Arrays und Array Slicing

Mit Arrays arbeiten, leicht gemacht.

Mark Zeman, Philip Stark

FHNW - Compilerbau

18.11.2014

- `http://github.com/hellerbarde/cpib`

Übersicht

Arrays

Wir definieren Arrays so:

- Multidimensional
- Simple Types

Die Deklaration sieht mittlerweile so aus:

```
var a:arr (5) int;
```

und der Zugriff so:

```
a[5];
```

Array Slicing

- Unorthodox
- Expressive Syntax
- Übersichtlicher Array-handling Code

Ein Slice hat die folgende Syntax: `a[2..4];`

Gedanken zur Syntax der Deklaration

Mehrere Versuche, bis wir zur definitiven Syntax kamen:

- `var a:TYPE[LENGTH][DIMENSION];`
- `var b:arr (array or type) LENGTH;`
- `var c:arr LENGTH (array or type);`

Schlussendlich:

- `var d:arr (LENGTHS) type;`

Gedanken zur Syntax der Initialisierung und des Zugriffs

- Immer vollständig initialisiert
- Syntaktischer Zucker um Arrays zu “nullen”
- Kann genutzt werden um beliebig zu füllen

Die Syntax für die Initialisierung und den Zugriff sieht so aus:

- `a init := [0,1,2,3]`
- `b init := fill 5`
- `c init := a[3]`

Gedanken zur Syntax der Array-Slices

- `b := a[2..4]`

Wir wollten möglichst grosse Ähnlichkeit zum Zugriff und zudem wollten wir das – nicht überladen, also haben wir uns für `..` entschieden.

Lexikalische Erweiterungen

- Nur wenig zwingend nötig
- Fill für den syntaktischen Zucker
- DOTDOT damit wir Minus nicht überladen müssen

Pattern	Token
`[`	LBRACKET
`]`	RBRACKET
`fill`	FILL
`array (int) type`	(ARRAY,Length,Type)
`..`	DOTDOT

Grammatikalische Produktionen

Matrix Multiplikation

- Beispiel beispiel

```
Fooobar baz
program
functions and stuffs
declarations and so on
```

Datensatzauswertung

- explanation of the example

```
program bla
```