ΕΙΣΑΓΩΓΗ ΣΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ (2005-06)

Άσκηση 5

Γράψτε ένα πρόγραμμα C (έστω ότι το πηγαίο αρχείο του ονομάζεται "`words.c`") το οποίο να διαβάζει ένα κείμενο από την είσοδο, να απομονώνει τις λέξεις που περιέχει και να τις οργανώνει σ' ένα ταξινομημένο δυαδικό δέντρο. Σαν λέξη στο κείμενο να θεωρείται οποιαδήποτε ακολουθία χαρακτήρων βρίσκεται μεταξύ χαρακτήρων που είναι διαχωριστικά λέξεων και δεν περιλαμβάνει η ίδια τέτοια διαχωριστικά. Κλασικά διαχωριστικά λέξεων είναι τα λευκά διαστήματα (κενό, στηλογνώμονας, αλλαγή γραμμής), τα σημεία στίξης στη φυσική γλώσσα, όπως η τελεία (`.`), το κόμμα (`,`), αλλά μπορούμε να θεωρήσουμε και άλλους ειδικούς χαρακτήρες των υπολογιστών, όπως `@`, `#`, κλπ.

Αφού διαβάσει το πρόγραμμά σας όλο το κείμενο να εκτυπώσει την πιο συχνά εμφανιζόμενη λέξη, καθώς και το πλήθος των εμφανίσεών της. Αν υπάρχουν περισσότερες της μίας λέξεις με το ίδιο πλήθος πιο συχνών εμφανίσεων, να εκτυπωθεί εκείνη που προηγείται αλφαβητικά. Επίσης να εκτυπώσει όλες τις λέξεις του κειμένου σε αύξουσα αλφαβητική σειρά, το πλήθος των εμφανίσεων κάθε λέξης στο κείμενο, καθώς και τους αύξοντες αριθμούς γραμμών στις οποίες εμφανίζονται αυτές.

Κάποια παραδείγματα εκτέλεσης είναι τα εξής: [1]

```
% cat test_words.txt
Hello there! How are you?
Hello!!! I am fine. What about you? Are you OK?
Yes, I am fine. Thank you.
%
% ./words < test_words.txt
Most frequent word is "you" (4 occurrences)

Word: Are
Occurrences: 1
Lines: 2

Word: Hello
Occurrences: 2
Lines: 1 2

Word: How
Occurrences: 1
Lines: 1

Word: I
Occurrences: 2
Lines: 2 3

Word: OK
Occurrences: 1
Lines: 2
```

[1] Τα διαχωριστικά λέξεων που χρησιμοποιήθηκαν ήταν οι χαρακτήρες `‘`, `~`, `!`, `@`, `#`, `$`, `%`, `^`, `&`, `*`, `(`, `)`, `-`, `=`, `+`, `[`, `]`, `{`, `}`, `;`, `:`, `’`, `"`, `\`, `|`, `<`, `>`, `,`, `.`, `/`, `?`, δηλαδή όλοι οι ειδικοί χαρακτήρες στον υπολογιστή εκτός από το `_`, καθώς και τα λευκά διαστήματα ␣, `\t` και `\n`.

```
Word: Thank
Occurrences: 1
Lines: 3

Word: What
Occurrences: 1
Lines: 2

Word: Yes
Occurrences: 1
Lines: 3

Word: about
Occurrences: 1
Lines: 2

Word: am
Occurrences: 2
Lines: 2 3

Word: are
Occurrences: 1
Lines: 1

Word: fine
Occurrences: 2
Lines: 2 3

Word: there
Occurrences: 1
Lines: 1

Word: you
Occurrences: 4
Lines: 1 2 2 3
%
%
% cat KRexcerpt.txt
C is a general-purpose programming language. It has been closely
associated with the UNIX system where it was developed, since both
the system and most of the programs that run on it are written in
C. The language, however, is not tied to any one operating system
or machine; and although it has been called a "system programming
language" because it useful for writing compilers and operating
systems, it has been used equally well to write major programs in
many different domains.
%
% ./words < KRexcerpt.txt
```

```
Most frequent word is "it" (5 occurrences)

Word: C
Occurrences: 2
Lines: 1 4

Word: It
Occurrences: 1
Lines: 1

.............................................

Word: been
Occurrences: 3
Lines: 1 5 7

.............................................

Word: it
Occurrences: 5
Lines: 2 3 5 6 7

Word: language
Occurrences: 3
Lines: 1 4 6

.............................................

Word: writing
Occurrences: 1
Lines: 6

Word: written
Occurrences: 1
Lines: 3
%
%
% ./words < large_file.txt
Most frequent word is "the" (18417 occurrences)

Word: 0
Occurrences: 204
Lines: 6 156 199 202 210 377 803 ................

.............................................

Word: ALLOCATE
Occurrences: 12
Lines: 5330 5990 5996 5999 6008 6066 6074 13400 13400 28959 28973 29109
```

```
.................................................

Word: Ignoring
Occurrences: 4
Lines: 39101 39125 39151 39159

.................................................

Word: _blob
Occurrences: 1
Lines: 39212

.................................................

Word: debugging
Occurrences: 6
Lines: 13046 13049 21623 27578 27631 27762

.................................................

Word: the
Occurrences: 18417
Lines: 16 16 16 16 18 18 18 19 22 22 23 23 24 30 30 37 37 39 ..........

.................................................

Word: zeroth
Occurrences: 1
Lines: 20159
%
% ./words < large_file.txt | wc -w
  332777
%
%
% ./words < labyrinth.c
Most frequent word is "width" (32 occurrences)

Word: 0
Occurrences: 10
Lines: 7 8 44 44 60 71 122 131 132 141

.................................................

Word: CreateMaze
Occurrences: 3
Lines: 25 53 63

.................................................
```

```
Word: atoi
Occurrences: 5
Lines: 35 37 38 39 43

.............................................

Word: if
Occurrences: 20
Lines: 34 36 38 42 44 49 55 72 74 76 78 80 105 116 124 133 .............

.............................................

Word: printf
Occurrences: 6
Lines: 56 158 161 163 166 168

.............................................

Word: while
Occurrences: 2
Lines: 70 107

Word: width
Occurrences: 32
Lines: 30 35 44 45 48 53 54 57 63 67 69 72 72 74 76 76 .................
%
```

Τα αρχεία `test_words.txt`, `KRexcerpt.txt` και `large_file.txt` μπορείτε να τα βρείτε μέσα στο

`http://www.di.uoa.gr/~ip/hwprogs/wordsfiles.zip`

αλλά και σαν απλά αρχεία κειμένου στην ίδια θέση.

Για την άσκηση αυτή, θα παραδώσετε τουλάχιστον δύο αρχεία πηγαίου κώδικα C, το βασικό `words.c` καθώς και αρχείο (ή αρχεία) με βοηθητικές συναρτήσεις, για παράδειγμα για διαχείριση λιστών και δέντρων, και τουλάχιστον ένα αρχείο επικεφαλίδας. Όλα τα αρχεία που έχετε να παραδώσετε, μαζί πιθανώς και με ένα αρχείο `README.txt` με οδηγίες για το πώς θα πρέπει να κατασκευάσει κανείς το εκτελέσιμο πρόγραμμα, θα τα πακετάρετε σε ένα `zip` αρχείο, έστω με όνομα `words.zip`, το οποίο και θα υποβάλετε. Για την κατασκευή ενός αρχείου `zip` στο Unix, εκτελείτε την εντολή `zip <zipfile> <file1> <file2> ... <fileN>`, οπότε τα αρχεία `<file1>`, `<file2>` ... `<fileN>` πακετάρονται (και συμπιέζονται) μέσα στο `zip` αρχείο `<zipfile>`. Παράδειγμα:

```
% zip words.zip words.c blabla.c foofoo.c barbar.h README.txt
  adding: words.c (deflated ..............
.......................................
%
```

Αρχεία `zip` μπορεί να δημιουργήσει κανείς και στα Windows, για παράδειγμα με το πρόγραμμα `winzip`.

Οι οδηγίες υποβολής της άσκησης θα ανακοινωθούν έγκαιρα, μαζί με αυτές και της τελευταίας άσκησης που θα ακολουθήσει (έχει ανακοινωθεί άτυπα ήδη, πρόκειται για την υλοποίηση των προγραμμάτων `pgmread` και `pgmwrite` της Άσκησης 3).