Vorlesung Betriebssystemarchitektur WS 07/08
Aufgabenblatt 1 vom 1. November 2007
(Vorstellung der Lösungen bei den Tutoren bis zum 14.11.07)

Aufgabe 1.1: (15 Punkte)
Beantworten Sie folgende Fragen und erläutern Sie die Antworten Ihrem Tutor.

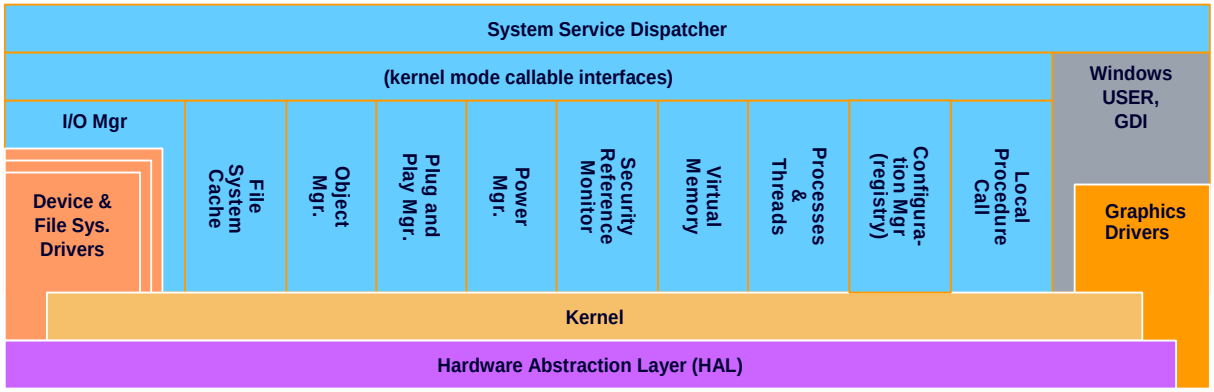
- Nennen und begründen Sie vier Ziele beim Entwurf eines Betriebssystems!
- Nennen Sie Beispiele für Betriebssystemfunktionalität die im Kernel- bzw. im User-Mode implementiert werden müssen!
- Was sind Vor- und Nachteile der Trennung von Kernel- und Usermode?

Aufgabe 1.2: (10 Punkte)
Erklären Sie Ihrem Tutor die Begriffe *Programm*, *Prozess* und *Thread*! Worin bestehen die Unterschiede zwischen diesen Begriffen.

Aufgabe 1.3: (20 Punkte)
Beantworten Sie folgende Fragen und erläutern Sie die Antworten Ihrem Tutor.

- Nennen und beschreiben Sie fünf Windows-Kernel Komponenten!
- Nennen und beschreiben Sie die Userspace Subsysteme von Windows NT/2000!

Aufgabe 1.4: (25 Punkte) -- WRK Dokumentation
Das folgende Bild zeigt den in der Vorlesung vorgestellten Aufbau des Windows-Kernels.



Machen Sie sich mit der Dokumentation des Windows Research Kernels (WRK) vertraut und beantworten Sie die folgenden Fragen!

- Wie lässt sich die Verzeichnisstruktur in `base/ntos` des WRK auf den Kernelaufbau abbilden?
- Welche Teile des Windows-Kernels sind nicht im WRK enthalten?
- In welcher Funktion werden die Kernelmodule initialisiert? Wie heißen die Initialisierungsfunktionen der Module?

- Laden Sie den WRK von https://maniac.hpi.uni-potsdam.de/ herunter und übersetzen Sie den Kernel!

Hinweis: Die Dokumentation für den Windows Research Kernel ist innerhalb des HPI unter https://pao.dcl.hpi.uni-potsdam.de/wrk/ erreichbar. Für den Zugriff sind ein Zertifikat und eine Anmeldung beim Abgabesystem erforderlich.

Aufgabe 1.5: (30 Punkte)
Implementieren Sie ein C-Programm für die Suche nach Primzahlen, welches unter Windows und Linux mittels `nmake` bzw. `make` übersetzt werden kann!

- Wird das Programm mit genau einem ganzzahligen Parameter gestartet (`prime 100`), soll nur die angegebene Zahl geprüft werden. Handelt es sich um eine Primzahl, soll das Programm die Zahl auf der Standardausgabe ausgeben. Ist die angegebene Zahl keine Primzahl, soll nichts ausgegeben werden.
- Wird das Programm mit einem Dateinamen als Parameter gestartet (`prime numbers.txt`), soll die Datei zeilenweise eingelesen werden und jede Zeile als zu prüfende Zahl interpretiert werden. Alle in der Datei enthaltenen Primzahlen sollen (einzeln und zeilenweise) auf der Standardausgabe ausgegeben werden.
- Wird das Programm mit zwei Parametern gestartet (`prime 100 200`) sollen alle Primzahlen in dem angegebenen Bereich ($100 < x < 200$) ausgegeben werden.
- Berücksichtigen Sie bei der Implementierung mögliche fehlerhafte Parameter und geben Sie entsprechende Fehlermeldungen aus.

Erläutern Sie Ihrem Tutor Ihre Implementierung und führen Sie Ihr Programm unter Windows und Linux vor!
Verpacken Sie alle Source-Code Dateien (inklusive Makefile) in eine ZIP-Datei (blatt1.zip). Diese ZIP-Datei muss mindestens 24 Stunden vor dem Tutoriumstermin in das Abgabesystem eingestellt werden.

Beispiel:

```
> prime 5
5
> prime 6
> prime 1 5
2
3
> cat numbers.txt
1
2
3
4
5
> prime numbers.txt
2
3
5
> prime foo.txt
foo.txt: No such file or directory
>
```