

Εργαστήριο #4: Δομές επανάληψης

Απαιτήσεις ασκήσεων: Απαγορεύεται αυστηρά η χρήση goto. Διαβάστε την εκφώνηση κάθε άσκησης προσεκτικά πριν ξεκινήσετε. Χρησιμοποιήστε την καταλληλότερη δομή επανάληψης για κάθε περίπτωση. Δεν επιτρέπεται η χρήση πινάκων.

Άσκηση 1

Αποθηκεύστε το πρόγραμμά σας σε αρχείο με όνομα **lab4a.c**.

Διευκρινίσεις: Τα μηνύματα εξόδου του προγράμματος εμφανίζονται παρακάτω με κίτρινο φόντο. Αυτή τη φορά περιλαμβάνουμε το χαρακτήρα αλλαγής γραμμής σε όλα τα μηνύματα που χρειάζεται.

Ζητούμενα: Γράψτε ένα πρόγραμμα το οποίο αρχικά διαβάζει ένα χρονικό διάστημα κατά το οποίο είναι δεσμευμένη μια αίθουσα για μαθήματα και ακολούθως ενημερώνει τον χρήστη για τη διαθεσιμότητα της αίθουσας κατά τη διάρκεια ωρών που αυτός ζητά να τη δεσμεύσει. Το πρόγραμμα λειτουργεί ως εξής:

- Εκτυπώνει το μήνυμα **"Reserved:\n"** και διαβάζει τα άκρα του χρονικού διαστήματος (σε ώρες) που είναι δεσμευμένη η αίθουσα. Η είσοδος δίνεται στη μορφή S-E όπου S η αρχική ώρα από την οποία η αίθουσα είναι δεσμευμένη και E η τελική ώρα από την οποία η αίθουσα είναι ελεύθερη. Όσο η αρχική ώρα δεν προηγείται της τελικής ή κάποια ώρα είναι αρνητική ~~δεν είναι θετική~~, το βήμα 1 επαναλαμβάνεται.
- Σε επανάληψη,
 - Εκτυπώνει το μήνυμα **"#\n"**
 - Εκτυπώνει το μήνυμα **"Request:\n"** και διαβάζει ένα διάστημα χρόνου, που αποτελεί αίτημα ελέγχου διαθεσιμότητας της αίθουσας, με τη μορφή X-Y όπου X η αρχική ώρα και Y η τελική ώρα.
 - Αν η αρχική και η τελική ώρα του αιτήματος έχουν τιμή μηδέν, η επανάληψη τερματίζει.
 - Εφόσον δεν είναι μηδέν, τότε αν κάποια ώρα είναι αρνητική ~~δεν είναι θετικές~~ ή η αρχική ώρα δεν προηγείται της τελικής, το πρόγραμμα εκτυπώνει το μήνυμα **"X-Y invalid.\n"**, όπου X η αρχική και Y η τελική ώρα του αιτήματος και προχωρά στην επόμενη επανάληψη.
 - Διαφορετικά, το πρόγραμμα ελέγχει αν το διάστημα του αιτήματος περιέχεται πλήρως μέσα στο διάστημα που η αίθουσα είναι δεσμευμένη. Αν ναι, τότε εκτυπώνει το μήνυμα **"X-Y not ok.\n"** όπου X η αρχική και Y η τελική ώρα του αιτήματος.
 - Διαφορετικά, ελέγχει αν το διάστημα του αιτήματος έχει μερική επικάλυψη με το διάστημα δέσμευσης. Αν ναι, τότε εκτυπώνει το μήνυμα **"X-Y maybe.\n"** όπου X η αρχική και Y η τελική ώρα του αιτήματος.
 - Εάν δεν υπάρχει επικάλυψη ανάμεσα στο διάστημα αιτήματος και στο διάστημα δέσμευσης, το πρόγραμμα εκτυπώνει το μήνυμα **"X-Y ok.\n"** όπου X η αρχική και Y η τελική ώρα του αιτήματος και η επανάληψη τερματίζει.
- Μετά το τέλος της επανάληψης, εκτυπώνει το μήνυμα **"#\n"**.
- Τέλος, εκτυπώνει το μήνυμα **"Valid X. Invalid Y. Maybe Z.\n"** όπου X το συνολικό πλήθος έγκυρων διαστημάτων (μη-invalid και μη-μηδενικών) που ελέγχθηκαν στο βήμα 2, Y το συνολικό πλήθος άκυρων (invalid) διαστημάτων που δόθηκαν ως είσοδος στο βήμα 2 και Z το συνολικό πλήθος αιτημάτων ελέγχου για τα οποία υπήρχε μερική αλλά όχι πλήρης επικάλυψη με το διάστημα που η αίθουσα είναι δεσμευμένη.

Υποθέσεις: Υποθέστε ότι η αίθουσα είναι δεσμευμένη μόνο για το διάστημα που δίνεται στο βήμα 1 και ελεύθερη για όλο το υπόλοιπο 24ωρο, και ότι τα αιτήματα ελέγχου διαθεσιμότητας γίνονται για πολλαπλάσια ολόκληρων ωρών εντός ενός 24ώρου.

Ενδεικτικές εκτελέσεις.

Τα μηνύματα του προγράμματος εμφανίζονται σε κίτρινο φόντο, η είσοδος του χρήστη σε λευκό φόντο.

Reserved: 0-0 Reserved: -1--2 Reserved: 8-5 Reserved: 5-8 ## Request: 0-0 ## Valid 0. Invalid 0. Maybe 0.	Reserved: 5-8 ## Request: 5-7 5-7 not ok. ## Request: 4-6 4-6 maybe. ## Request: 6-8 6-8 not ok. ## Request: 3-5 3-5 ok. ## Valid 4. Invalid 0. Maybe 1.	Reserved: 5-8 ## Request: 8-8 8-8 invalid. ## Request: 8-5 8-5 invalid. ## Request: 4-9 4-9 maybe. ## Request: 5-8 5-8 not ok. ## Request: 0-0 ## Valid 2. Invalid 2. Maybe 1.
---	---	--

Άσκηση 2

Αποθηκεύστε το πρόγραμμά σας σε αρχείο με όνομα **lab4b.c**.

Ζητούμενα: Γράψτε ένα πρόγραμμα το οποίο:

1. Εκτυπώνει το μήνυμα "Size: \n" και διαβάζει έναν ακέραιο αριθμό (μέγεθος). Μπορείτε να υποθέσετε ότι αυτός θα είναι τουλάχιστον 1.
2. Εκτυπώνει το μήνυμα "Chars: \n" και διαβάζει 2 χαρακτήρες που θα δοθούν χωρίς κενό ανάμεσα.
3. Εμφανίζει στην οθόνη ένα σχήμα που σχηματίζεται από τους δύο χαρακτήρες ως εξής: Κάθε γραμμή αποτελείται από τόσες εμφανίσεις του κάθε χαρακτήρα όσο ο ακέραιος αριθμός που δόθηκε στο 1^ο βήμα (μέγεθος), τοποθετώντας τους εναλλάξ, χωρίς κενό ανάμεσα, και **χαρακτήρα αλλαγής γραμμής** στο τέλος. Η πρώτη γραμμή ξεκινά από τον χαρακτήρα που δόθηκε πρώτος και κάθε επόμενη γραμμή ξεκινά από τον χαρακτήρα που είναι διαφορετικός από αυτόν που ξεκίνησε η προηγούμενη. Το σχήμα έχει συνολικά τόσες γραμμές όσο και το μέγεθος που διαβάστηκε στο βήμα 1. Μπορείτε να φανταστείτε το σχήμα σαν μισή σκακιέρα. Για παράδειγμα, αν το μέγεθος που δόθηκε στο βήμα 1 είναι 4, και οι χαρακτήρες είναι '-' και 'X' τότε το σχήμα που θα εμφανιστεί στην οθόνη είναι:

```
-X-X-X-X
X-X-X-X-
-X-X-X-X
X-X-X-X-
```

Σκεφτείτε έναν έξυπνο τρόπο να υλοποιήσετε την εναλλαγή χαρακτήρων.

Υποβολή

Ανοίξτε ένα τερματικό και μεταβείτε στον κατάλογο που βρίσκονται τα αρχεία lab4a.c και lab4b.c. Εκτελέστε την εντολή:

```
zip -r lab4.zip lab4a.c lab4b.c
```

Θα πρέπει να δημιουργηθεί ένα αρχείο με όνομα lab4.zip

Επιβεβαιώστε ότι τα περιεχόμενά του είναι σωστά γράφοντας την εντολή:

```
zip -sf lab4.zip
```

Θα πρέπει να σας εμφανίσει κάτι σαν το παρακάτω:

```
Archive contains:  
  lab4a.c  
  lab4b.c  
Total 2 entries
```

Υποβάλετε στο autolab το αρχείο lab4.zip.