ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ
ΥΠΟΛΟΓΙΣΤΩΝ - ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Λειτουργικά Συστήματα

Ομάδα 118:

Ρέππας Ευστράτιος (03120002) – Τζουρμανά Ελευθερία (03119927)

Άσκηση 1: Εισαγωγή στο περιβάλλον προγραμματισμού

1.1 Σύνδεση με αρχείο αντικειμένων:

- **Πηγαίος κώδικας (source code) της άσκησης:**

  - main.c:

```c
//main.c

#include<stdio.h>
#include "zing.h"

int main(int argc, char **argv){
          zing();
      return 0;
}
```

  - zing2.c:

```c
//zing2.c

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

```c
void zing(){

   char *os = getlogin(); //Η getlogin() επιστρέφει ένα
                          //string που περιέχει το όνομα του user
   printf("Welcome, %s\n", os);
}
```

- **Διαδικασία μεταγλώττισης και σύνδεσης:**

Η διαδικασία μεταγλώττισης και σύνδεσης ώστε να δημιουργήσουμε το εκτελέσιμο zing και το zing2 είναι η εξής:

```
//Makefile

all: zing zing2

zing: main.o
    gcc -o zing zing.o main.o

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

zing2: zing2.o main.o
    gcc -o zing2 zing2.o main.o

zing2.o: zing2.c
    gcc -Wall -c zing2.c
```

Παρακάτω φαίνεται επίσης και η έξοδος εκτέλεσής τους:

```
oslab118@orion:~$ cd ask1.1
oslab118@orion:~/ask1.1$ ./zing
Hello, oslab118
oslab118@orion:~/ask1.1$ ./zing2
Welcome, oslab118
```

- Ερωτήσεις:

1. *Ποιο σκοπό εξυπηρετεί η επικεφαλίδα;*

   Η χρήση αρχείων επικεφαλίδας επιτρέπει την διεπαφή προς άλλα κομμάτια κώδικα τα οποία βρίσκονται σε άλλα αρχεία. Αυτό που περιέχει μια επικεφαλίδα είναι η δήλωση μιας συνάρτησης και μιας ή περισσοτέρων καθολικών μεταβλητών. Συγκεκριμένα, όταν καλούμε μια συνάρτηση από διαφορετικό αρχείο .c, πρέπει να συμπεριλάβουμε το κατάλληλο αρχείο επικεφαλίδας που να περιέχει τις απαραίτητες δηλώσεις συναρτήσεων. Η επικεφαλίδα #include “header.h” επεξεργάζεται από τον preprocessor ο οποίος διαβάζει την συγκεκριμένη επικεφαλίδα και την αντικαθιστά την οδηγία με τα περιεχόμενα του αρχείου αυτού. Επιπλέον, στις επικεφαλίδες μπορεί να δηλώνονται βιβλιοθήκες της c.

2. *Ζητείται κατάλληλο Makefile για τη δημιουργία του εκτελέσιμου της άσκησης.*

Χρησιμοποιούμε το Makefile για την αυτόματη δημιουργία προγραμμάτων από αρχεία κώδικα. Το τελικό Makefile φαίνεται παρακάτω:

```
//Makefile

all: zing

zing: main.o
    gcc -o zing zing.o main.o

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

3. *Παράξτε το δικό σας zing2.o, το οποίο θα περιέχει zing() που θα εμφανίζει διαφορετικό αλλά παρόμοιο μήνυμα με τη zing() του* zing.o.

Η υλοποίηση του φαίνεται παραπάνω.

4. *Έστω ότι έχετε γράψει το πρόγραμμά σας σε ένα αρχείο που περιέχει 500 συναρτήσεις. Αυτή τη στιγμή κάνετε αλλαγές μόνο σε μία συνάρτηση. Ο κύκλος εργασίας είναι: αλλαγές στον κώδικα, μεταγλώττιση, εκτέλεση, αλλαγές στον κώδικα, κ.ο.κ. Ο χρόνος μεταγλώττισης είναι μεγάλος, γεγονός που σας καθυστερεί. Πώς μπορεί να αντιμετωπισθεί το πρόβλημα αυτό;*

Ο χρόνος μεταγλώττισης είναι μεγάλος καθώς οι συναρτήσεις μας βρίσκονται όλες στο ίδιο .c file. Αυτό συνεπάγεται ότι με κάθε αλλαγή σε μια μόνο συνάρτηση γίνεται μεταγλώττιση και εκτέλεση όλου του κώδικα. Επομένως, προκειμένου να αντιμετωπιστεί αυτό το πρόβλημα μπορούμε να διαχωρίσουμε την υλοποίηση των συναρτήσεων σε διαφορετικά αρχεία .c. Με αυτόν τον τρόπο κάθε αλλαγή σε συνάρτηση που υπάρχει σε ένα αρχείο .c θα χρειάζεται μεταγλώττιση μόνο του συγκεκριμένου αρχείου που την περιλαμβάνει. Επιπλέον, χρήσιμη είναι και η υλοποίηση Makefile για την αυτόματη δημιουργία προγραμμάτων από αρχεία κώδικα.

5. *Ο συνεργάτης σας και εσείς δουλεύατε στο πρόγραμμα foo.c όλη την προηγούμενη εβδομάδα. Καθώς κάνατε ένα διάλειμμα και ο συνεργάτης σας δούλευε στον κώδικα, ακούτε μια απελπισμένη κραυγή. Ρωτάτε τι συνέβη και ο συνεργάτης σας λέει ότι το αρχείο foo.c χάθηκε! Κοιτάτε το history του φλοιού και η τελευταία εντολή ήταν η: gcc -Wall -o foo.c foo.c. Τι συνέβη;*

Η εντολή gcc -Wall -o foo.c foo.c χρησιμοποιείται για να μεταγλωττίσει το αρχείο προέλευσης foo.c και να παράγει ένα εκτελέσιμο αρχείο με το ίδιο όνομα, δηλαδή foo.c. Το -o καθορίζει το όνομα του αρχείου εξόδου. Σε αυτή την περίπτωση χρησιμοποιήθηκε το ίδιο όνομα αρχείου και για το αρχείο εισόδου και για το αρχείο εξόδου. Επομένως, έγινε overwrite, δηλαδή αντικαταστάθηκε το περιεχόμενο του αρχείου foo.c με το νέο μεταγλωττισμένο που έχει το ίδιο όνομα.

1.2 Συνένωση δύο αρχείων σε τρίτο:

Ακολουθεί ο κώδικας:

```
//fconc.c

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

int main(int argc, char **argv){
    int i, fd;
    if (argc!=-1) {

//Για την περίπτωση κάποιου σφάλματος στο διάβασμα του αρχείου

        if ((argc > 4) || (argc < 3)) {
            printf("Usage: ./fconc infile1 infile2 [outfile
(                               default:fconc.out)]\n");

//Ελεγχος για σωστή κλήση της συνάρτησης και εκτύπωση του
//κατάλληλου μηνύματος
            return 1;
        }

        for (i = 1; i < 3; i++) {
            if (access(argv[i], F_OK) != 0) {

// δεν υπάρχει κάποιο αρχείο που δόθηκε ως όρισμα και τυπώνουμε το
//κατάλληλο μήνυμα σφάλματος

                printf("No such file or directory\n");
                return 1;
            }
        }

        if (argc < 4)
            argv[3] = "fconc.out";

//Σε περίπτωση που δεν έχει δοθεί όρισμα για το αρχείο εξόδου,
//ορίζουμε την default value του

        fd = open(argv[3], O_CREAT | O_WRONLY |O_TRUNC, 0644);

//Θέτουμε ένα file descriptor για το αρχείο εξόδου στο οποίο θα
γράψουμε (O_WRONLY),
//το οποίο το δημιουργούμε εάν δεν υπάρχει (O_CREAT) και γράφουμε από
//πάνω εάν υπάρχει ήδη (O_TRUNC). Το 0644 δίνει τα κατάλληλα
//permissions για άνοιγμα και εγγραφή του αρχείου

        for (i = 1; i < 3; i++) {

//καλούμε για το κάθε όρισμα τη write_file
```

```
                write_file(fd, argv[i]);
        }

        close(fd);

//Κλείνουμε τον file descriptor για να μην έχουμε resource leaks ή
//άλλα θέματα.
    }
    else {
        printf("An error occured\n");


//εκτύπωση κατάλληλου μηνύματος σφάλματος

        return 1;
    }
    return 0;
}
```

```
//write_file.c

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

void write_file(int fd, const char *infile) {
   char buff[1024];
   int fd1;
   ssize_t rcnt;
   fd1 = open(infile, O_RDONLY);

//Κάνουμε κλήση συστήματος για //άνοιγμα του infile για ανάγνωση
//(O_RDONLY) δίνοντας του έναν file descriptor

   for (;;) {
      rcnt = read(fd1, buff, sizeof(buff) - 1);

// Κάνουμε κλήση συστήματος για την ανάγνωση του infile. Η
//κλήση συστήματος διαβάζει μέχρι και
//sizeof(buff-1) χαρακτήρες από το αρχείο και τους γράφει στο
//buff. Επιστρέφει πόσα bytes έγραψε
// και τους γράφουμε στη μεταβλητή rcnt του κατάλληλου τύπου. Η
//επανάληψη εξασφαλίζει ότι θα διαβαστεί όλο το αρχείο. Κάθε φορά
//που ξαναγίνεται η κλήση συστήματος, αυτή θα ξαναγράψει στο
//buffer, εξασφαλίζοντας ότι όλο το αρχείο
//θα διαβαστεί ακόμα και αν είναι μεγαλύτερο του μεγέθους του buff.
//Ακόμα και αν υπάρχουν στο buff και άλλοι χαρακτήρες δεν θα τους
γράψει

      if (rcnt == 0) {
         break;
      }

//Οταν φτάνουμε στο τέλος του αρχείου, οπότε η read έγραψε 0 bytes
//στο buffer, τελειώνουμε την επανάληψη
```

```
      if (rcnt == -1) {

//Για κάποιο σφάλμα όταν γίνεται ανάγνωση, τυπώνουμε κατάλληλο
//μήνυμα σφάλματος

         perror("read");
         close (fd1);

//Κλείνουμε τον file descriptor για να μην έχουμε resource leaks ή
//άλλα θέματα.

         exit(1);
      }

      doWrite(fd, buff, rcnt);

//Καλούμε την doWrite()
   }
   close(fd1);

//Κλείνουμε τον file descriptor για να μην έχουμε resource leaks ή
//άλλα θέματα.

   return;
}
```

```
//doWrite.c

#include <sys/types.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>

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

//ορίζουμε κατάλληλου τύπου μεταβλητή η χρήση της οποίας εξηγείται
//στη συνέχεια

   ssize_t wcnt;

//ορίζουμε κατάλληλου τύπου μεταβλητή για να μετράμε πόσα
//bytes γράφει σε κάθε κλήση της η write

   do {
      wcnt = write(fd, buff + idx, len - idx);

//Η κλήση συστήματος write() γράφει ξεκινώντας απο την idx
//διεύθυνση στο buffer, μέχρι και len-idx του αρχείου.
//Έτσι διασφαλίζουμε ότι θα γράψει με τη σωστή σειρά όλο το buffer.
//Η write() επιστρέφει πόσα bytes έγραψε στο αρχείο, το οποίο
//αποθηκεύουμε κάθε φορά στη wcnt

      if (wcnt == -1) {

//Για κάποιο σφάλμα όταν γίνεται εγγραφή τυπώνουμε κατάλληλο μήνυμα
//σφάλματος
```

```
            perror("write");
            exit(1);
        }

        idx += wcnt;

//Κάθε φορά που τελειώνουμε μια κλήση
//της write(), φροντίζουμε να ανανεώσουμε κατάλληλα το idx, έτσι
//ώστε να μην ξαναγράψουμε χαρακτήρες

    } while (idx < len);

//Το loop τελειώνει όταν το idx γίνει όσο το len του buff, δηλαδή
//έχουμε γράψει στο αρχείο όλα τα περιεχόμενα του buff
}
```

```
//Makefile

fconc: fconc.o write_file.o doWrite.o
    gcc -o fconc fconc.o write_file.o doWrite.o

fconc.o: fconc.c
    gcc -Wall -c fconc.c

write_file.o: write_file.c
    gcc -Wall -c write_file.c

doWrite.o: doWrite.c
    gcc -Wall -c doWrite.c
```

Παραδείγματα εκτέλεσης, χρησιμοποιώντας δύο αρχεία A, B που περιέχουν τις λέξεις “Hello” & “World” αντίστοιχα:

```
oslab118@orion:~/ask1.1$ cat A
Hello
oslab118@orion:~/ask1.1$ cat B
World
```

```
oslab118@orion:~/ask1.1$ ./fconc A B C D
Usage: ./fconc infile1 infile2 [outfile (default:fconc.out)]
```

```
oslab118@orion:~/ask1.1$ ./fconc A
Usage: ./fconc infile1 infile2 [outfile (default:fconc.out)]
```

```
oslab118@orion:~/ask1.1$ ./fconc A B
oslab118@orion:~/ask1.1$ cat fconc.out
Hello
World
```

```
oslab118@orion:~/ask1.1$ ./fconc A B C
oslab118@orion:~/ask1.1$ cat C
Hello
World
```

Παράδειγμα εκτέλεσης αρχείων μεγαλύτερα από 1kB:

```
oslab118@orion:~/ask1.1$ ./fconc MasterOfPuppets AmericanPie
oslab118@orion:~/ask1.1$ cat fconc.out
End of passion play, crumbling away
I'm your source of self-destruction
Veins that pump with fear, sucking darkest clear
Leading on your death's construction
Taste me, you will see
More is all you need
Dedicated to
How I'm killing you
Come crawling faster (faster)
Obey your master (master)
Your life burns faster (faster)
Obey your master, master
Master of puppets, I'm pulling your strings
Twisting your mind and smashing your dreams
Blinded by me, you can't see a thing
Just call my name 'cause I'll hear you scream
Master, master
Just call my name 'cause I'll hear you scream
Master, master
Needlework the way, never you betray
Life of death becoming clearer
Pain monopoly, ritual misery
Chop your breakfast on a mirror
Taste me, you will see
More is all you need
Dedicated to
How I'm killing you
Come crawling faster (faster)
Obey your master (master)
Your life burns faster (faster)
Obey your master, master
Master of puppets, I'm pulling your strings
Twisting your mind and smashing your dreams
Blinded by me, you can't see a thing
Just call my name 'cause I'll hear you scream
Master, master
Just call my name 'cause I'll hear you scream
Master, master
(Master, master, master, master)
Master, master, where's the dreams that I've been after?
Master, master, you promised only lies
```

Laughter, laughter, all I hear or see is laughter
Laughter, laughter, laughing at my cries
Fix me!
Hell is worth all that, natural habitat
Just a rhyme without a reason
Never-ending maze, drift on numbered days
Now your life is out of season
I will occupy
I will help you die
I will run through you
Now I rule you too
Come crawling faster (faster)
Obey your master (master)
Your life burns faster (faster)
Obey your master, master
Master of puppets, I'm pulling your strings
Twisting your mind and smashing your dreams
Blinded by me, you can't see a thing
Just call my name 'cause I'll hear you scream
Master, master
Just call my name 'cause I'll hear you scream
Master, master

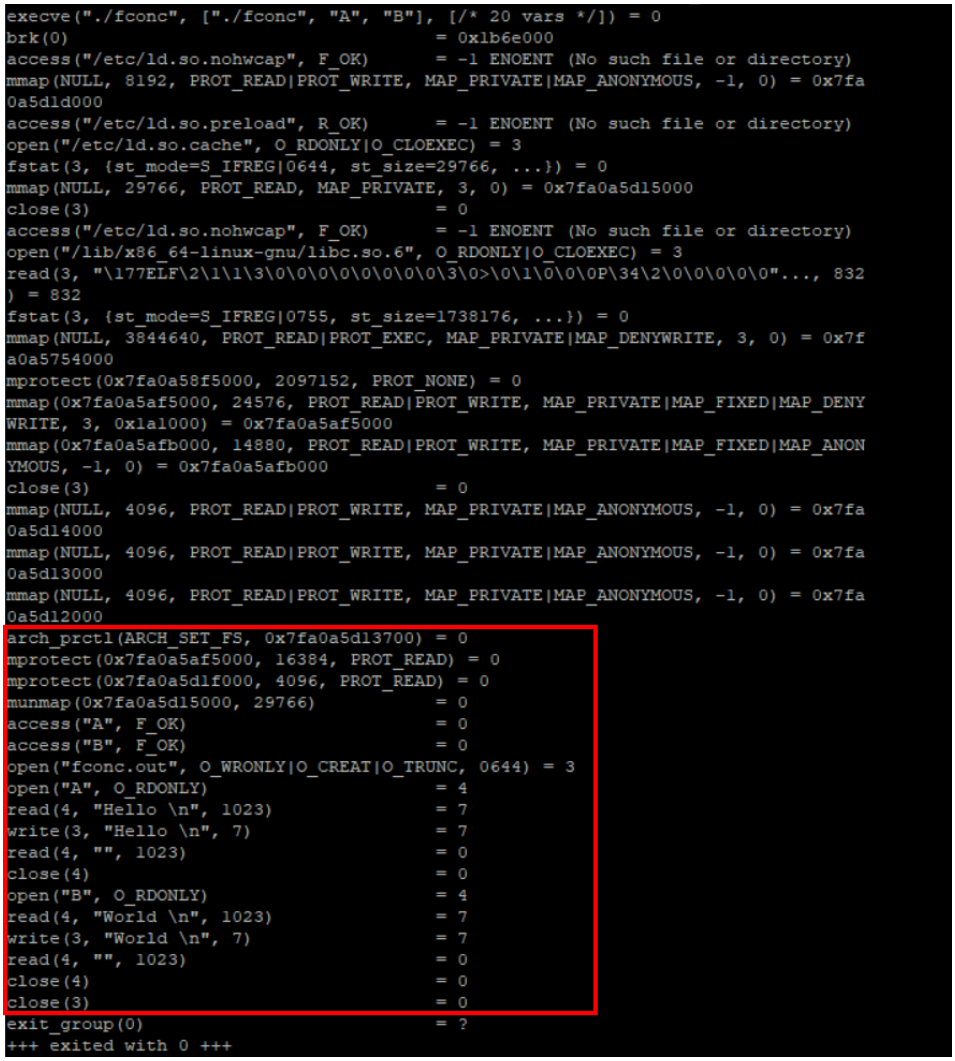
A long long time ago, I can still remember
How that music used to make me smile
And I knew if I had my chance
That I could make those people dance
And maybe they'd be happy for a while
But February made me shiver
With every paper I'd deliver
Bad news on the doorstep
I couldn't take one more step
I can't remember if I cried
When I read about his widowed bride
But something touched me deep inside
The day the music died
So bye-bye, Miss American Pie
Drove my Chevy to the levee
But the levee was dry
Them good old boys were drinking whiskey and rye
Singing, "This'll be the day that I die"
This will be the day that I die
Did you write the Book of Love?
And do you have faith in God above?
If the Bible tells you so
Do you believe in rock 'n' roll?
Can music save your mortal soul?
And can you teach me how to dance real slow?
Well I know that you're in love with him
'Cause I saw you dancing in the gym
You both kicked off your shoes
Then I dig those rhythm and blues
I was a lonely teenage broncin' buck
With a pink carnation and a pickup truck
But I knew I was out of luck
The day the music died
I started singing bye-bye, Miss American Pie
Drove my Chevy to the levee
But the levee was dry
Them good old boys were drinking whiskey and rye

Singing, "This'll be the day that I die"
This will be the day that I die
Now for ten years we've been on our own
And moss grows fat on a rolling stone
But that's not how it used to be
When the jester sang for the King and Queen
In a coat he borrowed from James Dean
And a voice that came from you and me
Oh and while the King was looking down
The jester stole his thorny crown
The courtroom was adjourned
No verdict was returned
And while Lenin read a book of Marx
The Quartet practiced in the park
And we sang dirges in the dark
The day the music died
We were singing, bye-bye Miss American Pie
Drove my Chevy to the levee
But the levee was dry
Them good old boys were drinking whiskey and rye
Singing, "This'll be the day that I die"
This will be the day that I die
Helter skelter in the summer swelter
The birds flew off with a fallout shelter
Eight miles high and falling fast
It landed foul on the grass, the players tried for a forward pass
With the jester on the sidelines in a cast
Now the halftime air was sweet perfume
While the sergeants played a marching tune
We all got up to dance
Oh, but we never got the chance
'Cause the players tried to take the field
The marching band refused to yield
Do you recall what was revealed
The day the music died?
We started singing bye-bye, Miss American Pie
Drove my Chevy to the levee but the levee was dry
Them good old boys were drinking whiskey and rye
And singing, "This'll be the day that I die"
This will be the day that I die
Oh, and there we were all in one place
A generation lost in space
With no time left to start again
So come on, Jack be nimble, Jack be quick
Jack Flash sat on a candlestick
'Cause fire is the devil's only friend
Oh, and as I watched him on the stage
My hands were clenched in fists of rage
No angel born in Hell
Could break that Satan's spell
And as the flames climbed high into the night
To light the sacrificial rite
I saw Satan laughing with delight
The day the music died
He was singing bye-bye, Miss American Pie
Drove my Chevy to the levee but the levee was dry
Them good old boys were drinking whiskey and rye
And singing, "This'll be the day that I die"
This will be the day that I die
I met a girl who sang the blues
And I asked her for some happy news

```
But she just smiled and turned away
I went down to the sacred store
Where I'd heard the music years before
But the man there said the music wouldn't play
And in the streets, the children screamed
The lovers cried and the poets dreamed
But not a word was spoken
The church bells all were broken
And the three men I admire most
The Father, Son, and the Holy Ghost
They caught the last train for the coast
The day the music died
And they were singing bye-bye, Miss American Pie
Drove my Chevy to the levee but the levee was dry
And them good old boys were drinking whiskey and rye
Singing, "This'll be the day that I die"
This will be the day that I die
They were singing bye-bye, Miss American Pie
Drove my Chevy to the levee but the levee was dry
Them good old boys were drinking whiskey and rye
Singing, "This'll be the day that I die"
```

1. *Εκτελούμε ένα παράδειγμα του fconc χρησιμοποιώντας την εντολή strace. Το κομμάτι της εξόδου της strace που προκύπτει από τον κώδικα φαίνεται παρακάτω μαζί με τις κλήσεις συστήματος που χρησιμοποιήθηκαν.*

```
oslab118@orion:~/ask1.1$ strace ./fconc
execve("./fconc", ["./fconc"], [/* 20 vars */]) = 0
brk(0)                                  = 0x19c9000
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) = 0x7f6
c6481a000
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) = 0x7f6c64812000
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) = 0x7f
6c64251000
mprotect(0x7f6c643f2000, 2097152, PROT_NONE) = 0
mmap(0x7f6c645f2000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENY
WRITE, 3, 0x1a1000) = 0x7f6c645f2000
mmap(0x7f6c645f8000, 14880, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANON
YMOUS, -1, 0) = 0x7f6c645f8000
close(3)                                = 0
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f6
c64811000
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f6
c64810000
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f6
c6480f000
arch_prctl(ARCH_SET_FS, 0x7f6c64810700) = 0
mprotect(0x7f6c645f2000, 16384, PROT_READ) = 0
mprotect(0x7f6c6481c000, 4096, PROT_READ) = 0
munmap(0x7f6c64812000, 29766)           = 0
fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 1), ...}) = 0
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f6
c64819000
write(1, "Usage: ./fconc infile1 infile2 ["..., 61Usage: ./fconc infile1 infile2
 [outfile (default:fconc.out)]
) = 61
exit_group(1)                           = ?
+++ exited with 1 +++
oslab118@orion:~/ask1.1$ strace ./fconc A B
```



Η execve σηματοδοτεί την εκτέλεση του προγράμματος. Στις υπόλοιπες γραμμές οι κλήσεις συστήματος που γίνονται φορτώνουν βιβλιοθήκες και προετοιμάζουν το πρόγραμμα (μέχρι και το munmap). Στην συνέχεια γίνονται οι κλήσεις συστήματος που εξηγούνται παρακάτω.

access("A", F_OK)=0: *το πρόγραμμα ελέγχει εάν υπάρχει το αρχείο A με την κλήση συστήματος access.*
access("B", F_OK)=0
open("fconc.out", O_WRONLY|O_CREAT|O_TRUNC, 0644) = 3: *γίνεται open το "fconc.out" με O_WRONLY|O_CREAT|O_TRUNC και 0644 permissions. Η κλήση συστήματος επιστρέφει έναν file descriptor με τιμή 3.*
open("A", O_RDONLY)=4: *ανοίγει το A αρχείο με read-only και επιστρέφει έναν fd με τιμή 4 ο οποίος αντιστοιχεί στο ανοιγμένο αρχείο.*
read(4, "Hello \n", 1023)=7: *διαβάζει από το αρχείο A, 7 bytes δεδομένων (Hello \n) και τα αποθηκεύει σε έναν buffer.*
write(3, "Hello \n", 7)=7: *γράφει το περιεχόμενο του buffer στο "fconc.out" χρησιμοποιώντας την κλήση συστήματος write. Γράφει 7 bytes και επιστρέφει τον αριθμό των bytes που έχουν γραφτεί (7).*
read(4, "", 1023)=0 : *ξαναδιαβάζει το A και επειδή η κλήση συστήματος read είναι 0, έχουμε φτάσει στο τέλος του αρχείου.*

close(4)=0: *κλείνει το αρχείο Α.*
Όμοια για το αρχείο Β.
open(“B”, O_RDONLY)=4
read(4, “World \n”, 1023)=7
write(3, “World \n”, 7)=7
read(4, “”, 1023)=0
close(4)=0
close(3)=0