Praktikum 2

1. Aufgabe

Folgende Fragen dienen neben der Vorlesung als Vorbereitung für den Ilias-Test. Nutzen Sie zur Lösung auch die Unix-Dokumentation von der Laborhomepage.

1. Der Aufruf `int main(int argc, char* argv[])` kennt noch einen dritten Parameter.
   Wie lautet diese Parametervereinbarung?
   Welche Inhalte werden darüber erreicht?
   Welche zweite Möglichkeit gibt es noch, um auf die gleichen Inhalte zuzugreifen?
2. Skizzieren Sie den Aufbau der Argumenten- bzw. der Umgebungsvariablenliste.
3. Jedes Programm in der Ausführung hat Umgebungsvariablen. Mit welchen C-Aufrufen kann man sie setzen, ändern, löschen oder abfragen? Nennen Sie zusätzlich noch einige wichtige Umgebungsvariablen.
4. Sie haben folgenden Aufruf: `./a.out eins zwei >file drei`
   Wie behandelt die Shell die Ausgabenumlenkung und welcher Wert steht in argc?
5. Sie haben folgenden Aufruf: `./a.out 3 * 4`
   Wie behandelt die Shell die Aufrufparameter und welcher Wert steht in argc?
6. Was passiert, wenn ein Elternprozess einen Kindprozess erzeugt, und wie kann ich feststellen, ob ich mich im Eltern- bzw. Kindprozess befinde?
7. Was erbt ein Kindprozess vom Elternprozess beim fork()-Aufruf, und was bleibt beim exec()-Aufruf erhalten?
8. Können nach dem Aufruf von fork() der Elternprozeß und der Kindprozeß noch Variablen oder Daten im jeweils anderen Prozeß manipulieren?

2. Aufgabe: Behandlung von Argumenten beim Programmaufruf

Schreiben Sie ein C-Programm, das einen einfachen Taschenrechner simuliert. Es sollen nur die Operationen +, -, * und / für Zahlen vom Typ `long` ausgeführt werden. Bei der Division soll einfachheitshalber eine Integerdivision durchgeführt werden. Die Eingabe der Zahlen und Operationssymbole soll beim Programmstart erfolgen.

Die Abarbeitung der Operationen erfolgt von links nach rechts.

Ein Aufruf könnte z.B. lauten: `./a.out 3 + 5 – 2 x 5 / 2`

Zur Konvertierung der Zahlen benutzen Sie bitte die Funktion strtol().Informieren Sie sich bitte genau über diese Funktion, damit Sie in der Lage sind, auch eine Null als Zahlenwert zuzulassen. Bei einer fehlerhaften Konvertierung oder unzulässigen Option geben Sie genau die Stelle aus und beenden das Programm.

Beispielhafte Ausgabe

Falls das Programm mit zu wenig Parametern aufgerufen wird, dann geben Sie eine Fehlermeldung mit einen Beispielaufruf aus.

Beispiel einer fehlerhaften Eingabe: `./a.out 3 + 8 – 0m`

Die Ausgabe könnte z.B. dann folgendermaßen aussehen:

Aufruf: `./a.out <zahl><op><zahl> ...`

```
Keine gültige Ganzzahl
Parameter 6 falsch (0m)
```

3. Aufgabe: Ausgabe und Setzen von Umgebungsvariablen per C-Programm

Schreiben Sie ein C-Programm, das zuerst eine Liste der Umgebungsvariablen ausgibt und anschließend die Umgebungsvariable `BSPRAKTIKUM` erzeugt und darin den Text `ISTSUPER` speichert. Anschließend liest man über die Tastatur eine Umgebungsvariable ein und prüft, ob sie existiert. Wenn ja, ist der Inhalt der Variablen auszugeben, sonst eine entsprechende Fehlermeldung.

4. Aufgabe: Prozessverwaltung (Schwierigkeitsgrad: leicht)

Schreiben Sie ein C-Programm, in dem der Vaterprozess per Laufschleife drei Prozesse erzeugt, die zwanzig mal eine 1 (erster Prozess), eine 2 (zweiter Prozess) und eine 3 (dritter Prozess) ausgeben.

Lassen Sie die Kindprozesse bei jedem Schleifendurchgang 1 Millisekunde warten. Der Vaterprozess soll per `waitpid()` auf das Ende der drei Kindprozesse warten und anschließend eine Abschlusssmeldung ausgeben.

Ändern Sie ihr Programm, in dem Sie nach jeder Zahlenausgabe eine Pufferentleerung von `stdout` programmieren. Erklären Sie die unterschiedlichen Ergebnisse.

So könnte z.B. die Ausgabe aussehen:

```
0102102102102102102102102102102102102102102102102102102102212
```

Hinweise

Informieren Sie sich zu den Befehlen `environ`, `strol`, `fork`, `waitpid`, `usleep`, `fflush`.

Lernziele

Unix-Grundlagen, Kommandozeilenargumente, Umgebungsvariablen, Prozesse