PC-9801

乱数を順次抽出する方法

ある範囲の乱数の中からランダムに数値を取り出すとき，同じ数値を取り出さないように乱数を選ぶにはどうしたらよいのでしょうか．

たとえば，壷の中に1から100までの番号が付いた玉が入っていて，それをランダムに取り出して再び壷の中に戻さない場合を考えてみてください．

単に1から100までの乱数を発生させて，それに基づいて玉を取り出していたのでは，同じ値の乱数が発生したときは無駄になってしまいます．

それに，残りの玉が少なくなるにつれて，壷の中に残っている玉の番号に該当する乱数が発生する確率は低くなり，時間がかかりすぎてしまいます．

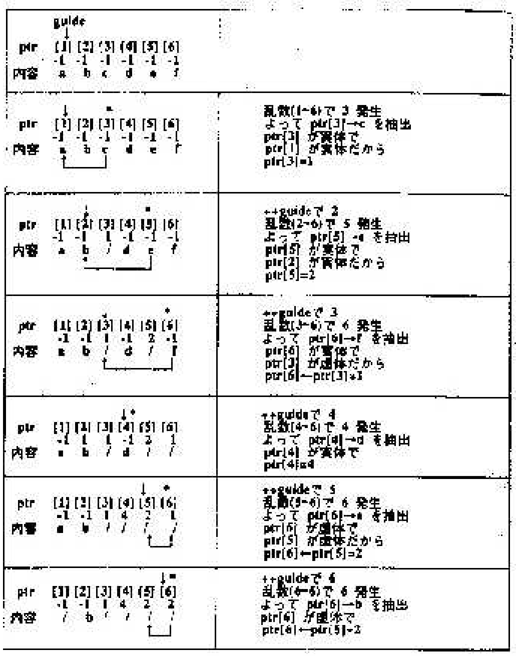
アルゴリズム

考え方としては，一度発生した乱数はまだ発生していない他の乱数を指し示すようにして，同じ乱数が再び発生しても重複しないようにします．そして，乱数が1個発生するたびに乱数の範囲を1だけ減らしていきます．

アルゴリズムの概略は次のようです．

❶配列ptr[n]を用意する．初期値は−1とする．
❷変数guide=1とする．
❸乱数r（範囲は1〜nまでの整数）を発生させる．
❹ptr[r]が実体（−1）ならば，その値rを取り出し，ptr[guide]を調べ，実体（−1）ならば，ptr[r]=guideとする．虚体（ポインタ）ならば，ptr[r]=ptr[guide]とする．虚体（ポインタ）ならば，以下，実体のときと同様．
❺++guideとする．
❻乱数の範囲を，guide〜nまでにする．
❼guide>nなら終了する．そうでないなら❸へ行く．

簡単な例題を示しましたので，確認して見てください．



この方法の欠点としては，乱数の個数と同じだけの配列を用意しなければならないということです．

サンプル・プログラム

サンプル・プログラムとして，n個の乱数（範囲は1〜5000）を順次抽出する "**getrnd.c**" を作りました．

乱数の個数はパラメータで，

```
A>getrnd 500 ⏎
```

のように指示します．パラメータがないと "n=100" になります．

また，実用例としてグラフィック画面を乱数により消去する "g_cls.c" を作りました．

グラフィック画面は640×400ドットで32Kバイトの

大きさがあるので，配列を32,000個用意しなければならず，容量オーバーになってしまいました．そこで，**farmalloc()** を使って配列を確保しました．

プログラム言語はTURBO C ver2.0で，スモール・モデルの環境版でコンパイルしました．

◘参考文献

TURBO C REFERENCE GUIDE, BORLAND

リスト 1　getrnd.c

```
/*  getrnd.c  ver1.0  1991.5.23  Turbo C ver2.0  SmallModel  by S.Furut
                          乱数を順次抽出する
*/

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <conio.h>
#include <dos.h>
#include <string.h>
int ptr[5001];

main(argc,argv)
int  argc;
char *argv[];
{
    unsigned int i,j,n,guide,r,x;

    if (argc==2) n=atoi(argv[1]);
    else          n=100;

    for (i=0;i<=5000;i++) ptr[i]=-1;
    guide=1;
    randomize();

    while (guide<=n) {
        r=random(n-guide+1)+guide;
        if (ptr[r]==-1) x=r;
        else            x=ptr[r];
        if (ptr[guide]==-1) ptr[r]=guide;
        else                ptr[r]=ptr[guide];

        printf("%c.=%u rnd=%u \n",guide,x);

        ++guide;
    }

}
```

リスト 2　g_cls.c

```
/*  g_cls.c  ver1.1  1991.8.3  Turbo C ver2.0  SmallModel  by S.Furuta
                 画面を乱数順次抽出方法で消去する
                 ANSI C 準拠                                  */

#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include <stdlib.h>
#include <alloc.h>

void ClearScreen(void);

char far *ptrB;                         /* GVRAMへのポインタ */
char far *ptrR;
char far *ptrG;
char far *ptrE;
char far *B_ptr;
char far *R_ptr;
char far *G_ptr;
char far *E_ptr;

main()
{
    union REGS in,out;
    int i;

    in.h.ah=0x42;           /* 400ライン */
    in.h.ch=0xc0;
    int86(0x18,&in,&out);
    outp(0x5e,1);           /* アナログ16色 */
    outp(0xa2,0x0d);        /* グラフィック ON */

    B_ptr=MK_FP(0xa800,0);
    R_ptr=MK_FP(0xb000,0);
    G_ptr=MK_FP(0xb800,0);
    E_ptr=MK_FP(0xe000,0);

    for (i=0;i<=31999;i++) {
        ptrR=R_ptr+i;
        *ptrR=0xff;
        ptrB=B_ptr+i;
        *ptrB=0xff;
    }

    clrscr();
    puts("何かキーを押して下さい");
    while (!kbhit()) ;

    ClearScreen();
    outp(0xa2,0x0c);        /* グラフィック OFF */
    clrscr();
    printf("\x1b[>1l");

}
/* 乱数で消す */
void ClearScreen(void)
{
    int far *ptr;
    unsigned int i,j,n,guide,r,x;

    B_ptr=MK_FP(0xa800,0);
    R_ptr=MK_FP(0xb000,0);
    G_ptr=MK_FP(0xb800,0);
    E_ptr=MK_FP(0xe000,0);

    ptr=(int far *)farmalloc(32001*sizeof(int far));
    if (ptr==NULL) {
        printf("memory over\n");
        exit(0);
    }

    for (i=1;i<=32000;i++) ptr[i]=-1;
    n=32000;
    guide=1;
    randomize();

    while (guide<=n) {
        r=random(n-guide+1)+guide;
        if (ptr[r]==-1) x=r;
        else            x=ptr[r];
        if (ptr[guide]==-1) ptr[r]=guide;
        else                ptr[r]=ptr[guide];

        ptrB=B_ptr+(x-1);
        *ptrB=0x00;
        ptrR=R_ptr+(x-1);
        *ptrR=0x00;
        ptrG=G_ptr+(x-1);
        *ptrG=0x00;
        ptrE=E_ptr+(x-1);
        *ptrE=0x00;
        ++guide;
    }

    farfree(ptr);

}
```

愛知県・アラミスタケオ

埼玉県・いずみ北斗

千葉県・伊集院伊織

「これだけ!」
使ってみよう，Turbo C，ちゃん，ちゃん，(ちょっと古かったね)，