CS第2　テーマ2　各種プログラム

addrnd.rb　乱数を各行の始めに付けるプログラム

```
# 乱数のシード（種）の入力
seed = 123        ←　この値が同じだと同じ乱数列になる

d = Array.new(100)
stop = false
while !stop do
  d = gets().chomp          ← データを 1 行入力
  if d[0] == '0' then       ← 最初の文字が 0 か？
    stop = true                 yes ==> ここで終了
  else                          no  ==> 以下を実行

    # 乱数を付けて，その行を出力する
    r = rand
    printf("%f; ", r)
    puts(d)

  end
end
```

先頭行が０となるまでデータを入力する典型

test.rb　条件式の精度を検査するプログラム

```
a = Array.new(20)
stop = false
num = 0;  numpos = 0;  err = 0;  epos = 0;  eneg = 0    ← 各変数を 0 にセット
while !stop do
  a = gets().split.map(&:to_i)
  if a[0] == 0 then
    stop = true
  else

    num = num + 1
    if a[0] == 1 then  numpos = numpos + 1  end

    hantei = true          ← ここに条件式を書く（現在はすべて true）

    if  hantei && a[0] != 1  then
      epos = epos + 1      ← 偽陽性の数を増やす
      err = err + 1        ← 誤りの数を増やす
    end
    if  !hantei && a[0] != -1  then
      eneg = eneg + 1      ← 偽陰性の数を増やす
      err = err + 1        ←  誤りの数を増やす
    end
  end
end
```

1つのキノコのデータを入力
先頭行が0 か？（終了か？）を判断する

↓結果を画面に出力する部分へ続く

test.rb　条件式の精度を検査するプログラム（結果出力部分）

出力方法は見よう見まねでOKです

得られた結果
num = データ（キノコの）総数
numpos = 毒キノコの数，　numneg = num - numpos 無毒キノコの数
err ＝ 判定間違いをしたキノコの数
epos ＝ 偽陽性（間違って「毒」+1 と判定した）キノコの数
eneg ＝ 偽陰性（間違って「無毒」-1 と判定した）キノコの数

```
printf("*** statisitcs ***¥n")
printf("basic stat: num. = %d, pos.num. = %d (%3.2f)¥n",   ← 出力書式
        num, numpos, numpos.to_f / num.to_f)  ← 実際に出力する値
```

小数扱いにする

```
printf("    error:     %d (%3.2f)¥n", err, err.to_f / num.to_f)
printf("    false positive: %d (%3.2f)¥n", epos, epos.to_f / numneg.to_f)
printf("    false negative: %d (%3.2f)¥n", eneg, eneg.to_f / numpos.to_f)
```

補足：　画面に出力する簡便な方法として puts を使ってきたが，見易さを考えると，コメントや**書式**（出し方）を指定できる printf の方が便利．なお，最後の ¥n は改行のコード．¥ 記号は Mac では Option + ¥ キーで出せる．

count.rb　属性値と毒・無毒の関係を調べるための数勘定のプログラム

```
a = Array.new(20)
num = 0;  numpos = 0;  n42p = 0;  n42n = 0    ←  初期値 0 にセット
stop = false
while !stop do
  1 つのキノコデータを入力. 終わりの行か？のチェック（test.rb と同じ）
  else
    num と numpos の勘定をする（test.rb と同じ）num = num + 1

    # a4 == 2 と a0 = +1/-1 の関係を数えるための部分
    if a[4] == 2 then
      if a[0] == 1 then
        n42p = n42p + 1    ← a[4] == 2 かつ a[0] == 1 となるキノコの数を増やす
      else
        n42n = n42n + 1    ← a[4] == 2 かつ a[0] == -1 となるキノコの数を増やす
      end
    end
  end
end
# 集計と出力
n42 = n42p + n42n              ← a[4] == 2 となるキノコの総数
r42p = n42p.to_f / n42.to_f    ← n42p の n42 に対する割合
r42n = n42n.to_f / n42.to_f    ← n42n の n42 に対する割合
結果を出力する
```

まとめ 条件式(論理式)の Ruby での書き方

【基本関係】

関係	使用例	意味
==	x == y	x は y と等しい
!=	x != y	x は y は等しくない

【論理演算子】

論理記号	使用例	意味
&&	x && y	x と y の論理積 (両方が真のとき真)
||	x || y	x と y の論理和 (少なくとも一方が真のとき真)
!	!x	x の論理否定 (x が真のとき偽, x が偽のとき真)