- ¥title{数値計算演習課題}
  ¥date{19.10.05}
  ¥author{--Linuxの基本操作(II Shell)--}

▼ Coin.c

▼ emacsでのCの整形(汚いという批判もありますが．．．)

- C-c C-q
- あるいはindent Coin.c

▼ bob% cat Coin.c

- ```
  #include <stdio.h>
  #include <stdlib.h>
  #include <math.h>

  int
  main(void)
  {
          int             i, j, up;
          int             m, n;
          double          t0, t1, t2;
          double          toss, disp;
          m = 3;
          n = 10;
          t0 = t1 = t2 = 0.0;
          for (j = 0; j < m; j++) {
                  up = 0;
                  for (i = 0; i < n; i++) {
                          toss = (double) random() / RAND_MAX;
                          printf("%10.5f %10.5f\n", toss, floor(toss +
  0.5));
                          up += (int) floor(toss + 0.5);
                  }
                  printf("%4d\n", up);
                  t0 += 1;
                  t1 += up;
                  t2 += up * up;
          }
          disp = sqrt(t2 / t0 - t1 / t0 * t1 / t0);
          printf("OUTPUT: %5d, %10.5f\n", n, disp);
          return 0;
  }
  ```
- gcc -lm -o Coin Coin.c

▼ 課題

- コイン投げのプログラムを改良し，一回あたりに投げるコインの数nとその試行回数mとをscanfで読み込み，nと分散を表示するプログラムを作れ．
- INPUTファイルを用意し，./a.out< INPUTとしてコイン投げを実行せよ．
- n=10から100までを10刻みで出力するshell scriptを作れ．
- 上記のshell scriptを修正して，sedで入力ファイルを変形し，Coin投げのプログラムに食わせて出力をresult.allに追記するようにせよ．
- プログラム，入力ファイル，シェルスクリプトを変更し，コインの枚数と分散を出力するプログラムに仕上げろ．試行回数mを10000，n=10-500として，結果を関連するファイルとともに提出せよ．
- Piの計算，Mandelbrotについてもコイン投げと同様にshellで計算を実行していくように改良せよ．Piでは試行回数とPiの値，Mandelbrotでは分割数を入力にして，出力が(x,y,count)となるようにせよ．

▼ 課題の解説

- unixの基本コンセプトである「できるだけあるものを利用しよう」を実現する手法がフィルタモデルである．これはプログラムが入力をどこから受け取り，どこへ出力するのかを，ユーザーが柔軟に操作できるような枠組み．unixでは多くの小さくてシンプルなユーティリティを提供し，それぞれが引数によって柔軟に調整しながら，ひとつの仕事をうまくこなす．複雑な仕事は，これらのシンプルな操作を組み合わせておこなう．
- 複雑な操作の組み合わせを「フィルタモデル」が担う．また，一度作った一連の複雑な操作をコンピュータに覚えさせるよう，「シェルスクリプト」が用意されている．
- 簡単なコイン投げのプログラムを修正して，この一連のunixの枠組みが利用できるように演習をおこなう．
  ```

- UNIXのフィルタモデル
  - ls -1
  - ls -1 > files.txt
  - sort -r < files.txt
  - ls -1 |sort -r
  - >: cmd > file
    - cmdを実行し，出力をfileを書き込む．fileに何があっても消して，上書きするので注意．
  - >>: cmd >> file
    - cmdを実行し，出力をfileの最後に追記する．
  - <: cmd < file
    - cmdを実行する際に，fileから入力を読み込む．
  - |: cmd1 | cmd2
    - cmd1を実行し，出力をcmd2の入力に渡す．
- shell script
  - パスを通す
    - set
    - env
    - setenv PATH .:$PATH
    - echo $PATH
  - シェルスクリプトの動作の基本
    - editor(emacs)でテキストファイルallcalcを作成
      - <bob> cat allcalc
        #!/bin/tcsh -f
        echo "START calc" > res.all
    - ファイルallcalcに実行権限を与える
      - ls -lで確認
      - chmod +x allcalc
    - allcalcの実行
      - ./allcalc
  - シェルスクリプトの文法
    - ここでは課題をこなすのに最低限必要な文法だけを記す．
      http://echoes.jp/saito/csh.html
      には適切にまとめられている．
    - @: 数値演算
      - @ trial = 10
      - @ trial += 10
    - while
      - while (${trial}<501)
            echo "#TRIAL" ${trial} >> res.all
            @ trial += 10
        end
- sed, awk
  - sed
    - 文字の置換は s/aaa/bbb/g で．
      sed "s/#TRIAL/${trial}/g" INPUT.orig
  - awk
    - awk -F: '{ print $2 }'
    - grep OUTPUT res.all | awk -F: '{print $2}'
    - printf("OUTPUT: %5d, %10.5f\n", n, disp);
      などとしてプログラムの中でマークをつけておく．
- process
  - history
  - ps
    - ps aux
  - kill -TERM PID番号
    - だめなら，-HUP，-KILLを試す．
  - jobs
  - fg,bg