Technische Informatik 2

1. Übungsblatt

Aufgabe 1 (6 Punkte, Gruppe)

Schreibt ein Programm `my_sum`, das die Größe aller Dateien in einem Verzeichnis samt seiner Unterverzeichnisse summiert. Wir haben Euch hierzu in `/home/ti2/ueb/01` ein `my_sum.cc` vorgegeben, das jedoch nicht vollständig ist und von Euch ergänzt werden muss. Der Sinn dieser Übungsaufgabe ist es, dass Ihr Euch mit den relevanten Bibliotheksfunktionen und Datenstrukturen vertraut macht.

Ihr könnt Euer Programm schrittweise erweitern: Zunächst sollte es die Größe aller Dateien im aktuellen Verzeichnis aufsummieren und ausgeben. Danach solltet Ihr rekursiv die enthaltenen Unterverzeichnisse einbeziehen. Schließlich soll es möglich sein, das oberste zu betrachtende Verzeichnis als Parameter an Euer Programm zu übergeben.

Das vorgegebene Programm öffnet das aktuelle Verzeichnis, das als `.` bezeichnet wird, mit der Bibliotheksfunktion `opendir()`, die in der entsprechenden *Manpage* dokumentiert ist.

```
if(!(verzeichnis=opendir("."))){
    perror("my_sum: open failed");
    exit(EXIT_FAILURE);
}
```

Im Fehlerfall wird eine entsprechende Meldung ausgegeben und das Programm mit einem Fehlerwert beendet. Analog solltet Ihr auf Fehler der anderen von Euch eingesetzten Funktionen reagieren. Achtet insbesondere auf zurückgegebene Pointer. Diese könnten `NULL` sein.

Ihr sollt für das Auslesen eines Verzeichnisses neben der Funktion `opendir()` die Funktionen `readdir()` und `closedir()` verwenden. Es sei erwähnt, dass die meisten Manpages auch Code-Beispiele enthalten, auf die Ihr ggf. zurückgreifen könnt. Ebenso zeigen die Manpages an, welche Include-Dateien Ihr einbinden müsst. Für `opendir()` war dies z.B. `dirent.h`.

Der Eintrag `d_name` in der von `readdir()` erstellten Datenstruktur beinhaltet den Namen des Verzeichniseintrags, über den Ihr mittels `lstat()` mehr erfahren könnt. Die zugehörige Manpage beschreibt u.a. das Makro `S_ISDIR`, mit dem Ihr ermitteln könnt, ob es sich bei diesem Eintrag um ein Unterverzeichnis handelt, in das Ihr ggf. rekursieren müsst.

Ihr sollt nur die im Eintrag `st_size` genannten Größen der *normalen Dateien* rekursiv aufsummieren. Alle anderen Dateitypen (vgl. Manpage zu `lstat()`) könnt Ihr ignorieren.

Für die Übergabe des obersten zu betrachtenden Verzeichnisses als Parameter via `argv[]` findet Ihr zumindest ein hinreichendes Beispiel in den Unterlagen des C/C++-Vorkurses.

Wir haben kein Makefile vorgegeben, da hier die impliziten Regeln von Make ausreichen. Testet Euer `my_sum` in einem beliebigen Verzeichnis unterhalb von `/home/ti2` und nennt in der Abgabe dessen Pfad, damit Euer Tutor das Beispiel testen kann.

Aufgabe 2 (2 Punkte, Gruppe)

Das Programm `caps.cc` aus den C/C++-Vorkurs liest von `STDIN` und gibt die Eingaben verändert auf `STDOUT` aus (Ihr findet das Programm auch unter `/home/ti2/ueb/01/caps.cc`). Was geschieht, wenn man die folgende Befehlsfolge auf einem der x-Rechner eingibt?

```
make caps
date
date | caps
date >> caps
date | caps
date > caps
date | caps
date < caps
```

Bitte zeilenweise *erklären*. Nicht nur die im Shell sichtbaren Auswirkungen beschreiben.

Aufgabe 3 (2 Punkte, Gruppe)

Die Rechner des Fachbereichs stehen Euch nicht nur in den Rechnerpools, sondern auch über das Internet zur Verfügung. Über das Secure-Shell-Protokoll (SSH) kann man sich an den Rechnern anmelden. Hierfür haben sich die meisten von Euch bereits ein entsprechendes Schlüsselpaar angelegt. (Falls nicht: siehe Wiki zum C/C++-Vorkurs in Stud.IP.)

1. Warum stellt es eigentlich kein Sicherheitsproblem dar, dass die Public Keys aller Nutzer im Fachbereichsnetz abgelegt sind, also potentiell von Dritten ausgelesen werden können?
2. Früher konnte man sich von außen per SSH am Fachbereichsnetz anmelden, indem man sein Passwort eingegeben hat. Diese Passwort-basierte Anmeldung wurde inzwischen nicht nur im Fachbereich 3, sondern weltweit an vielen Einrichtungen deaktiviert, insbesondere für den privilegierten Account `root`. Wieso?

Für diese beiden Fragen ist ggf. eine kleine Recherche erforderlich. Denkt daran, Eure Quellen zu nennen.

Abgabe

Bis 10ct am Mi 31.10.2012 ausgedruckt ins Postfach 20 in der MZH-Ebene 6 und per Mail an Euren Tutor. Es gelten die vereinbarten Scheinbedingungen (siehe Stud.IP). Bitte beachtet unsere ergänzenden Hinweise ebenda.

Wir möchten eine Statistik der Teilnehmer aus verschiedenen Studiengängen anlegen. Gebt daher auf dieser Abgabe den Studiengang und das Fachsemester der Teilnehmer an.

Weitere Aufgaben

1) Welche zwei Hauptaufgaben hat ein Betriebssystem?
2) Was ist ein Prozeß?
3) Wie ist ein Unix-Dateisystem strukturiert? Wie können Dateien darin (eindeutig) aufgefunden werden?
4) Was ist ein *symbolic link*?
5) Ist das Unix-Dateisystem wirklich ein Baum? Begründung.
6) Welche Zugriffsrechte kann man auf eine Unix-Datei haben? Welche Dateiattribute steuern dies, und wie?
7) Welche Vorteile bietet es, auf Terminals in Unix wie auf Dateien zuzugreifen? Was versteht man unter *Ein-/Ausgabeumlenkung*?
8) Welche Aufgabe hat ein Kommando-Interpreter (z.B. in Unix der *Shell*)?

Diese Aufgaben müssen nicht abgegeben werden. Sie dienen als Vorbereitung auf das Fachgespräch und werden im Tutorium besprochen.