Maple 入門

西谷@関西学院大・理工

平成 24 年 9 月 19 日

目 次

第１章　基本操作

1.1　最初の一葉 (FirstLeaf):文法とヘルプとプロット

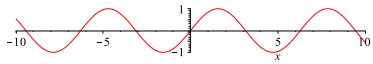
[[[解説]]]

1.1.1　入力領域と注意点 (ShiftEnter)

Maple を起動すると赤いプロンプトがともっている．ここに命令（コマンド）を打ち込んで Maple の計算部に『こちらの意志』を伝えて動作させる．例えば，

```
> plot(sin(x),x);
```

と入力し，enter を入力してみよ．sin 関数がプロットされる．



1. 赤い領域のどこにカーソルがあっても enter を入れれば，そのブロックごと Maple に命令として渡される．テキストでは enter を省略している．
2. テキストの修正は普通のワープロソフトと同じ．
3. 命令の入力ではなく，**改行だけをいれたいときは shift+enter を入れる．**
4. 命令は，enter を入れた順に解釈されるのであって，テキストの上下とは関係ない．

1.1.2　命令コマンドの基本形 (command();)

命令コマンドは全て次のような構造を取る．

```
> command(引数 1, 引数 2,...);
```

あるいは

```
> command(引数 1, オプション 1, オプション 2,...);
```

となる．

1. () の中の引数やオプションの間はコンマで区切る．
2. 最後の;（セミコロン）は次のコマンドとの区切り記号．
3. セミコロン (;) をコロン (:) に替えると Maple からの返答が出力されなくなるが，Maple への入力は行われている．
4. C 言語などの手続き型プログラミング言語の標準的なフォーマットと同じ．

命令コマンドを，英語の命令文と解釈してもよい．たとえば，

```
sin(x) を x について 0 から pi まで plot せよ．
```

という日本語を英語に訳すと，

```
plot sin(x) with x from 0 to Pi.
```

となる．この英語を Maple 語に訳して

```
> plot(sin(x),x=0..Pi);
```

となったとみなせる．英語文法の Verb (動詞), Object (目的語) を当てはめると，Maple への命令は，

```
> Verb(Object, その他の修飾);
```

である．英文でピリオドを忘れるなと中学時代に言われたのと同じく，Maple でセミコロンを忘れぬように．

1.1.3　ヘルプ (?)

ヘルプは少し違った構文で，例えば先程のコマンド plot のヘルプを参照するときには，

```
> ?plot;
```

である．

ヘルプ画面は，左側に操作アイコン，検索ウィンドウ，関連リストが表示され，右側にヘルプの本文がある．本文は，簡単な意味と使い方，説明，例，参照で構成される．ほとんどが日本語に訳されているが，古いテキストやあまり使わないコマンドは英語のまま．英語が分からなくても例を参考にすればだいたい予測できる．と言うより，日本語訳を読んでも初めはチンプンカンプン．Maple コマンドのコンセプトに慣れるまでは使用例をまねるのが一番の早道でしょう．

[[[課題]]]

plot に関する課題

1. plot に関する help を開けよ．
2. 例をいくつか copy して実行せよ．円をパラメータプロットする方法を確認せよ．
3. 2 つの関数，sin(x) と cos(x)，を x=-Pi..Pi で同時にプロットせよ．
4. 上記の 2 つの関数の表示で，オプションに color=[red,blue] および style=[point,line] を加えて，結果を観察せよ．plot[options] に関する help を開け，そのほかのオプションを試してみよ．

plot3d に関する課題

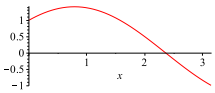
1. x,y の 2 次元平面を定義域とする関数のプロットには，3 次元で表示するコマンド plot3d が必要となる．2 変数関数 sin(x)*cos(y) を x=-Pi..Pi, y=-Pi..Pi でプロットせよ．
2. プロットをつまんで回してみよ．また，メニューバーにある表面あるいは軸のアイコンを操作して，等高線にしたり，軸を変更してみよ．

[[[解答例]]]

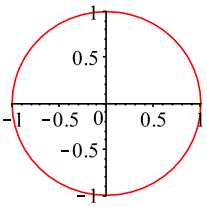
plot

1. ```
   > ?plot;
   ```

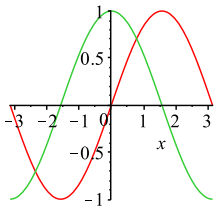
   ```
   > plot(cos(x)+sin(x), x = 0 .. Pi);
   ```


   ```

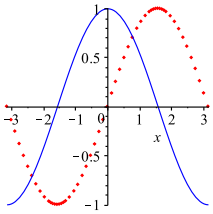
2. `> plot([sin(t), cos(t), t = -Pi .. Pi]);`



3. `> plot([sin(x),cos(x)],x=-Pi..Pi);`



4. `> plot([sin(x),cos(x)],x=-Pi..Pi,color=[red,blue],style=[point,line]);`



plot3d

plot 図に対するいくつかの操作は，plot を選んだときに window の上部に表示されるリボンのアイコンにあります．同じ操作は，plot3d に与える options によっても可能です．以下でいくつかの option を示します．詳しくは`?plot3d[option];`を参照．

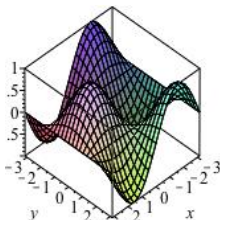
1. `> plot3d(sin(x)*cos(y),x=-Pi..Pi,y=-Pi..Pi);}`

2. 等高線は

   `> plot3d(sin(x)*cos(y),x=-Pi..Pi,y=-Pi..Pi,style=contour);`

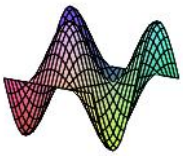
軸の変更は

```
> plot3d(sin(x)*cos(y),x=-Pi..Pi,y=-Pi..Pi,axes=boxed);
```



3. デフォルトの角度も plot3d の option で変更することが可能です.

```
> plot3d(sin(x)*cos(y),x=-Pi..Pi,y=-Pi..Pi,orientation=[45,80]);
```



[[[補足]]]

間違い (Error)

いくつかの典型的な間違い. 先ずは, 左右の括弧の数が合ってないとき.

```
> plot(sin(x,x=-Pi..Pi);
```

```
Error, ';' unexpected
```

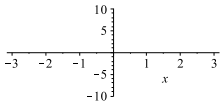
正しくは,

```
> plot(sin(x),x=-Pi..Pi);
```

です. 関数の中に変数が残ったまま plot しようとしたとき.

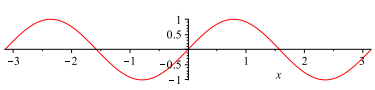
```
> plot(sin(a*x),x=-Pi..Pi);
```

```
Warning, unable to evaluate the function to numeric values in the region; see
the plotting command's help page to ensure the calling sequence is correct
```



何も表示されない. x に 1 を代入しても, sin(a*x) からは数値ではなく記号で答えが返って来ている. plot は関数が数値を返したときしか表示できない. 以下のように, 変数 a のかわりに数値を入れる.

```
> plot(sin(2*x),x=-Pi..Pi);
```



1.2 初等関数とそのほかの関数 (Functions)

[[[解説]]]

1.2.1 初等関数 (ElementaryFunctions)

四則演算と evalf 四則演算は"+-*/". 割り切れない割り算は分数のまま表示される.

```
> 3/4;
```

$$\frac{3}{4}$$

強制的に数値 (浮動小数点数) で出力するには evalf を用いる.

```
> evalf(3/4);
```

$$0.7500000000$$

多項式関数 (polynom) かけ算も省略せずに打ち込む必要がある. またベキ乗は"^"である.

```
> 3*x^2-4*x+3;
```

$$3x^2 - 4x + 3$$

平方根 (sqrt) 平方根は square root を略した sqrt を使う.

```
> sqrt(2);
```

$$\sqrt{2}$$

三角関数 (trigonal) sin, cos などの三角関数はラジアンで入力する. ただし, $\sin^2 x$ などは

```
> sin^2 x;
```

```
Error, missing operator or ';'
```

ではだめで,

```
> sin(x)^2;
```

$$\sin^2 x$$

と省略せずに打ち込まねばならない. 三角関数でよく使う定数 π は"Pi" と入力する. Maple は大文字と小文字を区別するので注意.

ラジアン (radian) に度 (degree) から変換するには以下のようにする.

```
> convert(90*degrees, radians);
  convert(1/6*Pi,degrees);
```

$$\frac{1}{2}\pi$$

$$30\ degrees$$

その他の関数 (inifnc)　その他の初等関数やよく使われる超越関数など，Maple の起動時に用意されている関数のリストは，

```
> ?inifnc;
```

で得られる．

1.2.2　ユーザー定義関数 (unapply)

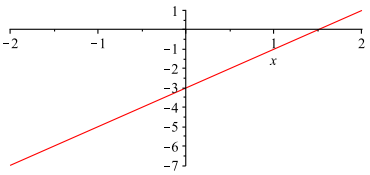
初等関数やその他の関数を組み合わせてユーザー定義関数を作ることができる．
関数 $f(x) = 2x - 3$ とおくとする場合，Maple では，

```
> f:=x->2*x-3;
```

$$f := x \to 2x - 3$$

と，矢印で書く．これが関数としてちゃんと定義されているかは，いくつかの数値や変数を $f(x)$ に代入して確認する．

```
> f(3);                #res: 3 (以降出力を省略する場合はこのように表記)
  f(a);                #res: 2 a - 3
  plot(f(x),x=-2..2);
```



もう一つ関数定義のコマンドとして次の unapply も同じ意味である．

```
> f:=unapply(2*x-3,x);
```

$$f := x \to 2x - 3$$

ただし，矢印での定義ではときどき変な振る舞いになるので，unapply を常に使うようにこころがけたほうが安全．

1.2.3　package の呼び出し (with)

Maple が提供する膨大な数の関数から，目的とするものを捜し出すには help を使う．普段は使わない関数は，使う前に明示的に呼び出す必要がある．例えば，線形代数によく使われる関数群は，

```
> with(LinearAlgebra):
```

としておく必要がある．この他にもいくつもの有益な関数パッケージが用意されている．

```
> ?index[package];
```

で用意されているすべての package が表示される．

[[[課題]]]

関数についての課題

1. evalf のヘルプを参照して，Pi を 1000 桁まで表示せよ．
2. 正接関数 (tan) とその逆関数 arctan を x=-Pi/2..Pi/2,y=-Pi..Pi,scaling=constrained で同時にプロットせよ．
3. 対数関数は ln(x) で与えられる．ヘルプを参照しながら次の値を求めよ．

$$\log_{10} 1000, \log_2 \frac{1}{16}, \log_{\sqrt{5}} 125$$

4. 次の関数は $y = 2^x$ とどのような位置関係にあるか x=-5..5,y=-5..5 で同時にプロットして観察せよ．

$$y = -2^x, y = (1/2)^x, y = -(1/2)^x$$

5. 指数関数は exp で与えられる．e^x と $\log x$ 関数および $y = x$ を同時に x=-5..5,y=-5..5 で plot せよ．またそれらの関数の位置関係を述べよ．
6. 階乗関数 factorial に 3 を代入して何を求める関数か予測せよ．ヘルプを参照し，よりなじみの深い表記を試してみよ．

[[[解答例]]]

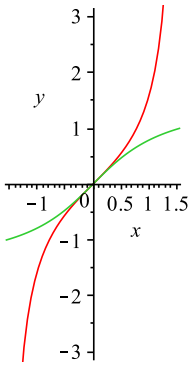
Functions

1. evalf のヘルプを参照して，Pi を 1000 桁まで表示せよ．

```
> ?evalf;
> evalf[1000](Pi);        #省略
```

2. 正接関数 (tan) とその逆関数 arctan を x=-Pi/2..Pi/2,y=-Pi..Pi,scaling=constrained で同時にプロットせよ．

```
> plot([tan(x),arctan(x)],x=-Pi/2..Pi/2,y=-Pi..Pi,scaling=constrained);
```



3. 対数関数は ln(x) で与えられる．ヘルプを参照しながら次の値を求めよ．

$$\log_{10} 1000, \log_2 \frac{1}{16}, \log_{\sqrt{5}} 125$$

```
> ?ln;
```

```
> log10(1000);
```

3

```
> log[2](1/16);
```

−4

```
> log[sqrt(5)](125);
```

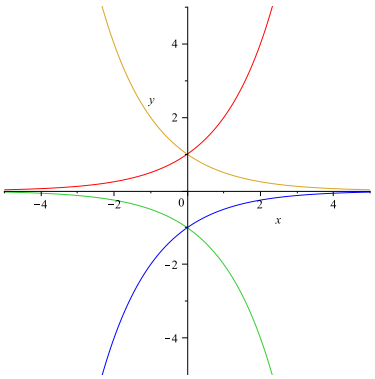
6

4. 次の関数は $y = 2^x$ とどのような位置関係にあるか x=-5..5,y=-5..5 で同時にプロットして観察せよ．

$$y = -2^x, y = (1/2)^x, y = -(1/2)^x$$

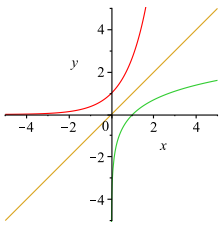
```
> plot([2^x,-2^x,(1/2)^x,-(1/2)^x],x=-5..5,y=-5..5);
```

注目している関数以外を消せばはっきりするが，i) x 軸に対称，ii) y 軸に対称, iii) 原点に対称．



5. 指数関数は exp で与えられる．e^x と $\log x$ 関数および $y = x$ を同時に x=-5..5,y=-5..5 で plot せよ．またそれらの関数の位置関係を述べよ．

```
> plot([exp(x),log(x),x],x=-5..5,y=-5..5);
```



6. 階乗関数 factorial に 3 を代入して何を求める関数か予測せよ．ヘルプを参照し，よりなじみの深い表記を試してみよ．

```
> factorial(3)
```

6

```
> ?factorial;
> 3!;
```

1.3 等号 (Equals)

[[[基本事項]]]

等号の意味 等号は，数学でいろいろな意味を持つことを中学校で学ぶ．それぞれの状況による意味の違いを人間は適当に判断できるが，プログラムである Maple では無理．Maple では，それぞれ違った記号や操作として用意され，人間が Maple に指示する必要がある．

1.3.1 変数への代入:= (colonequal)

変数に値を代入する時には:= (colonequal) を使う．例えば，

a=3，b=2 のとき，a+b はいくらか？

という問題を，Maple で解かす時には，

a に 3，b に 2 を代入したとき，a+b はいくらか？

と読み直し，

```
> a:=3; #res: 3
> b:=2; #res: 2
> a+b;  #res: 5
```

式の定義も同様．以下は $ax+b=cx^2+dx+e$ という式を eq1 と定義している．

```
> eq1:=a*x+b=c*x^2+d*x+e;
```

$$3x+2=cx^2+dx+e$$

a,b に値が代入されていることに注意．

1.3.2 変数の初期化 (restart)

一度何かを代入した変数を何も入れていない状態に戻す操作を変数の初期化という．すべての変数を一度に初期化するには，

```
> restart;
```

とする．なにか新たなひとまとまりの作業をするときには，このコマンドを冒頭に入れることを習慣づけるように．

作業の途中でひとつの変数だけを初期化するには，シングルクォート ’ でくくる．

```
> a:='a';
```

$$a$$

一時的代入に subs がある．

1.3.3 方程式の解 (solve)

3x=2 を満たす x をもとめよ．

という問題は，

```
> solve(3*x=2,x);
```

$$\frac{2}{3}$$

連立方程式は以下のとおり．

```
> solve({x+y=1,x-y=2},{x,y});
```

$$\left\{x = \frac{3}{2}, y = -\frac{1}{2}\right\}$$

ただし，solve だけでは，x,y に値は代入されない．

```
> sol1:=solve({x+y=1,x-y=2},{x,y});
> assign(sol1);
```

$$sol1 := \left\{x = \frac{3}{2}, y = -\frac{1}{2}\right\}$$

とする必要がある．確認してみると

```
> x,y;
```

$$\frac{3}{2}, -\frac{1}{2}$$

となり，値が代入されていることがわかる．

方程式の数値解 (fsolve)　解析的に解けない場合は，数値的に解を求める fsolve を使う．上で x に assign しているので，x を初期化している．

```
> x:='x';
> fsolve(log(x)-exp(-x),x);
```

$$x := x$$

$$1.309799586$$

1.3.4　恒等式 (Identity)

式の変形にも等号が使われる．例えば，

$$(x-2)^2 = x^2 - 4x + 4$$

というのが等号で結ばれている．式の変形とは，変数 x がどんな値であっても成り立つ恒等的な変形である．

この式変形も，問題としては，

(x-2)^2 を展開 (expand) せよ

と与えられるので，そのまま Maple コマンドに読み替えて

```
> expand( (x-2)^2 );
```

$$x^2 - 4x + 4$$

とすればよい．因数分解 (factor) や微分 (diff)・積分 (int) も同様に等号で結ばれるが，Maple には操作を指示する必要がある．詳しくは他の単元で．

[[[課題]]]

等号についての課題

1. a=3, b=4 として a,b の四則演算をおこなえ．また，べき乗 a^b を求めよ．
2. eq1=3*x+4=2*x-2, a=2 とした場合の eq1/a,eq1+a を試し，両辺を観察せよ．
3. 3 点 (1,2),(-3,4),(-1,1) を通る 2 次方程式を求めよ．
4. 方程式 $\sin(x+1) - x^2 = 0$ の 2 つの解を fsolve のヘルプを参照して求めよ．

[[[解答例]]]

Equals

1. a=3, b=4 として a,b の四則演算をおこなえ．また，べき乗 a^b を求めよ．

```
> a:=3:
> b:=4:
> a+b;a-b;a*b;a/b;a^b;        #省略
```

2. eq1=3*x+4=2*x-2, a=2 とした場合の eq1/a,eq1+a を試し，両辺を観察せよ．

```
> eq1:=3*x+4=2*x-2;
> a:=2;
> eq1/a;
> eq1+2;
```

$$eq1 := 3x + 4 = 2x - 2$$
$$a := 2$$
$$\frac{3}{2}x + 2 = x - 1$$
$$3x + 6 = 2x \tag{1.1}$$

3. 3 点 (1,2),(-3,4),(-1,1) を通る 2 次方程式を求めよ．

   まず 2 次関数を定義する．

```
> restart;
> f:=x->a*x^2+b*x+c;
```

$$f := x \mapsto ax^2 + bx + c$$

(1,2) を通ることから，f(1)=2 が成立．これを eq1 として保存．

```
> eq1:=f(1)=2;
```

$$eq1 := a + b + c = 2$$

他の点も同様に

```
> eq2:=f(-3)=4;
> eq3:=f(-1)=1;
```

$$eq2 := 9\,a - 3\,b + c = 4$$
$$eq3 := a - b + c = 1$$

この 3 個の連立方程式から，a,b,c を求めれば解となる．

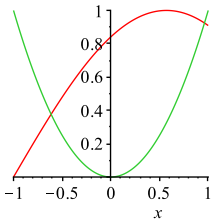
```
> solve({eq1,eq2,eq3},{a,b,c});
```

$$\{a = 1/2, b = 1/2, c = 1\}$$

4. 方程式 $\sin(x+1)-x^2=0$ の 2 つの解を fsolve のヘルプを参照して求めよ．

   まず，2 つの関数とみなしてプロット．

```
> plot([sin(x+1),x^2],x=-1..1);
```



   解が 2 つあることに注意．与えられた関数値が 0 となる方程式として定義し，これを solve でとく．

```
> eq1:=sin(x+1)-x^2=0;
> solve(eq1,x);
```

$$eq1 := \sin(x+1) - x^2 = 0$$
$$-1 + RootOf\left(-\sin(\_Z) + 1 - 2\,\_Z + \_Z^2\right)$$

   これでは解を求めてくれないので，fsolve で数値解を求める．

```
> fsolve(eq1,x);
```

$$0.9615690350$$

   これでは x の負にあるもう一つの解がでない．これを解決するには，fsolve で x に初期値を入れて実行する．

```
> fsolve(eq1,x=-1..0);
```

$$-.6137631294$$

第2章　微積分

2.1　微分 (Diff)-I

[[[解説]]]

2.1.1　単純な微分 (diff)

単純な一変数関数の一次微分は，以下の通り．

```
> diff(x^2-3*x+2,x); #res: 2x-3
```

高次の微分は，微分変数を必要なだけ並べる．

```
> diff(sin(x),x,x); #res: -sin(x)
```

さらに高次では次のように$を使った記法が便利．これは x についての 3 次微分を表わす．

```
> diff(x^4,x$3); #res: 24x
```

偏微分 (PartialDiff)　複数の変数を持つ多変数の関数では，微分する変数を明示すれば偏微分が求められる．

```
> eq1:=(x+y)/(x*y);
> diff(eq1,x);
```

$$eq1 := \frac{x+y}{xy}$$

$$\frac{1}{xy} - \frac{x+y}{x^2 y}$$

[[[例題]]]

2.1.2　例題:関数の微分と増減表

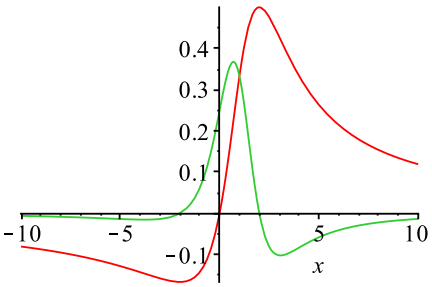
次の関数とその 1 次導関数を同時にプロットし概形を確認し，さらに増減表を求めよ．

$$\frac{x}{x^2-2x+4}$$

解答例

```
> f0:=unapply(x/(x^2-2*x+4),x):
> df:=unapply(diff(f0(x),x),x);
> plot([f0(x),df(x)],x);
```

$$df := x \mapsto \left(x^2 - 2\,x + 4\right)^{-1} - \frac{x\,(2\,x-2)}{\left(x^2 - 2\,x + 4\right)^2} \tag{2.1}$$



2.1.3　例題:接線 (Tangent)

次の関数の $x = 3$ での接線を求め，2 つの関数を同時にプロットせよ.

$$y = x^3 - 2\,x^2 - 35\,x$$

解答例　与関数を f0 と定義.

```
> f0:=unapply(x^3 - 2*x^2 - 35*x,x);
```

$$f0 := x \mapsto x^3 - 2\,x^2 - 35\,x$$

微分関数を df と定義

```
> df:=unapply(diff(f0(x),x),x);
```

$$df := x \mapsto 3\,x^2 - 4\,x - 35$$

接点 (x0,f0(x0)) で傾き df(x0) の直線を f1 と定義.

```
> x0:=3;
> eq1:=df(x0)*(x-x0)+f0(x0);
> f1:=unapply(eq1,x);
```

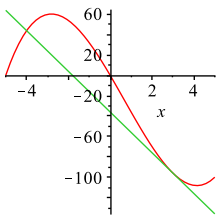
$$x0 := 3$$

$$eq1 := -20\,x - 36$$

$$f1 := x \mapsto -20\,x - 36$$

2 つの関数を同時にプロット.

```
> plot([f0(x),f1(x)],x=-5..5);
```



2.2 微分 (Diff)-II

[[[解説]]]

2.2.1 級数展開 (series)

Taylor 級数は以下のようにして，中心点 (x=a)，次数 (4 次) を指定する．

```
> t1:=series(sin(x),x=a,4);
```

$$t1 := \sin(a) + \cos(a)(x-a) - \frac{1}{2}\sin(a)(x-a)^2 - \frac{1}{6}\cos(a)(x-a)^3 + O((x-a)^4)$$

誤差の次数を示す $O((x-a)^4)$ があるため，このままでは関数として使えない．unapply で定義関数として使うためには，convert を使って多項式 (polynomial) に変換しておく．

```
> e1:=convert(t1,polynom);
> f1:=unapply(e1,x);
```

$$t1 := \sin(a) + \cos(a)(x-a) - \frac{1}{2}\sin(a)(x-a)^2 - \frac{1}{6}\cos(a)(x-a)^3$$

$$f1 := x \mapsto \sin(a) + \cos(a)(x-a) - \frac{1}{2}\sin(a)(x-a)^2 - \frac{1}{6}\cos(a)(x-a)^3$$

2.2.2 全微分 (D)

全微分を計算するときは，D を用いる．

```
> f:=unapply(x^4*exp(-y^2),(x,y));
> D(f(x,y));
> (D@@2)(f(x,y));
```

$$f := (x,y) \mapsto x^4 \exp(-y^2)$$

$$4\,D(x)\,x^3\exp(-y^2) + x^4 D\left(\exp(-y^2)\right)$$

$$4\left(D^{(2)}\right)(x)\,x^3\exp(-y^2) + 12\,(D(x))^2\,x^2\exp(-y^2) + 8\,D(x)\,x^3 D\left(\exp(-y^2)\right) + x^4\left(D^{(2)}\right)\left(\exp(-y^2)\right)$$

ここで，D(x) などは x の全微分を表わす．これは，x,y を変数としているので

```
> diff(x,x);
> diff(exp(-y^2),y);
```

$$1$$

$$-2\,y\exp(-y^2)$$

であるが Maple には分からない．そこで全微分の最終形を得るには，あらかじめ D(x) などの結果を求めておき，subs で明示的に代入する必要がある．

```
> dd:=D(f(x,y)):
> eqs:={D(x)=diff(x,x),D(exp(-y^2))=diff(exp(-y^2),y)};
> subs(eqs,dd);
```

$$eqs := \left\{D(x) = 1, D\left(\exp(-y^2)\right) = -2\,y\exp(-y^2)\right\}$$

$$4\,x^3\exp(-y^2) - 2\,x^4 y\exp(-y^2)$$

2.2.3　複合関数の微分

```
> diff(f(x)*g(x),x);
> diff(f(g(x)),x);
```

$$\left(\frac{d}{dx}f\left(x\right)\right)g\left(x\right)+f\left(x\right)\frac{d}{dx}g\left(x\right)$$

$$\mathrm{D}\left(f\right)\left(g\left(x\right)\right)\frac{d}{dx}g\left(x\right)$$

```
> f:=x->exp(x);
> g:=x->cos(x);
> diff(f(x)*g(x),x);
> diff(f(g(x)),x);
```

$$f := x \mapsto \exp(x)$$

$$g := x \mapsto \cos\left(x\right)$$

$$\exp(x)\cos\left(x\right)-\exp(x)\sin\left(x\right)$$

$$-\sin\left(x\right)\exp(\cos x)$$

[[[課題]]]

微分に関する課題

1. 次の関数を微分せよ．
   i)$x\log x$, ii) $\dfrac{1}{\left(1+x\right)^{3}}$, iii) $\sqrt{4\,x+3}$, iv) $\dfrac{1}{a^{2}+\left(x-x_{0}\right)^{2}}$
2. 次の関数の 1 次から 5 次導関数を求めよ．
   i) $\sin^{2}x$, ii) e^{x}
3. 以下の関数を x0 まわりで 3 次までテイラー展開し，得られた関数ともとの関数をプロットせよ．さらに 5 次まで展開した場合はどう変化するか．
   i) $y=\sin x, x_{0}=0$, ii) $y=\cos x, x_{0}=\dfrac{\pi}{2}$
4. (発展課題) $f\left(x,y\right)=e^{x}log\left(1+y\right)$ を $x=0$, $y=0$ のまわりで 3 次まで展開せよ．

[[[解答例]]]

Diff

1.
```
> diff(x*log(x),x);
> diff(1/(1 + x)^3,x);
> diff(sqrt(4*x + 3),x);
>diff(1/(a^2+(x-x0)^2),x);
```

$$\ln\left(x\right)+1$$

$$-3\,\left(1+x\right)^{-4}$$

$$2\,\left(\sqrt{4\,x+3}\right)^{-1}$$

$$-\frac{2\,x-2\,x0}{\left(a^{2}+\left(x-x0\right)^{2}\right)^{2}}$$

2.
```
> diff(sin(x)^2,x);
> diff(sin(x)^2,x$2);
```

$$2\sin(x)\cos(x)$$
$$2(\cos(x))^2 - 2(\sin(x))^2$$

以下略

3. 先ず与関数を f0 と定義

```
> f0:=unapply(sin(x),x);
```

$$f0 := x \mapsto \sin(x)$$

テイラー展開した結果を eq1 とする．関数として定義するために eq1 を多項式に変換し (convert)，unapply をかける．

```
> eq1:=series(f0(x),x=0,3);
> f1:=unapply(convert(eq1,polynom),x);
```

$$eq1 := x + O(x^3)$$
$$f1 := x \mapsto x$$

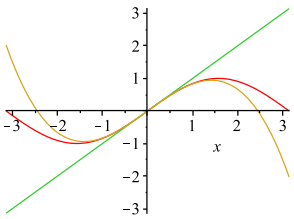
5 次についても同様

```
> eq2:=series(f0(x),x=0,5);
> f2:=unapply(convert(eq2,polynom),x);
```

$$eq2 := x - 1/6\,x^3 + O(x^5)$$
$$f2 := x \mapsto x - 1/6\,x^3$$

3 つの関数を同時プロット

```
> plot([f0(x),f1(x),f2(x)],x=-Pi..Pi);
```




```
> series(f0(x),x=Pi/2,3)
```

以外は前問とおなじ．

4. `f:=unapply(exp(x)*log(1+y),(x,y));`

$$f := (x, y) \mapsto \exp(x) \ln(1 + y)$$

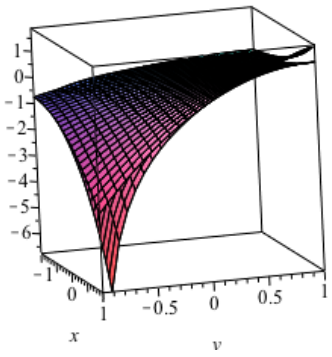
`eq1:=series(series(f(x,y),x=0,3),y=0,3);`

$$eq1 := O\left(x^3\right) + \left(1 + 1/2\,x^2 + x\right) y + \left(-1/2\,x - 1/2 - 1/4\,x^2\right) y^2 + O\left(y^3\right)$$

```
> g:=unapply(convert(convert(eq1,polynom),polynom),(x,y));
```

$$g := (x, y) \mapsto \left(1 + 1/2\,x^2 + x\right) y + \left(-1/2\,x - 1/2 - 1/4\,x^2\right) y^2$$

```
> plot3d([f(x,y),g(x,y)],x=-1..1,y=-1..1,axes=box);
```



きれいな表示

以下のようにすると表示がきれい.

```
> f:=unapply(x^4*exp(-y^2),(x,y));
> d:=Diff(f(x,y),x);
> d=value(d);
```

$$f := (x, y) \mapsto x^4 \exp(-y^2)$$

$$d := \frac{\partial}{\partial x}\left(x^4 \exp(-y^2)\right)$$

$$\frac{\partial}{\partial x}\left(x^4 \exp(-y^2)\right) = 4\,x^3 \exp(-y^2)$$

2.3 積分 (int)

[[[解説]]]

2.3.1 単純な積分 (int)

単純な不定積分.

```
> int(ln(x),x); #res: x ln(x) - x
```

定積分を実行するには，積分変数の範囲を指定する.

```
> int(sin(x),x=-Pi..0); #res: -2
```

特異点をもつ場合にも適切に積分結果を求めてくれる.

```
int(1/sqrt(x*(2-x)),x=0..2); #res: pi
```

無限区間 (infinity) における定積分も同様に計算してくれる.

```
> int(1/(x^2+4),x=-infinity..infinity); #res: 1/2 pi
```

部分積分法や置換積分法を用いる必要のある複雑な積分も一発で求まる.

```
> eq:=sqrt(4-x^2);int(eq,x);
```

$$eq := \sqrt{4-x^2}$$

$$\frac{1}{2}\,x\sqrt{4-x^2}+2\,\arcsin\left(1/2\,x\right)$$

数学の公式集に載っているような積分も同じコマンドで求まる.

```
eq2:=exp(-x^2);int(eq2,x=0..zz);
```

$$eq2 := \exp(-x^2)$$

$$\frac{1}{2}\,\sqrt{\pi}\,\mathrm{erf}\left(zz\right)$$

2.3.2 student パッケージいろいろ

ちょっとぐらい難しい積分も，Maple は単純に int コマンドだけで実行してくれる．しかし，時には，途中の計算法である部分積分，置換積分，部分分数展開が必要になる．このような計算は student パッケージに用意されている.

```
> with(student):
```

部分積分 (integration by parts)

```
> intparts(Int(x*exp(x),x),x);
```

$$x\exp(x)-\int \exp(x)dx$$

置換 (change of variables) による積分

```
> Int((cos(x)+1)^3*sin(x), x);
> changevar(cos(x)+1=u, Int((cos(x)+1)^3*sin(x), x=a..b), u);
> changevar(cos(x)+1=u, int((cos(x)+1)^3*sin(x), x), u);
```

$$\int (\cos(x)+1)^3 \sin(x)\, dx$$

$$\int_{\cos(a)+1}^{\cos(b)+1} -u^3 du$$

$$-\frac{1}{4} u^4 \tag{2.2}$$

部分分数 (partial fraction) 展開による積分　部分分数 (partial fraction) 展開による積分では，convert コマンドを用いる．

```
> pf1:=convert(1/(1+x^3),parfrac,x);
  int(pf1,x);
```

$$pf1 := \frac{1}{3}\,\frac{-x+2}{x^2-x+1} + \frac{1}{3\,(x+1)}$$

$$-\frac{1}{6}\ln\left(x^2-x+1\right) + \frac{1}{3}\sqrt{3}\arctan\left(1/3\,(2\,x-1)\sqrt{3}\right) + \frac{1}{3}\ln(x+1)$$

[[[課題]]]

1. 不定積分:次の不定積分を求めよ．
   i) $\int 4\,x+3dx$ ， ii)$\int \frac{1}{1+\mathrm{e}^x}dx$ ， iii)$\int \frac{1}{\mathrm{e}^{-x}+\mathrm{e}^x}dx$ ， iv)$\int \sqrt{1-x^2}dx$
2. 定積分:次の定積分を求めよ．
   i)$\int_0^{\pi} \sin x dx$ ， ii)$\int_0^1 \arctan x dx$ ， iii)$\int_{-2}^{2} \frac{1}{\sqrt{4-x^2}}dx$ ， iv)$\int_0^1 \frac{1}{x^2+x+1}dx$
3. (発展課題，重積分）次の 2 重積分を求めよ．

$$\int\int_D \sqrt{x^2+y^2}dxdy \; D: 0 \le y \le x \le 1$$

[[[解答例]]]

1.
```
> int(4*x+3,x);
> int( 1/(1+exp(x)),x);
> int(1/(exp(-x)+exp(x)),x);
> int(sqrt(1-x^2),x);
```

$$2\,x^2+3\,x$$

$$-\ln(1+\mathrm{e}^x)+\ln(\mathrm{e}^x)$$

$$\arctan(\mathrm{e}^x)$$

$$\frac{1}{2}\,x\sqrt{1-x^2}+\frac{1}{2}\arcsin(x)$$

2.
```
> int(sin(x),x=0..Pi);
> int(arctan(x),x=0..1);
> int(1/(sqrt(4-x^(2))),x=-2..2);
> int(1/(x^2+x+1),x=0..1);
```

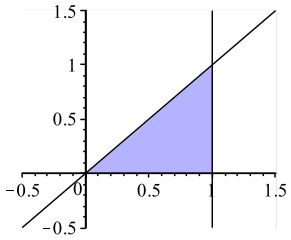
$$2$$

$$\frac{1}{4}\pi - \frac{1}{2}\ln(2)$$

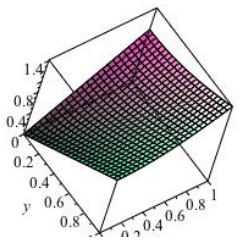
$$\pi$$

$$\frac{1}{9}\pi\sqrt{3}$$

3.
```
> with(plots):
> inequal({x-y>=0,x>=0,x<=1,y>=0},x=-0.5..1.5,y=-0.5..1.5,optionsexcluded=(color=white));
```




```
> f:=unapply(sqrt(x^2+y^2),(x,y)):
> plot3d(f(x,y),x=0..1,axes=box);
```




```
> int(int(f(x,y),y=0..x),x=0..1);
```

$$\frac{1}{6}\sqrt{2} + \frac{1}{6}\ln\left(1+\sqrt{2}\right)$$

第3章　線形代数

3.1　行列・ベクトル生成 (MatrixVector)

[[[解説]]]

線形代数の計算にはあらかじめ関数パッケージ (LinearAlgebra) を呼び出しておく.

```
> with(LinearAlgebra):
```

3.1.1　ベクトルの生成 (Vector)

ベクトルの生成は,

```
> v1 := Vector([x, y]);
```

$$v1 := \begin{bmatrix} x \\ y \end{bmatrix}$$

通常の方法では，縦（列）ベクトル (column) ができることに注意．横（行）ベクトル (row) を作るには，明示する必要あり.

```
> v2 := (Vector[row])([x, y, z]);
```

$$v2 := \begin{bmatrix} x & y & z \end{bmatrix}$$

新聞の囲み記事 (列）が column，劇場の座席（行）は row.

3.1.2　行列の生成 (Matrix)

標準的な行列 (Matrix) の生成は,

```
> A0 := Matrix([[1, 2, 3], [4, 5, 6]]); #res: 省略
```

リストリストからの変換は,

```
> LL1 := [[1, 2], [3, 4]]:
> A1 := Matrix(LL1);
```

$$A1 := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

単位行列の生成は,

```
> E := IdentityMatrix(2);
```

$$E := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

対角行列を生成する DiagonalMatrix もある．同じことは以下のようにしても生成が可能.

```
> Matrix(2,2,shape=identity);
```

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

3.1.3　縦横ベクトル・行列の簡易作成法 (MVShortcut)

かぎかっこ (< · · · >) を使って，ベクトルあるいは行列を直感的に作ることが可能．カンマで区切ると縦に積み，縦棒で区切ると横に積む．セミコロンで区切るとそこで次の行へ．

```
> v1:=<x,y>; #縦ベクトル，列
> v2:=<x|y|z>; #横ベクトル，行
> A1:=<1,2;3,4>; #2x2 行列
> <A1|v1>; #2x3 行列 (拡大係数行列などの作成)
> ?MVShortcut; #res:参照
```

$$v1 := \begin{bmatrix} x \\ y \end{bmatrix}$$

$$v2 := \begin{bmatrix} x & y & z \end{bmatrix}$$

$$A1 := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 & x \\ 3 & 4 & y \end{bmatrix}$$

3.1.4　行列，ベクトルの成分の抽出 (MVextraction)

行列 A1 の 1 行 2 列の成分を取り出すには，

```
> A1[1,2]; #res: 2
```

行列の一部を行列として取り出すには

```
> A1[1..2,1..2];
```

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

2x2 行列の 2 列目 (行の長さに関係なく) でつくるベクトルは

```
> A1[..,2..2];
```

$$\begin{bmatrix} 2 \\ 4 \end{bmatrix}$$

同じことが，行 (Row) あるいは列 (Column) 抽出関数でもできる．使い方は次の通り．

```
> Column(A1,2);
```

$$\begin{bmatrix} 2 \\ 4 \end{bmatrix}$$

[[[課題]]]

1. 次の行列，ベクトルを作れ．

(a) $\begin{bmatrix} 3 & 3 & 3 \\ 3 & 3 & 3 \end{bmatrix}$, (b) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$, (c) $\begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$, (d) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$, (e) $\begin{bmatrix} x & x^2 \\ x^2 & x^3 \end{bmatrix}$, (f) $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$

[[[解答例]]]

1. (a)
```
> Matrix([3,3,3],[3,3,3]]);
> Matrix(<3,3|3,3|3,3>);
```
$$\begin{bmatrix} 3 & 3 & 3 \\ 3 & 3 & 3 \end{bmatrix}$$

そのまま直打ちしてもいいが，少し賢い生成法も記しておく．詳しくはヘルプ参照．

```
> Matrix(2,3,3);
```
$$\begin{bmatrix} 3 & 3 & 3 \\ 3 & 3 & 3 \end{bmatrix}$$

(b)
```
> Matrix(2,3,shape=identity);
```
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

(c)
```
> Vector[row]([1,2,3]);
> Vector(<1|2|3>):
```
$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$

(d)
```
> with(LinearAlgebra):
> V:=Vector[row]([1,2,3]);
> DiagonalMatrix(V);
```
$$V := \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

(e)
```
> f:= (i,j) -> x^(i+j-1):
> Matrix(2,f);
```
$$\begin{bmatrix} x & x^2 \\ x^2 & x^3 \end{bmatrix}$$

(f)
```
> Matrix(3,[seq(i,i=1..9)]);
```
$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

```
> n:=3:
> f:=(i,j)->(i-1)*n+j;
> Matrix(3,3,f);
```
$$f := (i,j) \mapsto (i-1)\,n+j$$
$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

3.2　内積外積 (DotProduct,CrossProduct)

[[[解説]]]

線形代数の計算にはあらかじめ関数パッケージ (LinearAlgebra) を呼び出しておく.

```
> with(LinearAlgebra):
```

3.2.1　スカラーとのかけ算

```
> v1:=Vector([x, y]): 3*v1;
```

$$\begin{bmatrix} 3\,x \\ 3\,y \end{bmatrix}$$

3.2.2　行列, ベクトルの足し算，引き算

```
> LL1 := [[1, 2], [3, 4]]: A1 := Matrix(LL1): A2 := Matrix([[x, x], [y, y]]):
> 3*A1-4*A2;
```

$$\begin{bmatrix} 3-4\,x & 6-4\,x \\ 9-4\,y & 12-4\,y \end{bmatrix}$$

3.2.3　内積 (DotProduct, ‘.‘)

```
> v1:=Vector([1,1,3]): v2:=Vector([1,2,-1]): v1.v2;
```

$$0$$

3.2.4　外積 (CrossProduct, ‘&x‘)

```
> CrossProduct(v1, v2); v1 &x v2:
```

$$\begin{bmatrix} -7 \\ 4 \\ 1 \end{bmatrix}$$

3.2.5　スカラー 3 重積

```
> v3 := Vector([-1,2,1]); CrossProduct(v1,v2).v3;
```

$$v3 := \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix}$$

$$16$$

3.2.6 転置 (Transpose, '&T')

は，行列 A の ij 成分 a[i,j] を a[j,i] にする．

```
> Transpose(A1);
```

$$\begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}$$

また横ベクトルを縦ベクトル (あるいはその逆) にするのも同じ．

```
> Transpose(v1);
```

$$\begin{bmatrix} 1 & 1 & 3 \end{bmatrix}$$

[[[課題]]]

1. 行列 $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$, $B = \begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix}$，およびベクトル $v = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ を作り，以下の計算を行い結果を観察せよ．
   i) $A+3B$, ii) $A-B$, iii) $A+E$, iv) $A.B$, v) $B.A$, vi) $A.v$, vii) $v.A$, viii) v の転置 (Transpose) を A に左側から掛けよ, ix) A^3
2. 2 次元平面上で原点の周りの角度 t の回転行列は

   ```
   > Ar:=t->Matrix([[cos(t),-sin(t)],[sin(t),cos(t)]]);
   ```

   で定義できる．
   i) Pi/6 回転させる行列を作り，単位ベクトル (1,0),(0,1) がどの点に移動するか確認せよ．
   ii) Pi/6 回転させた後，続けて Pi/4 回転させる操作を続けて行う回転行列を求めよ．また，角度を直接入力して要素を比較せよ．
3. 行列 $A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$ について $A+A^t$, $A-A^t$ を求めて交代行列，対称行列を作れ．

[[[解答例]]]

1. ```
   > with(LinearAlgebra): A:=Matrix([[1,2],[3,4]]); B:=Matrix([[2,3],[4,5]]);
   > v:=Vector([1,2]); E:=IdentityMatrix(2);
   ```

$$A := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$B := \begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix}$$

$$v := \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$E := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

   i)–vi)
   ```

```
> A+3*B; A-B; A+E; A.B; B.A; A.v;
```

$$\begin{bmatrix} 7 & 11 \\ 15 & 19 \end{bmatrix}$$
$$\begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix}$$
$$\begin{bmatrix} 2 & 2 \\ 3 & 5 \end{bmatrix}$$
$$\begin{bmatrix} 10 & 13 \\ 22 & 29 \end{bmatrix}$$
$$\begin{bmatrix} 11 & 16 \\ 19 & 28 \end{bmatrix}$$
$$\begin{bmatrix} 5 \\ 11 \end{bmatrix}$$

vii)

```
> v.A;
```

```
Error, (in LinearAlgebra:-VectorMatrixMultiply) invalid input:
LinearAlgebra:-VectorMatrixMultiply expects its 1st argument, v, to be of type
Vector[row] but received Vector(2, {(1) = 1, (2) = 2})
```

v.A は次元が合わないので計算できない．次元を合わすためには，v に転置 (Transpose) をかけて横ベクトルにしておく必要がある．

viii)

```
> Transpose(v).A;
```

$$\begin{bmatrix} 7 & 10 \end{bmatrix}$$

ix)

```
> A^3;
```

$$\begin{bmatrix} 37 & 54 \\ 81 & 118 \end{bmatrix}$$

2. i)

```
> with(LinearAlgebra): e1:=Vector([1,0]); e2:=Vector([0,1]);
> Ar:=t->Matrix([[cos(t),-sin(t)],[sin(t),cos(t)]]); Ar(Pi/6).e1; Ar(Pi/6).e2;
```

$$e1 := \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$e2 := \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$Ar := t \mapsto \begin{bmatrix} \cos(t) & -\sin(t) \\ \sin(t) & \cos(t) \end{bmatrix}$$

$$\begin{bmatrix} 1/2\sqrt{3} \\ 1/2 \end{bmatrix}$$

$$\begin{bmatrix} -1/2 \\ 1/2\sqrt{3} \end{bmatrix}$$

ii.) 2 つの関数を別々に計算.

```
> Ar(Pi/4).Ar(Pi/6);
> Ar(Pi/6+Pi/4);
```

$$\begin{bmatrix} 1/4\sqrt{2}\sqrt{3} - 1/4\sqrt{2} & -1/4\sqrt{2} - 1/4\sqrt{2}\sqrt{3} \\ 1/4\sqrt{2}\sqrt{3} + 1/4\sqrt{2} & 1/4\sqrt{2}\sqrt{3} - 1/4\sqrt{2} \end{bmatrix}$$

$$\begin{bmatrix} \cos\left(\frac{5}{12}\pi\right) & -\sin\left(\frac{5}{12}\pi\right) \\ \sin\left(\frac{5}{12}\pi\right) & \cos\left(\frac{5}{12}\pi\right) \end{bmatrix}$$

2 つの操作の差の evalf をとるとほぼ 0, つまり一致していることが確認できる.

```
> evalf(Ar(Pi/6+Pi/4)-Ar(Pi/6).Ar(Pi/4));
```

$$\begin{bmatrix} -0.00000000020000000000 & 0.0 \\ 0.0 & -0.00000000020000000000 \end{bmatrix}$$

3. `> A:=Matrix([[1,2,3],[4,5,6],[7,8,9]]); As:=A+Transpose(A); Aa:=A-Transpose(A);`

$$A := \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

$$As := \begin{bmatrix} 2 & 6 & 10 \\ 6 & 10 & 14 \\ 10 & 14 & 18 \end{bmatrix}$$

$$Aa := \begin{bmatrix} 0 & -2 & -4 \\ 2 & 0 & -2 \\ 4 & 2 & 0 \end{bmatrix}$$

3.3　行列の基本操作，掃き出し (LUDecomposition)

[[[解説]]]

線形代数の計算にはあらかじめ関数パッケージ (LinearAlgebra) を呼び出しておく．

```
> with(LinearAlgebra):
```

3.3.1　行列の基本操作

行列の掃き出しに必要となる行列の基本操作は RowOperation, ColumnOperation を参照．

3.3.2　掃き出し法，LU 分解 (LUDecomposition)

掃き出し法の計算は，LUDecomposition でおこなう．まず拡大係数行列を作る．

```
> A1:=<1,2;3,4>; b:=<2,3>; <A1|b>;
```

$$A1 := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$b := \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 & 2 \\ 3 & 4 & 3 \end{bmatrix}$$

これに LU 分解をかける．それぞれ P(permutation，置換), L(lower triangle，下三角), U(upper triangle，上三角) 行列に代入している．

```
> P,L,U:=LUDecomposition(<A1|b>);
```

$$P,\ L,\ U := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 3 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 2 & 2 \\ 0 & -2 & -3 \end{bmatrix}$$

さらに被約階段行列 (row reduced echelonmatrix; 後退代入までおこなって，解まで求めた状態) を求めるには，output='R' を指定する．

```
> LUDecomposition(<A1|b>, output='R');
```

$$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 3/2 \end{bmatrix}$$

3.3.3　階数 (Rank)

行列の性質の中でも特に重要な階数 (Rank) は次のコマンドで求められる．

```
> Rank(A1);
```

$$2$$

[[[課題]]]

1. 行列 $A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$ について，RowOperation のヘルプを参照して，次の行基本操作をおこない，階数を求め，コマンド LUDecomposition, Rank の結果と比べよ．

   i) 2 行目から 1 行目の 4 倍を引く．

   ii) 3 行目から 1 行目の 7 倍を引く．

   iii) 2 行目を-1/3 倍する．

   iv) 3 行目に 2 行目の 6 倍を足す．

   RowOperation のコツは，最初は inplace=false でやってみて，うまくいけば true にかえる．

2. 次の連立方程式の解を掃き出し法で求めよ．GenerateMatrix を使えば連立方程式から拡大係数行列を直接生成することも可能．

   (i)
   $$\begin{cases} x + y - z &= 2 \\ 2x - 3y + z &= 4 \\ 4x - y + 3z &= 1 \end{cases}$$

   (ii)
   $$\begin{cases} 2x + 4y - 3z &= 1 \\ 3x - 8y + 6z &= 58 \\ x - 2y - 9z &= 23 \end{cases}$$

   (iii)
   $$\begin{cases} 1x - 10y - 3z - 7u &= 2 \\ 2x - 4y + 3z + 4u &= -3 \\ x - 2y + 6z + 5u &= -1 \\ x + 8y + 9z + 3u &= 5 \end{cases}$$

   (iv)
   $$\begin{cases} x + y + z &= a + b + c \\ ax + by + cz &= ab + bc + ca \\ bc\,x + ca\,y + ab\,z &= 3\,abc \end{cases}$$

3. 次の連立方程式
   $$\begin{cases} x_1 + 2x_2 - x_3 &= 0 \\ x_1 + x_2 + 3x_4 &= 0 \\ x_1 + 5x_2 - 2x_3 + 3x_4 &= 0 \\ x_1 + 3x_2 - 2x_3 - 3x_4 &= 0 \end{cases}$$
   を solve を使って，x_1, x_2, x_3, x_4 について解け．次に GenerateMatrix を使って，拡大係数行列にした後，LUDecomposition を用いて掃き出しを行い結果を比較せよ．

[[[解答例]]]

1.
```
> A:=Matrix([[1,2,3],[4,5,6],[7,8,9]]); with(LinearAlgebra): ?RowOperation;
```
$$A := \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

   ヘルプに書かれてある例を見本にして，コマンドを記述．

```
> RowOperation(A,[2,1],-4);
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \\ 7 & 8 & 9 \end{bmatrix}$$

結果を最初の引数 (A) に上書きする option(inplace=true) をつける．最初からではなく，うまくいったのを確認してからつけるのがコツ．

```
> RowOperation(A,[3,1],-7,inplace=true);
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \\ 0 & -6 & -12 \end{bmatrix}$$

```
> RowOperation(A,2,-1/3,inplace=true);
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 2 \\ 0 & -6 & -12 \end{bmatrix}$$

```
> RowOperation(A,[3,2],6);
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 0 \end{bmatrix}$$

LUDecomposition による結果と見比べる．

```
> A0:=Matrix([[1,2,3],[4,5,6],[7,8,9]]):
> LUDecomposition(A0);
```

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 4 & 1 & 0 \\ 7 & 2 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \\ 0 & 0 & 0 \end{bmatrix}$$

最後の行がすべて 0 になっているので，階数は 2 となる．Rank により確認．

```
> Rank(A);
```

$$2$$

2. i) GenerateMatrix による係数行列と右辺のベクトルを生成する方法は以下のとおり．

```
> eqs:={x+y-z=2,2*x-3*y+z=4,4*x-y+3*z=1}; GenerateMatrix(eqs,{x,y,z});
```

$$eqs := \{x + y - z = 2, 2\,x - 3\,y + z = 4, 4\,x - y + 3\,z = 1\}$$

$$\begin{bmatrix} 1 & 1 & -1 \\ 2 & -3 & 1 \\ 4 & -1 & 3 \end{bmatrix}, \begin{bmatrix} 2 \\ 4 \\ 1 \end{bmatrix}$$

i)

```
> A:= Matrix([[1,1,-1],[2,-3,1],[4,-1,3]]); b:=<2,4,1>;
> LUDecomposition(<A|b>,output='R');
```

$$A := \begin{bmatrix} 1 & 1 & -1 \\ 2 & -3 & 1 \\ 4 & -1 & 3 \end{bmatrix}$$

$$b := \begin{bmatrix} 2 \\ 4 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & \frac{13}{10} \\ 0 & 1 & 0 & -\frac{21}{20} \\ 0 & 0 & 1 & -7/4 \end{bmatrix}$$

ii)

```
> A:= Matrix([[2,4,-3],[3,-8,6],[8,-2,-9]]); b:=<1,5,-23>;
> LUDecomposition(<A|b>,output='R');
```

$$A := \begin{bmatrix} 2 & 4 & -3 \\ 3 & -8 & 6 \\ 8 & -2 & -9 \end{bmatrix}$$

$$b := \begin{bmatrix} 1 \\ 5 \\ -23 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

iii)

```
> A:= Matrix([[1,-10,-3,-7],[2,-4,3,4],[3,-2,6,5],[1,8,9,3]]); b:=<2,-3,-1,5>;
> LUDecomposition(<A|b>,output='R');
```

$$A := \begin{bmatrix} 1 & -10 & -3 & -7 \\ 2 & -4 & 3 & 4 \\ 3 & -2 & 6 & 5 \\ 1 & 8 & 9 & 3 \end{bmatrix}$$

$$b := \begin{bmatrix} 2 \\ -3 \\ -1 \\ 5 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1/2 \\ 0 & 0 & 1 & 0 & 1/3 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

iv)

```
> restart; with(LinearAlgebra): A:= Matrix([[1,1,1],[a,b,c],[b*c,c*a,a*b]]);
> bb:=<a+b+c,a*b+b*c+c*a,3*a*b*c>; RR:=LUDecomposition(<A|bb>,output='R');
```

$$A := \begin{bmatrix} 1 & 1 & 1 \\ a & b & c \\ bc & ca & ab \end{bmatrix}$$

$$bb := \begin{bmatrix} a+b+c \\ ab+bc+ca \\ 3\,abc \end{bmatrix}$$

$$RR := \begin{bmatrix} 1 & 0 & 0 & -\frac{a\left(b^2-2\,bc+c^2\right)}{(a-b)(a-c)} \\ 0 & 1 & 0 & \frac{\left(a^2-2\,ca+c^2\right)b}{(b-c)(a-b)} \\ 0 & 0 & 1 & -\frac{c\left(-2\,ab+b^2+a^2\right)}{ab-bc+c^2-ca} \end{bmatrix}$$

```
> factor(Column(RR,4)[1]);
```

などとすればさらに見やすく，変形される

$$-\frac{a\,(b-c)^2}{(a-b)\,(a-c)}$$

3.

```
> eqs:={x1+2*x2-x3=0,x1+x2+3*x4=0,3*x1+5*x2-2*x3+3*x4=0,x1+3*x2-2*x3-3*x4=0};
> solve(eqs,{x1,x2,x3,x4});
```

$$eqs := \{x1 + x2 + 3\,x4 = 0, x1 + 2\,x2 - x3 = 0,$$
$$x1 + 3\,x2 - 2\,x3 - 3\,x4 = 0, 3\,x1 + 5\,x2 - 2\,x3 + 3\,x4 = 0\}$$
$$\{x1 = -6\,x4 - x3, x2 = 3\,x4 + x3, x3 = x3, x4 = x4\}$$

```
> A1,b:=GenerateMatrix(eqs,[x1,x2,x3,x4]);
> LUDecomposition(<A1|b>,output='R');
```

$$A1,\, b := \begin{bmatrix} 1 & 1 & 0 & 3 \\ 1 & 2 & -1 & 0 \\ 1 & 3 & -2 & -3 \\ 3 & 5 & -2 & 3 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 1 & 6 & 0 \\ 0 & 1 & -1 & -3 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

3.4 逆行列 (MatrixInverse)

[[[解説]]]

線形代数の計算にはあらかじめ関数パッケージ (LinearAlgebra) を呼び出しておく.

```
> with(LinearAlgebra):
```

3.4.1 行列式 (Determinant)

```
> A0 := Matrix([[x,y],[z,u]]); Determinant(A0);
```

$$A0 := \begin{bmatrix} x & y \\ z & u \end{bmatrix}$$

$$xu - yz$$

3.4.2 逆行列 (MatrixInverse)

```
> A2:=MatrixInverse(A0); simplify(A0.A2);
```

$$A2 := \begin{bmatrix} \frac{u}{xu-yz} & -\frac{y}{xu-yz} \\ -\frac{z}{xu-yz} & \frac{x}{xu-yz} \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

3.4.3 その他の演算

随伴 (Adjoint) などもコマンドだけで求まる. 詳しくはヘルプ参照.

[[[課題]]]

1. 次の連立方程式の係数行列の行列式を求めよ.

(i)

$$\begin{cases} x + y - z &= 2 \\ 2x - 3y + z &= 4 \\ 4x - y + 3z &= 1 \end{cases}$$

(ii)

$$\begin{cases} 2x + 4y - 3z &= 1 \\ 3x - 8y + 6z &= 58 \\ x - 2y - 9z &= 23 \end{cases}$$

(iii)

$$\begin{cases} 1x - 10y - 3z - 7u &= 2 \\ 2x - 4y + 3z + 4u &= -3 \\ x - 2y + 6z + 5u &= -1 \\ x + 8y + 9z + 3u &= 5 \end{cases}$$

(iv)

$$\begin{cases} x+y+z & = a+b+c \\ ax+by+cz & = ab+bc+ca \\ bc\,x+ca\,y+ab\,z & = 3\,abc \end{cases}$$

2. 上の連立方程式の係数行列の逆行列を求めよ．またベクトル b に作用して解を求めよ．

[[[解答例]]]

1.
```
> with(LinearAlgebra): eqs:={x+y-z=2,2*x-3*y+z=4,4*x-y+3*z=1};
> A,b:=GenerateMatrix(eqs,{x,y,z}); Determinant(A);
```

$$eqs := \{x+y-z=2, 2\,x-3\,y+z=4, 4\,x-y+3\,z=1\}$$

$$A,\, b := \begin{bmatrix} 1 & 1 & -1 \\ 2 & -3 & 1 \\ 4 & -1 & 3 \end{bmatrix}, \begin{bmatrix} 2 \\ 4 \\ 1 \end{bmatrix}$$

$$-20$$

2.
```
> MatrixInverse(A); simplify(MatrixInverse(A).b);
```

$$\begin{bmatrix} 2/5 & 1/10 & 1/10 \\ 1/10 & -\frac{7}{20} & \frac{3}{20} \\ -1/2 & -1/4 & 1/4 \end{bmatrix}$$

$$\begin{bmatrix} \frac{13}{10} \\ -\frac{21}{20} \\ -7/4 \end{bmatrix}$$

3.5 固有値 (EigenVectors)

[[[解説]]]

線形代数の計算にはあらかじめ関数パッケージ (LinearAlgebra) を呼び出しておく.

```
> with(LinearAlgebra):
```

3.5.1 固有値 (EigenVectors)

固有値 (Eigenvalues) と固有ベクトルを共に求めるには Eigenvectors を使う. 下の例では, 固有値と固有ベクトルを変数 l,v に代入している.

```
> A0 := Matrix(2, 2, [[1,2], [2,1]]); l,v:=Eigenvectors(A0);
```

$$A0 := \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$$

$$l,\ v := \begin{bmatrix} -1 \\ 3 \end{bmatrix}, \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}$$

3.5.2 固有ベクトルの取り出し (Column)

行列の列を要素とするベクトル生成 Column を使って, 一番目の固有値に対応する固有ベクトルを取り出す.

```
> Column(v,1);
```

$$\begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

これを使って, 固有値 (l) と固有ベクトル (v) の関係

$$A_0.v = \lambda.v$$

が確認できる.

```
> A0.Column(v,1); l[1]*Column(v,1);
```

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

3.5.3 固有値ベクトルの規格化 (Normalize)

用意されているコマンドが確かめられる.

```
> ?Normalize;
```

一般的な内積を使って長さを規格化するには, 以下のコマンドを使う.

```
> Normalize(Column(v,1),Euclidean);
```

$$\begin{bmatrix} -1/2\sqrt{2} \\ 1/2\sqrt{2} \end{bmatrix}$$

3.5.4　対角化

規格化された固有値ベクトルを用いて，次のとおり行列は対角化される．

```
> Transpose(v).A0.v;
```

$$\begin{bmatrix} -2 & 0 \\ 0 & 6 \end{bmatrix}$$

3.5.5　その他の演算

対角和 (Trace)，ジョルダン標準形 (JordanForm) などもコマンドだけで求まる．詳しくはヘルプ参照．

[[[課題]]]

1. 行列

$$A = \begin{bmatrix} 1 & -2 & 1 \\ -1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$$

の固有値を固有方程式

$$|A - \lambda E| = 0$$

を解いて求めよ．EigenVectors を用いて固有値と固有ベクトルを求めよ．固有値，固有ベクトルの関係

$$A.v = \lambda v$$

を確認せよ．さらに，固有ベクトルを長さ 1 に規格化せよ．

2. 行列

$$A = \begin{bmatrix} 2 & 0 & 1 \\ 0 & 3 & 0 \\ 1 & 0 & 2 \end{bmatrix}$$

を対角化する変換行列 P を求め，対角化せよ．

[[[解答例]]]

1.
```
> A:=Matrix([[1,-2,1],[-1,2,1],[1,2,1]]); E:=Matrix(3,3,shape=identity):
> eq:=Determinant(A-x*E); solve(eq,x);
```

$$A := \begin{bmatrix} 1 & -2 & 1 \\ -1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$$

$$eq := 4x^2 - x^3 - 8$$

$$2, 1 + \sqrt{5}, 1 - \sqrt{5}$$

```
> l,v:=Eigenvectors(A); v1:=Column(v,3); evalf(A.v1); evalf(l[3].v1);
> Normalize(v1,Euclidean); evalf(Normalize(v1,Euclidean));
```

$$l,\ v\ :=\ \begin{bmatrix} \sqrt{5}+1 \\ 1-\sqrt{5} \\ 2 \end{bmatrix},\ \begin{bmatrix} \frac{(\sqrt{5}-3)\sqrt{5}}{-5+3\sqrt{5}} & -\frac{(-3-\sqrt{5})\sqrt{5}}{-5-3\sqrt{5}} & 1 \\ -\frac{-5+\sqrt{5}}{-5+3\sqrt{5}} & -\frac{-5-\sqrt{5}}{-5-3\sqrt{5}} & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$v1\ :=\ \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 2.0 \\ 0.0 \\ 2.0 \end{bmatrix}$$

$$\begin{bmatrix} 2.0 \\ 0.0 \\ 2.0 \end{bmatrix}$$

$$\begin{bmatrix} 1/2\sqrt{2} \\ 0 \\ 1/2\sqrt{2} \end{bmatrix}$$

$$\begin{bmatrix} 0.7071067810 \\ 0.0 \\ 0.7071067810 \end{bmatrix}$$

(3.1)

2.
```
> A:=Matrix([[2,0,1],[0,3,0],[1,0,2]]); l,v:=Eigenvectors(A);
> MatrixInverse(v).A.v;
```

$$A\ :=\ \begin{bmatrix} 2 & 0 & 1 \\ 0 & 3 & 0 \\ 1 & 0 & 2 \end{bmatrix}$$

$$l,\ v\ :=\ \begin{bmatrix} 3 \\ 3 \\ 1 \end{bmatrix},\ \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 3 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

第4章　式変形

4.1　センター試験I(CenterExamI)

[[[解説]]]

数式変形実践課題（大学入試センター試験の解法を通して）　今まで出てきたコマンドを使えば，典型的なセンター試験の問題を解くのも容易である．以下の例題を参照して課題を解いてみよ．使うコマンドは，unapply, solve, diff, expand(展開), factor(因数分解) と subs(一時的代入) である．expand 等の数式変形によく使うコマンドは次節以降で詳しく解説している．subs は以下を参考にせよ．

4.1.1　一時的代入 (subs)

代入 (:=) が永続的なのに対して，一時的な代入は subs で行う．

```
> subs(a=1,a+2); #res: 3
```

典型的な使い方は，solve で求めた解などを式 (equation) として代入しておいて，それを subs で一時的に当てはめる．

```
> eq1:=a=solve(a+b=0,a); subs(eq1,a+2);
```

$$a = -b$$
$$-b+2$$

[[[例題]]]

4.1.2　例題:2 次関数の頂点

a, b を定数とし, $a <> 0$ とする．2 次関数

$$y = a\,x^2 - b\,x - a + b \cdots (1)$$

のグラフが点 $(-2, 6)$ を通るとする. このとき

$$b = -a + \boxed{\text{ア}}$$

であり, グラフの頂点の座標を a を用いて表すと

$$\left(\frac{-a + \boxed{\text{イ}}}{\boxed{\text{ウ}}\,a}, -\frac{(\boxed{\text{エ}}\,a - \boxed{\text{オ}})^2}{\boxed{\text{カ}}\,a} \right)$$

である (2008 年度大学入試センター試験数学 I より抜粋).

解答例

まず，与えられた 2 次関数を $f(x)$ で定義する．

```
> restart; f:=unapply(a*x^2-b*x-a+b,x);
```

$$f := x \mapsto a\,x^2 - b\,x - a + b$$

与えられた点の座標を関数に入れる．

```
> eq1:=f(-2)=6;
```

$$eq1 := 3\,a + 3\,b = 6$$

これを b について解く．

```
> eq2:=b=solve(eq1,b);
```

$$eq2 := b = 2 - a$$

次は，頂点の座標で傾きが 0 になることを用いて解いていく．

```
> solve(diff(f(x),x)=0,x);
```

$$\frac{1}{2}\,\frac{b}{a}$$

b の値は eq2 で求まっているので，それを代入 (subs) する．

```
> subs(eq2,solve(diff(f(x),x)=0,x));
```

$$\frac{1}{2}\,\frac{2-a}{a}$$

これを x0 として eq3 で定義しておく．

```
> eq3:=x0=subs(eq2,solve(diff(f(x),x)=0,x));
```

$$eq3 := x0 = \frac{1}{2}\,\frac{2-a}{a}$$

頂点の y 座標は，$f(x0)$ で求まる

```
> f(x0);
```

$$a\,x0^2 - b\,x0 - a + b$$

eq2, eq3 で求まっている x0, b を代入する．

```
> eq4:=subs({eq2,eq3},f(x0));
```

$$eq4 := -\frac{1}{4}\,\frac{(2-a)^2}{a} - 2\,a + 2$$

これを因数分解 (factor) する．

```
> factor(subs({eq2,eq3},f(x0)));
```

$$-\frac{1}{4}\,\frac{(3\,a-2)^2}{a}$$

[[[課題]]]

1. $P = x(x+3)(2x-3)$ とする．また，a を定数とする．$x = a+1$ のときの P の値は

$$2a^3 + \boxed{\text{ア}}a^2 + \boxed{\text{イ}}a - \boxed{\text{ウ}}$$

である．

$x = a+1$ のときの P の値と，$x = a$ のときの P の値が等しいとする．このとき，a は

$$3a^2 + \boxed{\text{エ}}a - \boxed{\text{オ}} = 0$$

を満たす．したがって

$$a = \frac{\boxed{\text{カキ}} \pm \sqrt{\boxed{\text{クケ}}}}{\boxed{\text{コ}}}$$

である．

2. （例題 1．に引き続いて,）さらに，2 次関数 (1) のグラフの頂点の y 座標が-2 であるとする．このとき，a は

$$\boxed{\text{キ}}a^2 - \boxed{\text{クケ}}a + \boxed{\text{コ}} = 0$$

を満たす．これより，a の値は

$$a = \boxed{\text{サ}}, \frac{\boxed{\text{シ}}}{\boxed{\text{ス}}}$$

である．以下，$a = \dfrac{\boxed{\text{シ}}}{\boxed{\text{ス}}}$ であるとする．

このとき，2 次関数 (1) のグラフの頂点の x 座標は $\boxed{\text{セ}}$ であり，(1) のグラフと x 軸の 2 交点の x 座標は $\boxed{\text{ソ}}, \boxed{\text{タ}}$ である．

また，関数 (1) は $0 \leqq x \leqq 9$ において

$x = \boxed{\text{チ}}$ のとき，最小値 $\boxed{\text{ツテ}}$ をとり，

$x = \boxed{\text{ト}}$ のとき，最大値 $\dfrac{\boxed{\text{ナニ}}}{\boxed{\text{ヌ}}}$ をとる．

(2008 年度大学入試センター試験数学 I より抜粋).

[[[解答例]]]

1. P を x の関数として定義,

```
> restart:
> P:=unapply(x*(x+3)*(2*x-3),x);
```

$$P := x \mapsto x\,(x+3)\,(2\,x-3)$$

$P(a+1)$ および $P(a)$ を形式的に出してみる．

```
> expand(P(a+1)), expand(P(a));
```

$$2\,a^3 + 9\,a^2 + 3\,a - 4,\ 2\,a^3 + 3\,a^2 - 9\,a$$

2 式を差し引く．

```
> eq1:=(expand(P(a+1))-expand(P(a)))/2;
```

$$eq1 := 3\,a^2 + 6\,a - 2$$

出題にそろえるため 2 で割っている．その式を eq1 として代入し，eq1=0 を x について解く (solve).

```
> sol1:=solve(eq1=0,a);
```

$$sol1 := -1 + \frac{1}{3}\sqrt{15},\ -1 - \frac{1}{3}\sqrt{15}$$

2. 例題の eq3,eq4 までを確認

```
> eq3, eq4;
```

$$x0 = \frac{1}{2}\,\frac{2-a}{a},\ -\frac{1}{4}\,\frac{(2-a)^2}{a} - 2\,a + 2$$

eq4 が頂点の y 座標の値なので，これから -2 を引いて展開.

```
expand((eq4-(-2)));
```

$$-\frac{1}{a} + 5 - \frac{9}{4}\,a$$

これではわかりにくいので，出題にそう形にするため，$-4a$ を掛け，eq5 とする.

```
> eq5:=expand((eq4+2)*(-4)*a);
```

$$eq5 := 4 - 20\,a + 9\,a^2$$

これを a について解いて (solve)

```
> solve(eq5=0,a);
```

$$2,\ \frac{2}{9}$$

$a = 2/9$ を eq3 に代入して，頂点の x 座標を出す.

```
> subs(a=2/9,eq3);
```

$$x0 = 4$$

a, b を $f(x)$ に代入して，

```
> eq6:=subs({a=2/9,b=2-2/9},f(x));
```

$$eq6 := \frac{2}{9}\,x^2 - \frac{16}{9}\,x + \frac{14}{9}$$

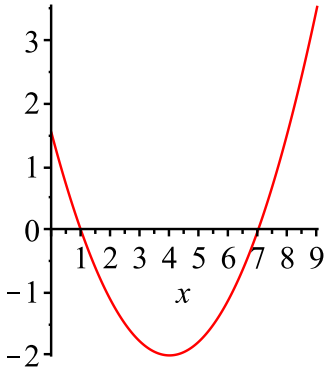
これを解いて，x 座標の交点を求める.

```
> solve(eq6=0,x);
```

$$7,\ 1$$

最大値，最小値を求めるために，今まで求めたパラメータを代入して，plot してみる.

```
> plot(subs({a=2/9,b=2-2/9},f(x)),x=0..9);
```



目視で分かるとおり，最小値 $x=4$, 最大値 $x=9$ で

```
> subs({a=2/9,b=2-2/9},f(4)), subs({a=2/9,b=2-2/9},f(9));
```

$$-2,\ \frac{32}{9}$$

4.2　数式処理コマンドの分類 (EquationManipulationTable)

[[[解説]]]

数式の変形は，手で直すほうが圧倒的に早くきれいになる場合が多い．しかし，テイラー展開や，複雑な積分公式，三角関数と exp 関数の変換などの手間がかかるところを，Maple は間違いなく変形してくれる．ここで示すコマンドを全て覚える必要は全くない．というか忘れるもの．ここでは，できるだけコンパクトにまとめて，悩んだときに参照できるようにする．初めての人は，ざっと眺めた後，鉄則からじっくりフォローせよ．

表 4.1: 数式処理で頻繁に使うコマンド．

式の変形	式の分割抽出	省略操作，その他	代入，置換，仮定
simplify:簡単化	lhs, rhs:左辺，右辺	|| :連結作用素	subs:一時的代入
expand:展開	numer, denom:分子，分母	seq:for-loop の簡易表記	restart,a:=’a’:初期化
factor:因数分解	coeff:係数	map:関数の要素への適用	assume:仮定
normal:約分・通分	nops, op	add,mul:単純な和，積	assuming:一時的仮定
combine:公式でまとめる		sum,product:数式に対応した和，積	assign:値の確定
collect:次数でまとめる		limit:極限	about:仮定の中身
sort:昇べき，降べき			anames(’user’):使用変数名
convert:形式の変換			

このほかにも，solve(解), diff(微分), int(積分)，series(級数展開) 等は頻繁に数式の導出・変形に登場する．

4.3 式の変形に関連したコマンド (Simplify)

[[[解説]]]

4.3.1 簡単化 (simplify)

```
> simplify(exp1, 副関係式):
> simplify(3*x+4*x+2*y);
```

$$7\,x + 2\,y$$

```
> exp1:=3*sin(x)^3-sin(x)*cos(x)^2;
> simplify(exp1);
```

$$exp1 := 3\,(\sin(x))^3 - \sin(x)\,(\cos(x))^2 - \left(4\,(\cos(x))^2 - 3\right)\sin(x)$$

```
> simplify(exp1,{cos(x)^2=1-sin(x)^2});
```

$$4\,(\sin(x))^3 - \sin(x)$$

オプションとして size を指定するとより簡単になる場合がある.

```
> simplify(exp1,size):
```

4.3.2 展開 (expand)

```
> expand((x+y)^2);
```

$$x^2 + 2\,xy + y^2$$

4.3.3 因数分解 (factor)

```
> factor(4*x^2-6*x*y+2*y^2);
```

$$2\,(2\,x - y)\,(x - y)$$

4.3.4　約分・通分 (normal)

```
> normal((x+y)/(x^2-3*x*y-4*y^2));
```

$$\frac{1}{x-4\,y}$$

```
> normal(1/x+1/y);
```

$$\frac{y+x}{xy}$$

4.3.5　項を変数でまとめる (collect)

```
> collect(4*a*x^2-3*y^2/x+6*b*x*y+3*c*y+2*y^2,y);
```

$$\left(-3\,x^{-1}+2\right)y^2+\left(6\,bx+3\,c\right)y+4\,ax^2$$

4.3.6　項を公式でまとめる (combine)

```
> combine(sin(x)^2+3*cos(x)^2);
```

$$2+\cos\left(2\,x\right)$$

4.3.7　昇べき，降べき (sort)

```
> sort(exp1,[x,y]);
> sort(exp1, [x],opts);
opts=tdeg(総次数順), plex(辞書式順), ascending(昇順), descending(降順)

> exp1:=x^3+4*x-3*x^2+1:
> sort(exp1);
```

$$x^3-3\,x^2+4\,x+1$$

```
> sort(exp1,[x],ascending);
```

$$1+4\,x-3\,x^2+x^3$$

```
> exp2:=x^3-a*x*y+4*x^2+y^2:
> sort(exp2);
```

$$-axy + x^3 + 4\,x^2 + y^2$$

```
> sort(exp2,[x]);
```

$$x^3 + 4\,x^2 - ayx + y^2$$

```
> sort(exp2,[a,y]);
> sort(exp2,[a],plex);
```

$$-xay + y^2 + x^3 + 4\,x^2$$
$$-xya + x^3 + 4\,x^2 + y^2$$

[[[課題]]]

1. 以下の式を簡単化せよ.

i) $x^{100} - 1$, ii) $x^2 - y^2 + 2x + 1$, iii) $(a+b+c)^3 - (a^3 + b^3 + c^3)$

[[[解答例]]]

```
> factor(x^100-1);
```

$$(x-1)\left(1+x^4+x^3+x^2+x\right)\left(1+x^{20}+x^{15}+x^{10}+x^5\right)$$
$$(1+x)\left(1-x+x^2-x^3+x^4\right)\left(1-x^5+x^{10}-x^{15}+x^{20}\right)$$
$$\left(1+x^2\right)\left(x^8-x^6+x^4-x^2+1\right)\left(x^{40}-x^{30}+x^{20}-x^{10}+1\right)$$

```
> factor( x^2-y^2+2*x+1);
```

$$(x+1+y)\,(x+1-y)$$

```
> factor((a+b+c)^3-(a^3+b^3+c^3));
```

$$3\,(b+c)\,(c+a)\,(a+b)$$

4.4　式の分割抽出 (convert)

[[[解説]]]

4.4.1　形式の変換 (convert(exp1,opt))

数式の記述形式を変えるのに頻繁に使う.

opt	意味
polynom	級数を多項式 (polynomial) に
trig	三角関数 (trigonal) に
sincos	tan を含まない，sin,cos に
exp	指数関数形式に
parfrac	部分分数 (partial fraction) に
rational	浮動小数点数を有理数形式に

```
> s1:=series(sin(x),x,4);
> convert(s1,polynom);
```

$$s1 := x - \frac{1}{6}x^3 + O\left(x^4\right)$$

$$x - \frac{1}{6}x^3$$

```
> convert(sin(x),exp);
```

$$-\frac{1}{2}I\left(\exp(Ix) - \exp(-Ix)\right)$$

```
> convert(sinh(x),exp);
```

$$\frac{1}{2}\exp(x) - \frac{1}{2}\exp(x)$$

```
> convert(tan(x),sincos);
```

$$\frac{\sin(x)}{\cos(x)}$$

```
> convert(exp(I*x),trig);
```

$$\cos(x) + I\sin(x)$$

```
> convert(1/(x-1)/(x+3),parfrac);
```

$$-\frac{1}{4(x+3)} + \frac{1}{4(x-1)}$$

```
> convert(3.14,rational);
```

$$\frac{157}{50}$$

4.4.2 左辺，右辺 (lhs, rhs)

それぞれ，左辺，右辺を意味する left hand side, right hand side の略

```
> lhs(sin(x)^2=1-1/x);
> rhs(sin(x)^2=1-1/x);
```

$$\sin(x)^2$$
$$1-\frac{1}{x}$$

4.4.3 分母，分子 (denom,numer)

それぞれ，分母，分子を意味する denominator, numerator の略

```
> numer(a*x/(x+y)^3);
> denom(a*x/(x+y)^3);
```

$$ax$$
$$(x+y)^3$$

4.4.4 係数 (coeff)

```
> coeff(4*a*x^2-3*y^2/x+6*b*x*y+3*c*y+2*y^2,y^2);
```

$$-\frac{3}{x}+2$$

4.4.5 要素の取りだし，要素数 (op, nops)

op, nops は list 配列から要素や要素数を取り出すのに頻繁に使われる．数式を含めた，より一般的な構造に対しても作用させることができる．

```
> op(4*a*x^2-3*y^2/x+6*b*x*y+3*c*y+2*y^2);
```

$$4\,ax^2,\ -\frac{3\,y^2}{x},\ 6\,bxy,\ 3\,cy,\ 2\,y^2$$

```
> nops(4*a*x^2-3*y^2/x+6*b*x*y+3*c*y+2*y^2); #res: 5
```

[[[課題]]]

1. 以下の関数を x0 まわりで 3 次までテイラー展開し，得られた関数ともとの関数をプロットせよ．さらに高次まで展開した場合はどう変化するか．

   i) $y=\cos(x), x0=0$, ii) $y=\ln(x), x0=1$, iii) $y=\exp(-x), x0=0$

2. $\dfrac{x+1}{(x-1)(x^2+1)^2}$ を部分分数に展開せよ．
3. $\dfrac{1}{1-x^4} = \dfrac{a}{x^2+1} + \dfrac{b}{x+1} + \dfrac{c}{x-1}$ が常に成立する a, b, c を定めよ．
4. $\dfrac{8}{3-\sqrt{5}} - \dfrac{2}{2+\sqrt{5}}$ を簡単化せよ．
5. $x^2 + 2kx + 5 - k = 0$ が重根をもつように k を定めよ．

[[[解答例]]]

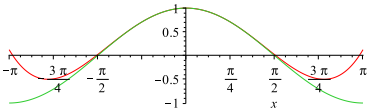
1. series で Taylor 展開した後，convert で多項式 (polynom) に変換する．

```
> convert(series(cos(x),x),polynom);
```

$$1 - \frac{1}{2}x^2 + \frac{1}{24}x^4$$

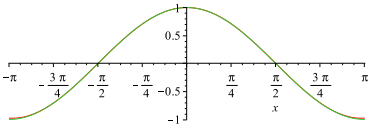
```
> plot([convert(series(cos(x),x),polynom),cos(x)],x=-Pi..Pi);
```

$$1 - \frac{1}{2}x^2 + \frac{1}{24}x^4 - \frac{1}{720}x^6 + \frac{1}{40320}x^8$$



高次展開する場合には，series の最後の引数に次数を指定する．デフォルトは 6 次．関数の一致具合が向上していることに注目．

```
convert(series(cos(x),x,9),polynom);
plot([convert(series(cos(x),x,9),polynom),cos(x)],x=-Pi..Pi);
```

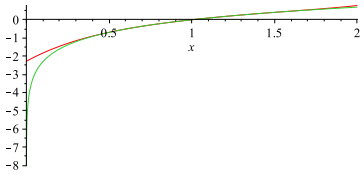


series での展開で，x=1 とするとその周りでの展開になる．

```
> convert(series(ln(x),x=1),polynom);
```

$$x - 1 - \frac{1}{2}(x-1)^2 + \frac{1}{3}(x-1)^3 - \frac{1}{4}(x-1)^4 + \frac{1}{5}(x-1)^5$$

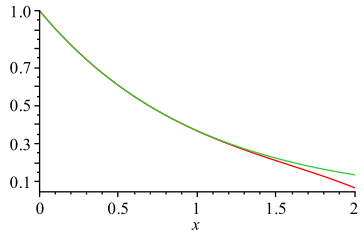
```
> plot([convert(series(ln(x),x=1),polynom),ln(x)],x=0..2);
```




```
> convert(series(exp(-x),x),polynom);
```

$$1 - x + \frac{1}{2}x^2 - \frac{1}{6}x^3 + \frac{1}{24}x^4 - \frac{1}{120}x^5$$

```
> plot([convert(series(exp(-x),x),polynom),exp(-x)],x=0..2);
```



2.
```
> restart; eq1:=(x+1)/((x-1)*(x^2+1)^2);
```

$$eq1 := \frac{x+1}{(x-1)(x^2+1)^2}$$

```
> convert(eq1,parfrac);
```

$$\frac{1}{2(x-1)} - \frac{x}{(x^2+1)^2} + \frac{1}{2}\frac{-x-1}{x^2+1}$$

3.
```
> convert(1/(1-x^4),parfrac);
```

$$-\frac{1}{4(x-1)} + \frac{1}{4(x+1)} + \frac{1}{2(x^2+1)}$$

4.
```
> eq2:=8/(3-sqrt(5))-2/(2+sqrt(5));
```

$$eq2 := \frac{8}{3-\sqrt{5}} - \frac{2}{2+\sqrt{5}}$$

まずはデフォルトの簡単化 (simplify).

```
> simplify(eq2); #res: 10
```

5. 方程式をたてる.

```
> eq3:=x^2+2*k*x+(5-k);
```

$$eq3 := x^2 + 2kx + 5 - k$$

単純に解を求めて,

```
> sol1:=solve(eq3=0,x);
```

$$sol1 := -k + \sqrt{k^2 - 5 + k}, -k - \sqrt{k^2 - 5 + k}$$

それが一致する場合の k を解く．

```
> solve(sol1[1]=sol1[2],k);
```

$$-\frac{1}{2} + \frac{1}{2}\sqrt{21}, -\frac{1}{2} - \frac{1}{2}\sqrt{21}$$

別解．まず，係数を coeff で取り出す．

```
> aa:=coeff(eq3,x^2); #res:1
> bb:=coeff(eq3,x);   #res: 2 k
> cc:=coeff(eq3,x,0); #res: 5 - k
```

判別式 $D = b^2 - 4ac$ を計算して，$= 0$ とおいて k について解く．

```
> solve(bb^2-4*aa*cc=0,k);
```

$$-\frac{1}{2} + \frac{1}{2}\sqrt{21}, -\frac{1}{2} - \frac{1}{2}\sqrt{21}$$

4.5 代入，置換，仮定に関連したコマンド (assume, subs)

[[[解説]]]

4.5.1 一時的代入 (subs)

関係式 (x=2) を式 (exp1) に一時的に代入した結果を表示.

```
> exp1:=x^2-4*x*y+4; subs(x=2,exp1);
```

$$exp1 := x^2 - 4\,xy + 4$$
$$8 - 8\,y$$

```
> subs({x=a+2,y=sin(x)},exp1);
```

$$(a+2)^2 - 4\,(a+2)\sin(x) + 4$$

4.5.2 仮定 (assume)

変数になにかの条件を仮定するときに使う．たとえば，根を開くときに，与式が正と仮定すると開かれる.

```
> sqrt(b^2);
> assume(a>0); sqrt(a^2);
```

$$\sqrt{b^2}$$
$$a \sim$$

4.5.3 一時的仮定 (assuming)

assume と同じだが，一時的に仮定するときに使われる.

```
> exp1:=x^2-4*x+4;
> sqrt(exp1);
```

$$exp1 := x^2 - 4\,x + 4$$
$$\sqrt{(-2+x)^2}$$

```
> sqrt(exp1) assuming x>2;
```

$$-2 + x$$

assume に加えての仮定に additionally がある.

4.5.4　solve で求めた値の確定 (assign)

解を求める solve をして

```
> x:='x':y:='y':
> s1:=solve({x-y+1=0,x+y-2=0},{x,y});
```

$$s1 := \left\{x = \frac{1}{2}, y = \frac{3}{2}\right\}$$

このまま，x,y の中身を見ても，

```
> x,y;
```

$$x, y$$

代入されていない．s1 を assign(確定) すると，

```
> assign(s1);
> x,y;
```

$$\frac{1}{2}, \frac{3}{2}$$

と代入される．一時的代入 subs と使い分ける．

4.5.5　assume で仮定した内容の確認 (about)

```
> about(a);
```

```
Originally a, renamed a~:
  is assumed to be: RealRange(Open(0),infinity)
```

4.5.6　ユーザが定義した変数の確認 (anames('user'))

```
> anames('user');
```

$$s1, y, x, a$$

4.5.7　値の初期化 (restart,a:='a')

4.5.8 連結作用素 (||)

前後の変数をくっつけて新たな変数とする.

```
> a||1; #res: a1
> a||b; #res: ab
```

プログラムの中で使うとより便利.

```
> for i from 1 to 3 do
    a||i:=i^2;
  end do;
```

$$a1 := 1$$
$$a2 := 4$$
$$a3 := 9$$

4.5.9 for-loop の簡略表記 (seq)

数列を意味する sequence の略.

```
> seq(k,k=4..7);
```

$$4,\ 5,\ 6,\ 7$$

4.5.10 リスト要素への関数の一括適用 (map)

```
> f:=x->exp(-x);
> map(f,[0,1,2,3]);
```

$$f := x \mapsto \exp(-x)$$
$$[1, \exp(-1), \exp(-2), \exp(-3)]$$

上記の 3 つを組み合わせると，効率的に式を扱うことができる.

```
> map(sin,[seq(theta||i,i=0..3)]);
```

$$[\sin(\theta 0), \sin(\theta 1), \sin(\theta 2), \sin(\theta 3)]$$

4.5.11 単純な和，積 (add,mul)

```
> add(x^i,i=0..3);
```

$$1 + x + x^2 + x^3$$

```
> mul(x^i,i=0..3);
```

$$x^6$$

4.5.12　数式にも対応した和，積 (sum,product)

```
> add(x^i,i=0..n);
```

```
Error, unable to execute add
```

```
> sum(x^i,i=0..n);
```

$$\frac{x^{n+1}}{x-1}-\frac{1}{x-1}$$

```
> product(x^i,i=0..n);
```

$$x^{\frac{1}{2}(n+1)^2-\frac{1}{2}n-\frac{1}{2}}$$

4.5.13　極限 (limit)

```
> limit(exp(-x),x=infinity); #res: 0
```

```
> limit(tan(x),x=Pi/2);
```

$$undefined$$

```
> limit(tan(x),x=Pi/2,left);
> limit(tan(x),x=Pi/2,complex);
```

$$\infty$$
$$-\infty+\infty\,I$$

4.6 式の変形の基本 (BottomLine)

[[[解説]]]

どうしても解かなければならない課題を前にコマンドリファレンスのあちこちを参照しながら解いていくのが数式処理を修得する最速法である．とびかかる前にちょっとした共通のコツがある．それをここでは示す．数式処理ソフトでの数式処理とは，数式処理ソフトが『自動的にやって』くれるのではなく，実際に紙と鉛筆で解いていく手順を数式処理ソフトに『やらせる』ことであることを肝に銘じよ．

4.6.1 鉄則

Maple をはじめとする数式処理ソフトの習得にあたって初心者がつまづく共通の過ちを回避する鉄則がある．

鉄則０：restart をかける 続けて入力すると前の入力が生きている．違う問題へ移るときや，もう一度入力をし直すときには，restart; を入力して初期状態からはじめる．入力した順番が狂っている場合もある．頭から順に return をやり直す．

鉄則１：出力してみる 多くのテキストではページ数の関係で出力を抑止しているが，初心者が問題を解いていく段階ではデータやグラフをできるだけ多く出力する．最後のコロンをセミコロンに変える，あるいは途中に print 文を入れる．

鉄則２：関数に値を代入してみる 数値が返ってくるべき時に変数があればどこかで入力をミスっている．plot で以下のようなエラーが出た場合にチェック．

```
> plot(f(x),x);

Warning, unable to evaluate the function to numeric values in the region; see
the plotting command's help page to ensure the calling sequence is correct
```

鉄則３：内側から順に入力する 長い入力や for-loop を頭から打ち込んではいけない !! 内側から順に何をしているか解読・確認しながら打ち込む．括弧が合わなかったり，読み飛ばしていたりというエラーが回避できる．

4.6.2 具体例：無限積分

以下に示す積分を実行せよ．

$$\int_{-\infty}^{\infty} x \exp(-\beta c\, x^2)\left(1+\beta g\, x^3\right) dx$$

最新版の Maple では改良が施されていて，このような複雑な積分も一発で求まる．

```
> f1:=unapply(x*exp(-beta*c*x^2)*(1+beta*g*x^3),x);
```

$$f1 := x \mapsto x \exp(-\beta c\, x^2)\left(1+\beta\, g x^3\right)$$

```
> int(f1(x),x=-infinity..infinity);
```

$$\begin{cases} \frac{3}{4}\,\frac{g\sqrt{\pi}}{\beta\, c^2\sqrt{c\beta}} & csgn\left(c\beta\right)=1 \\ \infty & otherwise \end{cases}$$

ここでは，$c\beta$ が正の場合 (csgn(beta c)=1) とそれ以外の場合 (otherwise) に分けて答えを返している．しかしこのような意図したきれいな結果をいつも Maple が返してくれるとは限らない．これだけしか知らないと，なにかうまくいかないときにお手上げになってしまう．このようなきれいで簡単な結果に行き着く前の，裏でおこなういくつかの予備計算を省略せずに示そう．

先ず鉄則 0 にしたがって restart をかけ，関数を定義する．

```
> restart; f1:=unapply(x*exp(-beta*c*x^2)*(1+beta*g*x^3),x);
```

$$f1 := x \mapsto x \exp(-\beta c x^2)\left(1+\beta g x^3\right)$$

次には鉄則 1 にしたがって積分する前にどのような関数かプロットしてみる．そのまま plot へ投げると怒られる．

```
> plot(f1(x),x=-10..10);
```

```
Warning, unable to evaluate the function to numeric values in the region; see
the plotting command's help page to ensure the calling sequence is correct
```

これは鉄則 2 にあるとおり，数値を代入すれば，

```
> f1(10);
```

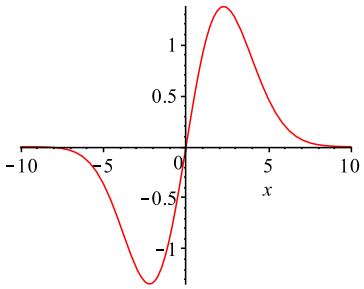
$$10\ \exp(-100\, c\beta)\,(1+1000\,\beta\, g)$$

beta,c,g などのパラメータの値が入っていないためとわかる．適当に値を代入する．

```
> c:=1; g:=0.01; beta:=0.1; #res:1 0.01 0.1
```

再度プロットを試みる．

```
> plot(f1(x),x=-10..10);
```



実際に積分してみる．ここでは，鉄則 3 にしたがって，式を頭から打ち込むのではなく内側からみていくことが肝要である．これは問題を解いていく時に，思考が必ずたどるであろう順番に相当する．先ず変数に入れた数値をクリアする．

```
> c:='c': g:='g':beta:='beta':
```

不定積分でこの関数が積分できることを確認する．

```
> int(f1(x),x);
```

$$-\frac{1}{2}\,\frac{1}{\exp(c\beta\, x^2)\beta\, c}+\beta\, g\left(-\frac{1}{2}\,\frac{x^3\exp(-c\beta\, x^2)}{c\beta}+\frac{3}{2}\left(-\frac{1}{2}\,\frac{x\exp(-c\beta\, x^2)}{c\beta}+\frac{1}{4}\,\frac{\sqrt{\pi}erf\left(\sqrt{c\beta}x\right)}{c\beta\sqrt{c\beta}}\right)c^{-1}\beta^{-1}\right)$$

次に x=-alpha..alpha の定積分を実行する．これは上記のコマンドに付け足すようにしていく．

```
> int(f1(x),x=-alpha..alpha);
```

$$-\frac{1}{4}\,\frac{g\left(4\,\alpha^3\exp(-c\beta\,\alpha^2)\beta\, c\sqrt{c\beta}+6\,\alpha\,\exp(-c\beta\,\alpha^2)\sqrt{c\beta}-3\,\sqrt{\pi}erf\left(\sqrt{c\beta}\alpha\right)\right)}{\beta\, c^2\sqrt{c\beta}}$$

さらに $\alpha \mapsto \infty$ としてみる．

```
> limit(int(f1(x),x=-alpha..alpha),alpha=infinity);
```

$$\lim_{\alpha\to\infty} -\frac{1}{4}\,\frac{g\left(4\,\alpha^3\exp(-c\beta\,\alpha^2)\beta\,c\sqrt{c\beta}+6\,\alpha\,\exp(-c\beta\,\alpha^2)\sqrt{c\beta}-3\,\sqrt{\pi}erf\left(\sqrt{c\beta}\alpha\right)\right)}{\beta\,c^2\sqrt{c\beta}}$$

ところがこれでは答えを返してくれない．積分した後のそれぞれの項を見ると $\beta\,c>0$ を仮定すれば簡単になることが分る．assume を使って，このような変数の仮定おこなう．

```
> assume(beta*c>0);
```

結果として最初に出した解答を得る．

```
> limit(int(f1(x),x=-alpha..alpha),alpha=infinity);
```

$$\frac{3}{4}\,\frac{\sqrt{\pi}g}{\beta\,c^2\sqrt{\beta\,c}}$$

4.6.3 式のフォローのデフォルト

Maple で実際に数式をいじる状況というのは，ほとんどの場合が既知の数式変形のフォローだろう．例えば，論文で「(1) 式から (2) 式への変形は自明である」とかいう文章で済ましている変形が本当にあっているのかを確かめたい時．一番単純なやり方は自明と言われた前後の式が一致していることを確かめるだけで十分である．最も単純な確認法は以下の通り，変形の前後の式を手入力してその差を expand した結果が 0 か否かでする．

```
> ex1:=(x-3)^4;
```

$$ex1 := (x-3)^4$$

```
> ex2:=x^4-12*x^3+54*x^2-108*x+81;
```

$$ex2 := x^4-12\,x^3+54\,x^2-108\,x+81$$

```
> expand(ex1-ex2); #res: 0
```

0 ならば式の変形は保証されているので，その導出が間違いでなく誤植などもないことが確認できる．ただ，これだけでは変形の哲学や技法が身に付くわけではない．あくまでも苦し紛れのデフォルトであることは心に留めておくように．論理値として確かめたいときには，evalb を使う．

```
> evalb(expand(ex1-ex2)=0); #res: true
```

第5章　描画

5.1　CG(ComputerGraphics)

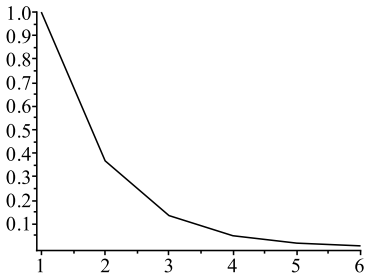
[[[解説]]]

5.1.1　listplot, pointplot

リスト構造にある離散的なデータは listplot で表示してくれる．listplot は受け取った list の要素を y 値に，1 から始まる添字を x 値にして，デフォルトでは線でグラフを書く．

```
> T:=[seq(exp(-i),i=0..5)];
> listplot(T);
```

$$T := [1, \exp(-1), \exp(-2), \exp(-3), \exp(-4), \exp(-5)]$$



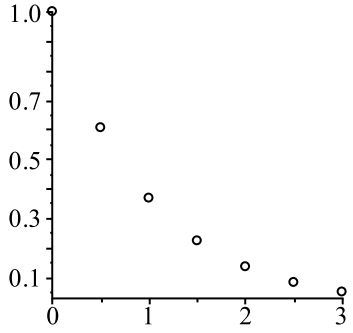
以下のように option をつけると point で描く．

```
> listplot(T,style=point):
```

それぞれの値の横軸 x が 1,2,3,.. では不都合なときには，2 次元の listlist 構造を用意し，[x[i],y[i]] を入れて pointplot 関数で表示する．

```
> T:=[seq([i/2,exp(-i/2)],i=0..6)];
> pointplot(T,symbol=circle,symbolsize=20);
```

$$T := [[0, 1], [1/2, \exp(-1/2)], [1, \exp(-1)], [3/2, \exp(-3/2)], [2, \exp(-2)], [5/2, \exp(-5/2)], [3, \exp(-3)]] \tag{5.1}$$



listplot のように線でつなぎたい時には，以下のように option をつける．

```
> pointplot(T,connect=true):
```

5.1.2　写像の表示

ある行列によって点を移動させる写像の様子を示すスクリプトを通して，plottools が提供する disk, arrow の使い方を示す．先ず描画に必要なライブラリーパッケージ (plots および plottools) を with で読み込んでおく．

```
> restart; with(plots):with(plottools):
```

行列 $A = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$ によって点 $a_0(1, 2)$ が $a_1(5, 4)$ に移動するとする (LinearAlgebra 参照)．

```
> with(LinearAlgebra): A:=Matrix([[1,2],[2,1]]): a0:=Vector([1,2]): a1:=A.a0;
```

$$a1 := \begin{bmatrix} 5 \\ 4 \end{bmatrix}$$

ベクトルが位置座標を意味するように list へ変換 (convert) する．

```
> p0:=convert(a0,list):p1:=convert(a1,list):
```

位置 p0 に円 (disk) を半径 0.2, 赤色で描く．同じように位置 p1 に半径 0.2，青色で disk を描く．

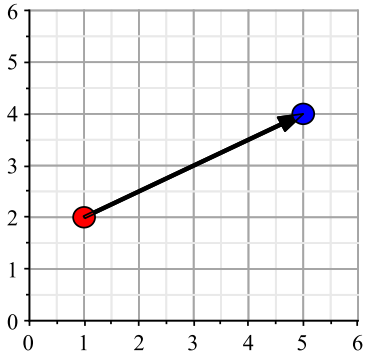
```
> point1:=[disk(p0,0.2,color=red),disk(p1,0.2,color=blue)]:
```

もう一つ，p0 から p1 に向かう矢印 (arrow) を適当な大きさで描く．後ろの数字をいじると線の幅や矢印の大きさが変わる．

```
> line1:=arrow(p0,p1,0.05,0.3,0.1):
```

これらをまとめて表示 (display)．このとき，表示範囲を 0..6,0..6 とする．

```
> display(point1,line1,view=[0..6,0..6],gridlines=true);
```



$a_0(1, 2)$ の赤点が，$a_1(5, 4)$ の青点へ移動していることを示している．

5.1.3 回転写像

次に原点周りでの回転の様子を示す．回転の行列．

```
> Matrix([[cos(theta),sin(theta)],[-sin(theta),cos(theta)]]);
```

$$\begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}$$

これを関数のように定義している．

```
> A:=t->Matrix([[cos(t),sin(t)],[-sin(t),cos(t)]]);
```

$$A := t \mapsto \begin{bmatrix} \cos(t) & \sin(t) \\ -\sin(t) & \cos(t) \end{bmatrix}$$

t に回転角 (Pi/3) を入れている．

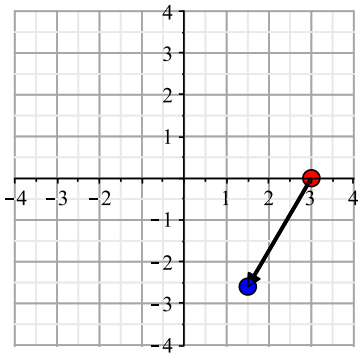
```
> a0:=Vector([3,0]);
> a1:=A(Pi/3).a0;
```

$$a0 := \begin{bmatrix} 3 \\ 0 \end{bmatrix}$$

$$a1 := \begin{bmatrix} 3/2 \\ -3/2\sqrt{3} \end{bmatrix}$$

表示の仕方は，前節と同じ．

```
> p0:=convert(a0,list):p1:=convert(a1,list):
> point1:=[disk(p0,0.2,color=red),disk(p1,0.2,color=blue)]:
> line1:=arrow(p0,p1,0.05,0.3,0.1):
> display(point1,line1,view=[-4..4,-4..4],gridlines=true);
```



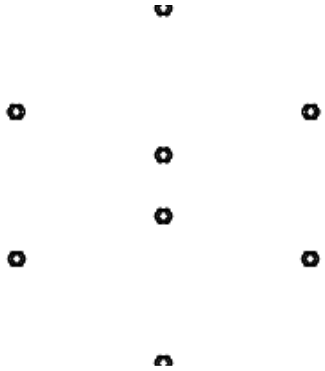
5.1.4 平行投影図の作成

もう少し複雑な対象物として，ここでは立方体の表示を考える．まず３次元座標を打ち込む．

```
> restart; with(plots): with(plottools):
  p:=[[0,0,0],[1,0,0],[1,1,0],[0,1,0],
  [0,0,1],[1,0,1],[1,1,1],[0,1,1]]:
```

次にこれを pointplot3d で簡便に表示．

```
> points:= { seq(p[i],i=1..8) }:
> pointplot3d(points,symbol=circle,symbolsize=40,color=black);
```

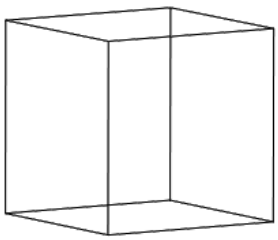


もうすこし見やすいように頂点を結んでおく．たとえば，p[1] と p[2] との間を線で結ぶには，

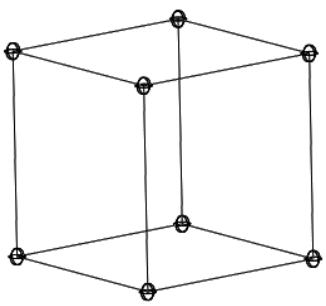
```
> line(p[1],p[2]);
```

とする．それを seq で複数の点間に対して施す．

```
> ll:=[[1,2],[2,3],[3,4],[4,1],[1,5],[2,6],[3,7],[4,8],
> [5,6],[6,7],[7,8],[8,5]]:
> lines:=[seq(line(p[ll[i][1]],p[ll[i][2]]),i=1..nops(ll))]:
> display(lines,scaling=constrained,color=black);
```



```
> l3:=display(lines,scaling=constrained,color=black):
> p3:=pointplot3d(p,symbol=circle,symbolsize=40,color=black):
> display([p3,l3],scaling=constrained,color=black);
```



画像をつまんでぐるぐる回してみよ．Maple ではこんな操作は簡単にできるが，よく見ればわかるように，この 3 次元表示では透視図ではなく，平行投影図といわれるものを書いている．

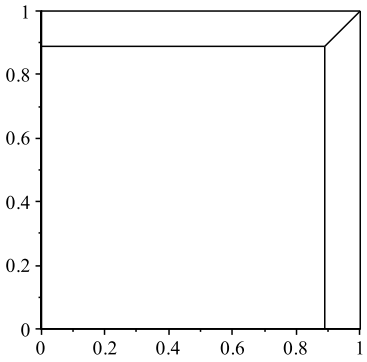
5.1.5 透視図

透視図のもっとも簡単な変換法は

```
> proj2d:=proc(x,z)
    local x1,y1;
    x1:=x[1]*z/(z-x[3]);
    y1:=x[2]*z/(z-x[3]);
    return [x1,y1];
  end proc:
```

z に視点の距離を入れて，x で座標を受け取って変換した結果を [x1,y1] として返している．この関数を前の表示と組み合わせれば透視図の描画ができる．

```
> z_p:=-8:
  lines:=[seq(line(proj2d(p[ll[i][1]],z_p), proj2d(p[ll[i][2]],z_p)), i=1..nops(ll))]:
  display(lines);
```



5.1.6 Maple の描画関数の覚書

maple にはいくつかの描画レベルに合わせた関数が用意されている．どのような目的にどの関数（パッケージ）を使うかの選択指針として，それぞれがどのような意図で作られ，それらの依存関係は以下の通り．

描画の下位関数 plot[structure] にある PLOT,PLOT3D データ構造が一番下で CURVES,POINTS,POLYGONS,TEXT データを元に絵を描く．

plottools パッケージ PLOT よりもう少し上位で，グラフィックスの基本形状を生成してくれる関数群．arc, arrow, circle, curve, line, point,sphere などの関数があり，PLOT 構造を吐く．表示には plots[display] を使う．

plots パッケージ 簡単にグラフを書くための道具．たとえば listplot は，list データを簡易に表示する事を目的としている．

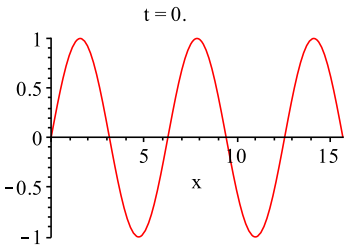
5.2　動画 (Animation)

[[[解説]]]

5.2.1　animate 関数

plots パッケージにある animate 関数を使う．構文は以下の通りで，[] 内に動画にしたい関数を定義し，t で時間を変えていく．

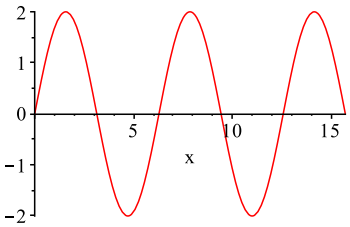
```
> with(plots): animate(plot, [sin(x-t),x=0..5*Pi], t=0..10);
```



5.2.2　リストに貯めて，display 表示

おなじ動作を，display 関数でオプションとして insequence=true としても可能．この場合は第一引数に入れるリスト ([]) に一連の画像を用意し，コマ送りで表示させる．

```
> tmp:=[]: n:=10: for i from 0 to n do t:=i; tmp:=[op(tmp),
> plot(sin(x-t)+sin(x+t),x=0..5*Pi)]; end do:
> display(tmp,insequence=true);
```



5.2.3　凝った例

ヘルプにある凝った例．F という動画のコマを吐く関数を用意する．これを，animate 関数から適当な変数を入れて呼び出す．background には動かない絵を指定することができる．

```
> with(plottools,line): F := proc(t) plots[display](
> line([-2,0],[cos(t)-2,sin(t)],color=blue),
> line([cos(t)-2,sin(t)],[t,sin(t)],color=blue),
> plot(sin(x),x=0..t,view=[-3..7,-5..5]) ); end:
> animate(F,[theta],theta=0..2*Pi, background=plot([cos(t)-2,sin(t),t=0..2*Pi]),
> scaling=constrained,axes=none);
```

5.2.4 ファイルへの保存

animation などの gif 形式の plot を外部ファイルへ出力して表示させるには，以下の一連のコマンドのようにする．

```
> plotsetup(gif,plotoutput=file2): display(tmp,insequence=true);
> plotsetup(default):
```

こいつを quicktime などに食わせれば，Maple 以外のソフトで動画表示が可能となる．3 次元図形の標準規格である vrml も同じようにして作成することが可能 (?vrml; 参照)．

第6章　Programming

6.1　代入と出力 (Variables, printf)

[[[解説]]]

6.1.1　値の変数への代入 (:=)

Maple は変数の初期設定で型宣言をする必要がない．数式処理の章で示したとおり，変数への代入は:=を使う．変数 a,b にそれぞれ 10,3 を代入し，a+b の結果を c に代入するというプログラムは以下の通り．

```
> a:=10: b:=3: c:=a+b;
```

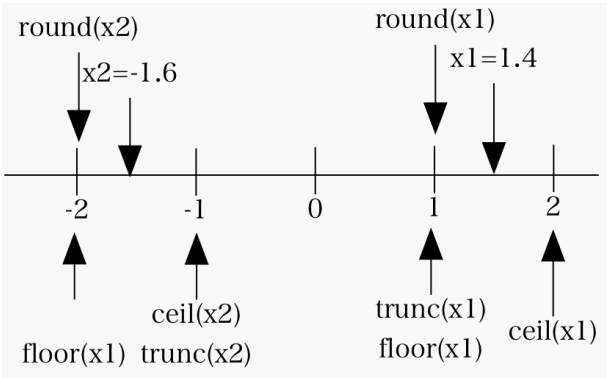
$$c := 13$$

6.1.2　整数と浮動小数点数

浮動小数点数から整数に直すにはいくつかの関数がある．

- trunc：数値から数直線で 0 に向って最も近い整数
- round：数値の四捨五入
- floor：数値より小さな最も大きな整数
- ceil：数値より大きな最も小さな整数

負の値の時に floor と trunc は違った値を返す．



小数点以下を取りだすには frac が用意されている．

```
> frac(1.7);
```

$$0.7$$

整数の割り算は irem(余り) と iquo(商).

```
> irem(7,3); #res: 1
> iquo(7,3); #res: 2
```

6.1.3 出力 (print, printf)

Maple ではデフォルトで結果が出力される．これを抑えるには行末の ”;” を ”:” に変える必要がある．出力を明示的におこなうには print を使う．デバッグの時に便利．

```
> x:=1: print(x); #res: 1
```

さらに，出力を整えるのに便利な printf 関数がある．これは C 言語と同じ構文で，

```
> printf("Hello world!!\n");


Hello world!!
```

と打ち込んで enter を押せば，出力が即座に表示される．値を表示するときには，

```
> i:=3: printf("%3d\n",i);

  3
```

となる．これは

「変数 i に入っている値を，3 桁の整数形式で打ち出した後，改行せよ」

と言う意味．%3d が出力の形式，\n が改行を意味する．OS によっては，\は ¥ と画面あるいはキーボードで表示されているかもしれない．実数の出力指定は%10.5f で，全部で 10 桁，小数点以下 5 桁で浮動小数点数を表示．複数の変数の出力は

```
> printf("%3d : %10.5f \n",i,a);
```

などとなる．

表 6.1: printf の書式指定

%指定	意味
%o	整数を 8 進数で表示.
%d	整数を 10 進数で表示.
%x,%f	浮動小数点数として表示.
%e,%s	文字列を出力.

6.2 ループ (Loop)

[[[解説]]]

6.2.1 for-loop

繰り返す操作は loop でおこなう．もっとも単純な for-loop.

```
> for i from 1 to 3 do
    i;
  end do;
```

```
1
2
3
```

初期値や増減を調整した for-loop

```
> for i from 10 by -2 to 0 do
    i;
  end do;
```

```
10
8
6
4
2
0
```

loop 回数が少ないときは，loop の中身も出力される．これを止めるには，end do; の最後のセミコロンをコロンに変える．

6.2.2 二重ループ

i,j という二つの変数を使って 2 重化したループ.

```
> for i from 1 to 3 do
    for j from 1 to 3 do
      print(i,j);
    end do;
  end do;
```

1, 1
1, 2
1, 3
2, 1
2, 2
2, 3
3, 1
3, 2
3, 3

while-loop も同じように使える.

```
> i:=0;
  while i<5 do
    i:=i+1;
  end do;
```

0
1
2
3
4
5

[[[課題]]]

1. printf を使って次のように表示せよ.

   i) Hello world. ii) 1+1=2

2. 次の数を順に表示せよ.

   i) 1 から 5 までの整数. ii) 5 から 1 までの整数 . iii) 1 から 10 にある偶数.

3. 9x9 表を作れ.

4. 1 から 5 までの和を求めよ.

5. n を 5 にして，$n! = n \times (n-1) \times (n-1) \cdots 3 \times 2 \times 1$ を求めよ.

[[[解答例]]]

1.
```
> printf("Hello world!!\n");

Hello world!!
```

```
> i:=1;
> printf("%d+%d=%d\n",i,i,i+i);
```

$$1$$

$$1+1=2$$

2. i)

```
> for i from 1 to 5 do
    i;
  end do;
```

$$1$$

$$2$$

$$3$$

$$4$$

$$5$$

ii)

```
> for i from 5 to 1 by -1 do
    i;
  end do;
```

$$5$$

$$4$$

$$3$$

$$2$$

$$1$$

iii)

```
> for i from 2 to 10 by 2 do
    i;
  end do;
```

$$2$$

$$4$$

$$6$$

$$8$$

$$10$$

3.
```
> for i from 1 to 9 do
    for j from 1 to 9 do
      printf("%4d",i*j);
    end do;
    printf("\n");
  end do;
```

1	2	3	4	5	6	7	8	9
2	4	6	8	10	12	14	16	18
3	6	9	12	15	18	21	24	27
4	8	12	16	20	24	28	32	36
5	10	15	20	25	30	35	40	45
6	12	18	24	30	36	42	48	54
7	14	21	28	35	42	49	56	63
8	16	24	32	40	48	56	64	72
9	18	27	36	45	54	63	72	81

```
> sum1:=0; for i from 1 to 5 do
    sum1:=sum1+i;
  end do;
```

0

1

3

6

10

15

4.
```
> n:=5:
  total1:=1:
  for i from 1 to n do
    total1:=total1*i;
  end do;
```

1

2

6

24

120

6.3 配列 (List)

[[[解説]]]

配列は変数を入れる箱が沢山用意されていると考えればよい．配列を使うときは，箱を指す数 (示数，index) をいじっているのか，箱の中身 (要素) をいじっているのかを区別すれば，動作を理解しやすい．Maple にはいくつかの配列構造が用意されている．もっとも，頻繁に使う list を示す．

6.3.1 基本

リスト構造は，中に入れる要素を [] でくくる．

```
> restart; list1:=[1,3,5,7];
```

$$list1 := [1, 3, 5, 7]$$

要素にアクセスするには，以下のようにインデックスを指定する．

```
> list1[2]; list1[-1]; list1[2..4];
```

$$3$$
$$7$$
$$[3, 5, 7]$$

-1,-2 等は後ろから 1 番目，2 番目を指す．C 言語と違い 0 番目はない．

```
> list1[0];
```

```
Error, invalid subscript selector
```

ひとつの要素を書き換えるには，以下のようにする．

```
> list1[3]:=x: list1;
```

$$[1, 3, x, 7]$$

要素の数，および要素の中身を取り出すには以下のようにする．

```
> nops(list1);
> op(list1);
```

$$4$$
$$1, 3, x, 7$$

6.3.2 for-loop の省略形

for-loop を省略するのによく使う手を二つ．(#より後ろはコメント文です)

配列の生成 (seq)

```
> aa:=[]; #空で初期化
  for i from 1 to 3 do
    aa:=[op(aa),i]; #付け足していく
  end do:
  print(aa);
```

$$aa := []$$

$$[1, 2, 3]$$

同じことを seq を使って短く書くことができる.

```
> aa :=[seq(i,i=1..3)];
```

$$aa := [1, 2, 3]$$

配列の和 (sum)

```
> n:=nops(aa):
  total:=0:
  for i from 1 to n do
    total:=total+aa[i];
  end do:
  print(total):
```

$$6$$

同じことを sum を使って短く書くことができる.

```
> sum(aa[i],i=1..nops(aa));
```

```
Error, invalid subscript selector
```

sum や seq を使っていると，このようなエラーがよくでる．これは，for-loop をまわすときに i に値が代入されているため引っかかる．変数を換えるか，i を初期化すればよい.

```
> i;
```

$$4$$

```
> sum(aa[j],j=1..nops(aa));
```

$$6$$

6.3.3 リストへの付け足し (append, prepend)

op を用いると，リストに新たな要素を前後，あるいは途中に付け足すことができる.

```
> list1:=[op(list1),9];
```

$$list1 := [1, 3, x, 7, 9]$$

6.3.4 2 つの要素の入れ替え

要素の 3，4 番目の入れ替えは以下の通り.

```
> tmp:=list1[3]:
  list1[3]:=list1[4]:
  list1[4]:=tmp:
  list1;
```

$$[1, 3, 7, x, 9]$$

6.3.5 2 次元配列 (listlist)

[ ] を二重化することで 2 次元の配列を作ることも可能で, リストのリスト (listlist) と呼ばれる.

```
> l2:=[[1,2,3,4],[1,3,5,7]];
```

$$l2 := [[1, 2, 3, 4], [1, 3, 5, 7]]$$

要素へのアクセスは以下の通り.

```
> l2[2]; l2[2,3]; l2[2][3];
```

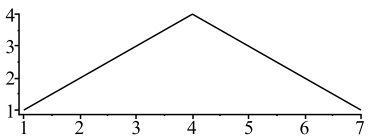
$$[1, 3, 5, 7]$$
$$5$$
$$5$$

6.3.6 list の表示 (listplot)

list に入っている数値を視覚化するのには listplot が便利.

```
> la:=[1,2,3,4,3,2,1];
  with(plots):
  listplot(la);
```

$$[1, 2, 3, 4, 3, 2, 1]$$



[[[課題]]]

1. 1 から 100 までの整数のうち 5 個をランダムに含んだ配列を生成せよ.

   1 から 6 までのランダムな数を生成する関数は,

   ```
   > roll:=rand(1..6):
   ```

として作ることができる．実行は次の通り．

```
> seq(roll(),i=1..10);
```

$$5, 2, 5, 6, 2, 3, 4, 4, 6, 5$$

2. さいころを１００回振って，出た目１から６が何回出たかを表示せよ．
3. コイン６枚を一度に投げて，表向きの枚数を数えるプログラムを書け．
4. 0 から 9 までの整数 5 個から 5 桁の整数を作れ．（1 桁目が 0 になっても気にするな）
5. 小数点以下 8 桁のそれぞれの桁の数を配列に格納せよ．8 桁の少数は以下のようにして作られる．
6. 255 以下の 10 進数をランダムに生成して，8 桁の 2 進数へ変換せよ．
   整数の割り算には irem(余り) と iquo(商) がある．使用法は以下の通り．

```
> irem(7,3); #res: 1
> iquo(7,3); #res: 2
```

[[[解答例]]]

1.
```
> roll:=rand(1..100):
  [seq(roll(),i=1..5)];
```

$$[27, 96, 17, 90, 34]$$

2.
```
> roll:=rand(1..6):
> A:=[seq(0,i=1..6)];
```

$$[0, 0, 0, 0, 0, 0]$$

```
> for i from 1 to 100 do
    i1:=roll();
    A[i1]:=A[i1]+1;
  end do:
  A;
```

$$[16, 18, 21, 18, 18, 9]$$

3.
```
> toss:=rand(0..1):
  n:=6:
  up:=0:
  for i from 1 to n do
    up:=up+toss();
  end do:
  up;
```

4.
```
> roll:=rand(0..9):
  n:=5:
  A:=[seq(roll(),i=1..n)];
```

$$[5, 7, 3, 7, 6]$$

```
> sum1:=0:
  for i from 1 to n do
    sum1:=sum1*10+A[i];
  end do:
  sum1;
```

$$57376$$

5.
```
> restart;
  n:=8:
  roll:=rand(10^(n-1)..10^n):
  B:=evalf(roll()/10^n,8);
  A:=[]:
```

$$0.19550684$$

```
> B:=10*B;
  for i from 1 to n do
    A:=[op(A),floor(B)];
    B:=(B-A[i])*10;
  end do:
  A;
```

$$1.95506840$$

$$[1, 9, 5, 5, 0, 6, 8, 4]$$

6.
```
> n:=8:
  roll:=rand(0..2^n-1):
  B:=roll();
```

$$246$$

```
> A:=[seq(0,j=1..n)]:
  for i from 1 to n do
    A[n-i+1]:=irem(B,2);
    B:=iquo(B,2);
  end do:
  A;
```

$$[1, 1, 1, 1, 0, 1, 1, 0]$$

6.4 交通整理 (If)

[[[解説]]]

6.4.1 if

もっとも簡単な if 文の例.

```
> x:=-4:
  if (x<0) then
     y:=-x;
  end if;
```

4

例外付き.

```
> x:=3:
  if (x<0) then
    y:=-x;
  else
    y:=x;
  end if;
```

3

2 個の条件がある例

```
> x:=3:
  if (x<0) then
    y:=-x;
  elif (x>5) then
    y:=x;
  else
    y:=2*x;
  end if;
```

6

条件文に使える式と意味 関係演算子は<，<=，>，>=，=，<>で表記される．論理演算子には and, or, xor, not がある．その他にもブール値を返す関数として implies, evalb, type などいくつかあり，条件分岐に使える．

表 6.2: 条件分岐のいくつかの例

x と y の値が一致	(x=y)
x と y の値が一致しない	(x<>y)
条件文を複数つなぐ	((x>0) and (x<4))
	((x<0) or (x>4))
	not (x=0)

6.4.2 next と break

do-loop の途中で流れを変更するための命令．next は do-loop を一回スキップ．break はそこで do-loop を一つ抜ける．以下のコードの出力結果を参照．

```
> for i from 1 to 5 do
    if (i=3) then
  next;
    end if;
    print(i);
  end do:

#res: 1 2 4 5

> for i from 1 to 5 do
    if (i=3) then
      break;
    end if;
    print(i);
  end do:

#res: 1 2
```

[[[課題]]]

1. 西暦を代入したら，明治，大正，昭和，平成で答えてくれるプログラムを作成せよ．西暦 1868, 1912,1926,1989 年をそれぞれの元年とする．
2. 整数を代入したら，それ以下の素数をすべて表示するプログラムを作れ．素数かどうかの判定は Maple コマンドの isprime を用いよ．
3. p が素数で p+2 も素数のとき，これらは双子の素数と呼ばれる．10 以上，100 以下の双子の素数を全部見つけて出力せよ．
4. 素数判定を原理から実現せよ．ある数 n が素数かどうか (自分自身の数 n と 1 以外の数で割りきれないかどうか) を判定せよ．割り算の余り (剰余) は irem で求まる．例えば

   ```
   > residue:=irem(9,2);
   ```

   として変数 residue(余りの英語) を printf してみよ．番兵を置いておいて，n-1 から 2 までの数で n を次々と割っていき，一度でも割り切れれば番兵にマークをつける．ループが終わった後に番兵のマークを見て素数 (prime number) かどうかを判定すればよい．
5. うるう年かどうかを表示するプログラムをかけ．
   うるう年は 4 で割り切れる数の年．ただし，100 で割り切れる年はうるう年でなく，400 で割り切れる年はうるう年．
6. ゴールドバッハの予想
   「6 以上の偶数は二つの素数の和として表わされる」という予想を 100 以下の偶数について検証せよ．あらかじめ 100 までの素数をリストアップしておいてそのなかから組み合わせを探すと便利．

[[[解答例]]]

1.
```
> year:=1890;
  if year<1868 then printf("明治より前です. \n");
  elif year<1912 then printf("明治%d 年です. \n",year-1868+1);
  elif year<1926 then printf("大正%d 年です. \n",year-1912+1);
  elif year<1989 then printf("昭和%d 年です. \n",year-1926+1);
  elif year<2011 then printf("平成%d 年です. \n",year-1989+1);
  else printf("今年より後です. \n");
  end;
```

  明治 23 年です.

2.
```
> n:=10:
  for i from 1 to n do
    if (isprime(i)) then
      print(i);
    end if;
  end do;
```

  #res: 2 3 5 7

3.
```
> for i from 10 to 100-2 do
    if (isprime(i) and isprime(i+2)) then
      print(i,i+2);
    end if;
  end do;
```

  11, 13
  17, 19
  29, 31
  41, 43
  59, 61
  71, 73

4.
```
> n:=12:
  banpei:=0:
  for i from 2 to n-1 do
    residue:=irem(n,i);
    # print(n,residue):
    if residue=0 then
      banpei:=1;
      break;
    end if;
  end do:
  if banpei=1 then
    printf("%d is not prime number.\n",n);
  else
    printf("%d is prime number.\n",n);
  end if;
```

```
12 is not prime number.
```

5.
```
> year:=[2010,1984,2004,1800,1900,1600,2000]:
  for i from 1 to nops(year) do
    if (irem(year[i],400)=0) then
      printf("%d is a leap year.\n",year[i]);
    elif (irem(year[i],4)=0) and (irem(year[i],100)<>0) then
      printf("%d is a leap year.\n",year[i]);
    else printf("%d is not a leap year.\n",year[i]);
    end if;
  end do;
```

```
2010 is not a leap year.
1984 is a leap year.
2004 is a leap year.
1800 is not a leap year.
1900 is not a leap year.
1600 is a leap year.
2000 is a leap year.
```

別解

```
> for i from 1 to nops(year) do
    if (irem(year[i],4)=0) and ((irem(year[i],100)<>0) or (irem(year[i],400)=0)) then
      printf("%d is a leap year.\n",year[i]);
    else
      printf("%d is not a leap year.\n",year[i]);
    end if;
  end do;
```

略

6.
```
> prime1:=[];
  for i from 1 to 100 do
    if isprime(i) then
      prime1:=[op(prime1),i];
    end if;
  end do;
  prime1;
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71,
73, 79, 83, 89, 97]
```

```
> nops(prime1);
```

```
> for i from 6 to 100 by 2 do
    for j1 from 1 to nops(prime1) do
      for j2 from 1 to nops(prime1) do
        if i=(prime1[j1]+prime1[j2]) then
          print(i,prime1[j1],prime1[j2]);
          break;
        end if
      end do;
      if j2<=nops(prime1) then
        break;
      end if;
    end do;
  end do;
```

```
6, 3, 3
8, 3, 5
10, 3, 7
中略
98, 19, 79
100, 3, 97
```

6.5 手続き関数 (Procedure)

[[[解説]]]

6.5.1 基本

複雑な手続きや，何度も繰り返すルーチンは proc で作る．proc は以下のようにして作る．

ユーザ関数名:=proc(仮引数)
  動作
end proc;

```
> test1:=proc(a)
    print(a);
  end proc:
```

proc の呼び出しは，以下のようになる．

```
> test1(13);
```

13

仮引数としてはどんな型 (変数や配列) でもよい．複数指定するときにはコンマで区切る．仮引数を proc の中で変更することはできない．下で示す global で取り込むか，local 変数にコピーして使う．

6.5.2 戻り値

proc の戻り値は return で指定される．return 文がないときは，最後の動作結果が戻り値となる．

```
> test2:=proc(a)
    return a+1;
  end proc:

> test2(13);
```

14

6.5.3 グローバル (大域)，ローカル (局所) 変数

proc の内部だけで使われるのが local，外部を参照するのが global．global,local を省略しても Maple が適当に判断してくれるが，あまり信用せず，明示的に宣言した方が良い．宣言の仕方は以下の通り．

変数名:=proc(引数)
  local 変数, 変数...;
  global 変数, 変数...;
  動作の記述
end proc;

[[[課題]]]

1. 三角形の面積底辺と高さを引数として，面積を返す関数 area を作れ．

2. MyIsprime 前章の課題 4 で求めた素数判定プログラムを proc にせよ．

3. ルートの距離

   二つの位置座標 x1:=[0.0, 0.0] x2:=[1.0, 1.0] から距離を求める MyDistance 関数を作れ．

   次に，4 つの位置座標 x[1]=[0.0, 0.0] x[2]=[1.0, 1.0] x[3]=[1.0, 0.0] x[4]=[0.0, 1.0] を読み込んで，座標順に [1,2,3,4,1] と巡る距離を求めよ．

4. 最大数

   ランダムな整数が格納されたリストを受け取り，そのリスト中の最大数を返す関数 MyMax を作れ．1 から 100 までのランダムな整数のリストは次のようにして作れる．

```
> roll:=rand(1..100):
  n:=50:
  A:=[seq(roll(),i=1..n)];
```

$$A := [45, 96, 6, 98, 59, 44, 100, 38, 69, 27, 96, 17, 90, 34, 18, 52, 56, 43, 83, 25, 90, 93, 60, 93, 14, 50, 47, 8, 46, 44, 9, 77, 59, 16, 1, 70, 77, 39, 92, 71, 67, 78, 51, 53, 12, 19, 63, 40, 90, 3]$$

[[[解答例]]]

1.
```
> area:=proc(base,height)
    base*height/2;
  end proc:

> area(3,4); #res: 6
```

2.
```
> restart;
  n:=19:
  banpei:=0;
  for i from 2 to n-1 do
    amari:=irem(n,i);
    print(amari):
    if amari=0 then
      banpei:=1;
      break;
    end if;
  end do:
  if banpei=1 then
    print(n," is not prime number.");
  else
    print(n," is prime number.");
  end if;
```

0

1

1

1

19,” *is prime number.*”

```
> MyIsprime:=proc(n)
    local i,amari;
    for i from 2 to evalf(sqrt(n)) do
      amari:=irem(n,i);
      if amari=0 then
        return false;
      end if;
    end do:
    return true;
  end proc:

> MyIsprime(104729);
```

true

3.
```
> restart; x1:=[0.0, 0.0]: x2:=[1.0, 1.0]:

> MyDistance:=proc(x1,x2)
    local dx,dy;
    dx:=(x1[1]-x2[1]);
    dy:=(x1[2]-x2[2]);
    sqrt(dx^2+dy^2);
  end proc:

> MyDistance(x1,x2);
```

1.414213562

```
> x[1]:=[0.0, 0.0]: x[2]:=[1.0, 1.0]: x[3]:=[1.0, 0.0]: x[4]:=[0.0, 1.0]: x[5]:=x[1]:
  sum(MyDistance(x[i],x[i+1]),i=1..4);
```

4.828427124

4.
```
> MyMax:=proc(A)
    local imax,i;
    imax:=0;
    for i from 1 to nops(A) do
```

```
    if A[i]>imax then
        imax:=A[i];
    end if
  end do;
  return imax;
end proc:

> MyMax(A);
```

100