調査 5 モデルカリキュラムの提言 コースウェア

8. Linux のシステムプログラミングに関するスキル

Ⅰ. 概要	Linux をインストールしたマシン上で、多くの動作するプログラム例をコンパイル、実行して、システムプログラムを作成する。最終的には、DBM、割り込み処理、子プロセスを作成して親プロセスとプロセス間通信、ネットワークプログラミング等、システムプログラム全般の技術について学習する。
Ⅱ. 対象専門分野	職種共通
Ⅲ. 受講対象者、受講前提	本カリキュラムの「Linux のシステム管理」を受講済みであること。または、同等の知識を有すること。
Ⅳ. 学習目標	• エラー表示から問題点を見出して修正し、実際に動作できるまで繰り返し実行する。 • シェルスクリプトを読んで、どのような処理を行うかを理解できるようになる。 • ファイル処理プログラムを作成できるようになる。 • 実行プログラムが利用している共有ライブラリを確認できるようになる。 • 具体的に共有ライブラリを作成して、それを利用する。 • BerkeleyDBM を利用した処理プログラムを作成できるようになる。 • 割り込み処理プログラムを作成できるようになる。 • 親プロセスから子プロセスに処理データを与え、実行結果を親プロセスに返すプログラムを作成できるようになる。 • プロセス間通信のプログラム例を作成して理解する。 • サーバとクライアントのプログラム例を作成して、動作確認をする。
Ⅴ. 使用教科書、教材等	• 『C 言語による UNIX システムプログラミング入門』 河野清尊著、オーム社刊 • 『詳解 UNIX プログラミング』 W.Richard Stevens 著、大木敦雄訳、ピアソン・エデュケーション刊 • 『Linux Programming in 24 Hours』 Warren W.Gay 著、SAMS 刊 • 『Linux システムプログラミング』 羽山博著、オーム社刊

Ⅵ. 習得スキル の評価方法	実習 60 分の 10 テーマに関しては、プログラム例を与えてコンパイルし、実行させる。さらに、与えたプログラム例を改良する課題を与え、完成したプログラムと実行結果を提出させて評価する。全課題の 80%の提出をもって合格とする。
Ⅶ. カリキュラム の構成	レベル 1　第 1 回～第 7 回 レベル 2　第 8 回～第 15 回

※注 プログラミングなので、最初に講義形式でテーマを解説し、後は実習とワークショップ（実習 60 分の 10 テーマ）を主とする研修とする。

講座内容

第 1 回　ログイン手順とコンパイル手順（講義＋ワークショップ 90 分）

ユーザ名とパスワードを認証後に UNIX システムにログインできる。パスワード、PAM、LDAP 認証について解説する。ログインして簡単なプログラムを作成し、コンパイルして実行する。よく利用するエディタの操作やコマンド入力を練習する。

（1）ログイン手順

1. ログイン名とパスワードでログインする。exit でログアウトする。
2. /etc/passwd ファイル

   【パスワードファイルの行構造】

   ログイン名：パスワード：UID：GID：gecos：ホームディレクトリ：ログインシェル

   useradd、userdel コマンドでユーザの登録・削除を行う。
3. /etc/group ファイル：CVS を利用した開発メンバーに追加登録する。

   グループ名：パスワード：グループ ID：メンバーリスト

   groupadd、groupdel コマンドでグループの登録・削除を行う。
4. /etc/shadow ファイル
5. PAM 認証：モジュールタイプ、コントロールフラグ
6. LDAP 認証、シングルサインオンについて理解する。
7. ログイン時のユーザ環境変数を追加設定する方法を理解する。

（2）エディタ

1. vi や emacs エディタの操作

   作成、修正、削除、更新処理を自由にできるように練習する。

（3）コンパイル手順

1. cpp：プリプロセッサ、gcc：コンパイラ、ld：リンケージエディタの役割を理解する。
2. ソースプログラム（.c）、オブジェクトプログラム（.o）、実行プログラムの区別ができる。
3. コンパイル時に指定する項目

   -L：ディレクトリパスを指定、-l：ライブラリファイルを指定（例：libxyz.a ファイルは-lxyz）

第 2 回　shell プログラミング（講義＋ワークショップ　90 分）

エディタを使って#!で始まるコマンドを記述したスクリプトファイルを作成する。ファイルに実行権を与えて./ファイル名で実行して確認する。スクリプトファイルを作成することで、定型的な作業を簡単に実行できることを体験する。簡単なサーバ起動スクリプト等を例にあげて処理手順を理解する。

（1）シェル

1. シェルの種類

   bash（#!/bin/bash）、B シェル（#!/bin/sh）、K シェル（#!/bin/ksh）、

   C シェル（#!/bin/csh）、tcsh（#!/bin/tcsh）

2. スクリプトファイルは#!で始まるテキストファイルである。
3. chmod コマンドでスクリプトファイルを実行可能にする。
4. 変数名は$で始まる。
5. シェルのヒストリー機能を理解する。

（2）シェル構文

1. シェルコマンド

   break、continue、echo、exec、exit、export、printf、return、set、shift、unset 他

2. 制御構造

   if、elif、for、while、until、case 他

第 3 回　ファイル入出力プログラミング（講義＋ワークショップ 90 分）

標準入出力ライブラリだけでなく、低水準ファイルアクセスをするシステムコールを使って、ファイルを作成し、ファイルからデータを入出力し、ファイルを閉じるプログラムを作成する。

オーナ、グループ、その他に区分したファイルアクセス権限を理解する。

（1）低水準ファイル処理に関する説明事項

1. open で指定するファイルステータスフラッグとモードに関する事項

   O_RDONLY、O_WRONLY、O_RDWR、O_CREAT、O_APPEND 他

   S_IRUSR、S_IWUSR、S_IXUSR、S_IRWXU、S_IRWXG、S_IRWXO 他
2. creat で新しいファイルを作成する。
3. read、write でデータを読み書きし、lseek で読み書きポインタを移動する。
4. sync、fsync でバッファキャッシュの内容をディスクの実ファイルに記録する。
5. close でファイルをクローズする。
6. umask によるプロセスのファイルモード作成マスクを設定・変更する。
7. chmod、fchmod でファイルのモードを変更する。
8. chown、fchown、lchown でファイルのユーザ ID、グループ ID を変更する。
9. access でファイル操作のアクセス権を検査する。
10. link、unlink、remove、rename 等を理解する。

（2）ファイル操作に関する説明事項

1. stat、fstat、lstat で i ノード、ファイルモード、UID、GID、時間情報を取得する。
2. stat 構造体を理解する。

（3）標準入出力ライブラリに関する説明事項

1. fopen、fread、fwrite、fclose、fflush、fseek 等を理解する。
2. fgetc、getc、getchar、fputc、putc、putchar、fgets、gets 等を理解する。

（4）書式付き入出力

1. printf、fprintf、sprintf、scanf、fscanf、sscanf 等を理解する。

（5）一時ファイルを作成するときに tmpnam、tmpfile を利用する。

第 4 回　ファイルシステム（講義＋ワークショップ 90 分）

大容量のディスクを複数のパーティションに分割して独立したディスクとして扱うことができる。
パーティションは 1 つのブートブロックと複数のブロックに分けられる。分割したブロックを mount することで 1 つのルートディレクトリのツリーに付加できることを学習する。

（1）ディレクトリの保守

1. link、unlink、symlink、mkdir、rmdir、chdir、getcwd 等を理解する。
2. DIR 構造体、dirent 構造体を理解する。
3. opendir、readdir、telldir、seekdir、closedir 等を理解する。

（2）ファイルシステムの構造

1. データを保持するファイル、ディレクトリファイル、スペシャルファイルがある。
2. statfs によるスーパーブロックの情報を取得する。statfs 構造体 sync を確認する。
3. mount、umount でディレクトリの追加・削除ができる。/etc/fstab を理解する。
4. fat、ext2、ext3、ReiserFS 等ファイルシステムの特徴を理解する。
5. ファイルを管理する情報 i ノードを理解する。

（3）/proc ファイルシステムに保持される情報を理解する。

1. ディスク上に作成されるファイルではなく、カーネル情報を保持する仮想ファイルであることを理解する。

第 5 回 UNIX 環境(講義＋ワークショップ 90 分)

コマンドラインから main( )関数に渡す引数の処理手順を学習する。また、環境変数の取得・設定方法を学習する。さらに、共有ライブラリを使用する/しないによって、実行形式ファイルサイズの大きさや利点・欠点等を理解する。

(1)コマンドラインオプションと環境変数

1. main の引数:引数の数、引数の文字列データ、環境変数の文字列データを理解する。
2. コマンドラインのオプションとロングオプション処理手順を getopt で理解する。
3. printenv コマンド、set コマンドで環境変数を確認する。
4. getenv 取得関数、setenv 設定関数、putenv 変更関数、unsetenv 削除関数を理解する。
5. exit の戻り値を echo $?で表示する。

(2)代表的な環境変数

1. USER、HOME、PATH、LANG、PWD、SHELL、TERM 他を理解する。
2. .bash_profile や.bash_rc ファイルに環境変数を追加し、ログイン時に有効にする手順を理解する。

(3)共有ライブラリの利用とメモリ配置の確認

1. -static オプション付き/なしでコンパイルした実行ファイルの大きさを実感する。
2. ldd コマンドで 2 種類の実行形式のファイルに含まれるライブラリー覧を確認する。
3. size コマンドで 2 種類のメモリ配置情報を確認する。

第 6 回　ライブラリの利用方法と作成手順(講義＋ワークショップ 90 分)

開発された有益なオブジェクトモジュールをライブラリに登録しておくことで、重複して同じ開発をすることを回避できる。static と shared の 2 種類のライブラリの利用方法と登録方法について理解する。また、SQL データベースや暗号化ライブラリを利用したシステムプログラムのコンパイル手順を理解する。

(1)static ライブラリ

1. ar:アーカイブコマンド、nm でライブラリに登録されている情報を取得する。
2. コンパイル時にライブラリディレクトリ(-L)とライブラリファイル(-l)を指定する。

(2)shared ライブラリ

1. コンパイル時に-fPIC を指定し、共有ライブラリ作成時に-shared を指定する。
2. 実行ファイルが利用する共有ライブラリを ldd で確認する。
3. /etc/ld.so.conf に共有ライブラリディレクトリを追記し ldconfig を実行して有効にする。

(3)ライブラリ関数とシステムコールの違い

1. マニュアルの分類番号による区別

1:コマンド、2:システムコール、3:ライブラリ

2. 同じ関数に思えるが、OS の提供するサービスを利用する関数がシステムコールである。

第 7 回　データの管理(講義＋ワークショップ 90 分)

メモリは他のプロセスや不正プログラムによって書き換えられないように OS が管理している。ロックを掛けることで、複数のプロセスが同じファイルを同時に利用しないようにする。また、代表的な Berkeley データベースを処理するプログラム例を理解する。

(1)メモリの管理

1. メモリの割り当て、メモリの開放。
2. malloc、calloc、realloc 関数を理解する。

(2)ファイルのロック

1. O_CREATE フラグ、O_EXEL(排他)フラグを使ったロックファイルの作成と削除
2. fcntl 関数を使ったロック

   F_GETLK、F_SETLK、F_SETLKW コマンドによるファイルロック
3. lockf 関数を使ったロック

   F_LOCK(排他)、F_TEST(テスト)、F_ULOCK(解除)、F_TLOCK(テストと排他)

(3)データベース

1. dbm、ndbm、gdbm の解説と dbm にデータ登録、一覧表示、削除等の処理例
2. dbm アクセス関数:dbm_open、dbm_store、dbm_fetch、dbm_close
3. 登録データ確認と削除用の関数:dbm_firstkey、dbm_nectkey、dbm_delete

第 8 回　ソフトウェアの開発環境(講義＋ワークショップ 90 分)

コンパイル対象となるソースファイルの数が多くなった場合は、gcc でなく make を利用する。簡単な Makefile の内容を読むことで、どのように定義された処理が行われるかを理解する。CVS を利用してソフトウェア開発のバージョン管理を体験する。また、配布する圧縮ファイルの作成と解凍・展開手順、パッチファイルの作成方法を学習する。

(1)make と Makefile

1. make 時の表示に注目して、更新したソースだけをコンパイルすることを確認する。
2. makefile のマクロの利用、all:ターゲット
3. $@、$?、$<、$*等の特殊なマクロを理解する。

(2)ソースコード管理

1. RCS(Revision Control System)、CVS(Concurrent Wersion System)でバックアップ
2. CVS コマンド:cvsinit、import、checkout、commit、update、add 等を理解する。
3. RPM パッケージのインストール、アップグレード、削除手順を理解する。

(3)patch と tar

1. tar.gz や tar.bz2 ファイルの解凍展開方法と作成方法を理解する。
2. patch コマンドによるパッチの当て方と差分ファイルの作成方法を理解する。

第 9 回　デバッグ（講義＋ワークショップ　90 分）

ソースプログラムを作成するときに、最初からデバッグを目的としたコードを挿入しておき、プリプロセッサでデバッグモードにコンパイルすることを体験する。また、gdb デバッグツールを利用する手順を習得する。ただし、究極的なデバッグ作業はソースプログラムに printf を追加して確認する。

(1)ソースプログラムにデバッグ情報を埋め込む方法

1. エラー表示関数 perror と strerror を利用する。
2. プリプロセッサを使ってインスツルメンテーションコードを挿入する。

   プリプロセッサに-D 指定コード名を渡して、真にする。
3. _FILE_、_LINE_、_DATE_、_TIME_等のトレース・マクロを利用する。

(2)gdb によるデバッグ作業

1. デバッグするためのコンパイル方法：gcc -g
2. list、run、print、disp、cont コマンド。ブレークポイントの設定を理解する。

第 10 回　プロセスとスレッド（講義＋ワークショップ　90 分）

fork システムコールを利用して子プロセスを生成し、複数の処理を並行に実行する。fork の戻り値によって、親プロセス、子プロセス、エラーを区別できる。6 種類の exec ファミリーを使用して、新しいプロセスを実行するプログラム例を作成する。スレッドは 1 つのプロセス内に複数作成することができ、fork よりも処理が軽いことを理解する。

(1)プロセス

1. ps コマンドでプロセスの状態を確認する。すべてのプロセスは init から起動される。
2. fork でプロセスを生成し、戻り値によって次の 3 通りの場合に分けられる。
   戻り値が 0 は子プロセス、戻り値が正は親プロセス、戻り値が負はエラー
3. exec ファミリーで別プログラムを実行する。
4. wait で親プロセスは子プロセスの終了を待つ。
5. fork、exec、waitpid と同じことを行う system も利用する。
6. ゾンビ状態：プロセスは消滅しているが、プロセステーブルが開放されていない。

(2)プロセス管理情報

1. getpid、getppid、getuid、geteuid、getgid、getegid 等を理解する。
2. nice コマンドによる優先度（-20 から 19）を変更する。

(3)/proc/数字ディレクトリ

1. /proc/数字ディレクトリの数字はプロセス ID を示す。
2. /proc/数字ディレクトリ内を表示し、実行ファイル名や環境等の情報を確認する。

(4)スレッド

1. -lpthread オプションを付けてコンパイルする。
2. pthread_create 関数でスレッドを生成する。
3. pthread_join 関数で指定したスレッドの終了を待つ。
4. mutex 機能を使った排他制御を理解する。
   pthread_mutex_init、pthread_mutex_lock、pthread_mutex_unlock

第 11 回 シグナル(講義＋ワークショップ 90 分)

シグナルは割り込みの一種で、発生したことを非同期にプロセスに伝えることができる。signal で指定した事象が発生すると、指定したシグナルハンドラへ制御を渡す。具体的によく利用されるシグナルプログラム例を作成する。

(1)シグナルの種類

1. signal でシグナル番号とシグナルハンドラの関連付けをする。
2. signal.h ファイルに定義している SIGHUP、SIGINT 等のシグナルの種類を理解する。
3. デフォルトに戻す SIG_DFL、シグナルを無視する SIG_IGN がある。
4. sleep(n)で n 秒間プロセスを休止する。

(2) SIGALRM と alarm を使用してタイマー割り込みプログラムを作成する。

1. signal で SIGALRM とシグナルハンドラを指定する。
2. alarm(n)で n 秒後にアラームクロックが発生するように設定する。

(3)SIGINT を指定したプログラムを作成し、Ctrl-C キーを押した時の動作を確認する。

1. signal で SIGINT とシグナルハンドラを指定する。
2. Ctrl-C を押すと、シグナルハンドラに制御が移ることを確認する。

第 12 回 プロセス間通信とパイプ(講義＋ワークショップ 90 分)

fork によって起動したプロセス間で情報交換を行うプロセス間通信(IPC:InterProcess Communication)を学習し、pipe によってプロセス間で情報を通信する手順を理解する。
また、popen を利用したプログラム例を作成して実行する。

(1)低水準の pipe 関数

1. 単方向パイプと双方向パイプ
2. プロセスは pipe を作成した後、fork で子プロセスを作成する。
3. パイプの入出力を標準入出力に変更する dup、dup2 関数

(2)高水準の popen と pclose 関数

1. popen を利用して別プロセスを起動させ、データを容易に受け渡しすることができる。
2. popen 関数は起動するコマンド、オープンモード(“r”か”w”)の引数を指定する。

(3)名前付きパイプ(FIFO)

1. 親子関係のないプロセス間で通信できるパイプで、スペシャルファイルとも呼ぶ。
2. mkfifo で作成して、open する。ファイルの一種なので、read、write で読み書きする。

第 13 回　端末機器の入出力（講義＋ワークショップ 90 分）

すべてのデバイス（キーボード、ディスプレイ、プリンタ、ディスク、シリアルポート）をファイルとして扱える UNIX の特徴を理解する。シリアル端子に接続した機器（例えば蛍光表示管：VFD）にメッセージを表示するプログラムやパラレル端子に接続した LED やスイッチの情報を処理するプログラム例をあげて、動作させることで端末入出力に関する知識を習得する。

（1）標準入出力、標準エラー出力をファイルにリダイレクトする手順を理解する。

（2）/dev ファイルを使った入出力処理。

1. 例えば/dev/ttyS0 を open で利用する。
2. /dev ディレクトリのリスト表示から、次の情報を確認する。
   キャラクタデバイス、ブロックデバイス、メジャー番号、マイナー番号
3. termios 構造体の c_iflag、c_oflag、c_cfalg、c_lflag フィールドに設定するフラグ値
4. ボーレート関数、行制御関数、端末識別（isatty）を理解する。
5. カノニカルモード（行単位の制御）と非カノニカルモードの違いを理解する。
6. ioctl による情報の設定・取得手順を理解する。

第 14 回　セマフォ、共有メモリ、メッセージキュー（講義＋ワークショップ 90 分）

一般に SystemV のプロセス間通信と呼ばれているセマフォ、共有メモリ、メッセージキューについて理解する。セマフォは排他制御に利用される。パイプは親子関係にあるプロセス間でしか通信できないが、共有メモリはどのようなプロセス間でも利用できる。共有メモリはプロセス間で共通のメモリ領域を利用するが、メッセージキューはプロセス間でデータを送受信する機能であることを理解する。

（1）セマフォ（排他制御）

1. リソースを1つのプロセスのみが利用できるようにする排他制御に必要な機能である。
2. セマフォ変数を複数のプロセスで共有し、あるプロセスが UP（使用）すると、他のプロセスは使用したプロセスが DOWN（開放）するまで待機させられる。
3. セマフォ関連関数：semctl、semget、semop

（2）共有メモリ（Shared Memory）

1. 複数のプロセスで論理メモリを共有できるので、プロセス間でデータを転送できる。
2. 共有メモリ関連関数：shmget、shmat、shmdt、shmctl

（3）メッセージキュー

1. あるプロセスから送信したメッセージをメッセージキューに保持する（待ち行列）。
2. 別のプロセスがメッセージキューからメッセージを取得できる。
3. メッセージキュー関連関数：msgget、msgsnd、msgrcv、msgctl

第 15 回　ネットワークプログラミング（講義＋ワークショップ 90 分）

ネットワーク回線を介してクライアントとサーバ間でデータ通信を行うことを理解する。TCP、UDP、UNIX ソケットのサーバに複数のクライアントを接続するシステムも可能である。スーパーサーバ xinetd 配下で動作するサーバプログラム例を作成する。

(1)ソケット

1. socket でソケットを作成する時のプロトコルファミリーとタイプの種類

   PF_UNIX、PF_INET、PF_INET6 他

   SOCK_STREAM、SOCK_DGRAM、SOCK_RAW 他

2. socket で作成したソケットに名前を付ける:bind
3. サーバ側プロセス:listen、accept
4. クライアント側プロセス:connect
5. ストリームソケットのデータ送受信:send、recv、write、read
6. データグラムソケットのデータ通信:sendto、recvfrom
7. ソケット接続の終了:close
8. ホストバイトオーダとネットワークバイトオーダ:htons、htonl、ntohs、ntohl

(2)複数のクライアント

1. FD_ZERO、FD_SET、FD_ISSET、FD_CLR
2. 複数の入力機器の状態をしらべ、処理する機器を決定する:select

(3)現在サービスを受け付けているポート番号の確認

1. netstat コマンドによるサーバ動作確認
2. /proc/net/tcp ファイルによる動作確認

以上