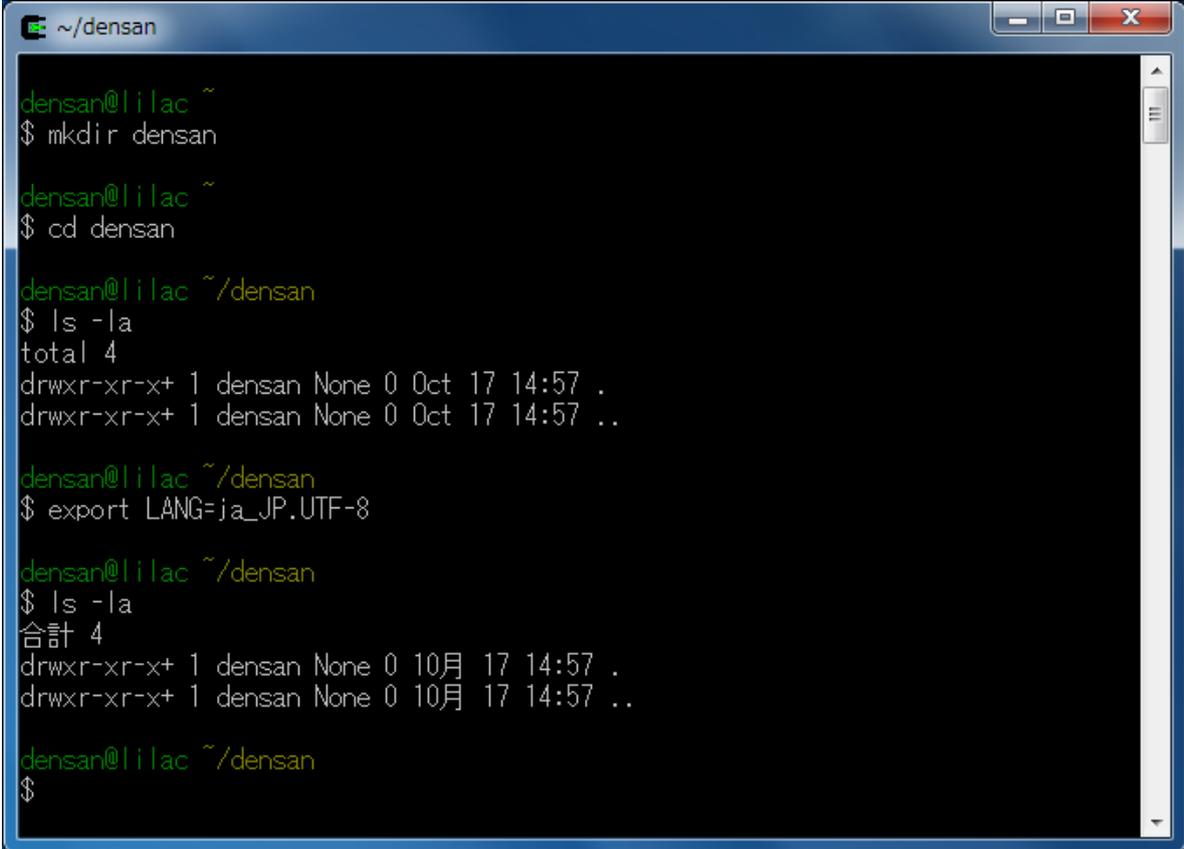


Cygwin 使いこなしまニュアル

2011/10/17

1. Cygwin の起動

Cygwin のインストールが完了したら、一般ユーザーで Windows にログインし、Cygwin を起動してみましょう。下の図は、ユーザー名 `densan` でログインして Cygwin を起動した後、いくつかコマンドを実行してみたところです。



```
~/densan
densan@lilac ~
$ mkdir densan

densan@lilac ~
$ cd densan

densan@lilac ~/densan
$ ls -la
total 4
drwxr-xr-x+ 1 densan None 0 Oct 17 14:57 .
drwxr-xr-x+ 1 densan None 0 Oct 17 14:57 ..

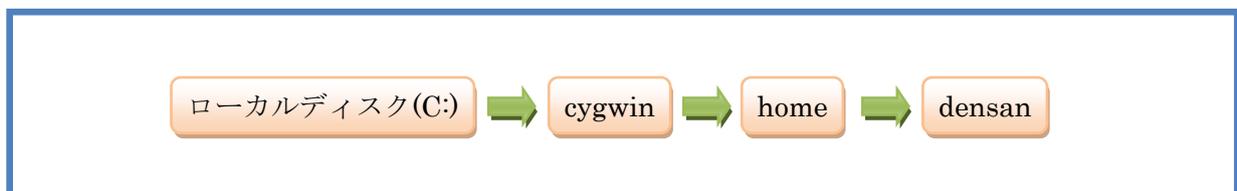
densan@lilac ~/densan
$ export LANG=ja_JP.UTF-8

densan@lilac ~/densan
$ ls -la
合計 4
drwxr-xr-x+ 1 densan None 0 10月 17 14:57 .
drwxr-xr-x+ 1 densan None 0 10月 17 14:57 ..

densan@lilac ~/densan
$
```

2. Cygwin のファイルはどこにある？

Windows から Cygwin 内のファイルを見たい場合は、マイコンピュータを開き、



のようにフォルダを辿っていきます (`densan` のところは各自のユーザー名に変える)。Windows では、ディスク上のこの場所 (path; パス) を `C:\cygwin\home\densan` のように表します。今後、このような表記をしますので、覚えておいてください。

3. Cygwin で日本語を使いたい

Cygwin で以下のコマンドを実行すると、それ以降、メッセージが日本語になります。

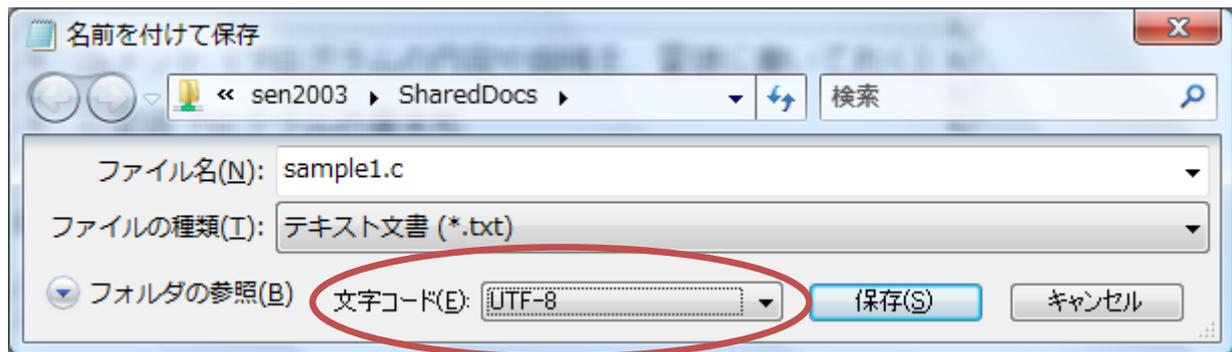
```
export LANG=ja_JP.UTF-8
```

これは、文字コードを日本語の UTF-8 に設定することを意味しています。UTF-8 は Cygwin で標準の文字コードで、多国語に対応しています。

ところで、Windows で標準の日本語文字コードは Shift-JIS (SJIS) と呼ばれるものです。何も意識せずにファイルを作成すると、そのファイル内の日本語文字コードは Shift-JIS になってしまいます。プログラムを Windows のメモ帳や秀丸エディタで作成した場合も同様です。この場合、文字コードを UTF-8 に指定してファイルを保存する必要があります。

Windows のメモ帳で文字コードを UTF-8 にする

メモ帳で日本語を含んだプログラムを作成し、メニューで [ファイル(F)] → [名前を付けて保存(A)] を選びます。以下のようなウインドウが開きますので、文字コードを UTF-8 に指定して、[保存(S)] ボタンをクリックします。秀丸エディタでも同様に、保存時に文字コードの指定ができます。



一度 UTF-8 で保存したファイルは、次回からは、特に何も指定しなくても、UTF-8 で保存されます。電子計算機のサンプルプログラムは UTF-8 で作成されていますので、ホームページから直接ダウンロードした場合（表示画面のコピー&ペーストではなく、リンク先のファイル sample1.c 等を直接保存した場合）は、文字コードを特に意識しなくても大丈夫です。自分で最初からファイルを作成した場合は、上記のように、文字コードを指定してください。なお、文字コードが違ったプログラムを Cygwin で実行すると、文字化けして、何が書いてあるのかわからなくなります。

4. Cygwin の基本コマンド

Cygwin の基本的なコマンドを、以下の表にまとめました。コマンドは基本的に、UNIX や Linux と同じです。詳細な使い方は、man コマンドで調べることができます。

コマンド名	機 能
ls -la	ディレクトリ（フォルダ）内のファイル一覧を見る。
cd ディレクトリ名	ディレクトリ（フォルダ）に移動する。
mkdir ディレクトリ名	ディレクトリ（フォルダ）を作成する。
cp コピー元 コピー先	ファイルをコピーする。
mv 現ファイル名 新ファイル名	ファイル名を変更する。ファイルを移動する。
rm ファイル名	ファイルを削除する。
less ファイル名	ファイルの内容を、画面 1 ページずつ表示する。
man コマンド名	コマンドの使い方を調べる。

プログラムのコンパイルと実行

プログラムのコンパイルと実行は、以下のコマンドで行います。

```
gcc -o sample1 sample1.c
./sample1
```

上記の gcc コマンドにより、C 言語のソースプログラム `sample1.c` から、コンパイル&リンクにより、実行形式ファイル `sample1.exe` が生成されます。`-o` は `output` の意味です。また、拡張子 `.exe` は省略できます。`./sample1` により `sample1.exe` が実行されます。

(注) `sample1.exe` の前についている `./` は、「カレントディレクトリ（今いるディレクトリ）“.”にあるファイル」`sample1` を表わしています。Windows ではパス名の区切りは `¥` ですが、UNIX では `/` を使います（たとえば、`/cygwin/home/densan` 等）。

ヒストリー（コマンド履歴）機能

Cygwin のターミナルにはヒストリー機能が付いています。カーソルキー `↑` `↓` により、以前に入力したコマンド文字列を呼び出し、Delete キーや `←` キーを使って編集後、Enter キーを入力することにより、再実行できます。

5. リダイレクトとパイプ（興味のある人向け）

画面出力をファイルに書き込む（リダイレクト）

多くのメッセージや結果を出力するプログラムでは、出力結果をファイルに残しておきたい場合があります。そんなときは、リダイレクト（`direct` は方向づけること。`re-direct` は向け直す）を使います。

```
./sample3 > out3.txt
./sample3 2> err3.txt
```

1つ目の例は、正常なメッセージ（標準出力）をテキストファイル `out3.txt` に書き込む方法です。記号 `>` が出力先を画面からファイルに切り替える「リダイレクト」を表わしています。2つ目の例は、エラーメッセージのみをテキストファイル `err3.txt` に書き込む方法です。`2>` は標準エラー出力のみをリダイレクトするという意味になります。この例では、標準出力（エラーではないメッセージ）は画面に表示されます。出力されたファイルはエディタで開くことができますし、`less` コマンドを使うと、1ページずつ画面出力できます。

画面出力を別のコマンドに渡す（パイプ）

出力結果をファイルではなく、別のコマンドに渡すためには、パイプを使います。記号は `|` です。

```
./sample3 | less
./sample3 2>&1 | less
```

1つ目の例は、`sample3` の標準出力を `less` コマンドに渡して、1ページずつ画面表示するときに使います。2つ目の例は、標準出力とエラー出力をまとめて `less` コマンドに渡しています。

リダイレクトとパイプの使い道

たとえば、コンパイル時に大量のエラーが出て、画面の遥か彼方までメッセージが出力されたとき、何があっているのかわかりません。エラーは先頭から順に解決していくのが常道ですから、困ってしまいます。こんなときは、

```
gcc -o sample3 sample3.c 2>&1 | less
```

とすると、エラーメッセージは1ページごとに表示され、スペースキーを押すごとに次のページが表示されていきます。また、文字コード変換コマンド `iconv` は、リダイレクトを用いて次のように使います。

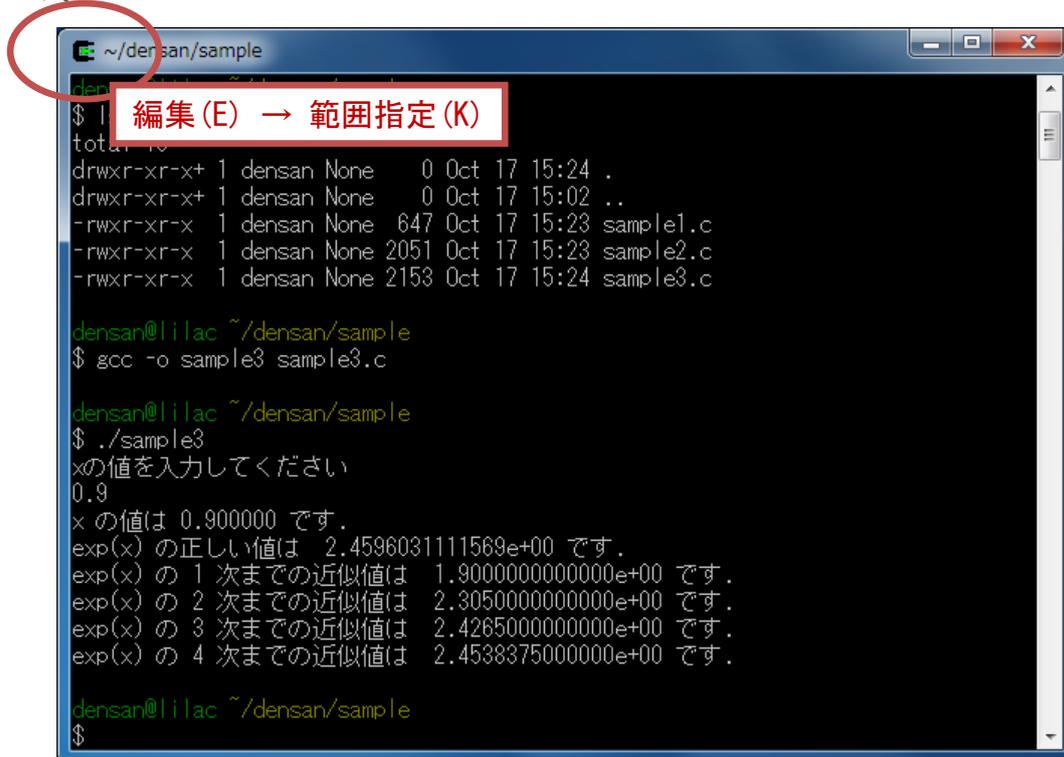
```
iconv -f SJIS -t UTF-8 元のファイル名 > 変換後のファイル名
```

`Shift-JIS` で書かれたファイルを、`UTF-8` に変換するために使います（`f` は `from`、`t` は `to`）。なお、元のファイル名と変換後のファイル名は違うものにしましょう。でないと内容が消えてしまいます。

UNIX (Linux) のコマンドは、いろいろと奥が深いので、興味のある人は調べて見ましょう。

6. Cygwin で提出用課題ファイルの作成

ターミナルウインドウ左上の  マークをクリックし、メニューから [編集(E)] → [範囲指定(K)] を選択します。



```

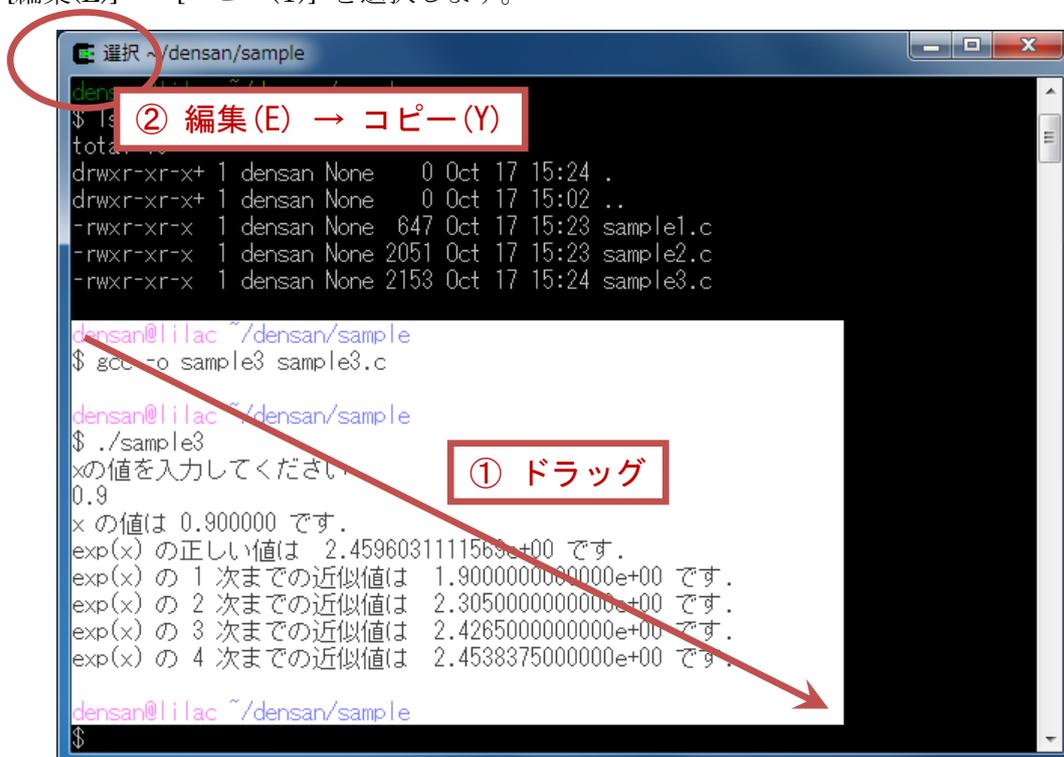
~/densan/sample
$ ls
total 16
drwxr-xr-x+ 1 densan None  0 Oct 17 15:24 .
drwxr-xr-x+ 1 densan None  0 Oct 17 15:02 ..
-rwxr-xr-x  1 densan None  647 Oct 17 15:23 sample1.c
-rwxr-xr-x  1 densan None 2051 Oct 17 15:23 sample2.c
-rwxr-xr-x  1 densan None 2153 Oct 17 15:24 sample3.c

densan@lilac ~/densan/sample
$ gcc -o sample3 sample3.c

densan@lilac ~/densan/sample
$ ./sample3
xの値を入力してください
0.9
xの値は 0.900000 です。
exp(x)の正しい値は 2.4596031111569e+00 です。
exp(x)の1次までの近似値は 1.9000000000000e+00 です。
exp(x)の2次までの近似値は 2.3050000000000e+00 です。
exp(x)の3次までの近似値は 2.4265000000000e+00 です。
exp(x)の4次までの近似値は 2.4538375000000e+00 です。

densan@lilac ~/densan/sample
$
  
```

ターミナル内の選択したい範囲の左上をマウスの左ボタンでクリックし、範囲の右下まで（左ボタンを押したまま）ドラッグすると、選択範囲が反転表示されます。この状態で、左上の  マークをクリックし、[編集(E)] → [コピー(Y)] を選択します。



```

選択 ~/densan/sample
densan@lilac ~/densan/sample
$ ls
total 16
drwxr-xr-x+ 1 densan None  0 Oct 17 15:24 .
drwxr-xr-x+ 1 densan None  0 Oct 17 15:02 ..
-rwxr-xr-x  1 densan None  647 Oct 17 15:23 sample1.c
-rwxr-xr-x  1 densan None 2051 Oct 17 15:23 sample2.c
-rwxr-xr-x  1 densan None 2153 Oct 17 15:24 sample3.c

densan@lilac ~/densan/sample
$ gcc -o sample3 sample3.c

densan@lilac ~/densan/sample
$ ./sample3
xの値を入力してください
0.9
xの値は 0.900000 です。
exp(x)の正しい値は 2.4596031111569e+00 です。
exp(x)の1次までの近似値は 1.9000000000000e+00 です。
exp(x)の2次までの近似値は 2.3050000000000e+00 です。
exp(x)の3次までの近似値は 2.4265000000000e+00 です。
exp(x)の4次までの近似値は 2.4538375000000e+00 です。

densan@lilac ~/densan/sample
$
  
```

次に提出用のテキストファイルを開き、メニューで [編集(E)] → [貼り付け(P)] を選ぶことにより、選択範囲を貼り付けることができます。貼り付けは、Ctrl と v キーを同時に押すだけでも可能です。

課題提出ファイルには、以下のように、プログラムと実行結果を、各週の課題の複数の問いについて1つのテキストファイルにまとめて記載し、提出してください。

kadail.txt (ファイル名は自動変換されるので、何でも良い。ただし、拡張子は .txt にすること。)

```

問1 (問い番号を書くこと)
/*-----
   テーラー展開による、関数の多項式展開
-----*/
#include <stdio.h> /* 入出力関数を使う時につける */
int main(void) {
    int i;          プログラム(1) 型の変数を宣言 */
    double x, exp_x, expTaylor; /* 倍精度実数型の変数を宣言 */
    . . . (略) . . . (プログラムには、要所にコメントを書くこと)
}

tamai@lilac ~/densan/sample
$ ./sample3
x の値を入力してください
0.9
x の値は 0.900000 です.
exp(x) の正しい値は 2.4596031 実行結果(1)
exp(x) の 1 次までの近似値は 1.9000000000000e+00 です.
exp(x) の 2 次までの近似値は 2.3050000000000e+00 です.
. . . (略) . . .

問2 (問い番号を書くこと)
int main(void) {
    . . . (略) . . . プログラム(2)
}
実行結果
. . . (略) . . . 実行結果(2)
. . . (略) . . .

問5
. . . (略) . . . プログラム(5)
. . . (略) . . . 実行結果(5)

```

なお、p.4 で説明した「リダイレクト」を使いこなせる人は、リダイレクトを使って結果をファイルに書き込んでもよいでしょう。