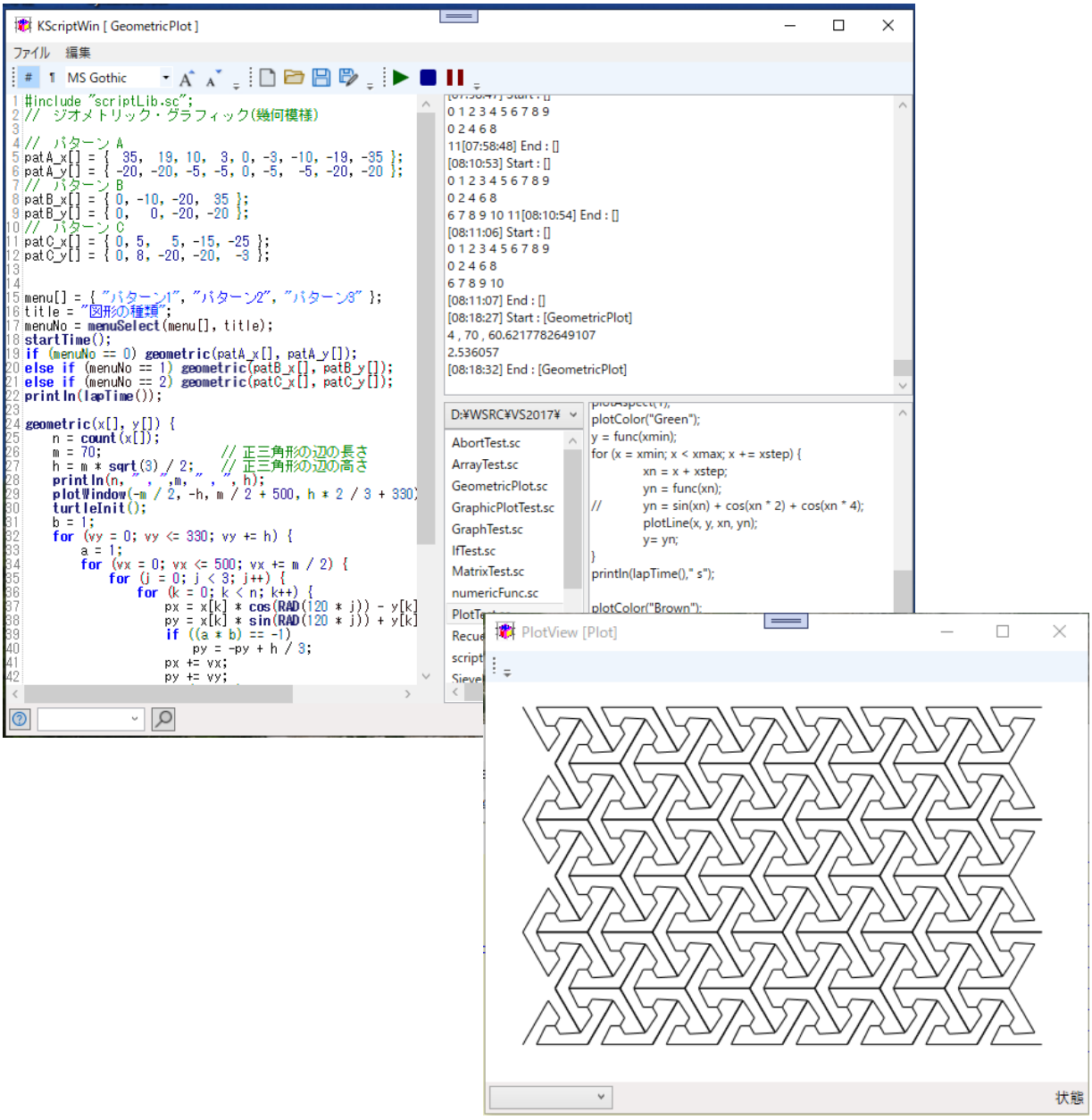
KScriptWin

C 言語風のスクリプト言語とその実行環境

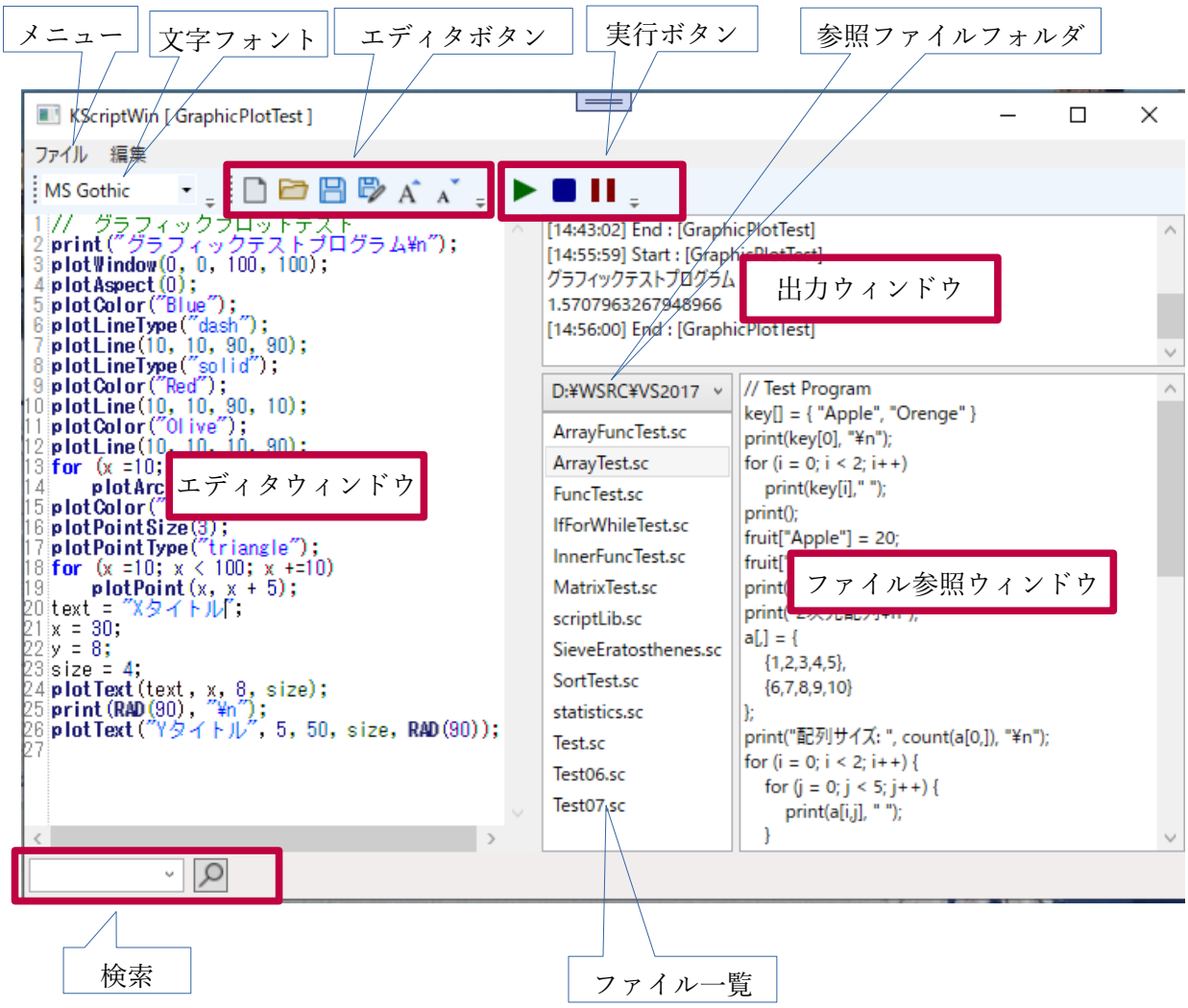
KScript は C 言語に似せたスクリプト言語でインタプリタとして動作する。
自前の CAD ソフト(CadApp や Min3DCad)に組み込んで動作させることを目的に作成した。
基本的な制御構造と型指定のない変数で作られている。

KScriptWin は Windows 上で KScript のコード作成とそれを実行する環境である。



目次

1. 画面の説明



・メニュー

ファイル

新規	エディタウィンドウをクリアし新規スクリプトの作成
開く	ファイル一覧からスクリプトファイルを読み込む
保存	編集したスクリプトをファイルに保存
名前を付けて保存	編集したスクリプトをファイル名を指定してに保存

編集

切取り

コピー

貼付け

検索

置換え

コメント追加/解除　　選択した行のコメント化またはコメント解除

・文字フォント　　エディタの文字フォントの選択

・エディタボタン

新規	エディタウィンドウをクリアし新規スクリプトの作成
開く	ファイル一覧からスクリプトファイルを読み込む
保存	編集したスクリプトをファイルに保存
名前を付けて保存	編集したスクリプトをファイル名を指定してに保存
拡大	エディタの文字の拡大
縮小	エディタの文字の縮小

・実行ボタン

実行	スクリプトの実行
中断	実行中のスクリプトの中断
一時停止	実行中のスクリプトの一時停止、再度押すと再開

・エディタウィンドウ　　スクリプトを作成するウィンドウ
通常のエディタの機能以外に関数などの入力候補機能を追加

・出力ウィンドウ　　print 文のメッセージを出力するウィンドウ

・参照ファイルフォルダ　　参照したいファイルのフォルダの選択
マウスでダブルクリックするとフォルダの登録ができる

・参照ファイル一覧　　指定されたフォルダのファイルの一覧を表示し選択したテキストファイルを参照ファイルウィンドウに表示する

・参照ファイルウィンドウ　　参照用のテキストファイルを表示するウィンドウ

・検索　　入力した文字列の検索

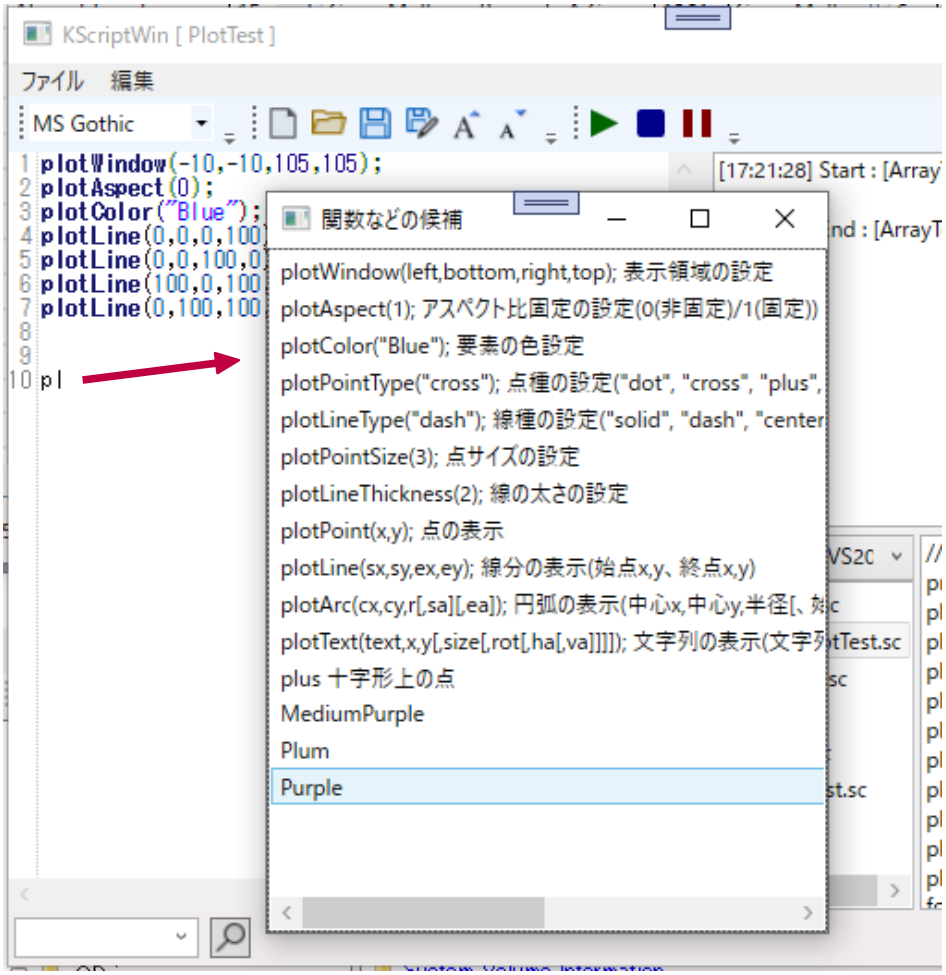
・キー入力

F5	プログラム実行
F8	入力候補
Apps(右 Win キー)	入力候補
Ctrl + C	コピー
Ctrl + D	一時停止
Ctrl + F	検索
Ctrl + N	新規
Ctrl + O	開く
Ctrl + S	上書き保存

Ctrl + V	貼付け
Ctrl + X	切取り
Ctrl + Z	アンドゥ
Ctrl + ‘/’	コメント追加/解除
(Shift + Ctrl + S	名前を付けて保存)

2. 入力候補

エディタウィンドウで入力した文字から検索した関数名や定数などを説明文と合わせて表示し、該当の選択して入力する。
検索対象文字は関数名だけではなく、説明文に含まれる文字も指定できる。
使用方法は検索したい文字列の入力途中で Apps(右側 Win キー)または F8 キーを押すと該当する関数名などが表示される。
下図では pl と入力した後に F8 キーを押すとメニューダイヤログが表示されこの中から選択する。



3. スクリプト文

C 言語風スクリプト言語

変数 ：型指定なし 実数または文字列(ダブルクォーテーション(")で囲む)
```
a = 12.5;
a = "abcd";
```

配列 ：1 次元配列　a[2] = 25;
　　a[m] = "abcd";
　　fruit["apple"] = 50;
2 次元配列　a[1,2] = 30;
　　a[m,n] = 10:
配列一括設定　a[] = { 1, 1, 5, 8 }
　　b[,] = { {1, 2, 3 }, { 2, 3, 4 } }

グローバル変数 :　変数名の先頭が “g_”で始まるものはグローバル変数でそれ以外がローカル変数になる。

```
g_abcd = 2.5;
```

識別子 :　= (代入文) , ; (文末)

演算子 :　* , - , * , / , %(剰余) , ^(べき乗) , ++ , -- , += , -= , *= , /= , ^=

関係演算子 :　== , != , < , > , <= , >=

論理演算子 :　&& , || , !

制御文 :　代入文 :　変数/配列 = 変数|配列|値|文字列|関数|数式 ;
　　if 文 :　if (条件) { 実行文 } [else { 実行分 }]
　　while 文 :　while (条件) { 実行文 }
　　for 文 :　for (初期値; 条件; 更新処理) { 実行文 }
　　return 文 :　return 戻り値 ;
　　break 文 :　break;
　　continue 文 :　continue;
　　exit 文 :　exit;

関数定義 : 関数名(引数 1, 引数 2...) { 実行分; ... return 戻り値; }

プロクラムの構造

```
main() {          main 関数(省略可)
    a = 3;          代入文
    b = 2;
    c = add(a, b);      関数呼び出し
```

```
        println(a + b, " ", c);
}
add(c, d) {                    関数定義
        return c + d;
}
```

4. スクリプト文の書き方

スクリプト文の書き方

スクリプト文(プログラム) は一つの main 関数と複数の sub 関数で構成される。
main 関数の main() { } は省略することもできる。
その中で使用する変数のうちローカル変数はその関数内でしか使用できないがグローバル関数(g_で始まる変数)はすべての関数を通して使用できる。

```
//    main  関数
main() {
    c = sub(a,b);
    println(c);
}
```

```
//    main  関数
//    main() { } は省略可能
c = sub(a,b);
println(c);
```

```
//    関数
sub(d, e) {
    return d - e;
}
```

:

```
//    関数
add(d, e) {
    return d + e;
}
```

関数内の文は基本的にはセミコロン(;)で終了する。

```
a = b + c;
```

{ } で囲まれた所には複数の文を書くことができる。

・{ } なしは 1 文のみ

```
if (a < b)
    println(a);
else
    println(b);
```

・{ } ありは複数文

```
if (a < b) {
   a = a +2;
   println(a);
} else {
   b = b -2;
   println(b);
}
```

コメント

// から行末までをコメントアウトする

/* から */ もコメントアウトするこの場合は複数行になってもかまわない

１行のみコメント化

```
//  コメント
a = a + 2;
```

以下はすべてコメント

```
/*
a = a + 2;
b = a * 4;
*/
```

ローカル変数

値を格納するもので特に型指定の必要はなくて値としては実数と文字列が使える

数値と文字列を足すと文字列として結合する(日本語可))

```
a = 3;
b = "abcd";
c = a + b;
print(c);    →  "3abcd"

住所 = "東京都";
println(住所);          →  東京都
```

グローバル変数

変数名の先頭が “g_” で始まるものはグローバル変数となる

```
g_val = 1;
val = 1;
inc();
println(g_val, ", ", val);      →      2, 1

inc() {
   g_val++;
   val++;
 }
```

配列

複数のデータをまとめて扱うもので[ ]の中の数値を変えて値を格納する

配列のインデックスには文字列も使用できる。
配列の大きさは設定する必要はなく、配列にデータを入れることによって追加されていく。

```
a[0] = 10;
a[1] = 20;
a[2] = 30;
a["apple"] = 10;
```

値をまとめて設定するときは { } で括ってカンマ区切りで入力する
配列のインデックスには文字列も使用でき、値も文字列と数値の混在も可能
一括で値を設定する時は配列のインデックスは 0 から始まる

```
a[] = { 10, 20, 30 };
a[] = { “name”, 18, “men”};
println(a[0], ”,”, a[1], ”,”, a[2]);        → name, 18, men
```

2 次元配列

2 次元配列に一括でデータを設定する場合には { } を 2 重に囲って行う。

```
a[,] = { { 1, 2, 3}, {10, 20, 30} };
println(a[1,0], " , ", a[1,1], " , ", a[1,2]);        → 10, 20, 30
```

行ごとに設定する場合

```
a[2,] = {  100, 200, 300 };
for (i = 0; i < count(a[2,]); i++)
        print(a[2,i], " , ");                          → 100, 200, 300
```

配列のインデックスに文字列を使用

```
person["yamada",] = { "tarou", 68, "men" };
println(person["yamada",0], ",", person["yamada",1], ",", person["yamada",2],);
                                                       → tarou, 68, men

person["山田", "名"] = "太郎";
person["山田", "年齢"] = 68;
person["山田", "性別"] = "男";
println(person["山田", "名"]);                         →  太郎

item[] = { "名", "年齢", "性別" };
for (i = 0; i < count(item[]); i++) {
        print(person["山田",item[i]], " ");
}                                                      →  太郎 68 男
```

演算子

数式を処理するための演算子

+ ：加算　　a = 2 + 3;
- ： 減算　　a = 3 – 2;

*：乗算　　a = 2 * 3;
/：除算　　a = 2 / 3;
%：剰余　　a = 2 % 3;　　(2 / 3 の余り→ 2)
^：べき乗　　a = 2 ^ 3;　　(2^3 → 8)

複合演算子

2 つの演算子を組み合わせて演算をおこなう

+= ： a += 3;　(a = a + 3 と同じ)
-= ： a -= 3;　(a = a - 3 と同じ)
*= ： a *= 3;　(a = a * 3 と同じ)
/= ： a /= 3;　(a = a / 3 と同じ)
%= ： a %= 3; (a = a % 3 と同じ)
^= ： a ^= 3;　(a = a ^ 3 と同じ)
++ ： a++;　　インクリメント(a = a + 1 と同じ)
-- ： a--;　　デクリメント(a = a – 1 と同じ)

比較演算子

条件文の中で 2 つの値を比較する演算子(if 文, while 文, for 文の条件文で使用)
条件文の中に数値や文字が設定された場合は常に true として処理される。

== ：等しい
!= ： 等しくない
< ： 小さい
> ： 大きい
<= ： 小さいか等しい
>= ： 大きいか等しい

```
if (a == b) print(a);
```

条件文の中に数値や文字が設定された場合は常に true として処理される。

```
if (1) println("true"); else println("false");
```
→ true

数値の前に！をつけると false となる

```
if (!1) println("true"); else println("false");
```
→ false

論理演算子

比較演算子の結果を演算する

&& ：a && b　a かつ b
|| ：a || b　a または b
! ： !a　a ではない

※ 論理演算子と比較演算子が混在する場合、優先順位がつかないので括弧をつけて優先順位をつける

a > 10 && a < 20 → (a > 10) && (a < 20)

```
a = 5;
if ((4 < a) && (a < 8))
```

```
    println("5 以上 7 以下");
else
    println("4 以下か 8 以上");
```

→ 5 以上 7 以下

制御文

if 文　：条件によって分岐させる

実行分が複数の場合は { } で囲む必要があるが、単独の場合は省略も可能

if (条件) { 実行文; } [else { 実行文; }]

```
if (a > b) println(a);
```
a より b が大きければ a の値を表示

```
if (a > b) println(a);
else println(b);
```
a より b が大きければ a の値を表示
そうでなければ b の値を表示

```
if (a > b) {
    println(a);
} else if ( a == b) {
    println(a);
    println(b);
} else {
     println(b);
}
```
a より b が大きければ a の値を表示
a と b が同じときは a と b の値を表示
そうでなければ b の値を表示

while 文 ：条件に入っている間繰り返す

while (条件) { 実行文; }

```
n = 0;
while (n < 10) {
    n = n + 1;
    print(n, " ");
}
```
→ 1 2 3 4 5 6 7 8 9 10

```
//          配列の中から s 以上の数を検索
a[] = { 6, 9, 12, 7, 2, 23, 10, 4 };
s = 15;
n = 0;
while ((n < count(a[])) && (a[n++] < s)) ;
println(n, " ", a[n]);
```
→ 6 10　　(検索された配列の次の値を表示)

```
n = -1;
while ((n < count(a[])) && (a[++n] < s)) ;
println(n, " ", a[n]);
```
→ 5 23　　(検索された値を表示)

for 文　：初期値、条件、増分を指定して繰り返す

for (初期値; 条件; 増分) { 実行文; }

```
for (i = 0; i < 10; i++)
   print(i, " ");
```

→　0 1 2 3 4 5 6 7 8 9

```
for (i = 0; i < 10; i += 2)
   print(i, " ");
```

増分を 2 とした場合

→　0 2 4 6 8

return 文 : 処理を中断して呼出し元に戻る

return (戻り値);

```
add(a,b) {
   return a+b;
}

max(a,b) {
   if (a>b)
      return a;
   else
      return b;
}
```

break 文 ：ループの中断

```
n = 0;
while (1) {                    //   条件文に数値を入れると常に条件が成立
   if(10 <n) break;
   n++;
}
```

continue 文　：ループの先頭に戻る

```
n = 0;
while (n++ < 10) {
   if(n < 6) continue;         6 未満の場合は表示しない
   print(n," ");
}
```

→ 6 7 8 9 10

exit 文 : プログラムを終了させる

```
a= 1;
b = a+2;
exit;                          ここでプログラムを打ち切る
println(b);
```

関数の定義

複数の文をまとめて実行する仕組み(日本語可)
関数名(引数 1, ...) { 実行文; ... return 返値; }

```
add(a, b) {
    c = a + b;
    return c;
}
```

呼出し側

```
s = 2;
t = 3;
sum = add(s, t);
```

引数に配列を使う場合

```
sum(a[]) {
    size = arraySize(a[]);
    sum = 0;
    for (i = 0; i < size; i++) {
        sum += a[i];
    }
    return sum;
}
```

呼出し側

```
array[] = { 1, 2, 3, 4, 5 }
total = sum(array[]);
```

ディレクティブ

#include　指定したスクリプトファイルを追加読み込みをおこなう。
作成した汎用の関数を使用するときに使う。

#include “スクリプトファイル名”;

```
#include “scriptLib.sc”;
```

フルパスを指定しない場合には実行するスクリプトファイルと同じディレクトリが指定される。

数式処理以外の関数

print(a, b, "\n");	: コンソールに文字出力(Win 版では出力ウィンドウに出力)
println(a, b);	: print と同じで末尾に改行コードがつく
a = input();	: コンソールからの文字列入力(Win では使用不可、代わりに inputBox() を使用)
c = contain(a[2]);	: 配列の有無(0:なし 1:あり)
size = count(a[]);	: 配列のサイズ
size = count(a[,]);	: 2 次元配列の全体の数
size = count(a[1,]);	: 2 次元配列(2 行目)の列数

max = max(a[]); : 配列の最大値
min = min(a[,]); : 配列の最小値
sum = sum(a[]); : 配列の合計
ave = average(a[,]); : 配列の平均
vari = variance(a[]); : 分散
std = stdDeviation(a[]); : 標準偏差
cov = covariance(a[], b[]); : 共分散
corr = corrCoeff(x[],y[]); : 配列の相関係数
clear(a[]); : 配列をクリア
remove(a[],st[,ed]); : 配列の要素削除(1 次元のみ)
sort(a[]); : 配列のソート(1 次元のみ)
reverse(a[]); : 配列の逆順化(1 次元のみ)
cmd(command); : Windows コマンドの実行
a[,] = unitMatrix(size); : 単位行列
d[,] = matrixTranspose(a[,]); : 転置行列
d[,] = matrixMulti(a[,], b[,]); : 行列の積
d[,] = matrixAdd(a[,], b[,]); : 行列の和
d[,] = matrixInverse(a[,]); : 逆行列 a^-1
d[,] = copyMatrix(a[,]); : 行列のコピー

Win 版固有の関数

a = inputBox(); : 文字入力ダイヤログ
messageBox(outString[, title]); : 文字列のダイヤログ表示
n = menuSelect(menu[],title); : 選択メニューを表示して選択項目 No を返す
plotWindow(left,bottom,right,top); : 表示領域の設定",
plotAspect(1); : アスペクト比固定の設定(0(非固定)/1(固定))",
plotColor("Blue"); : 要素の色設定",
plotPointType("cross"); : 点種の設定("dot", "cross", "plus", "box", "circle", "triangle")
plotLineType("dash"); : 線種の設定("solid", "dash", "center", "phantom")"
plotPointSize(3); : 点サイズの設定
plotLineThickness(2); : 線の太さの設定
plotPoint(x,y); : 点の登録
plotLine(sx,sy,ex,ey); : 線分の登録(始点 x,y、終点 x,y)
plotArc(cx,cy,r[,sa][,ea]); : 円弧の登録(中心 x,中心 y,半径[、始角][、終角])
plotText(text,x,y[,size[,rot[,ha[,va]]]]); : 文字列の登録(文字列,X 座標,Y 座標,サイズ,回転角,水平アライメント,垂直アライメント)
plotDisp(); : グラフィックデータを表示
graphSet(x[],y[][,Title]); :グラフデータの設定(X[],Y[][,Title])
graphFontSize(5); : グラフのフォントサイズの設定"

数式で使う関数

pi = PI; : 円周率(3.14..)
e = E; : 自然対数の底(2.71..)
rad = RAD(deg); : 度をラジアンに変換
deg = DEG(rad); : ラジアンを度に変換
hour = deg2hour(deg); : 度を時単位に変換(360° →24 時間)
deg = hour2deg(hour); : 時単位を度に変換(1 時間 → 15°)
hour = rad2hour(rad); : ラジアンを時単位に変換(PI → 12 時間)
rad = hour2rad(hour); : 時単位をラジアンに変換(12 時間 → PI)
a = mod(x,y); : 剰余(x / y の余り)
a = pow(x,y); : 累乗 (x^y)
a = max(x,y); : x と y の大きい方
a = min(x,y); : x と y の小さい方
a = combi(n,r); : 組合せの数(nCr)
a = permu(n,r); : 順列の数(nPr)
a = sin(x); : 正弦
a = cos(x); : 余弦
a = tan(x); : 正接
a = asin(x); : 逆正接
a = acos(x); : 逆余弦
a = atan(x); : 逆正接
a = atan2(x,y); : 逆正接
a = sinh(x); : 双曲線正弦
a = cosh(x); : 双曲線余弦
a = tanh(x); : 双曲線正接
a = asinh(x); : 逆双曲線正弦
a = acosh(x); : 逆双曲線余弦
a = atanh(x); : 逆双曲線正接
a = exp(x); : e の累乗
a = ln(x); : e を底とする自然対数
a = log(x); : 10 を底とする対数
a = log(x,y); : x を底とする y の対数
a = sqrt(x); : 平方根
a = abs(x); : 絶対値
a = ceil(x); : 切上げ(x 以上で最小の整数値)
a = floor(x); : 切捨て(小数点以下の数の内最大の整数値)
a = round(x); : 四捨五入(もっとも近い整数値)
a = trunc(x); : 浮動小数点の整数部
a = sign(x); : 符号示す値(1/0/-1)
a = round(x,y); : x を y の倍数に丸める
a = equals(x,y); : 等価判定 x == y ⇒ 1, x != y ⇒ 0
a = lt(x,y); : 大小判定(less than) x > y ⇒ 1,以外は 0

a = gt(x,y); : 大小判定(greater than) x < y ⇒ 1,以外は 0
a = compare(x,y); : 大小判定 x > y ⇒ 1,x == y ⇒ 0, x < y ⇒ -1
a = deg2dms(x); : 度(ddd.dddd) → 度分秒(ddd.mmss)
a = dms2dig(x); : 度分秒(ddd.mmss) → 度(ddd.dddd)
a = hour2hms(x); : 時(hh.hhhh) → 時分秒(hh.mmss)
a = hms2hour(x); : 時分秒(hh.mmss) → 時(hh.hhhh)
a = fact(x); : 階乗 (a = x * (x-1) * … * 2 * 1)
a = fib(x); : フィボナッチ数列 (Fn = Fn-1 + Fn-2 , F2 = 1, F1 = 0)
a = gcd(x,y); : 最大公約数
a = lcm(x,y); : 最小公倍数
a = JD(y,m,d); : 西暦年月日からユリウス日を求める
a = MJD(y,m,d); : 西暦年月日から準ユリウス日を求める
a = JD2Date(x); : ユリウス日を年月日に変換して yyyymmdd の実数にする
a = sum(f([@]),n,k); : 級数の和 n から k まで連続し値を計算式 f([@])で演算した値の合計を求める(sum([@]*[@],1,10) → 1*1+2*2+...10*10 → 385)
a = sum(f([@]),n1,n2...nm); : 級数の和 n1 から nm まで値を計算式 f([@])で演算した値の合計を求める (sum([@]^2,1,2,3,4,5) → 1^2+2^2+3^2+4^2+5^2 → 55)
a = product(f([@]),n,k); : 級数の積 n から k まで連続し値を計算式 f([@])で演算した値の積を求める (product([@],1,5) → 120)
a = product(f([@]),,n1,n2...nm); : 級数の積 n1 から nm まで値を計算式 f([@])で演算した値の積を求める (product([@]^2,1,2,3,4,5) → 14400)
a = repeat(f([@],[%]),i,n,k); : 計算式の[@]に n から k まで入れて繰返す,[%]に計算結果が入る,i は[%]の初期値 (repeat([%]*[@],10,2,6) → ((((10*2)*3)*4)*5)*6 → 7200)
例 : 元本 1 万円を年利 2%で 5 年預けた場合の金額
repeat([%]*1.02,10000,1,5) ⇒
((((10000*1.02)*1.02)*1.02)*1.02))*1.02 = 11040.808