α. Εξηγήστε συνοπτικά πώς υλοποιήσατε τις διεπαφές στα μέρη Α και Γ. Ειδικά για το Μέρος Γ, εξηγήστε την ιδέα με την οποία μπορείτε να αποφύγετε τη χρήση 2 δεικτών για την αρχή και το τέλος της ουράς (άνω όριο 3 σελίδες).

Μέρος Α:

StringQueueImpl

Στη συγκεκριμένη κλάση υλοποιήσαμε όλες τις μεθόδους της διεπαφής StringQueue!!!
Πιο συγκεκριμένα στις πρώτες γραμμές του κώδικα αρχικοποιήσαμε τα first-last όπου μας δείχνουν την αρχή και το τέλος της ουράς και βάλαμε έναν μετρητή counter που στην πορεία θα μας βοηθούσε να υλοποιήσουμε την μέθοδο size.

Έχουμε έναν default constructor και ορίζουμε την κλάση node για να κάνουμε χρήση των κόμβων και της εντολής next.Μέσα σε αυτήν ορίζουμε το item οπού μας δείχνει το string μέρος του κόμβου.

public Boolean isEmpty():

Εδώ κάνουμε έλεγχο αν υπάρχει κάποιο στοιχείο μέσα στην ουρά και μας επιστρέφει μια Boolean τιμή.

public void put (String item):

Δημιουργούμε έναν καινούριο κόμβο που τον προσθέτω στην ουρά. Αν είναι άδεια, τον αρχικοποιώ ως την αρχή κ το τέλος της ουράς. Αλλιώς, βάζω το τελευταίο στοιχείο να δείχνει στον καινούριο κόμβο. Τέλος, αυξάνω τον μετρητή.

public String get() :

Αν η ουρά δεν είναι άδεια επιστρέφει το παλαιότερο στοιχείο της ουράς και το αφαιρούμε μειώνοντας τον μετρητή. Ορίζουμε ως πρώτο στοιχείο τον αμέσως επόμενο κόμβο από το στοιχείο που αφαιρέσαμε.
Αν η ουρά είναι άδεια εμφανίζεται μήνυμα εξαίρεσης.

public String peek():

Αν η ουρά δεν είναι άδεια επιστρέφει το παλαιότερο στοιχείο της χωρίς να το αφαιρεί. Αλλιώς εμφανίζεται μήνυμα εξαίρεσης.

public void printQueue(PrintStream stream):

Αν η ουρά είναι άδεια εμφανίζει κατάλληλο μήνυμα ,αλλιώς εμφανίζει τα στοιχεία της ουράς από το παλαιότερο στο νεότερο.

public int size():

Επιστρέφουμε τον μετρητή counter που αυξανόταν σε κάθε προσθήκη στοιχείου στην ουρά.

StringStackImpl

Στη συγκεκριμένη κλάση υλοποιήσαμε όλες τις μεθόδους της διεπαφής StringStack!!!
Πιο συγκεκριμένα, στην αρχή αρχικοποιούμε πάλι έναν μετρητή αντίστοιχα κ μια μεταβλητή head όπου μας δείχνει την αρχή της στοίβας.

Έχουμε έναν default constructor και ορίζουμε την κλάση node για να κάνουμε χρήση των κόμβων και της εντολής next.Μέσα σε αυτήν ορίζουμε το item οπού μας δείχνει το string μέρος του κόμβου.

public Boolean isEmpty():

Εδώ κάνουμε έλεγχο αν υπάρχει κάποιο στοιχείο μέσα στην στοίβα και μας επιστρέφει μια Boolean τιμή.

public String push():

Προσθέτουμε καινούριο κόμβο στην στοίβα και αρχικοποιούμε ως head-κορυφή το καινούριο στοιχείο. Επίσης αυξάνουμε τον μετρητή counter κατά ένα.

public String pop():

Αν η στοίβα δεν είναι άδεια, επιστρέφουμε το νεότερο στοιχείο της και το αφαιρούμε μειώνοντας τον μετρητή και ορίζοντας ως head τον αμέσως επόμενο κόμβο από αυτόν που αφαιρέσαμε.

public String peek():

Αν η στοίβα δεν είναι άδεια, επιστρέφουμε το νεότερο στοιχείο της χωρίς να το αφαιρεί. Αλλιώς εμφανίζεται μήνυμα εξαίρεσης.

public void printStack (PrintStream stream):

Αν η στοίβα είναι άδεια εμφανίζει κατάλληλο μήνυμα ,αλλιώς εμφανίζει τα στοιχεία της από το νεότερο στοιχείο στο παλαιότερο.

public int size():

Επιστρέφουμε τον μετρητή counter που αυξανόταν σε κάθε προσθήκη στοιχείου στην στοίβα.

Μέρος Γ:

StringQueueWithOnePointer

Στο συγκεκριμένο σημείο δουλέψαμε με κυκλική λίστα χρησιμοποιώντας μόνο έναν δείκτη, τον last όπου δείχνει στο τέλος της ουράς. Έτσι, θα μπορούσαμε να έχουμε πρόσβαση και στο πρώτο στοιχείο της ουράς, αφού το τελευταίο στοιχείο θα δείχνει στο πρώτο (κυκλική). Άρα, last.next ---> πρώτο .

Στη συγκεκριμένο κλάση, ορίζουμε έναν default κατασκευαστή και αρχικοποιούμε το κόμβο last, δηλαδή τον pointer που θα χρησιμοποιήσουμε, έναν μετρητή και κάποιους κόμβους ακόμα που θα χρειαστούν παρακάτω.

Επίσης, ορίζουμε την κλάση node για να κάνουμε χρήση των κόμβων και της εντολής next.Μέσα σε αυτήν ορίζουμε το item οπού μας δείχνει το string μέρος του κόμβου.

public Boolean isEmpty():

Εδώ κάνουμε έλεγχο αν υπάρχει κάποιο στοιχείο μέσα στην ουρά και μας επιστρέφει μια Boolean τιμή με βάση τον μοναδικό δείκτη last.

public void put (String item):

Δημιουργούμε έναν καινούριο κόμβο που τον προσθέτω στην ουρά. Αν η ουρά δεν είναι άδεια, βάζω το νέο στοιχείο να δείχνει στην αρχή της ουράς που μπορώ να αναφερθώ ως το next στοιχείο από το last. Ως επόμενο στοιχείο από το last ορίζω το νέο κόμβο και ακόμα ονομάζω το καινούριο κόμβο ως last αφού τον πρόσθεσα στο τέλος της ουράς κ πλέον είναι αυτό το στοιχείο τελευταίο. Τέλος αυξάνω τον μετρητή. Αν είναι άδεια, αρχικοποιώ το νέο κόμβο ως last που δείχνει στον εαυτό του αφού είναι κυκλική ουρά.

public String get() :

Αν η ουρά δεν είναι άδεια επιστρέφει το παλαιότερο στοιχείο της ουράς και το αφαιρούμε μειώνοντας τον μετρητή. Με την εντολή last.next.next έχω πρόσβαση στο δεύτερο στοιχείο της ουράς. Αφού αφαιρώ το πρώτο στοιχείο βάζω το last στοιχείο να δείχνει πλέον στο δεύτερο στοιχείο της ουράς.
Αν έχω μόνο ένα στοιχείο στην ουρά, το αφαιρώ και πλέον ο pointer last θα δείχνει σε null.
Αν η ουρά είναι άδεια εμφανίζεται μήνυμα εξαίρεσης.

public String peek():

Αν η ουρά δεν είναι άδεια επιστρέφει το πρώτο στοιχείο της χωρίς να το αφαιρεί. Αλλιώς εμφανίζεται μήνυμα εξαίρεσης.

public void printQueue (PrintStream stream):

Αν η ουρά είναι άδεια εμφανίζει κατάλληλο μήνυμα ,αλλιώς εμφανίζει τα στοιχεία της ουράς από το παλαιότερο στο νεότερο. Αν έχω μόνο ένα στοιχείο, τότε το εμφανίζει, αν έχω περισσότερα εμφανίζει επαναληπτικά κάθε στοιχείο ξεκινώντας από το πρώτο (last.next) και σταματάει με το που φτάσει στο τελευταίο. Όταν βγει από την επανάληψη εμφανίζει και το τελευταίο.

public int size():

Επιστρέφουμε τον μετρητή counter που αυξανόταν σε κάθε προσθήκη στοιχείου στην ουρά.

β. Για το μέρος Β, εξηγήστε πώς χρησιμοποιήσατε την υλοποίηση από το μέρος Α για να φτιάξετε το πρόγραμμα που ζητείται (άνω όριο 3 σελίδες).

Μέρος Β:

Thiseas

Στην συγκεκριμένη κλάση κάνουμε χρήση της στοίβας που υλοποιήσαμε στο πρώτο ερώτημα. Μέσα σε αυτή γίνεται η αποθήκευση των συντεταγμένων των στοιχείων του πίνακα ,που αφορά το μονοπάτι που ακολουθήσαμε για να φτάσουμε στην έξοδο. Τα στοιχεία της στοίβας είναι τύπου string.
Η τοποθέτηση των στοιχείων γίνεται μέσω της μεθόδου push() ,ενώ η αφαίρεση μέσω της μεθόδου pop().

Όσον αφορά το backtracking:

Αφαιρούμε το στοιχείο που μας βγάζει σε αδιέξοδο με την μέθοδο pop(), ενώ στην θέση όπου βρισκόμασταν τοποθετούμε “1”.
Η μέθοδος peek() μας επιστρέφει το προηγούμενο στοιχείο του μονοπατιού μας, το οποίο το χωρίζουμε σε 2 κομμάτια με την βοήθεια της μεθόδου substring.
Τα κομμάτια αυτά μας δείχνουν τις συντεταγμένες (i,j) του στοιχείου αυτού.

Έπειτα συνεχίζουμε τον έλεγχο για να βρούμε άλλη πιθανή έξοδο .

Τέλος, αν η στοίβα μας είναι άδεια σημαίνει ότι κάθε κατεύθυνση που ακολουθήσαμε κατέληγε σε αδιέξοδο, και συνεπώς αφαιρέθηκε όλο το μονοπάτι ,οπότε εμφανίζει κατάλληλο μήνυμα και τερματίζει.

p3180186 – Trampari Larda Eleni
p3180172 – Spanos Charalampos