LANDESBERUFSSCHULE 4 SALZBURG

Informatik

Datei (File) Operationen

LBS 4

Dieses Skript dient als zusätzliche Lernunterlage für Informatik

Inhalt

Dateien benutzen 3
Dateien öffnen und schließen 3
Syntax von fopen() 3
Bearbeitsungsmodi 3
Syntax fclose() 3
Beispiel in C: 4
Lesen einer Datei: 4
Schreiben einer Datei 5
Syntax von fprintf() 5
Beispiel für fprintf(): 5
Beispiel in C: schreiben einer Datei 5

Dateien benutzen

Bis jetzt haben Sie Daten (Zeichen und Zahlen) eingegeben und verarbeitet, danach waren diese wieder verschwunden. Es fehlt noch die Möglichkeit, die Daten dauerhaft (persistent) zu speichern.

In der Programmiersprache C gibt es viele Funktion um Dateien zu erstellen und manipulieren. Dieses Dokument zeigt eine kleine Übersicht, welche Schritte in der Programmiersprache C nötig sind.

Dateien öffnen und schließen

Als erstes muss eine Datei geöffnet werden. Das OS stellt dann den nötigen Hauptspeicher zur Verfügung. Der Befehl in C lautet *fopen()*. Nach dem kann der Inhalt bearbeitet werden. Als letzten Schritt muss die Datei mit *fclose()* wieder geschlossen werden.

Syntax fopen()

Der Befehl fopen(„dateiname“) erstellt eine neue Datei. Sollte diese vorhanden sein wird Sie überschrieben.

Syntax von fopen()

```
FILE * fopen ( const char * filename, const char * mode );
```

fopen() erwartet einen Dateinamen und eine Zeichenkette für den Modus. Als Ergebnis wird ein Zeiger der Datei zurückgegeben.

Bearbeitsungsmodi

r	öffnet die Datei nur im Lesemodus
w	öffnet die Datei zum schreiben
a	öffnet die Datei zum Schreiben am Ende der Datei

Syntax fclose()

```
int fclose ( FILE * stream );
```

fclose erwartet einen Zeiger auf FILE. Wenn die Operation erfolgeich war, wird der Wert 0 zurückgegeben, sonst EOF.

Beispiel in C:

```
#include <stdio.h>

int main() {
        FILE *datei_ptr;              /*(1)*/
        datei_ptr = fopen("ac.aac","w");   /*(2)*/
        if(datei_ptr != NULL) {       /*(3)*/
                fclose(datei_ptr);    /*(4)*/
        }
return 0;
}
```

1 Ein Zeiger **datei_ptr* wird deklariert und zeigt auf den Type FILE

2 mit *fopen* wird versucht die Datei *„ac.aac“* zu öffnen. Falls erfolgreich wird ein Zeiger auf die Datei zurückgegeben.

3 Es wird geprüft, ob der Vorgang erfolgreich war. Sonst wird *NULL* zurückgegeben.

4 mit *fclose* wieder die Datei wieder geschlossen

Lesen einer Datei:

Im nächsten Beispiel wird gezeigt wie man eine Datei lesen kann und auf dem Bildschirm ausgegeben wird.

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

Int main() {
        FILE *stream; //Filepointer erstellen
        char dateiname[67]; //Array zum zwischenspeichern deklarieren
        int ch;
        printf("\nWelche Datei wollen Sie lesen? >");
        gets(dateiname);     //Dateiname einlesen

        if((stream = fopen(dateiname,"r")) //überprüfen ob Datei vorhanden
                        == NULL) {
                printf("\nFehler beim Öffnen!");
                exit(1);
        }
        ch=fgetc(stream);             //erstes Zeichen aus dem Stream holen

        while(!feof(stream)) {        //solange lesen bis Dateiende kommt (1)
                putchar(ch);          //auf dem Bildschirm ausgeben
                ch=fgetc(stream);
        }

        fclose(stream);               //Datei schließen
return 0;
}
```

(1) *feof()* prüft, ob an einem Stream noch Daten anliegen oder der End-Of-File-Indikator gesetzt ist. Es wird solange eine positive Zahl zurückgegeben bis das Dateiende erreicht ist. (-1)

Schreiben einer Datei

Zum Schreiben in eine Datei verwenden wir die Funktion fprintf().

fprintf() wird dazu verwendet, um einen String aus einem FormatString zu erzeugen und anschließend auf einem Ausgabestream (FILE) auszugeben.

Syntax von fprintf()

```
int fprintf( FILE * file, char const * formatString, ...);
```

fprintf erwartet einen Zeiger auf eine Datei,

formatString beschreibt wie sich die Zeichenkette zusammensetzt.

Beispiel für fprintf():

```
fprintf(fp, "%s %s %s %d", "C", "in der LBS 4", "Salzburg", 2015);

fp= Zeiger auf eine Datei
%s, %d = Platzhalter für die Variablen und Konstanten
Konstante Zeichenketten
```

Beispiel in C: schreiben einer Datei

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

int main() {
        FILE *stream;
        char zeile[81],dateiname[67]; //(1)
        int zeilen=0;

        printf("\n\t\tTextzeilen erfassen\n");
        printf("\nSpeichern unter >");
        gets(dateiname);

        if((stream = fopen(dateiname,"w")) == NULL) {
                printf("\nFehler beim Oeffnen!");
                exit(1);
        }

        printf("\nLeerzeile beendet das Programm.");
        printf("\n------------------------------\n");

        gets(zeile);                            //(2)
        while(strlen(zeile) > 0) {
                fprintf(stream,"%s\n",zeile);       //(3)
```

```
                zeilen++;
                gets(zeile);
        }
        fclose(stream);

        printf("\n%i Zeilen wurden geschrieben.\n",zeilen);
return 0;
}
```

(1) in dieser Zeile werden zwei Arrays für Zeichenketten erstellt (Dateiname, Text)
(2) in dieser Zeile wird der Text vom Benutzer eingelesen
(3) in dieser Zeile wird die eingelesene Zeile in den Stream geschrieben

Die While-Schleife läuft solange bis der Benutzer nichts mehr eingibt.

Weitere Dateioperationen

Rename()

Wird zum Umbenennen von Dateien verwendet.

Syntax von rebame()

int rename(const char *old_filename, const char *new_filename)

erster Parameter: bestehender Dateiname

zweiter Parameter: neuer Dateiname

Rückgabewert: bei Erfolg 0 sonst -1

Beispiel C-Code:

```
#include <stdio.h>

int main (){
int check;
char oldname[] = "datei.txt";
char newname[] = "neuedatei.txt";
ret = rename(oldname, newname);

if(check == 0) {
        printf("Umbenennen war erfolgreich\n");
}
else {
printf("Umbenennen war nicht erfolgreich\n");
}
return(0);
}
```

Remove()

Diese Funktion dient zum Löschen Dateien. Als Parameter wird ein Filepointer erwartet

Syntax:

int remove(const char *filename)

Parameter. Zeiger auf eine Datei

Rückgabewert: bei Erfolg 0; Fehler -1

Beispiel in C

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

int main (){
int check;
FILE *fp;
char filename[] = "datei.txt";
fp = fopen(filename, "w"); //Datei erstellen

fprintf(fp, "%s", "Zeichenkette für C"); //Zeichenkette einfügen
fclose(fp);

        if(check = remove(filename)== 0) {
                printf("Datei erfolgreich gelöscht");
        }
        else {
                printf("Löschen fehlgeschlagen");
        }
return(0);
}
```