Fakultät IV - Elektrotechnik und Informatik
Einführung in die Programmierung WS 2015/16
Feldmann / Semmler / Lichtblau / Streibelt / Pujol / Rost

Aufgabenblatt 7

letzte Aktualisierung: 22. Dezember, 14:23 Uhr

(d8053e2d311b068de6b325741fcd00b8a9af1863)

Ausgabe: Mittwoch, 16.12.2015
Abgabe: spätestens Freitag, 8.1.2016, 20:00

Thema: *Ein- und Ausgabe, Fehlerbehandlung*

Abgabemodalitäten

- Alle abzugebenden Quelltexte müssen ohne Warnungen und Fehler auf den Rechnern des tubIT/IRB mittels `gcc -std=c99 -Wall` kompilieren.
- Abgaben erfolgen prinzipiell immer in Gruppen à 2 Personen, welche in den Tutorien festgelegt wurden. Einzelabgaben sind explizit als solche gekennzeichnet.
- Die Abgabe erfolgt ausschließlich über SVN. Die finale Abgabe
  - für Gruppenabgaben erfolgt im Unterordner `Tutorien/t<xx>/Gruppen/g<xx>/Abgaben/Blatt<xx>/`
  - für Einzelabgaben erfolgt im Unterordner `Tutorien/t<xx>/Studierende/<tuBIT-Login>/Abgaben/Blatt<xx>/`

  wobei die Ordner von uns erstellt werden.
- Benutze für alle Abgaben von Programmcode das folgende Namensschema: `introprog_blatt0X_aufgabe0Y_Z.c`, wobei X durch die Blattnummer, Y durch die Aufgabe und Z durch die Unteraufgabe zu ersetzen ist.
  Beispiel: Aufgabe 1.2 wird zu: `introprog_blatt01_aufgabe01_2.c`
  Gib für jede Unteraufgabe maximal eine Quellcodedatei ab, es sei denn, die Aufgabenstellung erfordert explizit die Abgabe mehrerer Dateien pro Aufgabe.

  Benenne alle anderen Abgaben (Pseudocode, Textaufgaben) wie oben beschrieben. Die zugelassenen Abgabeformate sind PDF, ODT und Text (txt). Verwende auch hier eine Datei pro Aufgabe, nicht jedoch pro Unteraufgabe.

1. Aufgabe: Weihnachtliche Ein- und Ausgabe (4 Punkte)

Auf diesem Übungszettel soll die Ein- und Ausgabe mittels Funktionen wie `sscanf`, `fgets` etc. sowie das eigenständige Strukturieren von Code erübt werden. Konkret besteht die Aufgabe darin Parameter aus einer Datei einzulesen und mittels dieser dann einen Weihnachtsbaum zu zeichnen. Im Gegensatz zu den vorigen Übungszetteln gibt es auf diesem Übungszettel keinerlei Vorgabe für die abzugebende Quelldatei: die Entwicklung der `main` Funktion – sowie etwaig weiterer Funktionen zum Einlesen – müssen selbstständig entwickelt werden; nur die eigentliche Funktionalität, die Ausgabe einer "ASCII-Weihnachtsbaum-Karte", wird von uns bereitgestellt. Eine Beispielausgabe kann zum Beispiel wie folgt aussehen.

Listing 1: ausgabe_korrekt.txt

```
Dumping parameters..
         width: <40>
        height: <20>
     character: <*>
       message: <Frohe Festtage!>

                    *
                   /|\
                  ///\\
                 i///\\\i
                 ////\\\\
                i/////\\\\\i
                //////\\\\\\
               i///////\\\\\\\i
               ////////\\\\\\\\
              i/////////\\\\\\\\\i
              //////////\\\\\\\\\\
             i///////////\\\\\\\\\\\i
             ////////////\\\\\\\\\\\\
            i/////////////\\\\\\\\\\\\\i
            //////////////\\\\\\\\\\\\\\
          i////////////////\\\\\\\\\\\\\\\i
          //////////////////\\\\\\\\\\\\\\\\
        i////////////////////\\\\\\\\\\\\\\\\\i
          //////////////////\\\\\\\\\\\\\\\\
             ////////////\\\\\\\\\\\\
                   ///\\\
                    |||||
                    |||||

                Frohe Festtage!
```

Die vorgegebene Funktion `print_tree` zur Erstellung der Ausgabe befindet sich in der Datei `input_blatt07.c`. Da eigentlich nur die Schnittstelle und nicht der enthaltene Code von Bedeutung sind, geben wir an dieser Stelle nur den Inhalt der Header-Datei an (siehe Listing 2). Das `struct parameter_t` fasst die möglichen Parameter für die Ausgabe-Funktion `print_tree` logisch zusammen. Die Bedeutung der Werte in diesem `struct` ist wie folgt.

`int width` Die maximale Breite des Baumes. Dieser Parameter muss eine natürliche Zahl aus dem Wertebereich $\{10, 11, \ldots, 60\}$ sein. Der Default-Wert ist 42. **Beachtet, dass die maximale Breite aus kosmetischen Gründen bei der Ausgabe im Regelfall nicht erreicht wird. Dies ist normal.**

`int height` Die Höhe des Baums exklusive der Baumspitze. Dieser Parameter muss eine natürliche Zahl aus dem Wertebereich $\{15, 16, \ldots 80\}$ sein. Der Default-Wert ist 17.

`char character` Ein einzelnes Zeichen (außer dem Leerzeichen bzw. dem Tabulator), welches auf der Spitze des Baums platziert wird. Der Default-Wert ist `'*'`.

`char message[255]` Eine kurze Nachricht, welche unterhalb des Baums ausgegeben wird. Der Default-Wert ist `"There's no place like home."`. Der Text muss mindestens ein Nicht-Leerzeichen enthalten.

Hinweis: Es darf angenommen werden, dass die Länge einer eingelesenen Zeile nicht über 999 Zeichen hinausgeht.

Listing 2: input_blatt07.h

```
1  #include <stdio.h>
2
3  //maximale Länge der Nachricht in der Parameter-Eingabedatei
4  #define MAX_LEN_MESSAGE 255
5  //maximale Länge einer Zeile in der Parameter-Eingabedatei
6  #define MAX_LEN_LINE 1000
7
8  //Parameter zum Zeichnen der Weichnachtsbaum-Karte
9  struct parameters_t {
10     //maximale Breite des Weihnachtsbaums: {10,..,60}, default: 42
11     int width;
12     //Höhe des Weihnachtsbaum: {15,..,80}, default: 17
13     int height;
14     //Text unterhalb des Weihnachtsbaums: default "There's no place
          ↪ like home."
15     char message[MAX_LEN_MESSAGE];
16     //Spitze des Weihnachtsbaums: default '*'
17     char character;
18 };
19
20 //Um anstatt 'struct parameter_t' auch einfach 'parameter' zu schreiben
21 typedef struct parameters_t parameters;
22
23 //Initialisiert ein alloziertes parameter struct mit Defaultwerten
24 void init_params_with_defaults(parameters* params);
25
26 //Überprüft ob das angegebene parameter struct valide Werte enthält
27 //Wenn die Parameter korrekt sind wird 1, ansonsten 0 zurückgegeben.
28 int params_are_valid(parameters* params);
29
30 //Gibt die Weihnachts-Karte gemäß params auf file aus
31 void print_tree(parameters* params, FILE* file);
```

Verwendung der Dateien und Ausgabe von Fehlern

Die Aufgabe besteht darin die obigen Parameter aus einer Datei einzulesen und den Weihnachtsbaum mittels der Funktion `print_tree` entweder auf der Konsole (`stdout`) oder in einer Datei auszugeben. Das fertige Programm soll somit wie folgt aufgerufen werden können:


```
./introprog_blatt07_aufgabe01 <parameter-datei> [ausgabe-datei]
```

Die Angabe der Datei mit den Parametern ist verpflichtend, während die Ausgabedatei nicht spezifiziert werden muss. Wird keine Parameterdatei angegeben, so soll nur folgende Ausgabe auf `stdout` erfolgen und das Programm mittels `exit(2);` beendet werden:

```
"usage: ./introprog_blatt07_aufgabe01 <input-file>  [output-file]\n"
```

Die Angabe der Ausgabedatei ist optional. Wurde keine Ausgabedatei angegeben, so hat die Ausgabe auf der Standardausgabe (`stdout`) zu erfolgen. Fehlernachrichten müssen in jedem Fall über die Standardfehlerausgabe (`stderr`) ausgegeben werden.
Die Dateien können mit `fopen` geöffnet werden. `fopen` gibt einen Pointer auf die Datei in dem Format `FILE` zurück, wenn das Öffnen erfolgreich war. Andernfalls wird ein `NULL`-Pointer zurückgegeben. Nach dem Öffnen der Datei soll überprüft werden, ob das Öffnen der Datei gelungen ist. Schlägt das Öffnen der Eingabedatei fehl, soll folgende Nachricht mittels `perror` erfolgen:

```
Could not open input file!
```

Gleichermaßen soll, sollte das Öffnen der Ausgabedatei fehlgeschlagen, folgende Nachricht mittels `perror` erfolgen:

```
Could not open output file!
```

Die Parameter werden in der Eingabedatei zeilenweise – in beliebiger Reihenfolge – in der Form `<N> <W>\n` übergeben. Hierbei bezeichnet `<N>` den Namen des Parameters und `<W>` den Wert. Als Besonderheit wird der String `message`, welcher unterhalb des Weihnachtsbaums ausgegeben werden soll in der Form `<N> "<W>" \n` spezifiziert. Somit muss der eigentliche Text, welcher ausgegeben werden soll, von doppelten Anführungszeichen umfasst sein (vgl. Listing 3).
Die Eingabedatei zur Erstellung des in Listing 1 abgebildeten Weihnachtsbaums hat den folgenden Inhalt:

Listing 3: parameter_korrekt.txt

```
1 character *
2 width 40
3 height 20
4 message "Frohe␣Festtage!"
```

Umgang mit fehlerhaften Zeilen

Ein wichtiger Teil der Aufgabe ist es nicht darauf zu vertrauen, dass die Eingaben vollständig dem obigen (korrekten) Format entsprechen. Die Parameterdatei `parameter_inkorrekt.txt` (siehe Listing 4) gibt beispielhaft ausschließlich fehlerhafte Spezifikationen von Parametern an.

Listing 4: parameter_inkorrekt.txt

```
1  character
2  character X_X
3  character X X
4  width
5  width 5
6  width 20.0
7  width XYZ
8  height 19 !?
9  HEIGHT 19
10 message "There is no place like home.
11 message There is no place like home. ""
12 message
```

Falscher Parametername Die Parameter müssen genau wie im `struct` definiert benannt werden. `HEIGHT` ist somit ungültig.

Fehlender Parameterwert Wird ein Parametername angegeben, so muss auch ein Wert folgen.

Falscher Parameterwert Während für `width` und `height` ganze Zahlen spezifiziert werden müssen, kann `character` jedes beliebige (einzelne) Zeichen (außer dem Leerzeichen oder einem Tabulator sein). Weiterhin muss der String `message` in doppelte Anführungszeichen gesetzt sein. Insbesonders kann dem Parameter `width` nicht der Wert `"Hallo Welt"` übergeben werden. Weiterhin müssen die Wertebereiche der Parameter beachtet werden.

Zusätzliche Daten In einer Zeile mit einem validen Paar aus Name und Wert darf kein weiterer Text (außer Tabulatoren und Leerzeichen) vorkommen. Somit ist z.B. `20.0` falsch, weil 20 die eigentliche Zahl ist und gefolgt wird von `.0`. Weiterhin ist auch `height 19 !?` falsch, obwohl `height 19` prinzipiell korrekt ist.

In allen der obigen Fälle, in denen eine Zeile nicht leer war, jedoch keine validen Parameter extrahiert werden konnten, soll eine Fehlermeldung folgender Form über `stderr` ausgegeben werden:

```
Error while reading line <Z>\n
```

Es kann dafür `fprintf()` verwendet werden. Hierbei bezeichnet `<Z>` die Zeile, in welcher der Fehler aufgetreten ist. Die Ausgabe bei Übergabe der obigen inkorrekten Parameterdatei (siehe Listing 4) wird in Listing 5 dargestellt. In der durch den Programmaufruf erstellten Datei `ausgabe_inkorrekt.txt`, siehe Listing 6, ist der dazugehörige Baum abgebildet. Wie dort zu sehen ist, erfolgt aufgrund falscher Eingabedaten die Ausgabe gemäß der Default-Werte: wird ein Parameter in keiner der Zeilen erfolgreich gelesen, wird der Default-Wert benutzt.

Listing 5: Fehlernachrichten auf stderr beim Einlesen von parameter_inkorrekt.txt

```
> ./introprog_blatt07_aufgabe01 parameter_inkorrekt.txt
  ↪ ausgabe_inkorrekt.txt
Error while reading line 1
Error while reading line 2
Error while reading line 3
Error while reading line 4
Error while reading line 5
Error while reading line 6
Error while reading line 7
Error while reading line 8
Error while reading line 9
Error while reading line 10
Error while reading line 11
Error while reading line 12
```

Listing 6: Der Inhalt von ausgabe_inkorrekt.txt

```
Dumping parameters..
         width: <42>
        height: <17>
     character: <*>
       message: <There's no place like home.>

                      *
                     /|\
                    //|\\
                  i ///|\\\ i
                  /////|\\\\\
                i//////|\\\\\\i
                ///////|\\\\\\\
             i ////////|\\\\\\\\ i
             //////////|\\\\\\\\\\
          i///////////|\\\\\\\\\\\i
          ///////////|\\\\\\\\\\\\
       i ////////////|\\\\\\\\\\\\\ i
       /////////////|\\\\\\\\\\\\\\
     i//////////////|\\\\\\\\\\\\\\\i
     ///////////////|\\\\\\\\\\\\\\\\
      ///////////////|\\\\\\\\\\\\\\\\
           //////////|\\\\\\\\\\\
                  ///|\\\
                   |||||
                   |||||

          There's no place like home.
```

Umgang mit Leerzeichen / mehrfache Angabe von Parametern

Alle Zeilen der in Listing 4 Parameterdatei sind inhärent fehlerhaft: es werden keine, mehrere oder falsche Daten angegeben. Vor diesem Hintergrund können aus diesen Zeilen keine Daten extrahiert werden. Dies ist jedoch anders, falls einfach nur zusätzliche Leerzeichen bzw. Tabulatoren benutzt werden, da die Daten prinzipiell korrekt sind. Weiterhin soll, wenn ein Parameter in der Eingabedatei mehrfach angegeben wurde, der letzte Wert übernommen werden.
In Listing 8 wird der Programmaufruf und die Fehlerausgabe für eine Datei mit manchen korrekten und manchen inkorrekten Parameterspezifikationen (siehe 7) gezeigt. Der Inhalt der Ausgabedatei wird in 9 gezeigt. In Listings 7 und 9 geben wir ein abschließendes ein Beispiel mit manchen korrekten und manchen inkorrekten Parameterspezifikationen sowie die entsprechende Ausgabe an.

Listing 7: parameter_gemischt.txt

```
␣␣character
␣width␣␣X
␣␣␣height␣␣␣␣␣␣␣␣␣␣25
␣␣␣␣message␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"my␣brain␣is␣open"

␣␣message␣"Frohe␣Festtage
height␣15
␣␣width␣␣␣␣␣␣␣␣␣27
```

Listing 8: Fehlernachrichten auf stderr beim Einlesen von parameter_gemischt.txt

```
> ./introprog_blatt07_aufgabe01 parameter_gemischt.txt ausgabe_gemischt
  ↪ .txt
Error while reading line 1
Error while reading line 2
Error while reading line 6
```

Listing 9: Der Inhalt von ausgabe_gemischt.txt

```
Dumping parameters..
         width: <27>
        height: <15>
     character: <*>
       message: <my brain is open>

              *
              |
             /|\
           i//|\\i
           ///|\\\
         i////|\\\\i
         /////|\\\\\
       i//////|\\\\\\i
       ///////|\\\\\\\
     i////////|\\\\\\\\i
     /////////|\\\\\\\\\
    i/////////|\\\\\\\\\i
    //////////|\\\\\\\\\\
    //////////|\\\\\\\\\\
       ///////|\\\\\\\
            //|\\
              |||
              |||

     my brain is open
```

Abschließende Hinweise

Kompilierung

Auf Grund der Verwendung der Bibliothek `math.h` in der Eingabedatei `input_blatt07.c` muss die Mathematik-Bibliothek mittels `-lm` bei der Kompilierung mit verlinkt werden. Übersetze das Programm also wie folgt:

Listing 10: Übersetzung des Programms

```
gcc -std=c99 -Wall introprog_blatt07_aufgabe01.c input_blatt07.c
               -o introprog_blatt07_aufgabe01 -lm
```

Abgabe

Gib dein Programm als `introprog_blatt07_aufgabe01.c` im entsprechenden Abgabeordner ab.

Bewertung

Die 4 Punkte werden prinzipiell wie folgt verteilt.

1 Punkt für die korrekte Verwendung der Dateien und der korrekten Ausgabe auf jeweils `stderr` und `stdout`.

1 Punkt für das korrekte Einlesen ausschließlich korrekter Parameterdatei (wie z.B. in Listing 3 abgebildet) und die korrekte Ausgabe in eine Datei bzw. auf `stdout`.

1 Punkt für das korrete Verarbeiten von Leerzeichen in der Eingabe analog zu Zeilen 3,4 und 8 in Listing 7.

1 Punkt für das korrekte Erkennen fehlerhafter Zeilen in der Eingabe – und der Ausgabe entsprechender Zeilennummern – analog zu Listing 4.

Beachte insbesonders alle unterstrichenen Sätze auf diesem Aufgabenblatt und denk auch daran, den Speicher, den du allozierst hast, wieder frei zu geben. Dies umfasst auch auf das Schließen aller geöffneter Dateien.