Λουκας Οζουνογλου (ΑΜ: 3200270)
Βασιλης Σουλιος (ΑΜ:3200181)

Μερος Α)

Για την υλοποίηση του μέρους Α χρειάζεται να δημιουργήσουμε μια λίστα μονής σύνδεσης (List). Στη λίστα αυτή περιέχονται συγκεκριμένες μέθοδοι: Η “insertAtFront”, η οποία προσθέτει στην κεφαλή – “head” ενώ η “insertAtBack”, στην ουρα – “tail” της λίστας το εκάστοτε στοιχείο. Αυτό βεβαία συμβαίνει όταν η λίστα δεν είναι κενή. Εάν η λίστα είναι κενή, η κεφαλή με την ουρά είναι ίσες.
Στη μέθοδο “removeFromFront” αφαιρούμε από την ουρά το πρώτο στοιχείο, δηλαδή το “head” ενώ στη “removeFromBack” το τελευταίο στοιχείο, δηλαδή το “tail”.

Η κλάση “StringStackimpl” η οποία υλοποιεί τις μεθόδους της διεπαφής “StringStack” δηλώνει μέσα της μια λιστα που αναφέρεται παραπάνω και αποθηκεύει στοιχειά όπως και αντικείμενα τύπου “node”. Μες τις μέθοδος “push” και “pop” εισάγεις και εξάγεις στοιχειά από την στοίβα. Η μέθοδος “size” και “isEmpty” επιστρέφουν το μέγεθος και αντίστοιχα, αν η στοίβα είναι κενή. Τείος η μέθοδος “peek” αντιγράφει το πρώτο στοιχείο της στοίβας χωρίς να το διαγράφει.

Η κλάση “StringQueuelmpl” η οποία υλοποιεί τις μεθόδους της διεπαφής “StringQueue” όπως και η “StringStack” δηλώνεται μέσα της μια λίστα που αναφέρεται παραπάνω και αποθηκεύει στοιχειά όπως και αντικείμενα τύπου “node”. Στις μεθόδους “put” και “get” όπως και στη στοίβα, εισάγεις και εξάγεις στοιχειά και αντίστοιχα οι κλάσεις “size”, “isEmpty” και ”peek” υλοποιούν τις ίδιες λειτουργίες όπως αυτές της στοίβας.

Μέρος Β)

Αρχικά ,για την υλοποίηση του προγράμματος “Thiseas” δημιουργούμε δυο στοίβες με τις συντεταγμένες. Επιπλέον δηλώνουμε ένα αντικείμενο τύπου scanner στο οποίο ο χρήστης εισάγει την διαδρομή του αρχείο txt . Εν συνεχεία, ο κώδικας περνάει τα πρώτα στοιχειά, δηλαδή τις διαστάσεις του λαβυρίνθου όπως και την αρχική θέση. Έπειτα ελέγχει τα δεδομένα ώστε να μην βρεθεί λάθος (π.χ. στο μέγεθος του πινάκα) και δημιουργεί ένα δισδιάστατο πινάκα με τα δεδομένα του λαβυρίνθου. Τα στοιχειά της εισόδου αποθηκεύονται στις στοίβες ενώ δημιουργείται ένας πινάκας με όνομα “visited” που δηλώνει αν ο χρήστης έχει περάσει από το σημείο αυτό. Μετά, δηλώνεται μια μεταβλητή τύπου boolean και άλλες δυο τύπου integer οι οποίες με την μέθοδο peek αποθηκεύονται οι συνταγμένες από το τελευταίο βήμα του χρήστη. Ουσιαστικά , ο αλγόριθμος ελέγχει με μια λούπα (while) τις γύρω περιοχές του πινάκα δεξιόστροφα και ελέγχει τις περιπτώσεις που οι δείκτες δεν ξεπερνούν τις διαστάσεις του πινάκα. Επίσης ο κώδικας όταν βρίσκει αδιέξοδο επιστρέφει στις προηγούμενες συντεταγμένες και δηλώνει την αδιέξοδο ως “X” στο πινάκα ώστε να μην το ξανά επισκεύτεται. Στην εκάστοτε λούπα καλείται η μέθοδος “solved” η οποία ελέγχει αν το συγκεκριμένο σημείο είναι έξοδος. Εν κατακλείδι σε περίπτωση εξόδου το πρόγραμμα εμφανίζει τις τελικές συντεταγμένες ενώ στην περίπτωση που δεν υπάρχει έξοδος εμφανίζει αντίστοιχο μήνυμα.

Μέρος Γ)

Στο τρίτο μέρος δημιουργούμε μια κυκλική λίστα (circularlist) μεσώ της οποίας αποθηκεύουμε στοιχειά. Στην συνεχεία, υλοποιούμε την κλάση StringQueueWithOnePointer η οποία περιεχέι την μέθοδο “get”,”size”,”put” και ”isEmpty”.Οι μέθοδοι put και get εισάγουν και εξάγουν από την ουρά, το size επιστρέφει το μέγεθος ενώ το isEmpty επιστρέφει αν είναι κενή. Η ιδέα στην ουρά με ένα δείκτη είναι ότι με την χρήση της κυκλικής λίστας αποφεύγουμε την χρήση δεύτερου δείκτη,

διότι το αντικείμενο Node που φτιάξαμε στην λίστα περιεχέι δυο μεταβλητές, τις head και tail.