

# Kommandozeilenargumente

>> Bisher haben alle unsere Programme mehr oder weniger gleich begonnen.

```
int main(void) {
```

Da wir Benutzereingaben durch Eingabeaufforderungen im Programm gesammelt haben, mussten wir diese Deklaration von *main()* nicht ändern.

Wenn wir möchten, dass man einem Programm Daten schon vor der Ausführung übergeben kann, benötigen wir eine neue Form.

## Neue Deklaration von *main*

Um sogenannte **Kommandozeilenargumente** verarbeiten zu können, deklarieren Sie *main* wie folgt:

```
int main(int argc, string argv[])
```

Mit diesen beiden speziellen Argumenten können Sie ermitteln, **welche Argumente** (oft auch als „Parameter“ bezeichnet) beim Aufruf des Programms auf der Kommandozeile eingegeben wurden bzw. um **wie viele Argumente** es sich handelt.

**argc**: *argument count*

**argv**: *argument vector*

**argc** Diese Variable vom Typ Integer speichert die Anzahl der Kommandozeilenargumente, die bei der Ausführung des Programms eingegeben wurden.

`./greedy` => argc ist 1

`./greedy 1024 cs50` => argc ist 3

## **argv**

Dieses Array von Strings enthält den Text, der in der Kommandozeile eingegeben wurde, als das Programm ausgeführt wurde.

Das erste Element von argv befindet sich immer in `argv[0]`.

Das letzte Element von argv befindet sich immer in `argv[argc-1]`.

Sehen Sie, warum?

**argv** Nehmen wir an, der Benutzer führt das Programm greedy wie folgt aus:

```
./greedy 1024 cs50
```

```
argv[0] = "./greedy"
```

```
argv[1] = "1024"
```

```
argv[2] = "cs50"
```

```
argv[3] = ???
```

## Strings in Zahlen umwandeln

```
#include <stdlib.h>

int main(int argc, char *argv[]) {
    if (argc > 1) {
        int zahl = atoi(argv[1]);
        // Jetzt können Sie mit 'zahl'
        // als Integer arbeiten.
    }
    // Rest des Programms...
}
```

atoi(...): „ASCII to Integer“

Alternative strtol(...): „String to Long“

**EXTRAS IN 3 MINUTEN**  
FRAGEN – ANTWORTEN – RÄTSEL  
UND KURZE ZUSAMMENFASSUNG



- F1: Was passiert, wenn man willkürlich Argumente per Kommandozeile an ein Programm übergibt? Wie gehen Programme üblicherweise mit Kommandozeilenargumenten um?
- F2: Könnte man die *main*-Funktion zur Verarbeitung von Kommandozeilenargumenten auch mit `int main(int hello, string world[])` deklarieren?
- F3: Was ist zu beachten, wenn man dem Nutzer per Kommandozeile *int*-Werte als Argument übergeben lassen möchte?
- F4: Welche Zahl enthält der Argumentcount *argc*, wenn ein Programm mit zwei Kommandozeilenargumenten aufgerufen wird?

argc enthält die  
Zahl der Argumente  
(inkl. Programmname)

Alle Argumente werden  
als Strings gespeichert

argv ist ein  
Array von Strings,  
das die Argumente enthält

String-zu-Zahl Konvertierung  
mit *atoi()* oder *strtol()*

argv[0] enthält immer  
den Programmnamen

Arraygrenze argc-1  
nicht überschreiten!