Einführung in die Programmierung für Physiker

Die Programmiersprache C – Ein- und Ausgabe

Marc Wagner

Institut für theoretische Physik
Johann Wolfgang Goethe-Universität Frankfurt am Main

WS 2017/18

putchar, getchar, printf, scanf

- **printf** und **scanf** wurden bereits ausführlich behandelt.
- **int putchar(int c)**
  (in der Bibliothek **stdio.h** enthalten) gibt ein einzelnes Zeichen aus (das, dessen ASCII-Code an **c** übergeben wird).
- **int getchar(void)**
  (in der Bibliothek **stdio.h** enthalten) liest ein einzelnes Zeichen ein und liefert dessen ASCII-Code zurück bzw. **EOF** bei einem Fehler.

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

// ********************

// Einfache printf-Version, realisiert mit Hilfe von putchar().
void my_printf(const char string[])
{
  int pos = 0;

  while(1)
    {
      if(string[pos] == '\0')
        // String-Ende erreicht.
        return;

      putchar(string[pos]);
      pos++;
    }
}

// ********************

int main(void)
{
  putchar('A');
  putchar('B');
  putchar('C');
  putchar('\n');

  my_printf("Gib ein Zeichen ein: ");

  int c;

  if((c = getchar()) == EOF)
    {
      printf("Fehler: int main(...\n");
      exit(0);
    }

  printf("Das eingelesene Zeichen ist \"%c\".\n", (char)c);
}
```

```
ABC
Gib ein Zeichen ein: q
Das eingelesene Zeichen ist "q".
```

In Dateien schreiben, aus Dateien lesen

- Um in eine Datei zu schreiben oder aus ihr zu lesen, muss sie zunächst geöffnet werden; dies geschieht mit
  `FILE *fopen(const char *path, const char *mode)`
  (in der Bibliothek **`stdio.h`** enthalten):
  - **`path`** ist der Pfad und Dateiname der zu öffnenden Datei.
  - **`mode`** ist die Zugriffsart:
    - **`"r"`**: aus der Datei lesen.
    - **`"w"`**: in die Datei schreiben (Datei dabei neu anlegen).
    - **`"a"`**: in die Datei schreiben (am Ende der Datei anhängen).
  - Der Rückgabewert ist ein Dateizeiger (vom Typ **`FILE *`**); dieser wird beim Schreiben oder Lesen als Verweis auf die entsprechende Datei benötigt.
- Schreiben in oder Lesen aus einer Datei mit leicht modifizierten Versionen bekannter Funktionen (**`stream`** ist der von **`fopen`** zurückgelieferte Dateizeiger).
  - **`int fprintf(FILE *stream, const char *format, ...)`**
    statt
    `int printf(const char *format, ...)`.
  - **`int fscanf(FILE *stream, const char *format, ...)`**
    statt
    `int scanf(const char *format, ...)`.
  - **`int fputc(int c, FILE *stream)`**
    statt
    `int putchar(int c)`.
  - **`int fgetc(FILE *stream)`**
    statt
    `int getchar(void)`.
  - **`char *fgets(char *s, int size, FILE *stream)`**
    statt
    `char *fgets(char *s, int size, stdin)`.
- Ist das Schreiben in oder Lesen aus einer Datei beendet, sollte die Datei mit
  `int fclose(FILE *fp)`
  (in der Bibliothek **`stdio.h`** enthalten) wieder geschlossen werden; dabei ist **`fp`** der von **`fopen`** zurückgelieferte Dateizeiger; dieser ist nach **`fclose`** nicht mehr verwendbar.
- **Beispiel:** Schreiben in, Lesen aus und Anhängen an Dateien ...
  - Datei **`prog1.c`**:

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

int main(void)
{
  FILE *file;

  // Oeffnet die Datei "datei_xyz.txt" zum Schreiben.
  file = fopen("datei_xyz.txt", "w");

  if(file == NULL)
    // NULL zeigt einen Fehler bei fopen an.
    {
      printf("Fehler bei fopen.\n");
```

```
15.       exit(0);
16.     }
17.
18.   fprintf(file, "1.2\n");
19.   fprintf(file, "400.0\n   5.1\n");
20.
21.   // Schliesst die Datei "datei_xyz.txt".
22.   fclose(file);
23. }
```

```
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ls -l
insgesamt 48
-rwxrwxr-x 1 mwagner mwagner 8910 Jan 20 12:06 prog1
-rw-rw-r-- 1 mwagner mwagner  413 Jan 20 12:06 prog1.c
-rwxrwxr-x 1 mwagner mwagner 9241 Jan 20 12:07 prog2
-rw-rw-r-- 1 mwagner mwagner 1115 Jan 20 12:04 prog2.c
-rwxrwxr-x 1 mwagner mwagner 8911 Jan 20 12:10 prog3
-rw-rw-r-- 1 mwagner mwagner  399 Jan 20 12:10 prog3.c
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ./prog1
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ls -l
insgesamt 52
-rw-rw-r-- 1 mwagner mwagner   15 Jan 20 12:47 datei_xyz.txt
-rwxrwxr-x 1 mwagner mwagner 8910 Jan 20 12:06 prog1
-rw-rw-r-- 1 mwagner mwagner  413 Jan 20 12:06 prog1.c
-rwxrwxr-x 1 mwagner mwagner 9241 Jan 20 12:07 prog2
-rw-rw-r-- 1 mwagner mwagner 1115 Jan 20 12:04 prog2.c
-rwxrwxr-x 1 mwagner mwagner 8911 Jan 20 12:10 prog3
-rw-rw-r-- 1 mwagner mwagner  399 Jan 20 12:10 prog3.c
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ less datei_xyz.txt
```

```
1.2
400.0
   5.1
datei_xyz.txt (END)
```

- Datei **`prog2.c`**:

```
1. #include<stdio.h>
2. #include<stdlib.h>
3.
4. int main(void)
5. {
6.   double d1;
7.   int i1;
8.
9.   // Oeffnet die Datei "datei_xyz.txt" zum Lesen.
10.  FILE *file1 = fopen("datei_xyz.txt", "r");
11.
12.  if(file1 == NULL)
13.    // NULL zeigt einen Fehler bei fopen an.
14.    {
15.      printf("Fehler bei fopen.\n");
16.      exit(0);
17.    }
18.
19.  // Oeffnet die Datei "datei_xyz_exp.txt" zum Schreiben.
20.  FILE *file2 = fopen("datei_xyz_exp.txt", "w");
21.
22.  if(file2 == NULL)
23.    // NULL zeigt einen Fehler bei fopen an.
24.    {
25.      printf("Fehler bei fopen.\n");
26.      exit(0);
27.    }
28.
29.  while(1)
30.    {
31.      // Naechsten double-Wert aus "datei_xyz.txt" lesen.
32.      i1 = fscanf(file1, "%lf", &d1);
33.
34.      if(i1 == EOF)
35.        // Dateiende von "datei_xyz.txt" erreicht.
36.        break;
37.
38.      if(i1 != 1)
39.        // fscanf liefert die Anzahl der korrekt eingelesenen Werte.
```

```
40.         {
41.           printf("Fehler bei fscanf.\n");
42.           exit(0);
43.         }
44.
45.       // Bildschirmausgabe.
46.       printf("Lese double-Wert %f ...\n", d1);
47.
48.       // Den gelesenen double-Wert in Exponetialschreibweise in "datei_xyz_exp.txt" schreiben.
49.       fprintf(file2, "%+.5e\n", d1);
50.     }
51.
52.   fclose(file1);
53.   fclose(file2);
54. }
```

```
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ./prog2
Lese double-Wert 1.200000 ...
Lese double-Wert 400.000000 ...
Lese double-Wert 5.100000 ...
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ls -l
insgesamt 56
-rw-rw-r-- 1 mwagner mwagner   39 Jan 20 12:49 datei_xyz_exp.txt
-rw-rw-r-- 1 mwagner mwagner   15 Jan 20 12:47 datei_xyz.txt
-rwxrwxr-x 1 mwagner mwagner 8910 Jan 20 12:06 prog1
-rw-rw-r-- 1 mwagner mwagner  413 Jan 20 12:06 prog1.c
-rwxrwxr-x 1 mwagner mwagner 9241 Jan 20 12:07 prog2
-rw-rw-r-- 1 mwagner mwagner 1115 Jan 20 12:04 prog2.c
-rwxrwxr-x 1 mwagner mwagner 8911 Jan 20 12:10 prog3
-rw-rw-r-- 1 mwagner mwagner  399 Jan 20 12:10 prog3.c
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ less datei_xyz_exp.txt
```

```
+1.20000e+00
+4.00000e+02
+5.10000e+00
datei_xyz_exp.txt (END)
```

- Datei **`prog3.c`**:

```
 1. #include<stdio.h>
 2. #include<stdlib.h>
 3.
 4. int main(void)
 5. {
 6.   FILE *file;
 7.
 8.   // Oeffnet die Datei "datei_xyz.txt" zum Anhaengen.
 9.   file = fopen("datei_xyz_exp.txt", "a");
10.
11.   if(file == NULL)
12.     // NULL zeigt einen Fehler bei fopen an.
13.     {
14.       printf("Fehler bei fopen.\n");
15.       exit(0);
16.     }
17.
18.   fprintf(file, "abc def ghi ...\n");
19.
20.   // Schliesst die Datei "datei_xyz_exp.txt".
21.   fclose(file);
22. }
```

```
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ./prog3
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ls -l
insgesamt 56
-rw-rw-r-- 1 mwagner mwagner   55 Jan 20 12:50 datei_xyz_exp.txt
-rw-rw-r-- 1 mwagner mwagner   15 Jan 20 12:47 datei_xyz.txt
-rwxrwxr-x 1 mwagner mwagner 8910 Jan 20 12:06 prog1
-rw-rw-r-- 1 mwagner mwagner  413 Jan 20 12:06 prog1.c
-rwxrwxr-x 1 mwagner mwagner 9241 Jan 20 12:07 prog2
-rw-rw-r-- 1 mwagner mwagner 1115 Jan 20 12:04 prog2.c
-rwxrwxr-x 1 mwagner mwagner 8911 Jan 20 12:10 prog3
-rw-rw-r-- 1 mwagner mwagner  399 Jan 20 12:10 prog3.c
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ less datei_xyz_exp.txt
```

```
+1.20000e+00
+4.00000e+02
+5.10000e+00
abc def ghi ...
datei_xyz_exp.txt (END)
```

stdin, stdout, stderr

- Bei Eingabe- und Ausgabe-Funktionen, die einen Dateizeiger als Parameter besitzen, kann an Stelle eines "**fopen**-Dateizeigers" auch **stdin**, **stdout** oder **stderr** verwendet werden:
  - **stdin**:
    Ein Zeiger auf den Standard-Eingabe-Strom, d.h. Tastaturinput; z.B. sind
    int fscanf(stdin, const char *format, ...)
    und
    int scanf(const char *format, ...)
    äquivalent.
  - **stdout**:
    Ein Zeiger auf den Standard-Ausgabe-Strom, d.h. Bildschirmausgabe; z.B. sind
    int fprintf(stdout, const char *format, ...)
    und
    int printf(const char *format, ...)
    äquivalent.
  - **stderr**:
    Ein Zeiger auf den Standard-Fehlerausgabe-Strom, d.h. ebenfalls Bildschirmausgabe; **stderr** unterscheidet sich von **stdout** z.B. dadurch, dass die Ausgabe unmittelbar stattfindet (bei **stdout** kommt es gelegentlich zu kleinen Verzögerungen) und dass das Umleiten der Bildschirmausgabe in eine Textdatei mit **>** nur **stdout**, nicht aber **stderr** betrifft.

```c
#include<stdio.h>

int main(void)
{
  fprintf(stderr, "fprintf(stderr, ... (1)\n");
  fprintf(stdout, "fprintf(stdout, ...\n");
  fprintf(stderr, "fprintf(stderr, ... (2)\n");
}
```

```
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ls -l
insgesamt 4
-rw-rw-r-- 1 mwagner mwagner 178 Jan 20 13:54 prog.c
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ g++ -o prog prog.c
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ./prog
fprintf(stderr, ... (1)
fprintf(stdout, ...
fprintf(stderr, ... (2)
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ./prog > output.txt
fprintf(stderr, ... (1)
fprintf(stderr, ... (2)
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ls -l
insgesamt 20
-rw-rw-r-- 1 mwagner mwagner   20 Jan 20 13:55 output.txt
-rwxrwxr-x 1 mwagner mwagner 8795 Jan 20 13:55 prog
-rw-rw-r-- 1 mwagner mwagner  178 Jan 20 13:54 prog.c
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ less output.txt
```

```
fprintf(stdout, ...
output.txt (END)
```

Binäres Schreiben in und Lesen aus Dateien

- Daten können nicht nur in Textdateien, sondern auch bitweise in sogenannte **Binärdateien** geschrieben werden.
- **Vorteile:**
  - Festplattenplatz-sparend, da z.B. für einen **`double`**-Wert in Bitdarstellung (typischer Weise) nur 8 Byte benötigt werden, ein solcher Wert als Text aber deutlich mehr Zeichen (d.h. Bytes) beansprucht; gerade beim Umgang mit großen Datenmengen ist das ein entscheidender Vorteil.
  - Kein Genauigkeitsverlust bei binärer Speicherung von Gleitkommawerten; werden diese in Textform gspeichert, wird die Ausgabe dagegen häufig nach einigen Stellen abgeschnitten.
- **Nachteile:**
  - Eine Binärdatei ist vom Programmierer nicht direkt lesbar (z.B. mit **`less`**), muss also immer mit einem geeigneten Programm geöffnet werden.
  - Binäres Lesen und Schreiben ist plattform-, unter Umständen sogar compilerabhängig, da die Reihenfolge, in der die Bytes einer Integer- oder Gleitkommavariable im Speicher liegen, nicht festgelegt ist (**Big Endian** vs. **Little Endian**).
- Einen Speicherbereich binär in eine Datei zu schreiben geschieht mit **`size_t fwrite(const void *ptr, size_t size, size_t nmemb, FILE *stream)`** (in der Bibliothek **`stdio.h`** enthalten):
  - **`ptr`** ist die Adresse des zu schreibenden Speicherbereichs.
  - Es werden **`nmemb`** Werte der Bytegröße **`size`** geschrieben.
  - **`stream`** ist der von **`fopen`** zurückgelieferte Dateizeiger.
- Aus einer Binärdatei zu lesen und damit einen Speicherbereich zu füllen geschieht mit **`size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream)`** (in der Bibliothek **`stdio.h`** enthalten; Bedeutung der Parameter wie bei **`fwrite`**).
- **Beispiel:** Schreiben als Textdatei und Schreiben als Binärdatei im Vergleich ...
  - Datei **`prog1.c`**:

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

int main(void)
{
  FILE *file;
  int i1;

  // *****

  // Zehn Zufallszahlen im Array a speichern.

  const int n = 10;
  double a[n];

  for(i1 = 0; i1 < n; i1++)
    // rand() generiert eine Integer-Zufallszahl zwischen 0 und RAND_MAX.
    a[i1] = ((double)rand() + 0.5) / ((double)RAND_MAX + 1.0);

  // *****

  // Das Array a in einer formatierten Textdatei ausgeben.

  file = fopen("array_a.txt", "w");
```

```
25.
26.   if(file == NULL)
27.     {
28.       printf("Fehler bei fopen.\n");
29.       exit(0);
30.     }
31.
32.   for(i1 = 0; i1 < n; i1++)
33.     fprintf(file, "%+.10f\n", a[i1]);
34.
35.   fclose(file);
36.
37.   // *****
38.
39.   // Das Array a in einer binaeren Datei ausgeben.
40.
41.   file = fopen("array_a.bin", "w");
42.
43.   if(file == NULL)
44.     {
45.       printf("Fehler bei fopen.\n");
46.       exit(0);
47.     }
48.
49.   fwrite(a, sizeof(double), n, file);
50.
51.   fclose(file);
52. }
```

```
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ls -l
insgesamt 32
-rwxrwxr-x 1 mwagner mwagner 9239 Jan 20 14:28 prog1
-rw-rw-r-- 1 mwagner mwagner  855 Jan 20 14:22 prog1.c
-rwxrwxr-x 1 mwagner mwagner 9239 Jan 20 14:28 prog2
-rw-rw-r-- 1 mwagner mwagner 1150 Jan 20 14:27 prog2.c
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ./prog1
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ls -l
insgesamt 40
-rw-rw-r-- 1 mwagner mwagner   80 Jan 20 14:30 array_a.bin
-rw-rw-r-- 1 mwagner mwagner  140 Jan 20 14:30 array_a.txt
-rwxrwxr-x 1 mwagner mwagner 9239 Jan 20 14:28 prog1
-rw-rw-r-- 1 mwagner mwagner  855 Jan 20 14:22 prog1.c
-rwxrwxr-x 1 mwagner mwagner 9239 Jan 20 14:28 prog2
-rw-rw-r-- 1 mwagner mwagner 1150 Jan 20 14:27 prog2.c
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ less array_a.txt
```

```
+0.8401877174
+0.3943829271
+0.7830992240
+0.7984400337
+0.9116473582
+0.1975513695
+0.3352227559
+0.7682295950
+0.2777747110
+0.5539699560
array_a.txt (END)
```

```
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ less array_a.bin
"array_a.bin" may be a binary file.  See it anyway?
```

```
<A3><C5>^UZ<D1><E2><EA>?${r<E3><91>=<D9>?L^^<92>^Z&^O<E9>?<A4>^Y^S^]%G<E3>%@<E9>?nX<9A>^T7,<ED>?<BA><92><B2><FF>\I<C9>?<94><E8>j%Jt<D5>?<AC>*O
array_a.bin (END)
```

- Datei **`prog2.c`**:

```
1.  #include<stdio.h>
2.  #include<stdlib.h>
3.
4.  int main(void)
5.  {
6.    FILE *file;
7.    int i1;
8.
9.    const int n = 10;
10.   double a_txt[n], a_bin[n];
11.
12.   // *****
13.
```

```
14.   // Die zehn double-Werte aus der formatierten Textdatei "array_a.txt" lesen.
15.
16.   file = fopen("array_a.txt", "r");
17.
18.   if(file == NULL)
19.     {
20.       printf("Fehler bei fopen.\n");
21.       exit(0);
22.     }
23.
24.   for(i1 = 0; i1 < n; i1++)
25.     {
26.       if(fscanf(file, "%lf", a_txt+i1) != 1)
27.         // fscanf liefert die Anzahl der korrekt eingelesenen Werte.
28.         {
29.           printf("Fehler bei fscanf.\n");
30.           exit(0);
31.         }
32.     }
33.
34.   fclose(file);
35.
36.   // *****
37.
38.   // Die zehn double-Werte aus der binaeren Datei "array_a.bin" lesen.
39.
40.   file = fopen("array_a.bin", "r");
41.
42.   if(file == NULL)
43.     {
44.       printf("Fehler bei fopen.\n");
45.       exit(0);
46.     }
47.
48.   if(fread(a_bin, sizeof(double), n, file) != 10)
49.     // fread liefert die Anzahl der korrekt eingelesenen Werte.
50.     {
51.       printf("Fehler bei fread.\n");
52.       exit(0);
53.     }
54.
55.   fclose(file);
56.
57.   // *****
58.
59.   for(i1 = 0; i1 < n; i1++)
60.     {
61.       printf("a_txt[%d] = %+.12f , a_bin[%d] = %+.12f , difference = %+.5e\n", i1, a_txt[i1], i1, a_bin[i1], a_txt[i1]-a_bin[i1]);
62.     }
63. }
```

```
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ./prog2
a_txt[0] = +0.840187717400 , a_bin[0] = +0.840187717388 , difference = +1.24598e-11
a_txt[1] = +0.394382927100 , a_bin[1] = +0.394382927052 , difference = +4.80763e-11
a_txt[2] = +0.783099224000 , a_bin[2] = +0.783099223991 , difference = +8.56348e-12
a_txt[3] = +0.798440033700 , a_bin[3] = +0.798440033709 , difference = -8.90388e-12
a_txt[4] = +0.911647358200 , a_bin[4] = +0.911647358170 , difference = +3.03850e-11
a_txt[5] = +0.197551369500 , a_bin[5] = +0.197551369526 , difference = -2.62146e-11
a_txt[6] = +0.335222755900 , a_bin[6] = +0.335222755948 , difference = -4.77197e-11
a_txt[7] = +0.768229595000 , a_bin[7] = +0.768229595045 , difference = -4.47347e-11
a_txt[8] = +0.277774711000 , a_bin[8] = +0.277774711036 , difference = -3.60184e-11
a_txt[9] = +0.553969956000 , a_bin[9] = +0.553969956028 , difference = -2.82612e-11
```