# 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 : Cindy Aulia Rosada
NIM : 24030130066
Kelas : Pendidikan Matematika A 2024
Tugas : Tugas pekan 3-4 Perhitungan Aljabar Menggunakan EMT

## 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))))
```

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

expand(...) : fungsi untuk mengembangkan hasil perkalian antara dua polinomial
showev(...) : menampilkan hasil yang sudah dievaluasi dalam format yang rapi
$...$ di EMT yang kita pakai di contoh ($showev(...);$) membuat hasilnya diperlakukan sebagai LaTeX style output.

## Baris Perintah

Sebuah baris perintah (command line) di Euler terdiri dari satu atau beberapa perintah Euler yang diikuti dengan tanda titik koma ; atau tanda koma ,. Tanda titik koma berfungsi untuk mencegah hasil perintah ditampilkan. Sedangkan tanda koma pada perintah terakhir boleh dihilangkan.

Baris perintah berikut hanya akan menampilkan hasil dari ekspresi, bukan penugasan (assignments) atau perintah format.

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

```
  16.7551608191
```

## Pemisahan Perintah

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

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

```
  50.2654824574
  100.530964915
```

## Eksekusi Baris

Baris perintah dijalankan sesuai urutan ketika pengguna menekan tombol Enter. Jadi, setiap kali menjalankan baris perintah kedua, nilainya akan diperbarui.

```
>x := 3;
>x := cos(x) // nilai cosinus (x dalam radian)
```

```
  -0.9899924966
```

```
>x := sin(x) // nilai sinus (x dalam radian)
```

```
  0.521575206867
```

## Menghubungkan Baris

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

```
>x := 1.7; ...
>x := (x+2/x)/2, x := (x+2/x)/2, x := (x+2/x)/2,
```

```
  1.43823529412
  1.41441417058
  1.4142135766
```

Ini juga cara yang tepat untuk membagi sebuah perintah yang panjang ke dalam dua baris atau lebih. Bisa menekan Ctrl+Return untuk memecah satu baris menjadi dua pada posisi kursor saat ini, atau menekan Ctrl+Back untuk menggabungkan kembali baris-baris tersebut.

Untuk menyembunyikan semua baris yang saling terhubung, tekan Ctrl+L. Setelah itu, baris-baris berikutnya hanya akan terlihat jika salah satunya sedang dipilih (fokus). Untuk melipat satu baris multi-line saja, mulai baris pertamanya dengan %+ ".

```
>%+ x=4+5; ...
```

selanjutnya baris yang diawali %% akan sama sekai takterlihat.

```
  5.65685424949
```

## Loop (Perulangan)

Euler mendukung penggunaan perulangan (loops) di dalam baris perintah, selama perulangan tersebut dituliskan dalam satu baris atau menggunakan multi-baris. Namun, dalam program, pembatasan ini tentu tidak berlaku. Untuk informasi lebih lanjut, silakan merujuk pada pengantar berikut.

```
>x=1; for i=1 to 6; x := (x+2/x)/2, end; // menghitung akar 2
```

```
  1.5
  1.41666666667
  1.41421568627
  1.41421356237
  1.41421356237
  1.41421356237
```

Boleh menggunakan multi-baris. Pastikan setiap baris diakhiri dengan "....".

```
>x := 1.5; // comments go here before the ...
>repeat xnew:=(x+2/x)/2; until xnew~=x; ...
>   x := xnew; ...
>end; ...
>x,
```

```
  1.41421356237
```

## Struktur Kondisional

Struktur bersyarat juga dapat digunakan.

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

```
  Thought so!
```

if ... then ... endif;
=> Ini adalah struktur bersyarat (conditional structure) di EMT, mirip seperti if statement dalam bahasa pemrograman

E^pi > pi^E
=> Kondisi yang diuji (apakah benar atau salah).

Thought so!"
=> Teks yang akan ditampilkan jika kondisi bernilai benar.

; (titik koma)
=> Memisahkan perintah dalam satu baris.

## Navigasi Perintah

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

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

```
>sqrt(sin(45°)/cos(20°))
```

```
  0.867460269583
```

## Bantuan

Untuk melihat bantuan dari perintah terakhir, buka jendela help dengan F1.
Anda bisa mencari teks di sana. Jika baris kosong, akan muncul bantuan tentang jendela help.
Tekan Escape untuk menghapus baris atau menutup jendela bantuan.
Klik ganda pada perintah mana pun untuk membuka bantuan tentang perintah tersebut

```
>exp(log(4.5))
```

```
  4.5
```

## Copy Paste

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.

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

Untuk menetapkan variabel, gunakan tanda = atau :=. Demi kejelasan, pengantar ini menggunakan bentuk yang terakhir (:=). Spasi tidak menjadi masalah. Namun, spasi antarperintah tetap diperlukan.

Beberapa perintah dalam satu baris dipisahkan dengan tanda koma , atau titik koma ;. Tanda titik koma berfungsi untuk menyembunyikan (tidak menampilkan) keluaran dari perintah tersebut. Pada akhir baris perintah, tanda koma , akan dianggap ada jika tanda titik koma ; tidak dituliskan.

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

```
  30.65625
```

## Menuliskan Ekspresi Matematika

EMT menggunakan sintaks pemrograman untuk ekspresi. Memasuki

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

Anda harus menuliskan tanda kurung yang benar dan menggunakan untuk pecahan. Perhatikan tanda kurung yang disorot untuk bantuan.Perlu dicatat bahwa konstanta Euler ediberi nama E dalam EMT.

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

```
  8.77908249441
```

Untuk menghitung sebuah ekspresi yang rumit seperti

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

Anda perlu memasukkannya dalam bentuk satu baris.

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

```
  23.2671801626
```

Pasang tanda kurung dengan hati-hati pada sub-ekspresi yang harus dihitung terlebih dahulu. EMT akan membantu Anda dengan menyorot ekspresi yang ditutup oleh tanda kurung tersebut. Anda juga harus menuliskan nama pi untuk huruf Yunani p (pi).

Hasil perhitungan ini berupa bilangan desimal (floating point number). Secara bawaan (default), hasil tersebut ditampilkan dengan ketelitian sekitar 12 digit. Pada baris perintah berikutnya, kita juga akan mempelajari bagaimana cara merujuk hasil sebelumnya di dalam baris yang sama.

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

```
  0.47619047619
  10/21
```

Perintah dalam Euler dapat berupa sebuah ekspresi atau perintah dasar (primitive command). 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 diperhatikan bahwa EMT akan 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 di Euler mencakup:
+ operator plus tunggal (unary plus)
- operator minus tunggal (unary minus)
*, /
. perkalian matriks (matrix product)
a^b pangkat, untukpositif atau b bilangan bulat (a**b juga bisa digunakan)
n! operator faktorial
dan masih banyak lagi.
Berikut ini adalah beberapa fungsi yang mungkin Anda perlukan. 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 sebagai nama lain dari log.

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

```
  2
  0.5
```

```
>sin(30°)
```

```
  0.5
```

Pastikan untuk menggunakan tanda kurung (kurung biasa) setiap kali ada keraguan mengenai urutan eksekusi! Ekspresi berikut tidak sama dengan
(2^3)^4, karena di EMT bentuk bawaan dari 2^3^4(beberapa sistem numerik lain justru melakukannya dengan cara yang berbeda).

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

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

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

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

## String

String (teks) dalam Euler didefinisikan dengan tanda kutip ganda "..."

```
>"Sebuah string dapat berisi apa saja."
```

```
  Sebuah string dapat berisi apa saja.
```

## Menggabungkan String

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

```
>"Luas lingkaran dengan jari-jari " + 2 + " cm adalah " + pi*4 + " cm^2."
```

```
  Luas lingkaran dengan jari-jari 2 cm adalah 12.5663706144 cm^2.
```

## Fungsi Print

Fungsi print juga dapat mengubah suatu bilangan menjadi sebuah string. Fungsi ini bisa menerima argumen berupa jumlah digit dan jumlah tempat desimal (0 untuk output rapat), serta secara opsional satuan.

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

```
  Golden Ratio : 1.61803
```

Ada sebuah nilai khusus None, yang tidak dicetak. Nilai ini dikembalikan oleh beberapa fungsi ketika hasilnya tidak penting. Nilai None juga dikembalikan secara otomatis jika fungsi tidak memiliki pernyataan return.

## Nilai None

```
>v:= [none] ; v| ["a", "b"]
```

```
  a
  b
```

None adalah nilai khusus yang digunakan ketika suatu fungsi tidak menghasilkan apa-apa. Nilai ini bisa dianggap sebagai “kosong” atau “tidak ada hasil”, sehingga tidak akan menampilkan apa-apa saat dicetak.

Jika ada vektor string yang digabungkan seperti ["None", "a", "b"] dan fungsi tertentu mengabaikan nilai None, hasil yang ditampilkan hanya "a" dan "b" karena None dianggap kosong dan tidak ikut dicetak.

Untuk mengubah sebuah string menjadi angka, cukup evaluasi string tersebut. Cara ini juga berlaku untuk ekspresi matematika.

## Vektor String

```
>"135.7"()
```

```
  135.7
```

Untuk mendefinisikan sebuah vektor string, gunakan notasi vector [...].
Kalau string berisi angka dan akan digunakan sebagai angka sungguhan yang bisa dihitung, kita perlu evaluasi angka tersebut dengan memberikan tanda kurung () di belakang string.

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

```
  affe
  charlie
  bravo
```

Vektor string kosong dilambangkan dengan [none]. Vektor- vektor string dapat digabungkan (dikonkat). Untukmembuatliststring,gunakanformat[...]denganpemisahantarelemenberupa","tandakoma

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

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

Vektorstringdapatdigabungkandengantanda|

## Unicode

String dapat berisi karakter Unicode. Secara internal, string ini disimpan dalam kode UTF-8. Untuk membuat string seperti ini, gunakan u"..." dan salah satu entitas HTML.

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

```
   = 45°
```

I

```
>u"&pi;="+print(pi,2,0),
```

```
  =3.14
```

Di dalam komentar, entitas yang sama seperti , , dan sebagainya dapat digunakan. Ini bisa menjadi alternatif cepat untuk menulis notasi seperti di LaTeX.

## Fungsi Strtochar()

```
>w=strtochar (u"&alpha;&beta;&pi;")
```

```
  [945,  946,  960]
```

Terdapat 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]
```

Hasilnya adalah sebuah vektor yang berisi angka-angka Unicode. Fungsi sebaliknya adalah chartoutf().

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

```
  Ü is a German letter
```

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

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

```
  We have =.
```

Juga dimungkinkan untuk menggunakan entitas numerik.

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

```
  Ähnliches
```

## Nilai Boolean

Nilai boolean sebagai logika benar atau salah di representasikan dalam euler dengan 1=true(benar)dan 0=false(salah

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

```
  0
  1
```

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

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

```
  1
```

Operator Boolean mengikuti aturan bahasa matriks.

## Boolean dengan Vektor

```
>(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. Dalam contoh ini, kita menggunakan 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]
```

```
 Lalu untukmenyaring N agar menyisakan bilangan prima, bisa
```

menggunakan perintah[nonzeros(isprime(N))]

## Format Keluaran

Format output default EMT mencetak 12 digit. Untuk memastikan kita melihat format default, kita mengatur ulang formatnya.

## Defalut Format

```
>defformat; pi
```

```
  3.14159265359
```

Secara internal, EMT menggunakan standar IEEE untuk angka bertipe double dengan sekitar 16 digit desimal. Untuk melihat semua digit secara lengkap, gunakan perintah longestformat, atau kita dapat menggunakan operator longest untuk menampilkan hasil dalam format terpanjang.

## Longlest Format

```
>longest pi
```

```
        3.141592653589793
```

Berikut adalah representasi internal dalam format heksadesimal dari sebuah angka bertipe double.

## Format Heksadesimal

```
>printhex(pi)
```

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

printhex() merupakan cara komputer menyimpan pi yaitu dalam bentuk biner atau heksadesimal. Nilai pi bukan disimpan sebagai3.1415..namun dalam representasi biner.
Format output dapat diubah secara permanen menggunakan perintah format.

## Mengatur Tampilan Format

```
>format(12,4); 1/6, pi, sin(45)
```

```
       0.1667
       3.1416
       0.8509
```

Format tampilan dapat diatur seperti contoh berikut. Perintah format (12,4) artinya 12= lebar total (soasi yang digunakan) dan 4 = jumlah angka di belakang koma

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

```
  0.333333333333
```

Ini merupakan standar format yaitu kembali ke defalut 12 digit

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

## Shortestformat

```
>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 default untuk skalar adalah format(12). Namun, ini dapat diubah.

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

```
  3.1416
```

Mengubah format skalar yang defaultnya 12 menjadi 4 sehingga hanya menampil kan 4 digit.
Fungsi longestformat juga mengatur format untuk skalar.

## Longestformat

```
>longestformat; pi
```

```
  3.141592653589793
```

Sebagai referensi, berikut adalah daftar format output yang paling penting:
shortestformat

shortformat

longformat, longestformat

format(length,digits)

goodformat(length)

fracformat(length)

defformat

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

Namun, format output EMT dapat diatur dengan cara yang fleksibel.

```
>longestformat; pi,
```

```
  3.141592653589793
```

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

```
    3.14159
```

Format default adalah defformat().

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

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

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

```
        4.934802200544679
```

## Format Pecahan

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

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

```
  25/12
```

Karena format internal menggunakan cara biner untuk menyimpan angka, nilai 0.1 tidak akan terwakili secara tepat. Kesalahan ini akan sedikit bertambah, seperti yang terlihat pada perhitungan berikut.

### Masalah Angka Desimal dengan Komputer

```
>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 format default longformat, hal ini tidak akan terlihat. Untuk kenyamanan, 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
```

## Ekspresi

String atau nama variabel dapat digunakan untuk menyimpan ekspresi matematika, yang dapat dievaluasi oleh EMT. Untuk ini, gunakan tanda kurung setelah ekspresi. Jika Anda berniat menggunakan string sebagai ekspresi, gunakan konvensi penamaan seperti fx atau fxy dan sebagainya. Ekspresi memiliki prioritas lebih tinggi dibandingkan fungsi.

Variabel global dapat digunakan dalam proses evaluasi.

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

```
        12.56637061435917
```

Parameter akan diberikan ke x, y, dan z secara berurutan. Parameter tambahan dapat ditambahkan menggunakan parameter yang sudah ditetapkan.

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

```
  -0.919535764538
```

Perlu dicatat bahwa ekspresi akan selalu menggunakan variabel global, meskipun ada variabel dengan nama yang sama di dalam sebuah fungsi. (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)
```

```
  45
```

Sebagai catatan, kami ingin menekankan bahwa call collections (dibahas di bagian lain) dapat berisi ekspresi. Jadi, kita dapat membuat contoh di atas sebagai berikut.

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

```
  45
```

Ekspresi dalam x sering digunakan seperti fungsi.
erlu dicatat bahwa mendefinisikan sebuah fungsi dengan nama yang sama seperti ekspresi simbolik global akan menghapus variabel ini, 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 diberi nama fx, fxy, dan sebagainya. Skema penamaan ini sebaiknya tidak digunakan untuk fungsi.

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

Bentuk khusus dari sebuah ekspresi memungkinkan penggunaan variabel apa pun sebagai parameter tak bernama untuk evaluasi ekspresi, tidak hanya 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 dengan variabel lain untuk fungsi-fungsi EMT yang membutuhkan ekspresi dalam "x".

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

Seperti yang terlihat pada contoh berikut, variabel global dapat diakses 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 mengandung fungsi-fungsi yang hanya dikenal di numerical kernel, bukan di Maxima.

## Simbol Matematika

EMT melakukan perhitungan matematika simbolik dengan bantuan Maxima. Untuk informasi lebih lanjut, mulailah dengan tutorial berikut, atau lihat referensi untuk Maxima. Para ahli Maxima sebaiknya mencatat bahwa terdapat perbedaan sintaks antara sintaks asli Maxima dan sintaks default ekspresi simbolik di EMT.

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

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

```
>$&44!
```

$$26582715747884487680436258110146158903196385280000000000$$

Dengan cara ini, Anda dapat menghitung hasil yang sangat 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 hal ini (demikian juga bagian numerik dari EMT).

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

$$2481256778$$

Untuk mempelajari lebih lanjut tentang sebuah 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 akan mengetahui bahwa hal berikut juga berfungsi.

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

```
>$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 dari sebuah 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-contoh sebelumnya dan berikutnya, jika Anda telah menginstal LaTeX, 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 Anda tidak memiliki LaTeX yang terpasang.

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

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

Ekspresi simbolik diparsing oleh Euler. Jika Anda membutuhkan sintaks yang kompleks dalam satu ekspresi, Anda dapat menempatkan ekspresi tersebut di dalam tanda "...". Menggunakan lebih dari satu ekspresi kompleks dimungkinkan, tetapi sangat tidak disarankan.

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

25

Sebagai pelengkap, kami ingin menekankan bahwa ekspresi simbolik dapat digunakan dalam program, tetapi harus ditempatkan di dalam tanda kutip. Selain itu, akan jauh lebih efektif jika Maxima dipanggil pada saat compile time, 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 perintah Maxima juga tersedia. Mulailah baris perintah dengan ::. Sintaks Maxima disesuaikan dengan sintaks EMT (disebut sebagai 'mode kompatibilitas').

```
>&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 ahli dalam Maxima, Anda mungkin ingin menggunakan sintaks asli Maxima. Anda dapat melakukannya dengan ":::".

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

```
  2
g
```

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

```
  3   x
x   E
```

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

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

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

```
     5
125 E
```

$$125\, e^5$$

```
18551.64488782208
```

```
>fx(5)
```

```
   18551.6449
```

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

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

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

```
          10         5
1000 E    - 125 E
```

```
2.20079141499189e+7
```

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

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

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

```
>tex(fx)
```

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

Ekspresi simbolik dapat dievaluasi sama seperti ekspresi numerik.

```
>fx(0.5)
```

```
        0.2061
```

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

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

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

Penugasan juga dapat bersifat simbolik.

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

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

Perintah solve menyelesaikan ekspresi simbolik untuk suatu variabel di Maxima. Hasilnya berupa 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 solve numerik di Euler, yang membutuhkan nilai awal, dan opsional nilai target.

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

```
       1.5616
```

Nilai numerik dari solusi simbolik dapat dihitung dengan mengevaluasi hasil simbolik tersebut. Euler akan membaca penugasan seperti x= dan seterusnya. Jika Anda tidak memerlukan nilai numerik untuk perhitungan selanjutnya, Anda juga dapat membiarkan Maxima yang mencari nilai numeriknya.

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

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

```
      -3.2361       1.2361
```

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

Untuk mendapatkan solusi simbolik tertentu, seseorang 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 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 di Maxima. Beberapa flag juga dapat digunakan sebagai perintah, sedangkan yang lain tidak. Flag ditambahkan dengan | (bentuk yang lebih rapi 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

Di EMT, fungsi adalah program yang didefinisikan dengan perintah function. Fungsi ini bisa berupa fungsi satu baris atau multi-baris.
Fungsi satu baris dapat bersifat numerik atau simbolik. Fungsi satu baris numerik didefinisikan dengan :=.

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

Sebagai gambaran, kami menunjukkan semua kemungkinan definisi untuk fungsi satu baris. Sebuah fungsi dapat dievaluasi sama seperti fungsi bawaan Euler lainnya.

```
>f(2)
```

```
       4.4721
```

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

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

```
  Real 1 x 11 matrix

       0.0000       0.1005       0.2040       0.3132      ...
```

Fungsi dapat diplot. Alih-alih menuliskan ekspresi, kita hanya perlu memberikan nama fungsinya.
Berbeda dengan ekspresi simbolik atau numerik, nama fungsi harus diberikan dalam bentuk string.

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

```
       0.7862
```

Secara default, jika Anda ingin menimpa (overwrite) sebuah fungsi bawaan, Anda harus menambahkan kata kunci overwrite. Menimpa fungsi bawaan berisiko dan dapat menyebabkan masalah bagi fungsi lain yang bergantung padanya.

Anda masih dapat memanggil fungsi bawaan tersebut menggunakan _..., jika fungsi tersebut merupakan bagian dari inti Euler.

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

```
       0.7071
```

Sebaiknya kita menghapus redefinisi fungsi sin ini.

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

```
       0.7071
```

## Parameter Default

Fungsi numerik dapat memiliki parameter default.

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

Melewatkan parameter ini akan menggunakan nilai default.

```
>f(4)
```

```
      16.0000
```

Memberikan nilai akan menimpa nilai default.

```
>f(4,5)
```

```
  Function f needs only 1 arguments (got 2)!
  Use: f (x)
  Error in:
  f(4,5) ...
        ^
```

Parameter yang diberikan juga akan menimpa nilai default. Hal ini digunakan oleh banyak fungsi Euler, seperti plot2d dan plot3d.

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

```
Argument a not in parameter list of function f.

Parameters:
x

Error in:
f(4,a=1) ...
        ^
```

Jika sebuah variabel bukan parameter, variabel tersebut harus bersifat global. Fungsi satu baris dapat mengakses variabel global.

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

```
      24.0000
```

Tetapi parameter yang diberikan akan menimpa nilai global.

Jika argumen tidak ada dalam daftar parameter yang sudah ditentukan, argumen tersebut harus dideklarasikan dengan :=!

```
>f(2,a:=5)
```

```
      20.0000
```

Fungsi simbolik didefinisikan dengan &=. Fungsi ini didefinisikan di Euler dan Maxima, dan dapat bekerja di kedua lingkungan tersebut. Ekspresi yang mendefinisikannya dijalankan melalui Maxima sebelum dilakukan definisi.

```
>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, hal ini hanya akan berhasil jika EMT dapat menafsirkan semua yang ada di dalam fungsi tersebut.

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

```
     178.6351
```

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

```
>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)
```

```
       0.7035
```

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

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

```
       0.7035
```

```
>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)
```

```
     625.0000
```

```
>$&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 menjadi simbolik, dan dapat digunakan dalam ekspresi simbolik lainnya.

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

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

Dengan :=, fungsi 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 tersebut dengan kata kunci map, fungsi dapat digunakan untuk vektor x. Secara internal, fungsi dipanggil untuk setiap nilai x sekali, dan hasilnya disimpan dalam sebuah vektor.

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

```
      -0.7834      -0.4108       0.0000       0.6769       2.0504
```

Fungsi dapat memiliki nilai default untuk parameternya.

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

Sekarang fungsi dapat dipanggil dengan atau tanpa parameter base.

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

```
       2.0000
       6.7000
```

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

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

```
       2.0000
```

Seringkali, kita ingin menggunakan fungsi untuk vektor di satu tempat, dan untuk elemen tunggal di tempat lain. Hal ini memungkinkan dengan penggunaan 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 ini dapat digunakan untuk variabel simbolik.
Namun, fungsi tersebut juga dapat digunakan untuk vektor numerik.

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

```
      17.0000
```

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)
```

$$0$$

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

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

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

Kesimpulannya
- &= mendefinisikan fungsi simbolik,
- := mendefinisikan fungsi numerik,
- &&= mendefinisikan fungsi yang 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 membutuhkan nilai awal untuk memulai pencarian solusi. Secara internal, solve() menggunakan metode sekannya (secant method).

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

```
       1.4142
```

Hal ini juga berlaku 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])
```

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

```
>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. Pada solve(), nilai target default y=0 dapat diubah dengan menggunakan variabel yang diberikan (assigned variable).
Kita menggunakan y=2 dan memeriksa hasilnya dengan mengevaluasi polinomial pada solusi sebelumnya.

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

```
       0.9667
       2.0000
```

Menyelesaikan ekspresi simbolik dalam bentuk simbolik akan menghasilkan daftar solusi. Kita menggunakan pemecah 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 numerik adalah dengan mengevaluasi solusi secara numerik 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])
```

$$0$$

Memecahkan sistem persamaan secara simbolis dapat dilakukan dengan vektor persamaan dan penyelesai simbolis solve(). Jawabannya berupa 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, seringkali kita ingin menggunakan parameter lokal.

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

dengan a=3.

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

Salah satu cara untuk meneruskan parameter tambahan ke f() adalah dengan menggunakan daftar dengan nama fungsi dan parameter (cara lainnya adalah parameter titik koma).

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

```
       2.5412
```

Ini juga berlaku untuk ekspresi. Namun, elemen daftar bernama harus digunakan. (Selengkapnya tentang daftar ada di tutorial sintaksis EMT).

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

```
       2.5412
```

## 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])
```

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

```
>$&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])
```

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

Dokumentasi inti EMT berisi pembahasan terperinci tentang bahasa matriks Euler.

Vektor dan matriks dimasukkan dengan tanda kurung siku, elemen dipisahkan dengan koma, baris dipisahkan dengan titik koma.

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

```
                1              2
                3              4
```

Produk matriks dilambangkan dengan titik.

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

```
       3.0000
       4.0000
```

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

```
       3.0000      4.0000
```

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

```
  Variable or function A not found.
  Error in:
  inv(A) //inverse A ...
       ^
```

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

```
  Variable A not found!
  Error in:
  A.b //perkalian matriks ...
                        ^
```

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

```
  Variable or function A not found.
  Error in:
  A.inv(A) ...
         ^
```

Poin utama dari bahasa matriks adalah bahwa semua fungsi dan operator bekerja elemen demi 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)
```

```
       0.333333      0.666667
           0.75             1
```

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

```
              -2
             2.5
```

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

```
              -2
             2.5
```

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

```
               1             0
               0             1
```

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

```
               1             0
               0             1
```

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

```
               1               4
               9              16
```

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

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

```
               9
              16
```

Jika salah satu operan merupakan vektor atau skalar, ia diperluas dengan cara alami.

```
>2*A
```

```
               2               4
               6               8
```

Misalnya, jika operan adalah vektor kolom, elemen-elemennya diterapkan ke semua baris A.

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

```
               1               4
               3               8
```

Jika itu adalah vektor baris, maka diterapkan ke semua kolom A.

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

```
               2               6
               6              12
```

Seseorang dapat membayangkan perkalian ini seolah-olah vektor baris v telah diduplikasi untuk membentuk matriks berukuran sama dengan 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, yang satu merupakan vektor baris dan yang lainnya merupakan vektor kolom. Kita menghitung i*j untuk i,j dari 1 hingga 5. Triknya adalah mengalikan 1:5 dengan transposenya. Bahasa matriks Euler secara otomatis menghasilkan tabel nilai.

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

```
                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
```

Sekali lagi, ingatlah bahwa ini bukan produk matriks!

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

```
      55.0000
```

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

```
      55.0000
```

Bahkan operator seperti < atau == bekerja dengan cara yang sama.

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

```
  Real 1 x 10 matrix

      1.0000      1.0000      1.0000      1.0000     ...
```

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

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

```
       5.0000
```

Euler memiliki operator perbandingan, seperti "==", yang memeriksa kesetaraan.
Kita mendapatkan vektor 0 dan 1, dengan 1 berarti benar.

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

```
  Real 1 x 10 matrix

      0.0000      0.0000      0.0000      0.0000     ...
```

Dari vektor tersebut, "nonzeros" memilih elemen-elemen yang bukan nol.
Dalam hal ini, kita mendapatkan indeks semua elemen yang lebih besar dari 50.

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

```
       8.0000       9.0000      10.0000
```

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

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

```
      64.0000      81.0000     100.0000
```

Sebagai contoh, mari kita cari semua kuadrat angka 1 hingga 1000, yaitu 5 modulo 11 dan 3 modulo 13.e vektor indeks ini untuk mendapatkan nilai yang bersesuaian dalam t.

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

```
  Real 1 x 28 matrix

       4.0000      48.0000      95.0000     139.0000      ...
```

EMT tidak sepenuhnya efektif untuk komputasi integer. EMT menggunakan floating point presisi ganda secara internal. Namun, EMT seringkali sangat berguna.

Kita dapat memeriksa keutamaannya. Mari kita cari tahu, berapa banyak kuadrat ditambah 1 yang merupakan bilangan prima.

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

```
     112.0000
```

Fungsi nonzeros() hanya berfungsi untuk vektor. Untuk matriks, ada mnonzeros().

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

```
       0.765761       0.401188       0.406347       0.267829
        0.13673       0.390567       0.495975       0.952814
       0.548138       0.006085       0.444255       0.539246
```

Ia mengembalikan indeks elemen, yang bukan nol.

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

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

Indeks ini dapat digunakan untuk menetapkan elemen pada nilai tertentu.

```
>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 mengatur elemen pada indeks ke entri matriks lainnya.

```
>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 adalah mungkin untuk mendapatkan elemen-elemen dalam sebuah vektor.

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

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

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

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

```
  Variable or function A not found.
  Error in:
  ex=extrema(A) ...
              ^
```

Kita dapat menggunakan ini untuk mengekstrak nilai maksimal pada 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]
```

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

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

```
              1              1
              2              4
              3              1
  [-0.765761,  -0.952814,  -0.548138]
```

Untuk membangun matriks, kita dapat menumpuk satu matriks di atas matriks lainnya. Jika keduanya 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
```

Dengan cara yang sama, kita dapat menempelkan suatu matriks ke sisi lain yang berdampingan, jika keduanya 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 jumlah barisnya tidak sama, matriks yang lebih pendek akan diisi dengan 0.

Ada pengecualian untuk aturan ini. Bilangan riil yang melekat pada suatu 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 kolom.

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

```
       3.0000       4.0000
       3.0000       4.0000
```

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

```
       3.0000       3.0000
       4.0000       4.0000
```

Tujuan utama dari ini adalah untuk menafsirkan vektor ekspresi untuk vektor kolom.

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

```
       3.0000       9.0000
       4.0000      16.0000
```

Untuk mendapatkan ukuran A, kita dapat menggunakan fungsi berikut.

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

```
       2.0000
       4.0000
       2.0000       4.0000
       4.0000
```

Untuk vektor, ada length().

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

```
       9.0000
```

Ada banyak fungsi lain yang menghasilkan matriks.

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

```
       1.0000       1.0000
       1.0000       1.0000
```

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

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

```
       6.0000       6.0000       6.0000       6.0000       6.0000
```

Matriks bilangan acak juga dapat dihasilkan dengan acak (distribusi seragam) atau normal (distribusi Gauß).

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

```
       0.6554       0.2010
       0.8936       0.2819
```

Berikut adalah fungsi berguna lainnya, yang merestrukturisasi elemen-elemen suatu matriks menjadi matriks lain.

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

```
       1.0000       2.0000       3.0000
       4.0000       5.0000       6.0000
       7.0000       8.0000       9.0000
```

Dengan fungsi berikut, kita dapat menggunakan ini dan fungsi dup untuk menulis fungsi rep(), yang mengulang vektor n kali.

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

Mari kita uji.

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

```
  Real 1 x 15 matrix

       1.0000       2.0000       3.0000       1.0000      ...
```

Fungsi multdup() menduplikasi elemen vektor.

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

```
  Real 1 x 15 matrix

       1.0000       1.0000       1.0000       1.0000      ...
  Real 1 x 7 matrix

       1.0000       1.0000       2.0000       2.0000      ...
```

Fungsi flipx() dan flipy() membalikkan urutan baris atau kolom matriks. Dengan kata lain, fungsi flipx() membalik secara horizontal.

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

```
       5.0000       4.0000       3.0000       2.0000       1.0000
```

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

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

```
       2.0000       3.0000       4.0000       5.0000       1.0000
```

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

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

```
  Real 1 x 10 matrix

      10.0000      11.0000      13.0000      14.0000      ...
```

Perhatikan bahwa vektor i dalam drop(v,i) merujuk pada indeks elemen dalam v, bukan nilai elemennya. Jika Anda ingin menghapus elemen, Anda perlu menemukan elemennya terlebih dahulu. Fungsi indexof(v,x) dapat digunakan untuk menemukan elemen x dalam vektor v yang telah diurutkan.

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

```
  Real 1 x 15 matrix

       2.0000       3.0000       5.0000       7.0000      ...
  Real 1 x 11 matrix

       0.0000       5.0000       0.0000       6.0000      ...
  Real 1 x 11 matrix

       2.0000       3.0000       5.0000       7.0000      ...
```

Seperti yang Anda lihat, tidak ada salahnya menyertakan indeks di luar rentang (seperti 0), indeks ganda, atau indeks yang tidak diurutkan.

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

```
  Real 1 x 8 matrix

       1.0000       2.0000       3.0000       4.0000      ...
```

Ada beberapa fungsi khusus untuk mengatur diagonal atau menghasilkan matriks diagonal.
Kita mulai dengan matriks identitas.

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

```
       1.0000       0.0000       0.0000       0.0000       0.0000
       0.0000       1.0000       0.0000       0.0000       0.0000
       0.0000       0.0000       1.0000       0.0000       0.0000
       0.0000       0.0000       0.0000       1.0000       0.0000
       0.0000       0.0000       0.0000       0.0000       1.0000
```

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

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

```
       1.0000       0.0000       0.0000       0.0000       0.0000
       1.0000       1.0000       0.0000       0.0000       0.0000
       0.0000       2.0000       1.0000       0.0000       0.0000
       0.0000       0.0000       3.0000       1.0000       0.0000
       0.0000       0.0000       0.0000       4.0000       1.0000
```

Perhatikan bahwa kita tidak mengubah matriks A. Kita mendapatkan matriks baru sebagai hasil dari setdiag().
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)
```

```
2.0000    3.0000    0.0000    0.0000    0.0000
1.0000    2.0000    3.0000    0.0000    0.0000
0.0000    1.0000    2.0000    3.0000    0.0000
0.0000    0.0000    1.0000    2.0000    3.0000
0.0000    0.0000    0.0000    1.0000    2.0000
```

Diagonal suatu matriks juga dapat diekstraksi dari matriks tersebut. Untuk mendemonstrasikan hal ini, kami merestrukturisasi vektor 1:9 menjadi matriks 3x3.

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

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

Sekarang kita dapat mengekstrak diagonalnya.

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

```
    1.0000    1.0000    1.0000    1.0000    1.0000
```

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

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

```
     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
```

## Vektorisasi

Hampir semua fungsi di Euler juga berfungsi untuk input matriks dan vektor, jika diperlukan.
Misalnya, fungsi sqrt() menghitung akar kuadrat dari semua elemen vektor atau matriks.

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

```
    1.0000    1.4142    1.7321
```

Jadi, Anda dapat dengan mudah membuat tabel nilai. Ini adalah salah satu cara untuk memplot fungsi (alternatifnya menggunakan ekspresi).

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

Dengan ini dan operator titik dua a:delta:b, vektor nilai fungsi dapat dihasilkan dengan mudah.

Dalam contoh berikut, kita menghasilkan vektor nilai t[i] dengan spasi 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
```

```
  Real 1 x 21 matrix

       0.0000       0.1710       0.2880       0.3570      ...
```

EMT mengekspansi operator untuk skalar, vektor, dan matriks dengan cara yang mudah dipahami.

Misalnya, vektor kolom dikalikan vektor baris akan mengekspansi menjadi matriks jika operator 1 diterapkan. Selanjutnya, v' adalah vektor yang ditransposisikan (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
```

Perhatikan bahwa ini sangat berbeda dari perkalian matriks. Perkalian matriks dilambangkan dengan titik "." dalam EMT.

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

```
      55.0000
```

Secara default, vektor baris dicetak dalam format kompak.

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

```
       1.0000       2.0000       3.0000       4.0000
```

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

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

```
       5.0000
      25.0000
```

Untuk mentransposisi matriks, kita menggunakan tanda apostrof.

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

```
       1.0000
       2.0000
       3.0000
       4.0000
```

Jadi kita dapat menghitung matriks A dikali vektor b.

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

```
      30.0000
      70.0000
```

Perhatikan bahwa v masih merupakan vektor baris. Jadi, v'.v berbeda dari v.v'.

```
>v'.v
```

```
       1.0000       2.0000       3.0000       4.0000
       2.0000       4.0000       6.0000       8.0000
       3.0000       6.0000       9.0000      12.0000
       4.0000       8.0000      12.0000      16.0000
```

v.v' menghitung norma v kuadrat untuk vektor baris v. Hasilnya adalah vektor 1x1, yang bekerja seperti bilangan riil.

```
>v.v'
```

```
      30.0000
```

Ada juga norma fungsi (bersama dengan banyak fungsi Aljabar Linear lainnya).

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

```
      30.0000
```

Operator dan fungsi mematuhi bahasa matriks Euler.

Berikut ringkasan aturannya.

- Fungsi yang diterapkan pada vektor atau matriks diterapkan pada setiap elemennya.

- Operator yang beroperasi pada dua matriks berukuran sama diterapkan berpasangan pada elemen-elemen matriks tersebut.

- Jika kedua matriks memiliki dimensi yang berbeda, keduanya diekspansi dengan cara yang masuk akal, sehingga memiliki ukuran yang sama.

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

Berikut adalah kasus sederhana dengan operator ^.

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

```
       1.0000       4.0000       9.0000
```

Berikut kasus yang lebih rumit. Sebuah vektor baris dikalikan dengan vektor kolom akan mengembang keduanya dengan cara menduplikasi.

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

```
       1.0000       2.0000       3.0000
       2.0000       4.0000       6.0000
       3.0000       6.0000       9.0000
```

Perhatikan bahwa produk skalar menggunakan produk matriks, bukan *!

```
>v.v'
```

```
      14.0000
```

Ada banyak fungsi untuk matriks. Kami memberikan daftar singkatnya. Anda sebaiknya merujuk ke dokumentasi untuk informasi lebih lanjut tentang perintah-perintah ini.

sum,prod menghitung jumlah dan hasil kali baris-baris
cumsum,cumprod melakukan hal yang sama secara kumulatif

menghitung nilai ekstrem setiap baris
extrema mengembalikan vektor dengan informasi ekstrem
diag(A,i) mengembalikan diagonal ke-i
setdiag(A,i,v) menetapkan diagonal ke-i
id(n) matriks identitas
det(A) determinan
charpoly(A) polinomial karakteristik
eigenvalues(A) nilai eigen

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

```
       1.0000       4.0000       9.0000
      14.0000
       1.0000       5.0000      14.0000
```

Operator : menghasilkan vektor baris dengan spasi yang sama, secara opsional dengan ukuran langkah.

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

```
       1.0000       2.0000       3.0000       4.0000
       1.0000       3.0000       5.0000       7.0000       9.0000
```

Untuk menggabungkan matriks dan vektor ada operator "|" dan "_".

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

```
       1.0000       2.0000       3.0000       4.0000       5.0000
       1.0000       2.0000       3.0000
       1.0000       1.0000       1.0000
```

Elemen-elemen suatu matriks disebut dengan "A[i,j]".

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

```
       6.0000
```

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

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

```
       6.0000
       7.0000      8.0000      9.0000
```

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

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

```
       2.0000      4.0000
       2.0000
       5.0000
       8.0000
```

Bentuk singkat dari : adalah menghilangkan indeks sepenuhnya.

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

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

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

```
>A{4}
```

```
       4.0000
```

Matriks juga dapat diratakan menggunakan fungsi redim(). Fungsi 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 pada tabel, mari kita kembalikan ke format default, dan hitung tabel nilai sinus dan kosinus. Perhatikan bahwa sudut menggunakan radian secara default.

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

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

Sekarang kita tambahkan kolom ke 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 menggunakan bahasa matriks, kita dapat menghasilkan beberapa tabel dari beberapa fungsi sekaligus.

Dalam contoh berikut, kita menghitung t[j]^i untuk i dari 1 hingga n. Kita mendapatkan sebuah matriks, di mana setiap barisnya merupakan tabel t^i untuk satu i. Dengan kata lain, matriks tersebut memiliki elemen-elemen lateks: a_{i,j} = t_j^i, \quad 1 \le j \le 101, \quad 1 \le i \le n

Fungsi yang tidak berfungsi untuk input vektor harus "divektorkan". Hal ini dapat dicapai dengan kata kunci "map" dalam definisi fungsi. Kemudian, fungsi tersebut akan dievaluasi untuk setiap elemen parameter vektor.

Integrasi numerik 'integrate()' hanya berfungsi untuk batas interval skalar. Jadi, kita perlu memvektorkannya.

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

Kata kunci "map" akan memvektorisasi fungsi tersebut. Fungsi ini sekarang akan berfungsi 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.

```
>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 satu baris matriks yang lengkap.

```
>A[2]
```

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

Dalam kasus vektor baris atau kolom, ini mengembalikan elemen vektor.

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

```
  2
```

Untuk memastikan Anda mendapatkan baris pertama untuk matriks 1xn dan mxn, tentukan semua kolom 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 menginginkan baris pertama dan kedua dari A.

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

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

Kita bahkan dapat menyusun ulang A menggunakan vektor indeks. Lebih tepatnya, kita tidak mengubah A di sini, melainkan menghitung versi A yang telah disusun ulang.

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

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

Trik indeks juga berfungsi dengan kolom.
Contoh ini memilih semua baris A, kolom kedua, dan ketiga.

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

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

Untuk singkatan ":" menunjukkan semua indeks baris atau kolom.

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

```
3
6
9
```

Atau, biarkan indeks pertama kosong.

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

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

Kita juga bisa mendapatkan baris terakhir A.

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

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

Sekarang mari kita ubah elemen A dengan menetapkan submatriks A ke suatu nilai. Hal ini sebenarnya mengubah matriks A yang tersimpan.

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

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

Kita juga dapat menetapkan nilai ke baris A.

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

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

Kita bahkan dapat menetapkannya ke sub-matriks jika ukurannya tepat.

```
>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 yang melebihi batas akan menghasilkan matriks kosong, atau pesan kesalahan, tergantung pada pengaturan sistem. Standarnya adalah pesan kesalahan. Namun, perlu diingat bahwa indeks negatif dapat digunakan untuk mengakses elemen matriks yang dihitung dari akhir.

```
>A[4]
```

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

## Pengurutan dan Pengacakan

Fungsi sort() mengurutkan vektor baris.

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

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

Seringkali perlu mengetahui indeks vektor yang telah diurutkan dalam vektor asli. Ini dapat digunakan untuk mengurutkan ulang vektor lain dengan cara yang sama.

Mari kita acak sebuah vektor.

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

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

Seringkali perlu mengetahui indeks vektor yang telah diurutkan dalam vektor asli. Ini dapat digunakan untuk mengurutkan ulang vektor lain dengan cara yang sama.

Indeks tersebut memuat urutan v yang tepat.

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

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

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 unik mengembalikan daftar terurut dari elemen unik suatu vektor.

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

```
  [3,  8,  5,  4,  4,  2,  5,  3,  6,  8]
  [2,  3,  4,  5,  6,  8]
```

Ini juga berlaku untuk vektor string.

```
>unique(s)
```

```
  a
  aa
  d
  e
```

## Aljabar Linear

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 pencocokan linear. Operator A\b menggunakan versi algoritma Gauss.

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

```
      -4
     4.5
```

Untuk contoh lain, kita buat matriks 200x200 dan jumlah barisnya. Kemudian, kita selesaikan Ax=b menggunakan matriks invers. Kita ukur galat sebagai deviasi maksimum semua elemen dari 1, yang tentu saja merupakan solusi yang tepat.

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

```
    6.812328479099961e-13
```

Jika sistem tidak mempunyai solusi, penyesuaian linear meminimalkan norma kesalahan Ax-b.

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

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

Determinan matriks ini adalah 0.

```
>det(A)
```

```
  0
```

## Matriks Simbolik

Maxima memiliki matriks simbolik. Tentu saja, Maxima dapat digunakan untuk soal-soal aljabar linear sederhana seperti itu. Kita dapat mendefinisikan matriks untuk Euler dan Maxima dengan &:=, lalu menggunakannya dalam ekspresi simbolik. Bentuk [...] yang umum untuk mendefinisikan matriks dapat digunakan dalam Euler untuk mendefinisikan 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
```

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

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

```
  Maxima said:
  fullmap: arguments must have same formal structure.
   -- an error. To debug this try: debugmode(true);

  Error in:
   $&det(A-x*ident(2)), $&solve(%,x) ...
                        ^
```

Nilai eigen juga dapat dihitung secara otomatis. Hasilnya adalah sebuah vektor dengan dua vektor nilai eigen dan multiplisitas.

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

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

Untuk mengekstrak vektor eigen tertentu dibutuhkan pengindeksan 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 dalam Euler secara numerik seperti ekspresi simbolik lainnya.

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

```
                4               1               1
                1               4               1
                1               1               4
```

Dalam ekspresi simbolik, gunakan dengan.

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

$$\begin{pmatrix} t+1 & 1 & 1 \\ 1 & 4 & 1 \\ 1 & 1 & 4 \end{pmatrix}$$

Akses ke baris matriks simbolik bekerja seperti halnya matriks numerik.

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

$$[t+1, 1, 1]$$

Ekspresi simbolis dapat berisi suatu penugasan. Dan hal itu mengubah matriks A.

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

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

Terdapat fungsi simbolis di Maxima untuk membuat vektor dan matriks. Untuk informasi ini, silakan merujuk ke 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)
```

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

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

```
  Maxima said:
  Maxima encountered a Lisp error:

   The value |$b| is not of type LIST.
```

```
Automatically continuing.
To enable the Lisp debugger set *debugger-hook* to nil.

Error in:
 $&invert(B)() ...
           ^
```

Euler juga memiliki fungsi xinv() yang canggih, yang membutuhkan upaya lebih besar dan menghasilkan hasil yang lebih tepat.

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

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

```
Variable or function B not found.
Error in:
longest B.xinv(B) ...
                ^
```

Misalnya nilai E.g. A dapat dihitung secara numerik.

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

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

Atau secara simbolis. Lihat tutorial tentang Maxima untuk detailnya.

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

$$[[a+2, a-1], [1, 2]]$$

## Nilai Numerik dalam Ekspresi Simbolik

Ekspresi simbolik hanyalah string yang berisi ekspresi. Jika kita ingin mendefinisikan nilai untuk ekspresi simbolik dan ekspresi numerik, kita harus menggunakan "&:=".

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

```
                1        3.14159
                4              5
```

Masih terdapat perbedaan antara bentuk numerik dan bentuk simbolik. Saat mengubah matriks ke bentuk simbolik, pendekatan pecahan untuk bilangan riil akan digunakan.

```
>$&A
```

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

Untuk menghindari hal ini, ada fungsi "mxmset(variabel)".

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

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

Maxima juga dapat melakukan komputasi dengan bilangan floating point, bahkan dengan bilangan floating point besar dengan 32 digit. Namun, evaluasinya jauh lebih lambat.

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

$$1.414213562373095$$

Ketepatan angka floating point yang besar dapat diubah.

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

```
          3.14159265358979323846264338327950288419716939937510582097494\
  459230781640628620899862803482534211706 8b0
```

Variabel numerik dapat digunakan dalam ekspresi simbolik apa pun yang menggunakan "@var".
Perhatikan 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, kami menggunakan Euler Math Toolbox (EMT) untuk menghitung suku bunga. Kami melakukannya secara numerik dan simbolis untuk menunjukkan kepada Anda bagaimana Euler dapat digunakan untuk menyelesaikan masalah kehidupan nyata.
Asumsikan Anda memiliki modal awal sebesar 5000 (misalnya dalam dolar).

```
>K=5000
```

```
  5000
```

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

```
>K*1.03
```

```
  5150
```

Euler juga akan memahami sintaksis berikut.

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

```
  5150
```

Namun lebih mudah menggunakan faktor

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

```
  1.03
  5150
```

Selama 10 tahun, kita cukup mengalikan faktor-faktornya 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
```

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

```
>"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 ke-1 hingga ke-9? Untuk itu, bahasa matriks Euler sangat membantu. Anda tidak perlu menulis perulangan, cukup masukkan

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

```
  Real 1 x 11 matrix

     5000.00      5150.00      5304.50      5463.64      ...
```

Bagaimana keajaiban ini bekerja? Pertama, ekspresi 0:10 menghasilkan vektor bilangan bulat.

```
>short 0:10
```

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

Maka semua operator dan fungsi di Euler dapat diterapkan pada 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]
```

adalah vektor faktor q^0 hingga q^10. Ini dikalikan dengan K, dan kita mendapatkan vektor nilai.

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

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

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

Mari kita bandingkan kedua hasil, 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, dan kita harus mengulangnya selama bertahun-tahun. Euler menyediakan banyak solusi untuk ini.

Cara termudah adalah dengan fungsi iterate, yang mengiterasi fungsi yang diberikan 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 ramah, menggunakan format kami dengan tempat 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 vektor tertentu, kita menggunakan indeks dalam tanda kurung siku.

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

```
      5150.00
      5000.00     5150.00     5304.50
```

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

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

```
      6719.60
      6719.58
```

Selisihnya sangat kecil.

## Menyelesaikan Persamaan

Sekarang kita ambil fungsi yang lebih maju, yang menambahkan tingkat uang tertentu setiap tahun.

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

Kita tidak perlu menentukan q atau R untuk definisi fungsi. Kita hanya perlu menentukan nilai-nilai ini jika kita menjalankan perintah 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 menghilangkan 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 uangnya berkurang. Jelas, jika kita hanya mendapatkan bunga 150 di tahun pertama, tetapi mengurangi 200, kita kehilangan uang setiap tahun.

Bagaimana kita bisa menentukan berapa tahun uang itu akan bertahan? Kita harus menulis perulangan untuk ini. Cara termudah adalah dengan melakukan iterasi yang 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 nonzeros(VKR<0) mengembalikan vektor indeks i, dengan VKR[i]<0, dan min menghitung indeks minimal.

Karena vektor selalu dimulai dengan indeks 1, jawabannya adalah 47 tahun.

Fungsi iterate() memiliki satu trik lagi. Fungsi ini dapat mengambil kondisi akhir sebagai argumen. Kemudian, fungsi ini 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. Asumsikan kita tahu bahwa nilainya 0 setelah 50 tahun. Berapa tingkat bunganya?

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

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

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

Iterasinya sama seperti di atas

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

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

Selain itu, kita hanya tertarik pada nilai terakhir. Jadi, kita mengambil indeks [-1].

Mari kita coba uji coba.

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

```
       -19.83
```

Sekarang kita bisa memecahkan masalah kita.

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

```
         3.15
```

Rutin solve menyelesaikan ekspresi = 0 untuk variabel x. Jawabannya adalah 3,15% per tahun. Kita mengambil nilai awal 3% untuk algoritma tersebut. Fungsi solve() selalu membutuhkan nilai awal.

Kita dapat menggunakan fungsi yang sama untuk menyelesaikan pertanyaan berikut: Berapa banyak yang dapat kita hapus per tahun agar modal awal habis setelah 20 tahun dengan asumsi suku bunga 3% per tahun.

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

```
      -336.08
```

Perhatikan bahwa Anda tidak dapat menyelesaikan masalah jumlah tahun, karena fungsi kita mengasumsikan n sebagai nilai integer.

## Solusi Simbolis untuk Masalah Suku Bunga

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

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

$$R + q\,K$$

Sekarang kita dapat mengulanginya.

```
>$&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 memiliki

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

Rumus tersebut adalah rumus jumlah geometri yang diketahui 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 agak rumit. Jumlahnya dievaluasi dengan tanda "simpsum" untuk menyederhanakannya menjadi hasil bagi.

Mari kita buat 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 f sebelumnya. Namun, fungsinya lebih efektif.

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

```
       -19.82504734650985
       -19.82504734652684
```

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

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

```
        20.51
```

Jawaban ini menyatakan bahwa hasilnya akan negatif setelah 21 tahun.

Kita juga dapat menggunakan sisi simbolis Euler untuk menghitung rumus pembayaran.

Asumsikan kita mendapatkan pinjaman sebesar K, dan membayar n kali cicilan sebesar R (dimulai setelah tahun pertama) sehingga menyisakan utang sebesar Kn (pada saat pembayaran terakhir). Rumus untuk hal 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 diberikan 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 menghitung laju R secara simbolis.

```
>$&solve(equ,R)
```

$$\left[R=\frac{i\,Kn-i\,(i+1)^n\ K}{(i+1)^n-1}\right]$$

Seperti yang Anda lihat dari rumus, fungsi ini mengembalikan kesalahan floating point untuk i=0. Namun, Euler memplotnya.
Tentu saja, kita memiliki limit 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 lipat dari 500.
Persamaan ini juga dapat diselesaikan untuk n. Akan terlihat lebih baik jika kita menyederhanakannya.

```
>fn &= solve(equ,n) | ratsimp; $&fn
```

$$\left[n=\frac{\log\left(\frac{R+i\,Kn}{R+i\,K}\right)}{\log\,(i+1)}\right]$$

```
/* deklarasikan variabel sebagai konstan */
declare([a,b,c], constant);
/* deklarasikan variabel sebagai konstan */
declare([a,b,c], constant);
```

## Latihan Soal

## Soal R.2

1) Hitunglah 2^6*2^-3/2^10/2^-8

```
>2^6*2^-3/2^10/2^-8
```

```
  2
```

Penjelasan:
2^6*2^-3/2^10/2^-8
= 2^6*2^(-3-10-(-8))
= 2^6*2^(-5)
= 2^1
= 2

2)Hitunglah hasil dari (4*(8-6)^2-4*3+2*8)/(3^1+19^0)

```
>(4*(8-6)^2-4*3+2*8)/(3^1+19^0)
```

```
  5
```

Penjelasan
Hitung kurung:8-6=2
Pangkat:2^2 =4
Kali dengan 4:4*4 =16
Kurangi 4*3 =12 -> 16-12 =4
Tambah 2*8 = 16-> 4+16=20
Penyebut 3^1 +19^0 = 3+1 = 4
Hasil 20/4=5

3) Hitunglah hasil dari ((24a^10*b^-8*c^7)/(12a^6*b^-3*c^5))^-5

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

$$\frac{b^{25}}{32\,a^{20}\,c^{10}}$$

Penjelasan:
Koefisien: 24/12=2
Pangkat: a^(10-6)=a^4, b^(-8-(-3))=b^(-5),c^(7-5)=c^2
hasil: (2*a^4*b^(-5)*c^2)^(-5)
hasil:2^(-5)*a(-20)*b^25*c^(-10) = b^25/32*a^20*c^10

4) Hitunglah hasil dari (4*(8-6)^2 + 4)(3-2*8))/(2^2(2^3+5))

```
>((4*(8-6)^2 + 4)*(3 - 2*8)) / (2^2 * (2^3 + 5))
```

```
  -5
```

Penjelasan
Kurung pertama: 8-6=2
Pangkat: 2^2 = 4
Kali 4: 4*4 = 16
Tambah 4: 16+4 = 20
Kurung kedua: 3-2*8=3-16=-13
Perkalian pembilang: 20*(-13)=-260
Penyebut: 2^2*(2^3+5)=4*(8+5)=4*13=52
Pembagian terakhir: -260/52=-5
Hasil: -5

5) Hitunglah hasil dari (((125*p^12*q^(-14)*r^22)/(25*p^8*q^6*r^(-15)))^(-4))

```
>$ratsimp(((125*p^12*q^(-14)*r^22)/(25*p^8*q^6*r^(-15)))^(-4))
```

$$\frac{q^{80}}{625\,p^{16}\,r^{148}}$$

## Soal R.3

1) Hituglah hasil dari(x+3)^2

```
>&expand((x+3)^2)
```

```
   2
x  + 6 x + 9
```

Penjelasan
(x+3)^2
=(x+3)*(x+3)
=x^2+3x+3x+9
=x^2+6x+9

2) Hitunglah hasil dari (y-5)^2

```
>&expand((y-5)^2)
```

```
   2
y  - 10 y + 25
```

Penjelasan:
(y-5)^2
=(y-5)(y-5)
=y^2-10y+25

3) Hitunglah hasil dari (2*x+3*y+z-7)+(4*x-2*y-z+8)+(-3*x+y-2*z-4)

```
>&expand((2*x+3*y+z-7)+(4*x-2*y-z+8)+(-3*x+y-2*z-4))
```

```
- 2 z + 2 y + 3 x - 3
```

Penjelasan
(2*x+3*y+z-7)+(4*x-2*y-z+8)+(-3*x+y-2*z-4)
=(2*x+4*x-3*x)+(3*y-2*y+y)+(z-z-2*z)+(-7+8-4)
=3*x+2*y-2*z-3

4) Hitunglah hasil dari (x+6)*(x+3)

```
>&expand((x+6)*(x+3))
```

```
   2
x  + 9 x + 18
```

Penjelasan
(x+6)*(x+3)
=x^2+6*x+3*x+6*3
=x^2+9*x+18

5) Hitunglah hasil dari (a+b+c)^2

```
>&expand((a+b+c)^2)
```

```
  2                     2             2
c  + 2 b c + 2 a c + b  + 2 a b + a
```

Penjelasan
(a+b+c)^2
=(a+b+c)*(a+b+c)
=a^2+b^2+c^2+2bc+2ba+2ac

## Soal R.4

Tentukan faktor dari persamaan dibawah ini

```
>&factor(t^2+8*t+15)
```

```
(t + 3) (t + 5)
```

```
>&factor(z^2-81)
```

```
(z - 9) (z + 9)
```

```
>&factor(x^3+64)
```

$$(x + 4)\ (x^2 - 4\ x + 16)$$

```
>&factor(m^2-4)
```

$$(m - 2)\ (m + 2)$$

```
>&factor(p-64*p^4)
```

$$-\ p\ (4\ p - 1)\ (16\ p^2 + 4\ p + 1)$$

## Soal R.5

Tentukan nilai x dari persamaan-persamaan berikut ini!

```
>&solve(7*(3*x+6)=11-(x+2))
```

$$\left[x = -\frac{3}{2}\right]$$

```
>&solve(x^2-36=0)
```

$$[x = -6,\ x = 6]$$

```
>&solve(x^2=25)
```

$$[x = -5,\ x = 5]$$

```
>&solve(2*x^2-20=0)
```

```
[x = - sqrt(10), x = sqrt(10)]
```

```
>&solve(x^2+100=20*x)
```

```
[x = 10]
```

## Soal R.6

Tentukan hasil dari persamaan rasional berikut ini !

```
>&ratsimp((x^2-4)/(x^2-4*x+4))
```

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

```
>&ratsimp((6-x)/(x^2-36))
```

$$-\frac{1}{x + 6}$$

```
>&ratsimp((7/5*x)+(3/5*x))
```

$$2\,x$$

```
>&ratsimp((x/2*x-(3*y))+(y/3*y-(2*y)))
```

$$\frac{2\,y^2 - 30\,y + 3\,x^2}{6}$$

```
>&ratsimp ((a-a^(-1))/(a+a^(-1)))
```

$$\frac{a^2 - 1}{a^2 + 1}$$

## Soal 2.3

Diketahui f(x)=3*x+1, g(x)=x^2-2*x-6, dan h(x)=x^3, Tentukan hasil dari berikut ini!

```
>f &= 3*x+1
```

$$3\,x + 1$$

```
>g &= x^2-2*x-6
```

$$x^2 - 2\,x - 6$$

```
>h &= x^3
```

$$x^3$$

1) Tentukan f(g(-1))

```
>f(g(-1))
```

```
  -8
```

2) Tentukan h(f(1))

```
>h(f(1))
```

```
  64
```

3) Tentukan f(f(-4))

```
>f(f(-4))
```

```
  -32
```

4) Tentukan f(h(-3))

```
>f(h(-3))
```

```
  -80
```

5) Tentukan g(g(-2))

```
>g(g(-2))
```

```
  -6
```

## Soal 3.1

Tulis jawaban dalam bentuk a+bi, di mana a dan b adalah bilangan riil.

```
>&rectform((-6 - 5*i) + (9 + 2*i))
```

$$3 - 3\,i$$

```
>&rectform((7 - 2*i) + (4 - 5*i))
```

$$11 - 7\,i$$

```
>&rectform((3 + sqrt(-16)) + (2 + sqrt(-25)))
```

$$9\,I + 5$$

```
>&rectform((sqrt(-16))*(sqrt(-100)))
```

$$-40$$

```
>&rectform((sqrt(-4))*(sqrt(-56)))
```

```
- 4 sqrt(14)
```

## Soal 3.4

Tentukan hasil penyelesaian pada soal-soal berikut!

```
>&solve((1/4)+(1/5)=(1/t))
```

```
     20
[t = --]
     9
```

```
>&solve(((x+2)/4)-((x-1)/5)= 15)
```

```
[x = 286]
```

```
>&solve(((1)/(x-6))-((1)/x)=((6)/(x^2-6*x)))
```

```
[x = x]
```

```
>&solve((x)-((12)/x)=1)
```

```
[x = - 3, x = 4]
```

```
>&solve((sqrt(3*x)-4)=1)
```

```
     25
[x = --]
     3
```

## Soal 3.5

Tentukan daerah penyelesaiannya!
1) |x+3|-2 = 8

```
>&solve(abs(x+3)-2=8, x)
```

```
[mabs(x + 3) = 10]
```

```
>&solve(x+3=10, x)
```

```
[x = 7]
```

```
>&solve(x+3=-10, x)
```

```
[x = - 13]
```

2) |4*x-3|+1=7

```
>&solve(abs(4*x-3)+1 = 7, x)
```

```
[mabs(4 x - 3) = 6]
```

```
>&solve((4*x-3=6, x+3=-8), x)
```

```
[x = - 11]
```

## Soal 4.1

1) Gunakan substitusi untuk menentukan apakah 4, 5, dan -2
adalah hasil 0 dari f(x)=x^3-9*x^2+14*x+24

```
>f &= x^3-9*x^2+14*x+24
```

$$x^3 - 9\,x^2 + 14\,x + 24$$

```
>f(4), f(5), f(-2)
```

```
  0
  -6
  -48
```

Temukan nol dari fungsi polinomial dan nyatakan
multiplisitas masing-masing.

2)cari akar (nol) polinomial dan multiplisitasnya dari f(x) = (x^2+5*x+6)^2

```
>f &= (x^2 + 5*x + 6)^2
```

$$(x^2 + 5\,x + 6)^2$$

```
>&factor(f)
```

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

```
>&solve(f=0, x)\cari akar-akar nol
```

$$[x = -3,\ x = -2]$$

Hasilnya menunjukkan bahwa nol fungsi adalah -2 dan -3, masing-masing dengan multiplisitas 2.

## Soal 4.3

1) (2x^4 + 7x^3 + x - 12) ÷ (x + 3)

```
>&divide(2*x^4 + 7*x^3 + x - 12, x+3, x)
```

$$[2\,x^3 + x^2 - 3\,x + 10,\ -42]$$

Penjelasan
Hasil bagi = 2*x^3+x^2-3*x+10
Sisa= -42

2)(x^3 - 7*x^2 + 13*x + 3, x-2, x)

```
>&divide(x^3 - 7*x^2 + 13*x + 3, x-2, x)
```

$$[x^2 - 5\,x + 3,\ 9]$$

Penjelasan
Hasil bagi = x^2-5*x+3
Sisa= 9

3)(x^3 - 2*x^2 - 8, x+2, x)

```
>&divide(x^3 - 2*x^2 - 8, x+2, x)
```

$$[x^2 - 4\,x + 8,\ -24]$$

Penjelasan
Hasil bagi = x^2-4*x+8
Sisa= -24

4)(x^3 - 3*x + 10, x-2, x)

```
>&divide(x^3 - 3*x + 10, x-2, x)
```

$$[x^2 + 2\,x + 1,\ 12]$$

Penjelasan
Hasil bagi = x^2+2*x+1
Sisa= 12

5) f(x)= x^3 - 6*x^2 + 11*x - 6, mengehitung f(1), f(2), f(3)

```
>f &= x^3 - 6*x^2 + 11*x - 6
```

$$x^3 - 6x^2 + 11x - 6$$

```
>f(1)
```

```
  0
```

```
>f(-2)
```

```
  -60
```

```
>f(3)
```

```
  0
```

```
>&factor(f)
```

$$(x - 3)\ (x - 2)\ (x - 1)$$

Penjelasan
(x - 3) (x - 2) (x - 1): menunjukkan 1 dan 3 adalah nol sesuai hasil
Hasil:
f(1) = 0
f(-2) = -60
f(3) = 0