Α Μέρος

Interface StringdoubleEndedQueue: Interface είναι ένα προσχέδιο μιας κλάσης java που περιέχει αφηρημένες μεθόδους και σταθερές ιδιότητες. Η διεπαφή περιλαμβάνει τη δήλωση μεθόδων και αυτές οι μέθοδοι θα οριστούν στην κλάση. Υλοποιούμε το interface στην κλάση StringDoubleEndedQueueImpl. Ακόμη, κάνουμε χρήση generics έτσι ώστε τα ακόλουθα προγράμματα να μπορούν να χρησιμοποιηθούν με οποιονδήποτε τύπο δεδομένων

Οι κλασεις στο StringDoubleEndedQueueImpl είναι:

- **isEmpty():** Ελέγχει αν η λίστα είναι άδεια και επιστρέφει True ή False.
- **getFirst():** Παίρνει το πρώτο αντικείμενο της λίστας, αν δεν είναι άδεια.
- **getLast():** Παίρνει το τελευταίο αντικείμενο της λίστας, αν δεν είναι άδεια.
- **addFirst():** Εισάγει στην αρχή της λίστας ένα αντικείμενο T σε χρόνο O(1) (εκτελείται σε σταθερό χρόνο, ανεξαρτήτως του ορίσματος).
- **addLast():** Εισάγει στο τέλος της λίστας ένα αντικείμενο T σε χρόνο O(1) (εκτελείται σε σταθερό χρόνο, ανεξαρτήτως του ορίσματος).
- **removeFirst():** Διαγράφει το πρώτο αντικείμενο της λίστας σε χρόνο O(1) (εκτελείται σε σταθερό χρόνο, ανεξαρτήτως του ορίσματος), ελέγχοντας πρώτα αν είναι άδεια.
- **removeLast():** Διαγράφει το τελευταίο αντικείμενο της λίστας σε χρόνο O(1) (εκτελείται σε σταθερό χρόνο, ανεξαρτήτως του ορίσματος), ελέγχοντας πρώτα αν είναι άδεια.
- **size():** Επιστρέφει το μέγεθος του πίνακα. Κάθε φορά που μπαίνει σε μια add (addFirst ή addLast) αυξάνει το size κατά 1 και αντίστοιχα το μειώνει σε περίπτωση που μπει σε remove (removeFirst ή removeLast).
- **printQueue():** Τυπώνει τα αντικείμενα της λίστας με τη σειρά, ξεκινώντας από το πρώτο στοιχείο.

Σε αυτό το σημείο να αναφερθεί πως ο λόγος που το πρόγραμμα εκτελείται σε O(1) χρόνο, είναι επειδή κάθε φορά που εισάγεται ή διαγράφεται στοιχείο από την ουρά, αυξομειώνεται ανάλογα και το μέγεθος της μεταβλητής size, και δεν χρειάζεται να υπολογιστεί μετά την επεξεργασία της ουράς.

Β Μέρος

Το Μέρος Β υλοποιείται με τη βοήθεια της ουράς που έχουμε δημιουργήσει στο πρώτο μέρος της εργασίας.

Αναλυτικότερα, έχουν υλοποιηθεί οι ακόλουθες βοηθητικές συναρτήσεις:

- **isDigit(char c):** Η συνάρτηση τύπου boolean δέχεται έναν χαρακτήρα, και ελέγχει αν είναι μονοψήφιος ακέραιος αριθμός, επιστρέφοντας τιμή True ή False, η οποία χρησιμεύει για τους ελέγχους εγκυρότητας παρακάτω.
- **isOperator(char c):** Η συνάρτηση αυτή, που είναι επίσης τύπου boolean, δέχεται έναν χαρακτήρα, και ελέγχει αν είναι αριθμητικός τελεστής. Για ακόμη μία φορά, επιστρέφει τιμή True ή False, που χρησιμοποιείται παρακάτω για έλεγχο εγκυρότητας.

Έπειτα μπαίνουμε στο *ψητό* της υπόθεσης, στην συνάρτηση δηλαδή που υλοποιεί την μετατροπή prefix σε infix.

Η συνάρτηση convert(String str) δέχεται ως παράμετρο ένα String, του οποίου λαμβάνουμε το μέγεθος. Έπειτα επεξεργαζόμαστε το String αυτό χαρακτήρα-χαρακτήρα, ξεκινώντας από το τέλος. Η λογική είναι πως διαβάζει τους πρώτους αριθμούς, και τους τοποθετεί στην ουρά μέχρι να συναντήσει κάποιον αριθμητικό τελεστή. Μόλις γίνει αυτό, οι αριθμοί εξάγονται από την ουρά και τοποθετούνται σε βοηθητικές μεταβλητές operand1 και operand2, και τυπώνεται σε σωστή μορφή, δηλαδή με χρήση παρενθέσεων. Σε περίπτωση λανθασμένης εισόδου του χρήστη (δηλαδή η παράσταση να μην ξεκινάει με αριθμητικό τελεστή, η να περιέχει κιάλλους χαρακτήρες εκτός από τους επιτρεπόμενους), το πρόγραμμα τερματίζει, εμφανίζοντας μήνυμα λάθους. Υπάρχουν και άλλοι έλεγχοι εγκυρότητας που αποτρέπουν την είσοδο λανθασμένου πλήθους αριθμών σε σχέση με τους αριθμητικούς τελεστές (για παράδειγμα +-*532). Αν όλοι οι έλεγχοι εκτελεσθούν με επιτυχία, το πρόγραμμα προχωράει στη ζητούμενη μετατροπή.

Γ Μέρος

Η βασική ιδέα για την υλοποίηση του Μέρους Γ, είναι πως παίρνουμε ως είσοδο απ' τον χρήστη μια ακολουθία χαρακτήρων σε τύπο String, και έπειτα την επεξεργαζόμαστε χαρακτήρα-χαρακτήρα με τη βοήθεια της ουράς που έχουμε υλοποιήσει σε προηγούμενο κομμάτι της εργασίας.

Πιο συγκεκριμένα, η είσοδος του χρήστη εισάγεται στην ουρά, και επεξεργαζόμαστε τα δεδομένα έτσι ώστε να ελέγξουμε τη συμμετρικότητα στα δύο μισά της. Μετατρέπουμε τον χαρακτήρα DNA στον αντίστοιχό του, δηλαδή A -> T, T -> A, κοκ. και έπειτα τον συγκρίνουμε με εκείνο στην αντίστοιχη θέση απ' το τέλος της λίστας. Αν ο έλεγχος τελειώσει και όλα τα στοιχεία είναι συμμετρικά μεταξύ τους, τότε η ακολουθία DNA είναι Watson-Crick complemented palindrome. Σε διαφορετική περίπτωση ισχύει το αντίθετο.

ΣΥΝΤΑΚΤΕΣ: ΧΑΡΑΛΑΜΠΙΔΗΣ ΝΑΠΟΛΕΩΝ – 3220225,
ΤΣΑΚΑΝΙΚΑΣ ΝΙΚΟΛΑΟΣ - 3220205