オペレーティングシステムの機能を使ってみよう
第5章 ファイル操作システムコール

ファイル操作システムコール

- ユーティリティコマンド（ln，rm，mkdir，rmdir，chmod …）等が使用しているシステムコール．
- この章では主要な8種類だけ紹介する．
- 以下の内容は macOS 10.13 を基準にしているが，一部では分かり易さのために簡単化して説明している場合もある．

unlink システムコール

- ファイル（リンク）を削除するシステムコール．
- ディレクトリの削除には使用できない．
- rm コマンドは，このシステムコールを使用．

書式 path 引数でリンクのパスを一つ指定する．

```
#include <unistd.h>
int unlink(char *path);
```

解説 unistd.h をインクルードする．
正常時は 0，エラー発生時は -1 を返す．
エラー原因は perror() 関数で表示できる．

使用例 "a.txt"ファイルを削除する例を示す．

```
if (unlink("a.txt")<0) {    // "a.txt" 削除
  perror("a.txt");          // エラー原因表示
  exit(1);                  // エラー終了
}
```


mkdir システムコール

- ディレクトリ（フォルダ）作成するシステムコール．
- mkdir コマンドは，このシステムコールを使用．

書式 path，mode でパスと保護モードを指定する．

```
#include <sys/stat.h>
int mkdir(char *path, int mode);
```

解説 sys/stat.h をインクルードする必要がある．
正常時は 0，エラー発生時は -1 を返す．
エラー原因は perror() 関数で表示できる．
パスに含まれる途中のディレクトリは作らない．

使用例 "newdir"ディレクトリを rwxr-xr-x で作成する例．

```
if (mkdir("newdir", 0755)<0) { // "newdir"作成
  perror("newdir");            // エラー原因
  exit(1);                     // エラー終了
}
```


rmdir システムコール

- ディレクトリを削除するシステムコールである．
- rmdir コマンドは，このシステムコールを利用．
- **空でないディレクトリを削除することはできない．**

書式 path 引数でディレクトリのパスを一つ指定する．

```
#include <unistd.h>
int rmdir(char *path);
```

解説 unistd.h をインクルードする必要がある．
正常時は 0，エラー発生時は -1 を返す．
エラー原因は perror() 関数で表示できる．

使用例 "newdir"と名付けられたディレクトリを削除する例．

```
if (rmdir("newdir")<0) {  // "newdir" 削除
  perror("newdir");       // エラー原因表示
  exit(1);                  // エラー終了
}
```


link システムコール（1/2）

- リンク（ハードリンク）を作るシステムコール．
- ln コマンドは，このシステムコールを利用．

書式 存在するパスと新しいパスを指定する．

```
#include <unistd.h>
int link(char *oldpath, char *newpath);
```

解説 unistd.h をインクルードする必要がある．
正常時は 0，エラー発生時は -1 を返す．
エラー原因は perror() 関数で表示できる．

使用例 1 ファイルにリンク"b.txt"を追加する例．

```
if (link("a.txt", "b.txt")<0) { // "b.txt"作成
  perror("link");               // 原因は？
  exit(1);                      // エラー終了
}
```

link システムコール（2/2）

使用例2 ファイルの移動に応用した例.

```
unlink("b.txt");                  // 念のため
if (link("a.txt", "b.txt")<0) { // リンクを作る
  ... エラー処理 ...
}
if (unlink("a.txt")<0) {          // リンクを消す
    ... エラー処理 ...
}
```


symlink システムコール

- シンボリックリンクを作るシステムコール.
- ln -s コマンドは，このシステムコールを利用.

書式 シンボリックリンク自身のパスと内容を指定する.

```
#include <unistd.h>
int symlink(char *path1, char *path2);
```

解説 unistd.h をインクルードする必要がある.
正常時は 0，エラー発生時は-1 を返す.
エラー原因は perror() 関数で表示できる.

使用例 内容が"a.txt"のシンボリックリンク"b.txt"を作る.

```
if (symlink("a.txt", "b.txt")<0) { // リンクを作る
  perror("b.txt");                 // エラー表示
  exit(1);                         // エラー終了
}
```


rename システムコール

- ファイルの移動（ファイル名の変更）を行うシステムコール.
- mv コマンドは，このシステムコールを利用.

書式 新旧二つのパスを指定する.

```
#include <stdio.h>
int rename(char *from, char *to);
```

解説 stdio.h をインクルードする必要がある.
正常時は 0，エラー発生時は-1 を返す.
エラー原因は perror() 関数で表示できる.

引数 from は古いパス to は移動後の新しいパス.

使用例 "a.txt"のパスを"b.txt"に変更する例.

```
if (rename("a.txt", "b.txt")<0) { // パス名を変更
  perror("rename");               // エラー表示
  exit(1);                        // エラー終了
}
```


chmod（lchmod）システムコール（1/2）

- ファイルの保護モードを変更するシステムコール.
- chmod コマンドは，このシステムコールを利用.

書式 パスと保護モード（rwxrwxrws）を指定する.

```
#include <sys/stat.h>
int chmod(char *path, int mode);
int lchmod(char *path, int mode);
```

解説 sys/stat.h をインクルードする必要がある.
シンボリックリンクが指定された場合，lchmod はシンボリックリンクの保護モードを変更する.
正常時は 0，エラー発生時は-1 を返す.
エラー原因は perror() 関数で表示できる.

chmod（lchmod）システムコール（2/2）

使用例 "a.txt"の保護モードを"rw-r--r--"に変更する.

```
if (chmod("a.txt", 0644)<0) { // "rw-r--r--"
  perror("a.txt");            // エラー表示
  exit(1);                    // エラー終了
}
```

参考 strtol 関数
8 進数を表現する文字列から整数値（int 型）に変換する.

```
char *ptr;
char *ostr = argv[1];             // 8 進数文字列
int mod;
mod = strtol(ostr, &ptr, 8);      // int に変換
if (*ostr=='\0' || *ptr!='\0') { // エラーチェック
  fprintf(stderr,
          "\'%s\' : 8 進数の形式が不正\n", ostr);
  return 1;
}
```


readlink システムコール（1/2）

- シンボリックリンクの内容を読出すシステムコール.
- ls -l コマンドは，このシステムコールを利用.

書式 シンボリックリンクとバッファを指定する.

```
#include <unistd.h>
int readlink(char *path, char *buf, int size);
```

解説 unistd.h をインクルードする必要がある.
path で指定されるシンボリックの内容を buf に読み出す.
正常時は読み出した文字数，エラー発生時は-1 を返す．エラー原因は perror() 関数で表示できる.
読み出した文字列は'\0' で終端**されない**.

引数 path は目的のシンボリックリンクのパス.
buf は内容を読み出す領域（バッファ）のポインタ.
size は領域のサイズ.

readlink システムコール（2/2）

使用例 シンボリックリンク"b.txt"の内容を読み出し表示する例.

```
char *name = "b.txt";
char buf[100];                    // 100バイト
int n = readlink(name, buf, 99); // 99バイト！
if (n<0) {                        // エラー？
  perror(name);                   // エラー表示
  exit(1);                        // エラー終了
}
buf[n]='\0';                      // 文字列完成
printf("%s -> %s\n", name, buf); // ls -l 風に
```


課題 No.4

1. 簡易版のUNIXコマンドを作成しなさい.
   UNIXコマンド，rm，mkdir，rmdir，ln，mv，chmodの中から3つ以上を選択し，上記のシステムコールを使用して簡易版を作ってみる.
   使用方法のエラーチェック，システムコールのエラーチェックは行うこと.
2. 自分が作った簡易版と本物の機能の違いを調べる.
   「`$ man 1 rm`」等で本物のマニュアルを参照し比較してみる.
3. renameシステムコールの必要性を考察する.
   UNIXはSimple is bestの思想で作られてきた．既存の機能を組み合わせて実現できる場合は専用機能の追加はしないことが多い.
   renameシステムコールの機能はlink，unlinkシステムコールの組み合わせでも実現できるが，なぜ追加されたか考えなさい.
4. statシステムコール，readdir関数について調べてみる.
   これらは何コマンドを作る時に必要になるか？