Ασκήσεις 6ης διάλεξης – Κείμενα, Structs

Για την κάθε μία από τις παρακάτω εργασίες:

(α) χρησιμοποιείστε ένα ξεχωριστό project – το κατάλληλο smProject που δίνεται,

(β) βάλτε σχόλια στον κώδικα που εξηγούν τα βήματα της επίλυσης και

(γ) δημιουργήστε μέσα στην smMain κώδικα που θα επιδεικνύει την καλή λειτουργία των συναρτήσεων.

Σημείωση: Στον τελικό κώδικα μην χρησιμοποιήσετε τις `printf` μέσα στις ζητούμενες συναρτήσεις.

Συμβουλές: Διαβάστε προσεκτικά δύο ή τρεις φορές την εκφώνηση. Επιλέξτε περιγραφικά και αυτοεξηγούμενα ονόματα μεταβλητών. Χρησιμοποιήστε καλή στοίχιση. Κατά τη συγγραφή του προγράμματος προσθέστε όσες `printf` χρειαστείτε για να βλέπετε τις τιμές των μεταβλητών ώστε να εντοπίζετε ευκολότερα τυχόν λάθη, πριν παραδώσετε όμως, αφαιρέστε τις "βοηθητικές" και κρατήστε μόνο όσες είναι απαραίτητες.

ΠΡΟΣΟΧΗ! Μην ξεχάσετε να κατεβάσετε και να χρησιμοποιήσετε τα αντίστοιχα smProject για την κάθε άσκηση!

ΠΡΟΣΕΞΤΕ ΟΠΩΣΔΗΠΟΤΕ ΤΑ ΠΑΡΑΚΑΤΩ

Ο τρόπος με τον οποίο πρέπει να υποβάλλετε ερωτήσεις περιγράφεται εδώ:

https://qna.c-programming.allos.gr/doku.php?id=qna:technical:questions

Ο τρόπος με τον οποίο πρέπει να υποβάλλετε τον κώδικα των εργασιών στο σύστημα υποβολής περιγράφεται εδώ:

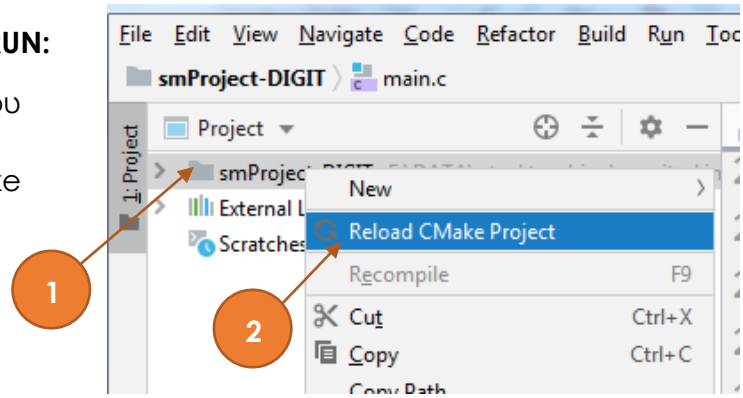
https://qna.c-programming.allos.gr/doku.php?id=qna:lesson:projects:how-to-submit

Οι έτοιμες συναρτήσεις της βιβλιοθήκης smLib που διατίθενται για χρήση στα smProjects, περιγράφονται (εκτός από τις διαφάνειες) και στο site ερωταποκρίσεων εδώ:

https://qna.c-programming.allos.gr/doku.php?id=qna:misc:sm-library

ΕΑΝ ΔΕΝ ΕΜΦΑΝΙΖΟΝΤΑΙ ΕΝΕΡΓΑ ΤΑ BUILD / RUN:

1. Κάνω **δεξί κλικ** πάνω στο όνομα του project και εμφανίζεται το μενού
2. Κάνω **απλό κλικ** στο Reload CMake Project, δηλαδή τη 2η επιλογή



Εργασία 6α – Κατάληξη κειμένου

Βαθμός δυσκολίας: **1/3**

Όνομα smProject: **smProject-HAS-SUFFIX**

Περιγραφή

Ζητείται να γράψετε την παρακάτω συνάρτηση, η οποία είναι

```
bool hasSuffix(char text[], char suffix[], bool caseSensitive)
```

η οποία απαντά στο ερώτημα εάν το κείμενο text τελειώνει στο κείμενο suffix. Η σύγκριση των κειμένων μπορεί να είναι caseSensitive (δηλαδή τα πεζά με τα κεφαλαία θεωρούνται διαφορετικά) ή όχι.

Για παράδειγμα θα πρέπει να επιστρέφει αληθές όταν είναι:

text : "Hello everyone"
suffix : "one"

και ψευδές όταν είναι :

text : "Hello everyone!"
suffix : "one"

αλλά το παρακάτω όταν είναι και `caseSensitive==true` επιστρέφει ψευδές, ενώ όταν είναι `caseSensitive==false` επιστρέφει αληθές:

text : "Hello everyone"
suffix : "One"

Προσέξτε να λάβετε υπόψη σας όλα τα ενδεχόμενα, με διάφορα μήκη των κειμένων. Μην θεωρήσετε δεδομένο ότι το κείμενο `text` είναι μακρύτερο από την κατάληξη `suffix`.

ΣΗΜΕΙΩΣΗ : Δίνονται έτοιμες οι `toLowerCase` που μετατρέπει ένα κείμενο σε πεζό και η `stringLength` που επιστρέφει το μήκος ενός κειμένου σε χαρακτήρες.

Εργασία 6β – Μακρύτερη λέξη

Βαθμός δυσκολίας: **1/3**

Όνομα smProject: **smProject-LONGEST-WORD**

Περιγραφή

Καλείστε να γράψετε τη συνάρτηση:

```
int longestWord(char text[])
```

η οποία επιστρέφει **το μήκος** της μεγαλύτερης λέξης που υπάρχει στο κείμενο `text`. Ως λέξη εννοούμε κάθε ομάδα **συνεχόμενων** χαρακτήρων a-z ή A-Z (όπως είναι και η λογική της `isalpha`).

Δείτε μερικά παραδείγματα υπολογισμού μήκους λέξεων:

```
This is a simple test, a test for the longest word of this Super-text!
         ^--6-^                    ^--7--^             ^-5-^
```

Για το παραπάνω κείμενο (που φαίνονται τα μήκη μόνο από μερικές λέξεις) η συνάρτηση θα πρέπει να επιστρέφει 7.

```
A b c d e f g h I j k l m n o p q r s t u v w x y z
```

Ενώ σε αυτό ⇧ πρέπει να επιστρέφει 1.

```
!-------------- 888 --------------!
```

Και σε αυτό ⇧ να επιστρέφει 0.