ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Δομές Δεδομένων - Εργασία 1

Τμήμα Πληροφορικής

Φθινοπωρινό Εξάμηνο 2021-2022

Διδάσκων: Ε. Μαρκάκης

Προθεσμία παράδοσης: Δευτέρα, 29/11/2021, 23:59

Ουρές: Υλοποιήσεις ΑΤΔ και εφαρμογές

Σκοπός της εργασίας είναι η εξοικείωση με βασικούς αφηρημένους τύπους δεδομένων όπως οι στοίβες και οι ουρές FIFO. Η εργασία αποτελείται από 2 υλοποιήσεις ΑΤΔ (Μέρος Α) και 2 εφαρμογές (Μέρος Β και Γ). Διαβάστε προσεκτικά την εκφώνηση και τα ζητούμενα της εργασίας.

Μέρος Α [30 μονάδες]. Στον φάκελο «Εγγραφα/Εργασίες/Εργασία 1» του eclass, δίνονται οι διεπαφές StringStack και IntQueue, που δηλώνουν τις βασικές μεθόδους για μια στοίβα και μια ουρά FIFO, με στοιχεία τύπου *String* και *int* αντίστοιχα. Δημιουργήστε μία υλοποίηση των ΑΤΔ StringStack και IntQueue, δηλαδή γράψτε 2 κλάσεις που υλοποιούν τα 2 interfaces.

Οδηγίες υλοποίησης:

- Οι κλάσεις σας **πρέπει να λέγονται** StringStackImpl και IntQueueImpl.
- Η υλοποίηση και για τις 2 διεπαφές θα πρέπει να γίνει χρησιμοποιώντας λίστα μονής σύνδεσης (η στοίβα και η ουρά δηλαδή πρέπει να αποθηκεύουν το περιεχόμενό τους σε λίστα μονής σύνδεσης).
- Κάθε μέθοδος εισαγωγής ή εξαγωγής στοιχείου θα πρέπει να ολοκληρώνεται σε χρόνο O(1), δηλαδή σε χρόνο ανεξάρτητο από τον αριθμό των αντικειμένων που μπορεί να είναι μέσα στην ουρά. Ομοίως, η μέθοδος size θα πρέπει να εκτελείται σε O(1).
- Όταν η στοίβα ή η ουρά είναι άδεια, οι μέθοδοι που διαβάζουν από την δομή θα πρέπει να πετάνε εξαίρεση τύπου NoSuchElementException. Η εξαίρεση NoSuchElementException ανήκει στην core βιβλιοθήκη της Java. Κάντε την import από το πακέτο **java.util**. Μην κατασκευάσετε δική σας εξαίρεση.
- Μπορείτε να χρησιμοποιήσετε μέρος του κώδικα που έχει παρουσιαστεί στα εργαστήρια του μαθήματος (διαθέσιμος στο eclass) ή να γράψετε εξ' ολοκλήρου τη δική σας λίστα ή να ορίσετε μόνο αντικείμενα τύπου Node μέσα στην κλάση της ουράς, για τους κόμβους της λίστας, χωρίς να ορίσετε κλάση λίστας. Για να αποκτήσετε καλύτερη εξοικείωση

προτείνουμε να ξεκινήσετε από την αρχή και να γράψετε τις δικές σας κλάσεις (σίγουρα δεν θα χάσετε μονάδες όμως χρησιμοποιώντας ό,τι έχετε δει στο εργαστήριο).

- Δεν απαιτείται να υπάρχει μέθοδος main στα αρχεία που θα μας παραδώσετε για το Μέρος Α. Όμως, καλό είναι να φτιάξετε μια main όπου μπορείτε να τρέξετε μερικά παραδείγματα για να βεβαιωθείτε για την ορθότητα των μεθόδων σας (δεν θα αποτελεί μέρος της βαθμολόγησης η main σας, καθώς εμείς θα χρησιμοποιήσουμε δικό μας πρόγραμμα για να αξιολογήσουμε τις υλοποιήσεις των interfaces).
- **Προαιρετικά:** μπορείτε να κάνετε την υλοποίηση σας με χρήση generics για να μπορείτε να χειρίζεστε ουρές και στοίβες με οποιονδήποτε τύπο αντικειμένων. Υπάρχει ένα 10% bonus σε όσους χρησιμοποιήσουν generics για το Μέρος Α. Αν κάνετε χρήση generics, πρέπει να τροποποιήσετε κατάλληλα τα interfaces που σας δίνονται για να δουλεύουν για οποιονδήποτε τύπο δεδομένων.
- **Δεν επιτρέπεται να χρησιμοποιήσετε έτοιμες υλοποιήσεις δομών τύπου λίστας, στοίβας, και ουράς, από την βιβλιοθήκη της Java** (π.χ. Vector, ArrayList κλπ).

Μέρος Β [30 μονάδες]. Χρησιμοποιώντας την υλοποίηση της στοίβας του μέρους Α, γράψτε ένα πρόγραμμα-πελάτη το οποίο θα κάνει έλεγχο για το ταίριασμα των ετικετών (tags) σε ένα HTML αρχείο. Σε κάθε HTML έγγραφο, οι ετικέτες οριοθετούν τμήματα κειμένου. Μια ετικέτα ανοίγματος έχει τη μορφή <tag_name> ενώ η αντίστοιχη ετικέτα κλεισίματος είναι η </tag_name>. Κάποιες απλές και συνηθισμένες ετικέτες είναι οι εξής:

- body, για να ξεκινήσει το κυρίως σώμα του εγγράφου
- h1, για την επικεφαλίδα
- center, για στοίχιση στο κέντρο
- ol, για αριθμημένη λίστα στοιχείων
- li, για ένα στοιχείο μιας λίστας

Ιδανικά, κάθε HTML έγγραφο έχει *ταιριασμένες* ετικέτες, αν δηλαδή υπάρχει κάπου η ετικέτα , θα πρέπει αργότερα στο αρχείο να εμφανίζεται η ετικέτα , που σηματοδοτεί το τέλος της στοίχισης στο κέντρο. Το ίδιο πρέπει να συμβαίνει για όλες τις ετικέτες. Επίσης, αν υπάρχουν φωλιασμένες ετικέτες μέσα σε ένα αρχείο, θα πρέπει να κλείνουν με τη σωστή σειρά. Αν έχουμε φωλιασμένες ετικέτες ανοίγματος, θα πρέπει να κλείσει πρώτα η τελευταία ετικέτα που άνοιξε προτού κλείσουν οι υπόλοιπες. Αν και αρκετοί browsers παρουσιάζουν ανοχή στην ύπαρξη κάποιων μη ταιριασμένων ετικετών, θα θέλαμε να μην έχουμε τέτοιες ετικέτες σε αρχεία που επεξεργαζόμαστε.

Το πρόγραμμα-πελάτη που θα γράψετε θα πρέπει να διαβάζει γραμμή προς γραμμή ένα HTML αρχείο, το όνομα του οποίου θα δίνεται από τον χρήστη, και να αποφασίζει στο τέλος αν το αρχείο έχει ταιριασμένες ετικέτες ή όχι, τυπώνοντας αντίστοιχο μήνυμα.

Οδηγίες υλοποίησης:

- Το πρόγραμμα σας **πρέπει να λέγεται** TagMatching.java.
- Θα πρέπει να χρησιμοποιήσετε την υλοποίηση της στοίβας από το Μέρος Α.
- Για το διάβασμα του html αρχείου, μπορείτε να χρησιμοποιήσετε έτοιμες κλάσεις της Java, όπως οι Scanner, FileReader, BufferedReader, κλπ.. Μπορείτε επίσης να χρησιμοποιήσετε έτοιμες μεθόδους για να επεξεργαστείτε μεταβλητές τύπου String. Πέρα από το διάβασμα όμως του αρχείου, το υπόλοιπο πρόγραμμα θα πρέπει να κάνει χρήση της στοίβας για να αποφασίσει για το ταίριασμα των ετικετών.
- Η εργασία σας θα αξιολογηθεί πάνω σε html αρχεία με απλές ετικέτες ανοίγματος και κλεισίματος, σαν αυτές που αναφέρθηκαν παραπάνω. Π.χ. μην ανησυχείτε για tags τύπου

link . Επίσης δεν χρειάζεται να ελέγχετε αν το html αρχείο έχει άλλα συντακτικά λάθη. Ασχοληθείτε μόνο με το ταίριασμα των ετικετών.

- Το πρόγραμμά σας θα έχει μέθοδο main και το μονοπάτι για το προς εξέταση html αρχείο θα δίνεται ως παράμετρος γραμμής εντολών κατά την εκτέλεση του προγράμματός σας (χρήση args[0] της main). Η εκτέλεση του προγράμματος θα πρέπει να είναι δυνατή είτε από τη γραμμή εντολών με μια εντολή της μορφής

```
> java TagMatching path_to_html_file.html
```

είτε από το IDE σας (δείτε το αρχείο Eclipse-command line arguments.pdf με οδηγίες, στον φάκελο της Εργασίας 1).

Μέρος Γ [30 μονάδες]. Χρησιμοποιώντας την υλοποίηση της ουράς FIFO του μέρους Α, γράψτε ένα πρόγραμμα-πελάτη, το οποίο λύνει το εξής λογιστικό πρόβλημα: όταν θέλουμε να υπολογίσουμε το καθαρό κέρδος που προκύπτει από πώληση μετοχών, υπάρχει αμφισημία ως προς το ποιες μετοχές θεωρούμε ότι πουλάμε (όταν έχουμε μετοχές που έχουν αγοραστεί σε διαφορετικές τιμές). Για παράδειγμα, έστω ότι την χρονική στιγμή t1 αγοράσαμε 50 μετοχές από κάποια εταιρεία, σε τιμή 25 ευρώ. Μετέπειτα, τη χρονική στιγμή t2, αγοράσαμε άλλες 40 μετοχές από την ίδια εταιρεία σε τιμή 22 ευρώ, και τη χρονική στιγμή t3, αγοράσαμε ακόμα 30 μετοχές σε τιμή 32 ευρώ. Έστω ότι αργότερα πουλάμε 110 μετοχές (από τις 120 που έχουμε) σε τιμή 30 ευρώ. Η λογιστική αρχή που χρησιμοποιείται για να καθορίσουμε το καθαρό κέρδος της πώλησης εκείνη τη στιγμή στηρίζεται στο πρωτόκολλο FIFO. Τα περισσότερα λογισμικά διαχείρισης επενδυτικών χαρτοφυλακίων χρησιμοποιούν αυτή την αρχή. Στο παράδειγμά μας, αυτό σημαίνει ότι από τις 110 μετοχές που πουλάμε, οι πρώτες 50 θεωρούμε ότι προέρχονται από αυτές που αγοράστηκαν πιο νωρίς, δηλαδή την στιγμή t1, οι επόμενες 40 από αυτές που αγοράστηκαν την στιγμή t2, και οι επόμενες 20 από την τελευταία αγορά. Έτσι το καθαρό κέρδος μας εκείνη τη στιγμή θεωρούμε ότι είναι:

$$50\ (30 - 25) + 40\ (30 - 22) + 20\ (30 - 33) = 250 + 320 - 60 = 510$$

Σημειώστε ότι στον παραπάνω υπολογισμό, δεν λαμβάνουμε καθόλου υπόψη στο κέρδος τις 10 μετοχές που έχουν περισσέψει από τις 120, και δεν έχουμε πουλήσει (αυτές θα ληφθούν υπόψη σε μελλοντικές πωλήσεις).

Γράψτε ένα πρόγραμμα το οποίο θα διαβάζει από ένα .txt αρχείο μια ακολουθία από διαδοχικές συναλλαγές της μορφής:

```
buy 50 price 25
buy 40 price 22
buy 30 price 33
sell 110 price 30
buy 25 price 35
sell 30 price 40
...
```

και θα τυπώνει το συνολικό καθαρό κέρδος ή ζημιά που προκύπτει ακριβώς μετά από την τελευταία πώληση. Το παραπάνω αρχείο απεικονίζει τις συναλλαγές με τη σειρά που γίνονται, δηλαδή στο παράδειγμα αυτό έγιναν πρώτα 3 αγορές, μετά πώληση, μετά μια αγορά και μετά ξανά πώληση. Το αρχείο θα ξεκινά πάντα με τουλάχιστον μια συναλλαγή αγοράς και θα τελειώνει πάντα με συναλλαγή πώλησης.

Οδηγίες υλοποίησης:

- Το πρόγραμμά σας **πρέπει να λέγεται** *NetBenefit.java*.
- Για το διάβασμα του αρχείου εισόδου, μπορείτε να χρησιμοποιήσετε μεθόδους αντίστοιχες με αυτές που αναφέρθηκαν στο Μέρος Β. Στη συνέχεια θα πρέπει να χρησιμοποιήσετε την υλοποίηση της ουράς FIFO του Μέρους Α, για να υπολογίσετε το καθαρό κέρδος.
- Θεωρούμε ότι σε όλες τις γραμμές του αρχείου εισόδου, ο αριθμός των μετοχών και οι τιμές είναι ακέραιοι αριθμοί. Δεν απαιτείται να ελέγξετε ότι όλα τα στοιχεία του αρχείου εισόδου είναι σε σωστή μορφή (οι εργασίες θα αξιολογηθούν πάνω σε αρχεία σαν το παράδειγμα παραπάνω). Θα πρέπει όμως να τυπώνετε μήνυμα λάθους αν μέσα στο αρχείο υπάρχει πώληση μεγαλύτερου αριθμού μετοχών από αυτές που έχουν αγοραστεί μέχρι εκείνη τη στιγμή (π.χ. αν ξεκινάει με συναλλαγή πώλησης).
- Το μονοπάτι προς το αρχείο εισόδου θα δίνεται ως όρισμα, σε αντιστοιχία με το Μέρος Β. Π.χ. η εκτέλεση από τη γραμμή εντολών θα γίνεται ως εξής

```
> java NetBenefit path_to_text_file.txt
```

Μέρος Δ - Αναφορά παράδοσης [10 μονάδες]. Ετοιμάστε μία σύντομη αναφορά σε pdf αρχείο (μην παραδώσετε Word ή txt αρχεία!) με όνομα project1-report.pdf, στην οποία θα αναφερθείτε στα εξής:

a. Εξηγήστε συνοπτικά πώς υλοποιήσατε τις διεπαφές στο μέρος Α (άνω όριο 2 σελίδες).
b. Για το μέρος Β, εξηγήστε πώς χρησιμοποιήσατε την υλοποίηση από το μέρος Α για να φτιάξετε το πρόγραμμα που ζητείται (άνω όριο 2 σελίδες).
c. Ομοίως για το μέρος Γ (άνω όριο 2 σελίδες).
d. 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).

Το συνολικό μέγεθος της αναφοράς θα πρέπει να είναι τουλάχιστον 2 σελίδες και το πολύ 6 σελίδες.

Οδηγίες Παράδοσης

Η εργασία σας θα πρέπει να μην έχει συντακτικά λάθη και να μπορεί να μεταγλωττίζεται. **Εργασίες που δεν μεταγλωττίζονται χάνουν το 50% της συνολικής αξίας. Η διόρθωση των εργασιών από τους βοηθούς του μαθήματος αφορά την αξιολόγηση μεταγλωττίσιμων υλοποιήσεων και όχι την εύρεση λαθών μεταγλώττισης.**

Η εργασία θα αποτελείται από:

1. Τον πηγαίο κώδικα (source code). Τοποθετήστε σε ένα φάκελο με όνομα **src** τα αρχεία java που έχετε φτιάξει. Χρησιμοποιήστε τα όνοματα των κλάσεων όπως ακριβώς δίνονται στην εκφώνηση. Επιπλέον, φροντίστε να συμπεριλάβετε όποια άλλα αρχεία πηγαίου κώδικα απαιτούνται για να μεταγλωττίζεται η εργασία σας. Φροντίστε επίσης να προσθέσετε επεξηγηματικά σχόλια όπου κρίνετε απαραίτητο στον κώδικά σας.
2. Την αναφορά παράδοσης.

Όλα τα παραπάνω αρχεία θα πρέπει να μπουν σε ένα αρχείο zip. Το όνομα που θα δώσετε στο αρχείο αυτό θα είναι ο αριθμός μητρώου σας πχ. 3130056_3130066.zip ή 3130056.zip (αν δεν είστε σε ομάδα). Στη συνέχεια, θα υποβάλλετε το zip αρχείο σας στην περιοχή του μαθήματος «Εργασίες» στο e-class. Δεν χρειάζεται υποβολή και από τους 2 φοιτητές μιας ομάδας.