Λειτουργικά Συστήματα – 1η Εργαστηριακή Άσκηση

Ομάδα εργαστηρίου: **oslabb30**

Φοιτητές: **Γεώργιος Λαγός 03117034, Παναγιώτης Ζευγολατάκος 03117804**

Στην αναφορά χρησιμοποιείται παντού Α' πληθυντικός γιατί τόσο η αναφορά όσο και η ίδια η εργασία έγιναν με πλήρη επικοινωνία μεταξύ των φοιτητών.

Άσκηση 1.1

Βήματα:

1) Για να γίνει αντιγραφή των αρχείων zing.h και zing.o στον κατάλογο εργασίας μας, πλοηγήθηκαμε στον φάκελο /home/oslab και έπειτα εκτελέσαμε την εντολή:
   cp –r code/zing oslabb30/giorgos
   cp –r code/zing oslabb30/panos
   που αντιγράφει τον φάκελο zing με τα περιεχόμενα του (zing.h και zing.o) στον φάκελο "giorgos" και "panos" αντίστοιχα.
2) Προκειμένου να δημιουργηθεί το main.o πρέπει αρχικά να δημιουργηθεί το main.c, το οποίο γίνεται μέσω της εντολής:
   vim main.c
   στο οποίο γράψαμε τον ακόλουθο κώδικα C:

```
#include <stdio.h>

#include "zing.h"

int main () {

      zing();

      return 0;

}
```

   Εν συνεχεία, δημιουργήσαμε το ζητούμενο object file main.o με την ακόλουθη εντολή:
   gcc –Wall –c main.c
3) Το linking των 2 object files έγινε με την ακόλουθη εντολή:
   gcc -o zing zing.o main.o

και το αποτέλεσμα ήταν η δημιουργία εκτελέσιμου (executable) αρχείου με όνομα “zing”. Τρέχοντας το αρχείο αυτό με την εντολή ./zing λάβαμε ως έξοδο στην κονσόλα το μήνυμα:
Hello, oslabb30

Ερωτήσεις:

1) Γνωρίζουμε πως με τα header files μπορούμε να ορίζουμε συναρτήσεις, οι οποίες χρησιμοποιούνται σε (πιθανώς περισσότερα από ένα) προγράμματα. Κάθε πρόγραμμα που τις χρειάζεται απλά πρέπει στην αρχή του κώδικα να κάνει “include” το header file που περιλαμβάνει την συνάρτηση που επιθυμεί. Πιο συγκεκριμένα, στην δική μας εργαστηριακή άσκηση το header file (zing.h) ορίζει την συνάρτηση zing().
2) Κατασκευάσαμε το ζητούμενο Makefile μέσω της εντολής:
   vim Makefile
   και σε αυτό γράψαμε τον ακόλουθο κώδικα:

```
zing: zing.o main.o
	gcc -o zing zing.o main.o
main.o: main.c
	gcc -Wall -c main.c
```

   τρέχοντας το με το command “Make” λάβαμε στην κονσόλα το αποτέλεσμα:
   ‘zing’ is up to date.
3) Με το command “vim zing2.c” δημιουργήσαμε το zing2, στο οποίο γράψαμε τον ακόλουθο κώδικα:

```
#include <stdio.h>

void zing () {
	printf("Hello and welcome to OS!\n");
}
```

Επεξεργαζόμαστε το Makefile που προηγουμένως δημιουργήσαμε, ώστε να επιτελεί τον στόχο που έθεσε η εκφώνηση. Μάλιστα, στην πρώτη του σειρά προσθέτουμε κατάλληλη εντολή, προκειμένου να δύναται να εκτελέσει και τα δύο εκτελέσιμα αρχεία:

```
all: zing zing2

zing: zing.o main.o
```

```
	gcc -o zing zing.o main.o

zing2: zing2.o main.o

	gcc -o zing2 zing2.o main.o
```

4) Για να αντιμετωπισθεί ο χρόνος μεταγλώττισης πρέπει αντί να γράψουμε και τις 500 συναρτήσεις στο ίδιο αρχείο να προτιμήσουμε να γράψουμε πολλά αρχεία με λίγες συναρτήσεις το καθένα και μετά να κάνουμε link τα object files τους. Με αυτήν την προσέγγιση η τροποποίηση μιας μόνο συνάρτησης δεν θα επιφέρει εκ νέου μεταγλώττιση όλων των συναρτήσεων παρά μόνο εκείνων που περιέχονται στο ίδιο αρχείο, και άρα υπάρχει σημαντική εξοικονόμηση χρόνου.
5) Θεωρήσαμε ότι ο καλύτερος τρόπος να εξηγήσουμε τι συνέβη είναι φυσικά να το τρέξουμε οι ίδιοι. Έτσι, φτιάχνοντας έναν βασικό πηγαίο κώδικα που απλά τύπωνε κάτι στην οθόνη, εκτελέσαμε την ακόλουθη 'προβληματική' εντολή:
   gcc -Wall -o foo.c foo.c
   Εύκολα παρατηρήσαμε πως δημιουργήθηκε εκτελέσιμο αρχείο με όνομα foo.c, το οποίο μάλιστα έτρεξε σωστά (μια απλή εκτύπωση ήταν). Ωστόσο, ο πηγαίος κώδικας χάθηκε και πλέον δεν θα μπορούσε κάποιος να τον τροποποιήσει.

Άσκηση 1.2

Για την επίλυση της ζητούμενης άσκησης βασιστήκαμε πλήρως στον προτεινόμενο σκελετό υλοποιήσης. Χωρίσαμε τον πηγαίο κώδικα μας σε δύο κομμάτια αποκλειστικά και μόνο για δική μας διευκόλυνση. Το κύριο κομμάτι γράφτηκε στο αρχείο fconc.c και το τμήμα που περιείχε τις βοηθητικές συναρτήσεις αποτέλεσε το header file fconc.h. Ακόμη, αντί να υποθέσουμε έναν αρκετά μεγάλο buffer, αποφασίσαμε να υπολογίσουμε ακριβώς το μέγεθος του, μιας και αυτό είναι το βέλτιστο για οικονομία χώρου. Προκειμένου να το καταφέρουμε αυτό κάναμε include κατάλληλη βιβλιοθήκη και χρησιμοποιήσαμε την stat. Επομένως, ο πηγαίος κώδικας που χρησιμοποιήθηκε ήταν ο ακόλουθος:

```c
#include <stdio.h>
#include <stdbool.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>
#include "fconc.h"
#include <string.h>

int main (int argc, char** argv) {
        if (argc<3 || argc>4) printf("Usage: ./fconc infile1 infile2 [outfile (default:fconc.out)]\n");
        int fd1,fd2,fd3;
        bool output_file=false;
        if (argc==4) output_file=true;
        if (argc==3 || argc==4) {
                fd1=open(argv[1],O_RDONLY);
                if (fd1<0) {
                        perror(argv[1]);
                        exit(1);
                }
                close(fd1);

                fd2=open(argv[2],O_RDONLY);
                if (fd2<0) {
                        perror(argv[2]);
                        exit(1);
                }
                close(fd2);
                if (output_file) {
                        if (strcmp(argv[2],argv[3]) != 0) {

                                fd3=open(argv[3],O_CREAT,S_IRUSR | S_IWUSR);
                                close (fd3);
                                fd3=open(argv[3],O_WRONLY);
                                write_file(fd3,argv[1]);
                                write_file(fd3,argv[2]);
                                if (close(fd3)<0) {

                                        perror("close");
                                        exit(1);
                                }
                        }
                        else    {

                                fd3=open("fconc.out",O_WRONLY);
                                write_file(fd3,argv[1]);
                                write_file(fd3,argv[2]);
                                fd2=open(argv[3],O_WRONLY);
                                if (close(fd3)<0) {
                                        perror("close");
                                        exit(1);
                                }
                                write_file(fd2,"fconc.out");
"fconc.c" 75L, 1497C
                                }
                        }
                        else {
                                fd3=open("fconc.out",O_WRONLY);
                                write_file(fd3,argv[1]);
                                write_file(fd3,argv[2]);
                                if (close(fd3)<0) {
                                        perror("close");
                                        exit(1);
                                }

                        }
                }
        }
}
```

```c
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>

void doWrite (int fd,const char* buff,int len) {
        ssize_t ret;
        ret=write(fd,buff,len);
        if (ret<0) {
                perror("write");
                exit(1);
        }


}

void write_file (int fd, const char *infile) {
        int fd_1=open(infile, O_RDONLY);
        struct stat st;
        stat(infile, &st);
        long size = st.st_size;

        ssize_t ret;
        char *buff;
        buff=malloc(st.st_size);

        ret=read(fd_1,buff,size);

        if (ret<0) {
                perror("read");
                exit (1);
        }

        doWrite(fd,buff,size);
                if (close(fd_1)<0) {
                        perror("close");
                        exit(1);
                }

}
```

Ερωτήση:

```
oslabb30@orion:~/giorgos/zing/OutputFileExercise1_2$ echo 'Users Giorgos and Panos' > A
oslabb30@orion:~/giorgos/zing/OutputFileExercise1_2$ echo 'Succesfully completed the task'> B
oslabb30@orion:~/giorgos/zing/OutputFileExercise1_2$ strace ./fconc A B C
execve("./fconc", ["./fconc", "A", "B", "C"], [/* 29 vars */]) = 0
brk(0)                                  = 0xfd0000
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f9e45a7f000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
open("/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=29766, ...}) = 0
mmap(NULL, 29766, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f9e45a77000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\34\2\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0755, st_size=1738176, ...}) = 0
mmap(NULL, 3844640, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f9e454b6000
mprotect(0x7f9e45657000, 2097152, PROT_NONE) = 0
mmap(0x7f9e45857000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1a1000) = 0x7f9e45857000
mmap(0x7f9e4585d000, 14880, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f9e4585d000
close(3)                                = 0
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f9e45a76000
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f9e45a75000
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f9e45a74000
arch_prctl(ARCH_SET_FS, 0x7f9e45a75700) = 0
mprotect(0x7f9e45857000, 16384, PROT_READ) = 0
mprotect(0x7f9e45a81000, 4096, PROT_READ) = 0
munmap(0x7f9e45a77000, 29766)           = 0
open("A", O_RDONLY)                     = 3
close(3)                                = 0
open("B", O_RDONLY)                     = 3
close(3)                                = 0
open("C", O_RDONLY|O_CREAT, 0600)       = 3
close(3)                                = 0
open("C", O_WRONLY)                     = 3
open("A", O_RDONLY)                     = 4
stat("A", {st_mode=S_IFREG|0644, st_size=24, ...}) = 0
brk(0)                                  = 0xfd0000
brk(0xff1000)                           = 0xff1000
read(4, "Users Giorgos and Panos\n", 24) = 24
write(3, "Users Giorgos and Panos\n", 24) = 24
close(4)                                = 0
open("B", O_RDONLY)                     = 4
stat("B", {st_mode=S_IFREG|0644, st_size=31, ...}) = 0
read(4, "Succesfully completed the task\n", 31) = 31
write(3, "Succesfully completed the task\n", 31) = 31
close(4)                                = 0
close(3)                                = 0
exit_group(0)                           = ?
+++ exited with 0 +++
oslabb30@orion:~/giorgos/zing/OutputFileExercise1_2$ cat C
Users Giorgos and Panos
Succesfully completed the task
```