ΕΡΓΑΣΙΑ #4 ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΗΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Α.Μ. : 1115201200106
Όνομα : Βασίλειος
Επίθετο : Μαυρομμάτης

ΠΑΡΟΥΣΙΑΣΗ ΚΩΔΙΚΑ

Η εργασία περιέχει τα **11** παρακάτω **αρχεία** :
Τα **9 πηγαία** αρχεία **Citizen.java**, **Escort.java**, **Building.java**, **Space.java**, **EntryArea.java**, **Floor.java**, **Ground.java**, **Elevator.java**, **Office.java** τα οποία υλοποιούν τις αντίστοιχες κλάσεις και συμπεριφορές. **1 πηγαίο αρχείο SimC1.java** το οποίο συνδέει τα παραπάνω σε μία προσομοίωση λειτουργίας του κτηρίου. **1 makefile** για την δημιουργία των αντίστοιχων αρχείων **.class**.
Αρχικά η main αρχικοποιεί, μέσω των 7 ορισμάτων της γραμμής εντολών τις διαστάσεις του κτηρίου τους πολίτες που θέλουμε καθώς και τους κύκλους λειτουργείας του ασανσέρ στις αντίστοιχες παραμέτρους και γίνεται ο απαραίτητος έλεγχος ακεραιότητας δεδομένων.
Στην συνέχεια ξεκινάει η προσομοίωση δημιουργώντας το κτήριο καθώς και τους πολίτες, δίνοντας τους τυχαίους προορισμούς ορόφου και γραφείου. Γίνεται απόπειρα εισόδου όλων αυτών με την σειρά στο κτήριο και στην συνέχεια υπολογίζοντας τον αριθμό που μπήκαν στο ισόγειο και πήραν αριθμό προτεραιότητας και αφαιρώντας τον από αυτους που δεν χώρεσαν και έμειναν μέσα στο κτήριο εισερχόμαστε στον κεντρικό βρόγχο του προγράμματος. Εδώ εκτελούνται οι απαραίτητες επαναλήψεις με την operate στις οποίες το ασανσερ πρώτα γεμίζει στο ισόγειο όσους χωράει με βάση την προτεραιότητα και στην συνέχεια ανεβαίνωντας ορόφους, βρίσκωντας τους προορισμούς του κάθε επιβάτη και χρησιμοποιώντας τον αριθμό προτεραιότητας του αν υπάρχει χώρος επιχειρεί να αδειάσει τους επιβάτες. Από τους ήδη υπάρχωντες στον όροφο καθώς και τους νεοεισέρχοντες απτο ασανσέρ επιλέγεται πάλι με βάση την προτεραιότητα ο πολίτης και αν υπάρχει χώρος εισέρχεται στο γραφείο διαφορετικά παραμένει στόν χώρο εισόδου του ορόφου. Έπειτα το ασανσερ κατεβαίνει ορόφους και για κάθε γραφείο του ορόφου επιλέγονται τυχαίος αριθμός ατόμων που θεωρείται οτι εξυπηρετήθηκαν και αν χωράνε στο ασανσερ μπαίνουν. Τέλος ο κύκλος λειτουργίας τελείωνει με αποβίβαση όλων των επιβατών που έχουν εξυπηρετηθεί και έξοδο τους από το κτήριο. Ξαναγίνεται απόπειρα να μπούν καινούρια άτομα στον χώρο του ισογείου για να πάρουν προτεραιότητα καθώς και απόπειρα να μπουν επισκέπτες εκτός του κτηρίου και ο βρογχος επανεκκινεί. Καθόλη την παραπάνω διαδικασία εκτυπώνονται διαγνωστικά μηνύματα στο stdout από όλους τους εμπλεκόμενους (ένψυχους και μη) καθώς και η τρέχουσα κατάσταση του ασανσέρ σε κάθε αλλαγή ορόφου.

ΠΑΡΑΤΗΡΗΣΕΙΣ

Οι διαφορές σε σχέση με την Άσκηση3 που έχει υλοποιηθεί σε **C++** είναι οι εξής:

1) Προσθήκη της κλάσης **Escort** ως υποκλάση της **Citizen** και προσθήκη των αντίστοιχων data members και functions για διαφοροποίηση μεταξύ των δύο.

2) Και οι Escorts και οι Citizens αποθηκεύονται στον κοινό πίνακα AngryNWaiting.

3) Ερμήνευσα την εκφώνηση με τον εξής τρόπο: Για κάθε Citizen x απ'τους K που δημιουργώ εισάγω πιθανότητα να είναι ο επόμενος (x+1) escort του x. Επομένως δημιουργούνται συνολικά K Citizens + Escorts.

4) Σε όλους τους χώρους μπαίνει ο κάθε Citizen μόνο αν χωράει και ο Escort του (αν υπάρχει) αλλιώς ελέγχεται ο επόμενος.

5) Τροποποίηση των constructors και αφαίρεση όλων των destructors και των εντολών delete μιας και μας καλύπτει ο garbage collector της java, καθώς και εκτενείς αλλαγές σε συναρτήσεις για ενσωμάτωση των Escorts.

ΔΗΜΙΟΥΡΓΕΙΑ ΕΚΤΕΛΕΣΙΜΟΥ ΚΑΙ ΕΚΤΕΛΕΣΗ

$ make
$ java SimC1 N Nf Ng No Nl K L(7 args διαχωρισμένα με κενό)
$ make clean

ΥΛΟΠΟΙΗΘΗΚΕ ΣΕ:

OS: Ubuntu 18.04
Text Editor: Sublime Text 3
Debugger: jdb version 9.0 (Java SE version 10.0.2)
Java environment: openjdk 10.0.2 2018-07-17
OpenJDK Runtime Environment (build 10.0.2+13-Ubuntu-1ubuntu0.18.04.4)
OpenJDK 64-Bit Server VM (build 10.0.2+13-Ubuntu-1ubuntu0.18.04.4, mixed mode)