情報処理 2 第 6 回

十進BASIC (4) 小休止

桂田 祐史 (かつらだ まさし)

2006 年 5 月 31 日

ホームページは http://www.math.meiji.ac.jp/~mk/syori2-2006/

どうも少し急ぎすぎたようなので、今日は演習時間を十分に取ってレポート課題を着実に消化してもらおうと考えています。

もう課題 4B を出してしまった人は、課題 4B の追加レポートを書くか、十進 BASIC のグラフィックス機能の予習 (チュートリアル『JIS Full BASIC 入門』の 2.6「グラフィックス」の例題を試す) でもして下さい。

1 連絡事項

- もう言う必要はないと思いますが、「情報科学センター生田分室 2006 年度利用登録ページ」[1]
- 課題 3B は大部分の人が提出してくれたようですが、念のため (解説するのは) もう一週待ちます。出しそびれた人はギブアップしないで頑張ってください。

2 課題 4A について

課題 4A については軽く考えていたのですが、こういうのは案外難しいようで、早目にフォローしておきます。

2.1 前回の繰り返し — 結果は読みやすいように

いかなる場合も、レポートは人が読むものだということを忘れないようにして下さい。

今回のように「小数点以下 100 位まで」と言われたら、ぴったりそこまでの結果を示すべきです。1000 桁分の計算結果をそのまま貼付けて「この中に 100 位まで入っているのだから構わないだろう」は甘えすぎです。

[1] https://ikuta-m.isc.meiji.ac.jp/cgi-isc/regist/reg_2006.cgi

それから出来れば、情報はよりアクセスしやすいように、場合によっては加工することを考えてください。この場合、例えば 10 桁あるいは 5 桁ずつ空白で区切るなどすると、ずっと読みやすくなるはずです。例えば 1000 桁分のデータを提出するとしても、10 桁ずつ区切って、1 行に 100 桁あるいは 50 桁ずつ書くことにすると、どこが小数点以下 100 位なのか一目瞭然ですから、認めても良いかな、と思います。

2.2 書式付きの出力

今度は、こちら (桂田) が反省すべき点です。

実はほとんどすべてのプログラミング言語には、出力の書式 (英語で format と言った方がピンと来る？) の指定 (**書式付きの出力**) ができるようになっています。十進 BASIC も例外ではなく、PRINT USING 文があります (十進 BASIC のオンライン・ヘルプを見てもよいですが、付録 A にも引用しておきます)。

```
REM sample-print-using.bas --- PRINT USING の例
OPTION ARITHMETIC DECIMAL_HIGH

PRINT USING "PI=#.#####": PI
PRINT USING "PI=#.##########": PI

REM 100 回 # を打つのは大変なのでちょっと工夫
FMT$="PI=#."&REPEAT$("#", 100)
PRINT USING FMT$: PI
END
```

実行結果

```
PI=3.14159
PI=3.1415926536
PI=3.14159265358979323846264338327950288419716939937510…中略…3421170680
```

この PRINT USING 文を使えば、課題 4A の実行結果を出力する部分は楽だったでしょうか。

実は前回の例題プログラム piarctan.bas でも、PRINT USING を使っていました。

```
print using "πとの差=-%.###^^^^^^";4*s-pi
```

これは誤差が小さくなって -0.000001 のようになると、一体どれくらいの大きさなのか分かりづらくなるので (0 は 6 つ、それとも 7 つ？)、-1.000E-0006 のような、いわゆる**指数形式**[2](exponential form) で表示しようということです。

[2]例えばアボガドロ数が 6.023×10^{23} というように、きわめて大きな数やきわめて小さな数を (1 程度の大きさの数) $\times 10^{\text{なんとか}}$ と表すやり方を指数形式と呼ぶのでした。

3　課題 4B のヒント

3.1　どの方法を使う？

3.1.1　とにかく早く安心したい

とにかくなるべく簡単に済ませようという場合は、Sharp の公式

$$\pi = 6 \arctan \frac{1}{\sqrt{3}}$$

を使うのでしょうか。BASIC では、平方根は `SQR()` という関数で計算できることを知っていれば、`piarctan.bas` をほんの少し直すだけで済むでしょう。

3.1.2　AGM でやってみようか

どうしてそれで良いかを理解するのはさておき、AGM 法の計算 (例えば Salamin-Brent のアルゴリズム) は比較的単純で、十進 BASIC でプログラムを書くのも簡単です。

Salamin-Brent のアルゴリズム(別名 Gauss–Legendre のアルゴリズム)

$a = 1$, $b = 1/\sqrt{2}$ として、

$$(1) \qquad \begin{cases} a_0 := a, \quad b_0 := b, & \\ a_{n+1} := \dfrac{a_n + b_n}{2}, \quad b_{n+1} := \sqrt{a_n b_n} & (n = 0, 1, 2, \cdots), \\ c_n := \sqrt{a_n^2 - b_n^2} & (n = 0, 1, 2, \cdots) \end{cases}$$

で定義された数列を用いて

$$\pi_n := \frac{2a_{n+1}^2}{1 - \sum_{k=0}^{n} 2^k c_k^2}$$

とおくとき、

$$\lim_{n\to\infty} \pi_n = \pi \quad (\text{単調増加}).$$

通常のプログラミング言語では、多数桁の数同士の四則演算を実現するのは大変なのですが、十進 BASIC では 1000 桁までの計算は、特別なことをしないで出来てしまいますから。

3.1.3　色々な arctan 公式を比較する

`http://www.math.meiji.ac.jp/~mk/syori2-2006/jouhousyori2-2006-05/node16.html` で紹介した公式を比較してみるのも面白いです。

こういうのは実際に試してみて初めて分かることが多く、挑戦できる腕力のある人は是非やってみることをお勧めします。

いくつかアドバイスをします。

(1) たくさんの arctan を計算するので、arctan の計算を副プログラム (関数またはサブルーチン) にすることをお勧めします。次のサンプル・プログラムは、piarctan.bas を外部副関数を使って計算するように書き換えたものです。

```
REM piarctan2.bas --- マーダヴァ・グレゴリー・ライプニッツ級数でπを計算
REM arctan の級数の計算は外部副関数に任せることにした。
DECLARE EXTERNAL FUNCTION arctan
INPUT x
INPUT n
LET  s=arctan(x,n)
PRINT "arctan(x)  ";s
print "その４倍";4*s
PRINT USING "πとの差=-%.###^^^^^^":4*s-PI
END

EXTERNAL FUNCTION arctan(x,n)
REM arctan x の級数を第 n 項まで計算
LET  f=-x*x
LET  t=x
LET  s=0
for j=1 to n
   LET  a=t/(2*j-1)
   LET  s=s+a
   LET  t=f*t
NEXT j
LET  arctan=s
END FUNCTION
```

(2) 一体級数の部分和 $s_n(x)$ は、n をどこまで大きく取れば十分な精度を持つのか。一般にはそれほど簡単ではありませんが、今考えている arctan の級数の場合は、各項の絶対値が単調に減少する交代級数になるので、次の Leibniz の定理によって簡単に分かります (s_n の誤差は、次に足す項の絶対値 a_{n+1} 以下である)。

交代級数に関する Leibniz の定理

数列 $\{a_n\}_{n\geq 1}$ は単調減少 $(a_1 \geq a_2 \geq \cdots)$ で、$\lim_{n\to\infty} a_n = 0$ を満たすとき、級数

$$s = \sum_{n=1}^{\infty} (-1)^{n-1} a_n = a_1 - a_2 + a_3 - a_4 + \cdots$$

は収束する。第 n 部分和を s_n とすると、

$$s_{2n} \leq s \leq s_{2n-1} \quad (n \in \mathbf{N}),$$

$$|s - s_n| \leq a_{n+1} \quad (n \in \mathbf{N})$$

が成り立つ。

次に掲げる結果は、100 桁程度の精度を得るために、級数の和を何項加える必要があるか概算し、実際にそれを実行するとどれくらいので精度になるかを計算したものである。

大体どれくらいか…

```
シャープ
第 205 項まで加えて要求精度を達成しました。
 9.81E-101

マチン 2 項公式
第 71 項まで加えて要求精度を達成しました。
第 22 項まで加えて要求精度を達成しました。
 1.20E-101

Gauss 3 項公式
第 40 項まで加えて要求精度を達成しました。
第 29 項まで加えて要求精度を達成しました。
第 22 項まで加えて要求精度を達成しました。
 1.23E-102

Gauss 4 項公式
第 32 項まで加えて要求精度を達成しました。
第 29 項まで加えて要求精度を達成しました。
第 22 項まで加えて要求精度を達成しました。
第 21 項まで加えて要求精度を達成しました。
 1.09E-103

Gauss 9 項公式
第 14 項まで加えて要求精度を達成しました。
第 13 項まで加えて要求精度を達成しました。
第 13 項まで加えて要求精度を達成しました。
第 12 項まで加えて要求精度を達成しました。
第 12 項まで加えて要求精度を達成しました。
第 11 項まで加えて要求精度を達成しました。
第 11 項まで加えて要求精度を達成しました。
第 10 項まで加えて要求精度を達成しました。
第 10 項まで加えて要求精度を達成しました。
 7.45E-106

高野喜久雄の 4 項公式
第 30 項まで加えて要求精度を達成しました。
第 29 項まで加えて要求精度を達成しました。
第 22 項まで加えて要求精度を達成しました。
第 11 項まで加えて要求精度を達成しました。
 7.29E-105

Stormer の 4 項公式
第 29 項まで加えて要求精度を達成しました。
第 22 項まで加えて要求精度を達成しました。
第 18 項まで加えて要求精度を達成しました。
第 13 項まで加えて要求精度を達成しました。
 7.73E-104
```

この結果をどう分析すべきか…書きたくてむずむずするけれど、誰かにやって欲しいので書きません。

A PRINT USING

PRINT 文（書式指定） ☆☆

PRINT USING

PRINT USING 書式指定文字列 : 数値式
書式を指定して数値式の値を表示する。

（1）指数部なしの形
例 PRINT USING "----%.####": a
a の値を整数部（符号を含む）5 桁，小数部 4 桁で表示する。
a= 12.5 のとき " 12.5000"
a=-12.5 のとき " -12.5000"
a= 0 のとき " 0.0000"
書式文字の使い方

(1) 整数部
+または-のいずれかをいくつか並べ，
それに続けて%か#のいずれかをいくつか並べる。
（注）いずれか…一方に限る，いくつか…0 個でもよい
意味
- 数字，負号，空白に置換される。
+ 数字，符号，空白に置換される。
% 数字に置換される。(ゼロを抑止しない)
# 数字，空白に置換する。
(2) 小数部
小数部の桁数を#で示す。
◇ 負数に対する書式には，+ か - を含めなければならない。
★ オプションメニューの「互換性ー動作」で「書式文字に#を用いたとき先行する空白列を生成しない」設定にすると，#を負号に置換することができる（#のみで構成した書式で負数が出力できる）。

（2）指数部ありの形

例 PRINT USING "-%.#####^^^^":a
符号部 1 桁，整数部 1 桁，小数部 5 桁，指数部 (E を含む)4 桁で表示する。
a= 12.5 のとき 1.25000E+01
a=-12.5 のとき -1.25000E+01
a= 0 のとき 0.00000E+00
書式文字の使い方
(1) 整数部と小数部
指数部なしの形の場合と同じ。
(2) 指数部
指数部 (E を含む) の桁数を^で示す（最低 3 個)。

Note

1. 書式指定文字列は文字列式で指定する。
文字列変数や組込み関数を使って書式を構成してよい。

例 PRINT USING "." & REPEAT$("#",15) : 1/7

2. 書式指定文字列に複数の書式を用意しておくと，1 個の PRINT USING 文で複数の数値式を出力できる。

例 PRINT USING "### #.############": n, 1/n

出力する数値式はコンマで区切って書く（セミコロンは使用しない)。
3. 指数部なし書式の整数部にコンマを含めると，その位置にコンマが挿入される。
例 PRINT USING "###,###,###,###,###": 2^32
出力結果 4,294,967,296
4. 書式指定文字列に書式文字以外の文字を含めると，数値に置換されないでそのまま出力される。

例　PRINT USING "A= ### B= ###": A, B

<Note> 書式文字は，次に示す 11 個の文字。

$ % * + , - . < > ^

5. 文字列式の出力に書式指定を用いることができる。

例　PRINT USING "<#####": s$

文字列式に対する書式の作り方

(1) 書式の最初の文字を<，または>にすることができる。

<は左寄せ，>は右寄せを指示する。

<，>のいずれも書かないと，中央寄せになる。

(2) 書式の長さが目的の桁数になるように#を続ける。

(3) 全角文字は 1 個で 2 桁を占める。

6. 書式指定文字列を IMAGE 行で指定することができる。

例

```
50 IMAGE: #####   #.##########
60 PRINT USING 50: N, 1/N
```

<参照> USING$関数