Prozedurale Programmierung – Übung 9

WS 2022/23 *Prof. Dr. Johannes Jurgovsky* Hochschule **Rosenheim** University of Applied Sciences

In der Community wird ein (fast leeres) Projekt in der Datei „mainparams.zip“ bereitgestellt.

Aufgabe 1

Entwickeln Sie ein Programm, das eine beliebige Anzahl von Wörtern aus der Kommandozeile (! – **nicht** über scanf()!) einliest und eine Statistik über die Häufigkeit des Vorkommens der einzelnen Buchstaben erstellt.

Aufruf über Kommandozeile:
(oder während der Entwicklung über Projekteinstellungen: Run->Edit Configurations… -> Program arguments)

Windows:

```
C:\... >stats argaaa argbbb AXY$$Z
```

Ubuntu / MacOS:

```
...$ ./stats argaaa argbbb AXY$$Z
```

Ausgabe:

```
stats argaaa argbbb AXY$$Z
Anzahl der Buchstaben der Parameter:
A: 1 X: 1 Y: 1 Z: 1 a: 5 b: 3 g: 2 r: 2
```

Anmerkungen:

- das Programm soll prinzipiell für beliebige der 256 ASCII-Zeichen funktionieren, sofern diese alphanumerisch sind (verwenden Sie zur Prüfung die Funktion `isalnum(..)` aus der `ctypes.h`)
- die Statistik soll in einem `int` array mit fester Größe `BINS=256` gespeichert werden.
- um das Programm auf einfache Weise testen zu können, sollten Sie zunächst Befehlszeilenargumente in den Projekteinstellungen eintragen. Gehen Sie im Hauptmenü von CLion auf Run->Edit configurations… und wählen Sie die Konfiguration „stats“ aus. Tragen Sie nun unter „Program arguments“ beliebigen Text ein, den Sie an ihr Programm beim Start übergeben wollen.
  Alternativ: Kompilieren Sie ihr Programm manuell über die Eingabeaufforderung/Shell und führen Sie es dort mit den gewünschten Argumenten aus.

Aufgabe 2

Entwickeln Sie ein Programm, das eine beliebige Anzahl von Wörtern aus der Kommandozeile einliest und diese in umgekehrter Reihenfolge wieder ausgibt. Es soll dynamische Speicherverwaltung verwendet werden.

Aufruf über Kommandozeile:

```
…$ ./stats argaaa argbbb AXY$$Z
```

Ausgabe:

```
argaaa argbbb AXY$$Z
AXY$$Z argbbb argaaa
```

Kopieren Sie die Wörter aus der Kommandozeile in ein Feld von Zeichenketten. Der **Speicher** des Felds soll **dynamisch** in der aktuell erforderlichen Größe alloziert werden. Geben Sie den Text aus. Invertieren Sie dann die Reihenfolge der Wörter und geben Sie den Text erneut aus. Geben Sie sämtlichen dynamisch allozierten Speicher wieder frei.

Implementieren Sie dazu folgende Funktionen:

```
// gibt einen Text ((Feld von Zeichenketten) mit nrWords Worten aus
void printText(char *text[], int nrWords);

// Invertiert in einem Text (Feld von Zeichenketten) die Reihenfolge
// der Wörter; die Invertierung erfolgt ausschließlich durch Vertauschung
// von Zeigern, nicht durch umkopieren der Wörter!
void reverseText(char *text[], int nrWords);

// gibt den Speicherplatz für die Wörter des Textes wieder frei
void deleteText(char *text[], int nrWords);
```