OPERATING SYSTEMS BEISPIEL 2

Aufgabenstellung – mygzip

Schreiben Sie ein Programm, das Eingaben mit `gzip(1)` komprimiert.

```
SYNOPSIS
    mygzip [file]
```

Anleitung

Das Programm erstellt zwei Pipes und führt zwei mal *fork(2)* aus (und erzeugt damit zwei Kinder – nicht Kind und Enkelkind). Das erste Kind biegt *stdin* auf die erste und *stdout* auf die zweite Pipe um und startet mit *execlp(3)* das Programm `gzip(1)` mit dem Parametern `-cf`. Der Elternprozess liest die zu komprimierenden Daten von *stdin* ein und schreibt sie über die erste Pipe zum `gzip`-Prozess. Das zweite Kind liest über die zweite Pipe vom `gzip`-Prozess und schreibt die gelesenen Daten in die Datei `file` bzw. auf *stdout*, wenn keine Datei angegeben wurde.

Achten Sie darauf, das Sie die Dateien binär öffnen.

Richtlinien

Beachten Sie unbedingt auch die *Richtlinien für die Erstellung von C-Programmen* ("Coding Guidelines") in TUWEL, sowie die folgenden allgemeinen Hinweise zur Beispielgruppe 2!

Dokumentation. Insbesondere ist es ab dieser Beispielgruppe notwendig, die Dokumentation in Doxygen zu führen. Eine kurze Einführung haben wir Ihnen im OSUE-Wiki bereitgestellt. Es muss zumindest das HTML-Output generierbar sein. Bitte dokumentieren Sie ausnahmslos alle Funktionen (auch `static`-Funktionen; siehe `EXTRACT_STATIC` in der Datei `Doxyfile`). Achten Sie weiters darauf, dass nach außen hin sichtbare Funktionen (exportierte Funktionen) in der Header-Datei und lokale (`static`) Funktionen nur in der C-Datei dokumentiert werden. Sie sollten auch Ihre Typen (insbesondere `structs`), Konstanten und globale Variablen dokumentieren.

Argumentbehandlung. Vergessen Sie auch bei diesem Beispiel nicht auf die Argumentbehandlung (auch bei einem Programm welches keine Argumente erhält ist eine Argumentbehandlung durchzuführen)!

Pipes. Falls Sie Pipes erzeugen, sollte das wie im Übungsskriptum beschrieben geschehen.

Ressourcen. Alle Ressourcen (wie z.B. Pipes) müssen ordnungsgemäß vor Terminierung entfernt werden.

Terminierung. Die Terminierung aller Kindprozesse ist sicherzustellen, ohne `kill(2)` oder `killpg(2)` zu verwenden. Der Exit-Status beendeter Kindprozesse muss vom Vaterprozess abgeholt werden (`wait(2)`, `waitpid(2)`, `wait3(2)`).

Signalbehandlung. Eine Signalbehandlung ist für diese Beispielgruppe (mit Ausnahme von einem Beispiel, bei dem explizit eine Signalbehandlung gefordert wird) nicht erforderlich!