Εξηγήστε συνοπτικά πώς υλοποιήσατε τις διεπαφές στο μέρος Α .

- ☞ Η StringStack και η IntQueue έχουν υλοποιηθεί με χρήση generics.

Αρχικά η IntQueue είναι Fifo, δηλαδή το πρώτο στοιχείο που μπαίνει σε αυτή πρέπει να είναι και το πρώτο που θα βγει. Για αυτό το λόγο η συνάρτηση put εισάγει τα δεδομένα από πίσω (δηλαδή από την ουρά) και οι συναρτήσεις peek/get δίνουν τα στοιχεία από μπροστά (δηλαδή από το head) . Στην StringStack που είναι lifo τα στοιχεία εισάγονται και αφαιρούνται από μπροστά ,αφού το στοιχείο που μπαίνει τελευταίο πρέπει να αφαιρεθεί και πρώτο.

Σχετικά με την IntQueueImpl ας δούμε αναλυτικά μία-μία συνάρτηση:

- ✓ isEmpty() ελέγχει αν το head είναι null
  - αν είναι null τότε επιστρέφεται true καθώς αυτό μας δείχνει πως δεν υπάρχει κάποιο στοιχείο.
- ✓ put() τοποθετεί ένα στοιχείο από πίσω.
  - Αν είναι άδεια η λίστα τότε το στοιχείο αυτό θα είναι και το head και το tail
  - Αν δεν είναι άδεια τότε το επόμενο στοιχείο του tail δείχνει στο στοιχείο που θέλαμε να προσθέσει
  - Βλέπουμε ότι αφού τοποθετήσαμε ένα στοιχείο στην λίστα αυξήθηκε το size κατά ένα (αυτό θα μας χρειαστεί για την υλοποίηση της size() και θα αναλυθεί στην συνέχεια).
  - Βλέπουμε ότι το στοιχείο που αναμένει να λάβει δεν είναι int αλλά T για να μπορέσει να υλοποιηθεί με generics αντίστοιχες αλλαγές έγιναν και στα impl και στα interface αρχεία και δεν θα αναφερθούμε εκ νέου σε αυτές τις λεπτομέρειες .
  - Η συνάρτηση put ολοκληρώνεται σε χρόνο O(1) όπως βλέπουμε καθώς το να προστεθεί ένα καινούργιο στοιχείο στην λίστα μας δεν επηρεάζεται από τον αριθμό των στοιχείων που ήδη βρίσκονται σε αυτήν. Αφού αυτό που κάνει είναι απλά να δημιουργεί μια καινούργια θέση στο τέλος για να μπει το καινούργιο μας στοιχείο και το tail να δείχνει σε αυτό.
- ✓ get() επιστρέφει και διαγράφει το αρχαιότερο στοιχείο της λίστας
  - Αρχικά και εδώ γίνεται έλεγχος για το αν είναι άδεια η λίστα.
  - Σε περίπτωση που είναι άδεια επιστρέφεται NoSuchElementException
  - Αν η λίστα δεν είναι άδεια τότε αποθηκεύουμε σε μια μεταβλητή την τιμή που έχει το head
  - Και αν αυτό ήταν το μοναδικό στοιχείο της λίστας το head και το tail γίνονται null (δηλαδή η λίστα αδειάζει)
  - Αν υπήρχαν και άλλα στοιχεία τότε το head απλά δείχνει στο επόμενο στοιχείο.
  - Και εδώ βλέπουμε ότι το size μειώνεται και αυτό συμβαίνει και πάλι για να βοηθήσουμε την size() να ολοκληρώνεται σε O(1)
  - Και τέλος επιστρέφεται το στοιχείο που είχαμε αποθηκεύσει σε μία μεταβλητή στην αρχή.
  - Η συνάρτηση ολοκληρώνεται σε χρόνο O(1) όπως βλέπουμε καθώς το να επιστραφεί το παλαιότερο στοιχείο της λίστας και να διαγραφεί δεν επηρεάζεται από τον αριθμό των στοιχείων που βρίσκονται στη λίστα. Αυτό που κάνει είναι να διαγράφει την πάνω θέση και το head να δείχνει στο επόμενο στοιχείο.
- ✓ peek() επιστρέφει το αρχαιότερο στοιχείο της λίστας
  - Όπως και στην get αν η λίστα είναι άδεια βγαίνει NoSuchElementException
  - Εάν δεν είναι άδεια τότε επιστρέφεται το στοιχείο που βρίσκεται στο head.
  - Και αυτή η συνάρτηση ολοκληρώνεται σε O(1) αφού δεν επηρεάζεται από τον αριθμό των στοιχείων που βρίσκονται στη λίστα. Επιστρέφει το στοιχείο που βρίσκεται πάνω πάνω χωρίς την χρήση κάποιου loop.
- ✓ printQueue() εμφανίζει τα στοιχεία της λίστας
  - Γίνεται έλεγχος αν η λίστα είναι άδεια με την χρήση της isEmpty() όπως και στις προηγούμενες συναρτήσεις.
  - Εάν είναι εμφανίζεται ενημερωτικό μήνυμα.

- Εάν δεν είναι άδεια τότε διατρέχουμε με while τα στοιχεία της λίστας κάνοντας το head να δείχνει κάθε φορά στο επόμενο στοιχείο μέχρι να μην υπάρχει άλλο.

✓ Size() επιστρέφει το μέγεθος της λίστας
- Όπως διακρίναμε παραπάνω υπήρχε μια μεταβλητή size η οποία αυξανόταν κατά ένα κάθε φορά που προσθέταμε κάποιο στοιχείο στην λίστα και μειωνόταν κατά ένα όταν διαγράφαμε κάποιο.
- Επιπλέον η μεταβλητή size έχει αρχικοποιηθεί με 0 για να είναι εφικτό να γίνουν οι παραπάνω πράξεις.
- Η size() ολοκληρώνεται σε χρόνο O(1) αφού δεν διατρέχει τα στοιχεία της λίστας για να τα μετρήσει. Επιστρέφει απλά την τιμή μιας μεταβλητής. Οπότε σε ανεξάρτητο χρόνο από του πλήθος των στοιχείων που περιέχονται στην λίστα η size() θα εκτελεστεί.

Σχετικά με την StringQueueImpl ας δούμε αναλυτικά μία-μία συνάρτηση:

✓ isEmpty() λειτουργεί και εδώ με τον ίδιο τρόπο οπώς και στην intQueueImpl

✓ push() τοποθετεί ένα στοιχείο από μπροστά.
- Αν είναι άδεια η λίστα τότε το στοιχείο αυτό θα είναι και το head και το tail
- Αν δεν είναι άδεια τότε το head είναι το στοιχείο που θέλαμε να προσθέσουμε.
- Βλέπουμε ότι αφού τοποθετήσαμε ένα στοιχείο στην λίστα αυξήθηκε το size κατά ένα (παρόμοια θα γίνει και στην pop με μείωση του size για να γίνει η υλοποίησει της size όπως και στο intQueue.(Για συντομία του λόγου δεν θα αναφερθούμε ξανά).
- Η συνάρτηση push ολοκληρώνεται σε χρόνο O(1) όπως βλέπουμε καθώς το να προστεθεί ένα καινούργιο στοιχείο στην λίστα μας δεν επηρεάζεται από τον αριθμό των στοιχείων που ήδη βρίσκονται σε αυτήν. Αφού αυτό που κάνει είναι απλά να δημιουργεί μια καινούργια θέση στην αρχή για να μπει το καινούργιο μας στοιχείο και το head να δείχνει σε αυτό.

✓ pop()
- λειτουργεί ακριβώς με τον ίδιο τρόπο όπως την get στην προηγούμενη σελίδα και ισχύουν και τα ίδια για το O(1) με μοναδική διαφορά πως το στοιχείο που δείχνει το head δεν είναι το αρχαιότερο αλλά το νεότερο αφού έχουμε λίστα lifo

✓ peek()
- λειτουργεί ακριβώς με τον ίδιο τρόπο όπως την peek στην προηγούμενη σελίδα και ισχύουν και τα ίδια για το O(1) με μοναδική διαφορά και εδώ πως το στοιχείο που δείχνει το head δεν είναι το αρχαιότερο αλλά το νεότερο αφού έχουμε λίστα lifo όπως προείπαμε.

✓ printStack
- λειτουργεί ακριβώς με τον ίδιο τρόπο όπως η printQueue και ισχύουν ακριβώς τα ίδια με αυτά που είπαμε παραπάνω.
- Το γεγονός ότι η μια λίστα τα εμφανιζει από το αρχαιότερο στο νεότερο και η άλλη(δηλαδή αυτή) από το νεότερο στο αρχαιότερο με ακριβώς τον ίδιο κώδικα επιτυγχάνεται από την διαφορετική δομή που έχουμε δώσει στις λίστες μας.

✓ Size()
- Αντίστοιχα και για την size ισχύουν ακριβώς τα ίδια που αναφέραμε για την size της IntQueue

b. Για το μέρος Β, εξηγήστε πώς χρησιμοποιήσατε την υλοποίηση από το μέρος Α για να φτιάξετε το πρόγραμμα που ζητείται (TagMatching).

- Αρχικά με την χρήση του BufferedReader διαβάζουμε το περιεχόμενο του αρχείου που μας δόθηκε μέσω του cmd και το λάβαμε με την χρήση του args[0].
- Περνάμε όλο το περιεχόμενο του αρχείου μας σε μία μεταβλητή string.
- Μόλις ολοκληρωθεί η διαδικασία με το διάβασμα του αρχείου τότε κλείνουμε το αρχείο.
- Έπειτα με χρήση μιας επαναληπτικής δομής for ελέγχουμε το περιεχόμενο που περάσαμε στην string μεταβλητή νωρίτερα.
- Εάν βρούμε τον χαρακτήρα < τότε μέσα σε μία μεταβλητή τον καταχωρούμε και χαρακτήρα χαρακτήρα συνεχίζουμε μέχρι και το σύμβολο > μόλις το βρούμε τότε βάζουμε το tag μέσα στην λίστα stack που έχουμε φτιάξει(από την υλοποίηση του α μέρους είναι η λίστα η StringStack). Αυτή η διαδικασία επαναλαμβάνεται μέχρι να διαβαστεί και ο τελευταίος χαρακτήρας του content (του περιεχομένου του αρχείου)
- Σε αυτό το σημείο όλα τα tags που έχουν χρησιμοποιηθεί βρίσκονται στην λίστα stack.
- Πέρα από την stack έχουμε ακόμη μία λίστα τύπου StringStack με όνομα tempstack και αυτό που κάνουμε είναι να ανταλλάζουμε τα στοιχεία μεταξύ τους
- Αρχικά αν η λίστα stack έχει κάποιο στοιχείο τότε με pop παίρνουμε το πάνω πάνω στοιχείο της και με κατάλληλες αλλαγές το παιρνάμε σε μία άλλη μεταβλητή tampa ως το αντίστροφο στοιχείο από αυτό που είναι . Για παράδειγμα αν το στοιχείο που πήραμε είναι το <h> τότε η μεταβλητή tampa είναι </h>
- Αν η stack δεν έχει κανένο στοιχείο τότε θα πάρουμε ξανά με pop στοιχείο από την tempstack
- Έπειτα «διατρέχουμε στην stack να δούμε αν υπάρχει το στοιχείο tempa μέσα αν υπάρχει σταματάμε να ψάχνουμε και μεταβαίνουμε στο επόμενο στοιχείο με βάση τα δύο προηγούμενα βήματα που αναλύσαμε. Αν το στοιχείο δεν βρεθεί στο stack τότε θα ψάξουμε και στο tempstack.
- Όταν λέμε ότι της διατρέχουμε εννοούμε να βλέπουμε με peek αν το στοιχείο μας ενδιαφέρει αν μας ενδιαφέρει το αφαιρούμε με pop αν όχι τότε το αφαιρούμε με pop και με push το βάζουμε στην άλλη λίστα από αυτή που βρίσκεται και συνεχίζουμε το ψάξιμο.
- Τα βήματα αυτά από την επιλογή και την αλλαγή του στοιχείου σε αυτό που ψάχνουμε μέχρι και να το βρούμε επαναλαμβάνεται όσο κάποια από τις δύο αυτές λίστες έχει μέσα στοιχεία.
- Κάθε φορά που βρίσκουμε ταύτιση αυξάνουμε τον μετρητή total κατά ένα και έχουμε αποθηκεύσει στην μεταβλητή total2 το μέγεθος της αρχικής stack.
- Μόλις ολοκληρωθούν οι while και φτάσουμε σε σημείο που και οι δύο λίστες είναι άδειες τότε βλέπουμε αν ο αριθμός των αλλαγών που κάναμε ισούται με το μισό μέγεθος της λίστας που διατρέξαμε. Αν αυτό δεν συμβαίνει εμφανίζεται μήνυμα ότι τα tags δεν είναι άρτια τοποθετημένα και το πρόγραμμα τερματίζει.
- Αν το πρόγραμμα δεν τερματίσει δηλαδή τα tags είναι σωστά τοποθετημένα τότε βγαίνει μήνυμα που λέει πως τα tags είναι εντάξει.

c. Ομοίως για το μέρος Γ (NetBenefit).

Αρχικά διαβάζουμε το αρχείο γραμμή- γραμμή.

Γίνεται έλεγχος αν το αρχείο ξεκινάει με την εντολή buy. Αν δεν ξεκινάει με αυτή την εντολή εμφανίζεται μήνυμα στον χρήστη και το πρόγραμμα τερματίζει. Το αρχείο διαβάζεται χαρακτήρα-χαρακτήρα μέχρι να βρεθεί κενό και το price το παίρνουμε ως το υπολειπόμενο της γραμμής. Αν οι εντολές δεν είναι buy/sell τότε εμφανίζεται και σε αυτή την περίπτωση μήνυμα στον χρήστη που τον ενημερώνει κατάλληλα και τερματίζει το πρόγραμμα όπως και αν μετά το νούμερο που αντιπροσωπεύει την ποσότητα δεν ακολουθεί η λέξη price.Η τιμή ελέγχεται για να είναι ακέραια και θετική όπως και η ποσότητα που έχει αγοραστεί/θέλει να πουληθεί. Σε περίπτωση απόκλισης εμφανίζεται μήνυμα και το πρόγραμμα τερματίζει.

Επιπλέον κάθε φορά που διαβάζουμε την εντολή sell δίνουμε «μήνυμα» στο πρόγραμμα ότι μπορεί να συνεχίσει ακόμη και αν αυτή είναι η τελευταία γραμμή. Κάνουμε δηλαδή τη λογική μεταβλητή end true και όταν διαβάζεται το buy ξαναγίνεται false. Με αυτό τον τρόπο το πρόγραμμά μας θα εμφανίσει μήνυμα στον χρήστη και θα τερματίσει αν δεν δοθεί σαν εντολή το sell στην τελευταία γραμμή του αρχείου. Σε κάθε γραμμή στο αρχείο παίρναμε την ποσότητα και έπειτα την τιμή(σε διαφορετική « γραμμή») . Όταν όμως έχουμε εντολή sell έχουμε βάλει σαν «διαχωριστικό» έναν μεγάλο αρνητικό αριθμό πριν την ποσότητα του sell και έπειτα από την price του sell. Σε αυτό το σημείο το queue μας είναι έτοιμο με τελικό στοιχείο την μεταβλητή β που μας δείχνει το τέλος της αρχικής ουράς.

Αυτή η διαδικασία ολοκληρώνεται στις πρώτες 125 γραμμές.

Τώρα προσπαθούμε να κάνουμε τις αντίστοιχες πράξεις για να εμφανίσουμε στον χρήστη το συνολικό κέρδος που έχει.

Όσο η λίστα μας δεν είναι άδεια εκτελούμαι loop όπου ελέγχει το παλαιότερο στοιχείο.

- ☞ Αν αυτό δεν είναι το ”αναγνωριστικό” α (δεν ακολουθεί πώληση δηλαδή) και δεν είναι αναγνωριστικό ότι ακολουθεί κάποιο άθροισμα κέρδους τότε διαγράφει και ξαναεισάγει στην λίστα από τα παλαιότερα στοιχεία μέχρι να βρει εντολή sell.(την μεταβλητή α δηλαδή) το while εκεί στην σειρά 151 έχει και μετρητή για να είμαστε σίγουροι πως δεν πέφτει σε ατέρμων βρόγχο σε περίπτωση που μας έχουν μείνει μετοχές απούλητες. Αν δει πως δεν υπάρχουν άλλα a στο query μας τότε κανει την μεταβλητή k true και σταματάει το while και υπολογίζεται το άθροισμα του κέρδους , αθροίζοντας τα στοιχεία της λίστας μας που βρίσκονται μετά από το αναγνωριστικό kl και εμφανίζονται στον χρήστη μας , έξω από την while)
- ☞ Αν το στοιχείο που διαβάζουμε είναι το a τότε έχουμε βρει εντολή sell. Τ επόμενα δύο στοιχεία είναι η ποσότητα που πουλήθηκε και η τιμή στην οποία πουλήθηκε και ακολουθεί ξανά η μεταβλητή a. Για να μας διαχωρίσει κάπως ότι τα στοιχεία της πώλησης τελείωσαν. Έπειτα διαγράφουμε και ξαναεισάγουμε τα υπόλοιπα δεδομένα. Να τονίσουμε εδώ πως τα στοιχεία της πώλησης διαγράφηκαν από την ουρά και κρατήθηκαν σε μεταβλητές totalamount/ sellprice.

Δημιουργούμαι ένα εσωτερικό while όπου τρέχει όσο το totalamount δεν έχει ικανοποιηθεί και μέχρι να βρει ένα από τα νούμερα μας που έχουμε θέσει για να διαχωρίζουμε πωλήσεις /κέρδη /αρχικό τέλος ουράς(b) .Εκεί παίρνουμε ένα ένα τα στοιχεία της ουράς αν είναι ίδια με το totalamount κάνουμε τις αντίστοιχες πράξεις που απαιτούνται μειώνουμε το totalamount προσθέτουμε στην λίστα μας την μεταβλητή kl (μετά από αυτή ακολουθεί κέρδος) και έπειτα προσθέτουμε το κέρδος μας. Και αφαιρούμε και το a (που είχε μείνει για να μας οριοθετεί που να ψάξουμε) .Κάθε στοιχείο που πωλείται εχει διαγραφτεί ήδη από την ουρά και δεν ξανατοποθετείται σε αυτήν.

Αν το amount που περιέχεται σαν παλαιότερο στοιχείο στην ουρά είναι μεγαλύτερο από το total διαγράφουμε το a / καθώς και τα στοιχεία του amount/price του στοιχείου αυτού και το ξαναεισάγουμε με ανανεωμένη την ποσότητα που έμεινε και έπειτα το kl και το κέρδος. Το totalamount σε αυτή την περίπτωση γίνεται αρνητικό δηλώνοντας πως ολοκληρώθηκε.

Σε περίπτωση που το total είναι μεγαλύτερο από το amount (δηλαδή του παλαιότερου στοιχείου της ουράς μας) τότε εισάγει το kl και έπειτα το κέρδος και τρέχει μέσα στη while μέχρι να σταματήσει αυτή διαπερνώντας δηλαδή και τα υπόλοιπα στοιχεία της ουράς μέχρι να βρει κάποιο φράγμα(ειδικο νουμερο που θεσαμε/ ολοκληρωθει το total amount)

Αν τελειώσει η while αλλά η ποσότητα που θέλαμε να πουλήσουμε δεν ικανοποιήθηκε ενημερώνεται ο χρήστης και το πρόγραμμα τερματίζει.

- ☞ Αν το στοιχείο που διαβάζουμε είναι το kl (που μετά από αυτό ακολουθεί πώληση) διαγράφουμε και ξαναεισάγουμε τα στοιχεία του εκ νέου στην ουρά. Δηλαδή τις 2 πρώτες γραμμές.

Αν καταφέραμε να βγούμε έξω από την μεγάλη while χωρίς να γίνει έξοδος του προγράμματος τότε όπως αναφέραμε και παραπάνω γίνεται υπολογισμός των συνολικών κερδών.Διατρέχουμε την λίστα (διαγράφοντας και ξαναεισάγοντας τα στοιχεία της εκ νέου) και αν βρούμε το kl προσθέτουμε στο sum το επόμενο στοιχείο που το ακολουθεί. Φυσικά διαγράφονται και αυτά καθώς διαγράφονται και τα στοιχεία που δεν είναι kl όπως το b ή οι μετοχές που δεν πουλήθηκαν.

- ! Το πρόγραμμα έχει κάποιο λάθος στην υλοποίηση που δεν μπόρεσα να το βρω. Το ερρορ βρίσκεται στην περίπτωση που βλέπουμε το «αναγνωριστικό» της πώλησης και κάνουμε τις πράξεις των κερδών. Αν παρατηρήσετε κάτω από το νούμερο που αντιπροσωπεύει τις πωλήσεις το 300000.. από κάτω ακριβώς τα κέρδη κάθε πώλησης μιας αγοράς εμφανίζονται σωστά. Απλά για κάποιο λόγο θεωρεί ότι η ποσότητα που θέλαμε δεν έχει καλυφθεί από τους διαθέσιμους πόρους και εμφανίζει στον χρήστη πως δεν μπορεί να πουλήσει αυτή τη ποσότητα.

Παραθέστε οποιαδήποτε πληροφορία κρίνετε απαραίτητη για να γνωρίζουν οι βοηθοί του μαθήματος πώς εκτελείται ο κώδικάς σας (π.χ. επιβεβαιώστε ότι χρησιμοποιήσατε τις οδηγίες για τα command line arguments, για να δίνετε σαν είσοδο το όνομα του αρχείου στο Μέρος Β)

- ☞ Ο κώδικας έχει γραφτεί σε java έκδοση 8.
- ☞ Κατά την δοκιμή τρέχει χωρίς κάποιο πρόβλημα μεταγαπό το cmd χρησιμοποιώντας τις εντολές
- ☑ java TagMatching path_to_html_file.html
- ☑ java NetBenefit path_to_text_file.txt

(όπου path_to_html_file δοκίμασα το όνομα του αρχείου που βρισκόταν στον ίδιο φάκελο)