Imperative Programmierung

Übung 6: Arrays

Justin Kreikemeyer
Informatik, Uni Rostock

Fragen?

Leitfragen

- Was ist ein Array?
- Welches Problem löst ein Array?
- Wie geht das in C?

Empfehlung

- Programmieren lernt man nur durch selber Programmieren!
- Löst auch nach der Übung weiter so viele Programmieraufgaben, wie irgendwie möglich ($\rightarrow$ Praktikum/Selbststudium)
- Hier: Überblick (sehr viel auf einmal!), Lösen einiger Aufgaben, Klärung von Fragen

Erinnerung: Grundstruktur

```c
#include <stdio.h>                    // Inklusion von externem Quellcode
int main(int argc, char* argv[]) {    // <- argv ist ein Array!

  // Programmcode steht hier.         // Zwei Schrägstriche (//) kennzeichnen Kommentar

  return 0;                           // Rückgabewert an das Betriebsystem (0 = Erfolg)
}
```

- Beobachtung: Farbliche Markierung von speziellen Wörtern in C (je nach Editor anders)
- `#include <stdio.h>`: Einbinden des Quelltextes aus der Datei `stdio.h` (beinhaltet Prozeduren zur Ein- und Ausgabe)
- `argc, argv` Anzahl und Werte der Kommandozeilenargumente (jetzt genauer)

Bisher

- Aufgabe: Lese zehn ganze Zahlen ein.
- Variante 1: `int x1, int x2, int x3, ...`

```c
#include <stdio.h>
int main(int argc, char* argv[]) {
  int x1, x2, x3, x4, x5, x6, x7, x8, x9, x10;
  scanf("%d", &x1);
  scanf("%d", &x2);
  // ...
  return 0;
}
```

Ok, jetzt seid ihr dran: lest 100 ganze Zahlen ein!

Mit Array

- Aufgabe: Lese zehn ganze Zahlen ein.
- Variante 2: Array der Länge 10 und Schleife!

```c
#include <stdio.h>
int main(int argc, char* argv[]) {
  int x[10];
  for (int i = 0; i < 10; i++) {
    scanf("%d", &x[i]);
  }
  return 0;
}
```

Jetzt können wir auch ohne weiteres 100 Zahlen einlesen.

Konzept: Array

- Zusammenfassung von Werten in aneinanderhängendem Speicherbereich
- Nützlich für Arbeit mit Vektoren (1D), Matrizen (2D), Tensoren (nD), . . .
- Nützlich für Mengen, z.B. Menge an Zeichen = Zeichenkette (später)
- Nützlich für Bilder (2D)

mehr s. Whiteboard

Arrays in C

- `<typ> <name> "[" <anzahl> "]" [ "=" <initialisierung> ] ";"`
- Array repräsentiert immer Speicher (Referenz, Erinnerung: `scanf`)
- Arrays werden also mittels *call-by-reference* an Funktionen übergeben und ihre Werte können in diesen verändert werden!
- Länge der Arrays muss beim Programmstart feststehen
- Array kennt "seine" Länge nicht → separat speichern

```
int l = 10;                        // l muss bei Ausführung feststehen!
int x[l];                          // deklaration
int y[10] = { 0 };                 // Inline-Initialisierung aller Elemente auf 0
double z[3] = {1.0, 2.0, 3.0};     // Inline-Init. jedes einzelnen Elements
x[1] = 0;                          // Zuweisung an ein bestimmtes Element mittels []-Operator
double u = z[0] * 3.0;             // Verwendung innerhalb eines Ausdrucks mittels []-Operator
```

Zählung der Elemente beginnt bei 0! → Das letzte Element hat Index Länge-1!
Der C Compiler überprüft das nicht!

Gemeinsames Beispiel: Maximierender Index

Schreiben Sie eine Funktion, die den Index des Maximums einer Liste an Zahlen bestimmt!

Zeichenketten

- Zeichenketten in C = Arrays von `char`s
- Jeder `char` repräsentiert (id eines) ASCII-Zeichen(s)
- Arithmetik möglich, z.B. `'a' + 1` ist `'b'`
- Initialisierung mit doppelten Anführungsstrichen `""` möglich
- Verwendung z.B. bisher schon in `printf`
- Hilfreiche Funktionen in `<string.h>` definiert, z.B. `strlen` für Längenberechnung (wie geht das!?)

```c
char name[] = "Justin";
// alternativ: char name[] = {'J', 'u', 's', ...};
char firstLetter = name[0];
printf("Erster Buchstabe ist %c", firstLetter);
```

Was ist die Ausgabe des obigen Programms?

Fragen?

Aufgaben: Funktionen in C

Schreiben Sie die folgenden C-Programme und Funktionen[1]! Nutzen Sie dazu die Konzepte aus dieser Übung und das Cheat Sheet!
Bearbeitungszeit: bis 10 Minuten vor Schluss. Dann Besprechung von häufigen Problemen.

[1] Weitere Aufgaben können jederzeit beim Übungsleiter erfragt werden.

Programmieren mit Arrays I

ArrayRead Schreiben Sie eine Funktion, die vom Nutzer *length* elemente Abfragt und diese nacheinander in einem Array der Länge *length* speichert.

ArrayPrint Schreiben Sie eine Funktion, die ein übergebenes Array in einem gut lesbaren Format auf dem Bildschirm ausgibt.

ArrayMax Schreiben Sie eine Funktion `int max(int arr[], int length)`, die für ein übergebenes Array das Maximum bestimmt und zurückgibt!

ArrayEvenSum Schreiben Sie eine Funktion `int evenSum(int arr[], int length)`, die für ein übergebenes Array die Summe aller geraden Einträge berechnet und zurückgibt!

Programmieren mit Zeichenketten

MakeLowerCase Schreiben Sie ein Programm, das eine Zeichenkette aus Großbuchstaben in eine Zeichenkette aus Kleinbuchstaben umwandelt! Hinweis: Berechnen Sie die Differenz zwischen verschiedenen Parren, e.g., `'a'-'A'` und `'l' - 'L'`. Was fällt auf?