DataStructures

ΆΣΚΗΣΗ 1 ΜΕΡΟΣ Δ

1. **Node.java:** Χρησιμοποιούμε αυτή την κλάση για να δημιουργήσουμε αντικείμενα τύπου Node τα οποία τα χρησιμοποιούμε για να υλοποιήσουμε τις δομές δεδομένων ουρά και στοίβα
    - Τα attributes της κλάσης είναι τα data τύπου T ώστε να αποθηκεύει όλους τους τύπους δεδομένων,next τύπου Node<T> που δείχνει στον επόμενο κόμβο
    - Μέθοδοι set και get για να έχουμε πρόσβαση στα attributes της κλάσης
2. **StringQueueImpl.java:** Η κλάση αυτή έχει ως attributes 2 αντικείμενα head,tail τύπου Node που βοηθάνε στην υλοποίηση συγκεκριμένων μεθόδων και την μεταβλητή size που περιέχει το μέγεθος της ουράς
    - isEmpty(): Ελένχει αν ο head node δηλαδή η αρχή της ουράς είναι null δηλαδή αν η ουρά είναι άδεια
    - put(T item): Δημιουργεί τον κόμβο που εισάγουμε και τσεκάρει αν η ουρά είναι άδεια.Αν είναι άδεια βάζει τον κόμβο στην πρώτη θέση και ενημερώνει τους δείκτες head και tail να δείχνουν στον κόμβο αυτον αλλιώς τοποθετεί τον κόμβο στο τέλος της ουράς ορίζει το next του να δείχνει στον επόμενο ο οποίος εκείνη την στιγμή είναι ο tail και μετακινεί τον tail να δείχνει στον κόμβο που κάναμε put και αυξάνει το size
    - T get(): Ελένχει αν η ουρά είναι άδεια και αν είναι πετάει exception αλλιώς κρατάει τα δεδομένα του Node που δείχνει ο δείκτης head σε μια προσωρινή μεταβλητή.Αν οι δείκτες head και tail δείχουν στον ίδιο κόμβο τότε τους κάνουμε null αλλιώς βάζουμε τον head να δείχνει στον επόμενο του.Τέλος επιστρέφουμε τα δεδομένα που περιέχει η προσωρινή μεταβλητή και μειώνει το size
    - T peek() : Ελένχει αν η ουρά είναι άδεια και αν είναι πετάει exception αλλιώς επιστρέφει τα δεδομένα του Node που δείχνει ο δείκτης head και μειώνει το size
    - printQueue(PrintStream stream) : Φτιάχνουμε έναν προσωρινό κόμβο που δείχνει στο head και με ένα loop τον ανανεώνουμε με τον επόμενο μέχρι να διατρέξουμε ολη την ουρά
    - int size(): Επιστρέφει το μέγεθος της ουράς
3. **StringStackImpl.java:** Η κλάση αυτή έχει ως attributes 1 αντικείμενο top τύπου Node που βοηθάει στην υλοποίηση συγκεκριμένων μεθόδων και την μεταβλητή size που περιέχει το μέγεθος της στοίβας
    - isEmpty(): Ελένχει αν ο top node είναι null δηλαδή αν η στοίβα είναι άδεια
    - push(T item): Δημιουργεί τον κόμβο που εισάγουμε και τσεκάρει αν η στοίβα είναι άδεια.Αν είναι άδεια βάζει τον κόμβο στην πρώτη θέση και ενημερώνει τον δείκτη top να δείχνει στον κόμβο αυτόν αλλιώς τοποθετεί τον κόμβο που δείχνει ο top ως next και κάνει top τον κόμβο που δημιουργήσαμε

- Ελένχει αν η στοίβα είναι άδεια και αν είναι πετάει exception αλλιώς κρατάει τα δεδομένα του Node που δείχνει ο δείκτης head σε μια προσωρινή μεταβλητή.Αν το μέγεθος της στοίβας είναι 1 τότε κάνει τον δείκτη top null αλλιώς θέτει ως top τον επόμενο του δηλαδή διαγράφει τον κόμβο που έδειχνε ο top και επιστρέφει τα δεδομένα του
- Ελένχει αν η στοίβα είναι άδεια και αν είναι πετάει exception αλλιώς επιστρέφει τα δεδομένα του Node που δείχνει ο δείκτης top και μειώνει το size
- printQueue(PrintStream stream) : Φτιάχνουμε έναν προσωρινό κόμβο που δείχνει στο top και με ένα loop τον ανανεώνουμε με τον επόμενο μέχρι να διατρέξουμε ολη την στοίβα
- int size(): Επιστρέφει το μέγεθος της στοίβας

4. **StringQueueWithOnePointer.java:** Η κλάση αυτή κάνει ακριβώς οτι και η StringQueueImpl απλά η μόνη διαφορά είναι στην υλοποίηση που αντί για 2 δείκτες χρησιμοποιούμε μόνο έναν τον tail
    - isEmpty(): Ελένχει αν ο tail node δηλαδή το τέλος της ουράς είναι null δηλαδή αν η ουρά είναι άδεια
    - put(T item): Φτιάχνει έναν καινούργιο κόμβο και αρχικά ελεχνει αν η ουρά είναι null.Αν είναι απλά βάζει τον next του κόμβου να δείχνει στο ίδιο τον κόμβο και κάνει τον tail να δείχνει στον κόμβο αυτον αλλιώς ορίζουμε σαν next του κομβου που εισάγουμε τον tail.next δηλαδή κάθε φορά που εισάγουμε έναν κομβο επειδή ο tail.next θα δείχνει πάντα στον πρωτό έτσι και ο κόμβος που εισάγουμε επειδή μετα την εισαγωγή θα είναι ο tail θελουμε να συνεχίσει να δείχνει στον πρώτο.Ορίζουμε στον κόμβο που δείχνει ο tail να έχει ως επόμενο αυτόν που εισάγουμε.Έτσι επιτυνχάνουμε ότι κάεε κόμβος να δείχνει στον επόμενο του και ο tail που είναι ο τελευταίος να δείχνει παντα στον πρώτο.
    - T get(): Αρχικά ελένχει αν η ουρά είναι άδεια και πετάει exception αλλιώς κρατάει τα δεδομένα του tail.next δηλαδή του πρώτου κόμβου της ουράς.Αν η ουρά έχει μόνο εναν κόμβο κάνει τον tail null αλλιώς ορίζει ως tail.next τον tail.next.next δηλαδή τον δεύτερο κόμβο της ουράς και έτσι διαγράφουμε τον πρώτο μειώνουμε το size και επιστρέφουμε τα δεδομένα του
    - T peek(): Αρχικά ελένχει αν η ουρά είναι άδεια και πετάει exception αλλιώς επιστρέφει τα δεδομενα του tail.next δηλαδή του πρώτου κόμβου της ουράς
    - printQueue(PrintStream stream) : Φτιάχνουμε έναν προσωρινό κόμβο που δείχνει στο head και με ένα loop τον ανανεώνουμε με τον επόμενο μέχρι να διατρέξουμε ολη την ουρά
    - int size(): Επιστρέφει το μέγεθος της ουράς

5. **TxTRead.java:** Η κλάση αυτή έχει ως attributes την μταβλητή path ώστε να βρει το αρχείο που πρέπει να διαβάσει και τις συντεταγμένες του Ε που θα έχει το αρχείο μέσα ώστε να τις χρησιμοποιήσουμε στο main προγραμμά μας.Μέσα στην μέθοδο read διαβάζουμε το αρχείο

και λαμβάνουμε υπόψην όλες τις περιπτώσεις για τις οποίες το αρχείο αυτο μπορεί να μην είναι σωστό ώστε να τρέξει το προγραμμά μας.Η κλάση αυτή έχει σχόλια μέσα που λένε τι ελένχους κάνει και επίσης για κάθε πρόβλημα που βρίσκει τιπώνει στον χρήστη τι ακριβώς φταίει ώστε να το διορθώσει

6. **Thiseas.java:** Η γενική ιδέα του main προγράμματος που έχει ως στόχο να εντοπίζει την έξοδο σε κάθε λαβύρινθο που δίνεται από τον χρήστη είναι ότι αφού βρει ποιο έιναι το πρώτο βήμα (το πρώτο βήμα έχει υλοποιηθεί ξεχωριστά γιατί επειδή η είσοδος θα είναι πάντα στις άκρες του λαβυρίνθου υπάρχουν μόνο 3 δυνατά βήματα ενώ γενικα υπάρχουν 4 άρα για κάθε περίπτωση της εισόδου κάνουμε το κατάλληλο βήμα) ώστε να μπει στην while η οποία τερματίζει μόνο όταν βρεθούμε σε καποια έξοδο δηλαδή στην πρώτη ή τελευταία γραμμή ή στήλη του πίνακα που περιέχει τον λαβύρινθο.Σε κάθε βήμα ο αλγόριθμος κοιτάει να βρει 0 οπότε όποιο move κάνει θα είναι επειδή βρήκε 0 και κάθε βήμα το κάνει push στην στοίβα κάνοντας το περιεχόμενο του σημείου που ήταν 1.Αν βρεθεί σε αδιέξοδο δηλαδή όλα τα σημεία γύρω του είναι 1 τότε κάνει pop από την στοίβα μεχρι να γυρίσει σε κάποιο σημείο που είχε 0 που δεν διάλεξε.Αν η στοίβα αδιάσει σημαίνει ότι δεν μπορεί να βρει κάποιο μηδενικό στοιχείο και τυπώνει οτί ο λαβύρινθος δεν έχει έξοδο