Assignment 1: "Translate" Program

Σκοπός
Ο σκοπός αυτής της άσκησης είναι να σας βοηθήσει να κάνετε επανάληψη (1) των βασικών στοιχείων της γλώσσας προγραμματισμού C, (2) της χρήσης των εργαλείων προγραμματισμού GNU/UNIX, ειδικά gcc, shell, και editing (emacs/vim/nano), και (3) να σχεδιάσετε ένα απλό C πρόγραμμα με ακρίβεια.

Background
Το iso8859 standard ορίζει character sets για διάφορες γλώσσες. Το iso8859-1 (latin-1) είναι το «συνηθισμένο» σετ με τους αγγλικούς χαρακτήρες. Το iso8859-7 περιλαμβάνει τους χαρακτήρες των ελληνικών. Για περισσότερες λεπτομέρειες: RFC 1345.

Η άσκηση
Η άσκηση σας ζητάει να γράψετε ένα πρόγραμμα που μεταφράζει το input από χαρακτήρες του iso8859-7 (αγγλικοί και ελληνικοί χαρακτήρες) σε χαρακτήρες του iso8859-1 (αγγλικοί χαρακτήρες). Το πρόγραμμα θα δουλεύει ως ένα filter του Unix shell, δηλαδή θα δέχεται input από το standard input και θα κάνει output στο standard output και πιθανόν στο standard error. Ειδικότερα το πρόγραμμά σας θα διαβάζει χαρακτήρες κειμένου από το standard input (το οποίο υποθέτουμε είναι κείμενο γραμμένο με ελληνικούς χαρακτήρες iso8859-7), γράφει το ίδιο κείμενο, μεταφρασμένο με βάση τους κανόνες που ακολουθούν στο standard output, και γράφει πιθανά error και warning μηνύματα στο standard error. Τυπικές εκτελέσεις του προγράμματος σας σε command line στο UNIX θα μπορούσε να είναι:

```
$ cat test.7 | translate
$ translate < test.7 > test.1
```

Με αυτό τον τρόπο θα μπορείτε να διαβάσετε σε οποιοδήποτε xterm ελληνικά text files που έχουν γραφτεί σε iso8859-7. Επίσης, αν χρησιμοποιείτε κάποιον text-based mail client (e.g. pine), θα μπορείτε να διαβάζετε μηνύματα που σας στέλνουν στα ελληνικά σε οποιοδήποτε τερματικό.

Λεπτομέρειες

Το πρόγραμμα σας πρέπει να αντικαθιστά τους χαρακτήρες του iso8859-7 με χαρακτήρες του iso8859-1 σύμφωνα με τους παρακάτω κανόνες:

- Οι απλοί χαρακτήρες αντικαθίστανται ως εξής με έναν ή δύο χαρακτήρες (ο πίνακας δείχνει μόνο τα κεφαλαία αλλά το ίδιο ισχύει και για τους μικρούς χαρακτήρες):

Α	A
Β	V
Γ	G
Δ	D
Ε	E
Ζ	Z
Η	H
Θ	8
Ι	I
Κ	K
Λ	L
Μ	M
Ν	N
Ξ	KS
Ο	O
Π	P
Ρ	R
Σ	S
Τ	T
Υ	Y
Φ	F
Χ	X
Ψ	PS
Ω	W

- Οι χαρακτήρες με τόνους αντικαθίστανται με τους αντίστοιχους αγγλικούς και με ένα quote (δύο χαρακτήρες) αντί για τόνο ως εξής:

Ά	'A
Έ	'E
Ή	'H
Ί	'I
Ό	'O
Ώ	'W
Ύ	'Y
ά	a'
έ	e'
ή	h'

ί	i'
ό	o'
ώ	w'
ύ	y'

- Οι χαρακτήρες που έχουν διαλυτικά αντικαθίστανται με δύο ή τρεις χαρακτήρες ως εξής (ο πίνακας δείχνει μόνο τους μικρούς χαρακτήρες):

ϊ	i"
ϋ	y'"
ΐ	i'"
ΰ	y'"

- Οι δίφθογγοι ΜΠ/μπ και ΝΤ/ντ αντικαθίστανται με ένα μόνο χαρακτήρα B/b και D/d αντίστοιχα (κεφαλαία και μικρά). Επίσης, το Μπ(Ντ) με B(D), το μΠ(νΤ) με b(d). Υλοποιήστε αυτόν τον κανόνα με ένα state machine, όπως εξηγήσαμε στην τάξη.

- Μπορείτε να αντικαταστήσετε τους υπόλοιπους χαρακτήρες του iso8859-7 με χαρακτήρες του iso8859-1 που νομίζετε ότι κάνουν το μεταφρασμένο κείμενο πιο ευανάγνωστο ή απλά να τους αντιστοιχήσετε στον χαρακτήρα με τον ίδιο κωδικό.

Δεν πρέπει να κάνετε κάποια υπόθεση για το μέγιστο μέγεθος του input.

Ακολουθήστε τα επόμενα βήματα, όπου κάθε βήμα μπορεί και θα πρέπει να είναι ένα ξεχωριστό commit στο git repository σας.

Βήματα - Συστάσεις

Υλοποιήστε τους κανόνες 1,2,3,5, όπου κάθε iso8859-7 χαρακτήρας του input αντικαθίσταται με έναν ή περισσότερους χαρακτήρες του iso8859-1. Χωρίστε το πρόγραμμά σας σε λίγες συναρτήσεις. Πιθανόν θα χρειαστείτε συναρτήσεις για:

- Την αρχικοποίηση της δομής που αποθηκεύει τα mappings των χαρακτήρων του iso8859-7 σε χαρακτήρες του iso8859-1.
- Την αντικατάσταση ενός δεδομένου χαρακτήρα που διαβάστηκε από το standard input με τους προδιαγεγραμμένους από τους κανόνες χαρακτήρες στο standard output.
- Τη συνάρτηση main που απλά θα καλεί τη συνάρτηση αρχικοποίησης και, σε ένα loop, θα διαβάζει χαρακτήρες από το standard input και θα τους μεταφράζει με την κλήση της κατάλληλης συνάρτησης.
- Για κάθε κανόνα 1,2,3,5 που υλοποιείται θα πρέπει να εκτελείτε ένα ξεχωριστό commit.

Στη συνέχεια υλοποιήστε τον κανόνα 4, όπου 2 χαρακτήρες του iso8859-7 αντικαθίστανται με 1 χαρακτήρα του iso8859-1. Χρησιμοποιήστε μια state machine όπως εξηγήσαμε στο μάθημα.

Αυτό μπορεί να γίνει απλά με την αλλαγή της main ώστε να υλοποιεί τη state machine.

Προσπαθήστε να μην χρησιμοποιήσετε καθόλου global μεταβλητές στο πρόγραμμά σας.

Χρησιμοποιήστε τις συναρτήσεις getchar()/putchar() για character I/O στη C.

Χρησιμοποιήστε defines για σταθερές του προγράμματος που εμφανίζονται πολύ συχνά ή που ενδεχομένως θα θέλατε να πειραματιστείτε αλλάζοντάς τες (π.χ. σύμβολο τόνου, διαλυτικά, κλπ).

Χρησιμοποιήστε έναν enum τύπο για τις καταστάσεις που θα χρειαστείτε.

Ένα αναγνώσιμο πρόγραμμα (περισσότερες λεπτομέρειες στα style guides στη σελίδα των Links):

- Χρησιμοποιεί ένα συνεπές και καλό σχήμα για indentation του κώδικα. Όλες οι εντολές που είναι nested σε blocks if, switc, while, for, do/while εντολών πρέπει να είναι indented. Τα περισσότερα προγράμματα χρησιμοποιούν 3 ή 4 κενά για κάθε επίπεδο indentation. Ο emacs μπορεί αυτόματα να κάνει indent το πρόγραμμά σας με την χρήση του πλήκτρου tab.
- Περιέχει περιγραφικά ονόματα. Τα ονόματα μεταβλητών, σταθερών, τύπων, και συναρτήσεων πρέπει να δείχνουν το σκοπό ύπαρξής τους.
- Περιέχει προσεκτικά γραμμένα σχόλια. Ξεκινήστε κάθε program file με ένα σχόλιο που περιέχει: το όνομά σας, το αριθμό της άσκησης, και το όνομα του αρχείου. Κάθε συνάρτηση – ειδικά η main, πρέπει να ξεκινά με ένα σχόλιο που περιγράφει τι κάνει η συνάρτηση όταν τρέχει, αναφέροντας καθαρά τον ρόλο των παραμέτρων της συνάρτησης και των τιμών που επιστρέφει. Επίσης, το σχόλιο πρέπει να αναφέρει τι διαβάζει η συνάρτηση από το standard input (αν διαβάζει κάτι) και τι γράφει στα standard output και error (αν γράφει κάτι), ή σε οποιοδήποτε άλλο stream.

Δυο απλά παραδείγματα είναι: hello.c f2c.c

Προσοχή στις δηλώσεις των μεταβλητών ώστε να έχουν τον κατάλληλο τύπο για τον σκοπό που προορίζονται. Αν έχετε error ή warning κατά το compilation του προγράμματος σας, μην πειραματίζεστε με αλλαγές στον κώδικά σας. Προσπαθήστε να καταλάβετε ακριβώς τι συμβαίνει (πιθανώς ανατρέχοντας σε manual και man pages) και να κάνετε αλλαγές/διορθώσεις με συγκεκριμένο στόχο.

Αν έχετε error ή warning κατά το compilation του προγράμματος σας, μην πειραματίζεστε με αλλαγές στον κώδικά σας. Προσπαθήστε να καταλάβετε ακριβώς τι συμβαίνει (πιθανώς ανατρέχοντας σε manual και man pages) και να κάνετε αλλαγές/διορθώσεις με συγκεκριμένο στόχο.

Logistics

Βήμα 1: Fork Repository
Κάντε fork το repository από την ομάδα του μαθήματος στο csd gitlab. Στη συνέχεια αλλάξτε τα permissions σε private όπως αναγράφει στα policies.

Προσθέστε ως members στο repo σας τους TAs του μαθήματος.

Βήμα 2: Γράψε τον source code
Γράψτε το πρόγραμμα σας στα μηχανήματα, αρχιτεκτονικής x86, του τμήματος με λειτουργικό σύστημα GNU/Linux (portokali, milo, rodakino, karpouzi, etc.) χρησιμοποιώντας τον gcc και τον αγαπημένο σας κειμενογράφο (emacs/vim/nano) (περισσότερες λεπτομέρειες στα reference cards των Tutorials).

Για αυτή την άσκηση μπορείτε να βάλετε όλο τον κώδικά σας στο αρχείο translate.c που βρίσκεται κάτα από τον φάκελο src. Δεν χρειάζεται να τον χωρίσετε σε περισσότερα αρχεία. Σε επόμενες ασκήσεις θα γράψετε προγράμματα που θα αποτελούνται από περισσότερα αρχεία.

Περιορίστε το μέγεθος των γραμμών (πλάτος) στο αρχείο σας σε 78 ή 80 χαρακτήρες. Αυτό σας επιτρέπει να τυπώνετε σε δύο στήλες σε χαρτί και να έχετε ταυτόχρονα ανοιχτά παράθυρα για editing και compilation και execution.

Βήμα 3: Preprocess, Compile, Assemble, and Link
Χρησιμοποιήστε τον gcc με τις command line παραμέτρους

```
gcc -Wall -ansi -pedantic
```

για να κάνετε preprocess, compile, assemble, και link το πρόγραμμά σας:

```
$ gcc -Wall -ansi -pedantic -o translate translate.c
```

Για δική σας ευκολία μπορείτε να τρέξετε την εντολή make translate . Προσοχή για να τρέξετε αυτή την εντολή θα πρέπει να είστε στον ίδιο φάκελο που βρίσκεται και το Makefile

Βήμα 4: Execute
Εκτελέστε το πρόγραμμά σας πολλαπλές φορές με διάφορα input files και δοκιμάστε όλα τα λογικά paths εκτέλεσης του κώδικά σας. Μπορείτε να χρησιμοποιήσετε τα test

- Input: test1.7 (iso8859-7) Output: test1.1 (iso8859-1)
- Input: test2.7 (iso8859-7) Output: test2.1 (iso8859-1)

αλλά θα πρέπει να κάνετε και άλλα δικά σας test.

Για δική σας ευκολία μπορείτε να τρέξετε όλα τα test με την εντολή:

```
make tests
```

Προσοχή για να τρέξετε αυτή την εντολή θα πρέπει να είστε στον ίδιο φάκελο που βρίσκεται και το Makefile.

Η εντολή (UNIX filter) `od` δείχνει σε διάφορες μορφές (ανάλογα με τις παραμέτρους της) τους χαρακτήρες που υπάρχουν στο input. Η εντολή

```
od -t uC
```

θα σας φανεί πολύ χρήσιμη ώστε να βλέπεται τι περιέχει το κάθε αρχείο που εξετάζετε. Κάντε

```
man od
```

για περισσότερες λεπτομέρειες. Π.χ,

```
$ cat test.7 | od -t uC
$ cat test.7 | translate | od -t uC
```

θα σας δείξει τους κωδικού για κάθε χαρακτήρα που υπάρχει στο αρχείο test.7 ή στο output του προγράμματος σας.

Βήμα 5: Readme file
Χρησιμοποιήστε τον αγαπημένο σας κειμενογράφο (emacs/vim/nano) για να δημιουργήσετε ένα ?readme? text file που περιέχει:

- Το όνομά σας
- Πράγματα που χειρίζεστε με διαφορετικό τρόπο από ότι ορίζει η άσκηση.
- Μια περιγραφή της βοήθειας που είχατε από άλλους στη δημιουργία του προγράμματος σας, και σε συμφωνία με το Policies section του web page του μαθήματος.
- (Προαιρετικά) Μία ένδειξη του πόσο χρόνο αφιερώσατε για την άσκηση.
- (Προαιρετικά) Οτιδήποτε άλλο θέλετε να αναφέρετε.

Σχόλια που περιγράφουν τον κώδικά σας **δεν** πρέπει να υπάρχουν στο readme file. Πρέπει να τα ενσωματώσετε στο κατάλληλο σημείο του προγράμματος σας.

Βήμα 5: Υποβολή
Η παράδοση της άσκησής σας θα γίνει μέσω git, σύμφωνα με τις οδηγίες που περιγράφονται στο policies. Συγκεκριμένα το repository σας στο gitlab θα πρέπει να είναι fork του repository assignment1 και θα πρέπει να προσθέσετε ως members τους TAs του μαθήματος. Σιγουρευτείτε ότι ο κώδικάς σας έχει γίνει σωστά commit και ότι φαίνονται στο online repository στο account σας στο csd-gitlab. Όταν όλα είναι έτοιμα και έχετε τελειώσει με την άσκησή σας βάλτε το tag assignment1 χρησιμοποιώντας την εντολή:

```
git add tag assignment1
```

και κάντε upload το tag χρησιμοποιώντας την εντολή:

```
git push origin --tags
```

Προσοχή! Μην κάνετε commit object και executables αρχεία.

Επειδή η εξέταση θα γίνει στα μηχανήματα του Τμήματος θα πρέπει να κάνετε clone το repository στα μηχανήματα του Τμήματος και να κάνετε compile και run τις ασκήσεις σας σε αυτά τα συστήματα. Αυτή είναι η ίδια διαδικασία που θα ακολουθηθεί την ημέρα της εξέτασης.

Στην προθεσμία της παράδοσης ένα script θα τρέξει και θα κατεβάσει όλα τα repositories που έχουν γίνει fork. Αυτά είναι τα repositories που θα βαθμολογηθούν.

Tips: Προσπαθήστε να οργανώσετε τον χρόνο σας ώστε να διεκπεραιώσετε τις ασκήσεις στην ώρα τους, χωρίς άγχος. Προσπαθήστε να βάλετε στις ασκήσεις πάθος και δημιουργικότητα, ώστε ο χρόνος που αφιερώνετε στην κάθε άσκηση να είναι όσο το δυνατόν πιο δημιουργικός, χαρούμενος και διδακτικός. Με λίγα λόγια... do it with meraki.

Bonus
Υλοποιήστε την άσκηση σας με την χρήση πίνακα συναρτήσεων. Η άσκηση σας θα πρέπει να καλεί "αυτόματα" την κατάλληλη συνάρτηση από τον πίνακα, βάσει του εκάστοτε state του state machine

Βαθμολογία
Η βαθμολογία θα βασιστεί και στην ορθότητα αλλά και στο σχεδιασμό, όπως αναφέρεται στη σελίδα Policies του μαθήματος. Η κατανόηση της άσκησης αλλά και η αναγνωσιμότητα ενός προγράμματος είναι σημαντικό μέρος του σχεδιασμού.

Last Modified: 12-02-2021 02:00