1. Übungsblatt - C++

Henrik Gerdes, Manuel Eversmeyer

4. November 2018

```c
/*
    @author: Henrik Gerdes, Manuel Eversmeyer

    A small programm that reads each character
    form stdin and puts it to stdout while
    replacing ' ' with a new line (\n)

*/
#include <stdio.h>

int main(){

  int c;

  do {
    c=getchar();

    if(c == ' '){
      putchar('\n');
    }else{
      putchar (c);
    };

  } while (c != '\n');

  return 0;

}
```

Aufgaben/Blatt01/Final/break_words.c

```c
/*
    @author: Henrik Gerdes, Manuel Eversmeyer

    Compares two txt Files and prints the first difference
    to the terminal.

    The files are given by commandline arguments.

    For testing reasons the files text1.txt und text2.txt have been created
        to be compaied
*/
#include <stdio.h>
#include <string.h>

```

```c

int main(int argc, char* argv[] ){

    if(argc != 3){
        printf("Unzulaessige Parameter \n");
         return -1;
    }

    FILE *fp1, *fp2;

    fp1 = fopen(argv[1],"r");
    fp2 = fopen(argv[2],"r");


    if(fp1 == NULL || fp2 == NULL){
        printf("Deteien nicht lesbar\n");
        return -1;
    }

    char buffer1[1024], buffer2[1024];


    char *isEnd1[64], *isEnd2[64];
    do{
        //Alternative: if(scanf("%[^\n]"),buffer)
        *isEnd1 = fgets(buffer1, 1024, fp1);
        *isEnd2 = fgets(buffer2, 1024, fp2);


        if(strcmp(buffer1,buffer2)!=0){
            printf("%s", buffer2);
            fclose(fp1);
            fclose(fp2);
            return -1;
        }
    }
    while(isEnd1 != NULL || isEnd2 != NULL);


    fclose(fp1);
    fclose(fp2);


    return 0;
}
```

Aufgaben/Blatt01/Final/first_diff.c

```c
/*
    @author: Henrik Gerdes, Manuel Eversmeyer

    Reads 4 Numbers in a line and 4 lines of a txt File
    and prints it to the terminal with the mean value of each
    line.

    The files are given by commandline arguments.
*/
#include <stdio.h>
```

```c
#include <string.h>
#include <stdlib.h>

void printUse(){
    printf("Bitte Datei als Komandozeilenparameter angeben\n");
}



int main(int argc, char* argv[] ){

    if(argc != 2){
        printf("Unzulaessige Parameter \n");
        printUse;
         return -1;
    }

    FILE *fp;
    fp = fopen(argv[1],"r");


    if(fp == NULL){
        printf("Deteien nicht lesbar\n");
        printUse;
        return -1;
    }
    char buffer1[20];
    char fehler[64];

    int i,l, sum = 0;

    for(l = 0;l<4;l++){
        char fehler[64];
        fehler[0] = '\0';
        for(i = 0; i<4; i++){
            buffer1[0] = '\n';
            fscanf(fp, "%s", &buffer1);
            int num;
            //sscanf(buffer1, "%d", &num);
            num = atoi(buffer1);
            // Rueckgabewert von atoi ist problematisch
            if(num == 0){
                strcpy(fehler,buffer1);
            }
            sum += num;
            printf("%3d", num);
        }
        double mean = sum/4;
        if(fehler[0] != '\0'){
            printf("    Es ist ein Fehler beim Umwandeln von Zeichen %s
                passiert", fehler);
        }
        printf("    Durchschnitt ist: %5.2f\n", mean);
    }

    fclose(fp);
    return 0;
}
```

Aufgaben/Blatt01/Final/line_mean.c

```
Aufgabe 1.2

Wo liegen unter Linux standardmaessig die vorhandenen Header- und Bibliotheksdateien?
    Standardmaessig sind diese unter "usr/include" zu finden, es gibt aber weitere oft genutzte Orte,
    die aber nach Compiler Verion und Konfiguration verschieden seinen koennen (Wie z.b "/usr/local/include")

Welche Unix-Umgebungsvariablen legen die Pfade fest, in denen gcc nach verfuegbaren Header-Dateien und Bibliotheken sucht?
    gcc -l Fuer das Linken zu einer Bibliotheksdateien
    gcc -L Fuer das Linken in einen Ordner fuer Bibliotheksdateien
    gcc -I Fuer das zeigen auf spezifische Header

Erklaeren Sie die Bedeutung der gcc-Parameter -I, -L, -l
    -I dir
    Fuegt das angegebene Verzeichnis dir zu der Liste von Verzeichnissen, in denen nach Header Files gesucht wird, hinzu.
    -Ldir
    Fuegt das angegebene Verzeichnis dir zu der Liste von Verzeichnissen, in denen mit -l nach Biblioteken gesucht wird, hinzu.
    -l library
    Sucht die Bibliothek names libary beim Linken.

Was ist der Unterschied zwischen #include "header.h" und #include <header.h>?
    #include "header.h": Fuer eigene header files
    #include <header.h>: Fuer system header files

Erklaeren Sie den Unterschied zwischen statischem und dynamischen Linken. Wie koennen Sie statisches Linken erzwingen? Welche Konsequenzen hat das?
    Statisches Linken
    Das statische Linken findet einmalig nach der Fertigstellung des Programms statt. Dadurch is das Programm portabel, da es ohne andere
    Dateien ausgefuehrt werden kann, aber benoetigt mehr Speicher und muss nach Wartung komplett neu kompiliert werden.
    Man kann statische Linken mit -static erzwingen.

    Dynamisches Linken
    Das dynamische Linken findet erst zur Laufzeit statt. Dazu verwendet man meistens dynamische Bibliotheken, die spaeter leicht ausgetauscht
    werden koennen. Darum werden die aufrufenden Programme kleiner und der Speicher fuer die Biblitheken wird nur einmal benoetigt, auch wenn
    diese mehrmals verwendet werden. Es kommt allerdings zu Problemen wenn nicht die richtige Bibliotheksversion vorliegt.

Welche Informationen liefert Ihnen das Linux-Tool ldd?
    ldd gibt die von einem Programm benoetigten Objekte/Biblioteken zurueck. Dabei inspiziert der Linker die dynamischen Abhaengigkeiten des
    Programms, gibt den Speicherort zurueck und laedt diese Objekte.
```

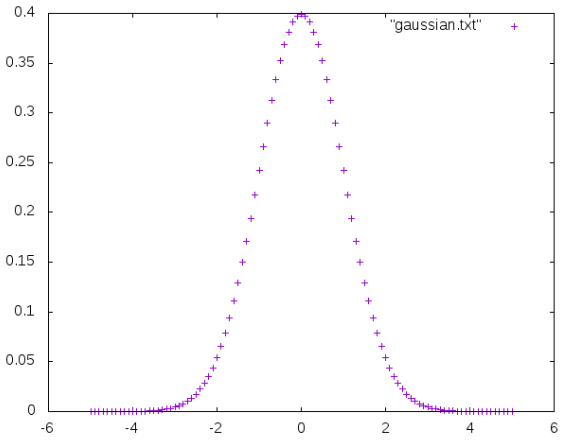
Aufgaben/Blatt01/Final/1_2.txt

```c
/*
    Prints a Gaussian distribution with the
    standard deviation sigma from -5 to 5 on
    stdout/terminal

*/
#include <stdio.h>
#include <gsl/gsl_randist.h>

int main(){

    double counter;
    double sigma = 1.0;



    for(counter = -5.0; counter < 5.0; counter+=0.1){
        printf("%5.2f    %.6f\n", counter, gsl_ran_gaussian_pdf(counter,
            sigma));
    }



    return 0;
}
```

Aufgaben/Blatt01/Final/gsl_ran.c

```
all: gsl_ran first_diff break_words line_mean

gsl_ran: gsl_ran.o
    gcc -L/usr/local/lib gsl_ran.o -lgsl -lgslcblas -o gsl_ran

gsl_ran.o: gsl_ran.c
    gcc -c gsl_ran.c

first_diff: first_diff.c
    gcc -o first_diff first_diff.c

break_words: break_words.c
    gcc -o break_words break_words.c

line_mean: line_mean.c
    gcc -o line_mean line_mean.c

clean:
    rm *o
    rm gsl_ran
    rm break_words
    rm first_diff
    rm line_mean
```

Aufgaben/Blatt01/Final/Makefile.txt


"gaussian.txt"
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-6
-4
-2
0
2
4
6