

Claudia Müller-Birn und Barry Linnert

Nichtsequentielle und verteilte Programmierung, SoSe 2020

Übung 1

TutorIn: Florian Alex
Tutorium Übung 05

Qianli Wang und Nazar Sopiha

21. April 2020

1 Aufgabe: Korrektheit

(10 Punkte)

Begriff(Korrektheit): Die Korrektheit eines Programms wird angenommen, wenn gesagt wird, dass das Programm in Bezug auf eine Spezifikation korrekt ist. Die funktionale Korrektheit bezieht sich auf das Eingabe-Ausgabe-Verhalten des Programms.

Vor dem Programmieren soll man zuerst ein Konzept entwickeln und an die Logik des Programms denken. Außerdem soll man Eindrücke haben, wie alle verwendeten Operationen auf Endeffekte der Ausgaben auswirken kann.

Während des Programmierens soll man das Programm mit richtiger Syntax der verwendeten Programmiersprache implementieren und auf die Kleinigkeiten achten, z.B. Coding Style (Tabs oder Leerzeichen überall eindeutig benutzen) oder gewisse Probleme von Befehlen vermeiden (z.B. bei `malloc()` können Memory Leaks entstehen u.s.w.). Darüber hinaus soll man auch Sequenz aller Operationen im Programm richtig einordnen, damit sie am Ende richtig ausgeführt werden können.

Nach dem Programmieren(bei der Ausführung):

1. Compiler oder Interpreter soll richtig den Code kompilieren oder interpretieren.
2. Linker soll ein Speicherabbild erstellen mit Maschinenbefehlen (ein ausführbarer Maschinencode) und initialisierten Daten.
3. Betriebssystem konfiguriert Ausführungsumgebung erfolgreich und Prozess außerhalb des verknüpften Speicherabbildes.
4. Hardware soll die Maschinenbefehle in der Ausführungsumgebung problemlos ausführen und Instruction Pointer automatisch inkrementieren.

Vor dem Anfang bis zum Ende sollte man immer an mögliche Fehler denken, den Code richtig testen und Warnings betrachten und eventuell richtig Debuggen, damit am Ende das erwartete Ergebnis rauskommen kann.

2 Aufgabe: Programmierung in C

(12 Punkte)

Betriebssysteme: Linux (als OS und ind Virtual Box) und Windows

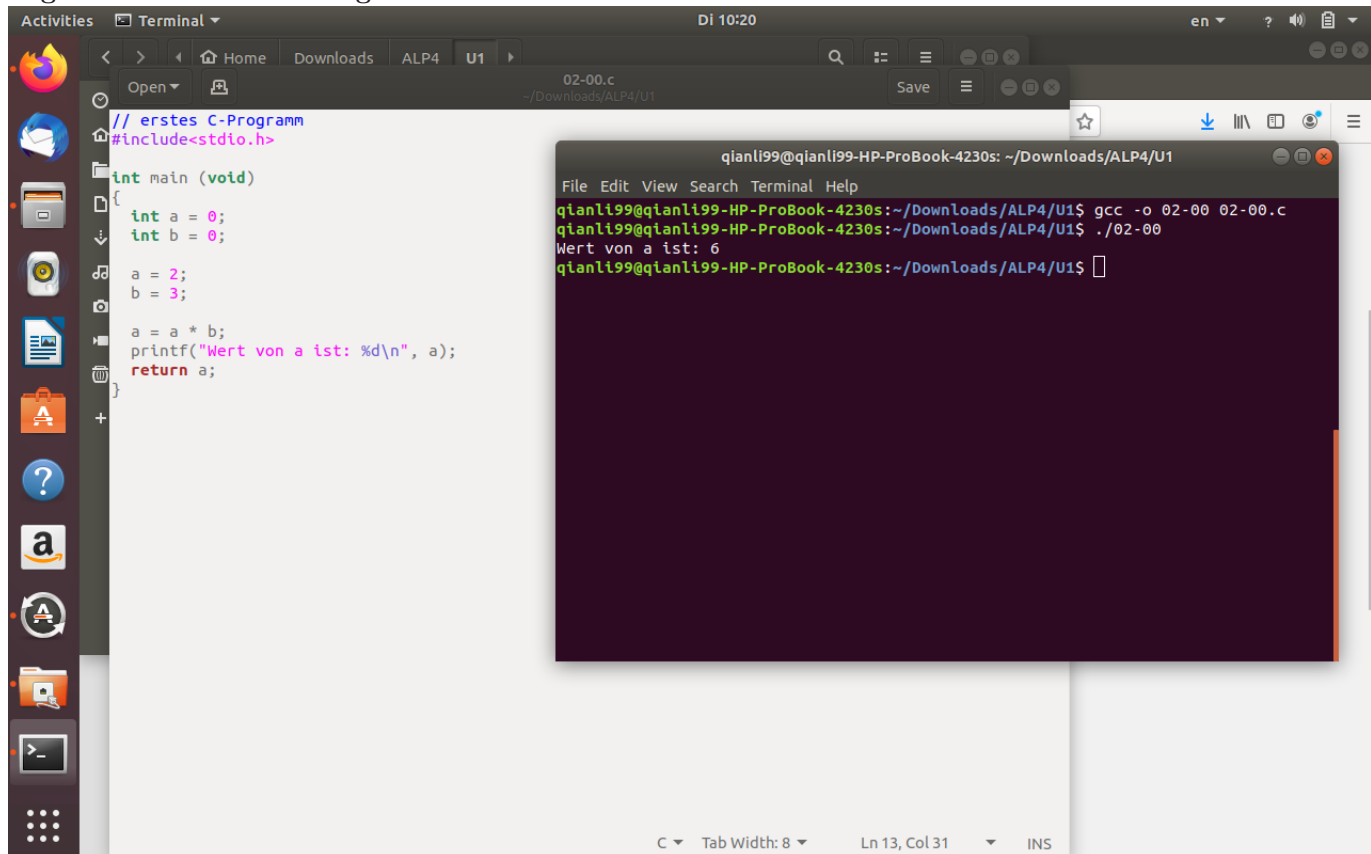
Compiler: gcc (ubuntu 7.5.0-3ubuntu1 18.04)

Text editor: Sublime Text, Notepad++, Qt

Fehlersuche: Warnings von Texteditor(für Syntaxfehler) oder printf()-Funktion, damit wir überprüfen können, dass die Zwischenergebnisse richtig sind.

Fehlerbehandlung: Terminal als Ausgaben auch für Warnings und Errors, Qt-Console, Qt-Debugging mit Stopp Punkten.

Ergebnisse der Ausführung:



The screenshot shows a Linux desktop environment with a file manager and a terminal window. The file manager displays a C program named `02-00.c` located in the directory `~/Downloads/ALP4/U1`. The code is as follows:

```
// erstes C-Programm
#include<stdio.h>

int main (void)
{
    int a = 0;
    int b = 0;

    a = 2;
    b = 3;

    a = a * b;
    printf("Wert von a ist: %d\n", a);
    return a;
}
```

The terminal window shows the execution of the program. The user runs `gcc -o 02-00 02-00.c` to compile the program and `./02-00` to run it. The output is `Wert von a ist: 6`.

```
qianli99@qianli99-HP-ProBook-4230s: ~/Downloads/ALP4/U1
File Edit View Search Terminal Help
qianli99@qianli99-HP-ProBook-4230s:~/Downloads/ALP4/U1$ gcc -o 02-00 02-00.c
qianli99@qianli99-HP-ProBook-4230s:~/Downloads/ALP4/U1$ ./02-00
Wert von a ist: 6
qianli99@qianli99-HP-ProBook-4230s:~/Downloads/ALP4/U1$
```

Activities Terminal DI 10:21 en ? ? ? ?

02-00.c
02-01.c
~/Downloads/ALP4/U1

Save

Open

Open

// C-Programm mit while-Schleife

```
#include<stdio.h>

int main (void)
{
    int a = 0;
    int b = 0;

    a = 2;
    b = 3;

    while (b > 0)
    {
        a += a;
        b--;
    }
    printf("Wert von a ist: %d\n", a);
    return a;
}
```

qianli99@qianli99-HP-ProBook-4230s: ~/Downloads/ALP4/U1

File Edit View Search Terminal Help

```
qianli99@qianli99-HP-ProBook-4230s:~/Downloads/ALP4/U1$ gcc -o 02-01 02-01.c
qianli99@qianli99-HP-ProBook-4230s:~/Downloads/ALP4/U1$ ./02-01
Wert von a ist: 16
qianli99@qianli99-HP-ProBook-4230s:~/Downloads/ALP4/U1$
```

C Tab Width: 8 Ln 18, Col 37 INS

Activities Terminal DI 10:23 en ? ? ? ?

02-02.c
~/Downloads/ALP4/U1

Save

Open

Open

// C-Programm mit Funktion

```
#include<stdio.h>

int foo (int a, int b)
{
    return a * b;
}

int main (void)
{
    int a = 0;
    int b = 0;

    a = 2;
    b = 3;

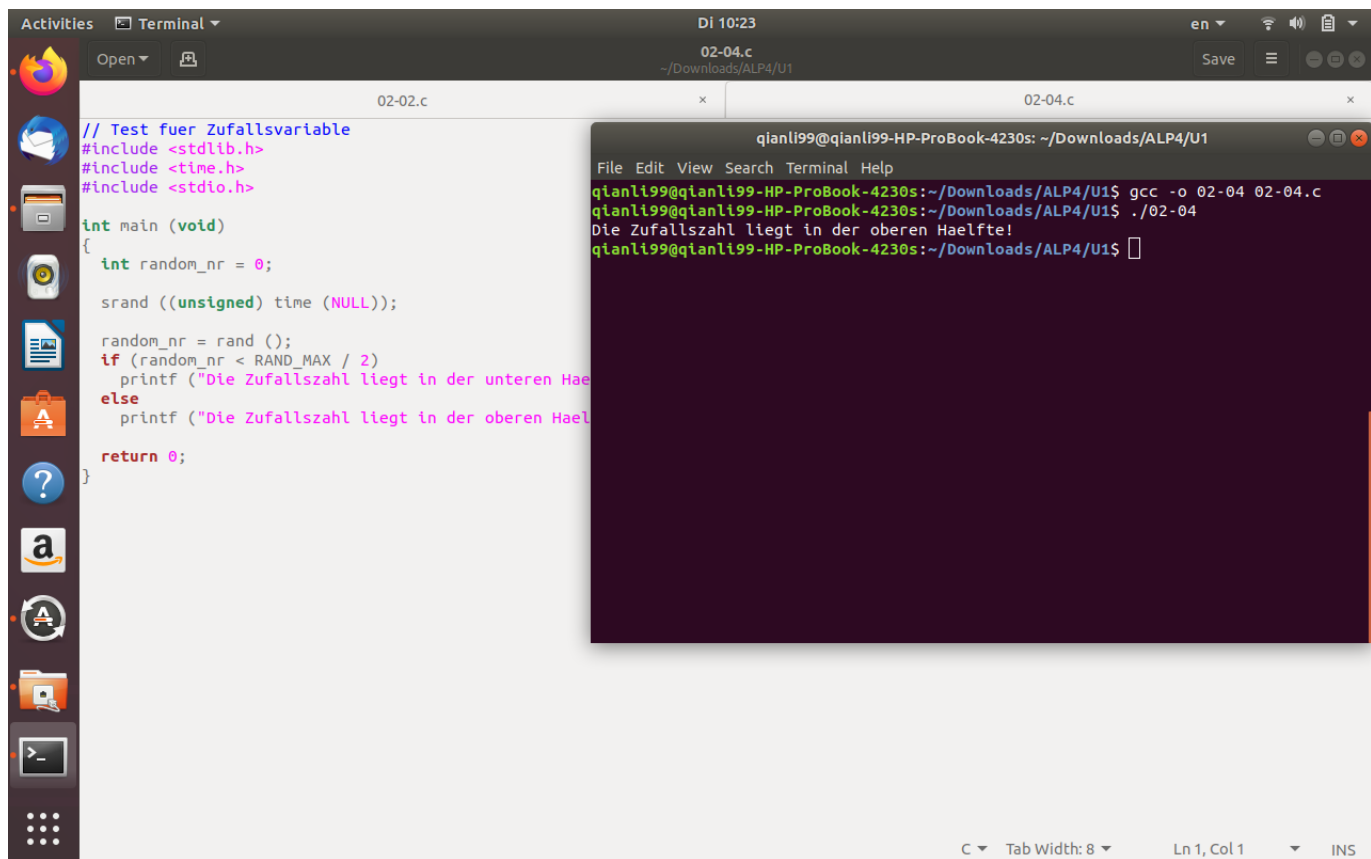
    a = foo (a, b);
    printf("Wert von a ist: %d\n", a);
    return a;
}
```

qianli99@qianli99-HP-ProBook-4230s: ~/Downloads/ALP4/U1

File Edit View Search Terminal Help

```
qianli99@qianli99-HP-ProBook-4230s:~/Downloads/ALP4/U1$ gcc -o 02-02 02-02.c
qianli99@qianli99-HP-ProBook-4230s:~/Downloads/ALP4/U1$ ./02-02
Wert von a ist: 6
qianli99@qianli99-HP-ProBook-4230s:~/Downloads/ALP4/U1$
```

C Tab Width: 8 Ln 20, Col 2 INS



3 Aufgabe: Performance

(8 Punkte)

1. **Für Android Programming** sollte man Interne (Logik wie Serveranfrage) und externe (wie User Interface Frame aufzeichnen) Prozesse unterscheiden, damit der User während Ausführung der schweren Befehle immer noch mit dem App kommunizieren kann.
2. **Sortieralgorithmen:** Insbesondere für Mergesort. Man kann mehrere Prozesse zur Verfügung stellen, dann kann man beim Teilen und Herrschen Zeit gewinnen, indem mehrere Prozesse eigene Listen bekommen und Listen sortieren(in denen die sortierten Elementen "ausgetauscht" werden).