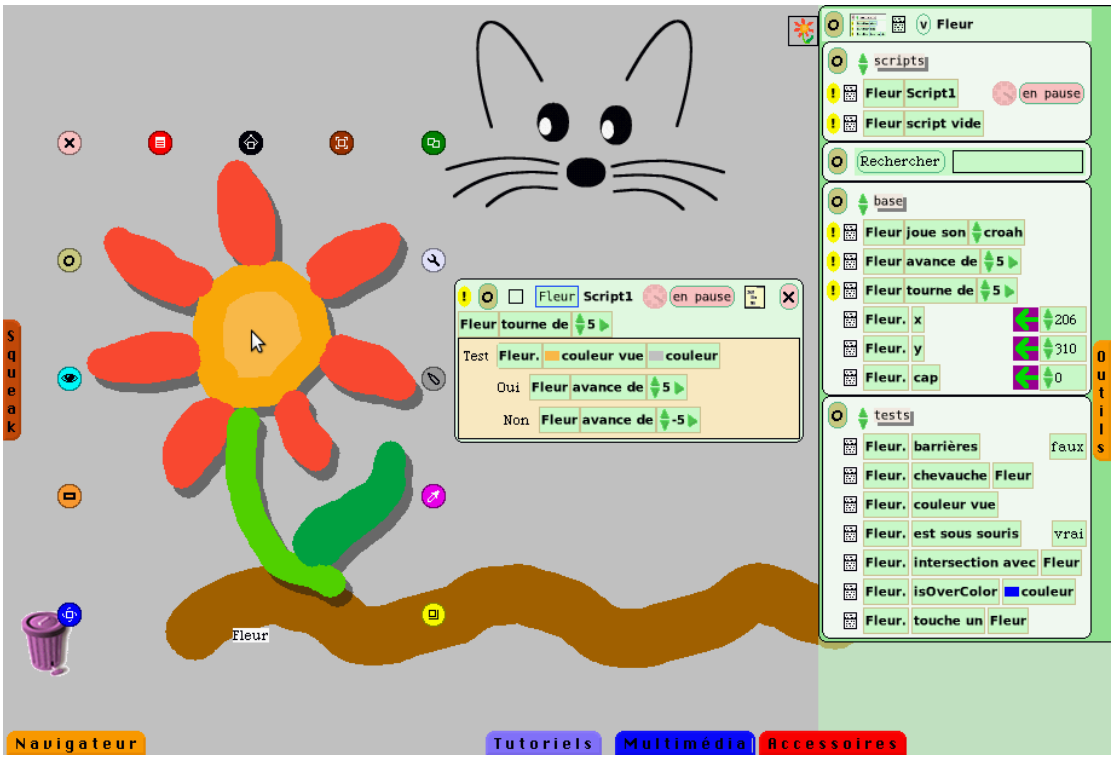
Πολλοί ερευνητές έχουν ασχοληθεί με αυτές τις γλώσσες και εργαλεία με σκοπό να εξάγουν πληροφορίες σχετικά με τις δυνατότητές τους όσο αφορά την υποστήριξη της διαδικασίας της μάθησης[18,19,20,21,22]. Περαιτέρω, έρευνες έχουν διεξαχθεί για τις δυσκολίες που αντιμετωπίζουν οι προγραμματιστές λόγω των συντακτικών λαθών [23,24] αλλά και γενικότερα για τα προβλήματα που αντιμετωπίζουν οι μαθητευόμενοι[25,26].

Οι VPLs και τα VETL συνήθως δεν είναι γενικού σκοπού αλλά ειδικού. Για παράδειγμα το Alice (Εικόνα 2.8) είναι ειδικού σκοπού και συγκεκριμένα για την δημιουργία γραφικών. Ένα ενδιαφέρον εργαλείο (VETL) γενικού σκοπού που χρησιμοποιεί την πλήρη λειτουργικότητα της γλώσσας για την οποία σχεδιάστηκε είναι το HyperPascal[27].

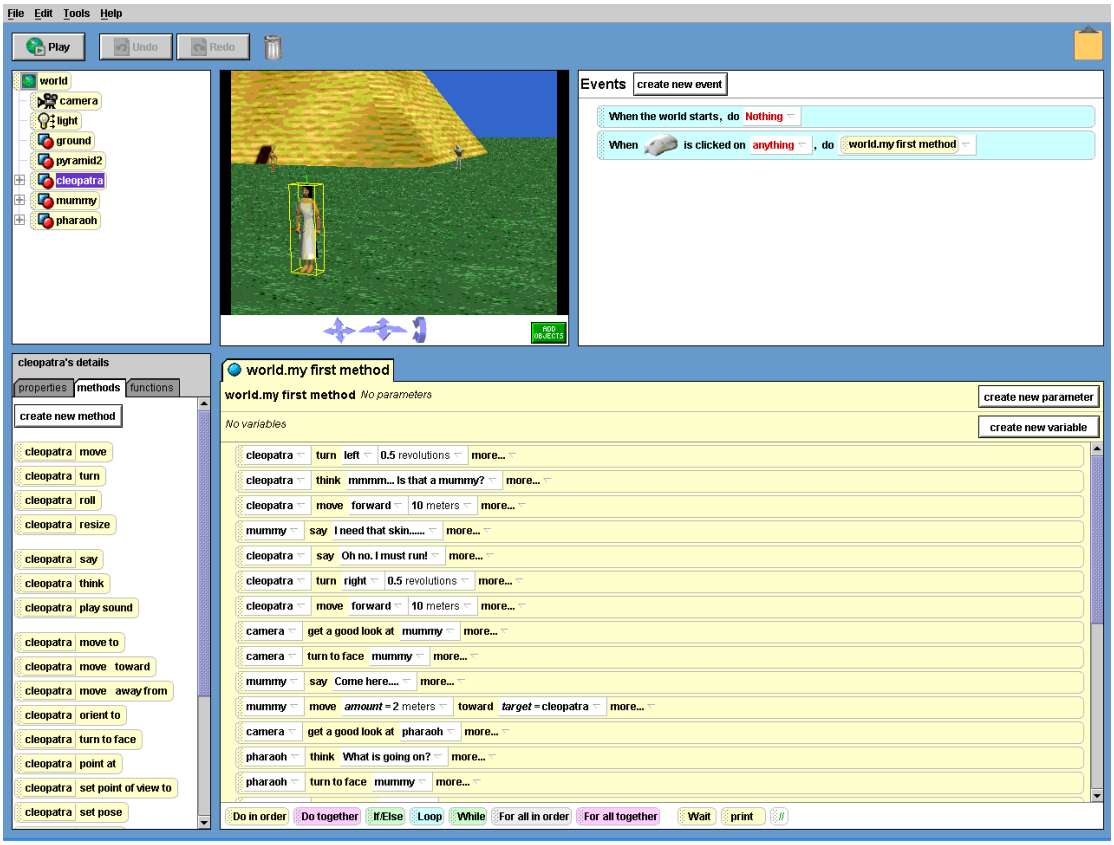
Το Block-C αποτελεί VETL γενικού σκοπού σχεδιασμένο για να ενισχύει την διαδικασία μάθησης της γλώσσας προγραμματισμού C. Τα προβλήματα της σύνταξης είναι αυτά που πρωταρχικά προσπαθεί να αντιμετωπίσει.



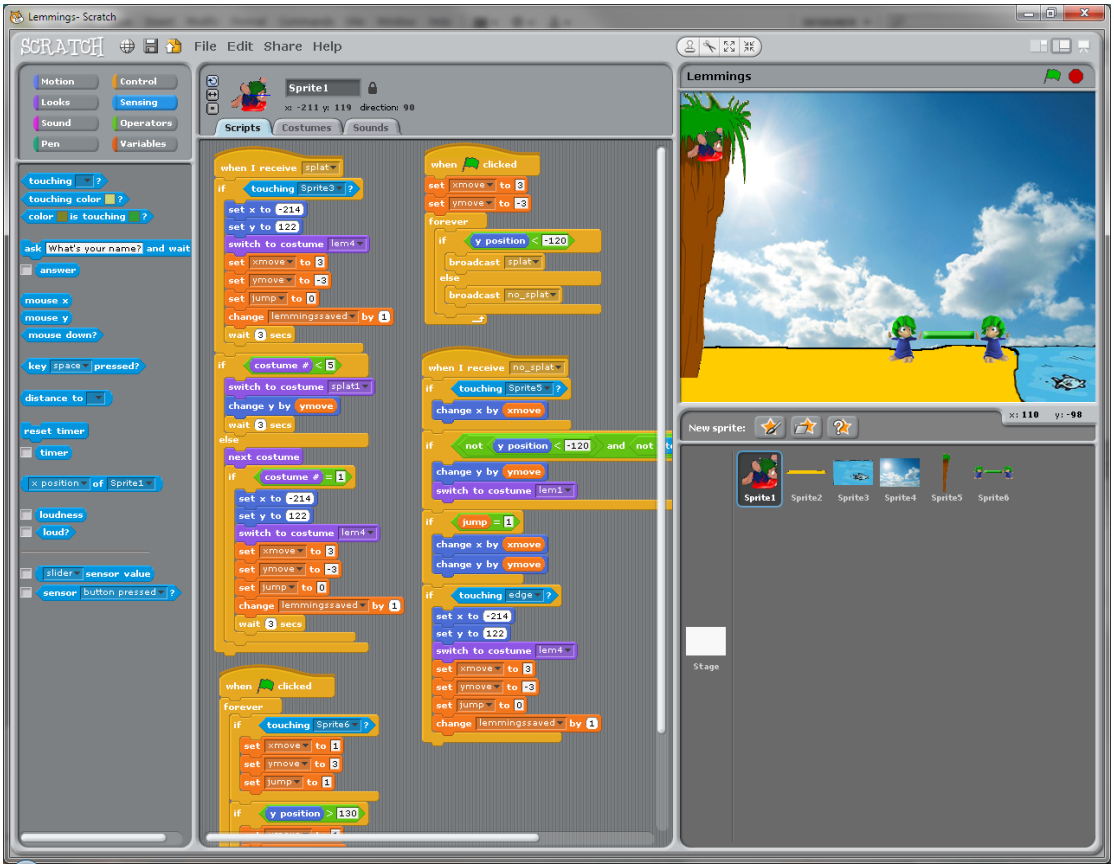
Εικόνα 2.6: Η γλώσσα προγραμματισμού StarLogo TNG. Αναπτύχθηκε ως η εξέλιξη της γλώσσας Logo.



Εικόνα 2.7: Η γλώσσα προγραμματισμού E-Toys



Εικόνα 2.8: Το εργαλεία Alice, που χρησιμοποιείται για προγραμματισμό γραφικών σε Java



Εικόνα 2.9: Η γλώσσα προγραμματισμού Scratch

Προγραμματισμός με γραφικά τουβλάκια (blocks)

Η πιο διαδεδομένη σύγχρονη τεχνική για την δόμηση προγραμμάτων στις VPLs και στα VETL εκπαιδευτικού σκοπού είναι τα blocks. Τα blocks αποτελούν γραφικά τουβλάκια/πλακίδια τα οποία μεταφράζονται σε λειτουργικότητα ή κώδικα.

Openblocks

Μια αρκετά αξιόλογη δουλειά για την δυναμική δημιουργία VETL και VPL είναι το Openblocks.

Το Openblocks[28] αποτελεί ένα πλαίσιο (framework) για την δημιουργία γραφικών γλωσσών προγραμματισμού. Δημιουργήθηκε από το εργαστήριο STEP[29] του MIT και είναι υλοποιημένο σε Java. Είναι ελεύθερο ως προς την χρήση αλλά και την επέκταση του. Δημιουργεί γραφικές γλώσσες προγραμματισμού δυναμικά. Η περιγραφή της γλώσσας γίνεται με την χρήση ενός αρχείου XML με συγκεκριμένους κανόνες περιγραφής. Σε αυτό το αρχείο περιγράφονται οι κατηγορίες Blocks της γλώσσας, τα ίδια Blocks και τα χαρακτηριστικά τους. Οι κανόνες περιγραφής που ακολουθεί το xml αρχείο υπάρχουν σε ένα .dtd αρχείο με όνομα «lang_def.dtd».

Το open blocks έχει ήδη χρησιμοποιηθεί για να δημιουργήσει γραφικές γλώσσες προγραμματισμού. Η γλώσσα StarLogo TNG [30] του MIT και το AppInventor[31] για την δημιουργία εφαρμογών σε Android κινητά της Google είναι οι γνωστότερες εφαρμογές του.

Κεφάλαιο 3
Σχεδιασμός, Υλοποίηση και Περιγραφή του Εργαλείου

Περίληψη

Σε αυτό το κεφάλαιο περιγράφονται οι σχεδιαστικές αποφάσεις και οι τεχνικές λεπτομέρειες που αφορούν την υλοποίηση του εργαλείου Block-C. Επίσης γίνεται μια σύντομη παρουσίαση του εργαλείου και της λειτουργικότητας του. Τέλος γίνεται επεξήγηση της επέκτασης που έγινε πάνω στο openblocks ώστε να ικανοποιηθούν οι απαιτήσεις που τέθηκαν κατά την φάση της σχεδίασης.

Σχεδιαστικές Αποφάσεις

Η βασική επιδίωξη ήταν να παρέχει το Block-C ευχρηστία μέσω μίας ευκατάληπτης γραφικής διεπαφής (Graphic User Interface). Έπρεπε ο χρήστης να μπορεί να χρησιμοποιεί την νοητική λειτουργία της «αναγνώρισης» παρά αυτήν της «ανάκλησης» (recognition over recall [1]) κατά την χρήση του Block-C. Όπως είναι γνωστό από μελέτες[23,1] , είναι πιο εύκολο για τον χρήστη να αναγνωρίσει κάτι που του χρειάζεται παρά να προσπαθήσει να το θυμηθεί (για παράδειγμα να αναγνωρίσει ότι η συνάρτηση *printf* χρησιμοποιείται για εκτύπωση παρά να προσπαθήσει να θυμηθεί ποια συνάρτηση χρησιμοποιείται για εκτύπωση).

Μια εύχρηστη γραφική διεπαφή διευκολύνει αρκετά τους χρήστες με άμεση συνέπεια την αύξηση αποτελεσματικότητα τους. Έτσι προέκυψαν οι εξής σχεδιαστικές αποφάσεις:

1. **Ο χρήστης πρέπει να αλληλεπιδρά μόνο με κατανοητές έννοιες**
   Οι αρχάριοι πρέπει να έρχονται σε επαφή μόνο με έννοιες οι οποίες τους χρησιμεύουν άμεσα. Δεν πρέπει να χρησιμοποιούν μονάδες ή πόρους που δεν μπορούν να αντιληφθούν την χρησιμότητα τους. Ένα παράδειγμα είναι οι βιβλιοθήκες και οι εντολές #include προς τον προ επεξεργαστή. Ο αρχάριος μαθαίνει να τα χρησιμοποιεί χωρίς να ξέρει ακριβώς σε τι του χρειάζονται.
2. **Πρόληψη συντακτικών λαθών**
   Για να ξεπεραστεί η δυσκολία την οποία εισάγουν τα συντακτικά λάθη στην την διαδικασία της μάθησης, αυτά τα λάθη πρέπει να εξαλειφθούν. Έτσι, οι χρήστες πρέπει να καθοδηγούνται για το πώς πρέπει να συντάσσουν τις εκφράσεις της γλώσσας.
3. **Παραγωγή δομημένου κώδικα**
   Οι μαθητευόμενοι πρέπει να μάθουν να γράφουν καλά δομημένο κώδικα, έτσι ώστε να είναι ευανάγνωστος. Άρα, πρέπει η καθοδήγηση που παρέχουν τα block στην οπτική δόμιση του προγράμματος να μεταφράζεται αυτόματα σε παρενθέσεις, αγκύλες, tabs κλπ.
4. **Μεταβατικότητα του εργαλείου**
   Πρέπει να δίνεται η δυνατότητα στους χρήστες να μπορούν σταδιακά να μεταβούν από την χρήση του Block-C στη κλασσική κειμενική C (plain-text C). Με την μετάφραση των γραφικών εντολών (blocks) σε κειμενική C και την οπτική αντιπαραβολή γίνεται ευκολότερη η κατανόηση των εντολών και του τρόπου σύνταξης τους.

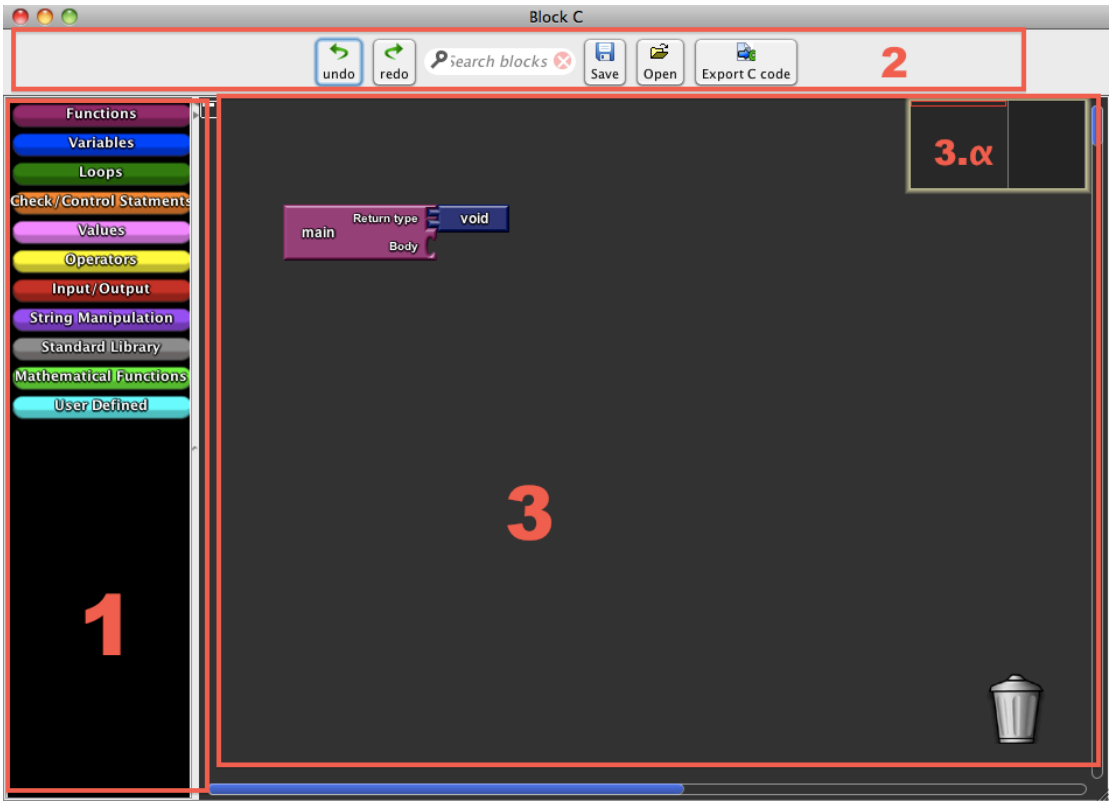
Βασική περιγραφή του Block-C

Το εργαλείο, όπως αυτό προέκυψε στην τελική του μορφή, αποτελείται από τρεις διακριτές περιοχές.

1. Η πρώτη περιοχή είναι η περιοχή που περιέχει τα blocks ομαδοποιημένα σε κατηγορίες.
2. Η δεύτερη περιοχή είναι η περιοχή όπου βρίσκονται τα κουμπιά με την λειτουργικότητα του εργαλείου (αποθήκευση, αναζήτηση κλπ).

3. Η τρίτη και τελευταία περιοχή είναι η περιοχή στην οποία ο χρήστης εναποθέτει τα block έτσι ώστε να δημιουργήσει ένα πρόγραμμα (χώρος εργασίας - workspace).
    3α. Η περιοχή 3.α είναι η περιοχή όπου βρίσκεται ένας χάρτης του χώρου εργασίας που δείχνει σε μικρογραφία όλο αυτόν τον χώρο.

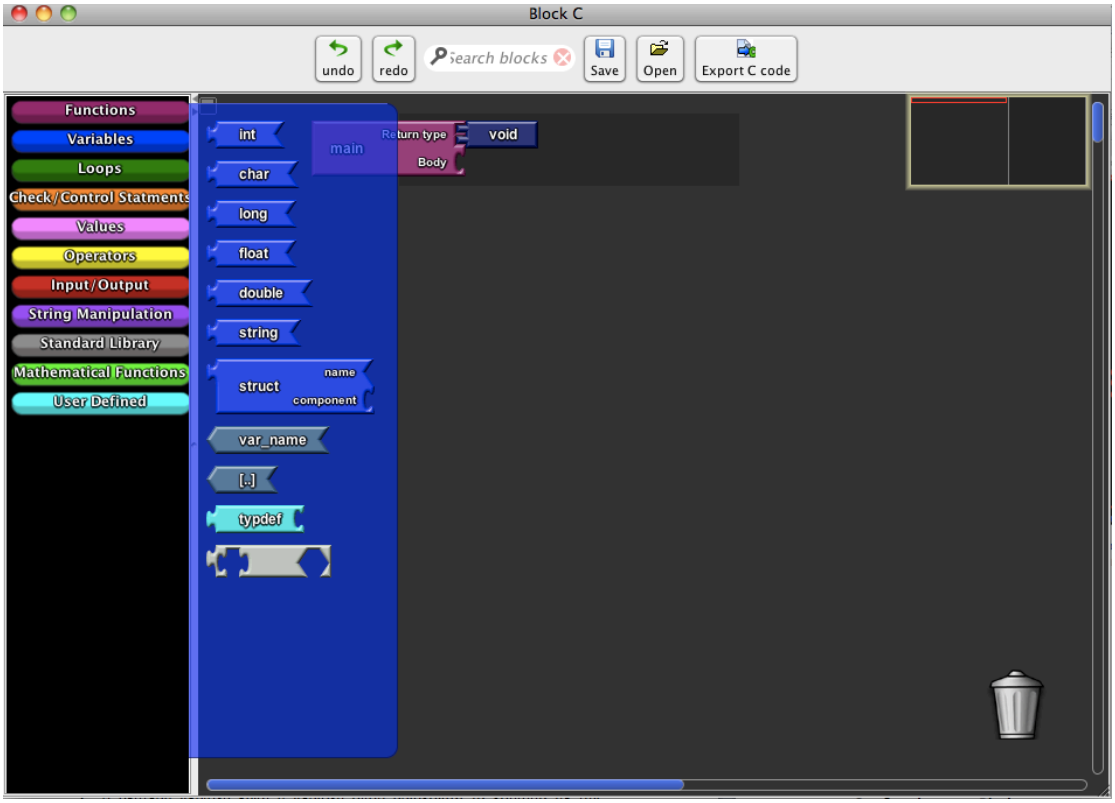
Μια αναλυτική απεικόνιση του γραφικού περιβάλλοντος του εργαλείου φαίνεται στην *Εικόνα 3.1*.



Εικόνα 3.1: Η γραφική διεπαφή του εργαλείου χωρίζεται σε 3 διακριτές περιοχές: 1) Η περιοχή με τα blocks ομαδοποιημένα σε κατηγορίες. 2) Η περιοχή με τα κουμπιά λειτουργικότητας. 3) Ο χώρος εργασίας του χρήστη .

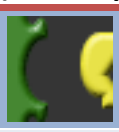
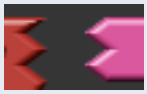
Η *Εικόνα 3.1* δείχνει την αρχική κατάσταση στην οποία βρίσκεται το πρόγραμμα όταν ανοίξει. Αρχικά το πρόγραμμα περιέχει το block της κύριας συνάρτησης main την οποία χρειάζεται ο χρήστης για να ξεκινήσει να συντάσσει ένα πρόγραμμα. Τα γραφικά στοιχεία του χειρίζονται άμεσα μέσω του ποντικιού. Κρατώντας πατημένο το αριστερό κουμπί του ποντικιού του ο χρήστης μπορεί να μετακινήσει οποιοδήποτε block. Αντίστοιχα αφήνοντας το κουμπί αφήνει και το block. Η λογική που λειτουργεί το σύστημα είναι βασισμένη στην λειτουργικότητα drag&drop η οποία είναι ευρέως διαδεδομένη στα περισσότερα γραφικά περιβάλλοντα διαχείρισης αρχείων.

Για να εισάγει ο χρήστης ένα block στον χώρο εργασίας πρέπει να επιλέξει την κατηγορία η οποία το περιέχει και έπειτα να το σύρει στον χώρο εργασίας



Εικόνα 3.2: Η κατηγορία "Variables" αφού την επιλέξει με το ποντίκι του ο χρήστης

Πίνακας 3.3: Πίνακας με την γραφική αναπαράσταση ω συνδέσμων και τις λεπτομέρειες που τους αφορούν

Γραφικός Σύνδεσμος Block	Λεπτομέρειες
	Θηλυκός και αρσενικός ημικυκλικός σύνδεσμος για λογικές εκφράσεις και μονάδες.
	Θηλυκός και αρσενικός puzzle σύνδεσμος για αυτοτελείς εκφράσεις ή μονάδες
	Θηλυκός και αρσενικός τριγωνικός σύνδεσμος για αριθμητικές εκφράσεις και τιμές
	Θηλυκός και αρσενικός διπλός-τριγωνικός σύνδεσμος για τον τύπο δεδομένων των συναρτήσεων printf και scanf
	Θηλυκός και αρσενικός διπλός-τετραγωνικός σύνδεσμος για τον τύπο επιστροφής συνάρτησης
	Θηλυκός και αρσενικός τετραγωνικός σύνδεσμος για τον τρόπο ανοίγματος αρχείου από την συνάρτηση fopen
	Διπλός Θηλυκός τριγωνικός σύνδεσμος για πράξεις αριθμητικές και σύγκρισης

	Block χωρίς θηλυκό (αριστερά) και αρσενικό (δεξιά) σύνδεσμο για block που δεν ακολουθούνται ή δεν συνδέονται με άλλα αντίστοιχα

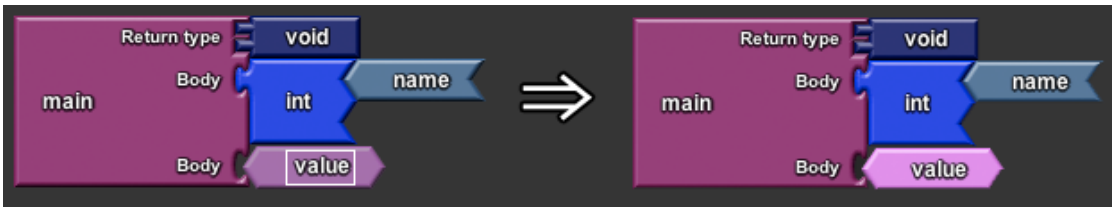
Για να ενώσει δύο block ο χρήστης πρέπει να σύρει και να αφήσει το ένα επάνω από το άλλο έτσι ώστε ο ένας σύνδεσμος να βρίσκεται πάνω από τον άλλο. Βασική προϋπόθεση είναι οι σύνδεσμοι να είναι του ιδίου τύπου και ό ένας να είναι αρσενικός και ο άλλος θηλυκός. Στον Πίνακα 3.3 οι τύποι των συνδέσμων και το την εννοιολογική σημασία με την οποία δημιουργήθηκαν.

Οι σύνδεσμοι αποτελούν το βασικό μηχανισμό αποφυγής των συντακτικών λαθών. Καθοδηγούν τους χρήστες στο να δομούν σωστές εκφράσεις υπενθυμίζοντας του οπτικά ποια block μπορούν να συνδεθούν μεταξύ τους και σε ποιον σημείο (για block με παραπάνω από ένα θηλυκό σύνδεσμο). Επίσης βοηθούν τους χρήστες να δημιουργήσουν τα κατάλληλα νοητικά μοντέλα, για να ξεχωρίσουν και να ομαδοποιήσουν block με βάση την συντακτική τους κατηγορία (πχ. λογικοί τελεστές έναντι αριθμητικών).

Αφού γίνει η σύνδεση αναπαράγεται ένας χαρακτηριστικός ήχος και το block «πατέρας», που περιέχει το block «παιδί», διευρύνεται γραφικά ώστε να το ενσωματώσει. Επίσης αν ο σύνδεσμος του «πατέρα» είναι επεκτεινόμενός σύνδεσμος (όσα block και αν ενωθούν σε αυτόν τον σύνδεσμο πάντα θα υπάρχει θέση και για άλλο block) τότε δημιουργείται μια κενή θέση.

Εικόνα 3.4: Η διαδικασία μια επιτυχούς συνδέσεως. Αριστερά είναι το block "int" το οποίο μεταφέρεται από το ποντίκι πριν απελευθερωθεί και δεξιά αφού απελευθερωθεί το αριστερό κλικ του ποντικιού. Παρατηρούμε ότι το block της main, μετά την σύνδεση, διευρύνθηκε και δημιουργήθηκε ακόμα μία θέση για το "Body"

Αντίστοιχα εάν οι δύο σύνδεσμοι δεν είναι του ίδιου τύπου, το εργαλείο δεν επιτρέπει την σύνδεση. Παράδειγμα ανεπιτυχούς συνδέσεως φαίνεται στην Εικόνα 3.5 .



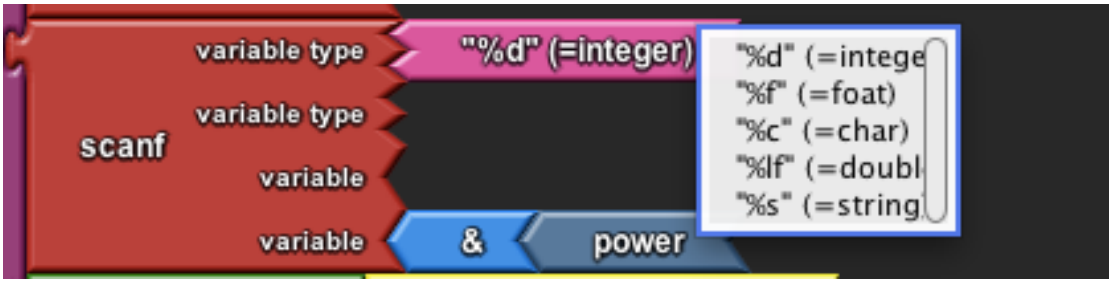
Εικόνα 3.5: Η διαδικασία μια ανεπιτυχούς συνδέσεως. Αριστερά είναι το block "value" το οποίο μεταφέρεται πριν απελευθερωθεί και δεξιά αφού απελευθερωθεί το αριστερό κλικ του ποντικιού. Παρατηρούμε ότι το block της main, μετά την αποτυχία σύνδεσης, δεν διευρύνθηκε.

Όποιο από τα block, που δημιούργησε ο χρήστης, δεν τα χρειάζεται πια μπορεί να τα διαγράψει απλά αφήνοντας τα πάνω στον κάδο που βρίσκεται στο κάτω δεξιά μέρος της οθόνης.

Εικόνα 3.6: Ο τρόπος με τον οποίο ο χρήστης διαγράφει ένα block. Παρατηρούμε ότι ο κάδος ανοίγει σε ένδειξη ότι ανταποκρίνεται στην κίνηση του χρήστη

Για τα block τα οποία μπορούν να έχουν διάφορες προκαθορισμένες τιμές (όπως τα %d, %f, % c κλπ) τα οποία δεν είναι απαραίτητο να έχει αποστηθίσει ο χρήστης το εργαλείο δίνει την δυνατότητα να επιλέξει μέσω μιας dropdown λίστας. Έτσι μπορεί εύκολα να αλλάξει ένα τέτοιο block γρήγορα, χωρίς να χρειαστεί να το αντικαταστήσει, απλά επιλέγοντας από την λίστα όπως φαίνεται στην Εικόνα 3.6 .

Τα block αυτά αφορούν κάποιους προσδιοριστές οι οποίοι χρησιμοποιούνται ως είσοδοι σε συναρτήσεις βιβλιοθηκών. Αυτοί οι προσδιοριστές δίνουν πληροφορία στις συναρτήσεις για το πως θα διαχειριστεί τις υπόλοιπες εισόδους της. Ένα παράδειγμα είναι η συνάρτηση scanf που δέχεται προσδιοριστή για τον τύπο της μεταβλητής που θα διαβάσει από το πληκτρολόγιο.



Εικόνα 3.7: Οι πιθανές τιμές που μπορεί να πάρει το block "%d", όπως αυτές εμφανίζονται στην dropdown λίστα των επιλογών

Ομαδοποίηση των Block

Το openblocks παρέχει την δυνατότητα ομαδοποίησης των blocks σε κατηγορίες. Χρησιμοποιώντας αυτήν τη δυνατότητα οργανώσαμε τα block εντολών του Block-C στις εξής κατηγορίες: «Functions, Variables, Loops, Check/Control Statements, Values, Operators, Input/Output, String Manipulation, Standard Library, and Mathematical Functions, User Defined».

Όλες οι παραπάνω κατηγορίες είναι κωδικοποιημένες χρωματικά, έτσι ώστε η εννοιολογική τους σημασία και η διαφοροποίηση τους να είναι ευδιάκριτη στον χρήστη.

Εικόνα 3.8 : Οι κατηγορίες των blocks

Αρχείο lan_def.dtd που ορίζει τους κανόνες που ακολουθεί το XML αρχείο περιγραφής της γλώσσας

Το αρχείο αυτό, όπως έχει προαναφερθεί, είναι το αρχείο που ορίζει τα tag και τα χαρακτηριστικά τους, τα οποία μπορεί να περιέχει το XML αρχείο που περιγράφει την γλώσσα.

Το αρχείο είναι το εξής:

```
1.<?xml version="1.0" encoding="ISO-8859-1"?>
2.<!--
3.    Document   : lang_def.dtd
4.   Description:
5.        This defines the language and initial workspace setup.-->
6.<!ELEMENT BlockLangDef ( BlockConnectorShapes, BlockGenuses, BlockFamilies?,
7.BlockDrawerSets?, Pages?, TrashCan?, MiniMap?)>
8.
9.<!--This defines a mapping between block connector shape type to number-->
10.<!ELEMENT BlockConnectorShapes (BlockConnectorShape*)>
11.<!ELEMENT BlockConnectorShape EMPTY>
12.<!ATTLIST BlockConnectorShape shape-type CDATA #REQUIRED>
13.<!ATTLIST BlockConnectorShape shape-number CDATA #REQUIRED>
14.
15.<!ELEMENT BlockGenuses (BlockGenus*)>
16.<!--This defines a single block genus-->
```

```
<!ELEMENT BlockGenus (description?, BlockConnectors?,Stubs?, Images?,
LangSpecProperties?)>
<!ATTLIST BlockGenus name CDATA #REQUIRED>
<!ATTLIST BlockGenus initlabel CDATA #REQUIRED>
<!-- the kind of a genus can affect the rendering of a block.  relevant kinds are:
 - command: performs an operation and may take in more than one input
 - data: returns primitive values such as number, string, boolean
 - function: takes in an input and performs an operation to produce an ouput-->
<!ATTLIST BlockGenus kind CDATA #REQUIRED>
<!ATTLIST BlockGenus color CDATA #REQUIRED>
<!ATTLIST BlockGenus editable-label (yes|no) "no">
<!ATTLIST BlockGenus label-unique (yes|no) "no">
<!ATTLIST BlockGenus is-label-value (yes|no) "no">
<!ATTLIST BlockGenus label-prefix CDATA #IMPLIED>
<!ATTLIST BlockGenus label-suffix CDATA #IMPLIED>
<!ATTLIST BlockGenus page-label-enabled (yes|no) "no">
<!--is-starter and is-terminator only apply to blocks of kind: command -->
<!ATTLIST BlockGenus is-starter (yes|no) "no">
<!ATTLIST BlockGenus is-terminator (yes|no) "no">
<!ATTLIST BlockGenus sockets-expandable (yes|no) "no">
<!--
is-creating-variable is for the blocks wich are creating a new variable. e.g. int
x; creaters a variable named x , wich has to be represented into the Variables
list-->
<!ATTLIST BlockGenus is-creating-variable (yes|no) "no">
<!--                B.KYF
    zebra-coding-color is the "alternative" color to be used when to of the same
BlockGenus Blocks are directly connected-->
<!ATTLIST BlockGenus zebra-coding-color CDATA #IMPLIED>
<!--
                            B.KYF
     semicolon-following tell if the <Block>, when "translated" to plain text
code, can be followed semicolon (;) at the ending.
     -depends: when the block is the last in order in a connector it must be
followed by ; -->
<!ATTLIST BlockGenus semicolon-following (allways|never|depends) "depends">
<!--
                            B.KYF
     brackets tell if the <Block>, when "translated" to plain text code, has { }
who can enclose code-->
<!ATTLIST BlockGenus brackets (yes|no) "no">
<!--
                            B.KYF
     parentheses tell if the <Block>, when "translated" to plain text code, has
( ) who can enclose code-->
<!ATTLIST BlockGenus parentheses (yes|no) "no">
<!--
                            B.KYF
     translated-to-code tell if the <Block>, when "translated" to plain text code,
what code will be produce. By default is "label" so the label is the code-->
<!ATTLIST BlockGenus code-translation  CDATA "label">
<!--               B.KYF
is-into-brackets tells by what symbol the intoParenthesis arguments is seperated-->
<!ATTLIST BlockGenus parenthesis-arg-seperator CDATA ",">
<!--               B.KYF
is-into-brackets tells by what symbol the intoParenthesis arguments is seperated-->
<!ATTLIST BlockGenus needs-library CDATA #IMPLIED>
<!--This defines a block description and the description of its block arguments-->
<!ELEMENT arg EMPTY>
<!ATTLIST arg n CDATA #REQUIRED name CDATA #IMPLIED>
<!ELEMENT description (text, arg-description*)>
<!ELEMENT text (#PCDATA|note|em|i|br|arg)*>
<!ELEMENT arg-description (#PCDATA)>
<!ATTLIST arg-description n CDATA #REQUIRED name CDATA #REQUIRED>
<!ELEMENT note (#PCDATA|arg|i)*>
```

```
<!ELEMENT em (#PCDATA)>
<!ELEMENT i (#PCDATA)>
<!ELEMENT br (#PCDATA)>
<!--BlockConnectors are where blocks get connected-->
<!ELEMENT BlockConnectors (BlockConnector*)>
<!ELEMENT BlockConnector (DefaultArg?)>
<!ATTLIST BlockConnector label CDATA #IMPLIED>
<!ATTLIST BlockConnector label-editable (yes|no) "no">
<!-- Order matters with socket connectors and at most one plug is allowed (no
multiple return types) -->
<!ATTLIST BlockConnector connector-kind (plug|socket) #REQUIRED>
<!-- for connector-type use the shape-type values specified in block connectors-->
<!ATTLIST BlockConnector connector-type CDATA #REQUIRED>
<!ATTLIST BlockConnector position-type (single|mirror|bottom) "single">
<!ATTLIST BlockConnector is-expandable (yes|no) "no">
<!ATTLIST BlockConnector expand-group CDATA #REQUIRED>
<!--               B.KYF
    is-creating-variable-type is used for commands like TYPEDEF which are creating
a variable type. e.g. IF i typedef a struct as node the struct will be now on,also,
refered as node -->
<!ATTLIST BlockConnector is-creating-variable-type (yes|no) "no">
<!--                       B.KYF
is-into-parentheses tells if the Block is enclosed code in () of the before block-
->
<!ATTLIST BlockConnector is-into-parentheses (yes|no) "no">
<!--                       B.KYF
is-into-brackets tells if the Block is enclosed code in the {} of the before
block-->
<!ATTLIST BlockConnector is-into-brackets (yes|no) "no">
<!ELEMENT DefaultArg EMPTY>
<!ATTLIST DefaultArg genus-name CDATA #REQUIRED>
<!ATTLIST DefaultArg label CDATA #IMPLIED>

<!ELEMENT Stubs (Stub*)>
<!--This defines a stub of a block, so that the block can exist as a single entity
and have mini-references to it-->
<!ELEMENT Stub  (LangSpecProperties)>
<!ATTLIST Stub scope CDATA #IMPLIED>
<!ATTLIST Stub stub-genus (getter|setter|caller|agent|inc) #REQUIRED>
<!-- Defines the images that are drawn on the block itself.
     Note: For now, only one image is enabled and wrap-text and image-editable
have no effect. Note: make sure FileLocation specified is relative to workspace
directory -->
<!ELEMENT Images (Image)>
<!ELEMENT Image (FileLocation)>
<!ATTLIST Image wrap-text (yes|no) "no">
<!ATTLIST Image image-editable (yes|no) "no">
<!ATTLIST Image block-location
(center|east|west|north|south|southeast|southwest|northeast|northwest) "center">
<!ATTLIST Image width CDATA #IMPLIED>
<!ATTLIST Image height CDATA #IMPLIED>
<!ELEMENT FileLocation (#PCDATA)>

<!ELEMENT LangSpecProperties (LangSpecProperty*)>
<!ELEMENT LangSpecProperty (#PCDATA)>
<!ATTLIST LangSpecProperty key CDATA #REQUIRED>
<!ATTLIST LangSpecProperty value CDATA #REQUIRED>

<!--This defines a BlockGenus Family-->
<!ELEMENT BlockFamilies (BlockFamily*)>
<!ELEMENT BlockFamily (FamilyMember*)>
<!--       B.KYF
family-name is the name of the family it is used to know where to add new block-->
<!ATTLIST BlockFamily family-name CDATA #REQUIRED>
<!ELEMENT FamilyMember (#PCDATA)>
```

```
146.
147.<!-- Defines BlockDrawerSets and their Block Drawer content-->
148.<!ELEMENT BlockDrawerSets (BlockDrawerSet*)>
149.<!ELEMENT BlockDrawerSet (BlockDrawer*)>
150.<!ATTLIST BlockDrawerSet type (bar|stack) "bar">
151.<!ATTLIST BlockDrawerSet name CDATA #REQUIRED>
152.<!ATTLIST BlockDrawerSet location
153.(east|west|north|south|northeast|southeast|southwest|northwest) "west">
154.<!-- window-per-drawer specifies if each drawer should be its own draggable
155.window.  otherwise, all the drawers are contained within one draggable window and
156.only one drawer can be opened at once. Whether or not the window is draggable
157.depends if drawer-draggable is set to "yes." -->
158.<!ATTLIST BlockDrawerSet window-per-drawer (yes|no) "yes">
159.<!ATTLIST BlockDrawerSet drawer-draggable (yes|no) "yes">
160.<!-- the width of all the drawers within this set -->
161.<!ATTLIST BlockDrawerSet width CDATA #IMPLIED>
162.<!--This defines BlockDrawers and their content-->
163.<!ELEMENT BlockDrawer (description?  , (BlockGenusMember | Separator |
164.NextLine)* )>
165.<!ATTLIST BlockDrawer name CDATA #REQUIRED>
166.<!ATTLIST BlockDrawer type (default|factory|page|custom) "default">
167.<!ATTLIST BlockDrawer is-open (yes|no) "no">
168.<!ATTLIST BlockDrawer button-color CDATA #REQUIRED>
170.<!-- B.KYF
171.the description and the text <ELEMENTS>.Used for the same purpose as the
172.<BlockGenus> Description and text! -->
173.<!ELEMENT description (text)>
174.<!ELEMENT text (#PCDATA)>
175.
176.<!ELEMENT BlockGenusMember (#PCDATA)>
177.<!ELEMENT Separator EMPTY>
178.<!ELEMENT NextLine EMPTY>
179.<!-- Defines Pages dividing the Block Canvas and the optional PageDrawers
180.associated with them. Each Page can have only one PageDrawer. For now, every page
181.must have a drawer or no pages can have drawers. The block canvas need not contain
182.any pages.  You may choose to have a blank canvas instead of a canvas of pages.-->
183.<!ELEMENT Pages (Page*)>
184.<!-- drawer-with-page auto generates a new drawer for each new page created by a
185.user and creates an empty drawer for each page that does not specify a page drawer
186.-->
187.<!ATTLIST Pages drawer-with-page (yes|no) "no">
188.<!ELEMENT Page (PageDrawer?)>
189.<!ATTLIST Page page-name CDATA #REQUIRED>
190.<!ATTLIST Page page-width CDATA #REQUIRED>
191.<!ATTLIST Page page-drawer CDATA #IMPLIED>
192.<!ATTLIST Page page-color CDATA #IMPLIED>
193.<!ATTLIST Page page-shape CDATA #IMPLIED>
194.
195.<!ELEMENT PageDrawer (BlockGenusMember*)>
196.<!-- If specified a trash can will appear on the workspace. For both of its child
197.elements, a location for the images should be specified relative to the working
198.directory.The open trash image appears when a user drags a block over the
199.trashcan.  The closed trash image is the default image during steady state. -->
200.<!ELEMENT TrashCan (OpenTrashImage, ClosedTrashImage)>
201.<!ELEMENT OpenTrashImage (#PCDATA)>
202.<!ELEMENT ClosedTrashImage (#PCDATA)>
203.
204.<!-- By default, a minimap will always appear in the upper right corner of the
205.block canvas, unless enabled is set to "no." -->
206.<!ELEMENT MiniMap EMPTY>
207.<!ATTLIST MiniMap enabled (yes|no) "yes">
208.<!-- By default, typeblocking will be enabled, such that when the user types onto
209.the canvas blocks will fly out that match the entered text.-->
210.<!ELEMENT Typeblocking EMPTY>
211.<!ATTLIST Typeblocking enabled (yes|no) "yes">
```

Όπως φαίνεται στο land_def.dtd τα tag τα οποία μπορεί να χρησιμοποιήσει ο χρήστης για να δημιουργήσει μια γλώσσα είναι τα εξής : «BlockLangDef, BlockConnectorShapes, BlockConnectorShape, BlockGenuses, BlockGenus, arg, description, text, note, em, i, br, BlockConnectors, BlockConnector, DefaultArg, Stubs, Stub , Images, Image, FileLocation, LangSpecProperties, LangSpecProperty, BlockFamilies, BlockFamily, FamilyMember, BlockDrawerSets, BlockDrawerSet, BlockDrawer, description, text, BlockGenusMember, Separator, NextLine, Pages, Page, PageDrawer, TrashCan, OpenTrashImage, ClosedTrashImage, MiniMap, Typeblocking»

1. Το tag BlockLangDef (γραμμές 6-7) είναι το βασικό tag μέσα στο οποίο δομείται όλο το υπόλοιπο αρχείο περιγραφής της γλώσσας
2. Το tag *BlockConnectorShapes* (γραμμή 10) περιέχει όλους τους τύπους των γραφικών συνδέσμων που μπορούν να έχουν τα blocks.
    i. Το BlockConnectorShape (γραμμές 11-13) δημιουργεί έναν συγκεκριμένο σύνδεσμο
3. Το tag BlockGenuses (γραμμή 15) είναι το tag μέσα στο οποίο περιέχεται η περιγραφή όλων των block.
    i. Το BlockGenus (γραμμές 17-72) περιγράφει ένα συγκεκριμένο «γένος» block το οποίο θα δημιουργεί στιγμιότυπα ενός block.
    ii. Το arg (γραμμές 74-75) είναι το tag το οποίο περιέχει πληροφορίες για τα ορίσματα του block
    iii. Το description (γραμμή 76) χρησιμοποιείται για την βοηθητική περιγραφή που θα εμφανίζεται στον χρήστη
        a) Το tag text (γραμμές 77) είναι το κείμενο που θα περιέχει το μήνυμα περιγραφής
            1. Τα note, em, i, br (γραμμές 80-83) είναι βοηθητικά για το πως θα εμφανίζεται το παραπάνω μήνυμα
    iv. Το tag BlockConnectors (γραμμή 85) περιέχει όλους τους συνδέσμους που θα έχει το block
        a) Το BlockConnector (γραμμές 86-109) δημιουργεί έναν συγκεκριμένο σύνδεσμο στο block στο οποίο περιέχεται.
            1. DefaultArg (γραμμές 110-112) δηλώνει ποιο θα είναι το προκαθορισμένο block το οποίο συνδέεται σε αυτόν τον σύνδεσμο.
            2. Το Stubs (γραμμή 114) περιέχει όλα τα στελέχη ενός block.
                i. Το Stub (γραμμές 117-119) ορίζει ένα στέλεχος ενός μπλοκ, έτσι ώστε το block μπορεί να υφίσταται ως μία ενιαία οντότητα και να υπάρχουν μίνι-αναφορές σ 'αυτό
            3. Το Images (γραμμή 124) περιέχει εικόνες
                i. Το Image (γραμμές 125-131) περιέχει μια εικόνα η οποία θα φαίνεται μέσα σε ένα block
                ii. Το FileLocation (γραμμή 132) είναι η τοποθεσία της εικόνας
    v. Το LangSpecProperties (γραμμή 134) περιέχει LangSpecProperty

a) Το LangSpecProperty (γραμμές 135-137) αποθηκεύει κάποιες πληροφορίες οι οποίες μπορεί να είναι χρήσιμες για την γλώσσα

4. Το BlockFamilies (γραμμή 140) είναι το tag το οποίο περιέχει μέσα του τις οικογένειες των block όπως αυτές ορίζονται από τον χρήστη.
   i. Μία BlockFamiliy (γραμμές 141-144) είναι μια οικογένεια όμοιων block τα οποία μπορούν να εναλλάσσονται μεταξύ τους από το γραφικό περιβάλλον αν το επιλέξει ο χρήστης (πχ. “%d” και “%f”)
      a) Το FamilyMember (γραμμή 145) είναι ένα block που ανήκει σε μια οικογένεια
5. BlockDrawerSets (γραμμή 148) περιέχει όλα και το BlockDrawerSet (γραμμές 149-161) ένα γραφικό «συρτάρι» το οποίο αποτελεί την κατηγορία με τα ομαδοποιημένα block (Εικόνα 3.7)
   i. BlockDrawer (γραμμές 163 -168) είναι ένα συγκεκριμένο «συρτάρι»
      a) Το description (γραμμή 173) περιέχει ένα κείμενο με μια περιγραφή
         1. Text (γραμμή 174) είναι το κείμενο της περιγραφής
      b) BlockGenusMember (γραμμή 176) είναι ένα BlockGenus (γραμμές 17-72) το οποίο θα εμφανίζεται μέσα σε αυτήν την κατηγορία
      c) Τα NextLine και Separator (γραμμές 177-178) είναι βοηθητικά για την εμφάνιση tag
6. Το Pages (γραμμές 183-187) περιέχει τις σελίδες του workspace
   i. Το Page (γραμμές 188-193) δημιουργεί μία συγκεκριμένη σελίδα
   ii. Το PageDrawer (γραμμή 195) δέχεται BlockGenusMember (γραμμή 176) αν η σελίδα έχει δικά τις BlockDrawer (γραμμές 163-168)
7. Το TrashCan (γραμμή 200) περιέχει τις πληροφορίες για τον κάδο που μπορεί να «πετάξει» ο χρήστης τα blocks για να διαγραφούν.
   i. Στο OpenTrashImage (γραμμή 201) δηλώνεται η εικόνα που θα εμφανίζεται όταν είναι ανοιχτός ο κάδος
   ii. Στο ClosedTrashImage (γραμμή 202) δηλώνεται η εικόνα που θα εμφανίζεται όταν είναι κλειστός ο κάδος
8. Το MiniMap (γραμμές 206-207) έχει πληροφορίες για τον χάρτη του workspace
9. Το Typeblocking (γραμμές 210-211) όταν ενεργοποιείται εισάγει στο workspace block όταν ο χρήστης πληκτρολογεί κείμενο που ταιριάζει με αυτά

Τα χαρακτηριστικά για τα παραπάνω tag που δημιουργήθηκαν ώς επέκταση του Openblocks παρουσιάζονται αναλυτικά παρακάτω.

Επέκταση του Openblocks

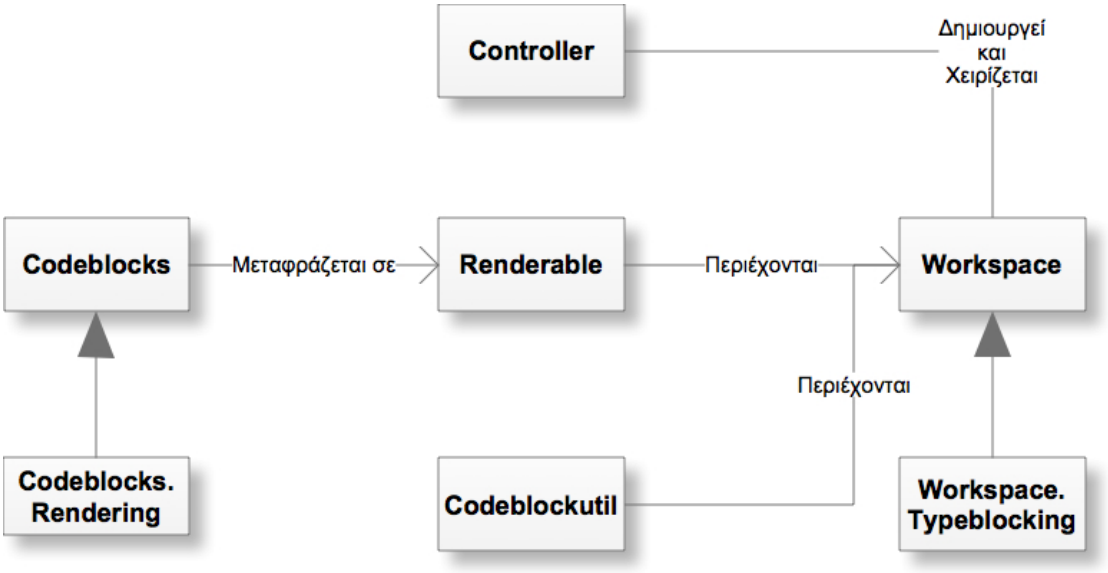
Η βασική λειτουργικότητα που παρέχει το openblocks δεν έφτανε για να καλύψει τις όλες τις αποφάσεις που λήφθηκαν πριν την υλοποίηση. Έτσι έπρεπε να επεκταθεί ενσωματώνοντας επιπρόσθετη λειτουργικότητα.

Η επέκταση αυτή έγινε με τέτοιο τρόπο ώστε να μην αλλάξει το αρχικό Openblocks αλλά να ενσωματώσει περισσότερη λειτουργικότητα. Αυτό έγινε με γνώμονα να μπορεί και το Block-C να επεκταθεί περαιτέρω ή να προσαρμοστεί.

Το openblocks αποτελείται από 7 πακέτα κλάσεων τα οποία αθροιστικά αποτελούνται από 133 αρχεία java. Αυτά τα πακέτα είναι τα εξής:

1. **Controller**: Περιέχει όλα τα βασικά αρχεία για την δημιουργία του γραφικού παραθύρου του εργαλείο, το οποίο περιέχει τα επιμέρους γραφικά στοιχεία.
2. **Codeblocks**: Περιέχει τα αρχεία τα οποία μοντελοποιούν σε java όλες εκείνες τις οντότητες και τις μεταξύ τους εξαρτήσεις και σχέσεις που περιγράφουν την γλώσσα. Αυτές οι οντότητες ορίζονται στο XML αρχείο που περιγράφει την γλώσσα.
3. **Codeblocks.Rendering**: Περιέχει πληροφορίες για τις λεπτομέρειες της γραφικής σχεδίασης των γραφικών μονάδων της γλώσσας (blocks)
4. **Workspace**: Ένα σύνολο από αρχεία για την ομαδοποίηση των block σε γραφικές μονάδες και (όπως γραφικά συρτάρια, βλέπε *Εικόνα 3.8*) και τις επιμέρους βοηθητικές γραφικές μονάδες που περιέχονται στις δύο κύριες περιοχές αλληλεπίδρασης του προγράμματος (1 και 3 από *Εικόνα 3.1*).
5. **Workspace.Typeblocking**: Τα αρχεία που μοντελοποιούν την κίνηση, τον τρόπο σύνδεσης και άλλες παρόμοιες διαδικασίες των γραφικών στοιχείων.
6. **Renderable**: Περιέχει τα αρχεία για την δημιουργία της γραφικής αναπαράστασης όλων των βασικών μονάδων του εργαλείου, όπως αυτά περιγράφονται στα αρχεία του πακέτου *Codeblocks,* με τα οποία αλληλεπιδρά ο χρήστης .
7. **Codeblockutil**: Περιέχει τα αρχεία για τα γραφικά στοιχεία λειτουργικότητας του εργαλείου και όχι της γλώσσας (όπως κουμπιά save, load)

Στο σχήμα 3.9 παρουσιάζεται περιληπτικά η αρχιτεκτονική του openblocks (κατά συνέπεια και του Block-C) βάση των πακέτων java και των σχέσεων μεταξύ τους.



Διάγραμμα 3.9: Η αρχιτεκτονική του openblocks βάση των πακέτων που περιέχουν αρχεία java και η μεταξύ τους σχέσεις.

Τα νέα χαρακτηριστικά που εισήχθησαν είναι δυναμικά προσαρμοζόμενα. Συνολικά έγιναν παρεμβάσεις (όπως δημιουργία συναρτήσεων, προσθήκη μεταβλητών, επέκταση συναρτήσεων, αλλαγή παραμέτρων, εισαγωγή listeners κλπ) σε 116 σημεία σε 20 διαφορετικά αρχεία του openblocks. Αυτά τα χαρακτηριστικά αφορούν τις ιδιότητες των blocks αλλά και την επιπλέον λειτουργικότητα που σχετίζεται με τα blocks ή την δημιουργία ενός προγράμματος.

Συγκεκριμένα, η πρόσθετη λειτουργικότητα που αναπτύχθηκε αφορά τα εξής:

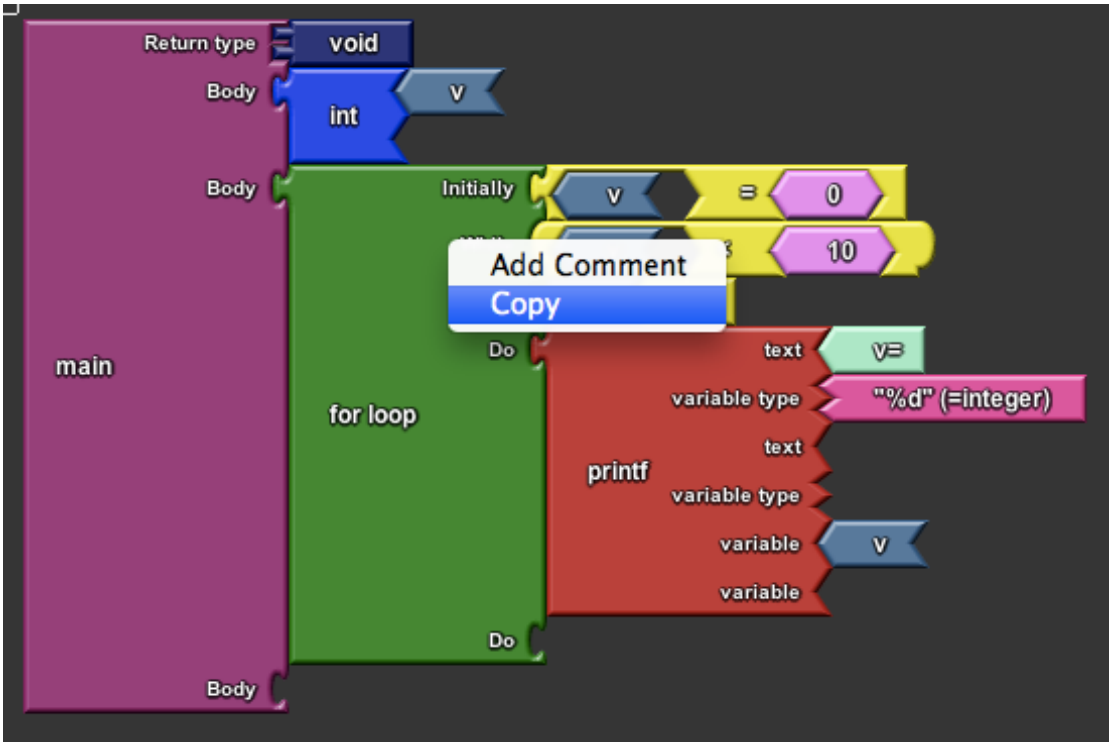
1. Δυνατότητα αντιγραφής blocks
2. Εναλλαγή χρωμάτων εμφωλευμένων block ιδίου τύπου (Zebra Coding)
3. Ειδική κατηγορία για τα block των δημιουργημένων από τον χρήστη μεταβλητών και συναρτήσεων (Κατηγορία «User Defined»)
4. Εξαγωγή του κώδικα που αναπαριστούν τα blocks σε C

Πιο συγκεκριμένα προσπαθήσαμε να αναθέσουμε την παραμετροποίηση και την δυνατότητα δημιουργίας σύνθετων block στον xml αρχείο το οποίο περιγράφει την γλώσσα. Έτσι δεν χρειάζεται η επέμβαση σε κώδικα για τον εμπλουτισμό του Block-C με καινούρια block.

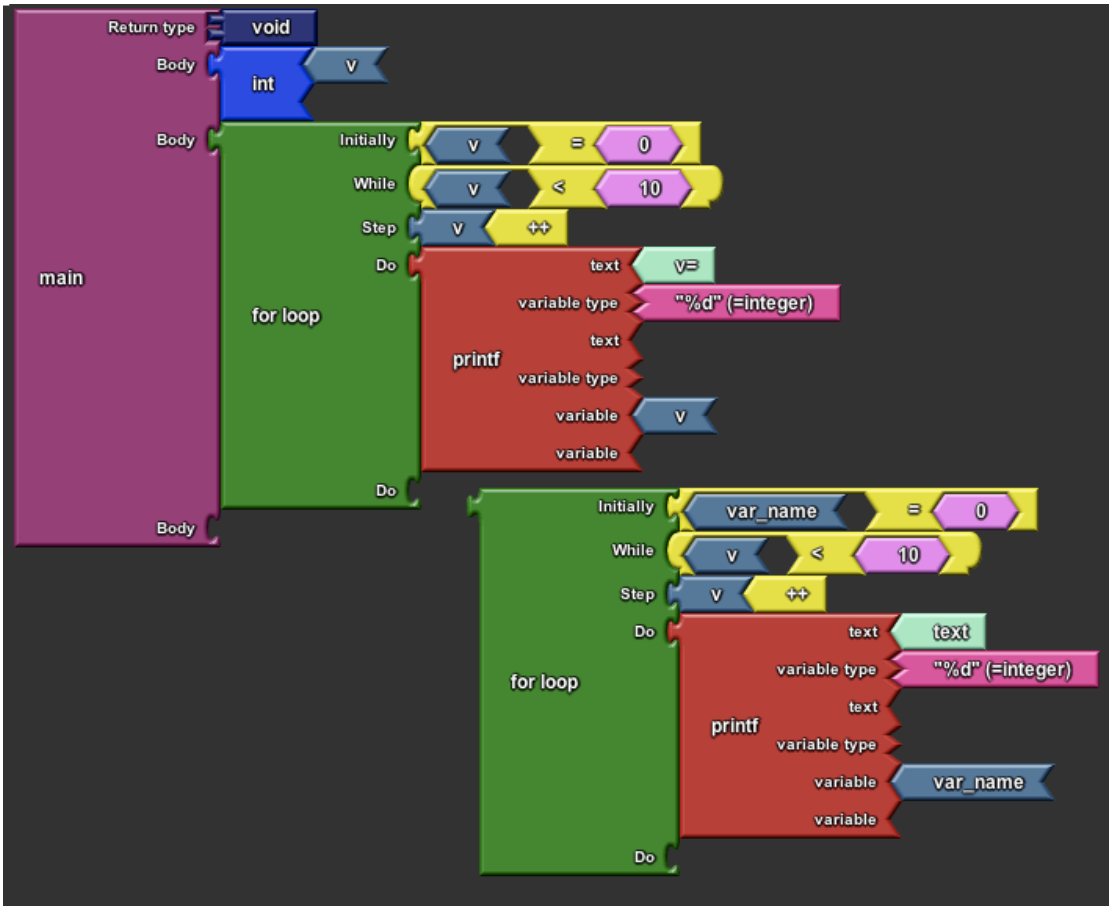
Αντιγραφή blocks

Όπως θα αναφερθεί και σε επόμενο κεφάλαιο, ενσωματώθηκε λειτουργικότητα για την αντιγραφή των block, έτσι ώστε να δίνεται η δυνατότητα επαναχρησιμοποίησης κώδικα. Το παρόν χαρακτηριστικό προτάθηκε από τους χρήστες της αξιολόγησης.

Έτσι ο χρήστης με δεξί κλικ εμφανίσει ένα αναδυόμενο μενού και από εκεί μπορεί να επιλέξει «copy», ώστε να αντιγράψει οποιοδήποτε block μαζί με τα συνδεδεμένα σε αυτό blocks. Εικόνες 3.10 και 3.11



Εικόνα 3.10: Πριν την Αντιγραφή



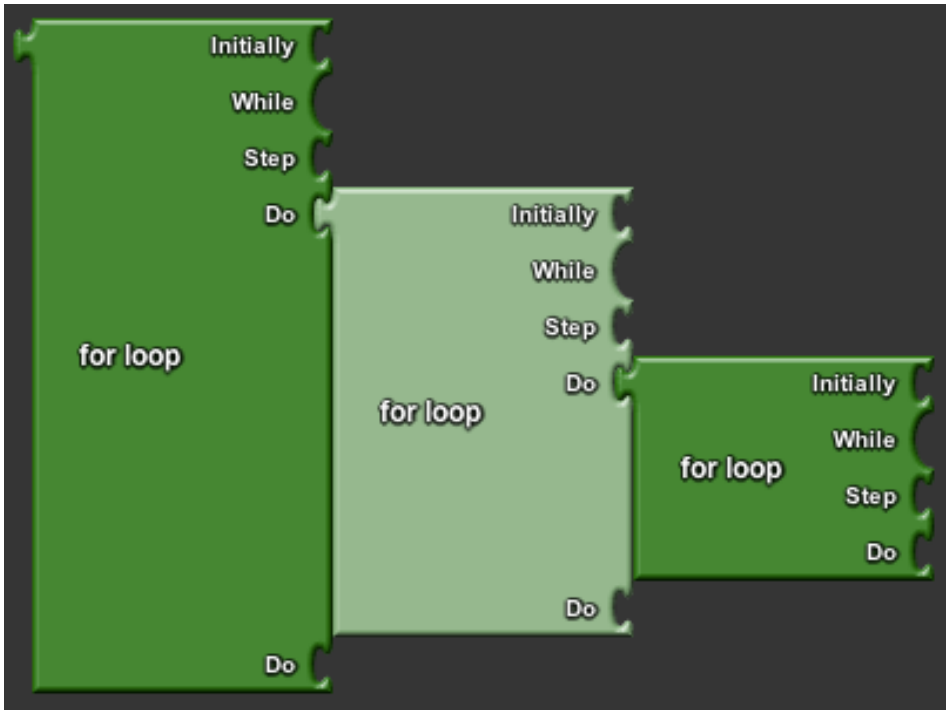
Εικόνα 3.11: Μετά την αντιγραφή

Zebra Coding

Επειδή παρατηρήθηκε δυσκολία στην αναγνώριση εμφωλευμένων block ιδίου τύπου λόγω του κοινού τους χρώματος, αποφασίστηκε να εναλλάσσονται τα χρώματά τους. Αυτό γίνεται αυτόματα από το πρόγραμμα και ο συντάκτης του xml αρχείου που περιγράφει τα blocks της γλώσσας μπορεί να διαλέξει ποιο θα είναι το αρχικό χρώμα και ποιο το χρώμα εναλλαγής.

Για να επιλέξει ο συντάκτης της γλώσσας ποιο block ακολουθεί την *κωδικοποίηση ζέβρας* αρκεί να εισάγει στο xml αρχείο που περιγράφει την γλώσσα στο tag ***BlockGenus*** (είναι το tag που δημιουργεί ένα καινούριο block το οποίο μπαίνει σε μια κατηγορία από τις προκαθορισμένες στα αριστερά και «γεννά» block τα οποία μπαίνουν στο workspace) το χαρακτηριστικό ***zebra-coding-color*** και να του δώσει τρεις τιμές για το κόκκινο το πράσινο και το μπλε αντίστοιχα.

Έτσι κατά την σύνδεση όμοιων block το πρόγραμμα αλλάζει το χρώμα του εμφωλευμένου block αν αυτό είναι ίδιο με το εξωτερικό. Ένα παράδειγμα της χρήσης του Zebra-Coding φαίνεται στην Εικόνα 3.12 .



Εικόνα 3.12: Παράδειγμα της χρωματικής κωδικοποίησης Zebra-Coding

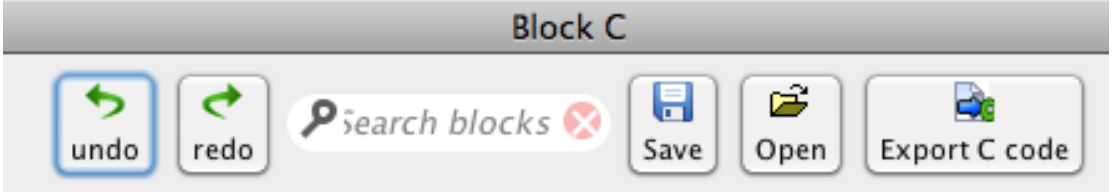
Το tag ***BlockGenus*** για το *for loop* (το οποίο φαίνεται στην εικόνα 3.12) είναι το εξής:

```
<BlockGenus name="for" kind="variable" color="2 128 2"
initlabel="for loop" zebra-coding-color="129 191 128">
```

Ήδη υπάρχουσα λειτουργικότητα που αφορά την διαχείριση του project του χρήστη

Εν συνεχεία, εισάγαμε λειτουργικότητα που αφορά την αποθήκευση και την ανάκτηση ενός προγράμματος αλλά και κουμπιά για undo και redo (για τα οποία υπήρχε πρόβλεψη στο Openblocks αλλά δεν ήταν ενεργοποιημένη η σχετική λειτουργικότητα). Το Openblocks παρείχε την δυνατότητα για την αναζήτηση blocks και κατηγοριών η οποία αξιοποιήθηκε στο Block-C.

Όλα τα κουμπιά και το πεδίο αναζήτησης που παρέχουν την προαναφερθείσα λειτουργικότητα φαίνονται στην Εικόνα 3.13



Εικόνα 3.13: Τα κουμπιά και το πεδίο αναζήτησης που παρέχουν λειτουργικότητα για undo/redo, αναζήτηση, αποθήκευση, άνοιγμα και εξαγωγή κώδικα.

Εξαγωγή Κώδικα σε C

Όπως αναφέρθηκε και στην ενότητα των σχεδιαστικών αποφάσεων αυτού του κεφαλαίου, το Block-C όντας μεταβατικό εργαλείο έπρεπε να δίνει την δυνατότητα στον χρήστη να δει πως μεταφράζονται τα block του προγράμματος που δημιούργησε σε κώδικα C.

Η αντίστοιχη λειτουργικότητα προστέθηκε σε μορφή συνάρτησης της java η οποία καλείται όταν πατηθεί το αντίστοιχο κουμπί.

Ο κώδικας που αναπαριστά ένα block μπορεί να παραμετροποιηθεί μέσω του xml αρχείου που ορίζει την γλώσσα. Ιδιαίτερα για τα block τα οποία περικλείουν άλλα block μπορούν να δηλώσουν μια σειρά από χαρακτηριστικά όχι μόνο στα tag του ***BlockGenus*** αλλά και στα tags των ***Connectors*** (συνδέσμων).

Τα χαρακτηριστικά αυτά που περιγράφουν το τρόπο που μεταφράζεται ένα block σε κώδικα παρατίθενται παρακάτω.

Τα χαρακτηριστικά που περιγράφουν ένα ***BlockGenus*** είναι τα εξής:

- semicolon-following το οποίο μπορεί να πάρει τις τιμές always, never και depends. Το συγκεκριμένο χαρακτηριστικό δίνει πληροφορία για το αν η μετάφραση του block σε κώδικα περιλαμβάνει «;» στο τέλος. Η default τιμή για αυτό το χαρακτηριστικό είναι το depends.
    - Η τιμή *always* σημαίνει ότι πάντα ο κώδικας του block ακολουθείται από ερωτηματικό (όπως πχ. η κλήση μιας void συνάρτησης)
    - Η τιμή *never* σημαίνει ότι ο κώδικας του block δεν ακολουθείται ποτέ από ερωτηματικό (όπως πχ. το block για το for loop)
    - Η τιμή *depends* σημαίνει ότι αν το block είναι το τελευταίο στην σειρά (δεν έχει κάτι άλλο συνδεδεμένο στα δεξιά του) ο κώδικας του ακολουθείται ποτέ από ερωτηματικό αλλιώς όχι (όπως πχ. το block μιας μεταβλητής αν είναι τελευταίο στην σειρά σημαίνει ότι πρέπει να ακολουθηθεί από ερωτηματικό)

- code-translation το οποίο δίνει πληροφορία για το ποια είναι η κειμενική μετάφραση του block σε κώδικα C. Η default τιμή είναι το label το οποίο σημαίνει ότι η ετικέτα που αναγράφεται στο block (το όνομα που φαίνεται στον χρήστη) είναι και η μετάφραση του σε κώδικα.
- needs-library είναι ένα χαρακτηριστικό το οποίο δίνει πληροφορία για το αν και ποια βιβλιοθήκη χρειάζεται για να εισαχθεί το block στο πρόγραμμα (πχ. το block *stpcpy* χρειάζεται την βιβλιοθήκη *string.h*).
- parenthesis-arg-seperator το οποίο χρησιμεύει για να ενημερώσει το πρόγραμμα ποιο είναι διαχωριστικό των ορισμάτων που είναι μέσα στην παρένθεση (πχ. στο for loop το διαχωριστικό είναι το «;»). Μπορεί να πάρει οποιαδήποτε τιμή με default το «,» .

Τα χαρακτηριστικά που περιγράφουν ένα ***BlockConnector*** (σύνδεσμος που υπάρχει μέσα σε ένα block) είναι τα εξής:

- is-into-parentheses είναι το χαρακτηριστικό εκείνο που δηλώνει ότι οποιοδήποτε συνδεθεί στον παρόν σύνδεσμο πρόκειται να είναι μέσα στην παρένθεση. Μπορεί να πάρει τις τιμές *yes* και *no* με default τιμή το *no*.
- is-into-brackets είναι το χαρακτηριστικό εκείνο που δηλώνει ότι οποιοδήποτε συνδεθεί στον παρόν σύνδεσμο πρόκειται να είναι μέσα στην αγκύλη. Μπορεί να πάρει τις τιμές *yes* και *no* με default τιμή το *no*.

Η διαδικασία με την οποία το εργαλείο μεταφράζει το γραφικό πρόγραμμά σε μια συμβολοσειρά με τον C κώδικα, γίνεται μέσω μιας συνάρτησης που ονομάζεται ***exportCodeFromBlocks***.

Η συνάρτηση αυτή είναι μια αναδρομική συνάρτηση η οποία δομεί την συμβολοσειρά από την αρχή προς το τέλος με έναν σειριακό τρόπο. Ουσιαστικά δομεί ένα δέντρο για κάθε ένα από τα block των συναρτήσεων που έχει φτιάξει ο χρήστης (κατ' ελάχιστο για την main). Ως κόμβος βάση/ρίζα (root) ορίζεται το block της εκάστοτε συνάρτησης. Για κάθε ένα από τα συνδεδεμένα σε αυτό block παράγεται ένα υποδέντρο. Σε αυτό το υποδέντρο ρίζα είναι το block το οποίο είναι συνδεδεμένο στην συνάρτηση και κάτω από αυτό δημιουργείται ένα υποδέντρο για κάθε ένα από τα συνδεδεμένα σε αυτό blocks.

Τα υποδέντρα οποιουδήποτε κόμβου δομούνται σειριακά ένα προς ένα. Μόνο όταν δημιουργηθεί πλήρως ένα υποδέντρο προχωράει η συνάρτηση στην δημιουργία του επόμενου.

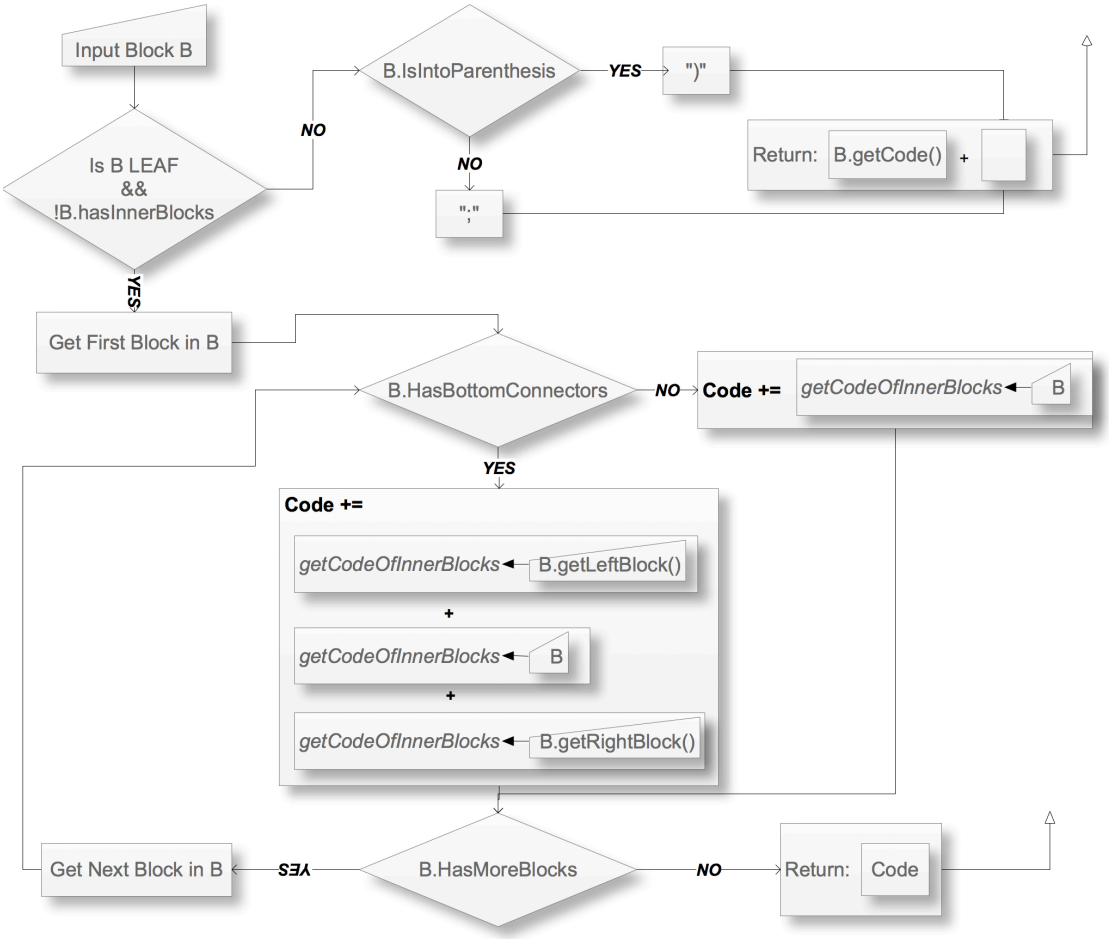
Κάθε φορά που εισάγεται ένας καινούριος κόμβος στο δέντρο προστίθεται και στην συμβολοσειρά της εξόδου ο κώδικας που αναπαριστά αυτός ο κόμβος. Στην μετάφραση αυτή του block σε κώδικα (μέσω του χαρακτηριστικού *code-translation* το οποίο αποθηκεύει την πληροφορία για τον κώδικα που αναπαριστά το block) λαμβάνονται υπόψιν και τα επιμέρους χαρακτηριστικά του block. Αυτά είναι το άμα βρίσκεται μέσα σε παρενθέσεις ή αγκύλες.

Όταν ένα υποδέντρο χτιστεί πλήρως και φτάσουμε στον κόμβο φύλλο (το οποίο δεν έχει άλλα block συνδεδεμένα σε αυτό) ελέγχεται εάν το block φύλλο είναι μέσα σε παρενθέσεις ή αγκύλες, έτσι ώστε να κλειστούν αυτές, και αν ακολουθείται από ερωτηματικό (χαρακτηριστικό *semicolon-following*) και ανάλογα εισάγεται ή όχι ερωτηματικό.

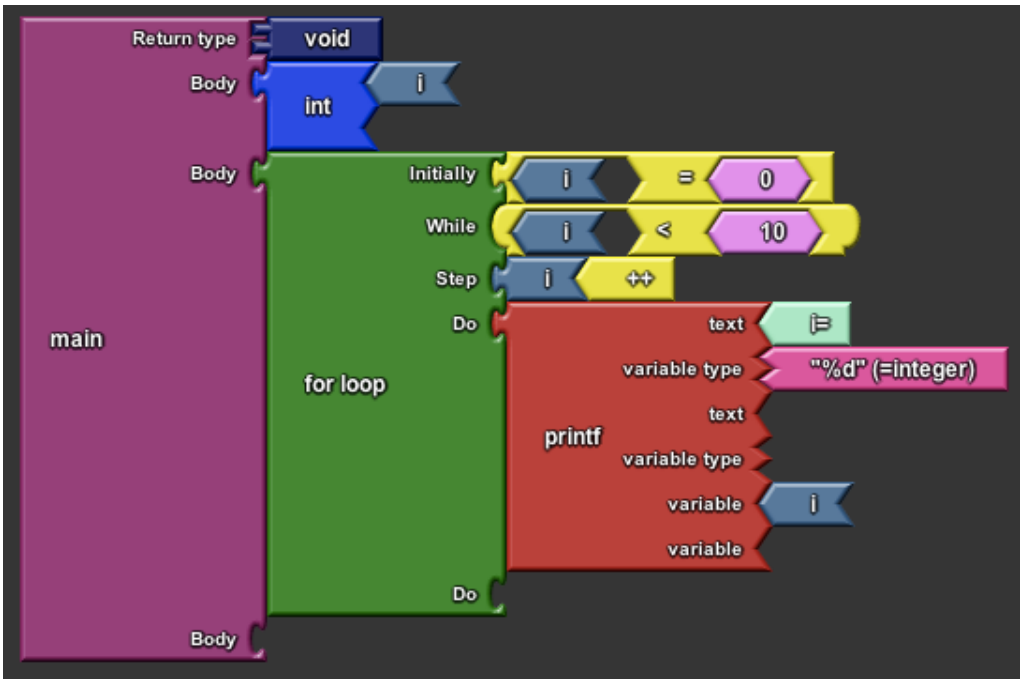
Η παραπάνω διαδικασία επαναλαμβάνεται για όλα τα block που είναι συνδεδεμένα στην συνάρτηση και έπειτα για όλες τις συναρτήσεις. Μια απλοποιημένη και περιληπτική μορφή της συνάρτησης παρουσιάζεται παρακάτω με την μορφή αλγόριθμου.

```
Void exportCodeFromBlocks()
      Functions[]= [UserDefinedFunctions[],Main]
      String C_code;
      For each F in Functions[]
            C_code += getCodeOfInnerBlocks(F)
      End
      Export(C_code)
End


String getCodeOfInnerBlocks(Block B)
      String code;
      If( B is LEAF && !B.hasInnerBlocks() )
        Return B.getCode()+((B.IsIntoParenthesis)? ")" : ";")
      Else
      For each block in B
        If(block.hasBottomConnetcors) //e.g. "="
                code +=
            getCodeOfInnerBlocks(block.getLeftBlock())+
            getCodeOfInnerBlocks(block) +
            getCodeOfInnerBlocks(block.getRigthBlock())
        Else
            code += getCodeOfInnerBlocks (block)
        End
      Return code
End
```

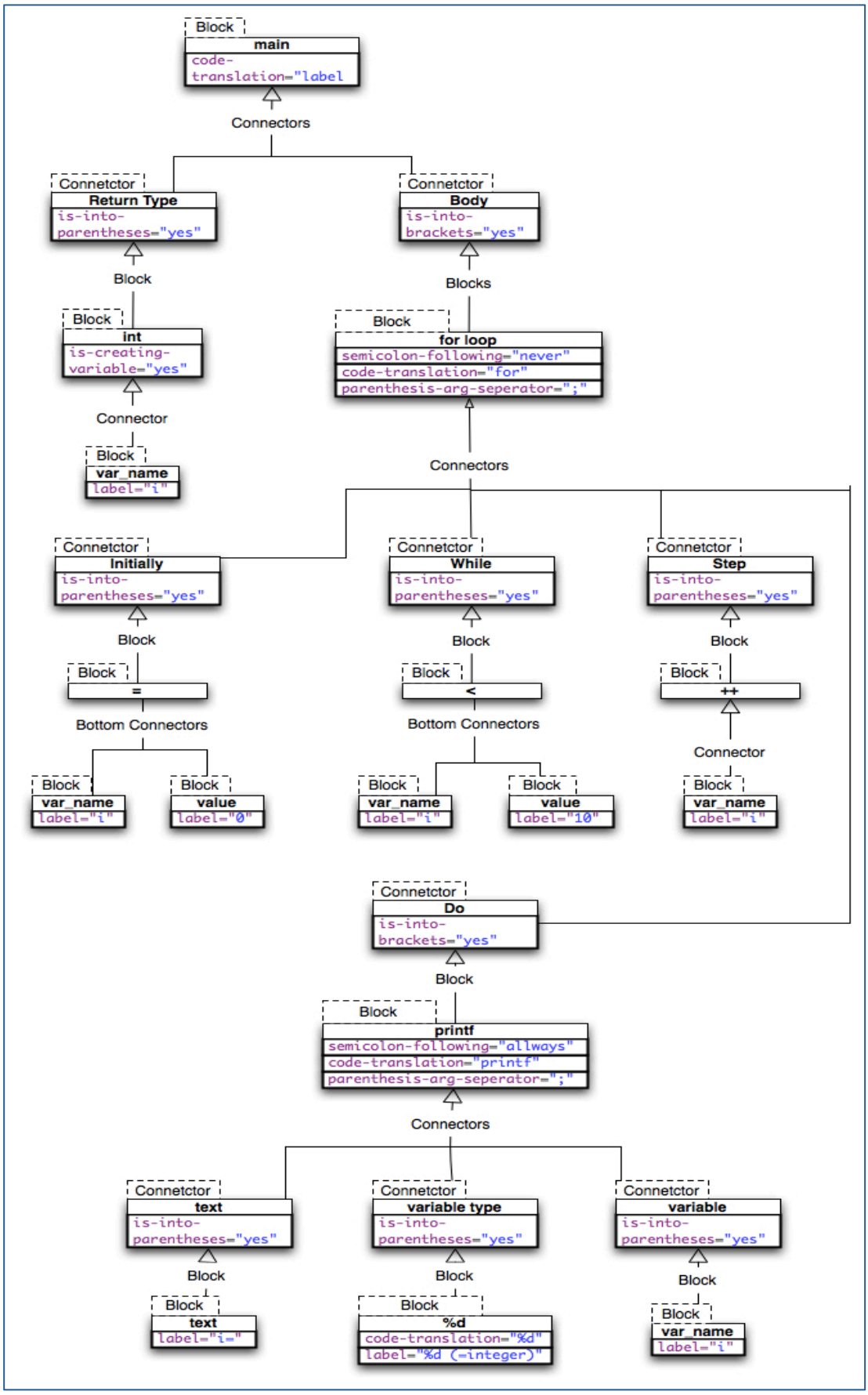
Ένα παράδειγμα για το πως δομείται αναλυτικά ένα δέντρο και πως από αυτό εξάγεται ο κώδικας φαίνεται παρακάτω. Το δέντρο που δομείται μέσω της συνάρτησης ***exportCodeFromBlocks*** του προγράμματος της εικόνας 3.14 φαίνεται στην εικόνα 3.15. Έπειτα στον πίνακα 3.16 εμφανίζεται βήμα βήμα το πως δομείται η εξαχθείσα συμβολοσειρά.



Εικόνα 3.14: Αναπαράσταση του αλγόριθμου εξαγωγής κώδικα C με την μορφή Flow Chart.



Εικόνα 3.15: Ένα απλό παράδειγμα προγράμματος εκτύπωσης των αριθμών από το 0 έως το 9. Σε αυτό το παράδειγμα χρησιμοποιούνται μία δομή επανάληψης (for loop) και μια συνάρτηση εξόδου (printf).



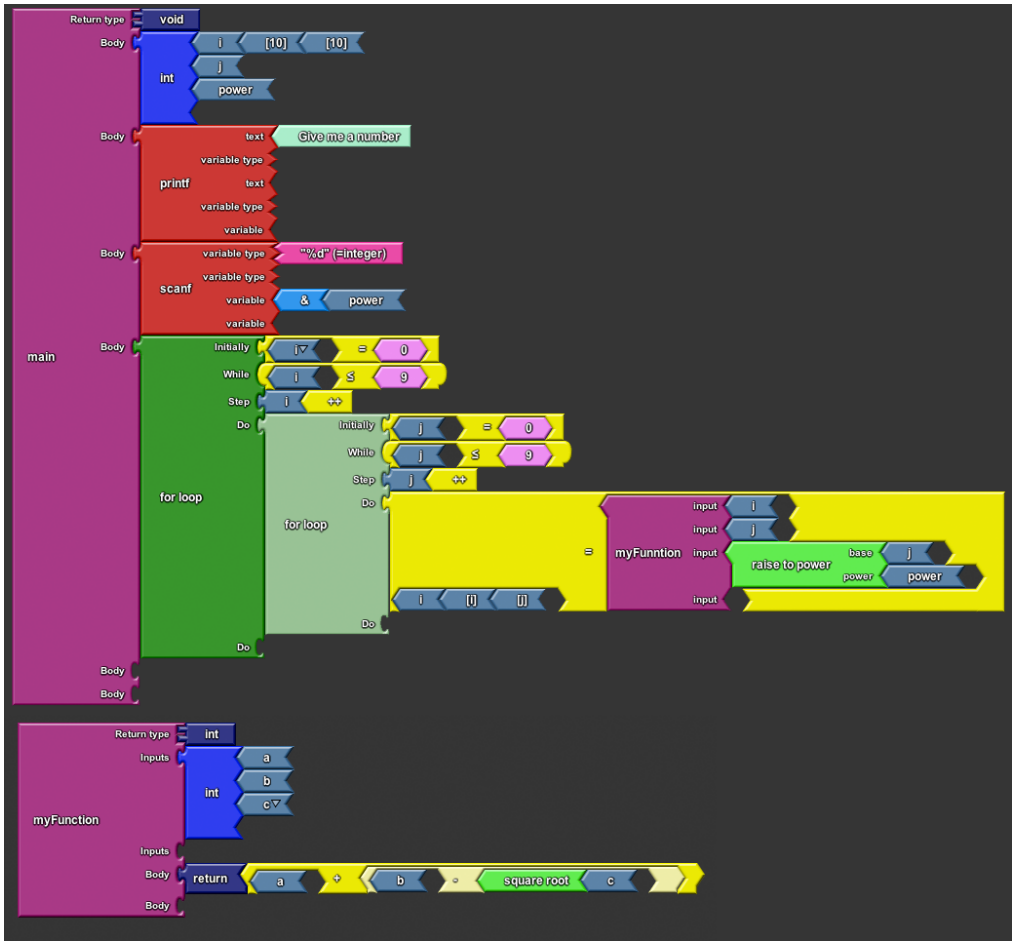
Διάγραμμα 3.16 : Το δέντρο που παράχθηκε από την συνάρτηση getCodeOfInnerBlocks για την εξαγωγή του κώδικα της συνάρτησης της εικόνας 3.13

```
|--Block:void
|  Code=void
|
|--Block:main
|  Code=void main
|
|--Block:main
|  Code=void main(
|
|--Block:Body
|  Code=void main()
|_____
|      |--Block:int
|      |  Code=int
|      |_____
|      |      |--Block:i
|      |      |  Code=i
|      |
|      |--Block:int
|      |  Code=int i
|      |
|      |--Block:int
|      |  Code=int i;
|
|--Block:Body
|  Code=void main()
|       {
|             int i;
|_____
|      |--Block:for loop
|      |  Code=for(
|      |
|      |_____
|      |      |--Block:i
|      |      |  Code=i
|      |
|      |--Block:for loop
|      |  Code=for(
|      |_____
|      |      |--Block:i
|      |      |  Code=i
|      |
|      |--Block:for loop
|      |  Code=for(
|      |
|      |      |_____
|      |      |      |--Block:i
|      |      |      |  Code=i
|      |_____
|      |      |--Block:=
|      |      |  Code=i=
|      |      |_____
|      |      |      |--Block:0
|      |      |      |  Code=0
|      |_____
|      |      |--Block:=
|      |      |  Code=i=0
|      |
```

```
|      |--Connector:Initially
|      |  Code=for(i=0
|      |
|      |--Connector:Initially
|      |  Code=for(i=0;
|      |
|      |      |_____
|      |      |      |--Block:i
|      |      |      |  Code=i
|      |_____
|      |      |--Block:<
|      |      |  Code=i<
|      |      |_____
|      |      |      |--Block:10
|      |      |      |  Code=10
|      |_____
|      |      |--Block:=
|      |      |  Code=i<10
|      |
|      |--Connector:While
|      |  Code=for(i=0;i<10
|      |
|      |--Connector:While
|      |  Code=for(i=0;i<10
|      |
|      |_____
|      |      |--Block:i
|      |      |  Code=i
|      |      |_____
|      |      |      |--Block:++
|      |      |      |  Code=++
|      |_____
|      |      |--Block:i
|      |      |  Code=i++
|      |
|      |--Connector:Step
|      |  Code=for(i=0;i<10;i++
|      |
|      |--Connector:Step
|      |  Code=for(i=0;i<10;i++)
|      |
|      |_____
|      |      |--Block: printf
|      |      |  Code=printf
|      |      |
|      |      |--Block: printf
|      |      |  Code=printf(
|      |      |
|      |      |--Block: printf
|      |      |  Code=printf(“
|      |      |_____
|      |      |      |--Block:i=
|      |      |      |  Code=i=
|      |      |
|      |      |--Block: printf
|      |      |  Code=printf(“i=
|      |      |_____
|      |      |      |--Block: "%d" (=integer)
```

```
|      |      |      |  Code=%d
|      |      |
|      |      |--Block: printf
|      |      |  Code=printf(“i=%d
|      |      |
|      |      |--Block: printf
|      |      |  Code=printf(“i=%d”
|      |      |
|      |      |--Block: printf
|      |      |  Code=printf(“i=%d”,
|      |      |_____
|      |      |      |--Block:i
|      |      |      |  Code=i
|      |      |
|      |      |--Block: printf
|      |      |  Code=printf(“i=%d”,i
|      |      |
|      |      |--Block: printf
|      |      |  Code=printf(“i=%d”,i)
|      |      |
|      |      |--Block: printf
|      |      |  Code=printf(“i=%d”,i);
|      |
|      |--Connector:Do
|      |  Code=for(i=0;i<10;i++)
|              {
|                    printf("i= %d",i);
|
|-- Connector:Body
|   Code=void main()
|         {
|            int i ;
|            for(i=0;i<10;i++)
|            {
|                   printf("i= %d",i);
|            }
|
|-- Connector:Body
|   Code=void main()
|         {
|            int i ;
|            for(i=0;i<10;i++)
|            {
|                   printf("i= %d",i);
|            }
|         }
```

Παρακάτω παρατίθεται ακόμα ένα παράδειγμα ολοκληρωμένου προγράμματος και πώς αυτό θα φαινόταν στο Block-C. Η εικόνα 3.16 περιέχει το πρόγραμμα γραμμένο στο Block-C και η εικόνα 3.17 τον εξαχθέντα κώδικα σε C που προκύπτει από αυτό.



Εικόνα 3.17: Ένα σύνθετο πρόγραμμα στο Block-C

Untitled-Notepad

```
#include <stdio.h>
#include <math.h>
int myFunction(int a,int  b,int c)
{
  return a+(b-sqrt(c));

}
void main()
{
  int i[10][10], j, power ;
  printf("Give me a number");
  scanf("%d",&power);
  for(i=0;i<=9;i++)
  {
    for(j=0;j<=9;j++)
    {
      i[i][j]=myFunntion(i,j,pow(j,power));

    }

  }

}
```

Εικόνα 3.18: Ο εξαχθείς κώδικας του προγράμματος στην εικόνα 3.15

Όπως παρατηρούμε από την εικόνα 3.17 το Block-C συμπληρώνει μόνο του παρενθέσεις, αγκύλες, «αυτάκια», σημεία στίξης και άλλες συντακτικές μονάδες της γλώσσας C, όπως αυτά έχουν καθοριστεί από τον χρήστη στο xml αρχείο που περιγράφει την γλώσσα με βάση τα προαναφερθέντα χαρακτηριστικά. Επίσης δομεί και τον κώδικα εισάγοντας tabs, έτσι ώστε να είναι ευανάγνωστος.

Κατηγορία «User Defined»

Το περιεχόμενο της κατηγορίας «User Defined» εμπλουτίζεται δυναμικά ανάλογα με τις μεταβλητές ή τις συναρτήσεις που δημιουργεί ο χρήστης. Αυτό επιτεύχθηκε με παρεμβολή και επέκταση του κώδικα του openblocks. Τα block που εμπεριέχονται σε αυτήν την κατηγορία διαγράφονται αυτόματα από την κατηγορία όταν ο χρήστης διαγράψει τα αντίστοιχα block που δημιούργησαν τις αντίστοιχες μεταβλητές.

Όπως και στην περίπτωση της εξαγωγής κώδικα, δημιουργήθηκαν τα αντίστοιχα χαρακτηριστικά για τα tags των ***BlockGenus*** και ***BlockConnector*** για να γίνεται δυναμικά η περιγραφή του ποια block δημιουργούν μεταβλητές ή συναρτήσεις.

Τα χαρακτηριστικά αυτά είναι τα is-creating-variable και is-creating-variable-type για τα tags των ***BlockGenus*** και ***BlockConnector*** αντίστοιχα. Μπορούν να πάρουν τις τιμές *yes* και *no* με default τιμή το *no*.

Στην εικόνα 3.18 βλέπουμε την λίστα με τα user defined block που δημιουργήθηκαν με βάση το πρόγραμμα της εικόνας 3.16

Εικόνα 3.19: Τα block που δημιούργησε το εργαλείο με βάσει τις μεταβλητές και την συνάρτηση του προγράμματος της εικόνας 3.15

Κεφάλαιο 4
Διαδικασία Αξιολόγησης της γραφικής διεπαφής του Εργαλείου

Περίληψη

Σε αυτό το κεφάλαιο θα αναλυθεί η διαδικασία της πρώτης αξιολόγησης του εργαλείου και το πώς αυτή βοήθησε στην περαιτέρω ανάπτυξη του, στην βελτίωση του και στην εξαγωγή αποτελεσμάτων για την αποτελεσματικότητα του.

Εισαγωγή

Κάθε σύγχρονη γραφική διεπαφή χρήστη (Graphic User Interface) η οποία στοχεύει στην ευχρηστία ή/και στην αποτελεσματικότητα είναι αναγκαίο να περάσει από μια σειρά αξιολογήσεων καθ' όλη την διάρκεια του σχεδιασμού και της ανάπτυξης της. Αυτές οι αξιολογήσεις βοηθάνε τους σχεδιαστές και προγραμματιστές να την δουν από μια άλλη οπτική γωνία: αυτή του τελικού χρήστη. Αν η διεπαφή δεν προσφέρει στον χρήστη αυτό για το οποίο σχεδιάστηκε ή είναι δύσχρηστη, πρέπει να διορθωθεί ή ακόμα και να επανασχεδιαστεί.

Έτσι και το Block-C κατά την διάρκεια της ανάπτυξης του πέρασε από τρία στάδια αξιολόγησης. Τα δύο πρώτα είχαν ως σκοπό να εξάγουν την πληροφορία εκείνη που αφορούσε την ευχρηστία της γραφική διεπαφής.

Ο στόχος της πρώτης αξιολόγησης, όπως προαναφέρθηκε, ήταν να αναγνωρίσει ελαττώματα που αφορούν την ευχρηστία, και χρήζουν διόρθωσης, μέσα από την χρήση του εργαλείου από αρχαρίους χρήστες. Διεξήχθη μόλις το πρώτο λειτουργικό πρωτότυπο του Block-C ήταν έτοιμο.

Διαδικασία

Ως χρήστες επιλέχθηκαν 3 προπτυχιακοί φοιτητές. Οι δύο φοιτούσαν στο Τμήμα Μηχανικών Παραγωγής και Διοίκησης και ο ένας στο τμήμα Ηλεκτρονικών Μηχανικών και Μηχανικών Υπολογιστών. Οι δύο από τους τρεις δεν είχαν περάσει το εισαγωγικό μάθημα της C του πρώτου εξαμήνου σπουδών του τμήματος. Στο σύνολο τους και οι τρεις είχαν καθόλου ή λίγη εμπειρία με τον προγραμματισμό και δει με τη C. Τα προφίλ των χρηστών παρατίθενται παρακάτω πίνακα.

Πίνακας 4.1: Πίνακας με μία σύντομη καταγραφή των προφίλ των χρηστών της πρώτης αξιολόγησης

Προφίλ χρηστών	**Έτος**	**Τμήμα**	**Φύλο**	**Έχει περάσει το μάθημα της C**	**Σε ποιο έτος πέρασε το μάθημα**
Χρήστης 1	3ο	ΗΜΜΥ	άρρεν	ΝΑΙ	3ο
Χρήστης 2	4ο	ΜΠΔ	άρρεν	ΟΧΙ	-
Χρήστης 3	4ο	ΜΠΔ	άρρεν	ΟΧΙ	-

Κλήθηκαν να υλοποιήσουν ένα βασικό πρόγραμμα σε C αποδομημένο σε 12 διακριτές σύντομες εργασίες (tasks). Το πρόγραμμα σχεδιάστηκε ώστε να περιέχει όλες τις βασικές έννοιες του προγραμματισμού που θα συναντήσει ένας αρχάριος (μεταβλητές, επαναλήψεις, είσοδο και έξοδο, ελέγχους, πίνακες και συναρτήσεις).

- Οι πρώτες έξι εργασίες αποτελούν από μόνες τους ένα πρόγραμμα με την βασική λειτουργικότητα. Πιο συγκεκριμένα, διαβάζει έναν αριθμό από το πληκτρολόγιο και εκτυπώνει όλα τα πολλαπλάσια του 2 από το μηδέν μέχρι τον αριθμό αυτό.
- Στην έβδομη και στην όγδοη εργασία οι χρήστες καλούνται να δημιουργήσουν δύο δικές τους συναρτήσεις. Η μία συνάρτηση επιστρέφει έναν ακέραιο ενώ η δεύτερη δεν επιστρέφει κάτι, έτσι ώστε να φανεί αν οι χρήστες μπορούν να καταλάβουν την διαφοροποίηση των αντίστοιχων block στην κατηγορία “User Defined”.
- Στις επόμενες δύο εργασίες ο χρήστης πρέπει να καλέσει τις δύο συναρτήσεις που δημιούργησε μέσα στην κύρια συνάρτηση.
- Στην ενδέκατη εργασία ο χρήστης καλείται να δημιουργήσει έναν δισδιάστατο πίνακα.
- Στην δωδέκατη εργασία ο χρήστης πρέπει να ενοποιήσει όλα τα παραπάνω βήματα για να λύσει το τελικό πρόβλημα.
- Τέλος, στην δέκατη τρίτη εργασία ο χρήστης καλείται να ολοκληρώσει το πρόγραμμα θέτοντας την κύρια συνάρτηση να επιστρέφει 1.

Για την αξιολόγηση χρησιμοποιήθηκε η μέθοδος «Think Aloud»[32].

Στην μέθοδο αξιολόγησης «Think Aloud» οι χρήστες καλούνται να πετύχουν ένα σύνολος στόχων μέσω κάποιων προκαθορισμένων εργασιών. Κατά την διάρκεια της διαδικασίας η οθόνη του υπολογιστή αλλά και η συνομιλία με τον αξιολογητή καταγράφονται για να μπορούν να εξαχθούν μετέπειτα τα λάθη του γραφικού περιβάλλοντος. Οι χρήστες προτρέπονται συνεχώς να λένε τι σκέφτονται και γιατί αυτό που επιλέγουν πιστεύουν ότι θα τους οδηγήσει στο στόχο που τους τέθηκε.

Οι εργασίες που κλήθηκαν να εκτελέσουν οι χρήστες είναι οι εξής:

1. Δημιουργία της **main** (κύριας) συνάρτησης με τύπο επιστροφής int (ακέραιος).
2. Τη δημιουργία μιας μεταβλητής μέσα στο σώμα της main τύπου **int** με όνομα **input**
3. Την ανάγνωση από το πληκτρολόγιο μιας ακέραιας τιμής και αποθήκευση της στην μεταβλητή **input**
4. Δημιουργία **συμβολοσειράς** μεγέθους 45 χαρακτήρων με όνομα **str**
5. Δημιουργία βρόχου επανάληψης **for** για **input επαναλήψεις**
6. **Σε κάθε επανάληψη** ελέγχεται η τιμή του μετρητή της επανάληψης και **αν η τιμή του είναι πολλαπλάσιο του 2** να εκτυπώνεται αντίστοιχο μήνυμα.
7. Δημιουργία συνάρτησης με όνομα **printValue**. Η συνάρτηση θα δέχεται σαν είσοδο έναν ακέραιο (int) θα τον εκτυπώνει και δεν θα επιστρέφει τίποτα (τύπος επιστροφής void).
8. Δημιουργία συνάρτησης με όνομα **getSquare** η οποία θα δέχεται είσοδο έναν int με όνομα input. Η συνάρτηση θα **επιστρέφει το τετράγωνο** της τιμής **input**.
9. Κλήση της συνάρτησης **getSquare** από την main **με είσοδο input**, και αποθήκευση της τιμής της σε μία μεταβλητή με όνομα που θα επιλέξει ο χρήστης.
10. Εκτύπωση της παραπάνω μεταβλητής με την χρήση της συνάρτησης **printValue**
11. Δημιουργία ενός **δισδιάστατου πίνακα int** με όνομα **array** και μέγεθος (input x input).
12. Γέμισμα του πίνακα
    - Κάθε θέση του πίνακα θα έχει την τιμή του τετραγώνου του αθροίσματος της θέσης (**στήλη + γραμμή στο τετράγωνο**).
      - Πχ η θέση στη γραμμή 4 και στη στήλη 3 του πίνακα array θα έχει την τιμή $(4+3)^2=7^2=49$. Άρα array[4][3]=49
13. Η συνάρτηση main να επιστρέφει 1η

Εικόνα 4.2: Οι εργασίες που κλήθηκαν να υλοποιήσουν οι χρήστες στην πρώτη αξιολόγηση

Στο τέλος της διαδικασίας οι χρήστες συμπλήρωσαν ένα ερωτηματολόγιο για να καταγραφεί η άποψη τους για το εργαλείο.

Το ερωτηματολόγιο που συμπλήρωσαν οι χρήστες είναι το εξής:

Ερωτηματολόγιο Αξιολόγησης BLOCK-C

Το παρών ερωτηματολόγιο είναι ανώνυμο. Προσπαθεί να καταγράψει την άποψη των χρηστών που πήραν μέρος στην διαδικασία της αξιολόγησης του BLOCK-C. Σκοπός του είναι η εξαγωγή θετικών και αρνητικών εντυπώσεων οι οποίες θα βοηθήσουν στην περαιτέρω εξέλιξη του προγράμματος. Σε καμία από τις ερωτήσεις δεν είναι υποχρεωτική η απάντηση.

Έτος:..... Τμήμα:..... Φύλο:.....

Έχετε περάσει το εισαγωγικό μάθημα της C (αν ναι σε ποιο έτος):.....

Οι απαντήσεις στις παρακάτω ερωτήσεις μπορούν να λάβουν τιμές μεταξύ 1 και 5 όπου, 1 είναι καθόλου και 5 πολύ.

Ερώτηση	1 καθόλου	2 λίγο	3 μέτρια	4 αρκετά	5 πολύ
Έχετε εμπειρία με τον προγραμματισμό;	1	2	3	4	5
Έχετε εμπειρία με την γλώσσα προγραμματισμού C;	1	2	3	4	5
Έχετε εμπειρία με κάποια γλώσσα προγραμματισμού ή εργαλείο προγραμματισμού για αρχάριους;	1	2	3	4	5
Πόσο ευχάριστα πέρασε η ώρα σας χρησιμοποιώντας το BLOCK-C;	1	2	3	4	5
Πιστεύετε ότι σας βοήθησε το BLOCK-C να αντιληφθείτε τις βασικές έννοιες του προγραμματισμού;	1	2	3	4	5
Πιστεύετε ότι σας βοήθησε το BLOCK-C να κατανοήσετε τον τρόπο που δομείται ένα πρόγραμμα;	1	2	3	4	5
Πιστεύετε ότι ο προγραμματισμός με «τουβλάκια» μπορεί να ενθαρρύνει αρχάριους να ασχοληθούν με τον προγραμματισμό;	1	2	3	4	5
Πιστεύετε ότι το BLOCK-C κατάφερε να κάνει τον προγραμματισμό σε C να μοιάζει με παιχνίδι;	1	2	3	4	5
Αν το BLOCK-C σας δινόταν σαν εργαλείο ,στα πλαίσια του εισαγωγικού μαθήματος της C στη σχολή σας, πιστεύετε ότι θα το χρησιμοποιούσατε;	1	2	3	4	5
Πιστεύετε ότι κουμπιά undo/redo θα ήταν χρήσιμα στο BLOCK-C;	1	2	3	4	5
Πόσο ικανοποιημένος/η μείνατε από το γραφικό περιβάλλον του BLOCK-C;	1	2	3	4	5
Πόσο ικανοποιημένος/η μείνατε από τις κατηγορίες των blocks («τουβλάκια»); Ήταν καλή η ομαδοποίηση;	1	2	3	4	5
Πόσο ικανοποιημένος/η μείνατε από την ευχρηστία του BLOCK-C;	1	2	3	4	5
Πόσο ικανοποιημένος/η μείνατε συνολικά από το BLOCK-C;	1	2	3	4	5

Εικόνα 4.3: Πρώτη σελίδα ερωτηματολογίου που κλήθηκαν να συμπληρώσουν οι χρήστες μετά το πέρας της διαδικασίας αξιολόγησης

Στις παρακάτω ερωτήσεις καλείστε να απαντήσετε γράφοντας τα προσωπικά σας σχόλια.

Κατά την άποψη σας τι θα μπορούσε να προστεθεί ή τι λείπει από το BLOCK-C;

Ποιο/α ήταν το χαρακτηριστικό/ά που σας άρεσε πιο πολύ στο BLOCK-C και γιατί;

Τι πιστεύεται οτι σας πρόσφερε το Block-C το οποίο δεν σας πρόσφερε ο κλασσικός τρόπος εκμάθησης προγραμματισμού;

Άμα σας ζητούσε κάποιος να του περιγράψετε σύντομα την λογική με την οποία λειτουργεί το BLOCK-C , πως θα του τη περιγράφατε;

Ευχαριστούμε πολύ για τον χρόνο σας!

Εικόνα 4.4: Δεύτερη σελίδα ερωτηματολογίου που κλήθηκαν να συμπληρώσουν οι χρήστες μετά το πέρας της διαδικασίας αξιολόγησης

Αποτελέσματα

Η πρώτη αυτή αξιολόγηση κατέδειξε αρκετά προβλήματα ευχρηστίας τα οποία μπέρδευαν τους χρήστες. Όλα αυτά τα λάθη ή τα σφάλματα διορθώθηκαν και μερικά μάλιστα από αυτά ήταν αρκετά σοβαρά.

Τα λάθη που εντοπίστηκαν από την διαδικασία και διορθώθηκαν ήταν τα εξής:

1. Την διαφοροποίηση του i++ και του ++i το οποίο μπέρδεψε όλους τους χρήστες.
2. Την καλύτερη σχεδίαση του printf και της δημιουργίας γραφικής αναπαράστασης για το αναγνωριστικό του τύπου της μεταβλητής (%d, %f κλπ)
3. Την δυνατότητα παρεμβολής block μεταξύ άλλων block (σε περίπτωση που ο χρήστης ξέχασε να εισάγει ένα block).

Επίσης αρκετά bugs εντοπίστηκαν και διορθώθηκαν.

Στους παρακάτω πίνακες παρουσιάζονται τα αποτελέσματα για τους χρόνους ολοκλήρωσης κάθε εργασίας και τον αριθμό σφαλμάτων για κάθε χρήστη.

Πίνακας 4.5: Αποτελέσματα 1ου χρήστη

Task	Έναρξη	Λήξη	Αριθμός Σφαλμάτων	Επιτυχία Ολοκλήρωσης	Σύνολο
1	00:00:00	00:01:13	2	ΝΑΙ	00:01:13
2	00:01:13	00:01:46	0	ΝΑΙ	00:00:33
3	00:01:46	00:04:11	0	ΝΑΙ	00:02:25
4	00:04:11	00:06:00	1	ΝΑΙ	00:01:49
5	00:06:00	00:21:47	5	ΝΑΙ	00:15:47
6	00:21:47	00:26:00	1	ΝΑΙ	00:04:13
7	00:26:00	00:29:23	0	ΝΑΙ	00:03:23
8	00:29:23	00:33:59	2	ΝΑΙ	00:04:36
9	00:33:59	00:37:08	1	ΝΑΙ	00:03:09
10	00:37:08	00:41:24	2	ΝΑΙ	00:04:16
11	00:41:24	00:48:30	2	ΝΑΙ	00:07:06
12	00:48:30	00:49:26	0	ΝΑΙ	00:00:56
		Σύνολο	16		00:49:26

Πίνακας 4.6: Αποτελέσματα 2ου χρήστη

Task	Έναρξη	Λήξη	Αριθμός Σφαλμάτων	Επιτυχία Ολοκλήρωσης	Σύνολο
1	00:00:00	00:03:46	3	ΝΑΙ	00:03:46
2	00:03:46	00:08:08	1	ΝΑΙ	00:04:22
3	00:10:47	00:15:10	2	ΝΑΙ	00:04:23
4	00:08:08	00:10:47	2	ΝΑΙ	00:02:39
5	00:15:10	00:39:35	2	ΝΑΙ	00:24:25
6	00:39:35	00:49:39	0	ΝΑΙ	00:10:04
7	00:49:39	00:55:37	2	ΝΑΙ	00:05:58
8	00:55:37	01:03:55	1	ΝΑΙ	00:08:18
9	01:03:55	01:08:13	0	ΝΑΙ	00:04:18
10	01:08:13	01:11:42	2	ΝΑΙ	00:03:29
11	01:11:42	01:28:29	2	ΝΑΙ	00:16:47
12	01:28:29	01:29:17	0	ΝΑΙ	00:00:48
		Σύνολο	17		01:29:17

Πίνακας 4.7: Αποτελέσματα 3ου χρήστη

Task	Έναρξη	Λήξη	Αριθμός Σφαλμάτων	Επιτυχία Ολοκλήρωσης	Σύνολο
1	00:00:00	00:02:32	0	ΝΑΙ	00:02:32
2	00:02:32	00:04:19	2	ΝΑΙ	00:01:47
3	00:04:19	00:06:53	1	ΝΑΙ	00:02:34
4	00:06:53	00:09:43	1	ΝΑΙ	00:02:50
5	00:09:43	00:20:17	2	ΝΑΙ	00:10:34
6	00:20:17	00:24:58	0	ΝΑΙ	00:04:41
7	00:24:58	00:30:34	2	ΝΑΙ	00:05:36
8	00:30:34	00:35:27	0	ΝΑΙ	00:04:53
9	00:35:27	00:39:39	0	ΝΑΙ	00:04:12
10	00:39:39	00:45:01	2	ΝΑΙ	00:05:22
11	00:45:01	00:54:52	2	ΝΑΙ	00:09:51
12	00:54:52	00:56:18	0	ΝΑΙ	00:01:26
		Σύνολο	12		00:56:18

Ακολουθούν τα συγκεντρωτικά αποτελέσματα

Πίνακας 4.8: Πίνακας Συγκεντρωτικών αποτελεσμάτων 1ης Αξιολόγησης

Task	Μέσος Χρόνος	Μέσος Αριθμός Σφαλμάτων	Ποσοστό Ολοκλήρωσης	Ελάχιστος Χρόνος Ολοκλήρωσης	Μέγιστος Χρόνος Ολοκλήρωσης
1	00:02:30	1,67	100%	00:01:13	00:03:46
2	00:02:14	1	100%	00:00:33	00:04:22
3	00:03:07	1	100%	00:02:25	00:04:23
4	00:02:26	1,33	100%	00:01:49	00:02:50
5	00:16:55	3	100%	00:10:34	00:24:25
6	00:06:19	0,33	100%	00:04:13	00:10:04
7	00:04:59	1,33	100%	00:03:23	00:05:58
8	00:05:56	1	100%	00:04:36	00:08:18
9	00:03:53	0,33	100%	00:03:09	00:04:18
10	00:04:22	2	100%	00:03:29	00:05:22
11	00:11:15	2	100%	00:07:06	00:16:47
12	00:01:03	0	100%	00:00:48	00:01:26
Μέσο:	05:25	1.25	100%	-	-

Τα αποτελέσματα που προέκυψαν από τα ερωτηματολόγια παρατίθενται παρακάτω:

Πίνακας 4.9: Πίνακας αποτελεσμάτων ερωτηματολογίων 1ης αξιολόγησης για τις ερωτήσεις που αφορούν την Εμπειρία των Χρηστών στον Προγραμματισμό

Ερώτηση	Χρήστης 1	Χρήστης 2	Χρήστης 3	Μ.Ο
1) Έχετε εμπειρία με τον προγραμματισμό;	3	3	3	3
2) Έχετε εμπειρία με την γλώσσα προγραμματισμού C;	3	1	2	2
3) Έχετε εμπειρία με κάποια γλώσσα προγραμματισμού ή εργαλείο προγραμματισμού για αρχάριους;	1	1	1	1
Μ.Ο.	2.33	1.66	2	2

Πίνακας 4.10: Πίνακας αποτελεσμάτων ερωτηματολογίων 1ης αξιολόγησης για τις ερωτήσεις που αφορούν την Γνώμη των Χρηστών για το Εργαλείο

Ερώτηση	Χρήστης 1	Χρήστης 2	Χρήστης 3	Μ.Ο
4) Πόσο ευχάριστα πέρασε η ώρα σας χρησιμοποιώντας το BLOCK-C;	4	4	4	4
5) Πιστεύετε ότι σας βοήθησε το BLOCK-C να αντιληφθείτε τις βασικές έννοιες του προγραμματισμού;	5	4	5	4.67
6) Πιστεύετε ότι σας βοήθησε το BLOCK-C να κατανοήσετε τον τρόπο που δομείται ένα πρόγραμμα;	5	4	4	4.33
7) Πιστεύετε ότι ο προγραμματισμός με «τουβλάκια» μπορεί να ενθαρρύνει αρχάριους να ασχοληθούν με τον προγραμματισμό;	4	5	4	4.33
8) Πιστεύετε ότι το BLOCK-C κατάφερε να κάνει τον προγραμματισμό σε C να μοιάζει με παιχνίδι;	4	4	4	4
9) Αν το BLOCK-C σας δινόταν σαν εργαλείο ,στα πλαίσια του εισαγωγικού μαθήματος της C στη σχολή σας, πιστεύετε ότι θα το χρησιμοποιούσατε;	5	4	3	4
10) Πιστεύετε ότι κουμπιά undo/redo θα ήταν χρήσιμα στο BLOCK-C;	4	4	2	3.33
11) Πόσο ικανοποιημένος/η μείνατε από το γραφικό περιβάλλον του BLOCK-C;	4	5	4	4.33
12) Πόσο ικανοποιημένος/η μείνατε από τις κατηγορίες των blocks («τουβλάκια»); Ήταν καλή η ομαδοποίηση;	4	4	4	4
13) Πόσο ικανοποιημένος/η μείνατε από την ευχρηστία του BLOCK-C;	4	4	4	4
14) Πόσο ικανοποιημένος/η μείνατε συνολικά από το BLOCK-C;	4	4	4	4
Μ.Ο.	4.27	4.18	3.81	4.09

Ενδιαφέρον είχαν επίσης οι απαντήσεις των χρηστών στις ερωτήσεις:

Στην ερώτηση «Ποιο/α χαρακτηριστικό/α του Block-C σας άρεσε πιο πολύ και γιατί; » οι χρήστες απάντησαν:

- Ήταν οι εσοχές που έμπαιναν τα τουβλάκια γιατί με τον τρόπο αυτό μπορούσα να κατανοήσω περισσότερο τι ακριβώς ήθελα να κάνω και αυτό

που ήθελα να κάνω ήταν σωστό, Δηλαδή η καθοδήγηση που μου έδιναν αυτές οι εσοχές

- Τα τουβλάκια γιατί σε βοηθάνε να κατανοήσεις πιο εύκολα ποιο πάει με ποιο

- Ότι το κάθε τουβλάκι δεν μπαίνει όπου να ΄ναι και έτσι έχεις έναν κατευθυντήριο μπούσουλα που σε βοηθάει

Στην ερώτηση «Τι πιστεύετε ότι σας προσέφερε το Block-C το οποίο δεν σας πρόσφερε ο κλασσικός τρόπος εκμάθησης προγραμματισμού;» χαρακτηριστικές ήταν οι εξής απαντήσεις:

- Ότι είναι ένα πρόγραμμα που σου δείχνει την σωστή δομή της C και σε βοηθάει να μάθεις καλύτερα πως να "φτιάχνεις" ένα πρόγραμμα

- Έκανε πιο απλό το γράψιμο κώδικα, κάτι που θα βοηθήσεις τους αρχάριους, οι οποίοι ξενίζονται αρχικά από τον τρόπο λειτουργίας της C

- Λεπτομερής σχηματική απεικόνιση του κώδικα και κατανόηση της λειτουργίας του (πχ. επαναλήψεων).

Στην ερώτηση «Άμα σας ζητούσε κάποιος να του περιγράψετε σύντομα την λογική με την οποία λειτουργεί το Block-C, πώς θα του την περιγράφατε; » οι χρήστες απάντησαν:

- Ότι είναι ένα πρόγραμμα που σου δείχνει την σωστή δομή της C και σε βοηθάει να μάθεις καλύτερα πως να "φτιάχνεις" ένα πρόγραμμα

- Είναι ένα πρόγραμμα που μέσω ενός παιχνιδιού puzzle σε βοηθά να εκτελέσεις ένα πρόγραμμα στη C κατανοώντας το παράλληλα με ένα εύκολο και διασκεδαστικό τρόπο. Το πρόγραμμα λειτουργεί ακριβώς όπως ένα puzzle συνδέοντας τουβλάκια ομοιόμορφα που βρίσκονται σε οργανωμένες λίστες.

- Το Block-C είναι ένα πρόγραμμα που θέλει να δείξει σε αρχάριους τις εντολές που χρησιμοποιεί η γλώσσα προγραμματισμού C με έναν πολύ πιο κατανοητό τρόπο από αυτόν που χρειάζεται η C για να δημιουργήσει ένα πρόγραμμα και δίνει τη δυνατότητα σε αρχάριους που δεν έχουν καμία εμπειρία με τη C να δημιουργήσουν εύκολα ένα πρόγραμμα και να πειραματιστούν πάνω σ' αυτό. Πιστεύω ότι είναι ένα βοήθημα, ώστε να μάθεις και να κατανοήσεις τις λειτουργίες της C.

Συμπεράσματα

Από ό,τι φαίνεται από τις παραπάνω απαντήσεις οι χρήστες αντιλήφθηκαν τον στόχο του Block-C και έμειναν ευχαριστημένοι από την αλληλεπίδραση τους με αυτό.

Αξιοσημείωτο είναι το γεγονός ότι και οι τρεις χρήστες ολοκλήρωσαν όλες τις εργασίες που τους ανατέθηκαν. Αυτό δείχνει ότι το εργαλείο τους παρακίνησε και δεν απογοητεύθηκαν παρά τα όποια λάθη ή δυσκολίες αντιμετώπισαν.

Επίσης στις ερωτήσεις που τους ζητήθηκε η γνώμη τους για το εργαλείο, η μέση τιμή των απαντήσεων των χρηστών για όλες τις απαντήσεις είναι 4.09 στα 5. Το αποτέλεσμα αυτό κάνει καταφανές το γεγονός ότι οι χρήστες έμειναν ικανοποιημένοι από το εργαλείο.

Κεφάλαιο 5
Διαδικασία Αξιολόγησης ευχρηστίας της γραφικής διεπαφής και της πρόληψης συντακτικών λαθών.

Περίληψη

Στο κεφάλαιο αυτό αναλύεται η διαδικασία της δεύτερης αξιολόγησης του εργαλείου και το πώς αυτή βοήθησε στην περαιτέρω ανάπτυξη του, στην βελτίωση του και στην εξαγωγή αποτελεσμάτων για την αποτελεσματικότητα του.

Εισαγωγή

Μετά το πέρας της πρώτης αξιολόγησης τα αποτελέσματα της μας τροφοδότησαν με αρκετές πληροφορίες για τη γραφική διεπαφή και την ευχρηστία της. Αρκετά προβλήματα διορθώθηκαν όπως επίσης και αρκετά bugs.

Μετά την ολοκλήρωση της επόμενης έκδοσης του Block-C, με διορθωμένα τα σφάλματα και τα λάθη της πρώτης έκδοσης, έπρεπε να δοκιμαστεί κι αυτή για τυχόν λάθη και σφάλματα.

Στόχος αυτής της δεύτερης αξιολόγησης ήταν να μελετηθεί περαιτέρω η ευχρηστία της γραφικής διεπαφής καθώς και η δυνατότητα του εργαλείου να προλαμβάνει τα συντακτικά λάθη. Γι΄ αυτήν την διαδικασία της αξιολόγησης αποφασίστηκε να χρησιμοποιηθούν χρήστες οι οποίοι γνωρίζουν τη γλώσσα προγραμματισμού C και έχουν εμπειρία από τη διδασκαλία μαθημάτων

προγραμματισμού βασισμένα στη γλώσσα αυτή. Έτσι επιλέχθηκαν τρεις καθηγητές του τμήματος Ηλεκτρονικών Μηχανικών και Μηχανικών Ηλεκτρονικών Υπολογιστών οι οποίοι είχαν διδάξει έστω μια φορά το εισαγωγικό μάθημα στον προγραμματισμό με την χρήση της γλώσσας C.

Η παραπάνω επιλογή έγινε με γνώμονα την σφαιρική αξιολόγηση, έτσι ώστε να μπορούμε να έχουμε μια συνολική προσέγγιση χρησιμοποιώντας ετερογενείς χρήστες μεταξύ των αξιολογήσεων. Έτσι κάθε αξιολόγηση είχε διαφορετικού υπόβαθρου χρήστες.

Διαδικασία

Η αξιολόγηση έγινε μέσω ημιδομημένης συνέντευξης. Οι χρήστες κλήθηκαν να δημιουργήσουν ένα απλό πρόγραμμα σε C και να πλοηγηθούν ελεύθερα στο εργαλείο ώστε να ανακαλύψουν τις δυνατότητές του. Παράλληλα οι χρήστες κλήθηκαν μέσω προφορικών ερωτήσεων, οι οποίες χρησιμοποιήθηκαν αντί ερωτηματολογίου, να αναφέρουν την άποψη τους για την χρήση του Block-C ως βοηθητικού εργαλείου, τις ευκολίες που προσφέρει, την δυνατότητα του να καλύψει μεγάλο εύρος της διαδικασίας της εκπαίδευσης και γενικότερα την άποψη τους για αυτό.

Τα προφίλ των χρηστών παρατίθενται στον παρακάτω πίνακα:

Πίνακας 5.1: Πίνακας με μία σύντομη καταγραφή των προφίλ των χρηστών της δεύτερης αξιολόγησης

Προφίλ χρηστών	**Βαθμίδα**	**Τμήμα**	**Φύλο**	**Έχει διδάξει C σε εισαγωγικό μάθημα;**
Χρήστης 1	Επίκουρος Καθηγητής	ΗΜΜΥ	άρρεν	ΝΑΙ
Χρήστης 2	Επίκουρη Καθηγήτρια	ΗΜΜΥ	θήλυ	ΝΑΙ
Χρήστης 3	Επίκουρος Καθηγητής	ΗΜΜΥ	άρρεν	ΝΑΙ

Οι ερωτήσεις που τους έγιναν κατά την διάρκεια της διαδικασίας είναι οι εξής:

1. Πώς θα περιγράφατε Block-C στους φοιτητές σας;
2. Πώς θα εξηγούσατε τη διαφορά “==”με το “=” με τη βοήθεια του Block-C;
3. Γενικότερα πώς θα εξηγούσατε γιατί διαφέρουν μεταξύ τους οι Connectors του Block-C με βάση τη θεωρία C;
4. Πιστεύετε ότι μπορεί να υποστηρίξει μεγάλο εύρος παραδειγμάτων;
5. Με βάση την ύλη και τις ασκήσεις πιστεύετε ότι μπορεί να χρησιμοποιηθεί

για την υποστήριξη της διδασκαλίας;

6. Θα θέλατε όταν ξεκινούσατε να μαθαίνετε προγραμματισμό για πρώτη φορά να σας δινόταν ένα αντίστοιχο;

7. Πιστεύετε ότι μπορεί να βοηθήσει τους φοιτητές;

8. Πιστεύετε ότι συνδέεται περισσότερο με την εκμάθηση, την εκπόνηση μιας εργασίας ή και με τα δύο;

9. Ποιο είναι το εύρος της ύλης που μπορεί να υποστηρίξει;

10. Τι πιστεύετε ότι λείπει ή σε τι υστερεί το Block-C;

11. Θα το χρησιμοποιούσατε στο μάθημα ή στο εργαστήριο;

Εικόνα 5.2: Οι ερωτήσεις που κλήθηκαν να απαντήσουν οι χρήστες που συμμετείχαν στην δεύτερη αξιολόγηση κατά την διάρκεια της συνέντευξης

Οι χρήστες κλήθηκαν (και προτρεπόταν συνεχώς) να λένε τι σκέφτονται καθ' όλη την διάρκεια της διαδικασίας. Επίσης η οθόνη των χρηστών καταγραφόταν για να μπορούν στη συνέχεια να μελετηθούν οι χειρισμοί που έγιναν ώστε να να εξαχθούν συμπεράσματα για την ευχρηστία του εργαλείου.

Αποτελέσματα

Στα πλαίσια αυτής της αξιολόγησης δεν καταγράφηκε ο αριθμός των λαθών των χρηστών γιατί κρίθηκε άσκοπο εφόσον δεν υπήρχαν «εργασίες» με τις οποίες να μπορούν να συνδεθούν τα λάθη. Παρ' όλα αυτά τα λάθη των χρηστών και οι δυσκολίες τους στην πλοήγηση και κατανόηση του εργαλείου βοήθησαν αρκετά στην εκσφαλμάτωση και στην περαιτέρω ανάπτυξη του.

Διαπιστώθηκαν κάποια σοβαρά προβλήματα της γραφικής διεπαφής που οδήγησαν στις παρακάτω βελτιώσεις:

1. Δημιουργία ξεχωριστής κατηγορίας για τα blocks των μεταβλητών (values)
2. Η ενσωμάτωση της συνάρτησης *main* στον χώρο που αναπτύσσεται το πρόγραμμα (workspace)
3. Η εισαγωγή επιπρόσθετης λειτουργικότητας για undo / redo.
4. Με την αντιγραφή ενός block να αντιγράφονται και τα συνδεδεμένα σε αυτό blocks

Πολλά από τα προβλήματα διαπιστώθηκαν κατά την καταγραφή των αποτελεσμάτων της αξιολόγησης (εξαγωγή τους μέσω της παρακολούθησης της βιντεοσκοπημένης συνέντευξης) και κάποια προτάθηκαν από τους χρήστες.

Τα σφάλματα που έκαναν οι χρήστες λόγω δυσχρηστίας της γραφικής διεπαφής του εργαλείου παρατίθενται στον πίνακα 5.3.

Πίνακας 5.3: Πίνακας με τον αριθμό των σφαλμάτων στα οποία οδηγήθηκε ο κάθε χρήστης κατά την διάρκεια της χρήσης του εργαλείου, λόγω δυσχρηστίας της γραφικής διεπαφής του.

Χρήστης	Αριθμός Σφαλμάτων
1	1
2	13
3	11
Μ.Ο.	8.33

Μεγαλύτερο όμως ενδιαφέρον έχουν οι απαντήσεις των χρηστών στις ερωτήσεις που έγιναν κατά την διάρκεια της συνέντευξης. Δεν απάντησαν όλοι οι χρήστες σε όλες τις ερωτήσεις είτε γιατί ήταν απασχολημένοι με τη χρήση του εργαλείου είτε επειδή ήταν υπερβολικά λακωνικοί. Παρακάτω παρατίθενται οι πιο χαρακτηριστικές απαντήσεις για κάθε ερώτηση:

Ερώτηση 1:

«*Πώς θα περιγράφατε Block-C στους φοιτητές σας;*»

- Θα το περιέγραφα στους φοιτητές ως εξής: «Ένα editor διαφορετικό από τους άλλους, στο οποίο δεν γράφετε κείμενο αλλά χρησιμοποιήστε γραφικά το drag & drop για να γράψετε το πρόγραμμα σας. »

Ερώτηση 2:

«*Πώς θα εξηγούσατε τη διαφορά "=="με το "=" με τη βοήθεια του Block-C;*»

- Ενώ και τα δύο είναι operators το ένα χρησιμοποιείται για ανάθεση και το άλλο για σύγκριση. Μπορείτε να δείτε την διαφορά από τον σύνδεσμο τους.

Ερώτηση 3:

«*Γενικότερα πώς θα εξηγούσατε γιατί διαφέρουν μεταξύ τους οι Connectors του Block-C με βάση τη θεωρία C;*»

- Κάθε διαφορετικός σύνδεσμος μας δείχνει τι πρέπει να βάλεις που.

Ερώτηση 4:

«*Πιστεύετε ότι μπορεί να υποστηρίξει μεγάλο εύρος παραδειγμάτων;*»

- Σίγουρα! Ειδικά στην αρχή που οι εργασίες που ανατίθενται στους φοιτητές είναι προγραμματάκια εύκολα αλλά απαιτητικά με if και for.

Ερώτηση 5:

«*Με βάση την ύλη και τις ασκήσεις πιστεύετε ότι μπορεί να χρησιμοποιηθεί για την υποστήριξη της διδασκαλίας;*»

- Σίγουρα, στα πρώτα θέματα του μαθήματος, δηλαδή όταν μπαίνει κανείς στις βασικές δομές και στις βασικές εντολές θα ήταν περισσότερο χρήσιμο.
- Δεν μπορώ να απαντήσω με σιγουριά. Υπάρχουν κάποιοι φοιτητές που κολλάνε πάρα πολύ στην αρχή. Θα μπορούσε να τους βοηθήσει να ξεκολλήσουν. Για κάποιους λοιπόν θα μπορούσε, για άλλους όχι. Οι φοιτητές που φοβούνται τον προγραμματισμό είναι πολλοί, και πάρα πολλές κοπέλες, δυστυχώς.

Ερώτηση 6:

«*Θα θέλατε όταν ξεκινούσατε να μαθαίνετε προγραμματισμό για πρώτη φορά να σας δινόταν ένα αντίστοιχο;»*

- Βεβαίως· σίγουρα! Αν το είχα αυτό όταν μάθαινα C, θα με βοήθαγε πάρα πολύ. Τουλάχιστον για να μάθω πως συντάσσεται σωστά ένα loop.

Ερώτηση 7:

«*Πιστεύετε ότι μπορεί να βοηθήσει τους φοιτητές;*»

- Ναι, θα τους βοηθούσε τουλάχιστον σε ένα υψηλό επίπεδο –σε επίπεδο block- να μην κάνουνε λάθη, έτσι ώστε να ξέρουνε κάθε block από ποια κομμάτια αποτελείται και τι ταιριάζει που.

Ερώτηση 8:

«*Πιστεύετε ότι συνδέεται περισσότερο με την εκμάθηση, την εκπόνηση μιας εργασίας ή εξίσου και με τα δύο;*»

- Περισσότερο με την διδασκαλία θα έλεγα, γιατί η εκπόνηση μιας εργασίας είναι ένα πράγμα που επιλέγει μόνος του ο καθένας πως θα το κάνει.

Ερώτηση 9:

«*Ποιο είναι το εύρος της ύλης που μπορεί να υποστηρίξει;»*

- Απ’ ό,τι παρατηρώ έχει και λειτουργικότητα για pointers κλπ άρα μπορεί να καλύψει μεγάλο εύρος της ύλης αλλά νομίζω ότι μετά από ένα σημείο δεν θα βοηθούσε η χρήση του.

Ερώτηση 10:

«*Τι πιστεύετε ότι λείπει ή σε τι υστερεί το Block-C;*»

- Νομίζω ότι είναι αρκετά διαισθητικό το όλο interface. Αυτό που θα μπορεί να προστεθεί είναι πριν κάνω export να είχε κάτι σαν verification αν και δεν έχει νόημα γιατί αυτό θα γίνει ύστερα στο compile. Αυτό το πρόγραμμα πρέπει να είναι συντακτικά σωστό εξ ορισμού, γιατί δεν μου επιτρέπει να βάλω πράγματα που δεν ταιριάζουν.
- Θα μπορούσε εάν αφήσω ένα block και δεν συνδέσω κάτι επάνω του –όπως το int που δηλώνεις μεταβλητή- να μου το επισημαίνει κάπως. Για παράδειγμα να μου το κοκκινίζει.

- Θα μπορούσαν οι κατηγορίες των blocks να είχαν υποκατηγορίες, όπως για παράδειγμα το «Operators» να είχε υποκατηγορίες τα «Arithmetic» και «Logic».

Ερώτηση 11:

«*Θα το χρησιμοποιούσατε στο μάθημα ή στο εργαστήριο;*»

- Θα μπορούσε να χρησιμοποιηθεί για να παρουσιάσει παραδείγματα στην τάξη. Έτσι, θα μπορούσα να εξηγήσω καλύτερα την λογική δομή. Μπορεί να χρησιμοποιηθεί για να αντιπαραβάλλω παραδείγματα. Αντιλαμβάνεσαι γραφικά την δομή πιο γρήγορα.

Συμπεράσματα

Με βάση τις παραπάνω απαντήσεις και το ενδιαφέρον και τον ενθουσιασμό που έδειξαν οι χρήστες αυτής της αξιολόγησης, πράγμα που δεν μπορεί να καταγραφεί από κανένα ηλεκτρονικό μέσο παρά από τα μάτια του αξιολογητή, φαίνεται ότι το εργαλείο δημιούργησε θετικές εντυπώσεις.

Οι ειδικοί χρήστες, στη C και στη διδασκαλία της, φάνηκαν ικανοποιημένοι από το εργαλείο και επεσήμαναν σφάλματα της γραφικής του διεπαφής τα οποία διορθώθηκαν στη συνέχεια. Η δυνατότητα του εργαλείου να κρατάει τους χρήστες όσο το δυνατόν πιο μακριά από τα συντακτικά λάθη, τονίστηκε και παρατηρήθηκε και από τους τρεις χρήστες.

Κεφάλαιο 6
Διαδικασία Αξιολόγησης της αποτελεσματικότητας του εργαλείου

Περίληψη

Σε αυτό το κεφάλαιο αναλύεται η αξιολόγηση της αποτελεσματικότητας του εργαλείου. Περιγράφονται η διαδικασία, οι αποφάσεις και οι στόχοι της τρίτης αξιολόγησης.

Εισαγωγή

Αυτή η αξιολόγηση σχεδιάστηκε έτσι ώστε να μας προσφέρει πληροφορίες πέρα από τα «απτά» θέματα της ευχρηστίας. Εφόσον παράχθηκε ένα λειτουργικό πρωτότυπο του εργαλείου έπρεπε να εξεταστεί η αποτελεσματικότητα του, δηλαδή η δυνατότητα του εργαλείου να να βοηθήσει τους εκπαιδευόμενους να καταφέρουν πράγματα τα οποία δεν θα καταφέρναν μόνοι τους. Έτσι ουσιαστικά θα φανεί αν το εργαλείο εισάγει τους χρήστες στην Ζώνη Επικείμενης Ανάπτυξης τους (ZPD Διάγραμμα 2.3) χωρίς την διαμεσολάβηση ανθρώπων ως βοηθούς.

Έτσι, σχεδιάστηκε ένα μεγάλης κλίμακας πείραμα με την συμμετοχή τριάντα δύο (32) φοιτητών. Οι συνθήκες κάτω από τις οποίες πραγματοποιήθηκε το πείραμα προσπαθήσαμε να προσομοιώνουν όσο το δυνατόν καλύτερα τις πραγματικές συνθήκες χρήσης του εργαλείου. Έτσι το πείραμα πραγματοποιήθηκε σε εργαστηριακό χώρο με αρχάριους στη C και γενικότερα στον προγραμματισμό χρήστες.

Διαδικασία

Ο καθηγητής του τμήματος των Μηχανικών Παραγωγής και Διοίκησης που δίδασκε το αντίστοιχο μάθημα, κύριος Δουλάμης Αναστάσιος, προσφέρθηκε να ενημερώσει τους πρωτοετείς φοιτητές για το πείραμα. Η συμμετοχή των φοιτητών ήταν προαιρετική και συνολικά δήλωσαν συμμετοχή τριάντα δύο άτομα. Τέλος, με απόφαση των εργαστηριακών βοηθών του μαθήματος, δόθηκε μία μονάδα επιπρόσθετα στην τελική βαθμολογία σε όλους όσους θα συμμετείχαν στο πείραμα, έτσι ώστε να δοθεί ένα κίνητρο για την συμμετοχή.

Το 40% των χρηστών ήταν γυναίκες και σχεδόν όλοι οι συμμετάσχοντες είχαν λίγη έως ελάχιστη εμπειρία με τον προγραμματισμό και την γλώσσα C. Ακολουθεί ένας πίνακας με μια σύνοψη των «προφίλ» των χρηστών που έλαβαν μέρος στο πείραμα.

Πίνακας 6.1: Πίνακας με μία σύντομη καταγραφή των προφίλ των χρηστών της τρίτης αξιολόγησης. «Καθόλου» δήλωσαν οι φοιτητές που δεν είχαν πρότερη επαφή με τον προγραμματισμό. «Λίγο» δήλωσαν οι φοιτητές που είχαν λίγη εμπειρία με τον προγραμματισμό (δευτεροβάθμια εκπαίδευση) αλλά καθόλου με τη γλώσσα C. Τέλος «Αρκετή» δήλωσε ένας φοιτητής που είχε εμπειρία και με τον προγραμματισμό αλλά και με την γλώσσα C.

	Καθόλου	Λίγο	Αρκετή	Σύνολο	Ποσοστό
Άρρεν	6	12	1	19	60%
Θήλυ	4	9	0	13	40%
Σύνολο	10	21	1	32	100%

Το πείραμα πραγματοποιήθηκε ως ένα «ελεγχόμενο πείραμα» (controlled experiment) [33]. Έτσι οι φοιτητές χωρίστηκαν σε δύο ομάδες εργασίας. Η πρώτη ομάδα αποτελούσε την «ομάδα πειραματισμού» (test group) και χρησιμοποιούσε το Block-C και η δεύτερη αποτελούσε την «ομάδα ελέγχου» (control group) και εργάστηκε με απλή C.

Η διαδικασία χωρίστηκε σε δύο ενότητες των δύο ωρών η κάθε μία.

1η ενότητα – εισαγωγή στο πείραμα

Η πρώτη ενότητα ήταν ουσιαστικά μια ενότητα εξοικείωσης με το πείραμα αλλά και με τα εργαλεία (Block-C και το IDE για τον προγραμματισμό στην κειμενική C).

Στην αρχή της ενότητας δόθηκαν κάποιες γενικές κατευθυντήριες γραμμές για τις εργασίες που θα είχαν οι χρήστες να εκπονήσουν και μια σύντομη παρουσίαση των εργαλείων.

Στην συνέχεια οι χρήστες κλήθηκαν να εκπονήσουν δέκα εργασίες κλιμακούμενης δυσκολίας. Οι χρήστες είχαν την δυνατότητα ανά πάσα στιγμή να διακόψουν όποια εργασία τους δυσκόλευε ή τους απογοήτευε και να προχωρήσουν στην επόμενη.

Τους τονίστηκε ότι δεν αξιολογούνταν οι ίδιοι αλλά το εργαλείο, έτσι ώστε να μην αγχωθούν και αισθανθούν ότι κρίνονται. Καθ' όλη την διάρκεια της αξιολόγησης υπήρχαν τέσσερις εργαστηριακοί βοηθοί, εξειδικευμένοι στη C, οι οποίοι βοηθούσαν τους χρήστες με οποιαδήποτε απορία τους και κρατούσαν στατιστικά για τους χρόνους ολοκλήρωσης κάθε εργασίας για κάθε φοιτητή.

Το έντυπο που περιέγραφε αναλυτικά τις εργασίες που κλήθηκαν να εκπονήσουν οι χρήστες είναι το εξής:

Περιγραφή Αξιολόγησης Block-C

Γενικά

- Κατά την διάρκεια της αξιολόγησης θα μετράται ο χρόνος που κάνετε για να ολοκληρώσετε κάθε «εργασία».
- Θα υπάρχουν βοηθοί που θα σας καθοδηγούν και θα λύνουν τις απορίες σας.
- Εάν δε τα καταφέρνετε σε κάποια «εργασία» μπορείτε να πάτε σε επόμενη, μετά από συνεννόηση με τους βοηθούς.
- Ο χρόνος της αξιολόγησης είναι 2 ώρες. Όποιος τελειώσει νωρίτερα μπορεί να φύγει

Διαδικασία

Οι εργασίες που καλείστε να κάνετε είναι οι εξής:

1. Φτιάξτε ένα πρόγραμμα που να εκτυπώνει την φράση «Hello World!»
2. Φτιάξτε ένα πρόγραμμα που διαβάζει έναν ακέραιο από το πληκτρολόγιο, τον αποθηκεύει σε μια αντίστοιχη μεταβλητή, και τον εκτυπώνει.
3. Φτιάξτε ένα πρόγραμμα το οποίο θα διαβάζει από το πληκτρολόγιο το ύψος και τη βάση ενός τριγώνου και θα υπολογίζει και θα εκτυπώνει το εμβαδόν του.
4. Φτιάξτε ένα πρόγραμμα που θα ελέγχει άν ο χαρακτήρας που διάβασε από τον χρήστη είναι ίσος με «a». Αν ναι θα εκτυπώνει αντίστοιχο μήνυμα.
5. Φτιάξτε ένα πρόγραμμα που θα διαβάζει 2 αριθμούς από το πληκτρολόγιο, θα τους αποθηκεύει σε 2 μεταβλητές. Έπειτα θα αντιστρέφει τις τιμές τους και θα τις εκτυπώνει.
    - Π.Χ. *αριθμός_1* = 10 *αριθμός_2* =5 μετά την αντιστροφή θα εκτυπώνει: *αριθμός_1* = 5 , *αριθμός_2* = 10
6. Φτιάξτε ένα πρόγραμμα που θα υπολογίζει και θα εκτυπώνει το άθροισμα των αριθμών από το 1 έως και το 100.
7. Φτιάξτε ένα πρόγραμμα που θα υπολογίζει και θα εκτυπώνει το παραγοντικό* ενός αριθμού που θα εισάγει ο χρήστης
    - * παραγοντικό: $(n)! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot \ldots \cdot n\text{-}1 \cdot n$
8. Φτιάξτε ένα πρόγραμμα που θα διαβάζει δύο ακεραίους από το πληκτρολόγιο.
    - Αν και οι 2 είναι μεγαλύτεροι του 10 θα εκτυπώνει αντίστοιχο μήνυμα
    - Αν και οι 2 είναι μικρότεροι του 10 θα εκτυπώνει αντίστοιχο μήνυμα
    - Αν 1 από τους 2 είναι μεγαλύτερος του 10 θα εκτυπώνει αντίστοιχο μήνυμα
9. Φτιάξτε ένα πρόγραμμα που θα διαβάζει έναν ακέραιο από το πληκτρολόγιο μέχρι ο χρήστης να δώσει την τιμή 1
    - Υπόδειξη: χρήση While επανάληψης (loop)
10. Φτιάξτε ένα πρόγραμμα που ο χρήστης θα δίνει 3 πλευρές ενός τριγώνου και το σύστημα θα του λέει άν είναι οξυγώνιο, αμβλυγώνιο ή ορθογώνιο
    - Υπόδειξη:
        i. **Αν**: $A^2 + B^2 = C^2$ **τότε** =>ορθογώνιο .
        ii. **Αν**: $A^2 + B^2 > C^2$ **τότε** =>οξυγώνιο.
        iii. **Αν**: $A^2 + B^2 < C^2$ **τότε** => αμβλυγώνιο.
11. Συμπλήρωση ερωτηματολογίου

Εικόνα 6.2: Το έντυπο περιγραφής της διαδικασίας και των εργασιών του πρώτου μέρους της τρίτης αξιολόγησης

2η ενότητα – εισαγωγή στο πείραμα

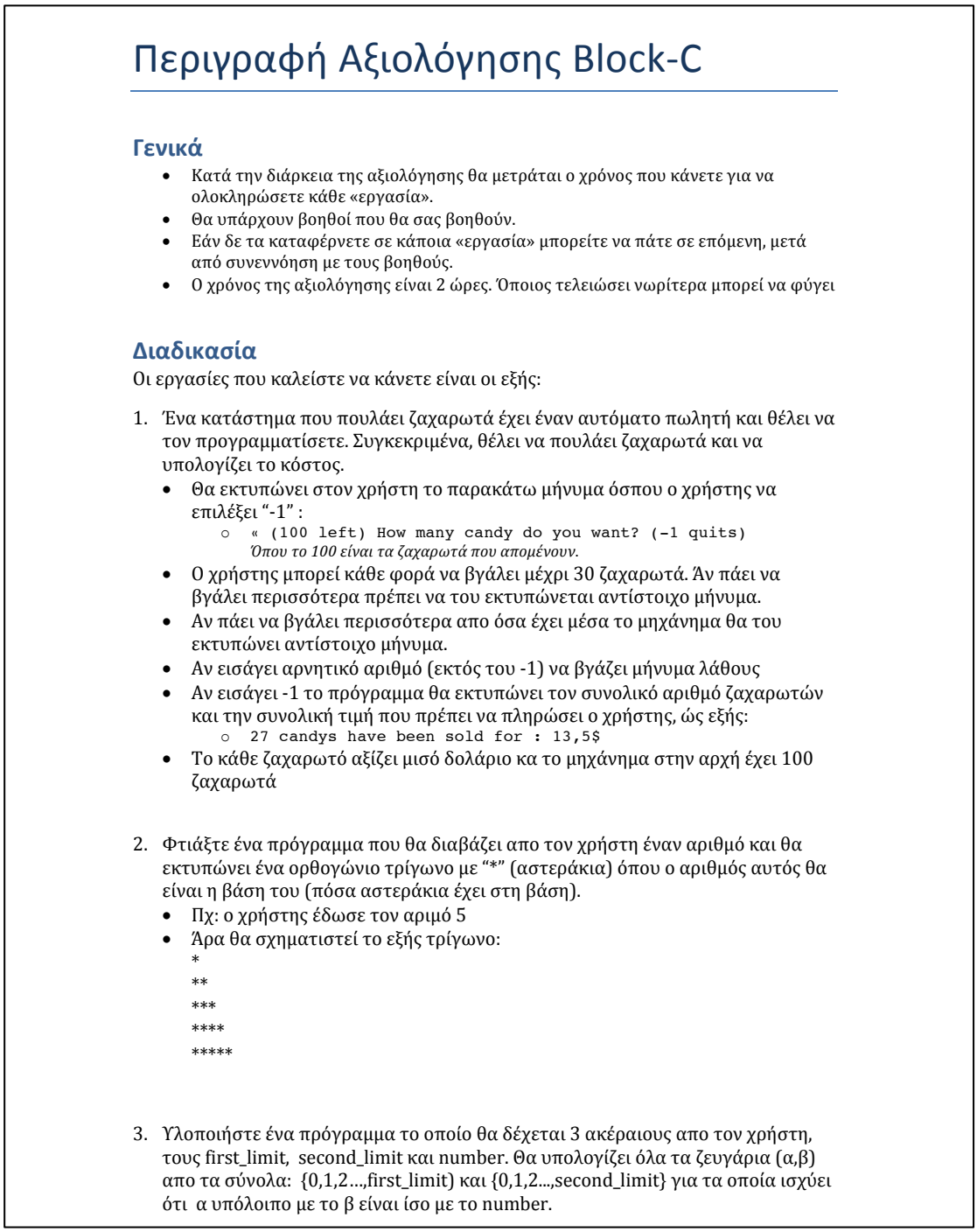
Η δεύτερη ενότητα ήταν ουσιαστικά το μέρος το πειράματος το οποίο αποσκοπούσε την εξαγωγή συμπερασμάτων. Σε αυτή την ενότητα οι χρήστες κλήθηκαν να αντεπεξέλθουν σε πιο δύσκολες εργασίες χωρίς την καθοδήγηση των βοηθών.

Ο αριθμός των εργασιών αυτή την φορά ήταν τρεις αλλά η δυσκολία ήταν αυξημένη. Οι εργασίες χρειαζόταν τον συνδυασμό διαφορετικών λογικών μονάδων (πχ. εντολές ελέγχου και επανάληψης) για να επιτευχθεί ο στόχος που έθετε έκαστη.

Όπως και στην προηγούμενη ενότητα έτσι και σε αυτήν ,τέσσερις εργαστηριακοί βοηθοί, εξειδικευμένοι στη C, κρατούσαν στατιστικά για τους χρόνους ολοκλήρωσης κάθε εργασίας για κάθε φοιτητή. Η βοήθεια που παρείχαν στους χρήστες ήταν περιορισμένη μόνο σε πολύ σοβαρά λογικά προβλήματα ή λάθη.

Έτσι, θα φαινόταν ξεκάθαρα αν το Block-C θα βοηθούσε τους χρήστες να εισαχθούν στο ZPD χωρίς την διαμεσολάβηση ανθρώπων. Επίσης θα διαφαίνονταν η αρνητική επίπτωση που έχουν τα συντακτικά λάθη στους χρήστες.

Το έντυπο που περιέγραφε αναλυτικά τις εργασίες που κλήθηκαν να εκπονήσουν οι χρήστες είναι το εξής:



Περιγραφή Αξιολόγησης Block-C

Γενικά

- Κατά την διάρκεια της αξιολόγησης θα μετράται ο χρόνος που κάνετε για να ολοκληρώσετε κάθε «εργασία».
- Θα υπάρχουν βοηθοί που θα σας βοηθούν.
- Εάν δε τα καταφέρνετε σε κάποια «εργασία» μπορείτε να πάτε σε επόμενη, μετά από συνεννόηση με τους βοηθούς.
- Ο χρόνος της αξιολόγησης είναι 2 ώρες. Όποιος τελειώσει νωρίτερα μπορεί να φύγει

Διαδικασία

Οι εργασίες που καλείστε να κάνετε είναι οι εξής:

1. Ένα κατάστημα που πουλάει ζαχαρωτά έχει έναν αυτόματο πωλητή και θέλει να τον προγραμματίσετε. Συγκεκριμένα, θέλει να πουλάει ζαχαρωτά και να υπολογίζει το κόστος.
   - Θα εκτυπώνει στον χρήστη το παρακάτω μήνυμα όσπου ο χρήστης να επιλέξει "-1" :
     - « (100 left) How many candy do you want? (-1 quits)
       Όπου το 100 είναι τα ζαχαρωτά που απομένουν.
   - Ο χρήστης μπορεί κάθε φορά να βγάλει μέχρι 30 ζαχαρωτά. Άν πάει να βγάλει περισσότερα πρέπει να του εκτυπώνεται αντίστοιχο μήνυμα.
   - Αν πάει να βγάλει περισσότερα απο όσα έχει μέσα το μηχάνημα θα του εκτυπώνει αντίστοιχο μήνυμα.
   - Αν εισάγει αρνητικό αριθμό (εκτός του -1) να βγάζει μήνυμα λάθους
   - Αν εισάγει -1 το πρόγραμμα θα εκτυπώνει τον συνολικό αριθμό ζαχαρωτών και την συνολική τιμή που πρέπει να πληρώσει ο χρήστης, ώς εξής:
     - 27 candys have been sold for : 13,5$
   - Το κάθε ζαχαρωτό αξίζει μισό δολάριο κα το μηχάνημα στην αρχή έχει 100 ζαχαρωτά

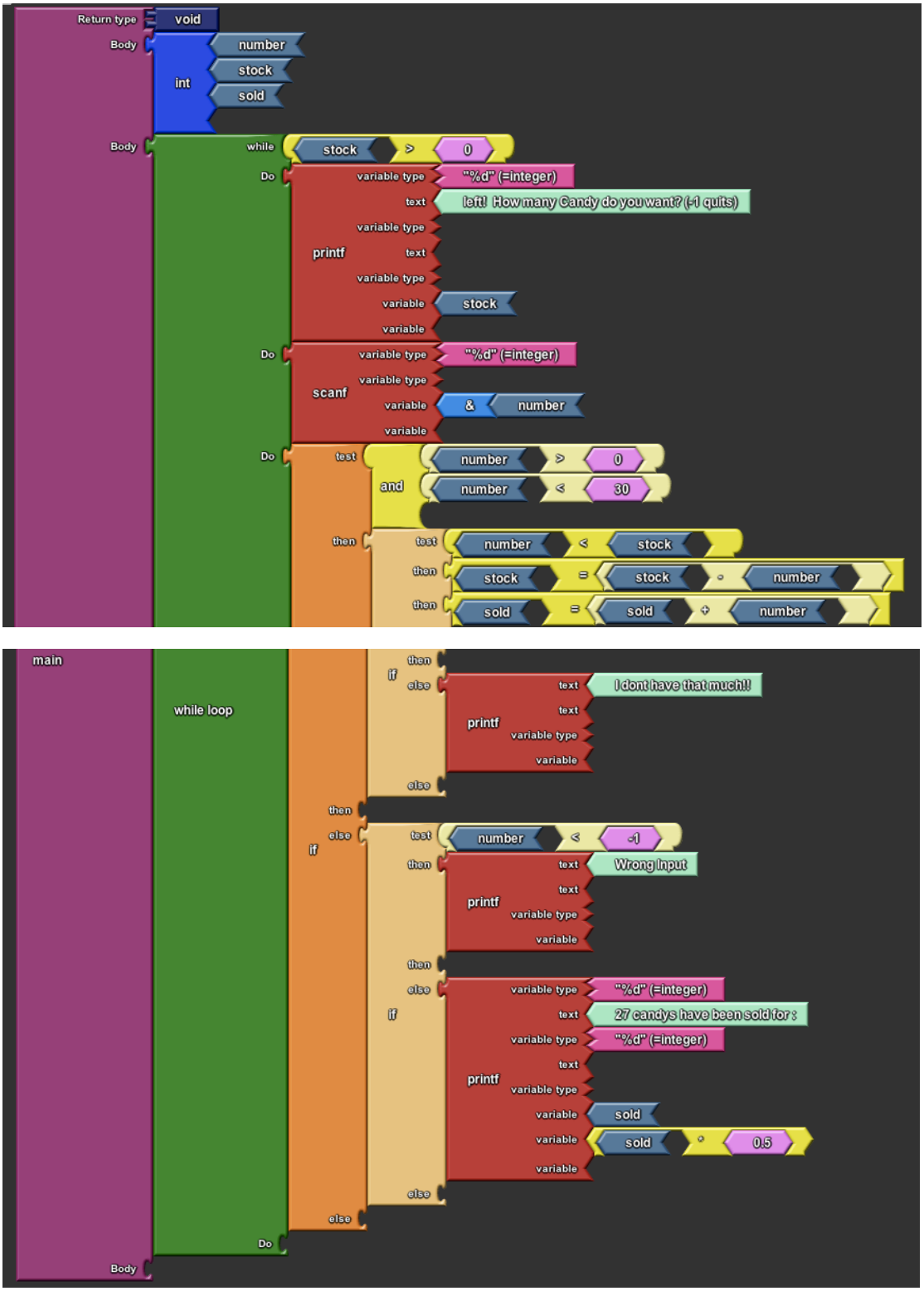
2. Φτιάξτε ένα πρόγραμμα που θα διαβάζει απο τον χρήστη έναν αριθμό και θα εκτυπώνει ένα ορθογώνιο τρίγωνο με "*" (αστεράκια) όπου ο αριθμός αυτός θα είναι η βάση του (πόσα αστεράκια έχει στη βάση).
   - Πχ: ο χρήστης έδωσε τον αριμό 5
   - Άρα θα σχηματιστεί το εξής τρίγωνο:
     *
     **
     ***
     ****
     *****

3. Υλοποιήστε ένα πρόγραμμα το οποίο θα δέχεται 3 ακέραιους απο τον χρήστη, τους first_limit, second_limit και number. Θα υπολογίζει όλα τα ζευγάρια (α,β) απο τα σύνολα: {0,1,2…,first_limit) και {0,1,2...,second_limit} για τα οποία ισχύει ότι α υπόλοιπο με το β είναι ίσο με το number.

Εικόνα 6.3: Το έντυπο περιγραφής της διαδικασίας και των εργασιών του δεύτερου μέρους της τρίτης αξιολόγησης

Οι παραπάνω εργασίες επιλέχθηκαν ώστε να καλύπτουν ένα σύνολο από βασικές έννοιες του προγραμματισμού σε C, όπως οι επαναλήψεις, οι έλεγχοι κλπ. Έτσι η πρώτη εργασία απαιτούσε τη δημιουργία ενός καταλόγου επιλογών (menu) εντός του οποίου ο χρήστης θα έπρεπε να ελέγξει αρκετές διαφορετικές περιπτώσεις και να χρησιμοποιήσει μια δομή επανάληψης *while*. Η δεύτερη για να λυθεί χρειαζόταν δύο επαναληπτικές δομές η μία εμφωλευμένη μέσα στην άλλη. Τέλος, το τρίτο χρειαζόταν πάλι εμφωλευμένες επαναλήψεις και την χρήση ενός

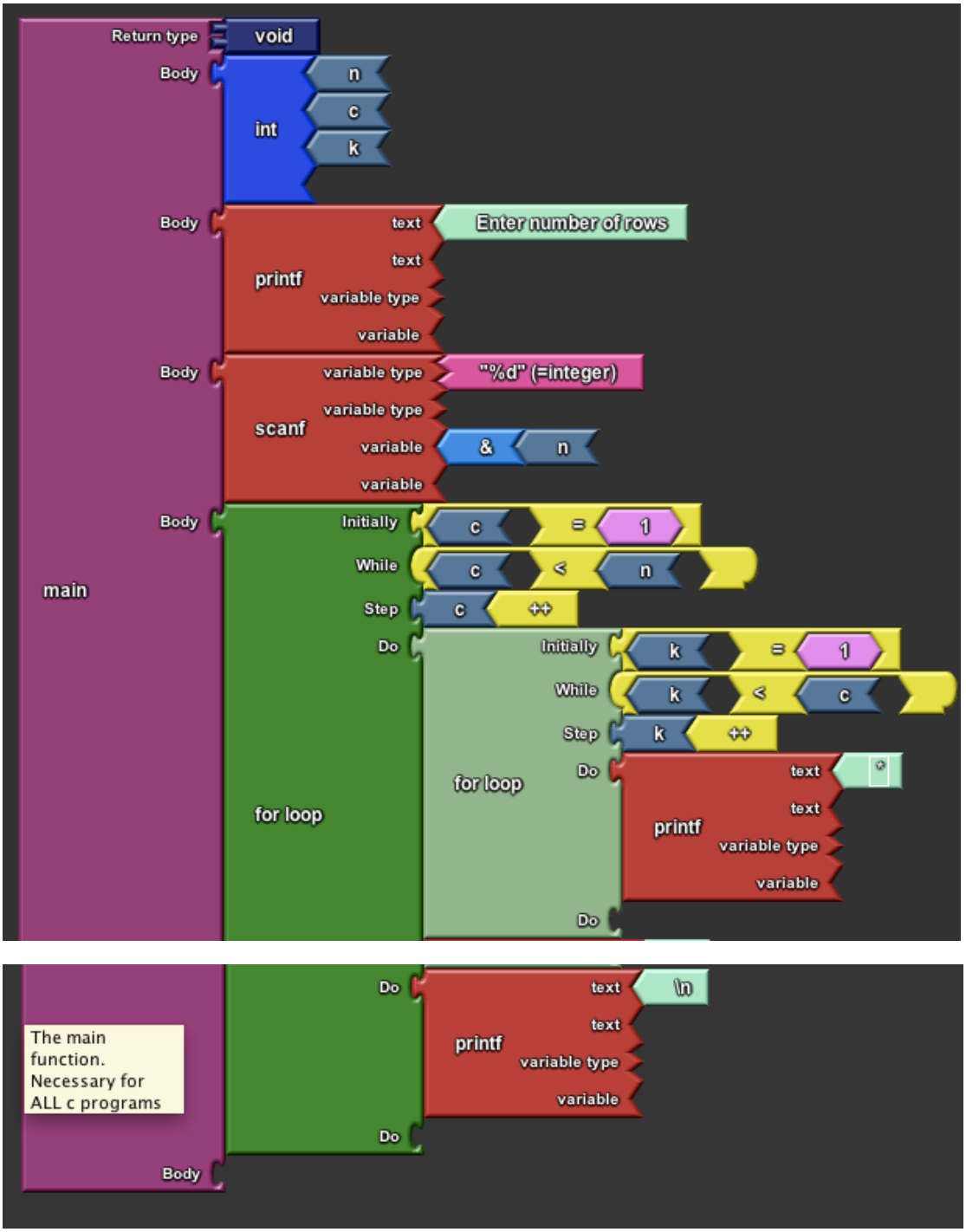
προηγμένου τελεστή.Η λύση των παραπάνω εργασιών φαίνεται στις εικόνες 6.4 έως 6.10.



Εικόνα 6.4: Η λύση της πρώτης εργασίας της αξιολόγησης σε Block-C

```c
#include <stdio.h>
void main()
{
    int number, stock, sold ;
    while(stock>0)
    {
        printf("%d left!  How many Candy do you want? (-1 quits)",stock);
        scanf("%d",&number);
        if(number>0 && number<30)
        {
            if(number<stock)
            {
                stock=(stock-number);
                sold=(sold+number);
            }
            else
            {
                printf("I dont have that much!!");

            }
        }
        else
        {
            if(number<-1)
            {
                printf("Wrong Input");
            }
            else
            {
                printf("%d 27 candys have been sold for : %d",sold,sold*0.5);

            }

        }

    }

}
```

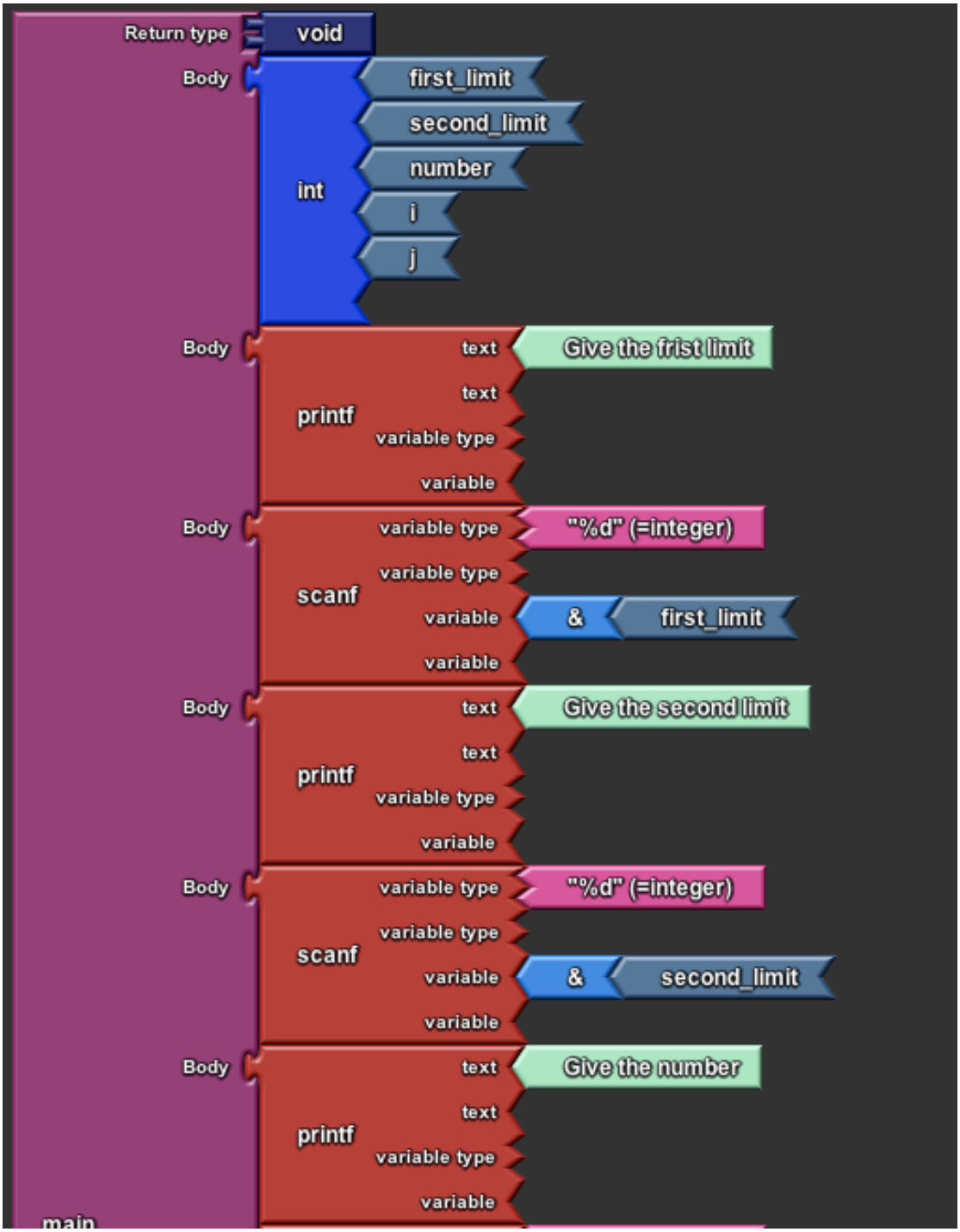
Πίνακας 6.5: Η λύση της πρώτης εργασίας της αξιολόγησης σε C



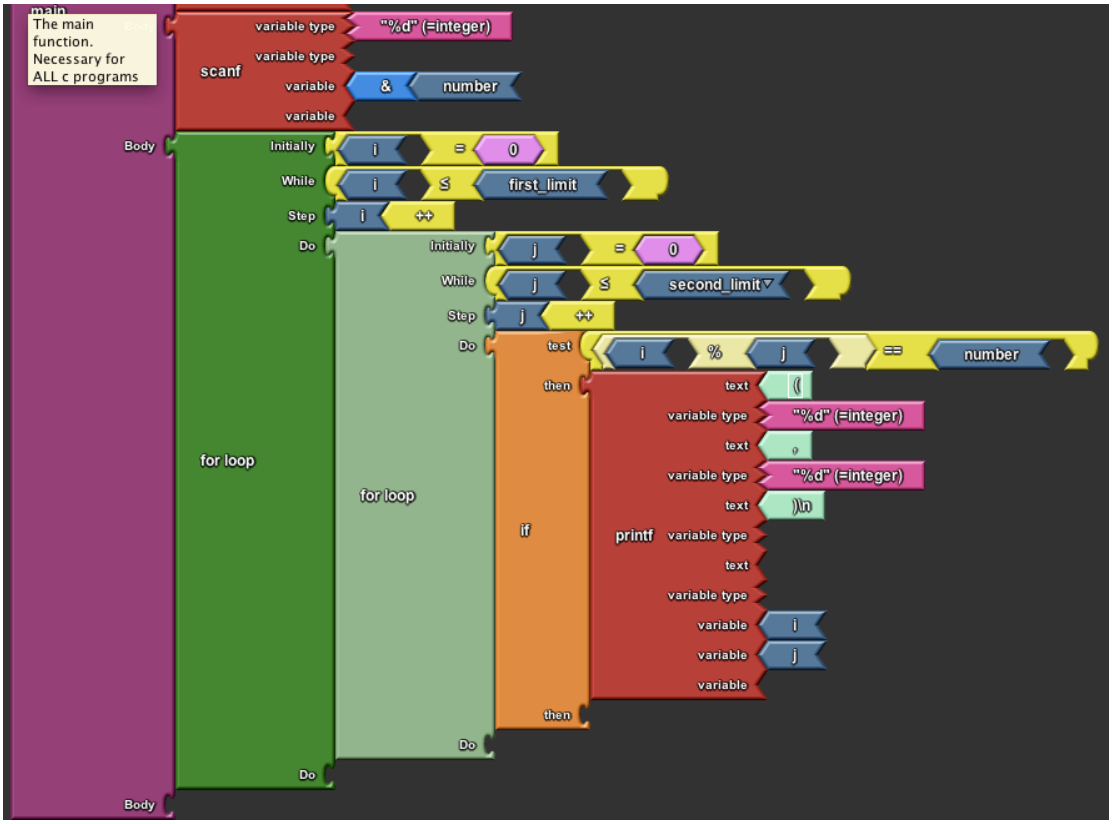
Εικόνα 6.6: Η λύση της δεύτερης εργασίας της αξιολόγησης σε Block-C

```c
#include <stdio.h>

int main()
{
    int n, c, k;

    printf("Enter number of rows\n");
    scanf("%d",&n);

    for ( c = 1 ; c <= n ; c++ )
    {
        for( k = 1 ; k <= c ; k++ )
            printf("*");

        printf("\n");
    }

    return 0;
}
```

Πίνακας 6.7: Η λύση της δεύτερης εργασίας της αξιολόγησης σε C



Εικόνα 6.8: Το πρώτο κομμάτι της λύσης της τρίτης εργασίας της αξιολόγησης σε Block-C



Εικόνα 6.9: Το δεύτερο κομμάτι της λύσης της τρίτης εργασίας της αξιολόγησης σε Block-C

```c
#include <stdio.h>
void main()
{
    int first_limit, second_limit, number, i, j ;
    printf("Give the frist limit");
    scanf("%d",&first_limit);
    printf("Give the second limit");
    scanf("%d",&second_limit);
    printf("Give the number");
    scanf("%d",&number);

    for(i=0;i<=first_limit;i++)
    {
        for(j=0;j<=second_limit;j++)
        {
            if((i%j)==number)
            {
                printf("( %d , %d )\n",i,j);

            }

        }

    }

}
```

Πίνακας 6.10: Η λύση της τρίτης εργασίας της αξιολόγησης σε C

Μετά το πέρας της διαδικασίας οι χρήστες της ομάδας πειραματισμού κλήθηκαν να συμπληρώσουν ένα ερωτηματολόγιο. Το ερωτηματολόγιο ήταν σχεδιασμένο ώστε να καλύπτει θέματα που άπτονται της αντιληπτής ,από τους χρήστες, χρησιμότητας και ευχρηστίας του προγράμματος [34]. Επίσης, ένα τμήμα του αφορούσε την ικανοποίηση / ευχαρίστηση που τους παρείχε το εργαλείο[35].

Το ερωτηματολόγιο που συμπλήρωσαν οι χρήστες είναι το εξής:

Ερωτηματολόγιο Αξιολόγησης BLOCK-C

Το παρών ερωτηματολόγιο είναι ανώνυμο. Προσπαθεί να καταγράψει την άποψη των χρηστών που πήραν μέρος στην διαδικασία της αξιολόγησης του BLOCK-C. Σε καμία από τις ερωτήσεις δεν είναι υποχρεωτική η απάντηση.

Τμήμα:..... Φύλο:..... Εμπειρία στον προγραμματισμό:……

Γενική γνώμη για το Block-C:

Χάλια	0	1	2	3	4	5	6	7	8	9	Καταπληκτικό
Δύσκολο	0	1	2	3	4	5	6	7	8	9	Εύκολο
Απογοητευτικό	0	1	2	3	4	5	6	7	8	9	Ικανοποιητικό
Βαρετό	0	1	2	3	4	5	6	7	8	9	Ενδιαφέρον

Οθώνη:

1)

Τα Block στην οθώνη ήταν:

Δυσανάγνωστα	0	1	2	3	4	5	6	7	8	9	Ευκολοδιάβαστα

2)

Η οργάνωση της πληροφορίας:

προκαλούσε σύγχυση	0	1	2	3	4	5	6	7	8	9	πολύ ξεκάθαρη

Ορολογία:

3)

Η ορολογία που χρησιμοποιήθηκε ήταν σχετική με αυτό που κάνετε;

Ποτέ	0	1	2	3	4	5	6	7	8	9	Πάντα

4)

Ανακαλύπτεις νέες δυνατότητες μέσω πειραματισμού και λαθών

Δύσκολα	0	1	2	3	4	5	6	7	8	9	Εύκολα

5)

Καταλαβαίνες την έννοια των ονομάτων, και την σημασία των χρωμάτων και σχημάτων;

Δύσκολα	0	1	2	3	4	5	6	7	8	9	Εύκολα

Εικόνα 6.11: Πρώτη σελίδα ερωτηματολογίου που κλήθηκαν να συμπληρώσουν οι χρήστες μετά το πέρας της διαδικασίας της τρίτης αξιολόγησης

6)

Η υλοποίηση ενος προγράμματος εκτελείται με απλό/άμεσο τρόπο

Ποτέ 0 1 2 3 4 5 6 7 8 9 Πάντα

<u>Δυνατότητες του συστήματος</u>:

7)

Ταχύτητα του Block-C:

πολύ αργό 0 1 2 3 4 5 6 7 8 9 πολύ γρήγορο

8)

Το να διορθώσεις τα λάθη σου ήταν:

Δύσκολο 0 1 2 3 4 5 6 7 8 9 Εύκολο

<u>Δημιουργικότητα και εκπαίδευση</u>:

9)

Θα εμπνεόσασταν να χρησιμοποιήσετε το Block-C για να δημιουργήσετε ένα πρόγραμμα:

Απίθανο 0 1 2 3 4 5 6 7 8 9 Βέβαιο

10)

Θα σας άρεσε να σας δείχνουν παραδείγματα στο μάθημα φτιαγμένα με το Block-C μας;

Καθόλου 0 1 2 3 4 5 6 7 8 9 Πολύ

11)

Πιστεύετε ότι θα μπορούσε να χρησιμοποιηθεί το Block-C για εκπαιδευτικούς σκοπούς;

Δύσκολα 0 1 2 3 4 5 6 7 8 9 Εύκολα

12)

Πιστεύετε ότι το Block-C μας μπορεί να ενθαρρύνει αρχάριους να ασχοληθούν με τον προγραμματισμό;

Καθόλου 0 1 2 3 4 5 6 7 8 9 Πολύ

<u>Χρησιμότητα</u>:

13)

Το να χρησιμοποιώ το Block-C θα με βοηθούσε να φτιάχνω προγράμματα πιο γρήγορα

Καθόλου 0 1 2 3 4 5 6 7 8 9 Πολύ

<u>Εικόνα 6.12</u>: Δεύτερη σελίδα ερωτηματολογίου που κλήθηκαν να συμπληρώσουν οι χρήστες μετά το πέρας της διαδικασίας της τρίτης αξιολόγησης

14) *Χρησιμοποιώντας το Block-C θα βελτίωνα την απόδοση μου στον προγραμματισμό σε C*

Απίθανο 0 1 2 3 4 5 6 7 8 9 Βέβαιο

15)

Χρησιμοποιώντας το Block-C θα προγραμμάτιζα πιο αποτελεσματικά

Απίθανο 0 1 2 3 4 5 6 7 8 9 Βέβαιο

16)

Χρησιμοποιώντας το Block-C θα προγραμμάτιζα σε C πιο εύκολα

Απίθανο 0 1 2 3 4 5 6 7 8 9 Βέβαιο

17)

Θα έβρισκα το Block-C εύχρηστο για να προγραμματίσω σε C

Καθόλου 0 1 2 3 4 5 6 7 8 9 Πολύ

18)

Χρησιμοποιώντας το Block-C για να φτιάχνω προγράμματα θα αξιοποιούσα καλύτερα τον χρόνο μου

Απίθανο 0 1 2 3 4 5 6 7 8 9 Βέβαιο

Ευκολία στη χρήση:

19)

Το να μάθω να χρησιμοποιώ το Block-C θα ήταν εύκολο για μένα

Καθόλου 0 1 2 3 4 5 6 7 8 9 Πολύ

20)

Θα έβρισκα εύκολο να καταφέρω με το Block-C να κάνω αυτό που θέλω

Καθόλου 0 1 2 3 4 5 6 7 8 9 Πολύ

21)

Η αλληλεπίδραση μου με το Block-C θα ήταν ξεκάθαρη και κατανοητή

Καθόλου 0 1 2 3 4 5 6 7 8 9 Πολύ

22)

Θα εύρισκα το Block-C ευέλικτο στην αλληλεπίδραση μαζί του

Καθόλου 0 1 2 3 4 5 6 7 8 9 Πολύ

23)

Θα μου ήταν εύκολο να γίνω ειδικός στη χρήση του Block-C

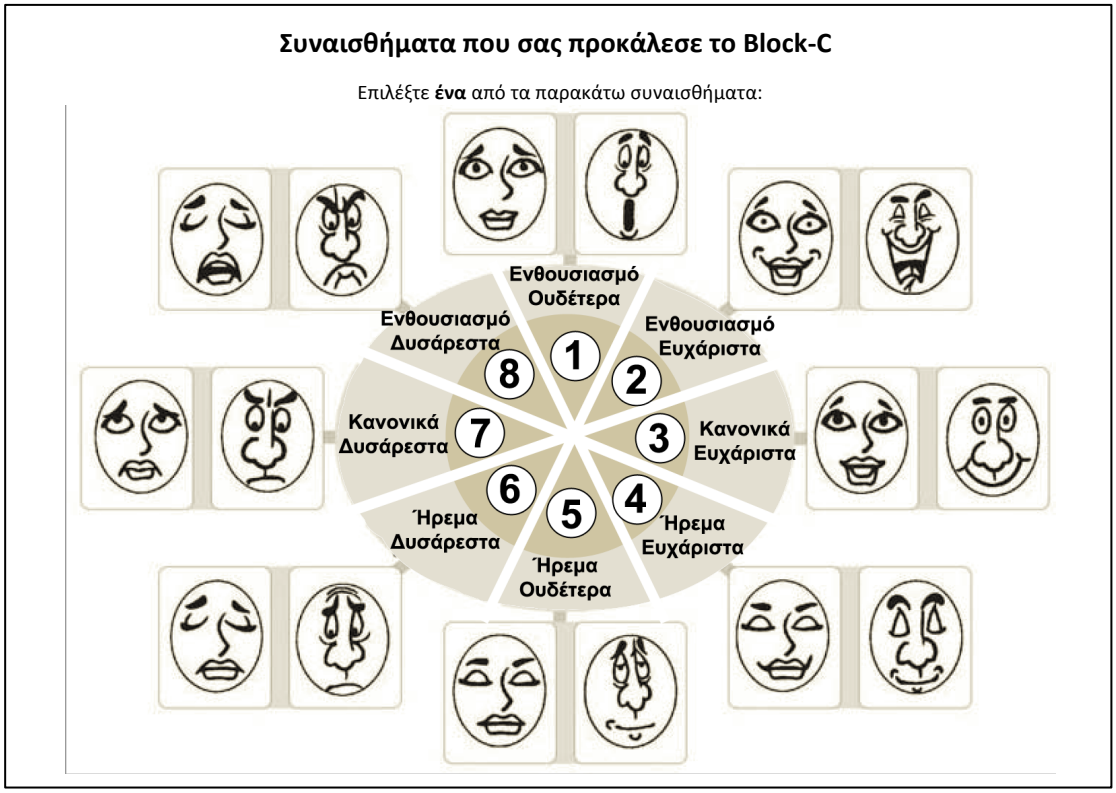
Απίθανο 0 1 2 3 4 5 6 7 8 9 Βέβαιο

24)

θα έβρισκα το Block-C εύκολο στη χρήση

Καθόλου 0 1 2 3 4 5 6 7 8 9 Πολύ

Εικόνα 6.13: Τρίτη σελίδα ερωτηματολογίου που κλήθηκαν να συμπληρώσουν οι χρήστες μετά το πέρας της διαδικασίας της τρίτης αξιολόγησης



Εικόνα 6.14: Τρίτη σελίδα ερωτηματολογίου, που περιείχε την ερώτηση για το συναίσθημα που προκάλεσε το Block-C κατά την χρήση του, που κλήθηκαν να συμπληρώσουν οι χρήστες μετά το πέρας της διαδικασίας της τρίτης αξιολόγησης [36]

Ερωτήσεις Ελεύθερης Απάντησης:

Πως θα περιγράφατε το Block-C σε ένα φίλο σας;

Ποιο/α ήταν το χαρακτηριστικό/ά που σας άρεσε πιο πολύ στο BLOCK-C και γιατί;

Κατά την άποψη σας τι θα μπορούσε να προστεθεί ή τι λείπει από το BLOCK-C;

Ευχαριστούμε για τον χρόνο σας!

Εικόνα 6.15: Τέταρτη και τελευταία σελίδα ερωτηματολογίου, με ερωτήσεις ελεύθερης απάντησης, που κλήθηκαν να συμπληρώσουν οι χρήστες μετά το πέρας της διαδικασίας της τρίτης αξιολόγησης

Κεφάλαιο 7
Αποτελέσματα 3ής αξιολόγησης

Περίληψη

Σε αυτό το κεφάλαιο γίνεται μια εκτενής παρουσίαση των αποτελεσμάτων της τρίτης και τελευταίας διαδικασίας αξιολόγησης. Επίσης εξάγονται τα συμπεράσματα από την αξιολόγηση.

Εισαγωγή

Η τρίτη αξιολόγηση χωρίστηκε σε δύο ενότητες. Η πρώτη ενότητα αποτελούσε εισαγωγική ενότητα στο πείραμα. Η δεύτερη ενότητα αποτελούσε το πείραμα καθεαυτό. Σε αυτό το κεφάλαιο παρουσιάζονται τα αποτελέσματα αυτών των δύο ενοτήτων.

Τα αποτελέσματά και των δύο ενοτήτων και παρουσιάζονται παρακάτω γραφικά και με την μορφή πινάκων. Τα αποτελέσματα της πρώτης, εισαγωγικής, ενότητας δεν ήταν ενθαρρυντικά για το εργαλείο μας. Σε όλες τις εργασίες η ομάδα ελέγχου πέτυχε καλύτερους χρόνους. Το γεγονός αυτό οφείλετε στο ότι η ομάδα που δούλευε με το Block-C έπρεπε παράλληλα με το πείραμα να εξοικειωθεί και με το εργαλείο. Έτσι τα αποτελέσματα αυτής της ενότητας δεν μπορούν να θεωρηθούν ως αξιόπιστα προς σύγκριση. Επίσης, σε αυτήν την ενότητα οι βοηθοί λειτουργούσαν επικουρικά προς κάθε απορία και πρόβλημα.

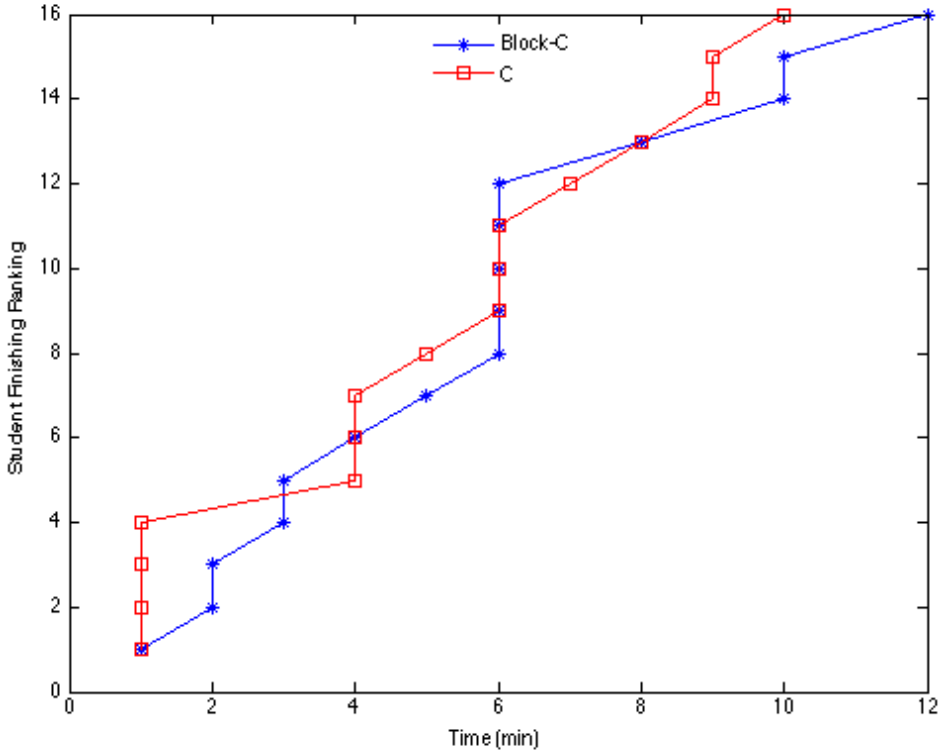
Στην δεύτερη ενότητα οι χρήστες είχαν πλέον αποκτήσει μια βασική εμπειρία με το εργαλείο. Επίσης, οι βοηθοί επενέβαιναν πλέον μόνο σε προχωρημένα θέματα λογικής του προγραμματισμού.

Κατά την δεύτερη ενότητα θα φαινόταν η διαφορά μεταξύ των υποβοηθούμενων από το εργαλείο και των μη υποβοηθούμενων. Όπως έχει προαναφερθεί η Ζώνη Επικείμενης Ανάπτυξης ερμηνεύεται ως η διαφορά μεταξύ του τι μπορεί να πετύχει ο μαθητευόμενος μόνος του και τι όταν υποβοηθείται.

Άρα αν υπήρχε θετική διαφορά (καλύτεροι χρόνοι και περισσότερες ολοκληρωμένες εργασίες για τους υποβοηθούμενους) κατά την εξαγωγή των αποτελεσμάτων οι, υποβοηθουμενοι από το εργαλείο, χρήστες θα είχαν εισαχθεί στην Ζώνη Επικείμενης Ανάπτυξής τους.

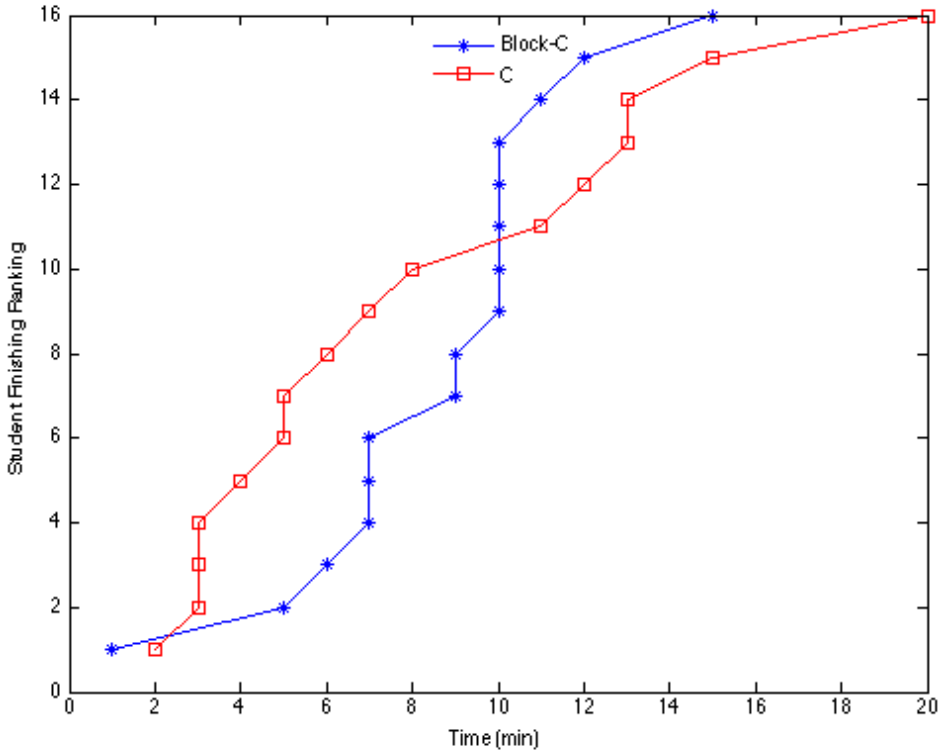
Ουσιαστικά όλο το τρίτο πείραμα αξιολόγησης ήταν σχεδιασμένο για την εξαγωγή αποτελεσμάτων από την δεύτερη ενότητα, γι΄ αυτό και η πρώτη ενότητα περιγράφεται ως «εισαγωγική».

Αποτελέσματα πρώτης «εισαγωγικής» ενότητας



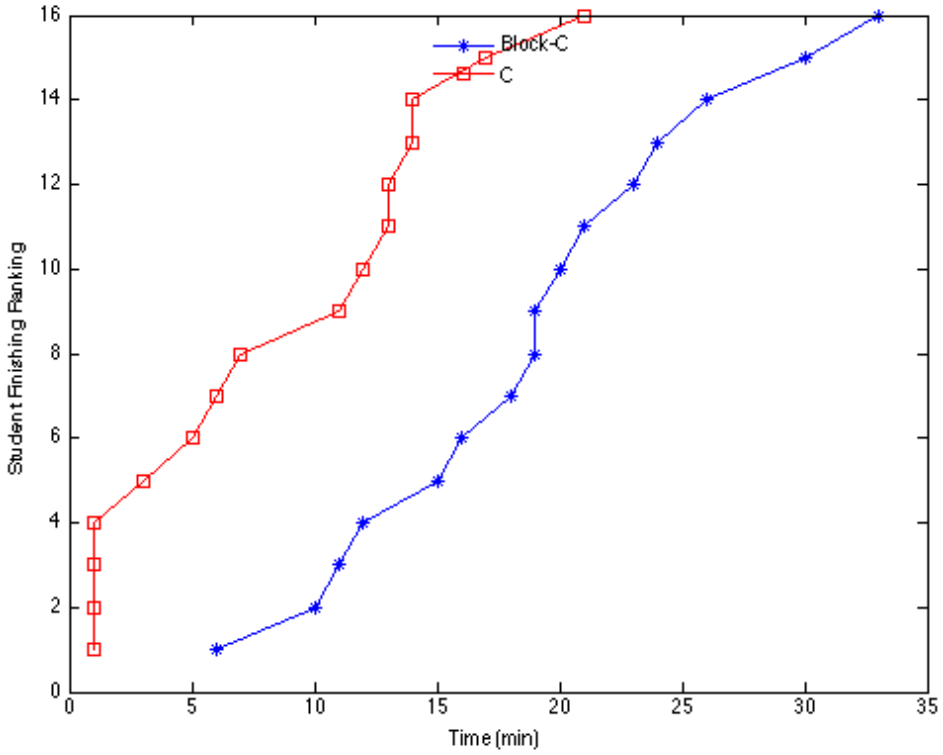
Διάγραμμα 7.1: Αποτελέσματα 1ης εργασίας της πρώτης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.1 παρατηρούμε τα αποτελέσματα για την πρώτη εργασία. Ο αριθμός των φοιτητών που ολοκλήρωσε την εργασία είναι ίδιος και για τις δύο ομάδες. Παρατηρούμε ότι οι φοιτητές που δούλεψαν με την C ήταν γενικά λίγο πιο «γρήγοροι» από αυτούς που δούλεψαν με το εργαλείο Block-C.



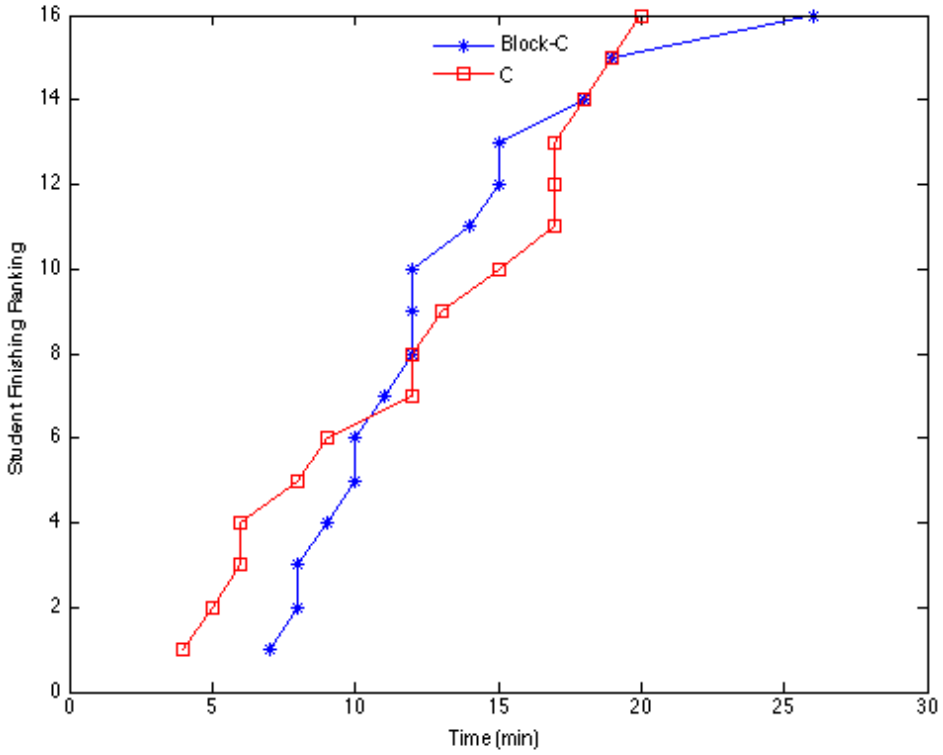
Διάγραμμα 7.2: Αποτελέσματα 2ης εργασίας της πρώτης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.2 παρατηρούμε τα αποτελέσματα για τη δεύτερη εργασία. Ο αριθμός των φοιτητών που ολοκλήρωσε την εργασία είναι ίδιος και για τις δύο ομάδες. Από το παρόν διάγραμμα φαίνεται ότι και οι δύο ομάδες τα πήγαν εξίσου καλά (οι πρώτοι δέκα φοιτητές με τη C ήταν καλύτεροι από αυτούς με το Block-C αλλά στου επόμενους έξι υπερτερεί το Block-C).



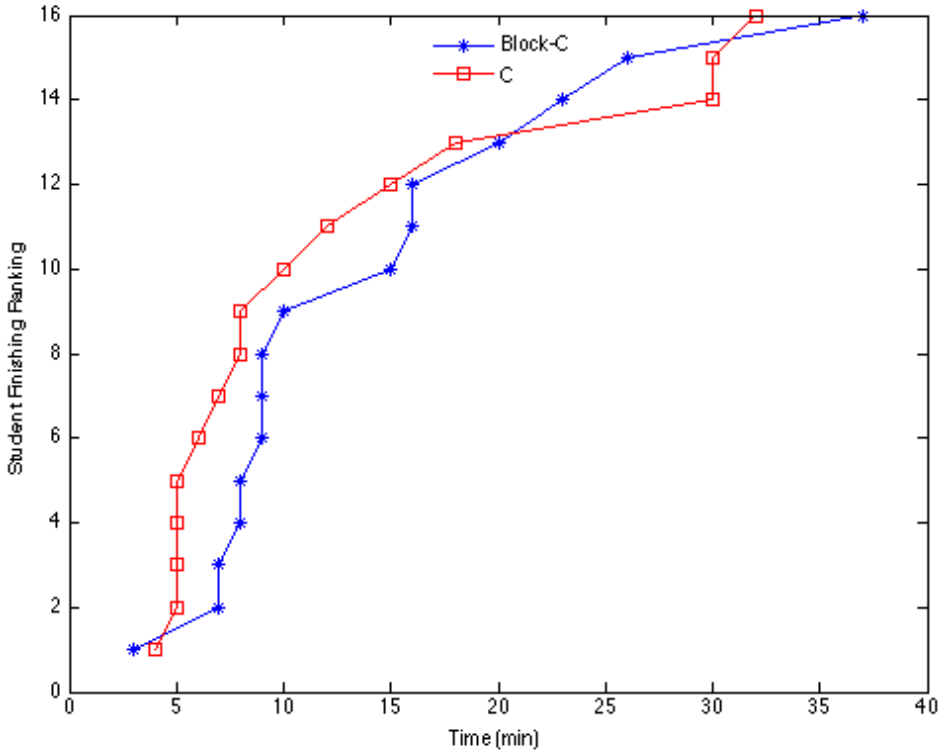
Διάγραμμα 7.3: Αποτελέσματα 3ης εργασίας της πρώτης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.3 παρατηρούμε τα αποτελέσματα για τη τρίτη εργασία. Ο αριθμός των φοιτητών που ολοκλήρωσε την εργασία είναι ίδιος και για τις δύο ομάδες. Από το παρόν διάγραμμα φαίνεται ξεκάθαρα ότι η ομάδα που δούλευε με τη C τα πήγε σαφώς καλύτερα, επειδή όλοι οι φοιτητές τελειώσαν πιο γρήγορα.



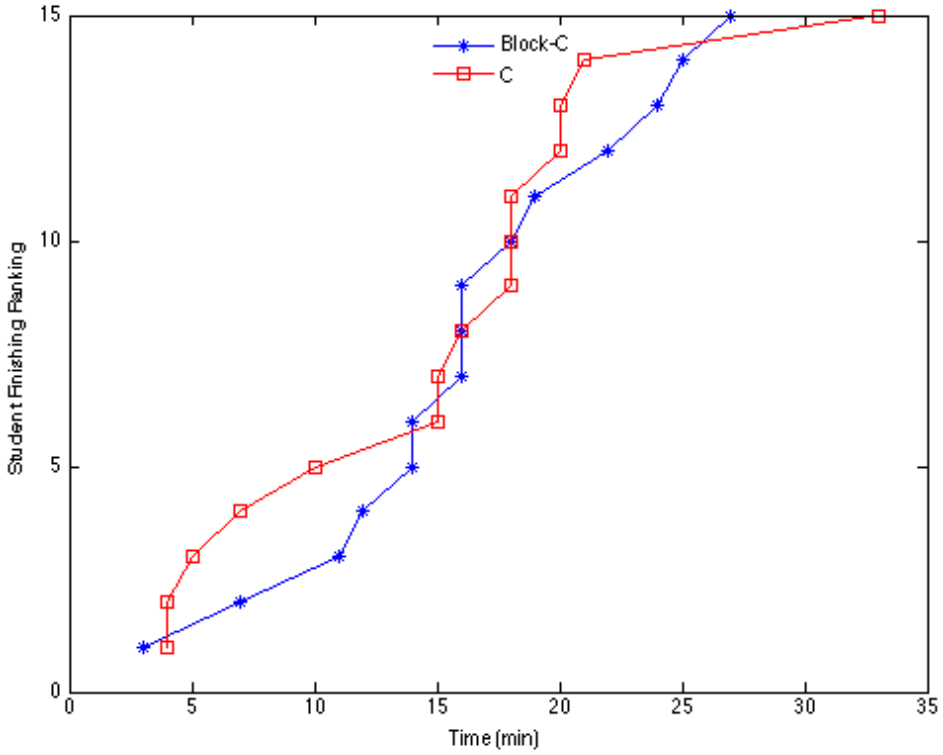
Διάγραμμα 7.4: Αποτελέσματα 4ης εργασίας της πρώτης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.4 παρατηρούμε τα αποτελέσματα για τη τέταρτη εργασία. Ο αριθμός των φοιτητών που ολοκλήρωσε την εργασία είναι ίδιος και για τις δύο ομάδες. Από το παρόν διάγραμμα φαίνεται ότι και οι δύο ομάδες τα πήγαν εξίσου καλά (οι πρώτοι έξι φοιτητές με τη C ήταν καλύτεροι από αυτούς με το Block-C αλλά στου επόμενους εφτά υπερτερεί το Block-C και στους υπόλοιπους τα αποτελέσματα είναι παρόμοια).



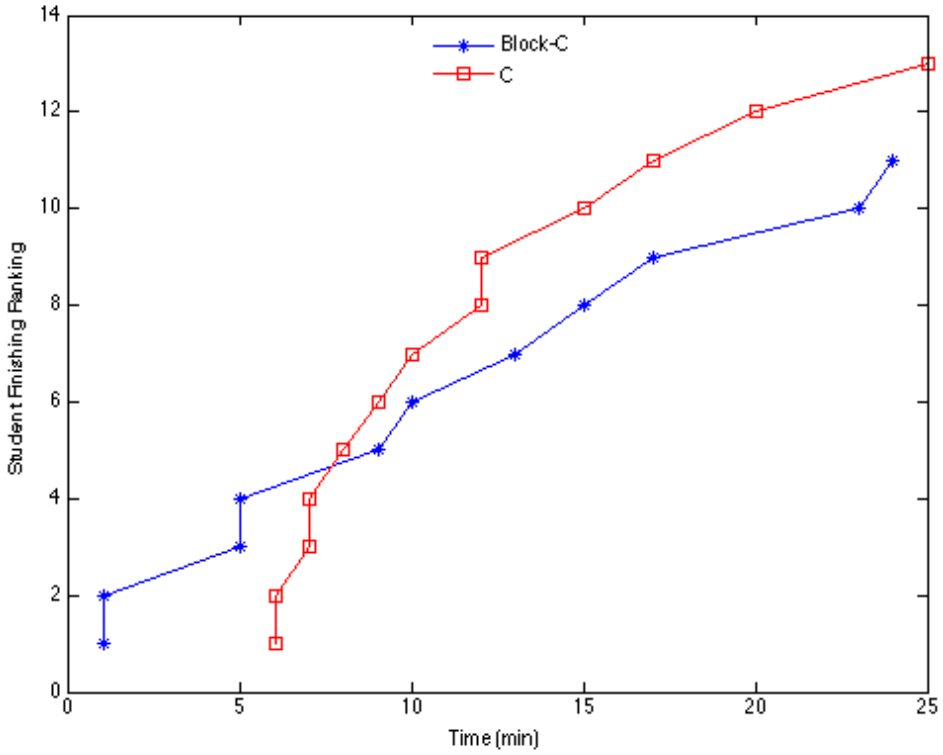
Διάγραμμα 7.5: Αποτελέσματα 5ης εργασίας της πρώτης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.5 παρατηρούμε τα αποτελέσματα για τη πέμπτη εργασία. Ο αριθμός των φοιτητών που ολοκλήρωσε την εργασία είναι ίδιος και για τις δύο ομάδες. Από το παρόν διάγραμμα φαίνεται ότι η ομάδα που δούλευε με τη C τα πήγε καλύτερα σε γενικές γραμμές, (εκτός των τριών τελευταίων φοιτητών).



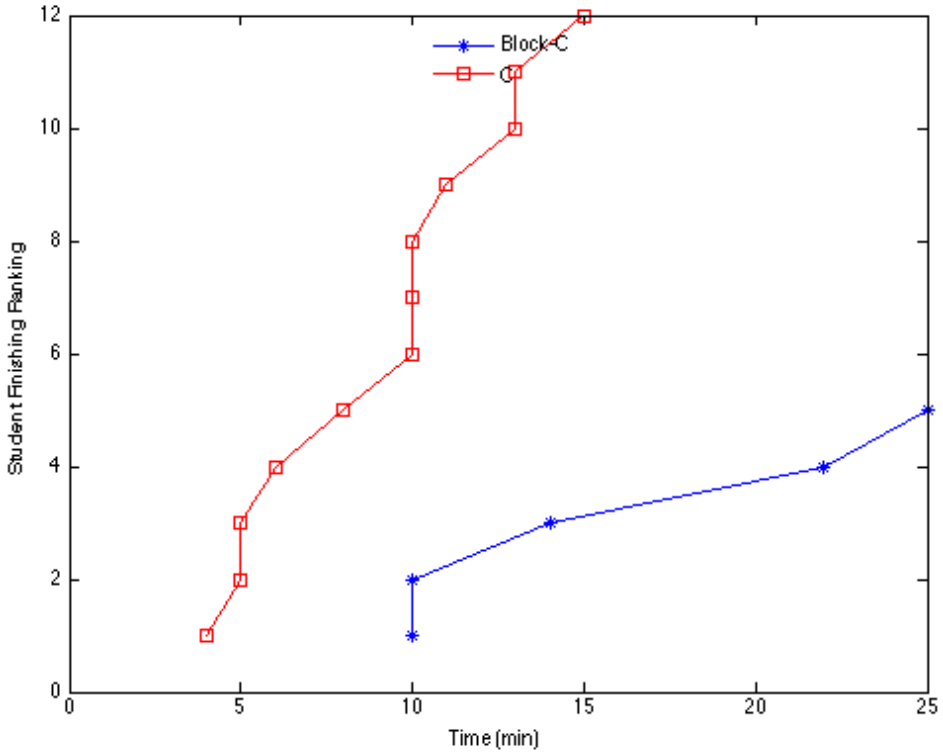
Διάγραμμα 7.6: Αποτελέσματα 6ης εργασίας της πρώτης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.6 παρατηρούμε τα αποτελέσματα για τη έκτη εργασία. Ο αριθμός των φοιτητών που ολοκλήρωσε την εργασία είναι ίδιος και για τις δύο ομάδες. Από το παρόν διάγραμμα φαίνεται ότι και οι δύο ομάδες είχαν πάνω κάτω παρόμοια αποτελέσματα.



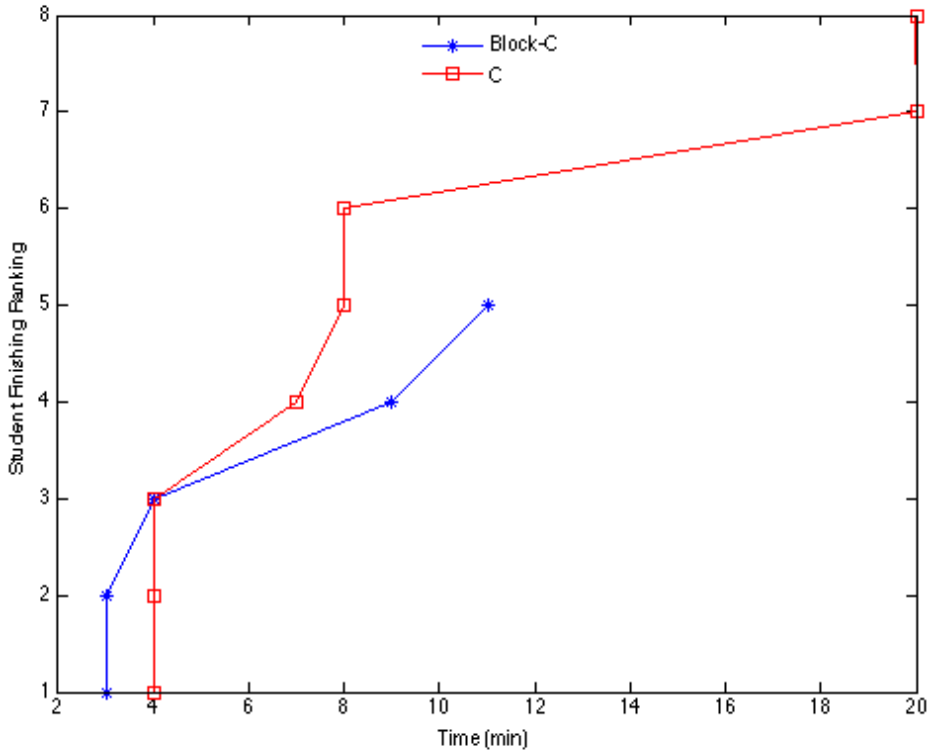
Διάγραμμα 7.7: Αποτελέσματα 7ης εργασίας της πρώτης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.7 παρατηρούμε τα αποτελέσματα για τη έβδομη εργασία. Την εργασία αυτή ολοκλήρωσαν 13 φοιτητές από την ομάδα της C έναντι 11 από την ομάδα του Block-C. Από το παρόν διάγραμμα φαίνεται επίσης ότι εκτός από τους τέσσερις πρώτους οι υπόλοιποι φοιτητές με C ολοκλήρωσαν γρηγορότερα από αυτούς του Block-C.



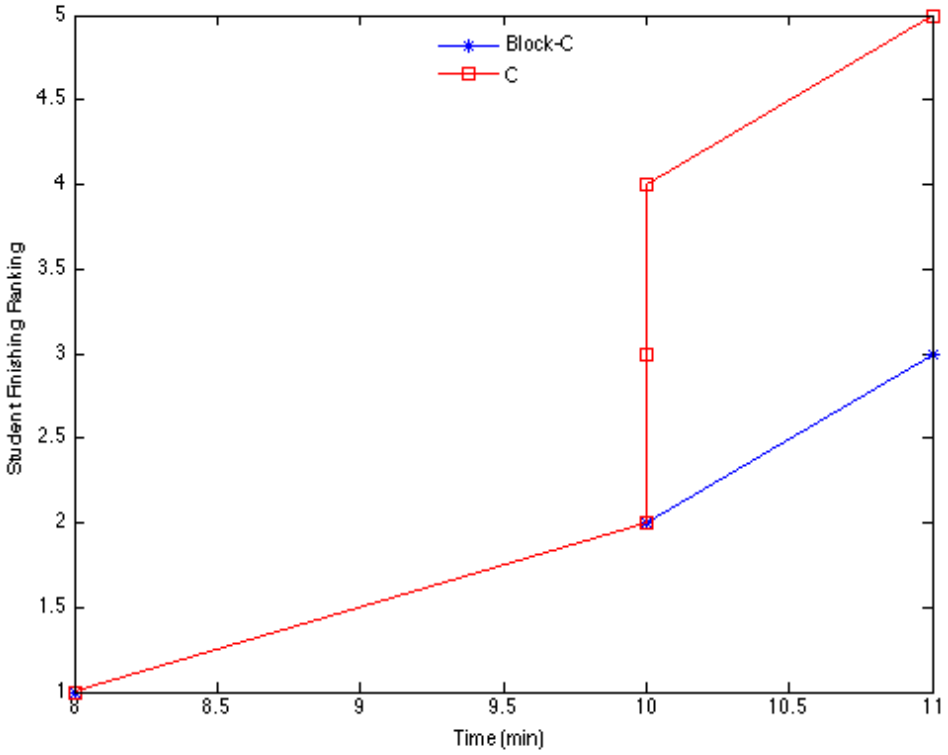
Διάγραμμα 7.8: Αποτελέσματα 8ης εργασίας της πρώτης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.8 παρατηρούμε τα αποτελέσματα για τη όγδοη εργασία. Την εργασία αυτή ολοκλήρωσαν 12 φοιτητές από την ομάδα της C έναντι 5 από την ομάδα του Block-C. Από το παρόν διάγραμμα φαίνεται ξεκάθαρα ότι οι φοιτητές που δούλεψαν με C τα πήγαν σαφώς καλύτερα από τους συναδέλφους τους που δούλεψαν με Block-C.



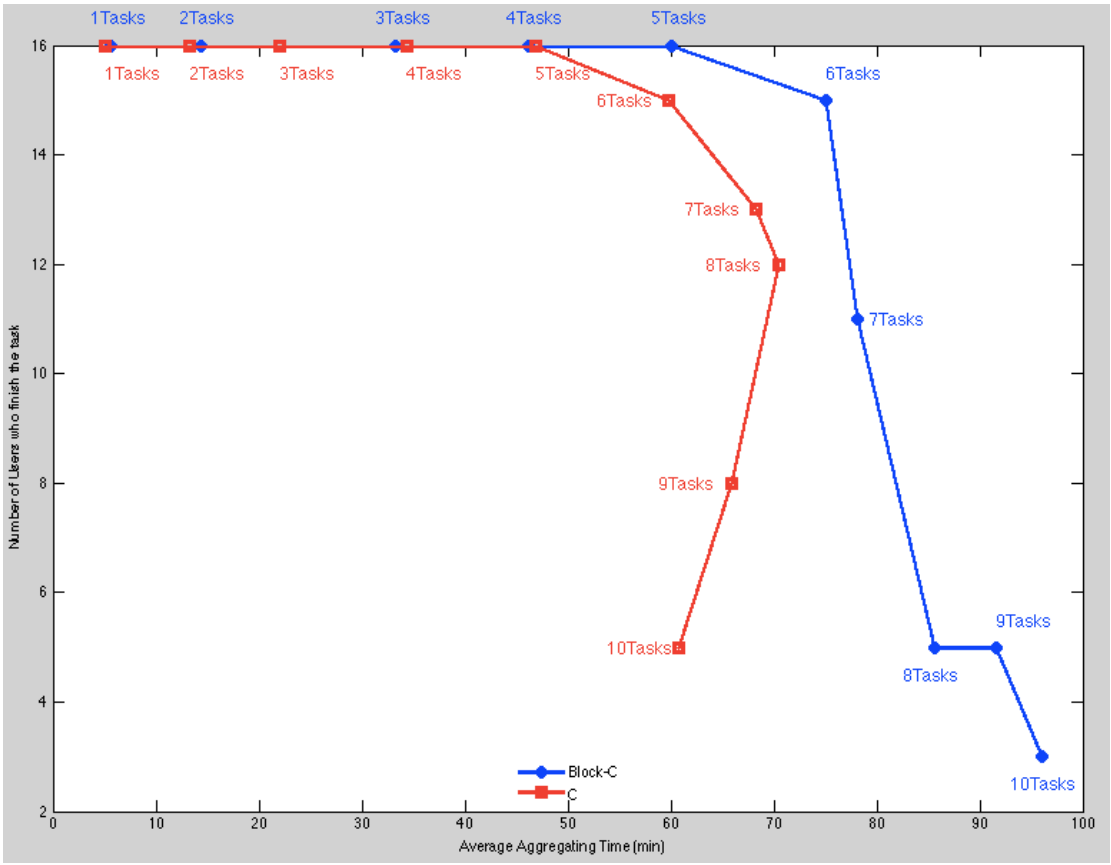
Διάγραμμα 7.9: Αποτελέσματα 9ης εργασίας της πρώτης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.9 παρατηρούμε τα αποτελέσματα για τη ένατη εργασία. Την εργασία αυτή ολοκλήρωσαν 8 φοιτητές από την ομάδα της C έναντι 5 από την ομάδα του Block-C. Από το παρόν διάγραμμα φαίνεται επίσης ότι εκτός από τους δύο πρώτους οι υπόλοιποι φοιτητές με C ολοκλήρωσαν γρηγορότερα από αυτούς του Block-C.



Διάγραμμα 7.10: Αποτελέσματα 10ης εργασίας της πρώτης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.10 παρατηρούμε τα αποτελέσματα για τη δέκατη και τελευταία εργασία. Την εργασία αυτή ολοκλήρωσαν 5 φοιτητές από την ομάδα της C έναντι 3 από την ομάδα του Block-C. Από το παρόν διάγραμμα φαίνεται ότι και οι δύο ομάδες είχαν ανάλογα αποτελέσματα όσο αφορά τον χρόνο ολοκλήρωσης της συγκεκριμένης εργασίας.



Διάγραμμα 7.11: Αθροιστικοί μέσοι όροι ολοκλήρωσης μίας, δύο ... έως δέκα εργασιών. Για παράδειγμα ο μέσος χρόνος ολοκλήρωσης της 3ης εργασίας είναι το άθροισμα των μέσων της 1ης της 2ης και της 3ης εργασίας. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο αθροιστικό μέσος χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα ο αριθμός των φοιτητών που ολοκλήρωσαν τον συγκεκριμένο αριθμό εργασιών. Το τέρμα αριστερά και επάνω σημείο δηλώνει την ολοκλήρωση μίας εργασίας· προχωρώντας ευθεία και έπειτα με βάση τη φορά του ρολογιού είναι τα στοιχεία για την ολοκλήρωση δύο ... έως δέκα εργασιών.

Στο διάγραμμα 7.11 παρατηρούμε τα αποτελέσματα για τους αθροιστικούς μέσους όρους για την ολοκλήρωσης μίας έως δέκα εργασιών. Η ομάδα των φοιτητών που δούλεψε με τη C φαίνεται ξεκάθαρα ότι υπερτερεί και σε χρόνο ολοκλήρωσης και σε αριθμό φοιτητών που ολοκλήρωσαν τις εργασίες.

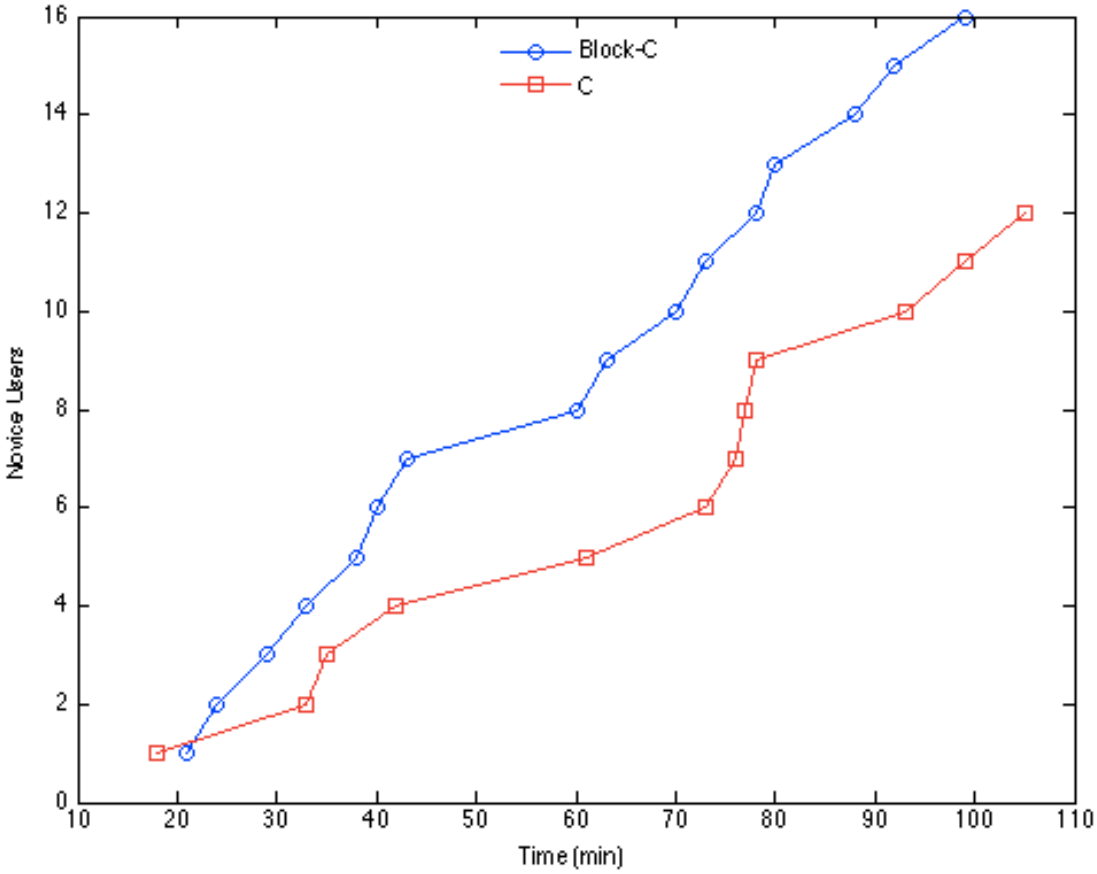
Συμπεράσματα

Τα συμπεράσματα αυτής της ενότητας ήταν αναμενόμενα και λογικά. Οι φοιτητές της ομάδας ελέγχου που δούλεψαν με C είχαν ένα σαφές πλεονέκτημα. Η συγκεκριμένη ομάδα δεν χρειάζονταν να μάθει κάποιο εργαλείο και οι φοιτητές που ανήκαν σε αυτή αφιερώθηκαν κατευθείαν στην επίλυση των εργασιών που τους ανατέθηκαν.

Από την άλλη, οι χρήστες του Block-C (τα μέλη της ομάδας πειραματισμού) έπρεπε παράλληλα με την εκπόνηση των εργασιών να μάθουν να χειρίζονται και το εργαλείο. Έτσι δεν μπορούσαν να αφιερωθούν εξ' ολοκλήρου στην επίλυση των εργασιών. Αποτέλεσμα ήταν να έχουν γενικά μειωμένη απόδοση.

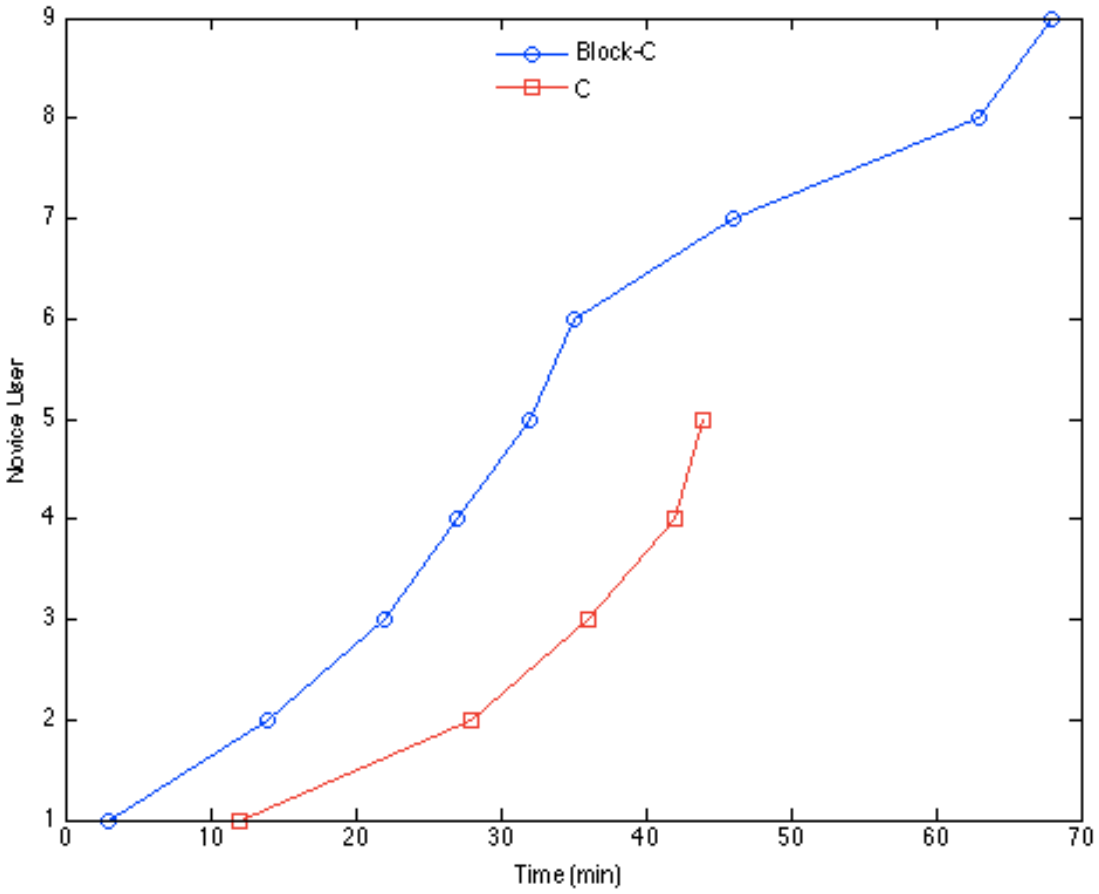
Έτσι διαφαίνεται η αναγκαιότητα της δεύτερης ενότητας η οποία διεξάγεται κάτω από συνθήκες ίσων όρων, εφόσον οι φοιτητές της ομάδας πειραματισμού έχουν πλέον αποκτήσει με εξοικείωση με το εργαλείο Block-C.

Αποτελέσματα δεύτερης ενότητας



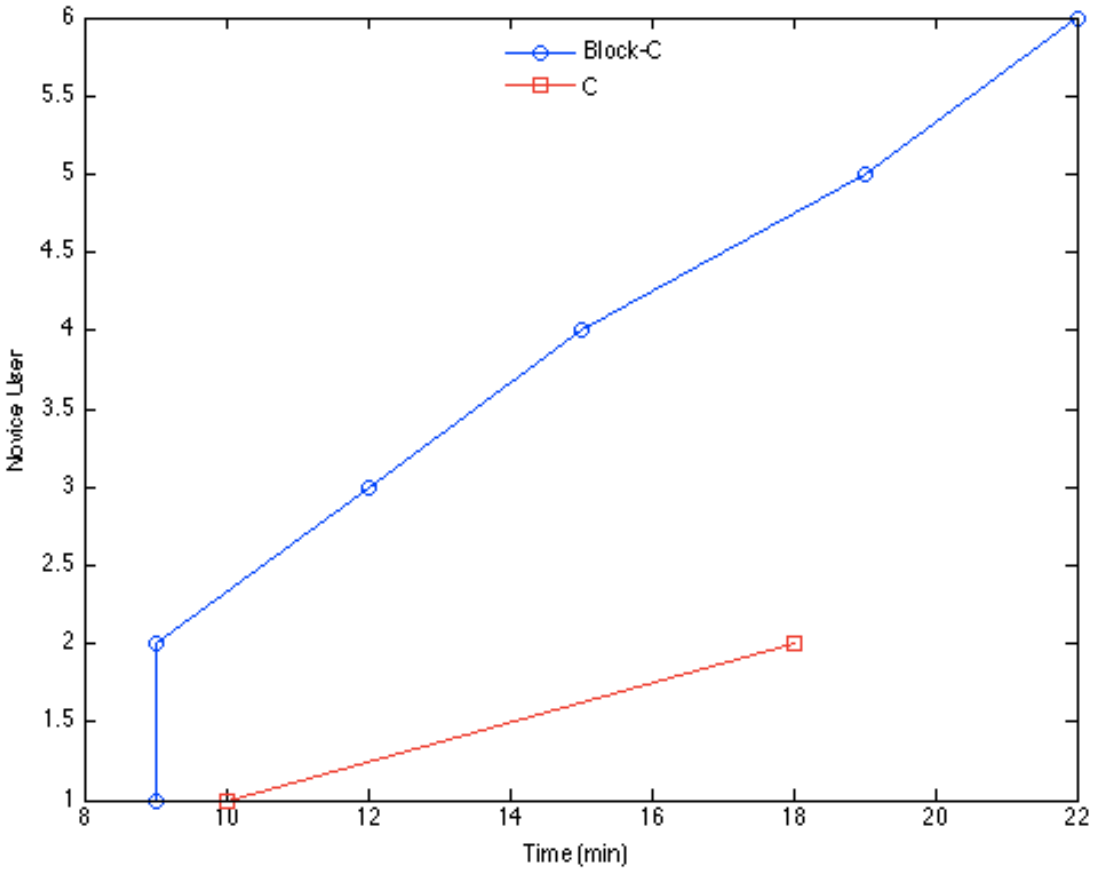
Διάγραμμα 7.12: Αποτελέσματα 1ης εργασίας της δεύτερης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.12 παρατηρούμε τα αποτελέσματα για τη πρώτη εργασία. Οι φοιτητές της ομάδας πειραματισμού (Block-C) ολοκλήρωσαν όλοι (δεκαέξι) την εργασία, ενώ από την ομάδα ελέγχου (C) ολοκλήρωσαν δώδεκα. Είναι ενδιαφέρον το ότι το 25% των φοιτητών της ομάδας ελέγχου (C) δεν κατάφερε να ολοκληρώσει, γιατί στην πρώτη ενότητα του πειράματος η ομάδα αυτή υπερτερούσε. Επίσης από το διάγραμμα παρατηρούμε ότι η ομάδα που δούλευε με το Block-C τα πήγε καλύτερα όσον αφορά τον χρόνο , επειδή, όλοι εκτός από έναν, οι φοιτητές τελειώσαν πιο γρήγορα.



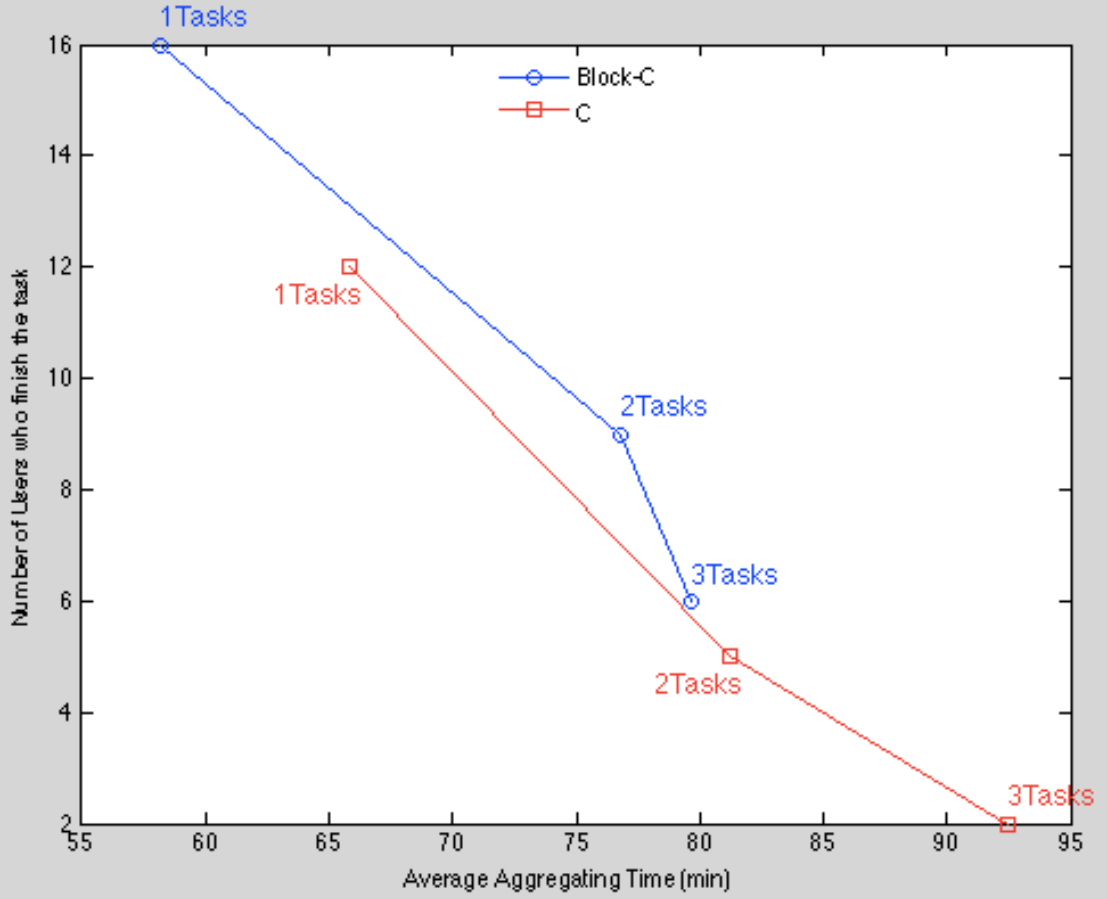
Διάγραμμα 7.13: Αποτελέσματα 2ης εργασίας της δεύτερης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.13 παρατηρούμε τα αποτελέσματα για τη δεύτερη εργασία. Από ομάδα πειραματισμού (Block-C) ολοκλήρωσαν εννέα φοιτητές την εργασία, ενώ από την ομάδα ελέγχου (C) ολοκλήρωσαν πέντε. Περίπου 45% περισσότεροι φοιτητές ολοκλήρωσαν με Block-C. Επίσης παρατηρούμε ότι η ομάδα που δούλευε με το Block-C τα πήγε καλύτερα όσον αφορά τον χρόνο ολοκλήρωσης.



Διάγραμμα 7.14: Αποτελέσματα 3ης εργασίας της δεύτερης ενότητας της αξιολόγησής. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα η κατάταξη των φοιτητών με βάση την σειρά που τελείωσαν.

Στο διάγραμμα 7.14 παρατηρούμε τα αποτελέσματα για τη τρίτη εργασία. Από ομάδα πειραματισμού (Block-C) ολοκλήρωσαν έξι φοιτητές την εργασία, ενώ από την ομάδα ελέγχου (C) ολοκλήρωσαν μόνο 2 . Από το παρόν διάγραμμα φαίνεται ξεκάθαρα ότι η ομάδα που δούλευε με το Block-C τα πήγε σαφώς καλύτερα αφού περίπου 66% περισσότεροι φοιτητές ολοκλήρωσαν.



Διάγραμμα 7.15: Αθροιστικοί μέσοι όροι ολοκλήρωσης μίας, δύο και τριών εργασιών. Για παράδειγμα ο μέσος χρόνος ολοκλήρωσης της 3ης εργασίας είναι το άθροισμα των μέσων της 1ης της 2ης και της 3ης εργασίας. Με μπλε είναι η ομάδα που δούλεψε με το Block-C και με κόκκινο η ομάδα που δούλεψε με την C. Στον οριζόντιο άξονα παρατίθεται ο αθροιστικό μέσος χρόνος ολοκλήρωσης της εργασίας και στον κάθετο άξονα ο αριθμός των φοιτητών που ολοκλήρωσαν τον συγκεκριμένο αριθμό εργασιών. Το τέρμα αριστερά και επάνω σημείο δηλώνει την ολοκλήρωση μίας εργασίας· προχωρώντας ευθεία και έπειτα με βάση τη φορά του ρολογιού είναι τα στοιχεία για την ολοκλήρωση δύο και τριών εργασιών.

Στο διάγραμμα 7.15 παρατηρούμε τα αποτελέσματα για τους αθροιστικούς μέσους όρους για την ολοκλήρωση μίας , δύο και τριών εργασιών. Η ομάδα των φοιτητών που δούλεψε με τη Block-C φαίνεται ξεκάθαρα ότι υπερτερεί και σε χρόνο ολοκλήρωσης καθώς και σε αριθμό φοιτητών που ολοκλήρωσαν τις εργασίες.

Αποτελέσματα ερωτηματολογίων

Τα ερωτηματολόγια περιείχαν 28 ερωτήσεις στις οποίες οι χρήστες μπορούσαν να διαλέξουν μεταξύ του 1 και του 9 (1 ήταν ο χαμηλότερος και 9 ο μέγιστος βαθμός για κάθε ερώτηση). Ο μέσος όρος για κάθε ερώτηση ήταν 7.437 με τυπική απόκλιση 0.436 και ο μέσος όρος βαθμολογίας ανά χρήστη ήταν 7.424 με τυπική απόκλιση 1.002. Τα αποτελέσματα των ερωτηματολογίων αξιολογήθηκαν με τη μέθοδο του Cronbach's Alpha [37]. Το Cronbach's Alpha είναι ένας συντελεστής

εσωτερικής συνοχής. Η χρήση του είναι ευρέως διαδεδομένη για την εκτίμηση της αξιοπιστίας ψυχομετρικών τεστ για ένα δείγμα εξεταζομένων. Το αποτέλεσμα του Alpha ήταν 0.93383508 το οποίο με βάση τη μέθοδο θεωρείται εξαιρετικό.

Πίνακας 7.16: Βασική περιγραφή του προφίλ των χρηστών της ομάδας πειραματισμού και του μέσου όρου βαθμολογίας τους στο σύνολο των ερωτήσεων του ερωτηματολογίου.

Χρήστης	Φύλο	Εμπειρία	Μ.Ο. βαθμολογίας
#1	Α	Όχι	8.36
#2	Α	Λίγο	8.32
#3	Α	Τεχνολογική	8.75
#4	Α	Λίγο	5.84
#5	Θ	Ναι	8.57
#6	Α	Τεχνολογική	7.29
#7	Α	Τεχνολογική	6.75
#8	Α	Όχι	6.57
#9	Θ	Όχι	7.11
#10	Θ	Ναι	8.52
#11	?	?	5.79
#12	Θ	Όχι	7.12
#13	Θ	Τεχνολογική	8.71
#14	?	?	7.14
#15	Θ	Όχι	6.54

Πίνακας 7.17: Μέσος όρος βαθμολογίας που συγκέντρωσε η κάθε ερώτηση.

Ερώτηση	Μ.Ο. Βαθμολογίας
1) Γενική γνώμη για το Block-C (Χάλια - Καταπληκτικό)	7.43
2) Γενική γνώμη για το Block-C (Δύσκολο – Εύκολο)	6.93
3) Γενική γνώμη για το Block-C (Απογοητευτικό - Ικανοποιητικό)	7.07
4) Γενική γνώμη για το Block-C (Βαρετό – Ενδιαφέρον)	7.64
5) Τα Block στην οθόνη ήταν (Δυσανάγνωστα – Ευανάγνωστα)	7.47
6) Η οργάνωση της πληροφορίας (Προκαλούσε σύγχυση – Πολύ ξεκάθαρη)	7
7) Η ορολογία που χρησιμοποιήθηκε ήταν σχετική με αυτό που κάνετε (Ποτέ – Πάντα)	7.4
8) Ανακαλύπτεις νέες δυνατότητες μέσω πειραματισμού και λαθών (Δύσκολα – Εύκολα)	7.73
9) Καταλάβαινες την έννοια των ονομάτων, και την σημασία των χρωμάτων και σχημάτων (Δύσκολα – Εύκολα)	7.4
10) Η υλοποίηση ενός προγράμματος εκτελείται με απλό/άμεσο τρόπο (Ποτέ – Πάντα)	6.6
11) Ταχύτητα του Block-C (Πολύ αργό – Πολύ γρήγορο)	7.6

12) Το να διορθώσεις τα λάθη σου ήταν (Δύσκολο – Εύκολο)	6.4
13) Θα εμπνεόσασταν να χρησιμοποιήσετε το Block-C για να δημιουργήσετε ένα πρόγραμμα (Απίθανο – Βέβαιο)	6.71
14) Θα σας άρεσε να σας δείχνουν παραδείγματα στο μάθημα φτιαγμένα με το Block-C μας (Καθόλου – Πολύ)	7.87
15) Πιστεύετε ότι θα μπορούσε να χρησιμοποιηθεί το Block-C για εκπαιδευτικούς σκοπούς (Δύσκολα – Εύκολα)	7.87
16) Πιστεύετε ότι το Block-C μας μπορεί να ενθαρρύνει αρχάριους να ασχοληθούν με τον προγραμματισμό (Καθόλου – Πολύ)	8.13
17) Το να χρησιμοποιώ το Block-C θα με βοηθούσε να φτιάχνω προγράμματα πιο γρήγορα (Καθόλου – Πολύ)	7.67
18) Χρησιμοποιώντας το Block-C θα βελτίωνα την απόδοση μου στον προγραμματισμό σε C (Απίθανο – Βέβαιο)	8.21
19) Χρησιμοποιώντας το Block-C θα προγραμμάτιζα πιο αποτελεσματικά (Απίθανο – Βέβαιο)	7.46
20) Χρησιμοποιώντας το Block-C θα προγραμμάτιζα σε C πιο εύκολα (Απίθανο – Βέβαιο)	7.5
21) Θα έβρισκα το Block-C εύχρηστο για να προγραμματίσω σε C (Καθόλου – Πολύ)	7.79
22) Χρησιμοποιώντας το Block-C για να φτιάχνω προγράμματα θα αξιοποιούσα καλύτερα τον χρόνο μου (Απίθανο – Βέβαιο)	7.5
23) Το να μάθω να χρησιμοποιώ το Block-C θα ήταν εύκολο για μένα (Καθόλου – Πολύ)	7.5
24) Θα έβρισκα εύκολο να καταφέρω με το Block-C να κάνω αυτό που θέλω (Καθόλου – Πολύ)	7.5
25) Η αλληλεπίδραση μου με το Block-C θα ήταν ξεκάθαρη και κατανοητή (Καθόλου – Πολύ)	7.36
26) Θα Εύρισκε το Block-C ευέλικτο στην αλληλεπίδραση μαζί του (Καθόλου – Πολύ)	7.14
27) Θα μου ήταν εύκολο να γίνω ειδικός στη χρήση του Block-C (Απίθανο – Βέβαιο)	7.21
28) θα έβρισκα το Block-C εύκολο στη χρήση (Καθόλου – Πολύ)	8.14

Ενδιαφέρουσες ήταν και οι απαντήσεις των χρηστών στις ερωτήσεις ελεύθερης απάντησης.

Στην ερώτηση «*Πώς θα περιγράφατε το Block-C σε ένα φίλο σας;*» μερικές οι πιο χαρακτηριστικές ήταν οι εξής:

- Άξιο δοκιμής
- Ενδιαφέρων και βοηθητικό στην κατανόηση της C
- Αρκετά ενδιαφέρων, απλά λόγο απογοητευτικό στην αρχή
- Εύκολο στην κατανόηση, αρκετά βοηθητικό αλλά δίνει μεγαλύτερη ικανοποίηση το να προγραμματίζεις κανονικά.

- Έναν εύκολο τρόπο απομνημόνευσης των εντολών της C και του τρόπου που αυτές λειτουργούν· πολύ χρήσιμο για αρχάριους.
- Πολύ καλό, άρτια ανεπτυγμένο πρόγραμμα
- Δύσκολο στην αρχή αλλά ενδιαφέρον
- Ένα πολύ εύχρηστο εργαλείο για αρχάριους πάνω στη C
- Αρκετά φιλικό, κατανοητό σε μεγάλο βαθμό, δύσκολο στην αρχή σχετικά με την εύρεση και την υλοποίηση υπολογισμών. Θα του πρότεινα αν έχει εμπειρία να μην το δοκιμάσει αλλιώς αν είναι αρχάριος, η τέλεια επιλογή.
- Θα το περιέγραφα ως έναν ευχάριστο και αποτελεσματικό τρόπο να προγραμματίζεις και να γίνεις καλός σε αυτό.

Στην ερώτηση «*Ποιο/α χαρακτηριστικό/ά του Block-C σας άρεσε πιο πολύ και γιατί;*» οι πιο χαρακτηριστικές απαντήσεις ήταν οι εξής:

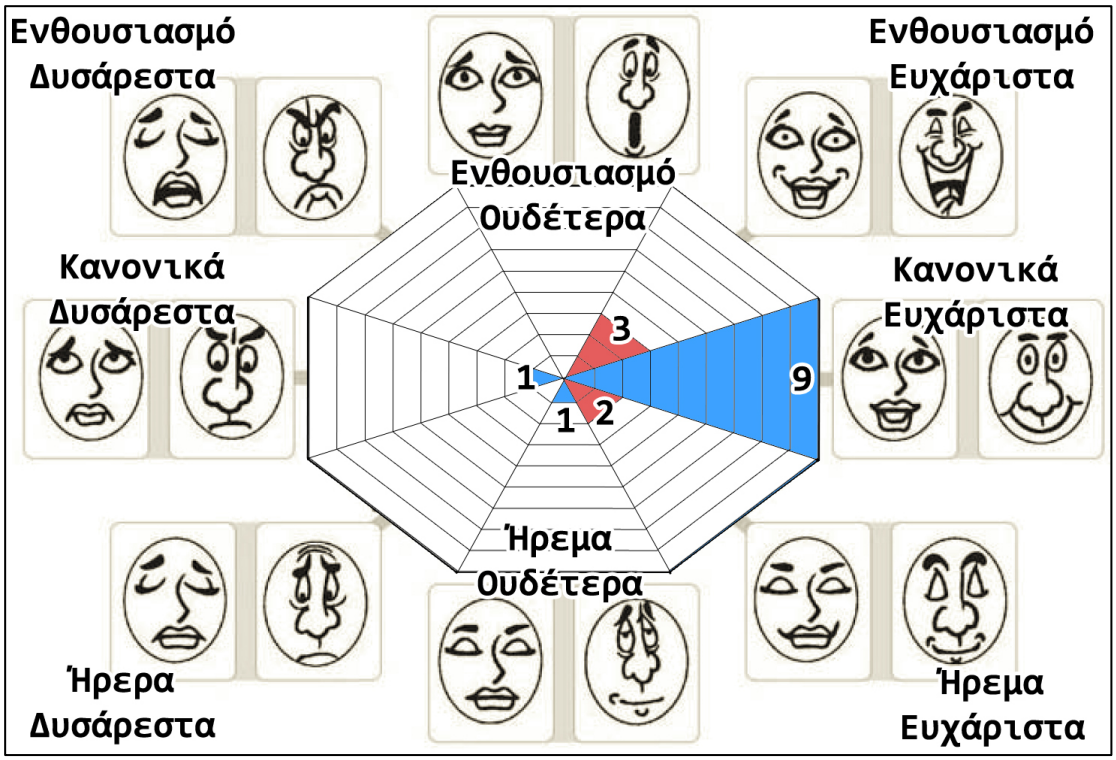
- Η κατηγοριοποίηση με χρώματα
- Με τα χρώματα και τα κουτιά ήταν εύκολο να κατανοήσεις το πρόβλημα
- Η ποικιλία των επιλογών όσο αφορά τους τρόπους «επίλυσης» των προβλημάτων
- Τα χρώματα και οι υποδοχές γιατί καθοδηγούν
- Μου έγιναν αρκετά πράγματα πιο κατανοητά
- Το στοιχείο που άρεσε περισσότερο, επειδή με βοήθησε περισσότερο είναι η μορφή puzzle που έχουν οι εντολές
- Ευκολία στην κατανόηση, σχεδιασμό-υλοποίηση προγραμμάτων
- Ήταν σαν puzzle και γι’ αυτό έμοιαζε με παιχνίδι
- Οι έτοιμες εντολές κλπ στο αριστερό μέρος του προγράμματος
- Το στυλ του παζλ συνδυάζει παιχνίδι και γνώση
- Το «παζλ» αφού σου έδινε μια extra βοήθεια εκεί που κολλούσες και δεν γνώριζες τι να κάνεις.
- Έχει έτοιμες ήδη τις εντολές και τις οδηγίες και μπορείς να δουλέψεις πιο εύκολα

Στην ερώτηση «*Κατά την άποψη σας τι θα μπορούσε να προστεθεί ή τι λείπει από το Block-C;*» οι πιο χαρακτηριστικές απαντήσεις ήταν οι εξής:

- Μεταγλώττιση στα ελληνικά με πιο απλές ενδείξεις και αύξηση λειτουργικότητας για πιο έμπειρους χρήστες
- Όχι, υπήρχε ακριβώς ό,τι χρειάστηκα!
- Μερικές διευκρινήσεις για τη συμβατότητα ανάμεσα σε χρώματα και σχήματα
- Θα μπορούσε να βελτιωθεί η ταχύτητα με την οποία λειτουργεί
- Από την πρώτη επαφή το βρίσκω αρκετά καλά οργανωμένο πρόγραμμα
- Επεξήγηση των συμβόλων στα αριστερά
- Δεν θεωρώ ότι λείπει κάτι. Είναι πιο εύκολο από τον κανονικό προγραμματισμό της γλώσσας C.
- Θα μπορούσε να είναι λίγο πιο επεξηγηματικό
- Το να τρέχει το πρόγραμμα
- Έτοιμα παραδείγματα πριν από την κανονική χρήση

- Δεν γνωρίζω εάν θα μπορούσε να προστεθεί κάτι . Εγώ το βρήκα φανταστικό

Τα αποτελέσματα στην ερώτηση «Συναισθήματα που σας προκάλεσε το Block-C» όπου οι χρήστες έπρεπε να απαντήσουν διαλέγοντας ένα συναίσθημα από μια συλλογή οκτώ διαφορετικών συναισθηματικών καταστάσεων [36] είναι τα εξής:



Διάγραμμα 7.18: Γραφική αναπαράσταση αποτελεσμάτων στην ερώτηση «Συναισθήματα που σας προκάλεσε το Block-C».

Κεφάλαιο 8
Συμπεράσματα και μελλοντική εργασία

Περίληψη

Σε αυτό το κεφάλαιο αναλύονται τα αποτελέσματα και παρουσιάζονται τα συμπεράσματα τα οποία προέκυψαν από την αξιολόγηση του Block-C . Επίσης εκτίθενται οι σκέψεις μας για μελλοντική επέκταση του εργαλείου.

Εισαγωγή

Η τρίτη αξιολόγηση αποτελούνταν από δύο ενότητες.

1. Η πρώτη αποτελούσε μια εισαγωγική ενότητα όπου οι χρήστες υποβοηθούταν από εργαστηριακούς βοηθούς και έτσι κλήθηκαν να υλοποιήσουν δέκα εργασίες κλιμακούμενης δυσκολίας.
2. Η δεύτερη ενότητα αποτέλεσε το ουσιαστικό μέρος του πειράματος το οποίο θα εξήγαγε και την πληροφορία για την αποτελεσματικότητα του εργαλείου. Οι χρήστες κλήθηκαν να εκπονήσουν τρεις εργασίες. Καθ' όλη την διάρκεια της ενότητας αυτής οι χρήστες δεν υποβοηθούνταν από τους εργαστηριακούς βοηθούς. Αυτό γινόταν για να δούμε κατά πόσο το εργαλείο βοηθούσε τους μαθητευόμενους και αν υπερτερούσε του κλασσικού τρόπου προγραμματισμού.

Τα αποτελέσματα της τρίτης αξιολόγησης παρουσιάστηκαν αναλυτικά στο προηγούμενο κεφάλαιο και σε αυτό θα εξαχθούν τα συμπεράσματα που προκύπτουν από αυτά.

Σύγκριση 2 μεθόδων

Μας ενδιαφέρει να δούμε κατά πόσο οι χρήστες εισάγονται στην Ζώνη Επικείμενης Ανάπτυξης τους, υποβοηθούμενοι μόνο από το εργαλείο. Η Ζώνη Επικείμενης Ανάπτυξης ερμηνεύεται ως η διαφορά μεταξύ του τι μπορεί να πετύχει ο μαθητευόμενος μόνος του και τι όταν υποβοηθείται.

Για να δούμε αυτή την διαφορά μπορούμε να συγκρίνουμε με βάση τον χρόνο για την επίτευξη κάθε εργασίας ή τον αριθμό των ατόμων που ολοκλήρωσε κάθε εργασία.

Το μέγεθος με βάση το οποίο θα συγκρίνουμε τα τελικά αποτελέσματα των δύο μεθόδων είναι ο αθροιστικός μέσος όρος του χρόνου για την ολοκλήρωση k εργασιών (όπου k=1,2,3). Επειδή οι δύο μέθοδοι δεν είχαν το ίδιο αριθμό χρηστών που ολοκλήρωσαν την κάθε εργασία, στους υπολογισμούς και στην σύγκριση των μέσων όρων επιλέξαμε όλους τους χρήστες που δούλεψαν με C, για παράδειγμα ***x*** χρήστες, και αντίστοιχα τους πρώτους ***x*** χρήστες που δούλεψαν με Block-C.

Ο αριθμός των χρηστών που ολοκλήρωσαν τις εργασίες με C, επιλέχθηκε ως «κανόνας» , επειδή σε κάθε εργασία ήταν μικρότερος από αυτόν των χρηστών της Block-C. Έτσι οι υπολογισμοί μπορούν να θεωρηθούν πιο «δίκαιοι» για την C.

Αποτελέσματα

Τα αποτελέσματα της δεύτερης ενότητας κατέδειξε το όφελος της πρόληψης των συντακτικών λαθών που παρέχει το Block-C. Επίσης σημαντικά είναι τα οφέλη της χρήσης της νοητικής διαδικασίας της αναγνώρισης, που χρησιμοποιήθηκε από τους χρήστες του Block-C, παρά αυτής της ανάκλησης, η οποία χρησιμοποιήθηκε από τους χρήστες της C (recognition over recall [1]). Με τον όρο αναγνώριση εννοούμε την δυνατότητα του χρήστη να αναγνωρίσει από ένα σύνολο παρεχομένων εντολών ποια κάνει τι. Με τον όρο ανάκληση εννοούμε την προσπάθεια του χρήστη να βρει από τις απομνημονευμένες, από αυτόν, εντολές ποια του χρειάζεται και πώς αυτή συντάσσεται.

Η έλλειψη της υποστήριξης και καθοδήγησης από της βοηθούς ανέδειξε την συνεισφορά του Block-C στην επικέντρωση των φοιτητών στη λογική για την επίλυση των εργασιών. Έτσι, όπως φαίνεται καθαρά στις εικόνες 7.12 μέχρι 7.15 οι φοιτητές του Block-C έχουν βελτιωμένη απόδοση.

Για κάθε εργασία της δεύτερης ενότητας οι χρήστες με το Block-C που ολοκλήρωσαν την εργασία ήταν περισσότεροι από αυτούς που δούλεψαν με C.

Τα καταληκτικά αποτελέσματα, που περιγράφονται από το μέσο όρο του χρόνου και από των αριθμό ολοκλήρωσης κάθε εργασίας, φαίνονται και στον πίνακα 8.1.

Πρέπει τέλος να τονίσουμε ότι οι χρήστες οι οποίοι χρησιμοποιήθηκαν για τον υπολογισμού των Μέσων Όρων στον πίνακα 8.1 δεν ήταν οι ίδιοι. Για παράδειγμα ο χρήστης που τελείωσε πρώτος την πρώτη εργασία δεν ολοκλήρωσε τις επόμενες δύο!

Πίνακας 8.1: Χρήστες που ολοκλήρωσαν κάθε εργασία και ο μέσος χρόνος για τους k_i πρώτους (k_i = ο αριθμός των χρηστών που τελείωσαν την εργασία i χρησιμοποιώντας την C)

Εργασία	Μέθοδος	Χρήστες που Ολοκλήρωσαν	Μέσος Χρόνος k_i Χρηστών	k_i
1	Block-C	16	47.6667	k_i = 12 χρήστες της C και οι 12 πρώτοι του Block-C
	C	12	60.7692	
2	Block-C	9	19.6	k_i = 5 χρήστες της C και οι 5 πρώτοι του Block-C
	C	5	32.4	
3	Block-C	6	9	k_i = 2 χρήστες της C και οι 2 πρώτοι του Block-C
	C	2	14	

Πίνακας 8.2: Βελτίωση Μαθησιακής Αποδοτικότητας του Block-C έναντι της C (βασισμένη στον μέσο χρόνο)

Εργασία	1	2	3
Βελτίωση	x1.27	x1.65	x1.55

Πίνακας 8.3: Βελτίωση Αριθμού που ολοκλήρωσαν εργασίες με Block-C έναντι της C (βασισμένη στον αριθμό ατόμων που ολοκλήρωσαν μία, δύο και τρεις εργασίες)

Εργασίες	1	2	3
Βελτίωση	x1.33	x1.8	x3

Ερωτηματολόγια

Από τα αποτελέσματα των ερωτηματολογίων παρατηρούμε ότι ο μέσος όρος για κάθε ερώτηση ήταν 7.437 με τυπική απόκλιση 0.436. Αυτό σημαίνει ότι οι χρήστες ήταν γενικά ικανοποιημένοι από το πρόγραμμα. Οι τρεις ερωτήσεις με το χαμηλότερο και οι τρεις με υψηλότερο μέσο όρο βαθμού φαίνονται στον πίνακα 8.4.

Πίνακας 8.4: Αποτελέσματα για τις τρεις ερωτήσεις με χαμηλότερη βαθμολογία και τις τρεις με την υψηλότερη. Παρουσιάζονται ο αριθμός της ερώτησης, η ερώτηση και η βαθμολογία που πήρε.

	Αριθμός ερ.	Ερώτηση	Βαθμολογία
Χαμηλότερη βαθμολογία	12	Το να διορθώσεις τα λάθη σου ήταν	**6.4**
	10	Η υλοποίηση ενός προγράμματος εκτελείται με απλό/άμεσο τρόπο	**6.6**
	13	Θα εμπνεόσασταν να χρησιμοποιήσετε το Block-C για να δημιουργήσετε ένα πρόγραμμα	**6.714**
Υψηλότερη βαθμολογία	16	Πιστεύετε ότι το Block-C μας μπορεί να ενθαρρύνει αρχάριους να ασχοληθούν με τον προγραμματισμό;	**8.133**
	28	θα έβρισκα το Block-C εύκολο στη χρήση	**8.1428**
	18	Χρησιμοποιώντας το Block-C θα βελτίωνα την απόδοση μου στον προγραμματισμό σε C	**8.214**

Από την 12η ερώτηση που πήρε την χαμηλότερη βαθμολογία συμπεραίνουμε ότι το οι χρήστες δυσκολεύτηκαν στην διόρθωση των λαθών τους. Αυτό πιθανώς να

οφείλεται στην έλλειψη εμπειρίας στους και στον προγραμματισμό αλλά και στο εργαλείο.

Η ερώτηση με την δεύτερη χαμηλότερη βαθμολογία είναι η 10η. Οι χρήστες παρατήρησαν ότι η δημιουργία τους προγράμματος δεν γίνεται με απλό και άμεσο τρόπο. Αυτό, ίσως οφείλεται στο γεγονός ότι οι χρήστες είχαν διαφορετικά συνδυάσει τον προγραμματισμό στο μυαλό τους. Αυτός ο τρόπος τους φάνηκε όχι τόσο άμεσος αλλά ούτε και τόσο απλός λόγω ότι ήταν η πρώτη φορά που ήρθαν σε επαφή μαζί του. Παρ' όλα αυτά, από ό,τι φαίνεται και από τις δύο υψηλότερες σε βαθμολογία ερωτήσεις (28η και 18η), δεν σημαίνει ότι δεν βρήκαν το εργαλείο αποτελεσματικό.

Η επόμενη χαμηλότερη βαθμολογία συναντάται στην 13η ερώτηση. Οι χρήστες δεν θεωρούν ότι θα τους ενέπνεε το εργαλείο για να δημιουργήσουν ένα πρόγραμμα. Ίσως οι περισσότεροι χρήστες να μην έλκονται από τον προγραμματισμό και έτσι τίποτα να μην τους ελκύει στην δημιουργία προγραμμάτων.

Παρά το γεγονός αυτό στην ερώτηση με την τρίτη καλύτερη βαθμολογία (16η) οι χρήστες πιστεύουν ότι μπορεί να ενθαρρύνει αρχάριους να ασχοληθούν με τον προγραμματισμό.

Τα παραπάνω αποτελέσματα ίσως φαίνονται αντιφατικά αλλά αντιθέτως είναι συμπληρωματικά. Οι χρήστες ενώ δεν θα εμπνέονταν να δημιουργήσουν προγράμματα ενθαρρύνθηκαν από το εργαλείο. Η υλοποίηση ενός προγράμματος στην C σίγουρα δεν εκτελείται με άμεσο ή απλό τρόπο όπως για παράδειγμα σε μια γλώσσα προγραμματισμού υψηλότερου επιπέδου όπως η python. Αυτό, λογικά, διαπίστωσαν οι χρήστες αλλά παράλληλα κατάλαβαν ότι το εργαλείο είναι εύκολο στη χρήση και μπορεί να τους βοηθούσε να βελτιωθούν στην C.

Τα παραπάνω αποτελέσματα σε συνδυασμό με αποτελέσματα στην ερώτηση «Συναισθήματα που σας προκάλεσε το Block-C», τα οποία φαίνονται αναλυτικά στο διάγραμμα 7.18, δείχνουν ότι το εργαλείο ικανοποίησε τους χρήστες.

Πίνακας 8.5: Αποτελέσματα στην ερώτηση «Συναισθήματα που σας προκάλεσε το Block-C». Παρουσιάζονται το συναίσθημα και το πλήθος των χρηστών που το διάλεξαν (ποσοτικά και ποσοστιαία).

Συναίσθημα	Αριθμός χρηστών	Ποσοστό
Ενθουσιασμό Ευχάριστα	3	18.75%
Κανονικά Ευχάριστα	9	56.25%
Ήρεμα Ευχάριστα	2	12.5%
Ήρεμα Ουδέτερα	1	6.25%
Κανονικά Δυσάρεστα	1	6.25%
Σύνολο	**16**	**100%**

Το 87.5% των χρηστών αισθάνθηκε ευχάριστα κατά την χρήση του εργαλείου. Οι περισσότεροι χρήστες (56.5%) αισθάνθηκαν «κανονικά ευχάριστα». Μόνο το 6.25% αισθάνθηκε δυσάρεστα, το οποίο πιθανόν να οφείλεται σε απογοήτευση η οποία προήλθε από έλλειψη πρότερης εμπειρίας με τον προγραμματισμό.

Συμπεράσματα

Η σύνταξη σε μία γλώσσα προγραμματισμού είναι, τις περισσότερες φορές, το πιο σημαντικό εμπόδιο για κάποιον που προσπαθεί να την μάθει. Οι εκπαιδευόμενοι μπορεί να αντιλαμβάνονται εύκολα τις λογικές μονάδες (πχ if/else) μιας γλώσσας προγραμματισμού επειδή μπορούν να τις αντιστοιχίσουν με νοητικές διαδικασίες τις καθημερινότητας, καθ' ότι αποτελούν μεταφορές (metaphors) τις ανθρώπινης σκέψης. Αντίθετα δεν είναι εξοικειωμένοι με την σύνταξη και τις συντακτικές μονάδες των γλωσσών προγραμματισμού.

Στην περίπτωση ιδρυμάτων (όπως πανεπιστημιακά τμήματα) όπου η διδασκαλία της C δεν μπορεί να αντικατασταθεί από εκπαιδευτικές γλώσσες ή εργαλεία προγραμματισμού που απευθύνονται σε αρχάριους (όπως Scratch και Alice) το Block-C θα μπορούσε να βοηθήσει τους φοιτητές, λόγω του ότι αποτελεί ένα εργαλείο που εισάγει ομαλά τους χρήστες του στη γλώσσα προγραμματισμού C.

Ειδικότερα, σε περιπτώσεις όπου οι ανθρώπινοι πόροι είναι περιορισμένοι, το Block-C μπορεί να βοηθήσει τους χρήστες του να αντιμετωπίσουν αποτελεσματικά τα προβλήματα που συναντά κάθε αρχάριος κατά την εισαγωγή του στον προγραμματισμό με τη γλώσσα C. Επίσης, μπορεί να τους βοηθήσει να ξεπεράσουν την απογοήτευση η οποία προκαλείται από την τραχύτητα της C -ειδικά αν πρόκειται για την πρώτη γλώσσα προγραμματισμού που συναντάει κάποιος.

Το Block-C φαίνεται ότι καταφέρνει να πετύχει τον στόχο για τον οποίο δημιουργήθηκε. Εισάγει τους χρήστες στη Ζώνη Επικείμενης Ανάπτυξής τους κάνοντας τους 1.27 έως 1.65 (Πίνακας 8.2) φορές πιο αποτελεσματικούς σε χρόνο αλλά και βοηθώντας 1.33 έως 3 (Πίνακας 8.3) φορές περισσότερους να ολοκληρώνουν πιο πολλές εργασίες.

Επίσης όπως φαίνεται και από τα σχόλια των χρηστών στις ερωτήσεις ελεύθερης απάντησης, οι χρήστες το βρίσκουν ενδιαφέρον και αποτελεσματικό.

Αυτό το γεγονός παρατηρήθηκε και κατά την διάρκεια της αξιολόγησής του όπου, παρά την σύγχυση που επικρατούσε μερικές φορές στους χρήστες, δεν φάνηκε να απογοητεύονται αλλά αντιθέτως έβλεπαν την αλληλεπίδραση με το εργαλείο ως ένα εκπαιδευτικό παιχνίδι!

Μελλοντική δουλειά

Η διαδικασία της αξιολόγησης του εργαλείου ήταν αρκετά ωφέλιμη από διάφορες απόψεις και μας ανατροφοδότησε με αρκετή πληροφορία για τη σταδιακή βελτίωσή του. Μέσα από δικές μας (των αξιολογητών) παρατηρήσεις αλλά και τις παρατηρήσεις των χρηστών βρήκαμε αρκετά προβλήματα και πράγματα που έλειπαν αρχικά από το Block-C καθώς και ιδέες για νέα λειτουργικότητα που θα μπορούσε να ενσωματωθεί σε αυτό στο άμεσο μέλλον.

Η κύρια σκέψη για την συνέχιση αυτής της εργασίας είναι η εισαγωγή λειτουργικότητας για την δημιουργία γραφικού εκσφαλματωτή (debugger). Η βασική ιδέα αυτής της προοπτικής είναι να μπορεί το εργαλείο να δέχεται κώδικα

σε κείμενο και να τον παρουσιάζει σε Blocks. Έτσι τα οποιαδήποτε συντακτικά λάθη ή οι αναντιστοιχίες θα μπορούν να αναγνωριστούν εύκολα.

Περαιτέρω, εάν αυτός ο εκσφαλματωτής μπορεί να «τρέχει» το πρόγραμμα δείχνοντας βήμα προς βήμα τις εντολές που εκτελούνται, τονίζοντας τα αντίστοιχα blocks, οι προγραμματιστές θα μπορούσαν να έχουν μια πιο ουσιαστική και διαισθητική ιδέα για το πρόγραμμα τους.

Επίσης, ως έχει τώρα το εργαλείο, πρέπει να ενσωματωθεί λειτουργικότητα ώστε ο χρήστης να μπορεί να εκτελεί το πρόγραμμα του και να μην χρειάζεται να το μεταφέρει σε εξωτερικό πρόγραμμα για να το μεταφράσει σε κώδικα μηχανής και να το εκτελέσει.

References / Βιβλιογραφία

1. William Lidwell, Kritina Holden, Jill Butler, 2003. Universal Principles of Design.
2. Gerhard Fischer. 2011. Understanding, fostering, and supporting cultures of participation. interactions 18, 3 (May 2011), 42-53. DOI=10.1145/1962438.1962450
3. G. Fischer, E. Giaccardi, Y. Ye, A. G. Sutcliffe, and N. Mehandjiev. 2004. Meta-design: a manifesto for end-user development. Commun. ACM 47, 9 (September 2004), 33-37. DOI=10.1145/1015864.1015884
4. Brad A. Myers, Andrew J. Ko, and Margaret M. Burnett. 2006. Invited research overview: end-user programming. In CHI '06 Extended Abstracts on Human Factors in Computing Systems (CHI EA '06). ACM, New York, NY, USA, 75-80. DOI=10.1145/1125451.1125472 http://doi.acm.org/10.1145/1125451.1125472
5. Andrew J. Ko, Robin Abraham, Laura Beckwith, Alan Blackwell, Margaret Burnett, Martin Erwig, Chris Scaffidi, Joseph Lawrance, Henry Lieberman, Brad Myers, Mary Beth Rosson, Gregg Rothermel, Mary Shaw, and Susan Wiedenbeck. 2011. The state of the art in end-user software engineering. ACM Comput. Surv. 43, 3, Article 21 (April 2011), 44 pages. DOI=10.1145/1922649.1922658 http://doi.acm.org/10.1145/1922649.1922658
6. Alfred Thompson. 2012. Programming With Blocks. Computer Science Teacher. December 11, 2012. Web. January 3, 2013. DOI=http://blog.acthompson.net/2012/12/ programming-with-blocks.html
7. Tamar Vilner, Ela Zur, Shay Tavor 2011.Using greenfoot in teaching inheritance in CS1. ITiCSE '11 ACM New York, NY, USA. DOI=10.1145/1999747.1999855
8. Zone of proximal development. http://en.wikipedia.org/wiki/Zone_of_proximal_development
9. Usability testing. http://en.wikipedia.org/wiki/Usability_testing
10. Logo. Wally Feurzeig, Seymour Papert. 1967. http://en.wikipedia.org/wiki/Logo_%28programming_language%29
11. Smalltalk. Alan Kay, Dan Ingalls, Adele Goldberg. 1972 http://www.squeak.org/Smalltalk/
12. Simula. Ole-Johan Dahl, Kristen Nygaard. 1967. http://en.wikipedia.org/wiki/Simula
13. Margaret M. Burnett, Maria J. Baker 1993. A Classification System for Visual Programming Languages. Technical Report Oregon State University Corvallis, OR, USA DOI=http://dl.acm.org/citation.cfm?id=891125
14. João Paulo Barros, Luís Biscaia and Miguel Vitória 2011. Java2Sequence: a tool for the visualization of object-oriented programs in introductory programming. ITiCSE '11 ACM New York, NY, USA DOI=10.1145/1999747.1999882

15. André L. Santos 2012. An Open-Ended Environment for Teaching Java in Context. ITiCSE '11 ACM New York, NY, USA. DOI=K10.1145/2325296.2325320
16. Rapid application development. http://en.wikipedia.org/wiki/Rapid_application_development
17. Integrated development environment. http://en.wikipedia.org/wiki/Integrated_development_environment
18. Georgios Fesakis, Kiriaki Serafeim 2009. Influence of the familiarization with "scratch" on future teachers' opinions and attitudes about programming and ICT in education. ITiCSE '09 ACM New York, NY, USA DOI=10.1145/1562877.1562957 and ACM SIGCSE Bulletin - ITiCSE '09 DOI=10.1145/1595496.1562957
19. Ryan Garlick, Ebru Celikel Cankaya 2010. Using alice in CS1- a quantitative experiment . ITiCSE '10 ACM New York, NY, USA. DOI=10.1145/1822090.1822138
20. Roque, Ricarose Vallarta 2007. OpenBlocks : an extendable framework for graphical block programming systems DOI=http://dspace.mit.edu/handle/1721.1/41550.
21. Poul Henriksen, Michael Kölling 2004. Greenfoot: combining object visualisation with interaction. OOPSLA '04 ACM New York, NY, USA. DOI= 10.1145/ 1028664.1028701
22. Michael Kölling 2008. Greenfoot - A Highly Graphical IDE for Learning Object-Oriented Programming.. ITiCSE '08 ACM New York, NY, USA. DOI= 10.1145/1384271.1384370 and ACM SIGCSE Bulletin ITiCSE '08 New York, NY, USA. DOI= 10.1145/1597849.1384370
23. Paul Denny, Andrew Luxton-Reilly, Ewan Tempero, Jacob Hendrickx 2012. All syntax errors are not equal. ITiCSE '12 annual conference on Innovation and technology in computer science education. DOI=10.1145/2325296.2325318
24. Paul Denny, Andrew Luxton-Reilly, Ewan Tempero, Jacob Hendrickx 2011. Understanding the syntax barrier for novices. ITiCSE '11 ACM New York, NY, USA. DOI= 10.1145/1999747.1999807
25. Rachel Cardell-Oliver, Patrick Doran Wu 2011. UWA Java tools- harnessing software metrics to support novice programmers. ITiCSE '11 ACM New York, NY, USA. DOI= 10.1145/1999747.1999854
26. Shuhaida Mohamed Shuhidan, Margaret Hamilton, Daryl D'Souza 2011. Understanding novice programmer difficulties via guided learning. ITiCSE '11 ACM New York, NY, USA. DOI= 10.1145/1999747.1999808
27. Paul Lyons , Craig Simmons , Palmerston North 1993. Hyperpascal: A Visual Language to Model Idea Space . Proceedings of the 13th New Zealand Computer Society Conference DOI=http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1. 1.112.744
28. Open Blocks. STEP, MIT http://education.mit.edu/openblocks
29. Scheller Teacher Education Program (STEP). MIT. http://education.mit.edu/
30. StarLogo TNG. MIT http://education.mit.edu/projects/starlogo-tng
31. Application Inventor for Android® devices. MIT. http://appinventor.mit.edu/
32. Think aloud protocol of usability testing. http://en.wikipedia.org/wiki/Think_aloud_protocol

33. Scientific Control on experiment or observation. http://en.wikipedia.org/wiki/Scientific_control
34. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology,Fred D. Davis, MIS Quarterly Vol. 13, No. 3 (Sep., 1989), pp. 319-340, Published by: Management Information Systems Research Center, University of Minnesota, URL: http://www.jstor.org/stable/249008
35. Development of an instrument measuring user satisfaction of the human-computer interface,John P. Chin, Virginia A. Diehl, Kent L. Norman, May 1988 , CHI '88: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, DOI=10.1145/57167.57203
36. Beyond usability: evaluating emotional response as an integral part of the user experience, Anshu Agarwal, Andrew Meyer, CHI EA '09: CHI '09 Extended Abstracts on Human Factors in Computing Systems. DOI= 10.1145/1520340.1520420
37. Cronbach, L. J. (1951). Coefficient alpha and the internal structure of tests. Psychometrika, 16(3), 297–334. DOI=http://en.wikipedia.org/wiki/Cronbach%27s_alpha
38. Acting with Technology: Activity Theory and Interaction Design, Victor Kaptelinin & Bonnie A. Nardi 2009
39. Recognition and free recall of organized lists. Kintsch Walter. Nov 1968. Journal of Experimental Psychology. Vol 78(3, Pt.1), 481-487.
40. A study of the difficulties of novice programmers. Essi Lahtinen, Kirsti Ala-Mutka, Hannu-Matti Järvinen. June 2005. ACM ITiCSE '05. doi>10.1145/1151954.1067453
41. Situated Learning: Legitimate Peripheral Participation (Learning in Doing: Social, Cognitive and Computational Perspectives). Jean Lave, Etienne Wenger. 1991
42. Scratch Etoys programming language. http://en.wikipedia.org/wiki/Scratch_%28programming_language%29
43. Etoys programming language. http://en.wikipedia.org/wiki/Etoys_%28programming_language%29
44. StarLogo: a programmable modeling environment for exploring the workings of decentralized systems. http://education.mit.edu/starlogo/
45. Alice: an innovative 3D programming environment for creating an animation for telling a story, playing an interactive game, or creating a video to share on the web. http://www.alice.org/index.php?page=what_is_alice/what_is_alice
46. Greenfoot, A tool for teaching object orientation with Java. http://www.greenfoot.org/overview
47. Microsoft Visual Basic. https://en.wikipedia.org/wiki/Visual_Basic
48. Using the ZPD in Accelerated Reader. https://hosted233.renlearn.com/265945/help/ar/ZPD.htm