EMT untuk Perhitungan Aljabar

Pada notebook ini Anda belajar menggunakan EMT untuk melakukan berbagai perhitungan terkait dengan materi atau topik dalam Aljabar. Kegiatan yang harus Anda lakukan adalah sebagai berikut:

- Membaca secara cermat dan teliti notebook ini;
- Menerjemahkan teks bahasa Inggris ke bahasa Indonesia;
- Mencoba contoh-contoh perhitungan (perintah EMT) dengan cara meng-ENTER setiap perintah EMT yang ada (pindahkan kursor ke baris perintah)
- Jika perlu Anda dapat memodifikasi perintah yang ada dan memberikan keterangan/penjelasan tambahan terkait hasilnya.
- Menyisipkan baris-baris perintah baru untuk mengerjakan soal-soal Aljabar dari file PDF yang saya berikan;
- Memberi catatan hasilnya.
- Jika perlu tuliskan soalnya pada teks notebook (menggunakan format LaTeX).
- Gunakan tampilan hasil semua perhitungan yang eksak atau simbolik dengan format LaTeX. (Seperti contoh-contoh pada notebook ini.)

Identitas

Nama : Intan Nur Setyarini
NIM : 24030130001
Kelas: PMA 24

Contoh pertama

Menyederhanakan bentuk aljabar:

$$6x^{-3}y^5 \times -7x^2y^{-9}$$

```
>$&6*x^(-3)*y^5*-7*x^2*y^(-9)
```

$$-\frac{42}{x\,y^4}$$

Menjabarkan:

$$(6x^{-3}+y^5)(-7x^2-y^{-9})$$

```
> $&showev('expand((6*x^(-3)+y^5)*(-7*x^2-y^(-9))))
```

$$expand\left(\left(-\frac{1}{y^9}-7\,x^2\right)\left(y^5+\frac{6}{x^3}\right)\right)=-7\,x^2\,y^5-\frac{1}{y^4}-\frac{6}{x^3\,y^9}-\frac{42}{x}$$

Menyederhanakan aljabar

$$\frac{24a^{10}b^{-8}c^{7}}{12a^{6}b^{-3}c^{5}}$$

```
>$&(24*a^(10)*b^(-8)*c^7)/(12*a^(6)*b^(-3)*c^5)
```

$$\frac{2\,a^{4}\,c^{2}}{b^{5}}$$

Bentuk aljabar tersebut tidak dapat dijabarkan karena bukan perkalian

Baris Perintah

Sebuah baris perintah pada Euler terdiri dari satu atau beberapa perintah Euler yang diikuti oleh titik koma ";" atau koma ",". Tanda titik koma berfungsi untuk mencegah hasil ditampilkan. Koma setelah perintah terakhir boleh dihilangkan.
Baris perintah berikut hanya akan menampilkan hasil dari ekspresi, bukan penugasan atau perintah format.

```
>r:=2; h:=4; pi*r^2*h/3 ...
```

```
  16.7551608191
```

Perintah harus dipisahkan dengan spasi. Baris perintah berikut akan menampilkan dua hasilnya.

```
>pi*2*r*h, %+2*pi*r*h // Ingat tanda % menyatakan hasil perhitungan terakhir sebelumnya
```

```
  50.2654824574
  100.530964915
```

Baris perintah dijalankan sesuai urutan saat pengguna menekan tombol Enter/Return. Jadi, Anda akan mendapatkan nilai baru setiap kali menjalankan baris kedua.

```
>x := 1;
>x := cos(x) // nilai cosinus (x dalam radian)
```

```
  0.540302305868
```

```
>x := cos(x)
```

```
  0.857553215846
```

Jika dua baris dihubungkan dengan tanda "...", maka kedua baris tersebut akan selalu dijalankan secara bersamaan.

```
>x := 1.5; ...
>x := (x+2/x)/2, x := (x+2/x)/2, x := (x+2/x)/2,
```

```
  1.41666666667
  1.41421568627
  1.41421356237
```

Ini juga merupakan cara yang baik untuk membagi sebuah perintah panjang ke dalam dua baris atau lebih. Anda dapat menekan Ctrl+Enter untuk membagi satu baris menjadi dua pada posisi kursor saat ini, atau Ctrl+Back untuk menggabungkan kembali baris-baris tersebut.

Untuk melipat semua multi-line, tekan Ctrl+L. Dengan begitu, baris-baris berikutnya hanya akan terlihat jika salah satunya sedang mendapat fokus. Untuk melipat satu multi-line saja, mulai baris pertamanya dengan "%+ ".

```
>%+ x=4+5;  // Baris ini tidak akan terlihat setelah kursor berpindah dari baris tersebut.
```

Sebuah baris yang diawali dengan %% akan sepenuhnya tidak terlihat.

```
  81
```

Euler mendukung penggunaan perulangan (loop) di dalam baris perintah, selama perulangan tersebut muat dalam satu baris atau dalam bentuk multi-line. Namun, dalam program, tentu saja pembatasan ini tidak berlaku. Untuk informasi lebih lanjut, silakan lihat pengantar berikut.

```
>x=1; for i=1 to 5; x := (x+2/x)/2, end; // menghitung akar 2
```

```
  1.5
  1.41666666667
  1.41421568627
  1.41421356237
  1.41421356237
```

Tidak masalah menggunakan multi-line. Pastikan baris diakhiri dengan "...".

```
>x := 1.5; // Komentar ditempatkan di sini sebelum bagian tersebut ...
>repeat xnew:=(x+2/x)/2; until xnew~=x; ...
>   x := xnew; ...
>end; ...
>x,
```

```
  1.41421356237
```

Struktur kondisional juga dapat berfungsi.

```
>if E^pi>pi^E; then "Thought so!", endif;
```

```
  Thought so!
```

Saat Anda menjalankan sebuah perintah, kursor dapat berada di posisi mana pun dalam baris perintah. Anda bisa kembali ke perintah sebelumnya atau melompat ke perintah berikutnya dengan tombol panah. Atau, Anda juga dapat mengklik bagian komentar di atas perintah untuk berpindah ke perintah tersebut.

Ketika Anda memindahkan kursor sepanjang baris, pasangan tanda kurung buka dan kurung tutup akan disorot. Selain itu, perhatikan status line. Setelah tanda kurung buka pada fungsi sqrt(), status line akan menampilkan teks bantuan untuk fungsi tersebut. Jalankan perintah dengan menekan tombol Enter.

```
>sqrt(sin(10°)/cos(20°))
```

```
  0.429875017772
```

Untuk melihat bantuan dari perintah terbaru, buka jendela bantuan dengan menekan F1. Di sana, Anda dapat memasukkan teks untuk melakukan pencarian. Pada baris kosong, bantuan mengenai jendela bantuan akan ditampilkan. Anda dapat menekan Escape untuk menghapus isi baris, atau untuk menutup jendela bantuan.

Anda juga dapat melakukan double click pada perintah apa pun untuk membuka bantuan terkait perintah tersebut. Cobalah klik ganda pada perintah exp di bawah ini pada baris perintah.

```
>exp(log(2.5))
```

```
  2.5
```

Anda juga dapat melakukan copy dan paste di Euler. Gunakan Ctrl+C dan Ctrl+V untuk itu. Untuk menandai teks, seret kursor mouse atau gunakan tombol Shift bersama dengan tombol panah. Selain itu, Anda juga dapat menyalin tanda kurung yang sedang disorot.

Sintaks Dasar

Euler mengenal fungsi-fungsi matematika yang umum. Seperti yang telah Anda lihat sebelumnya, fungsi trigonometri dapat bekerja dalam radian maupun derajat. Untuk mengubah ke derajat, tambahkan simbol derajat (dengan tombol F7) pada nilai, atau gunakan fungsi rad(x). Fungsi akar kuadrat disebut sqrt di Euler. Tentu saja, penulisan x^(1/2) juga dimungkinkan.

Untuk menetapkan variabel, gunakan tanda "=" atau ":=". Demi kejelasan, pengantar ini menggunakan bentuk yang terakhir. Spasi tidak berpengaruh, tetapi spasi antar perintah memang diperlukan.

Beberapa perintah dalam satu baris dipisahkan dengan "," atau ";". Tanda titik koma ";" akan menekan (menyembunyikan) keluaran perintah. Pada akhir baris perintah, tanda "," akan dianggap ada jika tanda ";" tidak dituliskan.

```
>g:=9.81; t:=2.5; 1/2*g*t^2
```

```
  30.65625
```

EMT menggunakan sintaks pemrograman untuk menuliskan ekspresi. Untuk memasukkan

$$e^2 \cdot \left(\frac{1}{3 + 4\log(0.6)} + \frac{1}{7} \right)$$

Anda harus menuliskan tanda kurung dengan benar serta menggunakan tanda / untuk pecahan. Perhatikan tanda kurung yang tersorot sebagai bantuan. Perlu dicatat bahwa konstanta Euler e disebut E di EMT.

```
>E^2*(1/(3+4*log(0.6))+1/7)
```

```
  8.77908249441
```

Untuk menghitung ekspresi yang rumit seperti

$$\left(\frac{\frac{1}{7}+\frac{1}{8}+2}{\frac{1}{3}+\frac{1}{2}}\right)^2 \pi$$

Anda perlu menuliskannya dalam bentuk baris (line form).

```
>((1/7 + 1/8 + 2) / (1/3 + 1/2))^2 * pi
```

```
  23.2671801626
```

Letakkan tanda kurung dengan cermat di sekitar sub-ekspresi yang harus dihitung terlebih dahulu. EMT membantu dengan menyorot ekspresi yang ditutup oleh kurung penutup tersebut. Anda juga perlu memasukkan nama "pi" untuk huruf Yunani pi.

Hasil perhitungan ini berupa bilangan floating point (titik-mengambang). Secara bawaan, hasil dicetak dengan ketelitian sekitar 12 digit. Pada baris perintah berikutnya, kita juga akan mempelajari cara merujuk hasil sebelumnya dalam baris yang sama.

```
>1/3+1/7, fraction %
```

```
  0.47619047619
  10/21
```

Sebuah perintah Euler dapat berupa ekspresi atau perintah primitif. Ekspresi tersusun dari operator dan fungsi. Jika diperlukan, ekspresi harus menggunakan tanda kurung untuk memastikan urutan eksekusi yang benar. Jika ragu, menambahkan tanda kurung adalah pilihan yang baik. Perlu dicatat bahwa EMT menampilkan pasangan kurung buka dan kurung tutup saat Anda mengedit baris perintah.

```
>(cos(pi/4)+1)^3*(sin(pi/4)+1)^2
```

```
  14.4978445072
```

Operator numerik yang tersedia di Euler meliputi:

```
+ unary or operator plus
- unary or operator minus
*, /
. perkalian matrik
a^b power for positive a or integer b (a**b works too)
n! the factorial operator
```

dan masih banyak lagi.

Berikut beberapa fungsi yang mungkin sering Anda perlukan (selain itu masih ada banyak fungsi lainnya

```
sin,cos,tan,atan,asin,acos,rad,deg
log,exp,log10,sqrt,logbase
bin,logbin,logfac,mod,floor,ceil,round,abs,sign
conj,re,im,arg,conj,real,complex
beta,betai,gamma,complexgamma,ellrf,ellf,ellrd,elle
bitand,bitor,bitxor,bitnot
```

Beberapa perintah memiliki alias, misalnya ln untuk log.

```
>ln(E^2), arctan(tan(0.5))
```

```
  2
  0.5
```

```
>sin(30°)
```

```
  0.5
```

Pastikan untuk selalu menggunakan (tanda kurung), jika ada keraguan mengenai urutan eksekusi! Berikut ini tidak sama dengan (2^3)^4
, karena dalam EMT, bentuk 2^3^4 secara bawaan dihitung sebagai 2^(3^4). (Beberapa sistem numerik lain justru menerapkan aturan sebaliknya).

```
>2^3^4, (2^3)^4, 2^(3^4)
```

```
  2.41785163923e+24
  4096
  2.41785163923e+24
```

Bilangan Real

Tipe data utama di Euler adalah bilangan real. Bilangan real direpresentasikan dalam format IEEE dengan ketelitian sekitar 16 digit desimal.

```
>longest 1/3
```

```
       0.3333333333333333
```

Representasi ganda internal menggunakan 8 byte.

```
> printdual(1/4)
```

```
  1.0000000000000000000000000000000000000000000000000000*2^-2
```

Dalam perhitungan manual, 1/4 dapat ditulis

$$1 * 2^{-2}$$

```
>printdual(1/3)
```

```
  1.0101010101010101010101010101010101010101010101010101*2^-2
```

```
>printhex(1/3)
```

```
  5.5555555555554*16^-1
```

String

Sebuah string di Euler didefinisikan dengan tanda kutip ganda "...".

```
>"A string can contain anything."
```

```
  A string can contain anything.
```

String dapat digabungkan menggunakan | atau +. Hal ini juga berlaku untuk angka, yang dalam kasus tersebut akan otomatis dikonversi menjadi string.

```
>"The area of the circle with radius " + 2 + " cm is " + pi*4 + " cm^2."
```

```
  The area of the circle with radius 2 cm is 12.5663706144 cm^2.
```

Fungsi print juga dapat mengonversi sebuah angka menjadi string. Fungsi ini bisa menerima parameter berupa jumlah digit dan jumlah tempat desimal (gunakan 0 untuk dense output), serta secara opsional juga bisa menyertakan satuan (unit).

```
>"Golden Ratio : " + print((1+sqrt(5))/2,5,0)
```

```
  Golden Ratio : 1.61803
```

Ada string khusus bernama none, yang tidak akan ditampilkan (tidak tercetak). String ini dikembalikan oleh beberapa fungsi ketika hasilnya tidak penting. (String ini juga otomatis dikembalikan jika sebuah fungsi tidak memiliki pernyataan return).

```
>none
```

Untuk mengonversi sebuah string menjadi angka, cukup evaluasi string tersebut. Cara ini juga berlaku untuk ekspresi (lihat penjelasan di bawah).

```
>"1234.5"()
```

```
  1234.5
```

Untuk mendefinisikan vektor string, gunakan notasi [...].

```
>v:=["affe","charlie","bravo"]
```

```
  affe
  charlie
  bravo
```

Vektor string kosong dilambangkan dengan [none]. Vektor string juga dapat digabungkan.

```
>w:=[none]; w|v|v
```

```
  affe
  charlie
  bravo
  affe
  charlie
  bravo
```

String dapat berisi karakter Unicode. Secara internal, string ini menggunakan kode UTF-8. Untuk membuat string semacam itu, gunakan u"..." bersama salah satu HTML entities.
String Unicode dapat digabungkan seperti string lainnya.

```
>u"&alpha; = " + 45 + u"&deg;" // pdfLaTeX mungkin gagal menampilkan secara benar
```

```
   = 45°
```

I

Dalam komentar, entitas yang sama seperti a, ß, dan sebagainya dapat digunakan. Ini bisa menjadi alternatif cepat selain menggunakan LaTeX. (Penjelasan lebih detail tentang komentar akan dibahas di bawah).

Ada beberapa fungsi untuk membuat atau menganalisis string Unicode. Fungsi strtochar() akan mengenali string Unicode dan menerjemahkannya dengan benar.

```
>v=strtochar(u"&Auml; is a German letter")
```

```
  [196,  32,  105,  115,  32,  97,  32,  71,  101,  114,  109,  97,  110,
  32,  108,  101,  116,  116,  101,  114]
```

Hasil dari fungsi tersebut adalah sebuah vektor bilangan Unicode. Fungsi kebalikannya adalah chartoutf().

```
>v[1]=strtochar(u"&Uuml;")[1]; chartoutf(v)
```

```
  Ü is a German letter
```

Fungsi utf() dapat menerjemahkan sebuah string dengan entitas di dalam variabel menjadi string Unicode.

```
>s="We have &alpha;=&beta;."; utf(s) // pdfLaTeX mungkin gagal menampilkan secara benar
```

```
  We have =.
```

Penggunaan entitas numerik juga dimungkinkan.

```
>u"&#196;hnliches"
```

```
  Ähnliches
```

Nilai Boolean

Nilai boolean di Euler direpresentasikan dengan 1 = true atau 0 = false. String dapat dibandingkan sama seperti angka.

```
>2<1, "apel"<"banana"
```

```
  0
  1
```

"and" adalah operator "&&" dan "or" adalah operator "||", seperti dalam bahasa C.
Kata "and" dan "or" hanya dapat digunakan dalam kondisi pada pernyataan if.)

```
>2<E && E<3
```

```
  1
```

Operator boolean mengikuti aturan dalam bahasa matriks.

```
>(1:10)>5, nonzeros(%)
```

```
  [0,  0,  0,  0,  0,  1,  1,  1,  1,  1]
  [6,  7,  8,  9,  10]
```

Anda dapat menggunakan fungsi nonzeros() untuk mengekstrak elemen-elemen tertentu dari sebuah vektor.
Pada contoh, digunakan kondisi isprime(n).

```
>N=2|3:2:99 // N berisi elemen 2 dan bilangan2 ganjil dari 3 s.d. 99
```

```
  [2,  3,  5,  7,  9,  11,  13,  15,  17,  19,  21,  23,  25,  27,  29,
  31,  33,  35,  37,  39,  41,  43,  45,  47,  49,  51,  53,  55,  57,
  59,  61,  63,  65,  67,  69,  71,  73,  75,  77,  79,  81,  83,  85,
  87,  89,  91,  93,  95,  97,  99]
```

```
>N[nonzeros(isprime(N))] //pilih anggota2 N yang prima
```

```
  [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]
```

Format Keluaran

Format keluaran bawaan EMT menampilkan 12 digit. Untuk memastikan kita melihat format default tersebut, kita dapat mereset formatnya.

```
>defformat; pi
```

```
  3.14159265359
```

Secara internal, EMT menggunakan standar IEEE untuk bilangan double dengan ketelitian sekitar 16 digit desimal. Untuk melihat jumlah digit secara penuh, gunakan perintah longestformat, atau gunakan operator longest untuk menampilkan hasil dalam format terpanjang.

```
>longest pi
```

```
        3.141592653589793
```

Berikut adalah representasi heksadesimal internal dari sebuah bilangan double.

```
>printhex(pi)
```

```
  3.243F6A8885A30*16^0
```

Format keluaran dapat diubah secara permanen dengan menggunakan perintah format.

```
>format(12,5); 1/3, pi, sin(1)
```

```
      0.33333
      3.14159
      0.84147
```

Bawaan (default) yang digunakan adalah format(12).

```
>format(12); 1/3
```

```
  0.333333333333
```

Fungsi seperti "shortestformat", "shortformat", dan "longformat" bekerja pada vektor dengan cara berikut.

```
>shortestformat; random(3,8)
```

```
    0.66    0.2   0.89   0.28   0.53   0.31   0.44    0.3
    0.28   0.88   0.27    0.7   0.22   0.45   0.31   0.91
    0.19   0.46  0.095    0.6   0.43   0.73   0.47   0.32
```

Format bawaan untuk skalar adalah format(12). Namun, format ini dapat diubah sesuai kebutuhan.

```
>setscalarformat(5); pi
```

```
  3.1416
```

Fungsi "longestformat" juga mengatur format untuk skalar.

```
>longestformat; pi
```

```
  3.141592653589793
```

Sebagai referensi, berikut daftar format keluaran yang paling penting:
shortestformat, shortformat, longformat, longestformat

format(length, digits), goodformat(length)

fracformat(length)

defformat

Ketelitian internal EMT sekitar 16 digit desimal, sesuai dengan standar IEEE. Angka-angka disimpan dalam format internal ini.

Namun, format keluaran EMT dapat diatur secara fleksibel sesuai kebutuhan.

```
>longestformat; pi,
```

```
  3.141592653589793
```

```
> goodformat(16); pi
```

```
  3.14159265359
```

```
>goodformat(10); pi
```

```
  3.14159265359
```

perubahan lenght pada goodformat tidak akan mengubah digit. Jumlah digit akan tetap 12 digit.

```
>format(10,5); pi
```

```
    3.14159
```

digits 5 akan menampilkan jumlah digit setelah koma
Bawaan (default) yang digunakan adalah defformat().

```
>defformat; // default
```

Terdapat operator pendek yang hanya mencetak satu nilai. Operator "longest" akan menampilkan semua digit valid dari sebuah angka.

```
>longest pi^2/2
```

```
        4.934802200544679
```

Ada juga operator pendek untuk mencetak hasil dalam format pecahan. Kita sudah pernah menggunakannya sebelumnya.

```
>fraction 1+1/2+1/3+1/4
```

```
  25/12
```

Karena format internal menggunakan penyimpanan angka secara biner, nilai 0.1 tidak akan direpresentasikan secara tepat. Kesalahan ini akan menumpuk sedikit, seperti yang terlihat pada perhitungan berikut.

```
>longest 0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1-1
```

```
   -1.110223024625157e-16
```

Namun dengan longformat bawaan, hal ini tidak akan terlihat. Untuk kenyamanan, hasil dari angka yang sangat kecil ditampilkan sebagai 0.

```
>0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1-1
```

```
  0
```

```
>longest (4.2*10^7)*(3.2*10^(-2))
```

```
                  1344000
```

```
>(4.2*10^7)*(3.2*10^(-2))
```

```
  1344000
```

Karena hasil perkalian merupakan bilangan bulat yang tidak mencapai lebih dari 12 digit, sehingga hasilnya sama meskipun memakai longest

Ekspresi

String atau nama dapat digunakan untuk menyimpan ekspresi matematika, yang dapat dievaluasi oleh EMT. Untuk melakukan ini, gunakan tanda kurung setelah ekspresi. Jika Anda ingin menggunakan string sebagai ekspresi, gunakan konvensi penamaan seperti "fx" atau "fxy", dan seterusnya.
Ekspresi memiliki prioritas lebih tinggi dibanding fungsi.
Variabel global juga dapat digunakan dalam proses evaluasi.

```
>r:=2; fx:="pi*r^2"; longest fx()
```

```
        12.56637061435917
```

Parameter akan ditetapkan ke x, y, dan z secara berurutan. Parameter tambahan dapat ditambahkan dengan menggunakan parameter yang telah ditetapkan sebelumnya.

```
>fx:="a*sin(x)^2"; fx(5,a=-1)
```

```
  -0.919535764538
```

Perlu dicatat bahwa ekspresi akan selalu menggunakan variabel global, meskipun ada variabel dalam fungsi dengan nama yang sama. (Jika tidak, evaluasi ekspresi di dalam fungsi bisa menghasilkan hasil yang sangat membingungkan bagi pengguna yang memanggil fungsi tersebut.)

```
>at:=4; function f(expr,x,at) := expr(x); ...
>f("at*x^2",3,5) // computes 4*3^2 not 5*3^2
```

```
  36
```

Jika Anda ingin menggunakan nilai lain untuk "at" selain nilai global, Anda perlu menambahkan "at=value".

```
>at:=4; function f(expr,x,a) := expr(x,at=a); ...
>f("at*x^2",3,5) // menghitung 5*3^2 karena nilai at didefinisikan pada nilai global
```

```
  45
```

Sebagai referensi, perlu dicatat bahwa call collections (dibahas di bagian lain) dapat berisi ekspresi. Jadi, contoh di atas dapat dibuat seperti berikut.

```
>at:=4; function f(expr,x) := expr(x); ...
>f({{"at*x^2",at=5}},3)
```

```
  45
```

Ekspresi dalam x sering digunakan sama seperti fungsi.
Perlu dicatat bahwa mendefinisikan fungsi dengan nama yang sama seperti ekspresi simbolik global akan menghapus variabel tersebut untuk menghindari kebingungan antara ekspresi simbolik dan fungsi.

```
>f &= 5*x;
>function f(x) := 6*x;
>f(2)
```

```
  12
```

Sebagai konvensi, ekspresi simbolik atau numerik sebaiknya dinamai fx, fxy, dan sebagainya. Skema penamaan ini tidak boleh digunakan untuk fungsi.

```
>fx &= diff(x^x,x); $&fx
```

$$x^x\ (\log x + 1)$$

Bentuk khusus dari sebuah ekspresi memungkinkan variabel apa pun digunakan sebagai parameter tanpa nama dalam evaluasi ekspresi, tidak terbatas pada x, y, dan sebagainya. Untuk ini, mulailah ekspresi dengan "@(variables) ...".

```
>"@(a,b) a^2+b^2", %(4,5)
```

```
  @(a,b) a^2+b^2
  41
```

Hal ini memungkinkan manipulasi ekspresi dalam variabel lain untuk fungsi-fungsi EMT yang membutuhkan ekspresi dalam x.

Cara paling sederhana untuk mendefinisikan fungsi adalah dengan menyimpan rumusnya dalam ekspresi simbolik atau numerik. Jika variabel utama adalah x, ekspresi tersebut dapat dievaluasi sama seperti sebuah fungsi.

Seperti yang terlihat pada contoh berikut, variabel global tetap terlihat selama proses evaluasi.

```
>fx &= x^3-a*x;  ...
>a=1.2; fx(0.5)
```

```
  -0.475
```

Semua variabel lain dalam ekspresi dapat ditentukan saat evaluasi dengan menggunakan parameter yang ditetapkan.

```
>fx(0.5,a=1.1)
```

```
  -0.425
```

Sebuah ekspresi tidak harus bersifat simbolik. Hal ini diperlukan jika ekspresi tersebut mengandung fungsi-fungsi yang hanya dikenal di kernel numerik, bukan di Maxima.

Matematika Simbolik

EMT melakukan matematika simbolik dengan bantuan Maxima. Untuk informasi lebih lanjut, mulai dengan tutorial berikut, atau telusuri referensi Maxima. Ahli Maxima perlu memperhatikan bahwa ada perbedaan sintaks antara sintaks asli Maxima dan sintaks default ekspresi simbolik di EMT.

Matematika simbolik terintegrasi secara mulus ke dalam Euler dengan &. Setiap ekspresi yang diawali dengan & adalah ekspresi simbolik. Ekspresi ini dievaluasi dan dicetak oleh Maxima.

Pertama-tama, Maxima memiliki aritmatika “tak terbatas” yang dapat menangani angka yang sangat besar.

```
>$&44!
```

$$26582715747884487680436258110146158903196385280000000000$$

Dengan cara ini, Anda dapat menghitung hasil besar secara tepat. Mari kita hitung:

$$C(44,10) = \frac{44!}{34! \cdot 10!}$$

```
>$& 44!/(34!*10!) // nilai C(44,10)
```

$$2481256778$$

Tentu saja, Maxima memiliki fungsi yang lebih efisien untuk ini (demikian juga bagian numerik EMT).

```
>$binomial(44,10) //menghitung C(44,10) menggunakan fungsi binomial()
```

$$2481256778$$

Untuk mempelajari lebih lanjut tentang fungsi tertentu, klik dua kali pada fungsi tersebut. Misalnya, coba klik dua kali pada "&binomial" di baris perintah sebelumnya. Ini akan membuka dokumentasi Maxima sebagaimana disediakan oleh pengembang program tersebut.
Anda juga akan mengetahui bahwa hal berikut ini juga dapat dilakukan.

$$C(x,3) = \frac{x!}{(x-3)!3!} = \frac{(x-3)(x-2)(x-1)x}{6}$$

```
>$binomial(x,3) // C(x,3)
```

$$\frac{(x-2)\ (x-1)\ x}{6}$$

Jika Anda ingin mengganti x dengan nilai tertentu, gunakan "with".

```
>$&binomial(x,3) with x=10 // substitusi x=10 ke C(x,3)
```

$$120$$

Dengan cara ini, Anda dapat menggunakan solusi suatu persamaan dalam persamaan lain.

Ekspresi simbolik dicetak oleh Maxima dalam bentuk 2D. Hal ini disebabkan oleh adanya flag simbolik khusus pada string.

Seperti yang telah Anda lihat pada contoh sebelumnya dan berikutnya, jika LaTeX sudah terpasang, Anda dapat mencetak ekspresi simbolik menggunakan LaTeX. Jika tidak, perintah berikut akan menghasilkan pesan kesalahan.

Untuk mencetak ekspresi simbolik dengan LaTeX, gunakan $ di depan & (atau Anda dapat menghilangkan &) sebelum perintah. Jangan jalankan perintah Maxima dengan $ jika LaTeX tidak terpasang.

```
>$(3+x)/(x^2+1)
```

$$\frac{x+3}{x^2+1}$$

Ekspresi simbolik diuraikan (parsed) oleh Euler. Jika Anda memerlukan sintaks kompleks dalam satu ekspresi, Anda dapat menempatkan ekspresi tersebut di dalam "...". Menggunakan lebih dari satu ekspresi sederhana memang memungkinkan, tetapi sangat tidak disarankan.

```
>&"v := 5; v^2"
```

25

Sebagai informasi tambahan, ekspresi simbolik dapat digunakan dalam program, tetapi harus ditempatkan dalam tanda kutip. Selain itu, akan jauh lebih efektif jika memanggil Maxima pada waktu kompilasi, jika memungkinkan.

```
>$&expand((1+x)^4), $&factor(diff(%,x)) // diff: turunan, factor: faktor
```

$$4\,(x+1)^3$$

Sekali lagi, % merujuk pada hasil sebelumnya.

Untuk mempermudah, kita menyimpan solusi ke dalam variabel simbolik. Variabel simbolik didefinisikan dengan "&=".

```
>fx &= (x+1)/(x^4+1); $&fx
```

$$\frac{x+1}{x^4+1}$$

Ekspresi simbolik dapat digunakan di dalam ekspresi simbolik lainnya.

```
>$&factor(diff(fx,x))
```

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

Input langsung untuk perintah Maxima juga tersedia. Mulailah baris perintah dengan ::. Sintaks Maxima ini sudah diadaptasi agar sesuai dengan sintaks EMT (disebut compatibility mode).

```
>$&factor(20!)
```

$$2432902008176640000$$

```
>::: factor(10!)
```

```
  8  4  2
2  3  5  7
```

```
>:: factor(20!)
```

```
  18  8  4  2
2   3  5  7  11 13 17 19
```

Jika Anda adalah seorang ahli dalam Maxima, Anda mungkin ingin menggunakan sintaks asli dari Maxima. Anda dapat melakukannya dengan menggunakan ":::".

```
>::: av:g$ av^2;
```

```
  2
g
```

```
>fx &= x^3*exp(x), $fx
```

```
  3   x
x   E
```

$$x^3\,e^x$$

Variabel semacam itu dapat digunakan dalam ekspresi simbolik lainnya. Perlu dicatat bahwa pada perintah berikut, sisi kanan dari &= dievaluasi terlebih dahulu sebelum dilakukan penugasan (assignment) ke Fx.

```
>&(fx with x=5), $%, &float(%)
```

$$125\ E^5$$

$$125\,e^5$$

$$18551.64488782208$$

```
>fx(5)
```

```
  18551.6448878
```

Untuk mengevaluasi sebuah ekspresi dengan nilai tertentu dari variabel-variabelnya, Anda dapat menggunakan operator with.

Baris perintah berikut juga menunjukkan bahwa Maxima dapat mengevaluasi suatu ekspresi secara numerik dengan menggunakan fungsi float().

```
>&(fx with x=10)-(fx with x=5), &float(%)
```

$$1000\ E^{10} - 125\ E^5$$

$$2.20079141499189e+7$$

```
>$factor(diff(fx,x,2))
```

$$x\left(x^2+6\,x+6\right)e^x$$

Untuk mendapatkan kode LaTeX dari suatu ekspresi, Anda dapat menggunakan perintah tex.

```
>tex(fx)
```

```
  x^3\,e^{x}
```

Ekspresi simbolik dapat dievaluasi sama seperti ekspresi numerik.

```
> $fx(0.5)
```

$$\left(x^3\, e^x\right)(0.5)$$

ketika ditamba $ terdapat perbedaan makna dengan maksud perhitungan

```
>fx(0.5)
```

```
  0.206090158838
```

Dalam ekspresi simbolik, hal ini tidak dapat dilakukan karena Maxima tidak mendukungnya. Sebagai gantinya, gunakan sintaks "with" (bentuk yang lebih sederhana dari perintah at(...) pada Maxima).

```
>$&fx with x=1/2
```

$$\frac{\sqrt{e}}{8}$$

Penugasan (assignment) juga dapat dilakukan secara simbolik.

```
>$&fx with x=1+t
```

$$(t+1)^3\ e^{t+1}$$

Perintah solve menyelesaikan ekspresi simbolik terhadap suatu variabel di Maxima. Hasilnya berupa sebuah vektor solusi.

```
>$&solve(x^2+x=4,x)
```

$$\left[x=\frac{-\sqrt{17}-1}{2}, x=\frac{\sqrt{17}-1}{2}\right]$$

Bandingkan dengan perintah numerik "solve" di Euler, yang membutuhkan nilai awal, dan secara opsional juga nilai target.

```
>solve("x^2+x",1,y=4)
```

```
  1.56155281281
```

Nilai numerik dari solusi simbolik dapat dihitung dengan mengevaluasi hasil simbolik tersebut. Euler akan membaca hasil dalam bentuk penugasan seperti x = etc.
Jika Anda tidak memerlukan hasil numerik itu untuk perhitungan lebih lanjut, Anda juga bisa membiarkan Maxima yang menghitung nilai numeriknya secara langsung.

```
>sol &= solve(x^2+2*x=4,x); $&sol, sol(), $&float(sol)
```

$$\left[x=-\sqrt{5}-1,x=\sqrt{5}-1\right]$$

```
[-3.23606797749979,  1.23606797749979]
```

$$[x=-3.23606797749979,x=1.23606797749979]$$

Untuk mendapatkan solusi simbolik tertentu, kita dapat menggunakan "with" dan sebuah indeks.

```
>$&solve(x^2+x=1,x), x2 &= x with %[2]; $&x2
```

$$\frac{\sqrt{5}-1}{2}$$

$$\frac{\sqrt{5}-1}{2}$$

Untuk menyelesaikan suatu sistem persamaan, gunakan vektor persamaan. Hasilnya berupa vektor solusi.

```
>sol &= solve([x+y=3,x^2+y^2=5],[x,y]); $&sol, $&x*y with sol[1]
```

$$2$$

Ekspresi simbolik dapat memiliki flag, yang menunjukkan perlakuan khusus dalam Maxima.
Beberapa flag dapat digunakan sebagai perintah juga, sementara yang lain tidak bisa.
Flag ditambahkan dengan tanda "|", yang merupakan bentuk lebih sederhana dari ev(...,flags).

```
>$& diff((x^3-1)/(x+1),x) //turunan bentuk pecahan
```

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

```
>$& diff((x^3-1)/(x+1),x) | ratsimp //menyederhanakan pecahan
```

$$\frac{2\,x^3+3\,x^2+1}{x^2+2\,x+1}$$

```
>$&factor(%)
```

$$\frac{2\,x^3+3\,x^2+1}{\left(x+1\right)^2}$$

Fungsi

Dalam EMT, fungsi adalah program yang didefinisikan dengan perintah "function". Fungsi dapat berupa satu baris atau multibaris.
ebuah fungsi satu baris bisa bersifat numerik atau simbolik.
ungsi satu baris numerik didefinisikan dengan ":=".

```
>function f(x) := x*sqrt(x^2+1)
```

Untuk memberikan gambaran, berikut ditunjukkan semua kemungkinan definisi untuk fungsi satu baris. Sebuah fungsi dapat dievaluasi sama seperti fungsi bawaan (built-in) di Euler.

```
>f(2)
```

```
  4.472135955
```

Fungsi ini juga akan bekerja untuk vektor, dengan mengikuti aturan bahasa matriks di Euler, karena ekspresi yang digunakan dalam fungsi tersebut sudah tervektorisasi.

```
>f(0:0.1:1)
```

```
  [0,  0.1004987562112089,  0.2039607805437114,  0.3132091952673166,
  0.4308131845707604,  0.5590169943749475,  0.699714227381436,
  0.8544588931013591,  1.024499877989256,  1.210826164236634,
  1.414213562373095]
```

Fungsi dapat digambarkan (dipetakan). Berbeda dengan ekspresi simbolik atau numerik, pada fungsi kita hanya perlu memberikan nama fungsi.
Namun, tidak seperti ekspresi simbolik atau numerik, nama fungsi harus diberikan dalam bentuk string.

```
>solve("f",1,y=1)
```

```
  0.786151377757
```

Secara default, jika Anda ingin menimpa (overwrite) fungsi bawaan, Anda harus menambahkan kata kunci overwrite.
Menimpa fungsi bawaan cukup berisiko karena dapat menimbulkan masalah pada fungsi lain yang bergantung pada fungsi tersebut.
Namun, Anda tetap bisa memanggil fungsi bawaan tersebut dengan menambahkan awalan _..., jika fungsi tersebut berasal dari inti (core) Euler.

```
>function overwrite sin (x) := _sin(x°) // redine sine in degrees
>sin(45)
```

```
  0.707106781187
```

Sebaiknya kita menghapus pendefinisian ulang fungsi sin

```
>forget sin; sin(pi/4)
```

```
  0.707106781187
```

Parameter Bawaan

Fungsi numerik dapat memiliki parameter bawaan.

```
>function f(x,a=1) := a*x^2
```

Menghilangkan parameter ini akan menggunakan nilai bawaan.

```
>f(4)
```

```
  16
```

Mengaturnya akan menimpa nilai bawaan.

```
>f(4,5) //5*4^2
```

```
  80
```

Parameter yang ditetapkan juga akan menimpa nilai tersebut.
Hal ini digunakan oleh banyak fungsi Euler seperti plot2d dan plot3d.

```
>f(4,a=1)
```

```
  16
```

Jika sebuah variabel bukan merupakan parameter, maka variabel tersebut harus bersifat global.
Fungsi satu baris (one-line functions) dapat melihat variabel global.

```
>function f(x) := a*x^2
>a=6; f(2)
```

```
  24
```

Namun, sebuah parameter yang ditetapkan (assigned parameter) akan menimpa nilai global.

Jika argumen tidak ada dalam daftar parameter yang telah didefinisikan sebelumnya, maka argumen tersebut harus dideklarasikan dengan ":=".

```
>f(2,a:=5) //5*2^2
```

```
  20
```

Fungsi simbolik didefinisikan dengan &=. Fungsi ini didefinisikan baik di Euler maupun di Maxima, sehingga dapat bekerja di kedua lingkungan tersebut. Ekspresi yang digunakan untuk mendefinisikan fungsi akan diproses terlebih dahulu melalui Maxima sebelum definisi tersebut dibuat.

```
>function g(x) &= x^3-x*exp(-x); $&g(x)
```

$$x^3 - x\,e^{-x}$$

Fungsi simbolik dapat digunakan di dalam ekspresi simbolik.

```
>$&diff(g(x),x), $&% with x=4/3
```

$$\frac{e^{-\frac{4}{3}}}{3} + \frac{16}{3}$$

$$\frac{e^{-\frac{4}{3}}}{3}+\frac{16}{3}$$

Fungsi simbolik juga dapat digunakan dalam ekspresi numerik. Tentu saja, ini hanya akan berhasil jika EMT dapat menginterpretasikan segala sesuatu di dalam fungsi tersebut.

```
>g(5+g(1))
```

```
  178.635099908
```

Fungsi simbolik dapat digunakan untuk mendefinisikan fungsi simbolik lain atau ekspresi lain.

```
>function G(x) &= factor(integrate(g(x),x)); $&G(c) // integrate: mengintegralkan
```

$$\frac{e^{-c}\left(c^4\,e^c + 4\,c + 4\right)}{4}$$

```
>solve(&g(x),0.5)
```

```
  0.703467422498
```

Hal berikut juga berfungsi, karena Euler akan menggunakan ekspresi simbolik dalam fungsi g, jika tidak menemukan variabel simbolik g, dan jika terdapat fungsi simbolik g.

```
>solve(&g,0.5)
```

```
  0.703467422498
```

```
>function P(x,n) &= (2*x-1)^n; $&P(x,n)
```

$$(2\,x-1)^n$$

```
>function Q(x,n) &= (x+2)^n; $&Q(x,n)
```

$$(x+2)^n$$

```
>$&P(x,4), $&expand(%)
```

$$16\,x^4-32\,x^3+24\,x^2-8\,x+1$$

```
>P(3,4)
```

```
  625
```

```
>$&P(x,4)+ Q(x,3), $&expand(%)
```

$$16\,x^4-31\,x^3+30\,x^2+4\,x+9$$

```
>$&P(x,4)-Q(x,3), $&expand(%), $&factor(%)
```

$$16\,x^4-33\,x^3+18\,x^2-20\,x-7$$

```
>$&P(x,4)*Q(x,3), $&expand(%), $&factor(%)
```

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

```
>$&P(x,4)/Q(x,1), $&expand(%), $&factor(%)
```

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

$$\frac{16\,x^4}{x+2}-\frac{32\,x^3}{x+2}+\frac{24\,x^2}{x+2}-\frac{8\,x}{x+2}+\frac{1}{x+2}$$

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

```
>function f(x) &= x^3-x; $&f(x)
```

$$x^3-x$$

Dengan &=, fungsi tersebut bersifat simbolik, dan dapat digunakan dalam ekspresi simbolik lainnya.

```
>$&integrate(f(x),x)
```

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

Dengan := fungsi tersebut bersifat numerik. Contoh yang baik adalah integral tentu seperti

$$f(x)=\int_1^x t^t\,dt,$$

yang tidak dapat dievaluasi secara simbolik.
Jika kita mendefinisikan ulang fungsi dengan kata kunci map, maka fungsi tersebut dapat digunakan untuk vektor x. Secara internal, fungsi akan dipanggil untuk setiap nilai x sekali, lalu hasilnya disimpan dalam sebuah vektor.

```
>function map f(x) := integrate("x^x",1,x)
>f(0:0.5:2)
```

```
  [-0.7834305107120823,  -0.4108156482543905,  0,  0.6768632787990813,
  2.050446234534731]
```

Fungsi dapat memiliki nilai bawaan (default) untuk parameternya.

```
>function mylog (x,base=10) := ln(x)/ln(base);
```

Sekarang fungsi tersebut dapat dipanggil dengan atau tanpa parameter "base".

```
>mylog(100), mylog(2^6.7,2)
```

```
  2
  6.7
```

Selain itu, dimungkinkan juga untuk menggunakan parameter yang ditetapkan (assigned parameters).

```
>mylog(E^2,base=E)
```

```
  2
```

Sering kali, kita ingin menggunakan fungsi untuk vektor di satu tempat, dan untuk elemen individual di tempat lain. Hal ini dimungkinkan dengan parameter vektor.

```
>function f([a,b]) &= a^2+b^2-a*b+b; $&f(a,b), $&f(x,y)
```

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

Fungsi simbolik seperti itu dapat digunakan untuk variabel simbolik.
Namun, fungsi tersebut juga dapat digunakan untuk vektor numerik.

```
>v=[3,4]; f(v)
```

```
  17
```

Ada juga fungsi yang murni simbolik, yang tidak dapat digunakan secara numerik.

```
>function lapl(expr,x,y) &&= diff(expr,x,2)+diff(expr,y,2)//turunan parsial kedua
```

```
diff(expr, y, 2) + diff(expr, x, 2)
```

```
>$&realpart((x+I*y)^4), $&lapl(%,x,y)
```

$$0$$

Namun tentu saja, fungsi tersebut dapat digunakan dalam ekspresi simbolik atau dalam definisi fungsi simbolik.

```
>function f(x,y) &= factor(lapl((x+y^2)^5,x,y)); $&f(x,y)
```

$$10\,\left(y^2+x\right)^3\,\left(9\,y^2+x+2\right)$$

Untuk merangkum:
-&= mendefinisikan fungsi simbolik,
-:= mendefinisikan fungsi numerik,
-
&&= mendefinisikan fungsi murni simbolik.

Menyelesaikan Ekspresi

Ekspresi dapat diselesaikan secara numerik maupun simbolik.
Untuk menyelesaikan ekspresi sederhana dengan satu variabel, kita dapat menggunakan fungsi solve(). Fungsi ini memerlukan nilai awal (start value) untuk memulai pencarian. Secara internal, solve() menggunakan metode secant.

```
>solve("x^2-2",1)
```

```
  1.41421356237
```

Ini juga berfungsi untuk ekspresi simbolik.
Ambil contoh fungsi berikut.

```
>$&solve(x^2=2,x)
```

$$\left[x=-\sqrt{2},x=\sqrt{2}\right]$$

```
>$&solve(x^2-2,x)
```

$$\left[x=-\sqrt{2},x=\sqrt{2}\right]$$

```
>$&solve(a*x^2+b*x+c=0,x)
```

$$\left[x=\frac{-\sqrt{b^2-4\,a\,c}-b}{2\,a},x=\frac{\sqrt{b^2-4\,a\,c}-b}{2\,a}\right]$$

```
>$&solve([a*x+b*y=c,d*x+e*y=f],[x,y])
```

$$\left[\left[x=-\frac{c\,e}{b\,(d-5)-a\,e},y=\frac{c\,(d-5)}{b\,(d-5)-a\,e}\right]\right]$$

```
>  px &= 4*x^8+x^7-x^4-x; $&px
```

$$4\,x^8+x^7-x^4-x$$

Sekarang kita mencari titik di mana polinomial bernilai 2.
Dalam solve(), nilai target bawaan y=0 dapat diubah dengan variabel terikat (assigned variable).
Kita gunakan y=2 dan memeriksa dengan mengevaluasi polinomial pada hasil sebelumnya.

```
>solve(px,1,y=2), px(%)
```

```
  0.966715594851
  2
```

Menyelesaikan suatu ekspresi simbolik dalam bentuk simbolik akan mengembalikan daftar solusi.
Untuk itu, kita menggunakan penyelesai simbolik solve() yang disediakan oleh Maxima.

```
>sol &= solve(x^2-x-1,x); $&sol
```

$$\left[x=\frac{1-\sqrt{5}}{2},x=\frac{\sqrt{5}+1}{2}\right]$$

Cara termudah untuk mendapatkan nilai numeriknya adalah dengan mengevaluasi solusi secara numerik, sama seperti mengevaluasi sebuah ekspresi.

```
>longest sol()
```

```
      -0.6180339887498949       1.618033988749895
```

Untuk menggunakan solusi secara simbolik dalam ekspresi lain, cara termudah adalah dengan menggunakan "with".

```
>$&x^2 with sol[1], $&expand(x^2-x-1 with sol[2])
```

$$0$$

Menyelesaikan sistem persamaan secara simbolik dapat dilakukan dengan vektor persamaan dan penyelesai simbolik solve(). Hasilnya berupa daftar yang berisi daftar persamaan.

```
>$&solve([x+y=2,x^3+2*y+x=4],[x,y])
```

$$[[x=-1,y=3],[x=1,y=1],[x=0,y=2]]$$

Fungsi f() dapat melihat variabel global.
Namun sering kali kita ingin menggunakan parameter lokal.
Contoh:

$$a^x-x^a=0.1$$

dengan a = 3.

```
>function f(x,a) := x^a-a^x;
```

Salah satu cara untuk melewatkan parameter tambahan ke fungsi f() adalah dengan menggunakan daftar (list) yang berisi nama fungsi dan parameternya (cara lainnya adalah menggunakan parameter titik koma).

```
>solve({{"f",3}},2,y=0.1)
```

```
  2.54116291558
```

Ini juga berfungsi untuk ekspresi. Namun, dalam hal ini, harus digunakan elemen daftar (list) yang bernama. (Lebih lanjut tentang daftar dibahas di tutorial mengenai sintaks EMT).

```
>solve({{"x^a-a^x",a=3}},2,y=0.1)
```

```
  2.54116291558
```

Menyelesaikan Pertidaksamaan

Untuk menyelesaikan pertidaksamaan, EMT tidak akan dapat melakukannya, melainkan dengan bantuan Maxima, artinya secara eksak (simbolik). Perintah Maxima yang digunakan adalah fourier_elim(), yang harus dipanggil dengan perintah "load(fourier_elim)" terlebih dahulu.

```
>&load(fourier_elim)
```

```
          C:/Program Files/Euler x64/maxima/share/maxima/5.35.1/share/f\
  ourier_elim/fourier_elim.lisp
```

```
>$&fourier_elim([x^2 - 1>0],[x]) // x^2-1 > 0
```

$$[1 < x] \vee [x < -1]$$

```
>$&fourier_elim([x^2 - 1<0],[x]) // x^2-1 < 0
```

$$[-1 < x, x < 1]$$

```
>$&fourier_elim([x^2 - 1 # 0],[x]) // x^-1 <> 0
```

$$[-1 < x, x < 1] \vee [1 < x] \vee [x < -1]$$

```
>$&fourier_elim([x # 6],[x])
```

$$[x < 6] \vee [6 < x]$$

```
>$&fourier_elim([x < 1, x > 1],[x]) // tidak memiliki penyelesaian
```

$$emptyset$$

```
>$&fourier_elim([minf < x, x < inf],[x]) // solusinya R
```

$$universalset$$

```
>$&fourier_elim([x^3 - 1 > 0],[x])
```

$$\left[1 < x, x^2 + x + 1 > 0\right] \vee \left[x < 1, -x^2 - x - 1 > 0\right]$$

```
>$&fourier_elim([cos(x) < 1/2],[x]) // ??? gagal
```

$$[1 - 2\cos x > 0]$$

```
>$&fourier_elim([y-x < 5, x - y < 7, 10 < y],[x,y]) // sistem pertidaksamaan
```

$$[y - 5 < x, x < y + 7, 10 < y]$$

```
>$&fourier_elim([y-x < 5, x - y < 7, 10 < y],[y,x])
```

$$[max\,(10, x - 7) < y, y < x + 5, 5 < x]$$

```
>$&fourier_elim((x + y < 5) and (x - y >8),[x,y])
```

$$\left[y + 8 < x, x < 5 - y, y < -\frac{3}{2}\right]$$

```
>$&fourier_elim(((x + y < 5) and x < 1) or  (x - y >8),[x,y])
```

$$[y + 8 < x] \vee [x < min\,(1, 5 - y)]$$

```
>&fourier_elim([max(x,y) > 6, x # 8, abs(y-1) > 12],[x,y])
```

```
          [6 < x, x < 8, y < - 11] or [8 < x, y < - 11]
   or [x < 8, 13 < y] or [x = y, 13 < y] or [8 < x, x < y, 13 < y]
   or [y < x, 13 < y]
```

```
>$&fourier_elim([(x+6)/(x-9) <= 6],[x])
```

$$[x = 12] \vee [12 < x] \vee [x < 9]$$

Bahasa Matriks

Dokumentasi inti EMT memuat pembahasan bahasa matriks Euler secara rinci.

Vektor dan matriks dimasukkan menggunakan tanda kurung siku [ ], dengan elemen dipisahkan oleh koma, dan baris dipisahkan oleh titik koma (;).

```
>A=[1,2;3,4]
```

```
                1                2
                3                4
```

Perkalian matriks ditandai dengan titik.

```
>b=[3;4]
```

```
                3
                4
```

```
>b' // transpose b
```

```
  [3,  4]
```

```
>inv(A) //inverse A
```

```
               -2                1
              1.5     -0.4999999999999999
```

```
>A.b //perkalian matriks
```

```
                        11
                        25
```

```
>A.inv(A)
```

```
       0.9999999999999998                         0
                        0                         1
```

Poin utama dari bahasa matriks adalah bahwa semua fungsi dan operator bekerja elemen per elemen.

```
>A.A
```

```
                         7                        10
                        15                        22
```

```
>A^2 //perpangkatan elemen2 A
```

```
                         1                         4
                         9                        16
```

```
>A.A.A
```

```
                        37                        54
                        81                       118
```

```
>power(A,3) //perpangkatan matriks
```

```
                        37                        54
                        81                       118
```

```
>A/A //pembagian elemen-elemen matriks yang seletak
```

```
                         1                         1
                         1                         1
```

```
>A/b //pembagian elemen2 A oleh elemen2 b kolom demi kolom (karena b vektor kolom)
```

```
       0.3333333333333333       0.6666666666666666
                     0.75                         1
```

```
>A\b // hasilkali invers A dan b, A^(-1)b
```

```
                       -2
                      2.5
```

```
>inv(A).b
```

```
       -1.999999999999999
        2.499999999999999
```

```
>A\A   //A^(-1)A
```

```
                        1                        0
                        0                        1
```

```
>inv(A).A
```

```
                        1                        0
                        0      0.9999999999999998
```

```
>A*A //perkalin elemen-elemen matriks seletak
```

```
                        1                        4
                        9                       16
```

Ini bukan perkalian matriks, melainkan perkalian elemen per elemen. Hal yang sama berlaku untuk vektor.

```
>b^2 // perpangkatan elemen-elemen matriks/vektor
```

```
                        9
                       16
```

Jika salah satu operand adalah vektor atau skalar, ia akan diperluas secara alami sesuai aturan matriks.

```
>2*A
```

```
                        2                        4
                        6                        8
```

Misalnya, jika operand adalah vektor kolom, elemennya akan diterapkan ke semua baris matriks A.

```
>[1,2]*A
```

```
                        1                        4
                        3                        8
```

Jika itu adalah vektor baris, elemennya akan diterapkan ke semua kolom matriks A.

```
>A*[2,3]
```

```
                    2                    6
                    6                   12
```

Kita dapat membayangkan perkalian ini seolah-olah vektor baris v diduplikasi untuk membentuk matriks dengan ukuran yang sama seperti A.

```
>dup([1,2],2) // dup: menduplikasi/menggandakan vektor [1,2] sebanyak 2 kali (baris)
```

```
                    1                    2
                    1                    2
```

```
>A*dup([1,2],2)
```

```
                    1                    4
                    3                    8
```

Hal ini juga berlaku untuk dua vektor di mana satu adalah vektor baris dan yang lain adalah vektor kolom.
Kita ingin menghitung i*j untuk i,j dari 1 hingga 5. Triknya adalah dengan mengalikan 1:5 dengan transpose-nya.
Bahasa matriks Euler secara otomatis akan menghasilkan tabel nilai.

```
>(1:5)*(1:5)' // hasilkali elemen-elemen vektor baris dan vektor kolom
```

```
  Real 5 x 5 matrix

                    1      ...
                    2      ...
                    3      ...
                    4      ...
                    5      ...
```

Sekali lagi, ingat bahwa ini bukan perkalian matriks!

```
>(1:5).(1:5)' // hasilkali vektor baris dan vektor kolom
```

```
  55
```

```
>sum((1:5)*(1:5)) // sama hasilnya
```

```
  55
```

Bahkan operator seperti < atau == bekerja dengan cara yang sama (elemen per elemen).

```
>(1:10)<6 // menguji elemen-elemen yang kurang dari 6
```

```
  [1,  1,  1,  1,  1,  0,  0,  0,  0,  0]
```

Misalnya, kita dapat menghitung jumlah elemen yang memenuhi kondisi tertentu menggunakan fungsi sum().

```
>sum((1:10)<6) // banyak elemen yang kurang dari 6
```

```
  5
```

Euler memiliki operator perbandingan, seperti ==, yang digunakan untuk memeriksa kesetaraan.
Hasilnya adalah sebuah vektor berisi 0 dan 1, di mana 1 berarti benar (true).

```
>t=(1:10)^2; t==25 //menguji elemen2 t yang sama dengan 25 (hanya ada 1)
```

```
  [0,  0,  0,  0,  1,  0,  0,  0,  0,  0]
```

Dari vektor seperti itu, fungsi nonzeros memilih elemen yang tidak nol.
Dalam contoh ini, kita mendapatkan indeks semua elemen yang lebih besar dari 50.

```
>nonzeros(t>50) //indeks elemen2 t yang lebih besar daripada 50
```

```
  [8,  9,  10]
```

Tentu saja, kita dapat menggunakan vektor indeks ini untuk mendapatkan nilai yang sesuai di t.

```
>t[nonzeros(t>50)] //elemen2 t yang lebih besar daripada 50
```

```
  [64,  81,  100]
```

Sebagai contoh, mari kita cari semua kuadrat dari bilangan 1 hingga 1000 yang memenuhi kondisi: sisa bagi 5 modulo 11 dan sisa bagi 3 modulo 13.

```
>t=1:1000; nonzeros(mod(t^2,11)==5 && mod(t^2,13)==3)
```

```
  [4,  48,  95,  139,  147,  191,  238,  282,  290,  334,  381,  425,
  433,  477,  524,  568,  576,  620,  667,  711,  719,  763,  810,  854,
  862,  906,  953,  997]
```

EMT tidak sepenuhnya efisien untuk perhitungan bilangan bulat, karena secara internal menggunakan floating point presisi ganda. Namun, fitur ini seringkali sangat berguna.
Kita dapat memeriksa bilangan prima. Misalnya, mari kita cari berapa banyak kuadrat ditambah 1 yang merupakan bilangan prima.

```
>t=1:1000; length(nonzeros(isprime(t^2+1)))
```

```
  112
```

Fungsi nonzeros() hanya bekerja untuk vektor.
Untuk matriks, digunakan fungsi mnonzeros().

```
>seed(2); A=random(3,4)
```

```
  Real 3 x 4 matrix

       0.7657608464777804     ...
       0.1367295433869637     ...
       0.5481376776433097     ...
```

It returns the indices of the elements, which are not zeros.

```
>k=mnonzeros(A<0.4) //indeks elemen2 A yang kurang dari 0,4
```

```
                1              4
                2              1
                2              2
                3              2
```

Fungsi ini mengembalikan indeks elemen-elemen yang tidak nol.

```
>mset(A,k,0) //mengganti elemen2 suatu matriks pada indeks tertentu
```

```
       0.765761       0.401188       0.406347              0
              0              0       0.495975       0.952814
       0.548138              0       0.444255       0.539246
```

Fungsi mset() juga dapat mengisi elemen pada indeks tertentu dengan nilai-nilai dari matriks lain.

```
>mset(A,k,-random(size(A)))
```

```
       0.765761       0.401188       0.406347      -0.126917
      -0.122404      -0.691673       0.495975       0.952814
       0.548138      -0.483902       0.444255       0.539246
```

Dan dimungkinkan untuk mengambil elemen-elemen tersebut dalam bentuk vektor.

```
>mget(A,k)
```

```
  [0.267829,  0.13673,  0.390567,  0.006085]
```

Fungsi lain yang berguna adalah extrema, yang mengembalikan nilai minimal dan maksimal di setiap baris matriks beserta posisinya.

```
>ex=extrema(A)
```

```
       0.267829              4       0.765761              1
        0.13673              1       0.952814              4
       0.006085              2       0.548138              1
```

Kita dapat menggunakan fungsi ini untuk mengambil nilai maksimal di setiap baris.

```
>ex[,3]'
```

```
  [0.765761,  0.952814,  0.548138]
```

Ini tentu saja sama dengan fungsi max().

```
>max(A)'
```

```
  [0.765761,  0.952814,  0.548138]
```

Namun dengan mget(), kita dapat mengambil indeks dan menggunakan informasi ini untuk mengambil elemen pada posisi yang sama dari matriks lain.

```
>j=(1:rows(A)),|ex[4], mget(-A,j)
```

```
  [1,  2,  3]
```

Fungsi Matriks Lain (Membangun Matriks)

Untuk membangun matriks, kita dapat menumpuk satu matriks di atas matriks lain.
Jika kedua matriks tidak memiliki jumlah kolom yang sama, matriks yang lebih pendek akan diisi dengan 0.

```
>v=1:3; v_v
```

```
             1             2             3
             1             2             3
```

Demikian pula, kita dapat menempelkan matriks secara berdampingan (side by side), asalkan kedua matriks memiliki jumlah baris yang sama.

```
>A=random(3,4); A|v'
```

```
      0.032444     0.0534171      0.595713      0.564454             1
       0.83916      0.175552      0.396988       0.83514             2
     0.0257573      0.658585      0.629832      0.770895             3
```

Jika kedua matriks tidak memiliki jumlah baris yang sama, matriks yang lebih pendek akan diisi dengan 0.

Ada pengecualian untuk aturan ini: bilangan riil yang ditempelkan pada matriks akan digunakan sebagai kolom yang diisi dengan bilangan riil tersebut.

```
>A|1
```

```
      0.032444     0.0534171      0.595713      0.564454             1
       0.83916      0.175552      0.396988       0.83514             1
     0.0257573      0.658585      0.629832      0.770895             1
```

Dimungkinkan untuk membuat matriks dari vektor baris dan vektor kolom.

```
>[v;v]
```

```
                1               2               3
                1               2               3
```

```
>[v',v']
```

```
                1               1
                2               2
                3               3
```

Tujuan utama dari ini adalah untuk menginterpretasikan vektor ekspresi sebagai vektor kolom.

```
>"[x,x^2]"(v')
```

```
                1               1
                2               4
                3               9
```

Untuk mendapatkan ukuran matriks A, kita dapat menggunakan fungsi-fungsi berikut.

```
>C=zeros(2,4); rows(C), cols(C), size(C), length(C)
```

```
  2
  4
  [2,  4]
  4
```

Untuk vektor, digunakan fungsi length().

```
>length(2:10)
```

```
  9
```

Terdapat banyak fungsi lain yang dapat menghasilkan matriks.

```
>ones(2,2)
```

```
                1               1
                1               1
```

Ini juga dapat digunakan dengan satu parameter.Untuk mendapatkan vektor dengan angka selain 1, gunakan cara berikut.

```
>ones(5)*6
```

```
  [6,  6,  6,  6,  6]
```

Selain itu, matriks bilangan acak dapat dibuat dengan random (distribusi uniform) atau normal (distribusi Gauss).

```
>random(2,2)
```

```
        0.66566      0.831835
          0.977      0.544258
```

Berikut adalah fungsi lain yang berguna, yaitu mengubah susunan elemen sebuah matriks menjadi matriks lain.

```
>redim(1:9,3,3) // menyusun elemen2 1, 2, 3, ..., 9 ke bentuk matriks 3x3
```

```
             1             2             3
             4             5             6
             7             8             9
```

Dengan fungsi berikut, kita dapat menggunakan ini bersama fungsi dup untuk membuat fungsi rep(), yang mengulang sebuah vektor sebanyak n kali.

```
>function rep(v,n) := redim(dup(v,n),1,n*cols(v))
```

Mari kita uji coba.

```
>rep(1:3,5)
```

```
  [1,  2,  3,  1,  2,  3,  1,  2,  3,  1,  2,  3,  1,  2,  3]
```

Fungsi multdup() menggandakan elemen-elemen dari sebuah vektor.

```
>multdup(1:3,5), multdup(1:3,[2,3,2])
```

```
  [1,  1,  1,  1,  1,  2,  2,  2,  2,  2,  3,  3,  3,  3,  3]
  [1,  1,  2,  2,  2,  3,  3]
```

Fungsi flipx() dan flipy() membalik urutan baris atau kolom dari sebuah matriks.
Artinya, flipx() membalik secara horizontal.

```
>flipx(1:5) //membalik elemen2 vektor baris
```

```
  [5,  4,  3,  2,  1]
```

Untuk rotasi, Euler memiliki fungsi rotleft() dan rotright().

```
>rotleft(1:5) // memutar elemen2 vektor baris
```

```
  [2,  3,  4,  5,  1]
```

```
>rotright(1:5)
```

```
  [5,  1,  2,  3,  4]
```

Fungsi khusus adalah drop(v,i), yang menghapus elemen-elemen dengan indeks i dari vektor v.

```
>drop(10:20,3)
```

```
  [10,  11,  13,  14,  15,  16,  17,  18,  19,  20]
```

Perlu dicatat bahwa vektor i dalam drop(v,i) merujuk pada indeks elemen di v, bukan nilai elemen.
Jika ingin menghapus elemen tertentu, Anda harus menemukan elemen tersebut terlebih dahulu.
Fungsi indexof(v,x) dapat digunakan untuk menemukan elemen x dalam vektor v yang sudah diurutkan.

```
>v=primes(50), i=indexof(v,10:20), drop(v,i)
```

```
  [2,  3,  5,  7,  11,  13,  17,  19,  23,  29,  31,  37,  41,  43,  47]
  [0,  5,  0,  6,  0,  0,  0,  7,  0,  8,  0]
  [2,  3,  5,  7,  23,  29,  31,  37,  41,  43,  47]
```

Seperti yang Anda lihat, tidak masalah jika menyertakan indeks di luar jangkauan (misalnya 0), indeks ganda, atau indeks yang tidak berurutan.

```
>drop(1:10,shuffle([0,0,5,5,7,12,12]))
```

```
  [1,  2,  3,  4,  6,  8,  9,  10]
```

Terdapat beberapa fungsi khusus untuk mengatur diagonal atau membuat matriks diagonal.
Kita mulai dengan matriks identitas.

```
>A=id(5) // matriks identitas 5x5
```

```
1  0  0  0  0
0  1  0  0  0
0  0  1  0  0
0  0  0  1  0
0  0  0  0  1
```

Kemudian, kita mengatur diagonal bawah (-1) menjadi 1:4.

```
>setdiag(A,-1,1:4) //mengganti diagonal di bawah diagonal utama
```

```
1  0  0  0  0
1  1  0  0  0
0  2  1  0  0
0  0  3  1  0
0  0  0  4  1
```

Perlu dicatat bahwa kita tidak mengubah matriks A.
ungsi setdiag() menghasilkan matriks baru sebagai hasilnya.

Berikut adalah fungsi yang mengembalikan matriks tri-diagonal.

```
>function tridiag (n,a,b,c) := setdiag(setdiag(b*id(n),1,c),-1,a); ...
>tridiag(5,1,2,3)
```

```
     2     3     0     0     0
     1     2     3     0     0
     0     1     2     3     0
     0     0     1     2     3
     0     0     0     1     2
```

Diagonal dari sebuah matriks juga dapat diambil dari matriks itu sendiri.Untuk mendemonstrasikan ini, kita mengubah vektor 1:9 menjadi matriks 3x3.

```
>A=redim(1:9,3,3)
```

```
     1     2     3
     4     5     6
     7     8     9
```

Sekarang kita dapat mengambil diagonalnya.

```
>d=getdiag(A,0)
```

```
  [1,  5,  9]
```

Misalnya, kita dapat membagi matriks dengan diagonalnya.
Bahasa matriks secara otomatis memastikan bahwa vektor kolom d diterapkan ke matriks baris demi baris.

```
>fraction A/d'
```

```
     1     2     3
   4/5     1   6/5
   7/9   8/9     1
```

Vektorisasi

Hampir semua fungsi di Euler juga bekerja untuk input matriks dan vektor, selama hal ini masuk akal.
Misalnya, fungsi sqrt() menghitung akar kuadrat dari semua elemen pada vektor atau matriks.

```
>sqrt(1:3)
```

```
  [1,  1.41421,  1.73205]
```

Sehingga Anda dapat dengan mudah membuat tabel nilai.
Ini adalah salah satu cara untuk mem-plot fungsi (alternatifnya menggunakan ekspresi).

```
>x=1:0.01:5; y=log(x)/x^2; // terlalu panjang untuk ditampikan
```

Dengan ini dan operator kolon a:delta:b, vektor nilai fungsi dapat dibuat dengan mudah.

Dalam contoh berikut, kita membuat vektor nilai t[i] dengan jarak 0.1 dari -1 hingga 1.
Kemudian kita menghasilkan vektor nilai fungsi

$$s = t^3 - t$$

```
>t=-1:0.1:1; s=t^3-t
```

```
  [0,  0.171,  0.288,  0.357,  0.384,  0.375,  0.336,  0.273,  0.192,
  0.099,  0,  -0.099,  -0.192,  -0.273,  -0.336,  -0.375,  -0.384,
  -0.357,  -0.288,  -0.171,  0]
```

EMT memperluas operator untuk skalar, vektor, dan matriks dengan cara yang logis dan konsisten.

Misalnya, vektor kolom dikalikan dengan vektor baris akan diperluas menjadi matriks jika operator diterapkan.
Dalam contoh berikut, v' adalah vektor transpose (vektor kolom).

```
>shortest (1:5)*(1:5)'
```

```
       1       2       3       4       5
       2       4       6       8      10
       3       6       9      12      15
       4       8      12      16      20
       5      10      15      20      25
```

Perlu dicatat bahwa ini sangat berbeda dari perkalian matriks.
erkalian matriks dalam EMT dilambangkan dengan titik "."

```
>(1:5).(1:5)'
```

```
  55
```

Secara default, vektor baris dicetak dalam format yang ringkas.

```
>[1,2,3,4]
```

```
  [1,  2,  3,  4]
```

Untuk matriks, operator khusus . digunakan untuk perkalian matriks, dan A' digunakan untuk transpose.
Matriks 1x1 dapat digunakan seperti bilangan riil.

```
>v:=[1,2]; v.v', %^2
```

```
  5
  25
```

Untuk mentranspos matriks, kita menggunakan tanda apostrof (').

```
>v=1:4; v'
```

```
                1
                2
                3
                4
```

Sehingga kita dapat menghitung perkalian matriks A dengan vektor b.

```
>A=[1,2,3,4;5,6,7,8]; A.v'
```

```
               30
               70
```

Perlu dicatat bahwa v masih berupa vektor baris. Sehingga v'.v berbeda dengan v.v'.

```
>v'.v
```

```
                1                2                3                4
                2                4                6                8
                3                6                9               12
                4                8               12               16
```

v.v' menghitung norma kuadrat dari vektor baris v.
Hasilnya adalah vektor 1x1, yang dapat digunakan seperti bilangan riil.

```
>v.v'
```

```
  30
```

Terdapat juga fungsi norm (beserta banyak fungsi lain dari Aljabar Linear).

```
>norm(v)^2
```

```
  30
```

Operator dan fungsi mematuhi bahasa matriks Euler.

Berikut ringkasan aturannya:

-Fungsi yang diterapkan pada vektor atau matriks diberlakukan pada setiap elemen.

-Operator yang bekerja pada dua matriks dengan ukuran sama diberlakukan secara pasangan elemen.

-

Jika dua matriks memiliki dimensi berbeda, keduanya diperluas secara logis agar memiliki ukuran yang sama.

Misalnya, nilai skalar dikalikan dengan vektor akan mengalikan nilai tersebut dengan setiap elemen vektor.
Atau matriks dikalikan dengan vektor (dengan *, bukan .) akan memperluas vektor ke ukuran matriks dengan menggandakannya.

Berikut contoh sederhana dengan operator ^.

```
>[1,2,3]^2
```

```
  [1,  4,  9]
```

Berikut contoh yang lebih rumit.
Vektor baris dikalikan dengan vektor kolom akan memperluas keduanya dengan menggandakan elemen.

```
>v:=[1,2,3]; v*v'
```

```
                1               2               3
                2               4               6
                3               6               9
```

Perlu dicatat bahwa perkalian skalar menggunakan perkalian matriks (.), bukan *!

```
>v.v'
```

```
  14
```

Terdapat banyak fungsi untuk matriks. Berikut daftar singkatnya. Untuk informasi lebih lengkap, sebaiknya membaca dokumentasi fungsi-fungsi ini:

sum, prod: menghitung jumlah dan hasil perkalian elemen tiap baris

cumsum, cumprod: melakukan hal yang sama secara kumulatif

extrema: menghitung nilai ekstremal tiap baris

diag(A,i): mengembalikan diagonal ke-i

setdiag(A,i,v): mengatur diagonal ke-i dengan vektor v

id(n): matriks identitas

det(A): determinan

charpoly(A): polinomial karakteristik

eigenvalues(A): nilai eigen

```
>v*v, sum(v*v), cumsum(v*v)
```

```
  [1,  4,  9]
  14
  [1,  5,  14]
```

Operator : menghasilkan vektor baris dengan spasi sama, dengan opsi menentukan ukuran langkah (step size).

```
>1:4, 1:2:10
```

```
  [1,  2,  3,  4]
  [1,  3,  5,  7,  9]
```

Untuk menggabungkan matriks dan vektor, terdapat operator | dan _.

```
>[1,2,3]|[4,5], [1,2,3]_1
```

```
  [1,  2,  3,  4,  5]
                1              2              3
                1              1              1
```

Elemen-elemen dari matriks direferensikan dengan A[i,j].

```
>A:=[1,2,3;4,5,6;7,8,9]; A[2,3]
```

```
  6
```

Untuk vektor baris atau kolom, v[i] adalah elemen ke-i dari vektor.
Untuk matriks, v[i] mengembalikan seluruh baris ke-i dari matriks.

```
>v:=[2,4,6,8]; v[3], A[3]
```

```
  6
  [7,  8,  9]
```

Indeks juga dapat berupa vektor baris dari indeks.
Simbol : menunjukkan semua indeks.

```
>v[1:2], A[:,2]
```

```
  [2,  4]
                2
                5
                8
```

Bentuk singkat dari : adalah dengan menghilangkan indeks sepenuhnya.

```
>A[,2:3]
```

```
                2              3
                5              6
                8              9
```

Untuk keperluan vektorisasi, elemen-elemen matriks dapat diakses seolah-olah mereka adalah vektor.

```
>A{4}
```

```
  4
```

Matriks juga dapat diubah menjadi vektor datar (flattened) menggunakan fungsi redim().
Hal ini diimplementasikan dalam fungsi flatten().

```
>redim(A,1,prod(size(A))), flatten(A)
```

```
  [1,  2,  3,  4,  5,  6,  7,  8,  9]
  [1,  2,  3,  4,  5,  6,  7,  8,  9]
```

Untuk menggunakan matriks sebagai tabel, mari kita atur ulang ke format default, dan menghitung tabel nilai sinus dan kosinus.
Perlu dicatat bahwa sudut secara default dalam radian.

```
>defformat; w=0°:45°:360°; w=w'; deg(w)
```

```
     0
    45
    90
   135
   180
   225
   270
   315
   360
```

Sekarang kita menambahkan kolom ke dalam matriks.

```
>M = deg(w)|w|cos(w)|sin(w)
```

```
     0         0         1         0
    45  0.785398  0.707107  0.707107
    90    1.5708         0         1
   135   2.35619 -0.707107  0.707107
   180   3.14159        -1         0
   225   3.92699 -0.707107 -0.707107
   270   4.71239         0        -1
   315   5.49779  0.707107 -0.707107
   360   6.28319         1         0
```

Dengan bahasa matriks, kita dapat menghasilkan beberapa tabel dari beberapa fungsi sekaligus.

Dalam contoh berikut, kita menghitung t[j]^i untuk i dari 1 sampai n. Hasilnya adalah matriks, di mana setiap baris adalah tabel t^i untuk satu i. Dengan kata lain, elemen matriks adalah:

$$a_{i,j} = t_j^i, \quad 1 \le j \le 101, \quad 1 \le i \le n$$

Sebuah fungsi yang tidak mendukung input vektor perlu divektorisasi. Ini dapat dilakukan dengan kata kunci map dalam definisi fungsi. Maka fungsi tersebut akan dievaluasi untuk setiap elemen dari parameter vektor.

Misalnya, integrasi numerik integrate() hanya bekerja untuk batas interval skalar, sehingga kita perlu vektorisasi fungsi ini.

```
>function map f(x) := integrate("x^x",1,x)
```

Kata kunci map melakukan vektorisasi fungsi. Fungsi tersebut sekarang akan bekerja untuk vektor angka.

```
>f([1:5])
```

```
  [0,  2.05045,  13.7251,  113.336,  1241.03]
```

Sub-Matriks dan Elemen Matriks

Untuk mengakses elemen matriks, gunakan notasi kurung siku.

```
>A=[1,2,3;4,5,6;7,8,9], A[2,2]
```

```
                1              2              3
                4              5              6
                7              8              9
  5
```

Kita dapat mengakses seluruh baris dari sebuah matriks.

```
>A[2]
```

```
  [4,  5,  6]
```

Dalam kasus vektor baris atau kolom, ini akan mengembalikan sebuah elemen dari vektor tersebut.

```
>v=1:3; v[2]
```

```
  2
```

Untuk memastikan, Anda mendapatkan baris pertama dari matriks 1xn atau mxn, tentukan semua kolom dengan menggunakan indeks kedua yang kosong.

```
>A[2,]
```

```
  [4,  5,  6]
```

Jika indeksnya adalah vektor indeks, Euler akan mengembalikan baris-baris matriks yang sesuai.
Di sini kita ingin baris pertama dan kedua dari matriks.

```
>A[[1,2]]
```

```
                1              2              3
                4              5              6
```

Kita bahkan dapat mengubah urutan A menggunakan vektor indeks. Tepatnya, kita tidak mengubah A di sini, tetapi menghitung versi A yang telah diubah urutannya.

```
>A[[3,2,1]]
```

```
7  8  9
4  5  6
1  2  3
```

Trik indeks ini juga berlaku untuk kolom.
Contoh ini memilih semua baris dari A dan kolom kedua serta ketiga.

```
>A[1:3,2:3]
```

```
2  3
5  6
8  9
```

Sebagai singkatan, ":" menunjukkan semua indeks baris atau kolom.

```
> A[:,3]
```

```
3
6
9
```

Sebagai alternatif, biarkan indeks pertama kosong.

```
>A[,2:3]
```

```
2  3
5  6
8  9
```

Kita juga dapat mengambil baris terakhir dari A.

```
>A[-1]
```

```
[7,  8,  9]
```

Sekarang mari kita ubah elemen-elemen A dengan menetapkan sebuah sub-matriks dari A ke suatu nilai. Ini memang akan mengubah matriks A yang tersimpan.

```
>A[1,1]=4
```

```
4  2  3
4  5  6
7  8  9
```

Kita juga dapat menetapkan sebuah nilai pada satu baris dari
A.

```
>A[1]=[-1,-1,-1]
```

```
              -1              -1              -1
               4               5               6
               7               8               9
```

Kita bahkan dapat menetapkan nilai pada sebuah sub-matriks jika ukurannya sesuai.

```
>A[1:2,1:2]=[5,6;7,8]
```

```
               5               6              -1
               7               8               6
               7               8               9
```

Selain itu, beberapa jalan pintas diperbolehkan.

```
>A[1:2,1:2]=0
```

```
               0               0              -1
               0               0               6
               7               8               9
```

Peringatan: Indeks di luar batas akan mengembalikan matriks kosong, atau pesan kesalahan, tergantung pada pengaturan sistem. Secara default, yang muncul adalah pesan kesalahan. Namun, ingat bahwa indeks negatif dapat digunakan untuk mengakses elemen matriks dengan menghitung dari akhir.

```
>A[4]
```

```
  Row index 4 out of bounds!
  Error in:
  A[4] ...
      ^
```

Penyortiran dan Pengacakan

Fungsi sort() digunakan untuk menyortir sebuah vektor baris.

```
>sort([5,6,4,8,1,9])
```

```
  [1,  4,  5,  6,  8,  9]
```

Seringkali perlu mengetahui indeks dari vektor yang telah disortir dalam vektor asli. Hal ini dapat digunakan untuk mengubah urutan vektor lain dengan cara yang sama.

Sekarang, mari kita acak sebuah vektor.

```
>v=shuffle(1:10)
```

```
  [6,  3,  1,  5,  10,  4,  9,  8,  2,  7]
```

Indeks-indeks tersebut berisi urutan yang sesuai dari vektor v.

```
>{vs,ind}=sort(v); v[ind]
```

```
  [1,  2,  3,  4,  5,  6,  7,  8,  9,  10]
```

Hal ini juga berlaku untuk vektor string.

```
>s=["a","d","e","a","aa","e"]
```

```
  a
  d
  e
  a
  aa
  e
```

```
>{ss,ind}=sort(s); ss
```

```
  a
  a
  aa
  d
  e
  e
```

Seperti yang Anda lihat, posisi entri ganda agak acak.

```
>ind
```

```
  [4,  1,  5,  2,  6,  3]
```

Fungsi unique mengembalikan daftar elemen unik dari sebuah vektor yang telah disortir.

```
>intrandom(1,10,10), unique(%)
```

```
  [9,  3,  8,  3,  5,  4,  10,  2,  5,  1]
  [1,  2,  3,  4,  5,  8,  9,  10]
```

Hal ini juga berlaku untuk vektor string.

```
>unique(s)
```

```
  a
  aa
  d
  e
```

EMT memiliki banyak fungsi untuk menyelesaikan sistem linear, sistem sparse, atau masalah regresi.

Untuk sistem linear Ax=b, Anda dapat menggunakan algoritma Gauss, matriks invers, atau linear fit. Operator A\b menggunakan versi dari algoritma Gauss.

```
>A=[1,2;3,4]; b=[5;6]; A\b
```

```
      -4
     4.5
```

Sebagai contoh lain, kita menghasilkan sebuah matriks 200×200 dan jumlah dari tiap barisnya. Kemudian kita menyelesaikan
Ax=b menggunakan matriks invers. Kita mengukur kesalahan sebagai deviasi maksimal dari semua elemen terhadap 1, yang tentu saja merupakan solusi yang benar.

```
>A=normal(200,200); b=sum(A); longest totalmax(abs(inv(A).b-1))
```

```
    3.140820936664568e-13
```

Jika sistem tidak memiliki solusi, linear fit meminimalkan norma dari kesalahan Ax-b.

```
>A=[1,2,3;4,5,6;7,8,9]
```

```
       1       2       3
       4       5       6
       7       8       9
```

Determinan dari matriks ini adalah 0.

```
>det(A)
```

```
  0
```

Matriks Simbolik

Maxima memiliki matriks simbolik. Tentu saja, Maxima dapat digunakan untuk masalah aljabar linier yang sederhana. Kita dapat mendefinisikan matriks untuk Euler dan Maxima menggunakan &:=, kemudian menggunakannya dalam ekspresi simbolik. Bentuk umum [...] untuk mendefinisikan matriks juga dapat digunakan di Euler untuk membuat matriks simbolik.

```
>A &= [a,1,1;1,a,1;1,1,a]; $A
```

$$\begin{pmatrix} a & 1 & 1 \\ 1 & a & 1 \\ 1 & 1 & a \end{pmatrix}$$

```
>$&det(A), $&factor(%)
```

$$(a-1)^2\,(a+2)$$

```
>$&invert(A) with a=0
```

$$\begin{pmatrix} -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \end{pmatrix}$$

```
>A &= [1,a;b,2]; $A
```

$$\begin{pmatrix} 1 & a \\ b & 2 \end{pmatrix}$$

Seperti semua variabel simbolik, matriks ini dapat digunakan dalam ekspresi simbolik lainnya.

```
>$&det(A-x*ident(2)), $&solve(%,x)
```

$$\left[x=\frac{3-\sqrt{4\,a\,b+1}}{2},x=\frac{\sqrt{4\,a\,b+1}+3}{2}\right]$$

$$\left[x=\frac{3-\sqrt{4\,a\,b+1}}{2},x=\frac{\sqrt{4\,a\,b+1}+3}{2}\right]$$

Nilai eigen juga dapat dihitung secara otomatis. Hasilnya adalah sebuah vektor yang berisi dua vektor: nilai eigen dan multiplikitasnya.

```
>$&eigenvalues([a,1;1,a])
```

$$[[a-1,a+1],[1,1]]$$

Untuk mengambil sebuah vektor eigen tertentu diperlukan penanganan indeks yang cermat.

```
>$&eigenvectors([a,1;1,a]), &%[2][1][1]
```

$$[[[a-1,a+1],[1,1]],[[[1,-1]],[[1,1]]]]$$

```
[1, -1]
```

Matriks simbolik dapat dievaluasi secara numerik di Euler sama seperti ekspresi simbolik lainnya.

```
>A(a=4,b=5)
```

```
                1                4
                5                2
```

Dalam ekspresi simbolik, gunakan with.

```
>$&A with [a=4,b=5]
```

$$\begin{pmatrix} 1 & 4 \\ 5 & 2 \end{pmatrix}$$

Akses ke baris matriks simbolik bekerja sama seperti pada matriks numerik.

```
>$&A[1]
```

$$[1, a]$$

Sebuah ekspresi simbolik dapat berisi penugasan. Dan itu akan mengubah matriks A.

```
>&A[1,1]:=t+1; $&A
```

$$\begin{pmatrix} t+1 & a \\ b & 2 \end{pmatrix}$$

Ada fungsi simbolik di Maxima untuk membuat vektor dan matriks. Untuk ini, lihat dokumentasi Maxima atau tutorial tentang Maxima di EMT.

```
>v &= makelist(1/(i+j),i,1,3); $v
```

$$\left[\frac{1}{j+1}, \frac{1}{j+2}, \frac{1}{j+3}\right]$$

```
>B &:= [1,2;3,4]; $B, $&invert(B)
```

$$\begin{pmatrix} -2 & 1 \\ \frac{3}{2} & -\frac{1}{2} \end{pmatrix}$$

$$\left(\begin{smallmatrix} -2 & 1 \\ \frac{3}{2} & -\frac{1}{2} \end{smallmatrix}\right)$$

Hasilnya dapat dievaluasi secara numerik di Euler. Untuk informasi lebih lanjut tentang Maxima, lihat pengantar Maxima.

```
>$&invert(B)()
```

```
          -2            1
         1.5         -0.5
```

Euler juga memiliki fungsi yang kuat xinv(), yang melakukan perhitungan lebih cermat dan menghasilkan hasil yang lebih akurat.

Perlu dicatat, dengan &:= matriks telah didefinisikan sebagai simbolik dalam ekspresi simbolik dan sebagai numerik dalam ekspresi numerik. Jadi kita dapat menggunakannya di sini.

```
>longest B.xinv(B)
```

```
                        1                        0
                        0                        1
```

Misalnya, nilai eigen dari

```
 dapat dihitung secara numerik.
```

```
>A=[1,2,3;4,5,6;7,8,9]; real(eigenvalues(A))
```

```
  [16.1168,  -1.11684,  0]
```

Atau secara simbolik. Lihat tutorial tentang Maxima untuk detail lebih lanjut mengenai hal ini.

```
>$&eigenvalues(@A)
```

$$\left[\left[\frac{15-3\sqrt{33}}{2},\frac{3\sqrt{33}+15}{2},0\right],[1,1,1]\right]$$

Nilai Numerik dalam Ekspresi Simbolik

Sebuah ekspresi simbolik hanyalah sebuah string yang berisi ekspresi. Jika kita ingin mendefinisikan sebuah nilai untuk ekspresi simbolik sekaligus ekspresi numerik, kita harus menggunakan &:=.

```
>A &:= [1,pi;4,5]
```

```
                1          3.14159
                4                5
```

Masih ada perbedaan antara bentuk numerik dan simbolik. Saat mentransfer matriks ke bentuk simbolik, pendekatan pecahan untuk bilangan real akan digunakan.

```
>$&A
```

$$\begin{pmatrix}1 & \frac{1146408}{364913}\\ 4 & 5\end{pmatrix}$$

Untuk menghindari hal ini, terdapat fungsi mxmset(variable).

```
>mxmset(A); $&A
```

$$\begin{pmatrix} 1 & 3.141592653589793 \\ 4 & 5 \end{pmatrix}$$

Maxima juga dapat menghitung dengan angka titik mengambang, bahkan dengan angka titik mengambang besar hingga 32 digit. Namun, proses evaluasinya jauh lebih lambat.

```
>$&bfloat(sqrt(2)), $&float(sqrt(2))
```

$$1.414213562373095$$

Presisi dari angka titik mengambang besar dapat diubah.

```
>&fpprec:=100; &bfloat(pi)
```

```
          3.14159265358979323846264338327950288419716939937510582097494\
  459230781640628620899862803482534211706 8b0
```

Variabel numerik dapat digunakan dalam ekspresi simbolik manapun menggunakan @var.

Perlu dicatat bahwa ini hanya diperlukan jika variabel telah didefinisikan dengan := atau = sebagai variabel numerik.

```
>B:=[1,pi;3,4]; $&det(@B)
```

$$-5.424777960769379$$

Demo – Suku Bunga

Di bawah ini, kita menggunakan Euler Math Toolbox (EMT) untuk perhitungan suku bunga. Kita melakukannya secara numerik dan simbolik untuk menunjukkan bagaimana Euler dapat digunakan untuk menyelesaikan masalah kehidupan nyata.

Misalkan Anda memiliki modal awal sebesar 5000 (misalnya dalam dolar).

```
>K=5000
```

```
  5000
```

Sekarang kita anggap suku bunga 3% per tahun. Mari kita tambahkan satu suku bunga sederhana dan hitung hasilnya.

```
>K*1.03
```

```
  5150
```

Euler juga akan memahami sintaks berikut ini.

```
>K+K*3%
```

```
  5150
```

Namun, lebih mudah menggunakan faktor.

```
>q=1+3%, K*q
```

```
  1.03
  5150
```

Untuk 10 tahun, kita cukup mengalikan faktor-faktor tersebut dan mendapatkan nilai akhir dengan suku bunga majemuk.

```
>K*q^10
```

```
  6719.58189672
```

Untuk keperluan kita, kita dapat mengatur format menjadi 2 digit setelah titik desimal.

```
>format(12,2); K*q^10
```

```
      6719.58
```

Mari kita cetak hasil tersebut dibulatkan menjadi 2 digit dalam sebuah kalimat lengkap.

```
>"Starting from " + K + "$ you get " + round(K*q^10,2) + "$."
```

```
  Starting from 5000$ you get 6719.58$.
```

Bagaimana jika kita ingin mengetahui hasil antara dari tahun 1 hingga tahun 9? Untuk ini, bahasa matriks Euler sangat membantu. Anda tidak perlu menulis sebuah loop, cukup masukkan:

```
>K*q^(0:10)
```

```
  Real 1 x 11 matrix

     5000.00      5150.00      5304.50      5463.64      ...
```

Bagaimana cara "ajaib" ini bekerja? Pertama, ekspresi 0:10 mengembalikan sebuah vektor bilangan bulat.

```
>short 0:10
```

```
  [0,  1,  2,  3,  4,  5,  6,  7,  8,  9,  10]
```

Kemudian semua operator dan fungsi di Euler dapat diterapkan ke vektor elemen demi elemen. Jadi,

```
>short q^(0:10)
```

```
  [1,  1.03,  1.0609,  1.0927,  1.1255,  1.1593,  1.1941,  1.2299,
  1.2668,  1.3048,  1.3439]
```

merupakan vektor faktor dari q^0 hingga q^10. Vektor ini dikalikan dengan K, sehingga kita mendapatkan vektor nilai-nilai.

```
>VK=K*q^(0:10);
```

Tentu saja, cara realistis untuk menghitung suku bunga ini adalah dengan membulatkan ke sen terdekat setelah setiap tahun. Mari kita tambahkan sebuah fungsi untuk ini.

```
>function oneyear (K) := round(K*q,2)
```

Mari kita bandingkan kedua hasil tersebut, dengan dan tanpa pembulatan.

```
>longest oneyear(1234.57), longest 1234.57*q
```

```
                 1271.61
               1271.6071
```

Sekarang tidak ada rumus sederhana untuk tahun ke-n, sehingga kita harus melakukan loop selama beberapa tahun. Euler menyediakan banyak solusi untuk ini.

Cara termudah adalah menggunakan fungsi iterate, yang menjalankan sebuah fungsi tertentu beberapa kali.

```
>VKr=iterate("oneyear",5000,10)
```

```
  Real 1 x 11 matrix

     5000.00      5150.00      5304.50      5463.64      ...
```

Kita dapat mencetaknya dengan cara yang mudah dibaca, menggunakan format kita dengan jumlah desimal tetap.

```
>VKr'
```

```
      5000.00
      5150.00
      5304.50
      5463.64
      5627.55
      5796.38
      5970.27
      6149.38
      6333.86
      6523.88
      6719.60
```

Untuk mendapatkan elemen tertentu dari vektor, kita menggunakan indeks di dalam tanda kurung siku.

```
>VKr[2], VKr[1:3]
```

```
      5150.00
      5000.00     5150.00     5304.50
```

Mengejutkan, kita juga dapat menggunakan vektor indeks. Ingat bahwa 1:3 menghasilkan vektor [1, 2, 3]. Mari kita bandingkan elemen terakhir dari nilai yang dibulatkan dengan nilai lengkapnya.

```
>VKr[-1], VK[-1]
```

```
      6719.60
      6719.58
```

Perbedaannya sangat kecil.

Menyelesaikan Persamaan

Sekarang kita mengambil fungsi yang lebih maju, yang menambahkan sejumlah suku bunga setiap tahun.

```
>function onepay (K) := K*q+R
```

Kita tidak perlu menentukan atau saat mendefinisikan fungsi. Hanya saat menjalankan perintah, kita harus mendefinisikan nilai-nilai tersebut. Kita pilih R=200.

```
>R=200; iterate("onepay",5000,10)
```

```
  Real 1 x 11 matrix

     5000.00     5350.00     5710.50     6081.82      ...
```

Bagaimana jika kita mengurangi jumlah yang sama setiap tahun?

```
>R=-200; iterate("onepay",5000,10)
```

```
  Real 1 x 11 matrix

     5000.00     4950.00     4898.50     4845.45      ...
```

Kita melihat bahwa uang berkurang. Jelas, jika kita hanya mendapatkan 150 dari bunga di tahun pertama, tetapi menarik 200, kita akan kehilangan uang setiap tahun.

Bagaimana kita bisa menentukan berapa tahun uang tersebut akan bertahan? Kita harus menulis sebuah loop untuk ini. Cara termudah adalah dengan melakukan iterasi cukup lama.

```
>VKR=iterate("onepay",5000,50)
```

```
  Real 1 x 51 matrix

     5000.00      4950.00      4898.50      4845.45      ...
```

Dengan menggunakan bahasa matriks, kita dapat menentukan nilai negatif pertama dengan cara berikut.

```
>min(nonzeros(VKR<0))
```

```
        48.00
```

Alasannya adalah bahwa nonzeros(VKR<0) mengembalikan vektor indeks i, di mana VKR[i] < 0, dan min menghitung indeks terkecil.
Karena vektor selalu dimulai dari indeks 1, jawabannya adalah 47 tahun.
Fungsi iterate() memiliki satu trik tambahan. Fungsi ini dapat menerima kondisi akhir sebagai argumen. Kemudian, ia akan mengembalikan nilai dan jumlah iterasi.

```
>{x,n}=iterate("onepay",5000,till="x<0"); x, n,
```

```
       -19.83
        47.00
```

Mari kita coba menjawab pertanyaan yang lebih ambigu. Misalkan kita tahu bahwa nilai uang menjadi 0 setelah 50 tahun. Berapa suku bunganya?

Ini adalah pertanyaan yang hanya dapat dijawab secara numerik. Di bawah ini, kita akan menurunkan rumus-rumus yang diperlukan. Kemudian Anda akan melihat bahwa tidak ada rumus mudah untuk suku bunga. Namun untuk saat ini, kita akan mencari solusi secara numerik.

Langkah pertama adalah mendefinisikan sebuah fungsi yang melakukan iterasi sebanyak n kali. Kita tambahkan semua parameter ke dalam fungsi ini.

```
>function f(K,R,P,n) := iterate("x*(1+P/100)+R",K,n;P,R)[-1]
```

Iterasinya sama persis seperti sebelumnya.

$$x_{n+1} = x_n \cdot \left(1 + \frac{P}{100}\right) + R$$

Namun sekarang kita tidak lagi menggunakan nilai global R dalam ekspresi kita. Fungsi seperti iterate() memiliki trik khusus di Euler. Anda dapat melewatkan nilai variabel dalam ekspresi sebagai parameter setelah titik koma. Dalam kasus ini, P dan R.

Selain itu, kita hanya tertarik pada nilai terakhir. Jadi kita ambil indeks [-1].
Mari kita coba sebuah uji coba.

```
>f(5000,-200,3,47)
```

```
       -19.83
```

Sekarang kita dapat menyelesaikan masalah kita.

```
>solve("f(5000,-200,x,50)",3)
```

```
         3.15
```

Rutin solve menyelesaikan persamaan expression = 0 untuk variabel x. Jawabannya adalah 3,15% per tahun. Kita menggunakan nilai awal 3% untuk algoritma. Fungsi solve() selalu memerlukan nilai awal.

Kita dapat menggunakan fungsi yang sama untuk menjawab pertanyaan berikut: Berapa banyak yang dapat kita tarik setiap tahun sehingga modal awal habis setelah 20 tahun dengan asumsi suku bunga 3% per tahun.

```
>solve("f(5000,x,3,20)",-200)
```

```
      -336.08
```

Perlu dicatat bahwa Anda tidak dapat menyelesaikan untuk jumlah tahun, karena fungsi kita mengasumsikan n sebagai nilai bilangan bulat.

Solusi Simbolik untuk Masalah Suku Bunga

Kita dapat menggunakan bagian simbolik dari Euler untuk mempelajari masalah ini. Pertama, kita mendefinisikan fungsi onepay() secara simbolik.

```
>function op(K) &= K*q+R; $&op(K)
```

$$R+q\,K$$

Sekarang kita dapat melakukan iterasi pada fungsi ini.

```
>$&op(op(op(op(K)))), $&expand(%)
```

$$q^3\,R+q^2\,R+q\,R+R+q^4\,K$$

Kita melihat sebuah pola. Setelah n periode kita peroleh

$$K_n=q^nK+R(1+q+\ldots+q^{n-1})=q^nK+\frac{q^n-1}{q-1}R$$

Rumus ini adalah rumus jumlah geometri, yang sudah dikenal oleh Maxima.

```
>&sum(q^k,k,0,n-1); $& % = ev(%,simpsum)
```

$$\sum_{k=0}^{n-1} q^k = \frac{q^n - 1}{q - 1}$$

Ini sedikit rumit. Jumlah tersebut dievaluasi dengan flag simpsum untuk mereduksinya menjadi bentuk pecahan.

Mari kita buat sebuah fungsi untuk ini.

```
>function fs(K,R,P,n) &= (1+P/100)^n*K + ((1+P/100)^n-1)/(P/100)*R; $&fs(K,R,P,n)
```

$$\frac{100\left(\left(\frac{P}{100}+1\right)^n-1\right)R}{P}+K\left(\frac{P}{100}+1\right)^n$$

Fungsi ini melakukan hal yang sama seperti fungsi n kita sebelumnya. Namun, fungsinya lebih efektif.

```
>longest f(5000,-200,3,47), longest fs(5000,-200,3,47)
```

```
  Function f not found.
  Try list ... to find functions!
  Error in:
  longest f(5000,-200,3,47), longest fs(5000,-200,3,47) ...
                             ^
```

Kita sekarang dapat menggunakannya untuk menentukan waktu n. Kapan modal kita habis? Tebakan awal kita adalah 30 tahun.

```
>solve("fs(5000,-330,3,x)",30)
```

```
  20.5061016552
```

Jawaban ini menyatakan bahwa nilainya akan menjadi negatif setelah 21 tahun.

Kita juga dapat menggunakan sisi simbolik Euler untuk menghitung rumus-rumus pembayaran.

Misalkan kita mendapat pinjaman sebesar K, dan membayar n kali pembayaran sebesar R (dimulai setelah tahun pertama) sehingga tersisa utang residu Kn (pada saat pembayaran terakhir). Rumus untuk ini jelas:

```
>equ &= fs(K,R,P,n)=Kn; $&equ
```

$$\frac{100\left(\left(\frac{P}{100}+1\right)^n-1\right)R}{P}+K\left(\frac{P}{100}+1\right)^n=Kn$$

Biasanya rumus ini dinyatakan dalam bentuk

$$i = \frac{P}{100}$$

```
>equ &= (equ with P=100*i); $&equ
```

$$\frac{((i+1)^n - 1)\ R}{i} + (i+1)^n\ K = Kn$$

Kita dapat menyelesaikan untuk R secara simbolik.

```
>$&solve(equ,R)
```

$$\left[R = \frac{i\,Kn - i\ (i+1)^n\ K}{(i+1)^n - 1}\right]$$

Seperti yang dapat Anda lihat dari rumus, fungsi ini menghasilkan kesalahan floating-point untuk i=0. Euler tetap memplotnya.
Tentu saja, kita memiliki batas berikut.

```
>$&limit(R(5000,0,x,10),x,0)
```

$$\lim_{x \to 0} R\,(5000, 0, x, 10)$$

Jelas, tanpa bunga kita harus membayar kembali 10 kali pembayaran sebesar 500.
Persamaan tersebut juga dapat diselesaikan untuk n. Akan terlihat lebih rapi jika kita menerapkan beberapa penyederhanaan padanya.

```
>fn &= solve(equ,n) | ratsimp; $&fn
```

$$\left[n = \frac{\log\left(\frac{R+i\,Kn}{R+i\,K}\right)}{\log\,(i+1)}\right]$$

LATIHAN SOAL

1. Jabarkan bentuk aljabar berikut

$$(x+6)(x+3)$$

```
>$&showev('expand((x+6)*(x+3)))
```

$$expand\,((x+3)\ (x+6)) = x^2 + 9\,x + 18$$

```
>$&expand((x+6)*(x+3))
```

$$x^2 + 9\,x + 18$$

Lebih ringkas menggunakan perintah &expand secara langsung

```
>expand((x+6)*(x+3))
```

```
  Variable x not found!
  Error in:
  expand((x+6)*(x+3)) ...
             ^
```

Penggunaan & perlu untuk mengirim ekspresi ke Maxima, karena bentuk tersebut menggunakan mode simbolik. Sedangkan jika tanpa & hanya dibaca mode numerik EMT biasa.

2. Hitung faktor dari bentuk aljabar berikut

$$x^2 + 9x + 20$$

```
>$&factor(x^2+9*x+20), $&diff(%)
```

$$(x + 4)\ (x + 5)$$

$$(2\,x + 9)\ dx$$

```
>$&diff(x^2 + 9*x + 20, x, 2), $&diff(%)
```

$$2$$

$$0$$

&diff(fungsi, variabel, 2) merupakan perintah untuk menghitung turunan kedua.

3. Carilah solusi dari persamaan berikut

$$x^2 + 5x = 0$$

```
>$& solve(x^2 + 5*x = 0, x)
```

$$[x = -5, x = 0]$$

Penggunaan & menunjukan bahwa persamaan tersebut mode simbolik, sehingga menggunakan Maxima. Sedangkan untuk menyelesaikan dengan mode numerik menggunakan perintah solve(fungsi, x, 0)

```
>solve("x^2+5*x", -5)
```

```
  -5
```

4. Sederhanakan bentuk aljabar berikut

$$\frac{x^3 - 6x^2 + 9x}{x^3 - 3x^2}$$

```
>$&fullratsimp((x^3 - 6*x^2 + 9*x) / (x^3 - 3*x^2))
```

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

```
>$&simplify((x^3 - 6*x^2 + 9*x) / (x^3 - 3*x^2))
```

$$simplify\left(\frac{x^3 - 6\,x^2 + 9\,x}{x^3 - 3\,x^2}\right)$$

```
>&simplify(( x^3 - 6*x^2 + 9*x) / (x^3 - 3*x^2))
```

```
                      3      2
                     x  - 6 x  + 9 x
simplify(----------------------------)
                        3      2
                       x  - 3 x
```

Perintah &simplify hanya akan menuliskan gaya LaTeX saja atau EMT langsung. sedangangkan perintah &fullratsimp akan menyederhanakan bentuk aljabar yang diberi.

5. Lamont ingin memiliki tabungan sebesar $200.000 di rekening pensiunnya pada usia 70 tahun. Jika ia mulai menabung dengan setoran bulanan sejak usia 30 tahun dan dapat mengandalkan tingkat bunga 4,5% dengan bunga bulanan, berapa besar setoran yang harus ia bayarkan setiap bulan agar tujuannya tercapai?

```
>function fs(K,R,P,n) &= (1+P/100)^n*K + ((1+P/100)^n-1)/(P/100)*R;
```

Keterangan:
- n: periode waktu menabung (bulan)
- K: modal awal=0
- R: setoran tiap bulan
- P: bungan per periode (pertahun)

```
>function fs(K,R,i,n) &= (1+i)^n*K + ((1+i)^n - 1)/i*R;
```

ket:
- i: P perbulan (0.045/12)
- n= (70-30)*12= 480
- K= 0

Akan dicari fs=200000

```
>equ &= fs(0,R,i,n) = FV;
```

Subtitusikan angka ke dalam parameter

```
>equ := equ with [i=0.045/12, n=480, FV=200000]
```

Perintahkan untuk memecahkan persamaan untuk R.

```
>$&solve(equ,R)
```

$$\left[R = \frac{i\,FV}{(i+1)^n - 1}\right]$$

```
>R := 200000*(0.045/12)/((1+0.045/12)^480 - 1)
```

```
  149.12567469
```

```
>ceil(149.1289)
```

```
  150
```

Berikan perintah untuk membulatkan angka ke atas dengan perintah ceil().
Jadi, Pak Lamont perlu menabung sebesar $150 setiap bulan untuk mencapai tujuannya.