# 17. Dateisytem: Feste Partitionen gleicher Größe

Die Datei `filesystem-aufg1.py` enthält ein Rumpfprogramm, mit dem Sie ein Dateisystem implementieren können. Das Dateisystem wird dabei innerhalb einer Image-Datei (`disk`) erzeugt. Die beiden grundlegendsten Funktionen in diesem Programm sind `read_block()` und `write_block()`, die Blöcke der Größe `BLOCKSIZE` (default: 1024) lesen bzw. schreiben. Die Größe des Dateisystems ist in `DISKSIZE` (default: 1 M) definiert.

**a)** Erweitern Sie das Programm so, dass es ein Dateisystem implementiert, das für die Plattenplatzaufteilung mit Partitionen fester, gleicher Größe arbeitet. Definieren Sie die Variable `PARTSIZE`, die ein Teiler von `DISKSIZE` sein muss, z. B. 65536 (64 KByte), was dann 16 Partitionen erlauben würde (16 x 64 KByte = 1 MByte). Das Dateisystem soll eine flache Struktur (keine Verzeichnisse) haben und Dateinamen im 8.3-Format (bis zu acht Zeichen, dann ein Punkt, dann eine bis zu dreistellige Dateiendung) unterstützen.

Definieren Sie dazu folgende Funktionen:

- `fs_get_no_of_partitions()`: gibt die Anzahl der Partitionen (`DISKSIZE` / `PARTSIZE`) zurück.
- `fs_read_partition(n)`: gibt den Inhalt der `n`. Partition (als String der Länge `PARTSIZE`) zurück. Die Nummerierung beginnt bei 1, weil die "0. Partition" nicht für Dateien zur Verfügung steht, sondern die Metadaten, also das Dateiverzeichnis enthalten soll. `fs_read_partition()` funktioniert unabhängig davon, ob bereits eine Zuordnung der gewünschten Partition zu einem Dateinamen besteht.
- `fs_write_partition(n, string)`: schreibt den String der Länge `PARTSIZE` in die `n`. Partition.
- `fs_create_fat_entry(filename, n)`: erzeugt einen neuen Eintrag in der FAT (in der "0. Partition", also in den absoluten Disk-Adressen `0` - `PARTSIZE-1`), der den angegebenen Dateinamen der Partition Nr. `n` zuordnet.
- `fs_create_file(filename)`: erzeugt eine neue Datei. Die Funktion sucht zunächst nach einer freien Partition, also einer, die noch nicht in der FAT eine Zuordnung zu einem Dateinamen hat, und ruft dann `fs_create_fat_entry()` auf. Rückgabewert der Funktion ist die Partitionsnummer, die im Folgenden wie ein File Descriptor verwendet werden soll. Die Datei ist nach dem Erzeugen zum Lesen/Schreiben geöffnet. Im Hauptspeicher müssen Sie eine Liste geöffneter Dateien verwalten, denn es soll möglich sein, mehrere Dateien gleichzeitig zu öffnen. Wenn der Dateiname schon vergeben ist, soll eine Fehlermeldung zurückgegeben werden, die Datei wird dann nicht geöffnet.
- `fs_read(fd,start,length)`: gibt einen String der Länge `l` < `length` zurück, der den Inhalt der geöffneten Datei ab Position `start` und mit Länge `length` repräsentiert. Beachten Sie dabei, dass sowohl `start` als auch `length` beliebige Werte annehmen können und dabei auch Block-übergreifend Daten gelesen werden können, z. B.: `BLOCKSIZE`=1024, `fs_read(fd,900,300)` benötigt Daten aus dem ersten und aus dem zweiten Block. Beachten Sie außerdem, dass `fs_read()` nicht über die Dateigrenze hinaus lesen darf. Die jeweils aktuelle Dateigröße müssen Sie auch in der FAT speichern, sie ist immer kleiner/gleich der Partitionsgröße. Ist die Datei kleiner als `start+length`, geben Sie einen kürzeren String zurück.
- `fs_write(fd,start,string)`: schreibt den String (der Länge `len(string)`) ab Position `start` in die Datei. Beachten Sie dabei, dass Sie durch das Schreiben evtl. die Dateigröße verändern, dass Sie dabei aber nicht über die Partitionsgrenze hinaus schreiben dürfen. Der Versuch, Daten jenseits der Partitionsgrenze zu schreiben, soll mit einer Fehlermeldung (und gar keinem Schreibvorgang) beantwortet werden.

- `fs_close(fd)`: schließt die Datei. Aktualisieren Sie im Hauptspeicher die Liste der geöffneten Dateien.
- `fs_open(filename)`: öffnet eine Datei, wenn der angegebene Dateiname in der FAT auftaucht. Rückgabewert ist die Partitionsnummer, wie bei `fs_create_file()`. Aktualisieren Sie im Hauptspeicher die Liste der geöffneten Dateien. Dateien sind immer zum Lesen und Schreiben geöffnet.
- `fs_ls()`: gibt ein Verzeichnis aller Dateien aus und zeigt zusätzlich an, welche Partitionen noch ungenutzt sind. Beispiel-Ausgabe:

```
TEST.TXT        Part. 1     3232 Bytes
KOPIE.TXT       Part. 2     3232 Bytes
TEST2.TXT       Part. 3       49 Bytes
Freie Partitionen: 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15
```

Hinweise zur FAT: Verwenden Sie für jeden Eintrag in der FAT 32 Byte, das sorgt später (in Aufgabenteil c) für eine besser lesbare Ausgabe. In der FAT speichern Sie in jedem Eintrag den Dateinamen (12 Byte, oder auch 11 Byte, wenn Sie eine Darstellung ohne den Punkt wählen), die zugeordnete Partitionsnummer und die aktuelle Dateigröße. Um Dateigrößen (oder allgemein Werte >255, aber <65536) zu verarbeiten, verwenden Sie zwei Bytes (High-Byte und Low-Byte), z. B. wie in folgendem Code:

```
def fatentry_get_size(entry):
  high = ord(entry[12])            # Umwandeln der ASCII-Zeichen
  low  = ord(entry[13])            # in Bytes
  return high*256+low

def create_fatentry(..., filesize):
  low  = filesize %  256           # %  = modulo
  high = filesize // 256           # // = div
  entry=entry+chr(high)+chr(low)   # umwandeln in zwei ASCII-Zeichen
  ...
```

Sie können auch eine Variante ohne Partitionsnummer wählen und den *n*-ten Eintrag in der FAT der *n*-ten Partition entsprechen lassen.

**b)** Testen Sie Ihr neues Dateisystem, indem Sie mindestens zwei Dateien erzeugen, eine Datei mit Inhalt füllen, dann schließen, erneut öffnen und den Inhalt der ersten in die zweite Datei kopieren. Schließen Sie dann alle Dateien und öffnen Sie die Kopie um zu prüfen, dass sie jetzt den richtigen Inhalt hat. Überprüfen Sie mit `fs_ls()`, welche Dateien existieren.

**c)** Betrachten Sie den Inhalt der Image-Datei `disk` mit einem Hex-Viewer:

```
hexdump -C disk
```

gibt den Dateiinhalt in einer übersichtlichen Form aus. Suchen Sie nach den FAT-Einträgen und nach den von Ihnen erzeugten Dateien.

[ ☛ Abgabe via Moodle ] Die Lösungen (Programmdateien und Beobachtungen zum jeweiligen Programmverlauf) laden Sie bitte als Zip- oder Tar.gz-Archiv im Moodle (http://hm.hgesser.de/moodle/) unter „Praktikum – Blatt 8“ hoch. Sie erhalten im Moodle-System dann auch die Bewertungen Ihrer Lösungen. Bitte tragen Sie im Programm die Namen aller an der Lösung beteiligten Teilnehmer ein und geben Sie Lösung nur über einen einzigen Moodle-Account ab.