6.2.F Extras/Lesen aus Dateien – Ergänzungen und Bilder

6.2.F.1 Einlesen in ein Array von Records – Schritt für Schritt

Die folgende Textdatei `xdatei.txt` enthält Daten zu mehreren Personen.

```
Meier
Heinz
123456
Schulze
Fritz
654321
```

Sie soll durch ein Programm eingelesen werden. Für erfahrene Programmierer ist das kein Problem. Für den Anfänger dient die folgende Schritt-für-Schritt-Vorgehensweise.

6.2.F.2 Lesen eines Zeichens in eine int-Variable

Zuerst reicht es, das erste Zeichen in eine int-Variable zu lesen. Zur Kontrolle soll der Inhalt der int-Variablen ausgegeben werden. Dazu dient das Programm `xlesen0.c`:

```c
#include <stdio.h>

int main(void)
{
    int zeichen;
    FILE *dm;
    dm=fopen("xdatei.txt", "r");
    if(dm==NULL)
    {
        perror("xdatei.txt");
        return 1;
    }
    zeichen=fgetc(dm);
    if(zeichen!=EOF)
    {
        printf("Gelesen: %i (ist ein >>%c<<)\n", zeichen, zeichen);
    }
    else
    {
        printf("Datei-Ende erreicht\n");
    }
    fclose(dm);
    return 0;
}
```

Terminal

```
schueler@debian964:~$ xlesen0
Gelesen: 77 (ist ein >>M<<)
```

6.2.F.3 Nacheinander jedes Zeichen in eine int-Variable lesen

Und nun soll jedes Zeichen nacheinander in dieselbe int-Variable gelesen werden. Dazu dient das Programm `xlesen1.c`:

```c
#include <stdio.h>

int main(void)
{
    int zeichen;
    FILE *dm;
    dm=fopen("xdatei.txt", "r");
    if(dm==NULL)
    {
        perror("xdatei.txt");
        return 1;
    }
    zeichen=fgetc(dm);
    while(zeichen!=EOF)
    {
        printf("Gelesen: %i (ist ein >>%c<<)\n", zeichen, zeichen);
        zeichen=fgetc(dm);
    }
    printf("Datei-Ende erreicht\n");
    fclose(dm);
    return 0;
}
```

Terminal

```
schueler@debian964:~$ xlesen1
Gelesen: 77 (ist ein >>M<<)
Gelesen: 101 (ist ein >>e<<)
..
Gelesen: 10 (ist ein >>

<<)
Datei-Ende erreicht
```

6.2.F.4 Nacheinander jedes Zeichen in ein Element eines Arrays einlesen

Das Einlesen in ein Array fordert besondere Umsicht. Man darf ja nicht über die Grenze des Arrays hinaus schreiben. Darum soll es nun darum gehen, wie am Anfang mit `fgetc()` einzelne Zeichen zu lesen. Wie in `xlesen1.c` soll das auch hier mit einer Schleife passieren, die so lange läuft, bis alle Zeichen aus der Datei gelesen wurden. Diesmal aber soll jedes gelesene Zeichen in ein anderes Element des Arrays geschrieben werden. Dazu dient das Programm `xlesen2.c`:

```c
#include <stdio.h>
#define MAXS 15
int main(void)
{
    int zeichen;
    char string[MAXS+1]="";
    int index=0;
    FILE *dm;
    dm=fopen("xdatei.txt", "r");
    if(dm==NULL)
    {
        perror("xdatei.txt");
        return 1;
```

```
14      }
15      zeichen=fgetc(dm);
16      while(zeichen!=EOF && index<MAXS)
17      {
18          string[index]=zeichen;
19          printf("Pos.%i: %i (>>%c<<)\n", index,
20                  string[index], string[index]);
21          zeichen=fgetc(dm);
22          ++index;
23      }
24      string[index]='\0'; // terminieren, weil String
25      if(zeichen==EOF)
26          printf("Datei-Ende erreicht, %i Bytes eingelesen\n", index);
27      else
28          printf("Zu viele Bytes in der Datei\n");
29      printf("Eingelesen: >>%s<<\n", string);
30      fclose(dm);
31      return 0;
32  }
```

Terminal

```
schueler@debian964:~$ xlesen2
Pos.0: 77 (>>M<<)
..
Pos.14: 50 (>>2<<)
Zu viele Bytes in der Datei
 Eingelesen: >>Meier
Heinz
12<<
```

Je nachdem, ob die Datei für das Array zu lang ist oder nicht, bekommt man eine entsprechende Meldung. Wichtig ist auch, wie viele Elemente in das Array gelesen wurden.

6.2.F.5 Lesen einer Zeile in eine String-Variable

Jetzt geht es um das Einlesen einer ganzen Zeile auf einmal. Mit der Funktion `fgets` werden mehrere Zeichen auf einmal in ein vom Benutzer bereitgestelltes Array gefüllt. Die Funktion endet bei Zeilenende, Erreichen der Maximalzahl an Zeichen und bei Dateiende oder Fehler. Das Dateiende oder einen Fehler erkennt man daran, dass die Funktion einen Nullzeiger zurückgibt. Das Zeilenende erkennt man daran, dass das Zeilenende-Zeichen `\n` im String vorkommt. Das Programm `xlesen3.c` zeigt das Vorgehen:

```
1   #include <stdio.h>
2
3   int main(void)
4   {
5       char str[81];
6       char *p;
7       FILE *dm;
8       dm=fopen("xdatei.txt", "r");
9       if(dm==NULL)
10      {
11          perror("xdatei.txt");
12          return 1;
```

```
13      }
14      p=fgets(str, 81, dm);
15      if(p!=NULL)
16      {
17          printf("Gelesen: >>%s<<\n", str);
18      }
19      else
20      {
21          printf("Datei-Ende erreicht\n");
22      }
23      fclose(dm);
24      return 0;
25  }
```

Terminal

```
schueler@debian964:~$ xlesen3
Gelesen: >>Meier
<<
```

Man sieht, dass am Ende des eingelesenen Strings ein Zeilenumbruch liegt.

6.2.F.6 Nacheinander jede Zeile in eine String-Variable lesen

Nun soll jede Zeile nacheinander in dieselbe String-Variable gelesen werden. Dazu dient das Programm `xlesen4.c`:

```
1   #include <stdio.h>
2
3   int main(void)
4   {
5       char str[81];
6       char *p;
7       FILE *dm;
8       dm=fopen("xdatei.txt", "r");
9       if(dm==NULL)
10      {
11          perror("xdatei.txt");
12          return 1;
13      }
14      p=fgets(str, 81, dm);
15      while(p!=NULL)
16      {
17          printf("Gelesen: >>%s<<\n", str);
18          p=fgets(str, 81, dm);
19      }
20      printf("Datei-Ende erreicht\n");
21      fclose(dm);
22      return 0;
23  }
```

Terminal

```
schueler@debian964:~$ xlesen4
Gelesen: >>Meier
<<
```

```
Gelesen: >>Heinz
<<
...
Datei-Ende erreicht
```

6.2.F.7 Nacheinander jede Zeile in ein Element eines Arrays (von Arrays) einlesen

Nun soll jede Zeile nacheinander in ein Array-Element gelesen werden. Dazu braucht man ein Array von Strings, also ein Array von Arrays von Zeichen. Das Programm `xlesen5.c` zeigt das:

```c
#include <stdio.h>
#include <string.h>
#define MAXZEILEN 5
int main(void)
{
    int index=0;
    char strings[MAXZEILEN][81]={""};
    char eingabe[81];
    char *p;
    FILE *dm;
    dm=fopen("xdatei.txt", "r");
    if(dm==NULL)
    {
        perror("xdatei.txt");
        return 1;
    }
    p=fgets(eingabe, 81, dm);
    while(p!=NULL && index< MAXZEILEN)
    {
        strncat(strings[index], eingabe,80);
        printf("Gelesen: >>%s<<\n", strings[index]);
        p=fgets(eingabe, 81, dm);
        ++index;
    }
    printf("%i Zeilen in das Array geschrieben.\n", index);
    if(p==NULL)
    {
        printf("Datei-Ende erreicht\n");
    }
    else
    {
        printf("Zu viele Zeilen in der Datei\n");
    }
    fclose(dm);
    return 0;
}
```

Terminal

```
schueler@debian964:~$ xlesen5
Gelesen: >>Meier
<<
Gelesen: >>Heinz
<<
Gelesen: >>123456
```

```
<<
Gelesen: >>Schulze
<<
Gelesen: >>Fritz
<<
5 Zeilen in das Array geschrieben.
Zu viele Zeilen in der Datei
```

Hier sieht man, dass unsere Datei mehr Zeilen hat als das Array Elemente.

6.2.F.8 Drei Zeilen in eine Record-Variable lesen

Jetzt steht eine Record-Variable zur Verfügung. Die ersten drei Zeilen sollen in diese Variable gelesen werden. Dazu dient das Programm `xlesen6.c`:

```c
#include <stdio.h>

struct persontyp
{
   char vn[30];
   char nn[30];
   long int gebdat;
};

int main(void)
{
   struct persontyp person={""};
   FILE *dm;
   int rc;
   dm=fopen("xdatei.txt", "r");
   if(dm==NULL)
   {
      perror("xdatei.txt");
      return 1;
   }
   fgets(person.nn, 30, dm);
   fgets(person.vn, 30, dm);
   rc=fscanf(dm, "%li", &person.gebdat);
   if(rc==1)
   {
      printf("%s%s%li\n", person.vn, person.nn, person.gebdat);
   }
   else
   {
      printf("Fehler oder Datei-Ende erreicht\n");
   }
   fclose(dm);
   return 0;
}
```

Terminal

```
schueler@debian964:~$ xlesen6
Heinz
```

```
Meier
123456
```

6.2.F.9 Nacheinander je drei Zeilen in eine Record-Variable lesen

Und jetzt sollen nacheinander je drei Zeilen in dieselbe Record-Variable gelesen werden. Dazu dient das Programm `xlesen7.c`:

```c
#include <stdio.h>

struct persontyp
{
   char vn[30];
   char nn[30];
   long int gebdat;
};

int main(void)
{
   struct persontyp person={""};
   FILE *dm;
   int rc;
   dm=fopen("xdatei.txt", "r");
   if(dm==NULL)
   {
      perror("xdatei.txt");
      return 1;
   }
   fgets(person.nn, 30, dm);
   fgets(person.vn, 30, dm);
   rc=fscanf(dm, "%li", &person.gebdat);   fgetc(dm);// !
   while(rc==1)
   {
      printf("%s%s%li\n", person.vn, person.nn, person.gebdat);
      fgets(person.nn, 30, dm);
      fgets(person.vn, 30, dm);
      rc=fscanf(dm, "%li", &person.gebdat);
   }
   printf("Fehler oder Datei-Ende erreicht\n");
   fclose(dm);
   return 0;
}
```

Terminal

```
schueler@debian964:~$ xlesen7
Heinz
Meier
123456
Fritz
Schulze
654321
Fehler oder Datei-Ende erreicht
```

6.2.F.10 Nacheinander je 3 Zeilen in eine Record-Variable lesen, die Element eines Arrays von Records ist

Zum Schluss sollen wieder nacheinander je drei Zeilen in eine Record-Variable gelesen werden (wie in `xlesen4.c`). Aber diese Record-Variable wird in ein Element eines Arrays kopiert. Dazu dient das Programm `xlesen8.c`:

```
1  #include <stdio.h>
2  #define MAXP 2
3  struct persontyp
4  {
5     char vn[30];
6     char nn[30];
7     long int gebdat;
8  };
9
10 int main(void)
11 {
12    int index=0;
13    struct persontyp personen[MAXP]={""};
14    struct persontyp zwischen; // Zwischenspeicher
15    FILE *dm;
16    int rc;
17    dm=fopen("xdatei.txt", "r");
18    if(dm==NULL)
19    {
20       perror("xdatei.txt");
21       return 1;
22    }
23
24    fgets(zwischen.nn, 30, dm);
25    fgets(zwischen.vn, 30, dm);
26    rc=fscanf(dm, "%li", &zwischen.gebdat); fgetc(dm);// !
27    while(rc==1 && index<MAXP)
28    {
29       personen[index]=zwischen;
30       printf("%s%s%li\n", personen[index].vn, personen[index].nn,
31                           personen[index].gebdat);
32       fgets(zwischen.nn, 30, dm);
33       fgets(zwischen.vn, 30, dm);
34       rc=fscanf(dm, "%li", &zwischen.gebdat);
35       ++index;
36    }
37    printf("%i Elemente in das Array gelesen\n", index);
38    if(rc!=1)
39       printf("Fehler oder Datei-Ende erreicht\n");
40    else
41       printf("Zu viele Datensaetze in der Datei\n");
42    fclose(dm);
43    return 0;
44 }
```

Die Kontroll-Ausgabe sieht wieder fast aus wie oben, aber die Inhalte liegen nun sauber in einem Array; sinnvollerweise wird die Anzahl der Elemente ausgegeben, die nun im Array liegen:

Terminal
```
schueler@debian964:~$ xlesen8
Heinz
...
654321
2 Elemente in das Array gelesen.
Fehler oder Datei-Ende erreicht
```

6.2.F.11 Was fehlt noch?

Einiges kann man noch besser machen:

- Mit `fgets()` wird der Zeilentrenner mit eingelesen. Man kann ihn durch den Terminator ersetzen:

```
1  int len=strlen(zwischen.vn);
2  if(len>0)
3  {
4      if(zwischen.vn[len-1]=='\n')
5      {
6          zwischen.vn[len-1]='\0';
7      }
8  }
```

- Es lohnt sich, das Programm in Funktionen zu zerteilen.

6.2.F.12 Übersicht

Tabelle 1 zeigt, worin sich die einzelnen Programme, die `xdatei.txt` einlesen, unterscheiden.

Aktion	1 Objekt einlesen	Alle Objekte einlesen in dieselbe Variable	Alle Objekte einlesen in ein Array
Prog.-Struktur	if,else	while not eof	while not eof and index < MAX
Zeichen char z fgetc()	xlesen0.c	xlesen1.c	xlesen2.c
String char s[] fgets()	xlesen3.c	xlesen4.c	xlesen5.c
Record struct persontyp z 2xfgets+fscanf	xlesen6.c	xlesen7.c	xlesen8.c

Abbildung 1: Unterschiede zwischen den Programmen