Ersatzkurs 2

Lecturers: Mirko Jantschke, Pascal Scholz

29. November 2018

Gliederung

1. Closer look at main

2. File I/O

Closer look at main

Main with arguments

The main function does have arguments

- *argc* which is the the count of the given arguments.
- *argv* which is basically an array with the arguments.

```
int main(int argc, char** agrv) {
    /*Code*/
    return expression;
}
```

The first entry in *argv* is the path of the program.

Example

Running a program *a.c*:

```c
int main(int argc, char** agrv) {
    int a = 0;
    while(a < argc){
        printf("Argument number %d is: %s\n", a, argv[a]);
        ++a;
    }
}
```

will result in the output:

```
    $ ./a.out firstArg secondArg
        Argument number 0 is ./a
        Argument number 1 is firstArg
        Argument number 2 is secondArg
```

Task 1

Write a programm with the shown main function which takes two file names as arguments and and print both.

File I/O

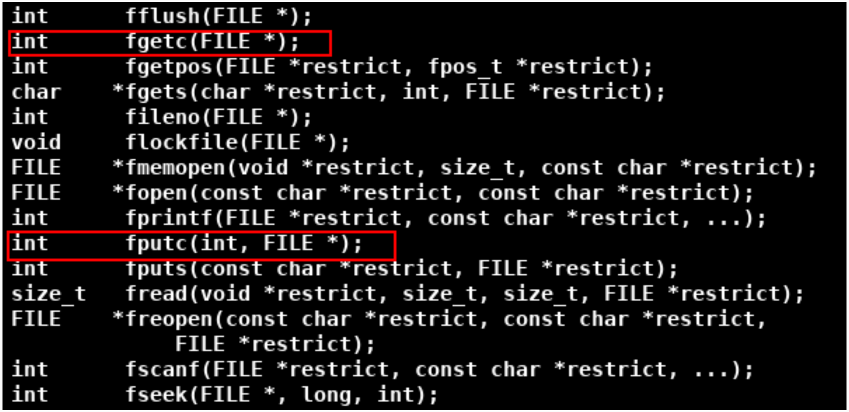
stdio.h with more functions

You already used the header *stdio.h*. It contains much more than *pritnf()* and *scanf()*. With the command

```
    $ man stdio.h
```

you can view the linux man(ual) pages which is the documentation for this files.
You can use any type of online documentation too.

stdio.h with more functions



```
int      fflush(FILE *);
int      fgetc(FILE *);
int      fgetpos(FILE *restrict, fpos_t *restrict);
char    *fgets(char *restrict, int, FILE *restrict);
int      fileno(FILE *);
void     flockfile(FILE *);
FILE    *fmemopen(void *restrict, size_t, const char *restrict);
FILE    *fopen(const char *restrict, const char *restrict);
int      fprintf(FILE *restrict, const char *restrict, ...);
int      fputc(int, FILE *);
int      fputs(const char *restrict, FILE *restrict);
size_t   fread(void *restrict, size_t, size_t, FILE *restrict);
FILE    *freopen(const char *restrict, const char *restrict,
             FILE *restrict);
int      fscanf(FILE *restrict, const char *restrict, ...);
int      fseek(FILE *, long, int);
```

We will use the functions *fegtc* and *fputc*.

How do they work

Both functions take a pointer of type FILE* which they will work with.

fgetc() will return the ASCII number of the read char.

fputc() takes additionally the number which will be written.

Both will return an error code if something wents wrong. So check the returned value!

To open a file use *fopen()* and to close a file after you are done use *fclose()*.

Example

The following example will open a file, read and print a byte:

```c
int main(int argc, char** agrv) {
    FILE *fp;
    int buff;
    char * fname;
    if(argc > 1)
        fname = argv[1];
    //open file
    if( access( fname, F_OK ) != -1 ) {
        fp = fopen(fname, "r");
        if((buff = fgetc((FILE*) fp)) != EOF)
            printf("The char which was read: %c", buff);
    }else{
        printf("Could not open file\n");
        return -1;
    }
    return 0;
}
```

Task

You already got a programm which can handle arguments.

Write a file test.csv with the contents: 4,5,6
Now your programm should except to be called with a file name.
Open this file, read and then print the first line. The ASCII-Code for the line ending used in POSIX Systems is 10.

File format

We define our file format to have the dimensions given in the first line. The lines after th first line are values.