

## Αναφορά άσκησης bonus: Fread και Read

### Οδηγίες εκτέλεσης:

Για την εκτέλεση των προγραμμάτων χρειάζεται να δώσουμε σαν παραμέτρους (argv[1], argv[2]) το όνομα του αρχείου που θέλουμε να διαβάσει το πρόγραμμα και στη συνέχεια ένα χαρακτήρα 'S' ή 's' για να διαβάσει απο το αρχείο σειριακά, ενώ με 'I' ή 'i' για να διαβάσει με το εναλλάξ μοτίβο.

```
karatziask@karatziask:~/code/oper_sys$ time ./read testfile_1KB.txt s
```

### Fread & Read:

Τα προγράμματα απαιτούν ένα αρχείο και ένα χαρακτήρα από το τερματικό. Σε περίπτωση που δεν τα λάβουν εμφανίζουν στη οθόνη μήνυμα με το αντίστοιχο error. Δημιουργείται μια λίστα απο χαρακτήρες με μέγεθος 1000 bytes. Ανάλογα με την είσοδο που δώσαμε στο τερματικό αν το argv[2] είναι 'S' ή 's' τότε διαβάζει σειριακά 1000 bytes απο το αρχείο με την fread() ή την read() αντίστοιχα και τερματίζει. Ενώ αν στο τερματικό επιλέξουμε να διαβάσει απο το αρχείο εναλλάξ μετακινούμε τον κέρσορα συνεχώς στο σωστό σημείο και διαβάζουμε 1 byte κάθε φορά.

The image shows two side-by-side code editors. The left editor displays the code for 'read.c', which uses the 'read()' system call to read data from a file. The right editor displays the code for 'fread.c', which uses the 'fread()' function to read data from a file. Both programs handle command-line arguments to specify the file name and the reading method (sequential or interleaved).

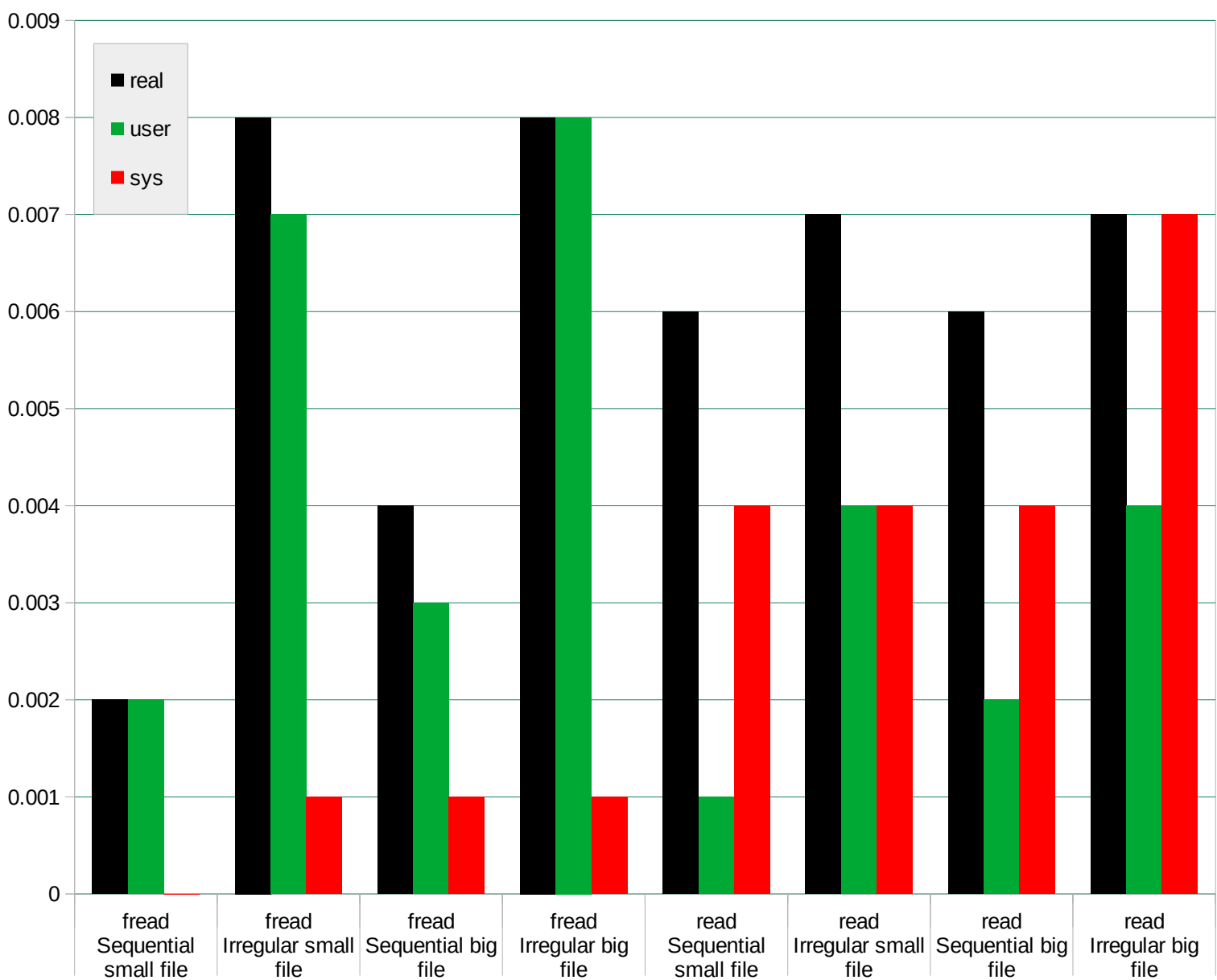
```

C read.c
home > karatziask > code > oper_sys > C read.c > main(int, char**)
1 #include<stdio.h>
2 #include<fcntl.h>
3 #include<unistd.h>
4 #include<stdlib.h>
5
6 int main(int argc, char **argv) {
7     int fd, i=0;
8     ssize_t size = 1000;
9
10    if(argc < 2){
11        printf("Not enough arguments!\n You have to add the \".txt\" file and method!\n");
12    }
13
14    fd = open(argv[1], O_RDONLY);
15
16    if(fd == -1) {
17        printf("Error opening file!\n");
18        exit(0);
19    }
20
21    if(size == -1) {
22        printf("Error reading from file");
23        close(fd);
24        exit(0);
25    }
26
27    int j = 0;
28    char buffer[size];
29    lseek(fd, 0, SEEK_SET);
30    if(argv[2][0] == 's' || argv[2][0] == 'S'){
31        for(int i = 0; i < size; i++){
32            read(fd, &buffer[i], 1);
33        }
34    }else if(argv[2][0] == 'i' || argv[2][0] == 'I'){
35        for(int i = 1; i <= size/2; i++){
36            lseek(fd, i-1, SEEK_SET);
37            read(fd, &buffer[j], 1);
38            lseek(fd, -1-1, SEEK_END);
39            read(fd, &buffer[j+1], 1);
40            j+=2;
41        }
42    }else {
43        printf("Wrong arguments!\n");
44    }
45    close(fd);
46
47    return 0;
48 }

C fread.c
home > karatziask > code > oper_sys > C fread.c > main(int, char**)
1 #include<stdio.h>
2 #include<stdlib.h>
3
4 #define size 1000
5
6 int main (int argc, char **argv){
7
8     if(argc < 2 ){
9         printf("Not enough arguments!\n You have to add the \".txt\" file!\n");
10    }
11
12    FILE *fl = fopen(argv[1], "r");
13    if(fl == NULL){
14        printf("Error opening file!\n");
15        exit(0);
16    }
17
18    char buffer[size];
19    int j = 0;
20
21    rewind(fl);
22    int i=0;
23
24    if(argv[2][0] == 'S' || argv[2][0] == 's'){
25        for(int i=0; i < size; i++){
26            fread(&buffer[i], 1, 1, fl);
27        }
28    }else if(argv[2][0] == 'I' || argv[2][0] == 'i'){
29        for(int i = 1; i <= (size+1)/2; i++){
30            fseek(fl, i-1, SEEK_SET);
31            fread(&buffer[j], 1, 1, fl);
32            fseek(fl, -1-1, SEEK_END);
33            fread(&buffer[j+1], 1, 1, fl);
34            j+=2;
35        }
36    }else {
37        printf("Wrong arguments!\n");
38    }
39
40    fclose(fl);
41 }

```

Γραφική παράσταση χρόνων:



## Τερματικό εκτέλεσης Fread:

```
karatziask@karatziask: ~/code/oper_sys
karatziask@karatziask:~/code/oper_sys$ time ./fread testfile_1KB.txt s
real    0m0,002s
user    0m0,002s
sys     0m0,000s
karatziask@karatziask:~/code/oper_sys$ time ./fread testfile_1KB.txt i
real    0m0,008s
user    0m0,007s
sys     0m0,001s
karatziask@karatziask:~/code/oper_sys$ time ./fread testfile_100MB.txt s
real    0m0,004s
user    0m0,003s
sys     0m0,001s
karatziask@karatziask:~/code/oper_sys$ time ./fread testfile_100MB.txt i
real    0m0,008s
user    0m0,000s
sys     0m0,009s
karatziask@karatziask:~/code/oper_sys$
```

## Τερματικό εκτέλεσης Read:

```
karatziask@karatziask: ~/code/oper_sys
karatziask@karatziask:~/code/oper_sys$ time ./read testfile_1KB.txt s
real    0m0,007s
user    0m0,001s
sys     0m0,006s
karatziask@karatziask:~/code/oper_sys$ time ./read testfile_1KB.txt i
real    0m0,007s
user    0m0,004s
sys     0m0,004s
karatziask@karatziask:~/code/oper_sys$ time ./read testfile_100MB.txt s
real    0m0,003s
user    0m0,003s
sys     0m0,000s
karatziask@karatziask:~/code/oper_sys$ time ./read testfile_100MB.txt i
real    0m0,007s
user    0m0,001s
sys     0m0,007s
karatziask@karatziask:~/code/oper_sys$
```

## **Χρόνοι εκτέλεσης:**

Παρατηρούμε στους χρόνους εκτέλεσης:

- Με την χρήση της fread ο χρόνος που χρειάζεται για να διαβάσει απο το αρχείο εναλλάξ είναι μεγαλύτερος από αυτό που διαβάζει σειριακά ανεξαιρέτως του μεγέθους του αρχείου. Αυτό οφείλεται λόγω των εκτελέσεων των fseek. Επίσης κατά βάση χρησιμοποιείται το user mode.
- Με την χρήση της read παρατηρούμε ότι ο χρόνος εκτέλεσης αφορά κυρίως το kernel mode λόγω της χρήσης system call (read). Ο χρόνος των user mode όταν διαβάζουμε το αρχείο εναλλάξ είναι αυξημένος σε σχέση με την σειριακή αναγνώση λόγω της lseek που χρησιμοποιούμε.
- Γενικά έχουμε μεγαλύτερο χρόνο στην εκτέλεση όταν διαβάζουμε το αρχείο εναλλάξ παρά σειριακά και αυτό οφείλεται στις μετακινήσεις που κάνουμε του κέρσορα. Δεν φάνηκε μεγάλη διαφορά όταν διαβάζαμε μεγάλο η μικρό αρχείο.