ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ

1η ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΝΑΦΟΡΑ

Γεωργία Μπουσμπουκέα- el19059

Παναγιώτης Μιχελάκης- el19119

Oslaba59

1.1

Κώδικας:

"main.c" :

```
#include "zing.h"
int main(int argc, char **argv){
        zing();
        return 0;
}
```

"zing.h":

```
#ifndef ZING_H__
#define ZING_H__

void zing(void);
void zing2(void);
#endif
```

1. Με τον ορισμό συναρτήσεων και κλάσεων (και macros) σε διαφορετικά αρχεία αυξάνεται η ευελιξία του κώδικα. Συγκεκριμένα, μια συνάρτηση μπορεί να χρησιμοποιηθεί σε διάφορα αρχεία (reusable), χωρίς να απαιτεί αντιγραφή του σώματός της, αρκεί να συμπεριληφθεί το αντίστοιχο αρχείο κεφαλίδας (περιέχει την δήλωση της συνάρτησης). Αυτό συντελεί και στην μείωση του μεγέθους του κώδικα. Ακόμη, αν θέλουμε να αλλάξουμε κάτι στην συνάρτηση, αρκεί να μεταβάλλουμε μόνο το αρχείο όπου ορίζεται. Το αρχείο κεφαλίδας ουσιαστικά ενημερώνει τον compiler για την ύπαρξη της συνάρτησης που χρησιμοποιούμε, για να μην πετάξει λάθος και για την τοποθεσία της, για να γίνει το linking.

2.

"Makefile":

```
all: zing zing2 zing2.o main.o

zing: zing.o main.o

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

zing2: zing2.o main.o

gcc -o zing2 zing2.o main.o

zing2.o: zing2.c

gcc -Wall -c zing2.c

main.o: main.c

gcc -Wall -c main.c

3.Ορίζουμε μια νέα συνάρτηση zing () στο αρχείο zing2.c και την δηλώνουμε στο header file. Στο main.c καλείται μία φορά η zing. Κατά την μεταγλώττιση γίνεται linking του main.o αρχικά με το zing.o για την δημιουργία του εκτελέσιμου zing και στη συνέχεια με το zing2.o για το zing2.

"zing2.c":

```
#include <stdio.h>
#include<unistd.h>

void zing(void){
                printf("Hello friends, %s\n",getlogin());
}
```

Output:

```
Hello friends, oslaba59
```

4.Για να μειωθεί ο χρόνος μεταγλώττισης μπορούμε να ορίσουμε τις συναρτήσεις σε διαφορετικά αρχεία, συμπεριλαμβάνοντας τα αρχεία κεφαλίδας τους στο αρχείο του προγράμματος μας. Κάνοντας έτσι την μεταγλώττιση με χρήση Makefile κάθε φορά θα μεταγλωττίζεται μόνο η συνάρτηση που μεταβλήθηκε.

5. Η εντολή `gcc –Wall –o foo.c foo.c` είχε ως αποτέλεσμα την μεταγλώττιση του c αρχείου foo.c για την δημιουργία του εκτελέσιμου με όνομα foo.c (το flag –o προσδιορίζει το όνομα). Καθώς ταυτίζονται τα ονόματα, γίνεται overwrite και ο αρχικός κώδικας χάνεται.

Σημείωση: Το –Wall θα εμφανίσει στο shell όλα τα πιθανά warnings.

1.2.

Κώδικας:

"fconc.c":

```c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>

int doWrite(int fd, const char *buff){
        size_t len, idx;
        len=strlen(buff);
        ssize_t wcnt;
        idx = 0;

        do{
                wcnt = write(fd,buff + idx, len - idx);
                if (wcnt == -1){ /* error */
                perror("write");
                return 1;
                }
                idx += wcnt;
        } while (idx < len);
        return 0;
}



int write_file(int fd, const char *infile){
        int fdvar;
        char buff[1024];
        ssize_t rcnt;
        fdvar = open(infile, O_RDONLY);
        if (fdvar ==-1){
                perror(infile);
                exit(1);
        }
        for (;;){
                rcnt = read(fdvar,buff,sizeof(buff)-1);
                if (rcnt == 0) {/* end-of-file */
                        return 0;
                }
```

```
                if (rcnt == -1){ /* error */
                        perror("read");
                        return 1;
                }
                doWrite(fd, buff);
        }

        buff[rcnt] = '\0';
        close(fdvar);
}


int main(int argc, char **argv){
        if(argc<3||argc>4) printf("Usage: ./fconc infile1 infile2
[outfile (default:fconc.out)]\n");
        int fd, oflags, mode;
        oflags = O_CREAT | O_WRONLY | O_TRUNC;
        mode = S_IRUSR | S_IWUSR;
        if(argc==4)  fd = open(argv[3], oflags, mode);
        else  fd= open("fconc.out", oflags, mode);
        if (fd == -1){
        perror("open");
        exit(1);
        }
        write_file(fd, argv[1]);
        write_file(fd, argv[2]);
        close(fd);
        return 0;

}
```

Output:

```
Goodbye,

and thanks for all the fish!
```

Η strace εμφανίζει διάφορες πληροφορίες για τις κλήσεις συστήματος που γίνονται στο πρόγραμμά μας:

```
strace ./fconc A B C
```

Output:

execve("./fconc", ["./fconc", "A", "B", "C"], [/* 20 vars */]) = 0

brk(0)   = 0x15a7000

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) = 0x7f0bf33be000

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) = 0x7f0bf33b6000

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) = 0x7f0bf2df5000

mprotect(0x7f0bf2f96000, 2097152, PROT_NONE) = 0

mmap(0x7f0bf3196000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1a1000) = 0x7f0bf3196000

mmap(0x7f0bf319c000, 14880, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f0bf319c000

close(3)   = 0

mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f0bf33b5000

mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f0bf33b4000

mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f0bf33b3000

arch_prctl(ARCH_SET_FS, 0x7f0bf33b4700) = 0

mprotect(0x7f0bf3196000, 16384, PROT_READ) = 0

mprotect(0x7f0bf33c0000, 4096, PROT_READ) = 0

munmap(0x7f0bf33b6000, 29766)   = 0

open("C", O_WRONLY|O_CREAT|O_TRUNC, 0600) = 3

```
open("A", O_RDONLY)                    = 4
read(4, "Goodbye, \n", 1023)           = 10
write(3, "Goodbye, \n", 10)            = 10
read(4, "", 1023)                      = 0
open("B", O_RDONLY)                    = 5
read(5, "and thanks for all the fish!\n", 1023) = 29
write(3, "and thanks for all the fish!\n\177", 30) = 30
read(5, "", 1023)                      = 0
close(3)                               = 0
exit_group(0)                          = ?
+++ exited with 0 +++
```