Aufgabe 2 (6 Punkte)

dir

Entwerfen und programmieren Sie ein Programm **dir**, mit dem Sie — ähnlich wie mit dem UNIX-Kommando ***ls(1)*** — den Inhalt eines oder mehrerer Directories anzeigen können. Der Name der Directories wird auf der Kommandozeile übergeben. Das Programm soll auf der Standardausgabe für jedes Directory jeweils zuerst den Directorynamen und in den darauffolgenden Zeilen - durch 4 Leerzeichen eingerückt - die Namen der Dateien in diesem Directory, gefolgt von ihren Inodenummern ausgeben. Dateien, deren Namen mit einem Punkt beginnt, werden wie bei UNIX-ls nicht angezeigt.

2.1 Die Ausgabe erfolgt unsortiert — ***directory(3)***, Funktionen *opendir*, *readdir*
(4 Punkte)

2.2 Erweitern Sie das Programm um die Funktionalität für eine Option **-o**. Wird diese Option beim Aufruf von **dir** angegeben, soll die Ausgabe alphabetisch sortiert werden — ***qsort(3).***
Erläutern Sie in einer Dokumentationsdatei (dir.doc) die Datenstrukturen, die Sie für den Sortiervorgang benutzen und die Vorgehensweise zur Sortierung.
(2 Punkte)

Fehlermeldungen (z. B. Directory nicht lesbar o. ä.) sind auf dem Standard-Fehlerkanal auszugeben (**auf diesen Punkt wird in Zukunft nicht mehr gesondert hingewiesen**)!!!

Speichern Sie Ihr C-Programm unter dem Namen dir.c!

Abgabetermin: 28.11.95 bis 12.00 Uhr

Aufgabe 3 (6 Punkte + 4 Zusatzpunkte)

stat

Entwerfen und programmieren Sie ein Programm **stat**, mit dem Sie für eine reguläre Datei bzw. ein Directory den Dateityp (reguläre Datei = R, Directory = D), die regulären Zugriffsrechte (rwx), die Anzahl der Links, die Dateigröße sowie Benutzer- und Gruppen-Id des Besitzers ausgeben können.

Beispiel: `stat datei1` liefert `Rrw-r--r--  1 10435/20456 196608 datei1`

3.1 Das Programm beschränkt sich auf die beschriebene Funktionalität. Bei anderen Dateitypen ist eine Fehlermeldung auszugeben (unbekannter Dateityp), andere Zugriffsrechte (s- oder t-Bits) sind zu ignorieren. – ***stat(2***)
(4 Punkte)

3.2 Erweitern Sie Ihre Programm um eine Option -t. In diesem Fall sollen zusätzlich noch die Zeiten aus der stat-Struktur (*atime, mtime, ctime*) ausgegeben werden.

Beispiel:

```
Rrw-r--r--  1 10435/20456 196608 Nov 17 14:58/Nov 17 14:58/Nov 17 14:58
datei1
```

(Das UNIX-Kommando ls formatiert die Zeit bei Dateien, die älter als ein halbes Jahr sind anders - diesen Mechanismus müssen Sie nicht nachbauen!). — ***ctime(3).***
(2 Punkte)

3.3 Beschreiben Sie in einer Dokumentationsdatei (stat.doc) die Bedeutung der verschiedenen Zeiten, die die stat-Struktur zu einer Datei enthält. Bei welchen Systemaufrufen werden welche Zeiten neu gesetzt. Gibt es eine Möglichkeit, diese Werte durch ein Programm zu beeinflussen? Wenn ja, wie macht man das, wozu kann dies sinnvoll sein, welche Gefahren für den Rechnerbetrieb gehen von solch einer Möglichkeit aus?

Diskutieren Sie diese Fragestellung mit Kommilitonen und in den Übungen!
(2 Zusatzpunkte)

3.4 Wie können Sie aus den numerischen Werten für die Benutzer- und Gruppen-Id den tatsächlichen Namen des Benutzers und den Gruppennamen ermitteln. Implementieren Sie die entsprechende Funktionalität. **- getpwent(3), getgrent(3)**

3.5 Welche weiteren Dateitypen (über die in 3.1 auszugebenden) gibt es und welche Bedeutung haben diese Dateitypen? Was hat es mit den s- und t-Bits auf sich?
(2 Zusatzpunkte)

Abgabetermin: 05.12.95 bis 12.00 Uhr

 J. Kleinöder, M. Steckermeier

Aufgabe 4 (12 Punkte + 8 Zusatzpunkte)

dir (nur alternativ zu Aufgaben 2 + 3 — aufwendig!!!)

Entwerfen und programmieren Sie ein Programm **dir**, das sich in etwa so wie das UNIX-Kommando ***ls(1)*** verhält. Ihr dir-Programm muß allerdings nur die Optionen **-a, -l**, **-n**, **-c**, **-u** und **-L** des UNIX-ls-Programms unterstützen. Es genügt, wenn pro Zeile eine Datei ausgegeben wird (die mehrspaltige Ausgabe des UNIX-ls muß nicht nachgebaut werden).

Das Programm kann folgendermaßen schrittweise entwickelt werden (die für die einzelnen Schritte interessanten Manual-Seiten sind jeweils mit angegeben).

- 4.1 Das dir-Programm gibt nur die Namen der Dateien aus, die in den, als Aufrufargument angegebenen Directories eingetragen sind. Keine weiteren Optionen, Ausgabe nicht sortiert — ***directory(3)***, Funktionen *opendir*, *readdir* (3 Punkte).
- 4.2 Wenn die Option **-a** angegeben wird, werden auch alle Dateien, die mit einem Punkt beginnen, angezeigt. Ohne diese Optionen sollen diese Dateien nicht mehr in der Ausgabe erscheinen.
- 4.2 Ausgabe alphabetisch sortieren — ***qsort(3)*** (2 Punkte).
- 4.3 Option **-l** einführen, uid und gid der Datei werden numerisch ausgegeben, es werden keine Angaben über die Modifikationszeit angezeigt. Symbolic Links werden nicht erkannt. Als Argument sind jetzt nicht nur Directorynamen, sondern auch Dateinamen zulässig — ***stat(2)*** (muß für alle Dateitypen und Zugriffsrechte funktionieren) (5 Punkte).
- 4.4 Optionen **-u** und **-c** einführen; die Modifikationszeit, bzw. die durch die Optionen angeforderten Zeiten werden mit ausgegeben **—** ***ctime(3)***, Funktion *localtime* (2 Punkte).
- 4.5 Option **-n** einführen; ohne Angabe der Option -n wird jetzt der textuelle Benutzer- bzw. Gruppenname ausgegeben **—** ***getpwent(3)*** Funktion *getpwuid*, ***getgrent(3)*** Funktion *getgrgid* (2 Zusatzpunkte).
- 4.6 Option **-L** einführen, ohne diese Option werden symbolic Links jetzt erkannt und korrekt angezeigt — ***stat(2)*** Funktion *lstat*, ***readlink(2)*** (2 Zusatzpunkte).
- 4.7 und 4.8 siehe Aufgaben 3.3 und 3.5 (je 2 Zusatzpunkte)

Testen Sie Ihre Lösung auch mit den Directories /bin, /dev und /etc aus!

Erläutern Sie in einer Datei dir.doc die zum Sortieren eingesetzten Datenstrukturen und die Vorgehensweise.

Bei *character special files* und *block special files* gibt das UNIX-ls statt der Dateigröße eine sog. *major-number* und eine *minor-number* aus (zu sehen bei ls -l auf /dev). Dies können Sie ignorieren. Ebenso müssen Sie die spezielle Datums-Darstellung bei älteren Dateien (Jahr statt Sekunden) nicht nachbauen.

Mit diesem Programm erhalten Sie einen relativ guten Nachbau des UNIX-ls-Kommandos. Andererseits ist das Programm ziemlich aufwendig (letztes Jahr waren die kurzen Lösungen ca. 8 Seiten lang). Wählen Sie diese Alternative also nur, wenn Sie sich das zutrauen.

Abgabetermin: 05.12.95 bis 12.00 Uhr