**Use-cases-v0.2**

BLOCKO

**Μέλη Ομάδας:**

**Όνομα**: Κωνσταντίνος

**Επώνυμο**: Καρακίτσος

**ΑΜ**: 1093381

**Όνομα**: Βασίλειος

**Επώνυμο**: Μιχανετζής

**ΑΜ:** 1093433

**Όνομα**: Νυμφοδώρα

**Επώνυμο:** Μπουζιάνη

**ΑΜ**: 1097429

**Όνομα**: Μιχαήλ

**Επώνυμο:** Σοκολάκης

**ΑΜ**: 1072589

**Όνομα**: Σωτήριος

**Επώνυμο**: Χασιώτης

**ΑΜ**: 1093510

**Εργαλεία :**

Τα εργαλεία που χρησιμοποιήσαμε για την σύνταξη των τεχνικών κειμένων είναι το Microsoft Word και για την δημιουργία των διαγραμμάτων το draw.io.

**Αλλαγές που πραγματοποιήθηκαν από την προηγούμενη έκδοση (Use-case-v0.1):**

Στο use case δήλωση και αρχικοποίηση μεταβλητών ενσωματώσαμε τα βήματα της βασικής ροής 3 και 6 της προηγούμενης έκδοσης στο βήμα 3 της καινούργιας έκδοσης και προσθέσαμε το βήμα 8 όπου το σύστημα θα καταχωρεί την νέα εγγραφή μεταβλητής που δημιουργήθηκε από τον χρήστη. Στο use case εκτέλεση δεν υφίσταται πια κουμπί εκκαθάρισης αλλά κουμπί ελέγχου (check) και έγινε η πρόσθεση 2 απαραίτητων εναλλακτικών ροών για τις περιπτώσεις οπού ο χρήστης έχει χρησιμοποιήσει δομή ελέγχου if και βρόγχο while. Στο use case διαγραφή block εντολών αφαιρέσαμε από την βασική ροή του use case την υπόθεση ότι το block που διαγράφτηκε αποτελεί μέλος εντολής δυαδικού block και το θέσαμε ως εναλλακτική ροή (αποτελεί την εναλλακτική ροή 2). Στο use case undo αλλάξαμε όλο το use case δηλαδή και την βασική ροή και την εναλλακτική του ροή. Στο use case υπορουτίνα από την προηγούμενη έκδοση Use-case-v0.1 έγιναν αλλαγές στις ενναλακτικές ροές του. Στο use case debug έγινε μια μικρή αλλαγή στο βήμα 8 της βασικής ροής και στην εναλλακτική ροή 2.

**Σημαντική σημείωση: Στο use-case-v0.2 έχει προστεθεί και ένα ακόμα use case το redo (το οποίο είναι παρόμοιας λογικής με το use case του undo) αλλά δεν το συμπεριλαμβάνουμε στα δέκα σημαντικά use cases του προτζεκτ που θα υλοποιήσουμε.**

**Ρόλοι ομάδας:**

**Editor:** Σωτήριος Χασιώτης, Νυμφοδώρα Μπουζιάνη

**Contributors:** Κωνσταντίνος Καρακίτσος, Βασίλειος Μιχανετζής, Μιχαήλ Σοκολάκης

**Use case Diagram:**

****

**Αναλυτική περιγραφή των use cases**

**Use case**: Δήλωση και αρχικοποίηση μεταβλητών

**Βασική ροή:**

**1**. Ο χρήστης επιλέγει το πεδίο δημιουργίας νέας μεταβλητής.

**2**. Το σύστημα εμφανίζει στον χώρο των δηλώσεων των μεταβλητών ένα μπλοκ με δύο κενές θέσεις εισαγωγής.

**3**. Ο χρήστης ορίζει το όνομα της μεταβλητής και εισάγει τιμή και πατάει το κουμπί ελέγχου.

**4**. Το σύστημα ελέγχει την σύνταξη του ονόματος και διαπιστώνει ότι είναι σωστή.

**5**. Το σύστημα ελέγχει αν υπάρχει άλλη μεταβλητή με το ίδιο όνομα και διαπιστώνει ότι δεν υπάρχει.

**7**. Το σύστημα ελέγχει αν η τιμή είναι ακέραια και διαπιστώνει ότι είναι.

**8**.Το σύστημα καταχωρεί την νέα εγγραφή μεταβλητής.

**Εναλλακτική Ροή 1:**

**4.1.1.** Το σύστημα διαπιστώνει ότι η σύνταξη είναι λάθος.

**4.1.2.** Το σύστημα χρωματίζει κόκκινο το μπλοκ.

**Εναλλακτική Ροή 2:**

**7.2.1.** Το σύστημα διαπιστώνει ότι η τιμή που έχει δοθεί δεν είναι ακέραια τιμή.

**7.2.2.** Το σύστημα χρωματίζει κόκκινο το μπλοκ.

**Use case**: Δημιουργία μπλοκ εντολών

**Βασική Ροή:**

**1.** O χρήστης δημιουργεί ένα καινούργιο τύπου block που επιθυμεί.

**2**. Το σύστημα δημιουργεί το μπλοκ που επέλεξε ο χρήστης.

**3**. Το σύστημα αλλάζει το χρώμα του μπλοκ σε κόκκινο.

**4**. Ο χρήστης εισάγει την τιμή και την μεταβλητή.

**5**. Το σύστημα ελέγχει την μεταβλητή και την τιμή της ως προς την εγκυρότητα.

**6**. Το σύστημα ελέγχει αν η μεταβλητή είναι δηλωμένη.

**7.** Το σύστημα κάνει το χρώμα του μπλοκ πράσινο, ως ένδειξη ότι δεν βρέθηκαν λάθη κατά τον έλεγχο στα βήματα 5 και 6.

**8.** Ο χρήστης παίρνει το νέο του μπλοκ και το τοποθετεί στην βασική ροή του προγράμματος ή του function του.

**Εναλλακτική Ροή 1:**

**6.1.1.** Το σύστημα διαπιστώνει ότι ο χρήστης έχει εισαγάγει μεταβλητή η οποία δεν έχει δηλωθεί.

**6.1.2.** Το σύστημα κάνει κόκκινο το μπλοκ.

**6.1.3**. Ο χρήστης εισάγει εκ νέου μεταβλητή.

**6.1.4.** Συνέχεια από το βήμα 4 της βασικής ροής.

**Εναλλακτική Ροή 2:**

**8.2.1.**Ο χρήστης αντί να τοποθετήσει το νέο του μπλοκ στην βασική ροή του προγράμματος ή του function του, πατάει ξανά το κουμπί δημιουργίας μπλοκ.

**8.2.2.** Το σύστημα ενημερώνει το χρήστη, ότι πρέπει να πάρει το νέο του μπλοκ από το πεδίο δημιουργίας πριν φτιάξει άλλο.

**Use case**: Compile

**Βασική Ροή:**

**1**. Ο χρήστης κάνει compile εφόσον έχει ολοκληρώσει την δημιουργία του κώδικα του.

**2**. Το σύστημα ελέγχει αν υπάρχουν start και end blocks.

**3.** Το σύστημα διαπερνά την λίστα με τα blocks και ελέγχει αν κάποιο από αυτά περιέχει συντακτικό λάθος στο όνομα του ή δεν έχει δηλωθεί.

**4**. Το σύστημα κάνει έλεγχο για δομές όπως if και βρόγχους while για να βρει τα αντίστοιχα end τους.

**5**. Το σύστημα ελέγχει για function block και εφόσον υπάρχουν διαπερνά, με τον ίδιο τρόπο του βήματος 3, την λίστα τους επαναλαμβάνοντας το δεύτερο, το τρίτο και το τέταρτο βήμα .

**6**. Εφόσον όλα είναι έγκυρα το σύστημα εμφανίζει στον χρήστη αντίστοιχο μήνυμα επιτυχίας.

**7**. Το σύστημα εμφανίζει στον χρήστη την επιλογή να τρέξει το πρόγραμμα του.

**Εναλλακτική Ροή 1:**

**2.1.1.** Το σύστημα διαπιστώνει ότι ο χρήστης δεν έχει εισάγει start ή/και end block

**2.1.2.** Το σύστημα ενημερώνει το χρήστη για απουσία start ή/και end block.

**Εναλλακτική Ροή 2:**

**3.2.1.** Το σύστημα διαπιστώνει ότι ο χρήστης έχει κάνει συντακτικό λάθος κατά τη δημιουργία κάποιου block.

**3.2.2.** Το σύστημα ενημερώνει τον χρήστη επισημαίνοντας πού εντοπίζονται τα συντακτικά λάθη καθώς και ποια είναι αυτά.

**Εναλλακτική Ροή 3:**

**4.3.1.** Το σύστημα διαπιστώνει ότι ο χρήστης δεν έχει προσθέσει end if και end while blocks για κάθε while και if block.

**4.3.2.** Το σύστημα ενημερώνει το χρήστη σε ποια σημεία του κώδικα λείπουν τα end blocks.

**Use case**: Εκτέλεση

**Βασική Ροή:**

**1.** Ο χρήστης εκτελεί το πρόγραμμα.

**2**. Το σύστημα ελέγχει αν έχει γίνει compile μετά από την τελευταία αλλαγή του κώδικα.

**3.** Το σύστημα ξεκινάει την εκτέλεση του κώδικα.

**4.** Το σύστημα εμφανίζει φωτεινό περίγραμμα γύρω από την εντολή προς εκτέλεση.

**5.** Το σύστημα κατά την εκτέλεση του κάθε μπλοκ πριν πάει στο επόμενο μπλοκ θα υποστεί μια μικρή αναμονή .

**6.** Το σύστημα εκτελεί την εντολή του προγράμματος που βρίσκεται εκείνη την στιγμή, γίνεται εκτύπωση των μεταβλητών στην οθόνη εξόδου, ο δείκτης προχωράει κατά ένα και πάμε στο βήμα 4.

**7**. Όταν φτάσει στο end block το σύστημα κρατάει τις τελευταίες τιμές στην οθόνη εξόδου .

**8.** Ο χρήστης πατάει το κουμπί ελέγχου (check).

**9.** Το σύστημα επαναφέρει την οθόνη εξόδου στην αρχική της κατάσταση.

**Εναλλακτική Ροή 1:**

**2.1.1.** Το σύστημα διαπιστώνει ότι δεν έχει γίνει compile.

**2.1.2.**Εμφανίζεται μήνυμα λάθους στον χρήστη.

**Εναλλακτική Ροή 2:  
6.2.1.** Το σύστημα διαπιστώνει ότι η εντολή είναι τύπου δομήελέγχου if και αποτελεί αληθής δήλωση.

**6.2.2.**Ο δείκτης προχωράει κατά ένα.

**Εναλλακτική Ροή 2.1 (Εναλλακτική Ροή της Εναλλακτικής Ροής 2):**

**(6.2.1).2.1.** Το σύστημα διαπιστώνει ότι η εντολή είναι τύπου δομήελέγχου if και αποτελεί εσφαλμένη δήλωση.

**(6.2.1).2.2.** Ο δείκτης επιστέφει στο ζευγάρι του end if.

**Εναλλακτική Ροή 3:**

**6.3.1.** Το σύστημα διαπιστώνει ότι η εντολή είναι τύπου βρόγχου while και αποτελεί αληθής δήλωση.

**6.3.2.** Ο δείκτης μετακινείται και γίνεται είσοδος στον βρόγχο while για να εκτελεστούν όλες οι εντολές μέχρι το end while .

**6.3.3.** O δείκτης επιστρέφει στο αρχικό while εφόσον η συνθήκη είναι αληθής ώστε να γίνει επανάληψη του βρόγχου.

**Εναλλακτική Ροή 3.1 (Εναλλακτική Ροή της Εναλλακτικής Ροής 3):**

**(6.3.3).3.1.** O δείκτης προχωράει κατά ένα από το end while και συνεχίζει κανονικά με την επόμενη εντολή εφόσον η συνθήκη while είναι εσφαλμένη .

**Use case**: Μετακίνηση και ένωση block εντολών

**Βασική Ροή:**

**1.** Ο χρήστης επιλέγει το μπλοκ που επιθυμεί να μετακινήσει μέσω drag.

**2.** Ο χρήστης μετακινεί το block πάνω από την θέση που θέλει να το τοποθετήσει.

**3.** Το σύστημα ελέγχει αν το block που μετακινεί ο χρήστης βρίσκεται σε κοντινή απόσταση από κάποιο άλλο.

**4.** Το σύστημα χρωματίζει την θέση στην οποία αναμένεται να προστεθεί το block.

**5.** Το σύστημα μετακινεί τα block που βρίσκονται δεξιά της θέσης αυτής κατά μία θέση δεξιά.

**6.** Ο χρήστης αφήνει το block στη θέση που επιθυμεί.

**7.** Το σύστημα επισυνάπτει το μετακινούμενο block στο κοντινό του block, δημιουργώντας την επιθυμητή σύνδεση.

**8.** Το σύστημα ελέγχει αν το block που μετακινήθηκε μπορεί να ενταχθεί στην κύρια ροή του κώδικα ή αν ανήκει σε κάποια υπορουτίνα βάση της θέσης του και της δομής του προγράμματος.

**9.** Ανάλογα με τον έλεγχο, το σύστημα προσθέτει το μετακινούμενο μπλοκ στην κατάλληλη θέση της λίστας.

**Εναλλακτική Ροή 1:**

**2.1.1.** Ο χρήστης προσπαθεί να μετακινήσει το μπλοκ εκτός ορίων οθόνης.

**2.1.2.** Το σύστημα κρατάει το block στα όρια του επιτρεπτού πεδίου.

**Use case**: Διαγραφή block εντολών

**Βασική Ροή:**

1. O χρήστης διαγράφει το block που επιθυμεί.

2. Το σύστημα εμφανίζει παράθυρο επιβεβαίωσης της διαγραφής.

3. O χρήστης επιβεβαιώνει.

4. Το σύστημα κλείνει το παράθυρο.

5. Το σύστημα ελέγχει αν το block που διαγράφτηκε αποτελεί μέλος εντολής δυαδικού block.

6. Το σύστημα αφαιρεί από τη λίστα του block διαγραφής .

7. Tο σύστημα διαγραφεί το block.

8. Το σύστημα συνδέει το επόμενο μπλοκ από αυτό που διαγράφτηκε στο προηγούμενο και επαναλαμβάνει μέχρι το τέλος των εντολών.

**Εναλλακτική Ροή 1:**

**3.1.1.** Ο χρήστης δεν επιβεβαιώνει.

**3.1.2.** Το σύστημα επιστρέφει το χρήστη στην κύρια οθόνη.

**Εναλλακτική Ροή 2:**

**5.2.1.** Το μπλοκ που διαγράφτηκε αποτελεί μέλος εντολής δυαδικού μπλοκ.

**5.2.2.** Το σύστημα επισημαίνει με κόκκινο.

**Use case**: Αποθήκευση

**Βασική Ροή:**

**1.** Ο χρήστης αποθηκεύει το πρόγραμμα.

**2.** Το σύστημα εμφανίζει ένα παράθυρο για όνομα αρχείου και τοποθεσία αποθήκευσης και δύο κουμπιά cancel και Save As.

**3.** Ο χρήστης εισάγει το όνομα του αρχείου.

**4.** Το σύστημα ελέγχει το όνομα του αρχείου ως προς την εγκυρότητα.

**5.** Εφόσον το όνομα του αρχείου είναι έγκυρο, ο χρήστης επιλέγει τοποθεσία αποθήκευσης.

**6.** Ο χρήστης κάνει Save As .

**7.** Το σύστημα αποθηκεύει το αρχείο στην επιλεγμένη τοποθεσία.

**8.** Το σύστημα εμφανίζει μήνυμα επιτυχούς αποθήκευσης.

**Εναλλακτική Ροή 1:**

**4.1.1.** Το σύστημα διαπιστώνει ότι το όνομα που εισήγαγε ο χρήστης δεν είναι έγκυρο.

**4.1.2.** Το σύστημα εμφανίζει στον χρήστη μήνυμα λάθους.

**4.1.3.** Η περίπτωση χρήσης συνεχίζεται από το βήμα 3 της βασικής ροής.

**Εναλλακτική Ροή 2:**

**2.2.1.** Ο χρήστης πατάει το κουμπί cancel.

**2.2.2.** Το σύστημα κλείνει το παράθυρο χωρίς να γίνεται η αποθήκευση του αρχείου.

**2.2.3.** Το σύστημα επιστρέφει τον χρήστη στην κύρια οθόνη.

**Use case**: Undo

**Βασική Ροή:**

**1.** Ο χρήστης κάνει αλλαγές στο πρόγραμμα του.

**2.** Το σύστημα αποθηκεύει τις τελευταίες δύο σε μορφή στιγμιότυπου.

**3.** Ο χρήστης κάνει Undo.

**4.** Το σύστημα ελέγχει την ύπαρξη στιγμιότυπων.

**5.** Το σύστημα έχει περισσότερα από ένα στιγμιότυπα και αποθηκεύει το τρέχον στιγμιότυπο.

**6.** Το σύστημα κάνει εκκαθάριση οθόνης.

**7.** Το σύστημα ανακατασκευάζει το πρότζεκτ στην προηγούμενη κατάσταση.

8. Το σύστημα ενημερώνει την οθόνη καμβά.

**9.** Το σύστημα ενεργοποιεί το κουμπί Redo.

**Εναλλακτική Ροή 1:**

**4.1.1.** Το σύστημα ελέγχει την ύπαρξη στιγμιότυπων και διαπιστώνει ότι υπάρχει μόνο ένα στιγμιότυπο.

**4.1.2.**Το κουμπί undo απενεργοποιείται και ο χρήστης παραμένει με την ίδια κατάσταση του προγράμματος του.

**Use case**: Υπορουτίνα

**Βασική Ροή:**

**1.** Ο χρήστης δημιουργεί μια νέα υπορουτίνα εφόσον δεν υπάρχει άλλη υπορουτίνα.

**2.** Το σύστημα εμφανίζει ένα κενό πεδίο στο οποίο ζητά από το χρήστη να εισάγει όνομα .

**3.** Ο χρήστης πληκτρολογεί όνομα για τη νέα function και έπειτα συνεχίζει.

4. Το σύστημα ελέγχει την εγκυρότητα του ονόματος της υπορουτίνας.

**5.** Το σύστημα εμφανίζει τα block "Start Function" και "End Function" μεταξύ των οποίων ο χρήστης πρέπει να τοποθετήσει μπλοκ εντολών και το ίδιο το μπλοκ κλήσης του function.

**6.** Ο χρήστης τοποθετεί τα block εντολών που επιθυμεί μεταξύ των μπλοκ "Start Function" και "End Function".

**6.** Το σύστημα αποθηκεύει τους δείκτες τους σε νέα λίστα από αυτή της βασικής ροής του προγράμματος.

**7.** O χρήστης βάζει το block κλήσης της function στην main.

**8.** Kατά την εκτέλεση του προγράμματος το σύστημα συναντά το block κλήσης, μεταφέρεται στην λίστα του function όπου εκτελεί ό,τι έχει μέσα και επιστρέφει στην main.

**Εναλλακτική Ροή 1:**

**3.1.1.** Ο χρήστης πληκτρολογεί μη έγκυρο όνομα στο κενό πεδίο και κάνει κλικ στο κουμπί "Continue".

**3.1.2.** Το σύστημα εμφανίζει μήνυμα λάθους.

**Εναλλακτική Ροή 2:**

**1.2.1.** Ο χρήστης πατάει το κουμπί new function, ενώ έχει ήδη δημιουργήσει μια υπορουτίνα.

**1.2.2.** Το σύστημα ενημερώνει τον χρήστη με μήνυμα σφάλματος.

**Εναλλακτική Ροή 3:**

**3.3.1**. Ο χρήστης αντί να συνεχίσει , ακυρώνει την διαδικασία.

**3.3.2.** Δεν υπάρχουν αλλαγές στο πρόγραμμα.

**Use case**: Debug Mode

**Βασική ροή:**

**1.** Ο χρήστης κάνει debug το πρόγραμμα και επιλέγει τα blocks που επιθυμεί να γίνει break της εκτέλεσης.

**2.** Tο σύστημα αλλάζει το χρώμα των επιλεγμένων blocks.

**3.** Εφόσον έχουν μπει breakpoints, το σύστημα ενεργοποιεί το debug mode.

**4.** Ο χρήστης κάνει εκτέλεση του προγράμματος.

**5.** Το σύστημα ελέγχει αν έχει γίνει compile από την τελευταία αλλαγή.

**6.** Το σύστημα ξεκινάει την εκτέλεση σε λειτουργία debug και εκτελεί μέχρι το πρώτο breakpoint.

**7.** Tο σύστημα κάνει ενημέρωση των νέων τιμών στο πεδίο εξόδου.

**8**. Tο σύστημα ελέγχει αν υπάρχει επόμενο breakpoint και περιμένει μέχρι ο χρήστης πατήσει run και επαναληφθούν τα βήματα 6,7,8 της βασικής ροής.

**9**. Μόλις το πρόγραμμα τερματίσει, το σύστημα απενεργοποιεί το debug mode.

**Εναλλακτική Ροή 1**

**5.1.1.** Το σύστημα διαπιστώνει ότι δεν έχει γίνει compile .

**5.1.2.** Εμφανίζεται μήνυμα λάθους στον χρήστη.

**Εναλλακτική Ροή 2**

**5.2.1.** Ο χρήστης κάνει debug το πρόγραμμα του χωρίς να βάλει breakpoints.

**5.2.2.** Το σύστημα εκτελεί όλο το πρόγραμμα.

**Use case: Redo**

**Βασική Ροή:**

1. Ο χρήστης έχει πατήσει το κουμπί undo τουλάχιστον μια φορά .
2. Το σύστημα ενεργοποιεί το κουμπί Redo.
3. Ο χρήστης πατάει το κουμπί Redo.
4. Το σύστημα ελέγχει για την ύπαρξη αποθηκευμένων στιγμιότυπων.
5. Το σύστημα αποθηκεύει την τρέχουσα κατάσταση στο ιστορικό-undo.
6. Το σύστημα ανακτά το στιγμιότυπο που αναιρέθηκε τελευταίο.
7. Το σύστημα κάνει εκκαθάριση οθόνης.
8. Το σύστημα ανακατασκευάζει το πρότζεκτ στην κατάσταση που έχει αναιρεθεί τελευταία.
9. Το σύστημα αφαιρεί το επαναφερόμενο στιγμιότυπο από το ιστορικό-redo.
10. Το σύστημα ενεργοποιεί το κουμπί undo.

**Εναλλακτική Ροή 1:  
4.1.1.** Το σύστημα ελέγχει την ύπαρξη στιγμιότυπων και διαπιστώνει ότι υπάρχει μόνο ένα στιγμιότυπο.

**4.1.2.**Το κουμπί redo απενεργοποιείται και ο χρήστης παραμένει με την ίδια κατάσταση του προγράμματος του.