



## ΕΡΓΑΣΤΗΡΙΟ 2

### Μέρος Α': Τεκμηρίωση με χρήση Doxygen

Για να εξασφαλιστεί ότι ο πηγαίος κώδικας σας θα έχει επαρκή τεκμηρίωση, το μάθημα ΕΠΛ232 θα απαιτεί την χρησιμοποίηση του λογισμικού *doxygen*, μια εφαρμογή για την μετατροπή των σχολίων ενός προγράμματος C (και όχι μόνο), σε HTML αρχεία τα οποία μπορούμε να τα δούμε χρησιμοποιώντας ένα browser.

#### Παράδειγμα:

Στο παράδειγμα φαίνεται πως μπορούμε να προσθέσουμε σχόλια στο κώδικα ώστε μέσω του *doxygen* να παράγουμε το *documentation* για το πρόγραμμα. Πιο κάτω βλέπουμε τα σχόλια που βάζουμε σε μια συνάρτηση:

```
/** @brief Prints the string s, starting at the current
 *      location of the cursor.
 *
 * If the string is longer than the current line, the
 * string should fill up the current line and then
 * continue on the next line. If the string exceeds
 * available space on the entire console, the screen
 * should scroll up one line, and then the string should
 * continue on the new line. If '\n', '\r', and '\b' are
 * encountered within the string, they should be handled
 * as per putbyte. If len is not a positive integer or s
 * is null, the function has no effect.
 *
 * @param s The string to be printed.
 * @param len The length of the string s.
 * @return Void.
 */
void putbytes(const char* s, int len);
```

1. Κατεβάστε το αρχείο lab2.files.zip από την ιστοσελίδα του μαθήματος. Το αρχείο αυτό περιλαμβάνει:
  - a. ένα πρόγραμμα C, με το όνομα functions.c
  - b. Το αρχείο README.dox
  - c. Ένα αρχείο lab2.conf, που είναι απαραίτητο για την παραγωγή σχολίων σχετικά με το πρόγραμμα functions.c. Το αρχείο lab2.conf παράγεται με την εντολή:  
doxygen -g lab2.confΌλα τα πιο πάνω αρχεία πρέπει να βρίσκονται στο ίδιο κατάλογο.
2. Ανοίξτε το αρχείο README.dox και προσθέστε γενικά σχόλια σχετικά με το πρόγραμμα σας. Το περιεχόμενο αυτού του αρχείου θα εμφανίζεται στο main page.
3. Μην αλλάζετε τις παραμέτρους του αρχείου lab2.conf, το έχουμε κάνει εμείς για εσάς. Οι σημαντικότερες παράμετροι είναι:



- a. `INPUT` = εδώ βάζετε τα ονόματα των αρχείων π.χ `functions.c` `README.dox` (δείτε το στο αρχείο `lab2.conf`)
  - b. `EXTRACT_ALL` = YES Extract documentation even from those elements you haven't yet commented.
  - c. `OPTIMIZE_OUTPUT_FOR_C` = YES
  - d. `JAVADOC_AUTOBRIEF` = YES
  - e. `INLINE_SOURCE` = YES Extract the relevant parts of the source and associate them with your description.
  - f. `GENERATE_LATEX` = NO Skip generating LaTeX sources for PDF.
  - g. `CALL_GRAPH` = NO Όταν θα γράψετε κώδικα με πολλά αρχεία η παράμετρος αυτή να είναι `CALL_GRAPH` = YES ούτως ώστε να δημιουργηθεί και ο γράφος εξάρτησης μεταξύ των αρχείων.
4. Στον πίνακα υπάρχουν πληροφορίες σχετικά με την χρησιμοποίηση οδηγιών του Doxygen.

Doxygen tag	Εμφάνιση στο html αρχείο
<b>@file</b> <i>file-name</i>	Prints <i>file-name</i> as header of page.
<b>@brief</b> <i>Brief description goes here.</i>	Description appears in HTML document generated.
<b>@author</b> <i>name1</i> <b>@author</b> <i>name2</i>	<b>Author:</b> <i>name1</i> <i>name2</i>
<b>@version</b> <i>Version number</i>	<b>Version:</b> <i>Version number()</i>
<b>@see</b> <i>FunctionA()</i>	<b>See also:</b> <i>FunctionA()</i>
<b>@param</b> <i>variableA DescriptionA</i> <b>@param</b> <i>variableB DescriptionB</i> <b>@param</b> <i>variableC DescriptionC</i>	<b>Parameters:</b> <i>variableA DescriptionA</i> <i>variableB DescriptionB</i> <i>variableC DescriptionC</i>
<b>@return</b> <i>variableA</i>	<b>Returns:</b> <i>variableA</i>
<b>@bug</b> <i>Description of bug goes here</i>	<b>Bug:</b> <i>Description of bug goes here</i>

5. Δίνεται επίσης το αρχείο `functionsWithComments.txt` το οποίο προκύπτει όταν στο αρχείο `functions.txt` προστεθούν τα κατάλληλα σχόλια. Όταν τελειώσει η προσθήκη σχολίων στο αρχείο πηγαίου κώδικα, εκτελείται η εντολή: `doxygen lab2.conf`
6. Στο τρέχον κατάλογο δημιουργείται ένας νέος κατάλογος `/html`
7. Εμφανίστε τα σχόλια με την εντολή: `firefox html/index.html`

**Μερικές διαφορές Doxygen με Javadoc**

- Το Javadoc είναι εργαλείο τεκμηρίωσης μόνο για την γλώσσα Java. Από την άλλη, το Doxygen υποστηρίζει τεκμηρίωση εκτός από την Java και σε άλλες γλώσσες όπως C++, C, Java, Objective-C, Python, IDL (Corba and Microsoft flavors), Fortran, VHDL, PHP, C#.
- Το Javadoc προϋποθέτει την γνώση ετικετών HTML (tags) π.χ. <p>, <ul>, <li>, <i> οι οποίες είθισται να περιλαμβάνονται στα σχόλια του κώδικα για σκοπούς μορφοποίησης του κειμένου. Με το Doxygen, τα σχόλια μέσα στον κώδικα είναι πολύ πιο συνοπτικά και απαλλαγμένα από προσμίξεις ετικετών, χωρίς να υπάρχει η ανάγκη για χρήση HTML.
- Το Doxygen επιτρέπει τη δημιουργία διαφόρων τύπων διαγραμμάτων για οπτική παρακολούθηση των εμπλεκόμενων αρχείων και του κώδικα π.χ. διάγραμμα συσχέτισης/εξάρτησης αρχείων.
- Το Doxygen παρέχει μια δομημένη εμφάνιση του πηγαίου κώδικα μέσω διαγραμμάτων, παραπομπών μέσα στον κώδικα, και επισήμανσης κώδικα (syntax highlighted code).
- Το Doxygen επιτρέπει την εξαγωγή διαφορετικών αρχείων εξόδου όπως π.χ. HTML, PDF, LATEX, man pages, κτλ.) εν αντιθέσει με το Javadoc παρέχει μόνο HTML.
- Οι πιο πάνω δυνατότητες του Doxygen είναι δεδομένες ακόμη και αν ο κώδικας δεν έχει καθόλου σχόλια.

**Μέρος Β': Ανάγνωση αριθμών από αρχείο με χρήση συναρτήσεων**

1. Γράψτε ένα πρόγραμμα στο οποίο η συνάρτηση `main()` θα καλεί μια συνάρτηση `readNumbers()` η οποία θα διαβάζει 20 ακέραιους αριθμούς από ένα αρχείο `data.txt` και να τους αποθηκεύει σε ένα πίνακα. Ο πίνακας θα είναι αρχικά δηλωμένος μέσα στη συνάρτηση `main()`. Στη συνέχεια θα καλείται από τη `main()` η συνάρτηση `largestNumber()` που θα βρίσκει και θα επιστρέφει τον μεγαλύτερο αριθμό μέσα από τον πίνακα. Και τέλος θα καλείται η συνάρτηση `printAll()` που θα τυπώνει το περιεχόμενο του πίνακα και τον μεγαλύτερο αριθμό σ' ένα αρχείο εξόδου με όνομα `output.txt`.

Πρέπει να γίνει χρήση των εντολών `fopen()`, `fclose()`, `fscanf()` και `fprintf()`.

**Σημείωση:** Προσθέστε στο πρόγραμμα τα απαραίτητα σχόλια και τρέξτε το εργαλείο *doxygen* για να δείτε τα αποτελέσματα.

Παράδειγμα εκτέλεσης:

Μετά την εκτέλεση του προγράμματος, το αρχείο `output.txt` θα περιέχει τα εξής:

```
The numbers stored in data.txt are:
```

```
86
15
23
19
46
82
11
67
17
33
71
39
6
87
72
44
27
16
74
1
```

```
The largest number in data.txt is 87
```

**Μέρος Γ': Ανάγνωση συμβολοσειρών από αρχείο και επεξεργασία με χρήση της βιβλιοθήκης `string.h`****Strings**

- Τα strings στην γλώσσα C είναι πίνακες από χαρακτήρες.
- Χρησιμοποιούνται για να αποθηκεύσουν λέξεις ή προτάσεις (κυρίως) στην Αγγλική γλώσσα.
- Κάποιες γλώσσες προγραμματισμού έχουν τύπο δεδομένων string (βλέπε Java, C++), αλλά ή C δεν έχει.
- Το τέλος ενός string στην γλώσσα C καθορίζεται από τον χαρακτήρα `\0` ο οποίος καλείται τερματικός χαρακτήρας NULL.
- Το πρόγραμμα πρέπει να προβλέπει χώρο αποθήκευσης του τερματικού χαρακτήρα NULL. Για παράδειγμα, ο πίνακας χαρακτήρων που θα αποθηκεύσει το string "Hello" πρέπει να έχει μέγεθος τουλάχιστον 6 bytes: `char hello[6];`

**String functions in `string.h`**

Η γλώσσα C περιέχει ένα σύνολο από συναρτήσεις που μπορούν να χρησιμοποιηθούν για να διαχειριστούμε strings.

- Για να χρησιμοποιήσουμε τις συναρτήσεις αυτές, κάνουμε `include` την βιβλιοθήκη `string.h`: `#include <string.h>`
- `int strlen(char str[])` επιστρέφει το μήκος ενός string χωρίς να λαμβάνεται υπόψιν ο τερματικός χαρακτήρας NULL  
Παράδειγμα:  

```
char string[10] = "john";  
printf("%d\n", strlen(string)); // θα τυπώσει 4
```
- `strcat(char str1[], char str2[])` συνενώνει το `str2` στο τέλος του `str1`  
Παράδειγμα:  

```
char string[15] = "john";  
strcat(string, "smith");  
printf("%d\n", strlen(string)); // θα τυπώσει 9  
printf("%s\n", string); // θα τυπώσει johnsmith
```
- `strcpy(char str1[], char str2[])` αντιγράφει το `str2` στο `str1`, ξεκινώντας από την πρώτη θέση του `str1`  
Παράδειγμα:  

```
char word[10] = "hello";  
strcpy(word, "world");  
printf("%s\n", word); // θα τυπώσει world
```



- `int strcmp(char str1[], char str2[])` συγκρίνει το `str1` με το `str2` και επιστρέφει 0 αν είναι του ίδιου μήκους και έχουν ακριβώς τους ίδιους χαρακτήρες σε όλες τις θέσεις. Επιστρέφει ένα αριθμό  $< 0$  αν το `str1 < str2` και ένα αριθμό  $> 0$  αν `str1 > str2`.

Παράδειγμα:

```
char word[10] = "hello";
printf("%d\n", strcmp(word, "hello")); // θα τυπώσει 0
printf("%d\n", strcmp(word, "world")); // αρνητικός αριθμός
printf("%d\n", strcmp(word, "friend")); // θετικός αριθμός
```

- `strtok(char str1[], char delim[])` διαχωρίζει το `str1` σε μια ακολουθία από μηδέν ή περισσότερα non-empty tokens. Ο/οι χαρακτήρας/ες διαχωρισμού (μπορεί να είναι πάνω από ένας) δίνονται στον πίνακα `delim`. Στην πρώτη κλήση της συνάρτησης δίνεται το `str1` που θα διαχωριστεί σε tokens. Σε κάθε επόμενη κλήση της `strtok`, το πρώτο όρισμα θα πρέπει να είναι NULL.

Παράδειγμα:

```
char str[80] = "This is-EPL232 : website";
char s[] = " -:";
char *token;
/* get the first token */
token = strtok(str, s);
/* walk through other tokens */
while( token != NULL ) {
    printf( "%s\n", token );
    token = strtok(NULL, s);
}
```

Αποτέλεσμα εκτέλεσης:

```
This
is
EPL232
website
```

1. Γράψτε ένα πρόγραμμα `textAnalyzer.c` το οποίο ζητά από τον χρήστη να δώσει ένα αρχείο εισόδου που περιέχει κείμενο. Το πρόγραμμα θα διαβάζει το αρχείο εισόδου, θα επεξεργάζεται το κείμενο και θα εκτυπώνει σε άλλο αρχείο εξόδου κάποια στατιστικά στοιχεία. Δίνεται το αρχείο εισόδου `file.txt`.

Παράδειγμα εκτέλεσης (με έντονο κόκκινο χρώμα η είσοδος του χρήστη):

```
Please enter the name of the input file.
Filename: file.txt
Please enter the name of the output file.
Filename: outfile.txt
Processing complete.
```



Το αρχείο εξόδου outfile.txt θα πρέπει να περιέχει τα εξής:

Statistics for file: file.txt

-----

Total # of characters in file: 417

Letters	297	71.22 %
White space	77	18.47 %
Digits	17	4.08 %
Other characters	26	6.24 %

Total # of words in file: 82

The words of file.txt in alphabetical order:

100  
123rd  
12:45  
13  
15  
17  
2  
And  
Didn't  
He  
How  
I'll  
PM  
The  
Well  
When  
and  
anything  
anyways?  
at  
better  
brown  
cats  
chasing  
checked  
darn  
do  
dog  
dog  
dog  
dogs  
dreamed



far  
fox  
fox  
foxes  
had  
have  
he  
he  
he  
he  
he?  
his  
his  
his  
jumped  
jumped  
jumping  
just  
lazy  
lazy  
lazy  
lazy  
leaped  
mailmen  
more  
of  
of  
over  
over  
over  
pretty  
quick  
rabbits  
sheep  
so  
speaking  
tell  
the  
the  
the  
time?  
to  
was  
was  
was  
watch  
who





why  
with  
you

**Μέρος Δ': Ασκήσεις για βρόχους και συναρτήσεις**

2. Γράψτε ένα πρόγραμμα `multiple3.c` το οποίο ζητά από τον χρήστη δυο αριθμούς, και βρίσκει και να τυπώνει όλους τους αριθμούς μεταξύ των δυο εισαγόμενων αριθμών (συμπεριλαμβανομένων) που διαιρούνται με 3.

**Σημείωση:** Προσθέστε στο πρόγραμμα τα απαραίτητα σχόλια και τρέξτε το εργαλείο *doxygen* για να δείτε τα αποτελέσματα.

3. Γράψτε ένα πρόγραμμα `losange.c` το οποίο ζητά από τον χρήστη έναν ακέραιο αριθμό  $n$ , και εμφανίζει έναν ρόμβο μεγέθους  $n$ . Π.χ. για  $n=5$  το πρόγραμμα εμφανίζει:

```
  *
 * *
*   *
* * *
* * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
```

4. Γράψτε ένα πρόγραμμα `primeNumbers.c` το οποίο ζητά από τον χρήστη έναν αριθμό  $n$ , και εμφανίζει όλους τους πρώτους αριθμούς μικρότερους του  $n$ .
5. Γράψτε ένα πρόγραμμα `triangle.c` το οποίο ζητά από τον χρήστη έναν αριθμό  $N$  και εμφανίζει ένα τρίγωνο με  $N$  γραμμές. Π.χ. για  $N = 8$  το πρόγραμμα εμφανίζει:

```
  *
 ***
*****
*****
*****
*****
*****
*****
*****
*****
*****
```

Οι πιο κάτω ασκήσεις να υλοποιηθούν με χρήση συναρτήσεων όπως καθορίζει η εκφώνηση της κάθε άσκησης.

6. Γράψτε ένα πρόγραμμα που διαβάζει ακέραιους αριθμούς από το χρήστη αριθμούς (το πολύ 10), και τους ταξινομεί σε αύξουσα σειρά χρησιμοποιώντας ένα πίνακα (μεγέθους



10) και τον αλγόριθμο selection sort. Οι αριθμοί θα εισάγονται από το χρήστη μέσω της συνάρτησης main. Η είσοδος θα τερματίζεται όταν ο χρήστης δώσει το 0. Η ταξινόμηση να γίνεται από την συνάρτηση selectionSort. Η εκτύπωση των ταξινομημένων αριθμών να γίνεται και πάλι μέσω της main.

7. Ένας ακέραιος αριθμός θεωρείται τέλειος εάν ισούται με το άθροισμα των διαιρετών του. Γράψτε ένα πρόγραμμα, το οποίο ζητά έναν αριθμό από τον χρήστη, και τυπώνει όλους τους τέλειους αριθμούς μέχρι αυτόν τον αριθμό. Το πρόγραμμα να χρησιμοποιεί μια συνάρτηση perfect η οποία λαμβάνει σαν είσοδο ένα ακέραιο αριθμό και να επιστρέφει 1 εάν ο δοθέντας αριθμός είναι τέλειος, 0 εάν δεν είναι.
8. Γράψτε ένα πρόγραμμα το οποίο αφού διαβάσει 2 αριθμούς, πρέπει να ελέγχει εάν και οι 2 αριθμοί είναι θετικοί (μεγαλύτεροι του 0). Σ' αυτή την περίπτωση, να διαιρεί το μικρότερο από το μεγαλύτερο και να τυπώνει το αποτέλεσμα. Σ' αντίθετη περίπτωση να τυπώνει ανάλογο μήνυμα (να αναφέρει ποιος αριθμός είναι ο θετικός και ποιος όχι) και το πρόγραμμα να τερματίζει. Η εισαγωγή των 2 αριθμών να γίνεται στη συνάρτηση main και όλα τα υπόλοιπα στη συνάρτηση div().
9. Γράψτε ένα πρόγραμμα christmasTree.c το οποίο εμφανίζει ένα Χριστουγεννιάτικο δέντρο όπως στην παρακάτω εικόνα:

```
  *
 ***
  *
 ***
*****
  *
 ***
*****
*****
  *
```

Το πρόγραμμα να χρησιμοποιεί μια συνάρτηση, triangle(int n), που τυπώνει ένα τρίγωνο παίρνοντας σαν παράμετρο τον αριθμό των γραμμών π.χ. για  $n = 3$  η συνάρτηση τυπώνει:

```
  *
 ***
*****
```

