ΕΡΓΑΣΙΑ 1 ΕΠΕΞΗΓΗΜΑΤΙΚΟ ΚΕΙΜΕΝΟ

Ανέστης Δήμου (2170052)

ΜΕΡΟΣ Α

- **Κλάση Node**
  Αρχικά δημιουργούμε μια κλάση Node με τις κατάλληλες μεθόδους get και set όπου θα μας βοηθήσει να αναπαραστήσουμε τους κόμβους της ουράς διπλής πρόσβασης.

- **Κλάση StringDoubleEndedQueueImpl**
  Στην συνέχεια δημιουργούμε την κλάση StringDoubleEndedQueueImpl όπου υλοποιεί την διεπαφή. Στον κατασκευαστή της δημιουργούμε δύο ψευδοκόμβους αρχής και τέλους όπου και τους αρχικοποιούμε με τιμή null, ενώ παράλληλα τους συνδέομαι μεταξύ τους καθώς δεν υπάρχει κάποιο στοιχείο μέσα στην ουρά.

  - **Μέθοδος isEmpty**
    Σε αυτή την μέθοδο εξετάζουμε εάν ο επόμενος κόμβος του ψευδοκόμβου κεφαλής είναι ίδιος με τον ψευδοκόμβο τέλους. Σε αυτή την περίπτωση μας επιστρέφει true καθώς η ουρά είναι άδεια.

  - **Μέθοδοι addFirst/addLast**
    Σε αυτές τις μεθόδους προσθέτουμε κόμβους στην αρχή και το τέλος της ουράς με τις κατάλληλες συνδέσεις και αποσυνδέσεις κόμβων. Η πολυπλοκότητα των μεθόδων αυτών είναι O(1) καθώς οι εντολές που θα εκτελεστούν είναι συγκεκριμένες ανεξαρτήτως του μεγέθους της ουράς.

  - **Μέθοδοι removeFirst/removeLast**
    Σε αυτές τις μεθόδους αφαιρούμε κόμβους είτε από την αρχή είτε από το τέλος με τις κατάλληλες συνδέσεις και αποσυνδέσεις κόμβων. Και σε αυτή την περίπτωση η πολυπλοκότητα είναι O(1) καθώς οι εντολές που εκτελούνται δεν είναι σε συνάρτηση του μεγέθους της ουράς.

  - **Μέθοδοι getFirst/getLast**
    Σε αυτές τις μεθόδους επιστρέφουμε τον επόμενο κόμβο του ψευδοκόμβου κεφαλής όπου είναι ό πρώτος κόμβος της ουράς και αντίστοιχα τον προηγούμενο κόμβο του ψευδοκόμβου τέλους για τον τελευταίο κόμβο.

  - **Μέθοδος printQueue**
    Σε αυτή την μέθοδο εκτυπώνουμε όλους τους κόμβους της ουράς εκτός από τους ψευδοκόμβους.

  - **Μέθοδος size**
    Σε αυτή την μέθοδο επιστρέφουμε το μέγεθος της ουράς μέσω ενός μετρητή count όπου αυξάνεται ή μειώνεται κατά την διάρκεια προσθήκης ή αφαίρεσης στοιχείων από την ουρά. Η πολυπλοκότητα της μεθόδου αυτής είναι O(1) καθώς ο αριθμός των εντολών που εκτελούνται κάθε φορά είναι ανεξαρτήτως από το μέγεθος της ουράς.

ΜΕΡΟΣ Β

ΕΚΤΕΛΕΣΗ: Το πρόγραμμα εκτελείται κανονικά από cmd και εισάγει στοιχεία με τον scanner.

- **Κλάση PostfixToInfix (Πρόγραμμα-Πελάτη)**
  - Αρχικά εισάγουμε με τον scanner μια αριθμητική παράσταση σε μεταθεματική μορφή και την αποθηκεύουμε στην μεταβλητή input όπου είναι τύπου String
  - Δημιουργούμε μια κενή ουρά διπλή πρόσβασης με βάση την υλοποίηση που έχουμε κάνει (StringDoubleQueueImpl)
  - Αρχικοποιούμε μια μεταβλητή val τύπου Boolean όπου θα μας βοηθήσει στον έλεγχο εγκυρότητας
  - Βρίσκουμε τον πρώτο και τον τελευταίο χαρακτήρα του του String που εισήχθηκε από τον χρήστη με την μέθοδο charAt και τα αποθηκεύουμε στις μεταβλητές s και f αντίστοιχα όπου είναι τύπου char
  - Πριν προχωρήσουμε στην μετατροπή πραγματοποιούμε έλεγχο εγκυρότητας στο String που εισήχθηκε από τον χρήστη έτσι ώστε ο πρώτος χαρακτήρας του να είναι πάντα αριθμός και ο τελευταίος να είναι πάντα τελεστής
  - Αν ο έλεγχος εγκυρότητας προχωρήσει κανονικά θέτουμε την μεταβλητή var σε true και μπαίνουμε μέσα στο μπλοκ της if όπου θα πραγματοποιηθεί η μετατροπή
  - Δημιουργούμε ένα βρόχο με την χρήση του for για να προσπελάσουμε όλους τους χαρακτήρες του String που εισήχθηκε με την χρήση της μεθόδου charAt
  - Στην συνέχεια δημιουργούμε μια δομή επιλογής με τρεις περιπτώσεις:

    1η Περίπτωση: Όταν ο χαρακτήρας που θα προσπελάσουμε είναι τελεστής τότε χρησιμοποιούμε τον τελεστή για να αναπαραστήσουμε την πράξη μεταξύ του δευτέρου στοιχείου της ουράς και του πρώτου. Αυτό επιτυγχάνεται ως εξής, αρχικά χρησιμοποιούμε την μέθοδο addLast, με όρισμα την μέθοδο removeFirst όπου επιστέφει το πρώτο στοιχείο της ουράς και ταυτόχρονα το αφαιρεί, για να αποθηκεύσουμε προσωρινά το πρώτο στοιχείο της ουράς στο τέλος. Στην συνέχεια, χρησιμοποιούμε την μέθοδο addFirst με όρισμα ένα String το οποίο θα περιέχει τις παρενθέσεις της έκφρασης, τον τελεστή που βρήκαμε καθώς και τις μεθόδους removeFirst και removeLast για να λάβουμε το πρώτο και το τελευταίο στοιχείο της ουράς και ταυτόχρονα να τα αφαιρέσουμε. Με αυτόν τον τρόπο θα δημιουργήσουμε και θα προσθέσουμε στην αρχή της ουράς την ολοκληρωμένη έκφραση.

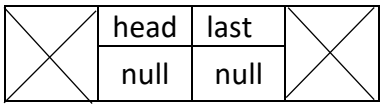
    2η Περίπτωση: Όταν ο χαρακτήρας που θα προσπελάσουμε είναι αριθμός τότε θα προσθέσουμε στην αρχή της ουράς τον συγκεκριμένο αριθμό. Αυτό θα γίνει χρησιμοποιώντας την μέθοδο addFirst με όρισμα την μέθοδο Character.toString με την οποία θα μετατρέψουμε το χαρακτήρα σε String για να μπορέσουμε να τον εισάγουμε στην ουρά.

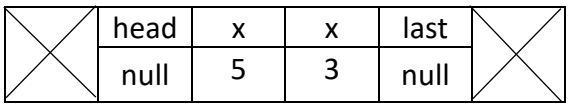
3η Περίπτωση: Όταν ο χαρακτήρας είναι οτιδήποτε άλλο τότε αυτό σημαίνει λάθος εισαγωγή, όποτε θα θέσουμε την μεταβλητή val σε false και θα τερματίσουμε τον βρόχο χρησιμοποιώντας την εντολή break.

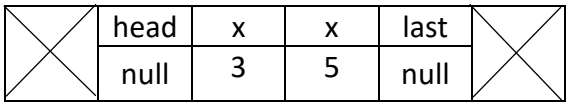
- Τέλος, μετά την ολοκλήρωση του βρόχου θα δημιουργήσουμε μια δομή ελέγχου με την χρήση του if για να ελέγξουμε να υπήρξε λανθασμένη εισαγωγή και να εμφανίσουμε κατάλληλο μήνυμα ή σε αντίθετη περίπτωση να εμφανίσουμε την αριθμητική παράσταση σε ενθεματική μορφή με την χρήση της μεθόδου getFirst καθώς η παράσταση θα βρίσκεται στην αρχή της ουράς.

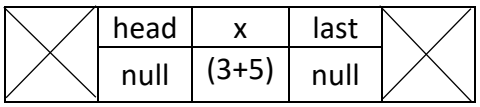
Ενδεικτικό Παράδειγμα με απεικόνιση

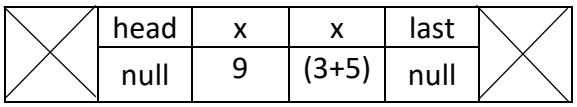
Αρχική Συνάρτηση σε μεταθεματική μορφή: 35+9*

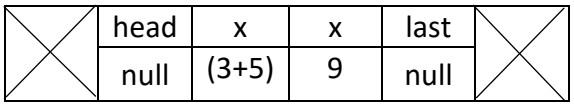


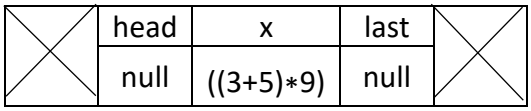












Τελική συνάρτηση σε ενθεματική μορφή: ((3+5)*9)

ΜΕΡΟΣ Γ

ΕΚΤΕΛΕΣΗ: Το πρόγραμμα εκτελείται κανονικά από cmd και εισάγει στοιχεία με τον scanner.

- **Κλάση DNAPalindrome (Πρόγραμμα-Πελάτη)**
  - Αρχικά εισάγουμε με τον scanner την ακολουθία DNA που μας δίνει ο χρήστης
  - Δημιουργούμε μια κενή ουρά διπλή πρόσβασης με βάση την υλοποίηση που έχουμε κάνει (StringDoubleQueueImpl)
  - Αρχικοποιούμε μια μεταβλητή var1 τύπου boolean όπου θα μας βοηθήσει στον έλεγχο εγκυρότητας
  - Στην συνέχεια προχωράμε στον έλεγχο εγκυρότητας, διατρέχοντας τους χαρακτήρες του String που εισήχθηκε από το τέλος προς την αρχή με την βοήθεια της μεθόδου charAt
  - Δημιουργούμε μια δομή επιλογής με δύο περιπτώσεις:

    1η Περίπτωση: Αν ο χαρακτήρας που θα βρούμε ανήκει σε μία από τις τέσσερις κατηγορίες νουκλεοτιδίων τότε τον εισάγουμε στην αρχή της ουράς διπλή πρόσβασης.
    2η Περίπτωση: Σε αντίθετη περίπτωση σημαίνει ότι ο χαρακτήρας δεν βρίσκεται στην σωστή μορφή ή ότι είναι λάθος όποτε θέτουμε την μεταβλητή var1 σε false και διακόπτουμε τον βρόχο.

  - Στην συνέχεια δημιουργούμε μια δομή επιλογής με δύο περιπτώσεις

    1η Περίπτωση: Αν το περιεχόμενο της μεταβλητής var1 είναι false τότε σημαίνει ότι έχει γίνει λάθος εισαγωγή οπότε εμφανίζουμε κατάλληλο μήνυμα και τερματίζουμε το πρόγραμμα.

    2η Περίπτωση: Σε αντίθετη περίπτωση σημαίνει ότι ο έλεγχος εγκυρότητας ολοκληρώθηκε επιτυχώς όποτε μπορούμε να προχωρήσουμε στην επεξεργασία της ουράς για να δούμε εάν η ακολουθία DNA είναι συμπληρωματικά παλίνδρομη.

Επεξεργασία Ακολουθίας DNA

- Αρχικοποιούμε μια μεταβλητή val2 τύπου Boolean όπου θα μας βοηθήσει στον έλεγχο για το εάν η ακολουθία DNA είναι συμπληρωματικά παλίνδρομο
- Αρχικοποιούμε μια μεταβλητή c τύπου String όπου θα παίζει τον ρόλο του συμπληρωματικού στο εκάστοτε νουκλεοτίδιο

- Στην συνέχεια κάνουμε ένα έλεγχο με την βοήθεια της πράξης modulo για να δούμε αν το πλήθος των νουκλεοτιδίων στην ακολουθία DNA είναι ζυγός αριθμός. Αυτό το κάνουμε καθώς σε περίπτωση μονού αριθμού είναι αδύνατον μια ακολουθία DNA να είναι συμπληρωματικά παλίνδρομη οπότε δεν πρέπει να προχωρήσουμε στην επεξεργασία της ακολουθίας.
- Για την επεξεργασία της ακολουθίας DNA δημιουργούμε μια δομή επανάληψης με την χρήση while
- Η συνθήκη αποτίμησης του βρόχου είναι το μέγεθος της ουράς να είναι μεγαλύτερο του μηδέν, δηλαδή η ουρά να περιέχει τουλάχιστον ένα στοιχείο μέσα
- Αρχικοποιούμε μια μεταβλητή n τύπου String όπου θα τοποθετούμε σε κάθε επανάληψη με την χρήση της μεθόδου removeFirst το κάθε νουκλεοτίδιο από την αρχή της ουράς, ενώ παράλληλα θα το αφαιρούμε
- Στην συνέχεια δημιουργούμε μια δομή επιλογής για την ανεύρεση του συμπληρωματικού του κάθε νουκλετιδίου και την εκχώρηση του στην μεταβλητή c
- Πριν από το τέλος κάθε επανάληψης θα ελέγχουμε αν το συμπληρωματικό του νουκλεοτίδιο στην αρχή της ουράς διαφέρει με το νουκλεοτίδιο στο τέλος της ουράς, με την χρήση της μεθόδου removeLast όπου παράλληλα θα αφαιρεί και το νουκελοτίδιο στο τέλος της ουράς.
- Αν αυτό συμβαίνει, τότε σημαίνει ότι η ακολουθία DNA δεν μπορεί να είναι συμπληρωματικά παλίνδρομη οπότε θέτουμε την μεταβλητή val2 σε false και διακόπτουμε τον βρόχο
- Σε αντίθετη περίπτωση η επανάληψη συνεχίζει κανονικά και ελέγχει κάθε φορά τα δύο ακραία νουκλεοτίδια για το εάν είναι συμπληρωματικά μέχρι να φτάσει στην μέση της ακολουθίας
- Τέλος, δημιουργούμε μια δομή επιλογής εξετάζοντας το περιεχόμενο της μεταβλητής val2:

  1η Περίπτωση: Εάν το περιεχόμενο της val2 είναι true τότε αυτό σημαίνει είτε ότι εισήχθηκε από τον χρήστη κενό String, είτε ότι ο βρόχος while ολοκληρώθηκε χωρίς προβλήματα και άρα η ακολουθία DNA είναι συμπληρωματικά παλίνδρομη, όποτε και εμφανίζουμε κατάλληλο μήνυμα.

  2η Περίπτωση: Σε αντίθετη περίπτωση σημαίνει είτε ότι ο βρόχος while διακόπηκε διότι βρήκε μη συμπληρωματικά νουκελοτίδια, είτε ότι το πλήθος νουκλεοτιδίων στην ακολουθία DNA είναι μονός αριθμός, άρα η ακολουθία DNA δεν είναι συμπληρωματικά παλίνδρομη οπότε και εμφανίζουμε κατάλληλο μήνυμα.