```
#1
   # MindMap:
   #=========
   "Synchronisierung"   "I/O Block-layer"
   |--cpus,             |--block scheduler      "VFS"
   |--process           |--implementation       |--Aufbau
   |--signale    \           der wichtigsten    |--Besonderheiten
                  \            |              / |--bsp anhand eines FS
                   \           |             /
                    \+-------------------------+
 "Scheduling" -------|"Design des Linux Kernel"|\
 |--cpu              +-------------------------+ \  "Subssysteme"
 |--io             /       |                 |    \ |--verschiedene Layer
                  /        |                 |      |-- zusammenspiel
      "syscalls" /         |                 |
      |--implementierung   |                 |
         spezieller        |               "module"
                           |               |--konzept
                      "mem-mangement"      |--implementierung
                      |--Design
                      |--implementierung

#2
   # Argumentationsprinzip:
   #======================
 chronologisch

   # Schrittmuster:
   #===============
 5 Schritt

   # Gliederung:
   #============
 * Einleitung
 * 1.0 Einführung wichtigste Subsysteme
 * 1.1 Implementierung der wichtigsten Systeme
 * 2.0 Zusammenspiel der Systeme
 * 3.0 Treiber im Kernel <NOCH NICHT GANZ KLAR>
 * Schluss

#3

   # Einleitung
   #-----------
 Als Linus Torvalds 1991 die erste Linux-Kernel Version in das Usenet stellte,
 ahnte noch niemand etwas von den Veränderungen die die Veröffentlichung des
 Quellcodes unter der GPL mit sich bringen würde.
   Ursprünglich für die 386er Architektur entwickelt und mit relativ wenig
 Funktionalität hat sich innerhalb der letzen 16 Jahren der Linux-Kernel
 ständig weiter entwickelt und wurde auf etliche weitere Architekturen portiert.
   Heute findet "Torvald's" Kernel nahezu jedem Bereich seine Anwendung:
 Beispielsweise im Serverbereich, als hochleistungs-Rechner, in Set-top-Boxen,
 Routern oder auch in Handys.
   Es ist erstaunlich wie gut der Kernel auf all diesen System skaliert,
 funktioniert und wie einfach es Dank frei zugänglichem Quellcode für
 jeden möglich ist den Kernel anzupassen. Aufgrund der vielen Features und
 Möglichkeiten die der Linux-Kernel bietet scheint er als Ganzes recht Komplex.
 Dank des Designskonzeptes der Unterteilung des Kernels in Subsysteme
 wird der Einstieg in die Kernelprogrammierung und die Implementierung
 eigener Features erleichtert.

   # Schluß
   #-------
 Durch das Zusammenspiel der einzelen Komponenten steht dem ... <MORE TO COME>
 Dank der guten Dokumentation des Kernels und die Möglichkeit sich aktiv daran
 zu beteiligen wird der Linux-Kernel in der Zukunft eine noch bedeutendere
 Rolle spielen.
```