Tutorium 3 - Programmieren 1

Eric Antosch

PR/01 - 30. April 2021

HAW HAMBURG

HAW HAMBURG

Inhaltsverzeichnis

HAW HAMBURG

Mini-Praktikum

...

HAW HAMBURG

Zusammenfassung

Sie haben schon gelernt...

- was Funktionen sind und wie man sie definiert,
- was Headerdateien sind, welche es gibt und wie man eigene erstellt,
- wo der Unterschied zwischen Call-By-Value und Call-By-Reference ist
- und was Rekursion ist und wie man diese erstellt.

HAW HAMBURG

Funktionen

Was ist eine Funktion?

Eine Funktion ist ein Block von Code, der durch einen Aufruf ausgeführt werden kann. Dabei besteht jede Funktion aus einem Funktionskopf und einem Funktionskörper. Im Funktionskopf wird der Name, der Typ des Rückgabewerts und die Parameter festgelegt, während im Funktionskörper der eigentliche Code vorgefunden werden kann.

Funktionen
Prototypen

Prototypen

Es ist ratsam, alle Funktionen in dem eigenen Code direkt unter den Includes und den Präprozessordirektiven mithilfe von Prototypen zu beschreiben. Der Prototyp ist im Prinzip nur der Funktionskopf ohne Funktionskörper.

HAW HAMBURG

Funktionen
Einfaches Beispiel: main()

```
1  int main(void){
2      return 0;
3  }
```

HAW HAMBURG

Funktionen
Einfaches Beispiel: plus(a,b)

```
1  int plus(int a, int b){
2      return a+b;
3  }
```

HAW HAMBURG

Headerdateien?

Was sind Headerdateien?

Headerdateien sind der zweite Dateientyp neben *.c-Dateien, die ansich jedoch keinen C-Code enthalten. Sie enthalten lediglich die Informationen über tatsächliche C-Dateien, sodass sie in den eigenen Code eingebunden werden können.

HAW HAMBURG

Headerdateien

Wichtige Headerdateien

Einige wichtige Headerdateien:

- stdio.h
- stdlib.h
- math.h
- conio.h

HAW HAMBURG

Value vs. Reference

Call-By-Value

Der Aufruf einer Funktion mit Parametern, die Variablen darstellen. Die Werte werden in der Funktion zur Verarbeitung von Anweisungen und Informationen genutzt.

HAW HAMBURG

Value vs. Reference

```
1  void tauschen(int a, int b){
2      int temp = a;
3      a = b;
4      b = temp;
5  }
```

HAW HAMBURG

Value vs. Reference

Call-By-Reference

Der Aufruf einer Funktion mit Parametern, die Zeiger auf Variablen darstellen. Die Speicheraddressen, die den Variablen zugeordnet sind, werden in der Funktion zur Verarbeitung von Anweisungen und Informationen genutzt. Es kann jedoch auch eine direkte Veränderung des Wertes aus der Funktion heraus geschehen.

HAW HAMBURG

Value vs. Reference

```
1  void tauschen(int *a, int *b){
2      int temp = *a;
3      *a = *b;
4      *b = temp;
5  }
```

HAW HAMBURG

Rekursion

Was ist Rekursion?

Als eine rekursive Funktion bezeichnet man eine Funktion, die sich in ihrer Ausführung selbst aufruft. Man kann solche Funktionen gut mit rekursiven Folgen vergleichen. Das nächste Folgenglied ist hierbei abhängig von den vorherigen Ergebnissen der Folge. Ein Anfang wird dabei meist als gegeben angenommen.

HAW HAMBURG

Rekursion

Stack

Zu beachten bei dem Erstellen von rekursiven Funktionen ist, dass ein sogenannter Stack existiert. Ruft eine Funktion sich selbst auf, dann wird zuerst die zuletzt aufgerufene Funktion bis zum Schluss ausgeführt. Nach dem Ende dieser Funktion, springt das Programm zurück in die Funktion, die die gerade beendete Funktion aufgerufen hat. Rekursion ist also nicht einfach nur eine andere Form von Iteration.

HAW HAMBURG

Rekursion

```
1  int addTillTen(int a){
2      if(a >= 10){
3          return a;
4      }
5      a++;
6      int b = addTillTen(a);
7      printf("%d\n", b);
8      return a;
9  }
```

Fragen?

HAW HAMBURG

Aufgaben

Funktionen

Aus welchen zwei Teilen besteht eine Funktion in ihrem Code? Nennen Sie ein entsprechendes Beispiel für beide Teile!

HAW HAMBURG

Aufgaben

Prototyp

Was ist der Prototyp einer Funktion und wofür wird er verwendet? Erstellen Sie eine Funktion und den entsprechenden Prototyp und stellen Sie diesen kurz vor!

HAW HAMBURG

Aufgaben

Headerdateien

Geben Sie in ihren eigenen Worten wieder, was eine Headerdatei ist! Welche Headerdateien sind schon bekannt? Geben Sie ein paar Beispiele für hilfreiche oder wichtige Funktionen aus diesen Dateien.

HAW HAMBURG

Aufgaben

Call-By-Value

Erklären Sie kurz, was eine Funktion ist, die auf dem Call-By-Value-Prinzip beruht. Erstellen Sie schnell eine Funktion, die dieser Regel gehorcht. Wofür sind diese Funktionen besonders geeignet?

HAW HAMBURG

Aufgaben

Call-By-Reference

Charakterisieren Sie nun das Call-By-Reference-Prinzip, indem Sie die Unterschiede zu Call-By-Value nennen und warum diese von Bedeutung sind. Schreiben Sie die gleiche Funktion aus der vorherigen Aufgabe nun auch mit Call-By-Reference. Ändert sich die Arbeitsweise ihrer Funktion und gibt es einen anderen Output? Wenn ja, warum?

HAW HAMBURG

Aufgaben

Rekursion

Was ist Rekursion und woran erkennt man Rekursion in einem Programm? Nennen Sie Beispiele von Rekursion aus der Mathematik. Schreiben Sie eine Funktion, die über Rekursion funktioniert. Wo liegt der Unterschied zwischen Iteration und Rekursion?

HAW HAMBURG

Aufgaben

Quersumme einer fünfstelligen Zahl

Schreiben Sie eine Funktion int quersumme(int n), die Ihnen die Quersumme einer fünfstelligen Zahl zurückgibt und geben Sie diese dann wieder in Ihrer Main-Funktion aus.

HAW HAMBURG

Aufgaben

Fakultät

Schreiben Sie eine Funktion, die, mithilfe von Rekursion, die Fakultät einer Zahl n berechnen kann.

HAW HAMBURG

Aufgaben

Dreiecke

Schreiben Sie ein Programm, welches zwei Dreiecke mit Seiten a_i, b_i, c_i nach ihrem Volumen sortiert. Um ein Dreieck mit den bestimmten Seiten auszurechnen bietet sich folgende Formel (Herons Formel) an:

$$A_i = \sqrt{p_i \cdot (p_i - a_i) \cdot (p_i - b_i) \cdot (p_i - c_i)}$$

Dabei gilt: $p_i = \frac{a_i + b_i + c_i}{2}$.

HAW HAMBURG

Mini-Praktikum

Mini-Praktikum

Erstellen Sie ein Programm, welches die folgenden Anforderungen erfüllt:

- Sie haben eine Main-Funktion, die sich als aller erste Funktion in ihren Programm befindet.
- Sie haben zwei weitere Funktionen:
  - Eine Funktion gibt Ihnen die Summe der Quadratzahlen bis zu einer Zahl n zurück.
  - Die zweite Funktion gibt Ihnen das n.te (oder auch m.te) Glied der Fibonacci-Folge aus.
- Beachten Sie, dass es manchmal sinnvoll sein kann, die Art, Funktionen zu schreiben, auf das Prinzip der Funktion anzupassen!