Η Υλοποίηση της ουράς

Για την υλοποίηση της χρησιμοποιήσαμε “Generics”, όπως και στη στοίβα, επομένως μπορεί να αποθηκεύει οποιοδήποτε τύπο δεδομένων. Στην αρχή ορίσαμε δύο object Node με όνομα head και tail. Η head δείχνει το τελευταίο στοιχείο της ουράς από το οποίο βγαίνουν και τα στοιχεία και η tail το πρώτο στοιχείο από το οποίο βγαίνουν τα στοιχεία. Επίσης ορίσαμε και την count η οποία αυξομειώνεται όταν βάζουμε και βγάζουμε στοιχεία από την ουρά. Η μέθοδος isEmpty() ελέγχει αν η ουρά είναι άδεια και επιστρέφει αντίστοιχο μήνυμα. Αυτό γίνεται εύκολα με τη συνθήκη (head == null) εφόσον όταν δεν υπάρχουν στοιχεία η head είναι null. Στη συνέχεια έχουμε την μέθοδο void put την οποία υλοποιήσαμε ως εξής. Η count αυξάνεται κατά ένα εφόσον προστίθεται στοιχείο στην ουρά. Δημιουργούμε έναν νέο κόμβο. Στη περίπτωση που η ουρά είναι άδεια και η head και tail δείχνουν στον μοναδικό κόμβο. Αλλιώς η προηγούμενη tail δείχνει στο καινούργιο στοιχείο και η ίδια η tail αλλάζει κόμβο. Την αντίθετη λειτουργία υλοποιήσαμε στην μέθοδο get(). Χρησιμοποιώντας τη μέθοδο isEmpty() ελέγχουμε εάν η ουρά είναι κενή και σε αυτή την περίπτωση επιστρέφουμε μήνυμα error NoSuchElementException στο τερματικό. Αν δεν είναι άδεια μειώνεται η count κατά ένα και αποθηκεύουμε προσωρινά τον τελευταίο κόμβο πριν διαγραφεί. Αν υπάρχει ένας μόνο κόμβος τότε η tail και η head γίνονται και οι δύο null. Αλλιώς διαγράφουμε τον τελευταίο κόμβο και αλλάζουμε την head να δείχνει στο επόμενο πιο παλιό κόμβο. Έπειτα η μέθοδος peek() χρησιμοποιείται για να δούμε τον πιο παλιό κόμβο χωρίς να τον διαγράψουμε . Σε περίπτωση που η ουρά είναι κενή επιστρέφει κατάλληλο μήνυμα error. Αλλιώς επιστρέφει τα δεδομένα της head. Η μέθοδος void printQueue εκτυπώνει ένα ένα τα στοιχεία της ουράς με τη χρήση της PrintStream και ενός προσωρινού αντικειμένου τύπου node για το iteration της ουράς. Τέλος η size() επιστρέφει την μεταβλητή count η οποία δείχνει τον αριθμό των κόμβων μέσα στην ουρά.

Η Υλοποίηση της στοίβας

Για την υλοποίηση της χρησιμοποιήσαμε “Generics” επομένως μπορεί να αποθηκεύει οποιοδήποτε τύπο δεδομένων. Αρχικά ορίσαμε ένα object Node με όνομα top . Η top δείχνει πάντα στην αρχή της στοίβας (κορυφή). Καθώς και μια count η οποία αυξομειώνεται αναλόγως με το αν προστίθεται κάποιο στοιχείο στη στοίβα ή όταν γίνεται pop. Στη συνέχεια υλοποιήσαμε τη μέθοδο isEmpty() η οποία ελέγχει άμα η στοίβα είναι κενή και επιστρέφει πίσω true η false . Αυτό γίνεται εύκολα με τη συνθήκη (top == null) διότι όταν η στοίβα δεν περιέχει κανένα στοιχείο τότε η top είναι null. Η επόμενη μέθοδος είναι η void push(E item) , η οποία προσθέτει ένα αντικείμενο στη στοίβα . Η count αυξάνεται κατά 1 . Κρατάμε το τρέχων στοιχείο της κορυφής . Δημιουργούμε νέο αντικείμενο με τα δεδομένα που δόθηκαν και βάζουμε να δείχνει στο παλιό top (previous) . Την αντίθετη λειτουργία υλοποιεί η μέθοδος pop() η οποία καταρχάς ελέγχει εάν η στοίβα είναι άδεια χρησιμοποιώντας την isEmpty() από πριν .Σε αυτή τη περίπτωση το πρόγραμμα πετάει NoSuchElementException στο τερματικό . Η count μειώνεται κατά ένα αφού αφαιρείται ένα στοιχείο. Αποθηκεύουμε το στοιχείο στη κορυφή ώστε να το επιστρέψουμε μετά και η νέα top δείχνει στο next στοιχείο της προηγούμενης top. Τέλος επιστρέφει το στοιχείο που αφαιρέθηκε. Η μέθοδος peek() επιτρέπει την πρόσβαση των δεδομένων του top node χωρίς να αφαιρεθεί από τη στοίβα. Αρχικά ελέγχει αν η στοίβα είναι άδεια και επιστρέφει NoSuchElementException αλλιώς επιστρέφει τα δεδομένα της top node. Η επόμενη μέθοδος η printStack(PrintStream stream) όταν καλείται εκτυπώνει όλα τα στοιχεία της στοίβας με τη χρήση printstream. Δημιουργούμε προσωρινό αντικείμενο για να γίνει iteration και με την while σε κάθε επανάληψη εκτυπώνεται το στοιχείο και στη συνέχεια η node δείχνει στο επόμενο στοιχείο που θα εκτυπωθεί. Η τελευταία μέθοδος size() είναι η πιο απλή και επιστρέφει το μέγεθος της στοίβας με τη χρήση της μεταβλητής count που αυξομειώνεται συνεχώς μέσα στο πρόγραμμα.

Υλοποίηση της NetBenefit

Αρχικοποιούμε την str η οποία θα χρειαστεί στο διάβασμα του αρχείου και τρεις Boolean μεταβλητές οι οποίες ειδοποιούν το πρόγραμμα όταν εντοπίζεται η λέξη buy , price και sell στο πρόγραμμα με τη χρήση της Buffered reader. Το αρχείο html δίνεται στη πρώτη θέση του πίνακα args[] από το command line. Δημιουργούμε δύο ουρές τύπου int , η μια για την αξία στην οποία αγόρασε ο πελάτης τις μετοχές και η άλλη για τη ποσότητα που αγόρασε . Δημιουργούμε αντικείμενο buffered reader για την ανάγνωση των γραμμών του αρχείου. Η while loop που υλοποιήσαμε τρέχει μέχρι να βρει γραμμή στην οποία δεν υπάρχουν χαρακτήρες δηλαδή όσο η επόμενη Line είναι != του null. Αρχικοποιούμε τις μεταβλητές για τις μετοχές τη διαφορά σε σχέση με την πιο παλιά στη λίστα καθώς και το τελικό αποτέλεσμα (κέρδος). Οι flags γίνονται false όπως και στην αρχή κάθε επανάληψης. Η for loop κάνει iterate τις λέξεις της γραμμης εφόσον έχουμε κάνει split στα κενά και τα αποθηκεύσαμε στο πίνακα line. Αναλόγως με τη λέξη αλλάζουν οι flag για το αν θα γίνει αγορά η πούλημα. Αν FlagBuy είναι true προστίθεται στην ουρά ο ακέραιος (ο οποίος μετατράπηκε σε Int από string με τη χρήση της valueof() (αριθμός της ποσότητας που αγόρασε). Το b γίνεται θετικό οπότε θα προστεθεί και η αξία της αγοράς στην άλλη ουρά. Στην περίπτωση της πώλησης πουλάμε συνεχώς μέχρι να μας μείνουν 0 που πρέπει να πουλήσουμε. Όσο δηλαδή shares > 0. Αν είναι περισσότερες από τον πιο παλιό κόμβο (αριθμός πιο παλιών μετοχών που αγόρασε) της ουράς τότε θα αφαιρεθούν από αυτές που πρέπει να πουλήσουμε αυτές που ήταν στην ουρά. Ο κόμβος αυτός διαγράφεται και μειώνεται ο αριθμός των μετοχών που μένουν να πουληθούν . Αλλιώς αν είναι λιγότερες αφαίρει όσες πρέπει να πουλήσει και αντικαταστεί στον παλιό κόμβο τη διαφορά του αριθμού των μετοχών που πρέπει να πουληθούν και αυτών που ήταν στην ουρά. Και στις δυο περιπτώσεις η result αυξάνεται σύμφωνα με την παλιά και την τωρινή αξία αυτών των μετοχών. Και οι δύο ουρές ενημερώνονται παράλληλα. Όταν βγει από αυτό το while loop εκτυπώνει τη μεταβλητή result δηλαδή το κέρδος από την πώληση αυτή. Στη συνέχεια η τελειώνει η συνεχίζει τις αγορές και τις πωλήσεις που μένουν

Υλοποίηση της TagMatching

Αρχικά, ορίζουμε ένα αντικείμενο StackHtml τύπου StringStackImpl. Αρχικοποιούμε μία μεταβλητή τύπου String με την τιμή null η οποία θα λαμβάνει την τιμή του τελευταίου στοιχείου της στοίβας. Χρησιμοποιούμε, την StringBuilder η οποία στην Java αντιπροσωπεύει μια μεταβλητή ακολουθία χαρακτήρων. Έπειτα διαβάζουμε το html αρχείο και κάθε γραμμή του προστίθεται στη μεταβλητή τύπου StringBuilder.
Χρησιμοποιούμε τρεις μεταβλητές τύπου Boolean οι οποίες ελέγχουν τα tag. Η μεταβλητή flag1 λαμβάνει την τιμή αληθής όταν βρίσκεται ένα tag ανοίγματος. Αντίστοιχα η μεταβλητή flag2 λαμβάνει την τιμή ψευδής όταν βρίσκεται ένα tag κλεισίματος. Χρησιμοποιούμε ακόμη δύο strings το s1 και το s2. Λαμβάνουμε έναν έναν τους χαρακτήρες που περιλαμβάνονται στη μεταβλητή content με τη χρήση της μεθόδου charAt έως ότου να φτάσουμε στο τέλος του for loop το οποίο σταματάει στο προτελευταίο στοιχείο της μεταβλητής content. Όταν βρούμε tag ανοίγματος τότε στην επόμενη επανάληψη όταν η flag1 γίνει αληθής θα προσθέσουμε τον πρώτο χαρακτήρα που βρίσκεται μετά το '<' στο string s1. Αυτή η διαδικασία θα συνεχιστεί μέχρι να βρεθεί ο χαρακτήρας κλεισίματος του tag '>' ,έπειτα θα κάνουμε push το s1 στην στοίβα και μετά θα το ξαναρχικοποιήσουμε με το "". Παρόμοια διαδικασία για το s2 μόνο που χρησιμοποιούμε και έναν counter οποίος επιτρέπει την προσθήκη χαρακτήρα στην μεταβλητή s2 μόνο όταν λάβει τιμή μεγαλύτερη του δύο. Αυτό συμβαίνει διότι, τα tag κλεισίματος πριν τον πρώτο χαρακτήρα έχουν ένα παραπάνω σύμβολο, το / και δεν θέλουμε να το προσθέσουμε στο string. Όταν βρούμε χαρακτήρα κλεισίματος (>), τότε ο counter και flag2 θα γίνουν αντίστοιχα 0 και ψευδής. Αν η στοίβα είναι άδεια σημαίνει πως τα tag είναι λάθος καθώς τα προηγούμενα έχουν κλείσει ήδη ή δεν υπήρχαν και προσπαθούμε να κλείσουμε ένα tag το οποίο δεν έχει καν ανοίξει. Σε αυτή την περίπτωση η flag γίνεται ψευδής. Εάν τώρα η στοίβα δεν είναι άδεια τότε θα χρησιμοποίησουμε την μέθοδο peek από την κλάση StringStackImpl η οποία υλοποιεί στοίβες και θα λάβουμε τα δεδομένα του τελευταίου στοιχείου της στοίβας. Αν ισούται με το s2 τότε κάνουμε pop το στοιχείο από την στοίβα αρχικοποιούμε πάλι το s2 με "" και συνεχίζουμε την επανάληψη. Εάν δεν είναι ίσα τότε η flag γίνεται ψευδής. Όταν τελειώσει η επαναληπτική διαδικασία τότε αν η flag είναι αληθής και η στοίβα άδεια (γιατί εάν περισσέψει στοιχείο τότε δεν υπήρχε κάποιο tag κλεισίματος ή τα tag ήταν λάθος) τότε τα tag κλείνουν σωστά και τυπώνεται η τιμή true. Σε οποιαδήποτε άλλη περίπτωση τα tag κλείνουν λάθος και τυπώνεται η τιμή ψευδής.