RealNumberCalculator の使い方

Nuget に公開している上記関数電卓ライブラリの説明書です。
そもそも公開したのはライブラリ化し,NET のバージョン違いで動かなかったときに Nuget 化したものは動くのかと試しただけで、一般の方に広く使ってもらうことを意図していませんが、なぜかいくつかダウンロードされており、筆者のアプリの宣伝になるかもとの思惑で公開したままとし本解説を書くこととしました。
これは筆者の作成している商用アプリの一部であり、権利関係は放棄しておらず、ソースも公開しておりません。
また、世間に公開されているオープンソース等プログラムは流用しておらず全て筆者が開発したものです。
本ライブラリの私的以外の利用、リバースエンジニアリング等はおやめください。

本文書最後の C#プログラム例は、Windows コマンドプロンプトで使用できる関数電卓です。標準入力から入力した式を計算し結果を標準出力に出力します。テキストファイルで数式を並べたものを作成し、"コマンド < 入力ファイル名 > 結果ファイル名" で計算結果を追加したファイルを作成します。コードは UTF-8 で chcp 65001 を前に実行の必要があります。筆者のアプリでもこの例と同じく数式文字列渡しで使っており、これ以上の使い方は一般的でない用途かと思われます。

使える関数・コマンドは次ページ以降です。性能は基本的な関数電卓＋α 程度です。分布,検定の類は装備しておりません。
特色として演算は全関数を通じて 10 進数 16 桁で行い、2 進変換誤差は含みません。筆者の確認では関数精度も double よりは高精度になっています。計算中 16 桁を超えるか否かを検知しており、結果が正確な値かを提示できます。加減乗除のみ 16 桁を超えたとき 34 桁まで拡張計算する機能もあります。
割り算は自動で結果を分数で保持・計算します。また平方根・円周率を含む数もある程度まではこの形状で保持・計算します。このように演算精度に拘ったものとなっています。
単純に演算桁数だけで見ると Windows の電卓は関数も 32 桁まで計算しますが 2 進計算です。関数精度に関しては要求されるものに最も合ったものであろうと自負しています。
数式処理と名のれるようなものではありませんが、真似事の類も装備されております。変数 X において数式の簡略化、微分、積分等ができます。(積分は簡単に部分積分、置換積分で可能な場合のみです。数式処理ソフトの積分は多大な辞書を参照している模様です。)

この説明だけで自分で何とか出来る技術のある大人はぜひ試してください。
報告して頂ければバグは修正しますが、実使用において何か損害を被ったとしても対応できません。
筆者のアプリで使っている以上の多大な機能を装備しており、全部はテストできておりません。
コマンドリストは説明がありませんが、筆者の売っている関数電卓がたくさん売れたらもっと説明を拡充した物を作る意思はあります。

使用できる文字・単語

(説明があるのは本機特有でなおかつ一般的な使用で必要であると思った物のみ，基本は関数電卓に準ずる)

数値入力･結果

0,1,2,3,4,5,6,7,8,9,.,-,e,

~ ： 誤差を含む数

` ： 循環小数循環開始位置

″,″,″ ″ ： 数値内区切り

_,/ ： 分数区切り

√‾ ： 平方根(無理数)

数値関数

sin,cos,tan,sin⁻¹,cos⁻¹,tan⁻¹,sinh,cosh,tanh,sinh⁻¹,cosh⁻¹,tanh⁻¹,sec,csc,cot,sec⁻¹,csc⁻¹,cot⁻¹

log₁₀,logₑ,log₂,pow10^,powe^,pow2^,abs,int,ipart,fpart,³√‾,prime ： 前置関数

＋,－,×,÷,＋%,－%,×%,÷%,int÷,mod, ^ ,ˣ√‾,nCr,nPr,GCM,LCM,=,≠,＞,≥,＜,≤ ： 2 項関数

^-¹,^²,^³, ！,% ： 後置関数

% ： パーセント実行用

→Frac ： 分数化

Frac(： 分数化(分母指定)

log(

round(： 丸め

数式入力用

(,)

; ： ( )内数値間区切り [',' は数値内区切り]

if(： 条件分岐

| ： マルチステートメント区切り

定数

π，e

Φ ： 黄金数

科学定数

@c0,@μ0,@ε0,@G0,@G,@h,@ħ,@e,@Φ0,@me,@mp,@mp/me,@α,@α-1,@R∞,@NA,@F,@R,@κ,@σ,

@eV,@u,@Z0,@μB,@a0,@μN,@Eh,@h/2me,@c1,@c2,@gn,@mn,@mμ,@Vm,@RK

乱数

RANDOM,R.DICE,R.COIN,R.INT

乱数関数

rnd-minmax(

メモリ

A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, θ

Val0, Val1, Val2, Val3, Val4, Val5, Val6, Val7, Val8, Val9

メモリへの値保持

➡, M+, M-

アンサーメモリ

ANS

アンサーメモリコマンド

AnswerON, AnswerOFF ： アンサーメモリに結果を保存するか否か

SI 接尾辞入力･結果

da, h, k, M, G, T, P, E, Z, Y, d, c, m, μ, n, p, f, a, z, y ： 長さ

$da^2, h^2, k^2, M^2, G^2, T^2, P^2, E^2, Z^2, Y^2, d^2, c^2, m^2, \mu^2, n^2, p^2, f^2, a^2, z^2, y^2$ ： 面積

$da^3, h^3, k^3, M^3, G^3, T^3, P^3, E^3, Z^3, Y^3, d^3, c^3, m^3, \mu^3, n^3, p^3, f^3, a^3, z^3, y^3$ ： 体積

Ki, Mi, Gi, Ti, Pi, Ei, Zi, Yi ： 2 進接尾辞

□ ： 接尾辞無し

SI 接尾辞コマンド

PfxDim ： 面積/体積指定用

結果コマンド

ImprFrac, MixedFrac, Decimal, Recurring ： 仮分数, 帯分数, 小数, 循環小数

Digit ： 数値桁数

小数結果コマンド

Tab ： 小数部桁数

RoundDown, RoundOff, RoundUp, RoundFloor, RoundCell, RoundEven ： 丸め方法

Normal, Fix, Sci, Eng ： 小数表示法

Prefix ： SI 接尾辞付き結果表示

Exact ： 16 または 34 桁まで高精度結果表示

PuncON, PuncON2, PuncON4, PuncOFF ： 数値区切り位置

FracPuncON, FracPuncOFF ： 小数部数値区切り有無

DoubleON, DoubleOFF ： 加減乗除で 16 桁超え時 34 桁まで高精度計算に切り替えするか否か

ExpInLen, ExpOutLen ： 指数部を数値桁数に含むか否か

EtcShow, EtcHide ： 誤差を示す ~ を結果に含むか否か

分数結果コマンド

FracInLen, FracOutLen ： 数値桁数内に収まる時のみ分数とするか否か

無理数結果コマンド

Commonenomi,IndiDenomi ： 分数通分の有無

循環小数結果コマンド

RecurParensis,RecurMark ： 循環位置を括弧にするか否か

SI 接尾辞結果コマンド

Prefix12ON,Prefix12OFF ： SI 接尾辞結果で指数 1,2,-1,-2 のものを使用するか否か

16 進数数値入力･結果

A,B,C,D,E,F ： 16 進数は 15 桁まで保持する

p ： 2 進数指数の 16 進数表現(C 言語 printf)時の指数指定

N 進数基数記号入力･結果

hex,duo,dec,oct,qui,qua,bin

N 進数変換

→hex,→duo,→dec,→oct,→qui,→qua,→bin

N 進数結果コマンド

BaseON,BaseOFF ： 基数記号を付けるか否か

Complement,HexMinus ： 負数の結果を補数とするか

Fullway,Halfway ： 結果を数値桁数で切るか否か

MagnifyON,MagnifyOFF ： 桁数カウント拡張するか

論理関数入力

NOT,NEG

AND,OR,XOR,NAND,NOR,XNOR

単位(角度)入力･結果

°,′,″,rad,grade,cycle

単位(角度)変換

→°,→′,→″,→rad,→grade,→cycle

単位(角度)結果コマンド

DegMinSec,Degree,Radian,Gradian,Cycle

単位(時間)入力･結果

hour,min,s

AM,PM

単位(時間)結果コマンド

Time12,Time24

単位(日付)入力･結果

year,month,day

単位(日付)結果コマンド

DateLong,DateShort,DateZero,DateSpace

DateYMD,DateMDY,DateDMY,Days360,Days365

単位(温度)変換

°F→°C,°C→°F,°F→K,K→°F,K→°C,°C→K

単位(ヤード・ポンド法)入力･結果

thou,mil,po,P/,barley,in,li,ft,yd,rd,pole,perch,ch,fur,mi,lea,ftm,cb,NM

in²,ft²,yd²,acre,mi²,section,twp,rd²,pole²,perch²,ch²,rood

m.,fl s,fl dr,tsp,Tbsp,fl oz,gi,cp,pt,qt,gal,pk,bu,bbl,hogshead

in³,ft³,yd³,acre-ft

gr,dr,oz,lb,st,qtr,cwt,ton,dwt,oz t,lb t

単位(その他)入力

Å,ua,l.y.,pc,xu

ha,are,b

L,t,carat,knot,Gal

dyn,kgf,lbf

atm,Torr,mHg,bar,mH²O,psi

eV,calIT,calth,BtuIT

hp,PS,rpm

Poise,St,Gs,γ,Oe,Mx,sb,ph,Ci,R,rad(a),rem,g

hare,dare

単位変換

→meter,meter→,→meter²,meter²→,→meter³,meter³→,meter³→in³,→L,L→,L→in³,→gram,gram→

単位(ヤード・ポンド法)コマンド

International,USSurvey,USArmy,UKVolume,USFluid,USBeer,USOil,USDry,USFDA,Canada,Austraria,

ShortMass,LongMass

MiYdFtIn,MiYdIn,MiFtIn,MiYd,MiFt,YdIn,FtIn,GalQtPtOz,GalPtOz,GalPt,GalOz,PtOz,

TonCwtLbOzDr,TonCwtLbOz,TonLbOz,TonLb,LbOz,StLb

換算用関数

→ ： 左右に変換したい任意の単位をつける

統計入力

X-data,Y-data,N-data ： リストを代入

統計結果メモリ

Σx,Σy,Σn,Σx2,Σxy,Σy2,Σx3,Σx2y,Σx4,x-ave,y-ave,δx,δy,sx,sy,

minx,maxx,medianx,Q1x,Q3x,modex,miny,maxy,mediany,Q1y,Q3y,modey,stat-a,stat-b,stat-c,stat-r

統計入力データ用

statsort

統計関数

x',y'

統計コマンド

StatSINGLE,StatLINEAR,StatQUADRA,StatEULAR,StatGENERAL,StatPOWER,StatLOG,StatINVERSE

金融計算入力/結果

Fi-PV,Fi-FV,Fi-PMT,Fi-I%,Fi-N,Fi-C/Y,Fi-P/Y,Fi-Pay

金融計算関数

→nom(,→eff(,bal(,Σprn(,Σint(,npv(,irr(

確率密度関数

P(,Q(,R(

→Z

複素数入力･結果

i,∠

複素関数

conj(,real(,image(,radius(,angle(

→rθ,→xy

複素結果コマンド

ComplexXY,ComplexRθ,NoComplex

リスト入力･結果

{,}

リスト関数

culsum,difflist,sum,prod,sortA,sortD,mean,median,Q1,Q3,mode,stdDev,variant,abs_list

MIN(,MAX(,sum(,prod(,sortA(,sortD(,mean(,median(,Q1(,Q3(,mode(,stdDev(,variant(

dot_prod(,cross_prod(

Lback(,Lfront(

行列入力･結果

[[,][,]]

行列関数

det,trans,identity,ref,rref

rnd-mat(,Rswap(,Radd(,Rmul(,Rmuladd(

Rback(,Rfront(,Cback(,Cfront(

行列コマンド

Horizontal,Vertical ： 行列要素の縦横方向を切り替え

行列・リスト関数

dim,fill(,aug(,getEle(,setEle(,Mat>List(,List>Mat(

Equation(: 方程式の求解

数式入力･結果

X : 数式入力用の変数

数式関数

Σ(,Π(,seq(,∫(,dx(,Value(

Solver(: 数式の求解

多項式・リスト関数

Poly>List(,List>Poly(

多項式関数

Vertex(: 極値の X 座標

結果出力用

±,?

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using RealNumber;

namespace ConsoleApp1 {

    class Program {

//  Length of a display buffer size
        private const int LINE_SIZE = 512;

        static void Main(string[] args) {

            UTF8byteArray OutputBuffer = new UTF8byteArray(LINE_SIZE);
            ErrorCode err_no;

            FunctionTable.FunctionAttributeTableCheck();

            while (true) {
                string InputString = Console.ReadLine();

                if (InputString == null)
                    break;

                if (InputString.Length >= LINE_SIZE - 1) {
                    Console.WriteLine("{0} -
> Error  Can't trest such large size.", InputString);
                    break;
                }

                if ((err_no = CalcExector.CalculatorExecute(ref InputString, out Complex
Struct Num)) != ErrorCode.OK) {
                    Console.WriteLine("{0} -
> {1}", InputString, DispFormatter.errors[(int)err_no]);
                }
                else {
```

```
                    if (Num.type == PairType.REAL)
                        CalcExector.UnitTypeByMode(ref Num.real);
                    DispFormatter.PrintComplexValue(OutputBuffer, in Num);
                    OutputBuffer.count = OutputBuffer.index;
                    OutputBuffer.index = 0;
                    Console.WriteLine("{0} = {1}", InputString, OutputBuffer.StringPart(
));
                }
            }
        }
    }
}
```