# KSP Aufgabe 2

1. Ab dieser Aufgabe haben Sie einen Assembler zur Verfügung, damit das lästige Kodieren der Befehle nicht mehr per Hand ausgeführt werden muss. Denken Sie daran, dass auch das Programm `nja` ausführbar sein muss! Schreiben Sie die drei kleinen Testprogramme aus der vorigen Aufgabe in jeweils eine Datei, die die Endung `.asm` haben sollte. Lassen Sie die Programme assemblieren; die entstehenden Dateien sollten die Endung `.bin` haben. Inspizieren Sie diese Binärdateien mit dem Kommando `hexdump -C`. Versuchen Sie genau zu erklären, wie die beobachtete Ausgabe zustande kommt! Studieren Sie dazu die Beschreibung des Ninja-Binärformats.
2. Jetzt ändern Sie Ihre VM vom vorigen Aufgabenblatt so ab, dass das Binärprogramm aus einer Datei in den Programmspeicher geladen wird.

a) Der Name des zu ladenden Programms soll als Kommandozeilenargument übergeben werden.

b) Bevor Sie etwas mit dem Inhalt der Datei tun können, müssen Sie die Datei `öffnen`:

```
FILE *fopen(const char *path, const char *mode);
```

c) Was Sie mit dem Inhalt einer Ninja-Binärdatei tun müssen, um das darin enthaltene Programm starten zu können, steht ebenfalls in der Beschreibung des Ninja-Binärformats .

d) Sie werden die Funktion `fread()` zum Lesen aus einer Datei brauchen:

```
size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream);
```

e) Ebenso brauchen Sie die Funktion `malloc()` zur Speicheranforderung:

```
void *malloc(size_t size);
```

f) Wenn Sie alle Informationen aus der Datei verwertet haben, wird die Datei wieder geschlossen:

```
int fclose(FILE * fp);
```

1. Ergänzen Sie nun Ihre VM auf die Version 2, in dem Sie die neuen Instruktionen aus VM Instruktionen implementieren. Eine Diskussion von globalen Variablen und den zugehörigen Instruktionen finden Sie hier, die Diskussion von lokalen Variablen und Stackframes mit ihren Instruktionen gibt's hier.
2. Prüfen Sie nun das Funktionieren Ihrer VM durch Assemblieren und Ausführen der drei kleinen Programme aus Aufgabenteil 1 sowie der beiden Testprogramme prog1.asm und prog2.asm. Es wird dringend empfohlen, mindestens fünf weitere selbstgewählte Berechnungen im Stackmaschinen-Assembler zu programmieren und ausführen zu lassen. Sie sollten in der Lage sein, zu jedem Zeitpunkt der Ausführung den Stack aufzeichnen zu können!

3. Und hier wie immer die Referenzimplementierung: njvm

*Table 1. VM Instruktionen*

| Instruktion | Opcode | Stack Layout |
|---|---|---|
| `pushg <n>` | 11 | `... -> ... value` |
| `popg <n>` | 12 | `... value -> ...` |
| `asf <n>` | 13 | |
| `rsf` | 14 | |
| `pushl <n>` | 15 | `... -> ... value` |
| `popl <n>` | 16 | `... value -> ...` |

# Hinweise Aufgabe 2

1. Alle benötigten Bibliotheksfunktionen kann man im Online-Manual nachschlagen. Das ruft man mit dem Kommando `man` auf. In diesem Manual stehen auch alle Kommandos drin, die das System kennt. Wenn man also z.B. nicht weiß, wie das Manual funktioniert: `man man` hilft. Das Manual ist sehr "dicht" geschrieben; in jedem Halbsatz stehen mehrere wichtige Informationen drin. Studieren Sie deshalb die Manual-Einträge genau; nur überfliegen reicht nicht!
2. Vergessen Sie nicht, alle Rückgabewerte zu prüfen; die aufgerufene Funktion könnte aus diversen Gründen fehlgeschlagen sein (mögliche Ursachen stehen ebenfalls im Manual).