Übung zur Vorlesung
Safety Critial Systems I
Wintersemester 04/05

AGBS
Jan Peleska
Tobias Hartmann

Serie 2

Design Criteria for Safety Critical Systems

In der Vorlesung wurde das Thema „Memory Leaks" behandelt. Hierbei wurden Probleme besprochen, welche bei der Verwendung von malloc() und free() auftauchen können.

Als eine praktikable Variante wurde hier die Möglichkeit einer eigenen Speicherverwaltung (also ein Verzicht auf ein Kernelverwaltetes malloc() und free()) vorgestellt.

Die Programmiersprache „Python"[1] verwendet ein solches Modell. Der Programmierer bekommt im Regelfall hiervon allerdings nichts mit, da eine automatische „Garbage Collection"[2] die nicht mehr benötigten Objekte verwerfen kann.

Aufgabe:

Findet heraus, wie Python Objekte anlegt, Speicher reserviert und nach Nichtgebrauch wieder verwirft.

Hilfreich ist hierzu die Dokumentation, die man auf den Seiten des Python-Projektes [3] finden kann.

Schreibt hierzu einen kleinen Aufsatz, wie dies funktioniert. Dieser sollte zwei Seiten nicht unterschreiten.

Wichtige Hinweise zur Bearbeitung dieses Zettels:

Auf folgende Dinge wird Wert gelegt, ihr solltet dies also beachten:
- Genaue und (*sehr*) ausführliche Dokumentation!
- Welche Pythonversion wurde untersucht?!
- Genaue Quellenangabe (Links mit Datum!)

Abgabe bis zum 10. Februar 2005

Weitere Hinweise:

[1] – Mehr zu der Sprache Python findet man unter:
generell: http://www.python.org
Dokumentation (Version 2.3.4): http://www.python.org/doc/2.3.4/index.html
Speziell hier der Punkt „Library Reference (keep this under your pillow)“.

[2] – diese kann man explizit ausschalten: gc.disable().
Hierzu möchte man sich vielleicht auch das „GarbageCollector“ Modul von Python ansehen. Dies ist verfügbar (bzw. kann verwendet werden), wenn Python mit dem „optional cyclic garbage detector“ übersetzt wurde (dies ist der default-Zustand).
Weitere Informationen zu diesem Modul findet man (z.B.) hier:
„gc -- Garbage Collector interface“: http://www.python.org/doc/2.3.4/lib/module-gc.html

[3] – Diese Links könnten hilfreich sein:

Generell:
http://www.python.org/doc/

Supporting cyclic garbage collection
http://www.python.org/doc/2.3.4/ext/node24.html

Reference Counts
http://www.python.org/doc/2.3.4/ext/refcounts.html

Python/C API Reference Manual / Overview
http://www.python.org/doc/2.3.4/api/memoryOverview.html

Sonst: siehe auch [1], [2]