Vorlesung Betriebssysteme
Wintersemester 2021/2022

Prof. Dr. Lars-Olof Burchard
Hochschule Darmstadt

Aufgabe 1 (VM Setup, Systemaufrufe und Informations-beschaffung aus dem Betriebssystem)

Sie sollen für die erste Praktikumsaufgabe eine eigene virtuelle Maschine mit Linux installieren, so dass Sie darin ein einfaches C/C++ Programm erstellen, kompilieren und starten können. Das Programm soll Systemaufrufe verwenden und sein Verhalten zur Laufzeit beobachtet werden.

Teil 1.

Installieren Sie auf Ihrem eigenen Notebook/Laptop/Rechner dazu zuerst die Software VirtualBox (oder VMWare Player, VMWare Fusion, etc., und erzeugen darin dann eine virtuelle Maschine mit einer **Linux** Distribution Ihrer Wahl. Geeignet ist zum Beispiel Debian oder Ubuntu Linux. In diesem System sollen die weiteren Praktika bearbeitet und die Lösungen lauffähig präsentiert werden können.
Hinweis: Dual-boot funktioniert auch. Windows/MacOS als Plattform können in dieser Lehrveranstaltung allerdings **nicht** verwendet werden.

Damit Sie mit der virtuellen Maschine arbeiten können, müssen Sie sicherstellen, dass folgende Software installiert und lauffähig ist:

- C/C++ Compiler (i.d.R. gcc bzw g++)
- Debugger (beliebig, z.B. gdb oder zusätzlich eine grafische Oberfläche z.B. Eclipse)
- Editor (beliebig)

Teil 2.

Schreiben Sie dann ein einfaches C/C++ Programm, das dynamisch Speicher allokiert und diesen Speicher dann anspricht. Dabei sollen Sie Informationen über CPU-Nutzung, maximal nutzbaren Hauptspeicher sowie maximale Stackgröße des laufenden Programms ermitteln und ausgeben. Dieses Programm soll von der Kommandozeile gestartet werden.

Dazu sollen Sie folgendes implementieren:
1. Schreiben Sie eine Funktion **`funcMem`**, die einen großen Array von Integern zuerst allokiert und dann jedes einzelne Arrayelement anspricht (z.B. durch eine Zuweisung).
2. Schreiben Sie eine rekursive Funktion **`funcRec`**, die im Wesentlichen sich selbst aufruft.

Beide Funktionen sollen geeignet in Ihrem Programm ausgeführt werden, und bei jedem *n*-ten Ansprechen eines Arrayelements (*n* geeignet gewählt, z.B. 1000) und bei jedem *m*-ten Aufruf von **`funcRec`** (*m* geeignet gewählt, z.B. 1000) sollen mithilfe eines Systemaufrufs **`getrusage`** die folgenden Informationen auf der Konsole (mittels **`printf`** oder **`cout`**) ausgegeben werden:

1. Die verbrauchte *user CPU time*
2. Die verbrauchte *system CPU time*
3. Der insgesamt verwendete Speicher des laufenden Programms in der Funktion **`funcMem`**
4. Die aktuelle Größe des Stacks in der Funktion **`funcRec`**. Achtung: Unter Linux kann die Funktion **`getrusage`** zur Ermittlung der

Stackgröße *nicht* verwendet werden, Sie müssen sich also einen eigenen Mechanismus überlegen.

Finden Sie durch geeignete Ausgaben Ihres Programms dann die maximale Größe des verwendbaren Hauptspeichers sowie die maximale Größe des Stacks Ihres Programms experimentell heraus.

Nutzen Sie dann in Ihrem Programm den Systemaufruf **`getrlimits`**, um Ihre experimentell ermittelten Ergebnisse für maximal nutzbaren Hauptspeicher sowie maximale Größe des Stacks zu bestätigen.

Dokumentieren Sie:

- die maximal nutzbare Größe an Hauptspeicher sowie die maximal nutzbare Stackgröße, getrennt als experimentell ermittelte bzw. aus dem Betriebssystem über **`getrlimits`** ausgelesene Größen
- die von Ihrem Programm benötigte *user* bzw. *system CPU time* und deren Verhältnis in Abhängigkeit von *n* bzw. *m* (s.o.). Dazu können Sie eine Tabelle oder ein Diagramm verwenden.

Speichern Sie die Ergebnisse in einer Datei und laden Sie diese Datei zusammen mit Ihrem Quellcode in Ihr Repository auf gitlab.

Hinweis: Sie müssen sich zuerst über die Systemaufrufe **`getrusage`** und **`getrlimits`** informieren, damit Sie diese in Ihrem Programm nutzen können. Wie bei anderen Systemaufrufen auch können Sie Informationen über diese Funktionen auf der Linux Kommandozeile durch Eingabe von '**`man getrlimits`**' erhalten. Alternativ nutzen Sie eine Suchmaschine.