

MAKALAH

PROGRAM KOMPUTER UNTUK
MANIPULASI MATEMATIKA *)

MILIK PERPUSTAKAAN IKIP PADANG	
DITERIMA TGL. :	24 SEP 1997
SUMBER / HARGA :	K. /
KOLEKSI :	K. K. I
NO. INVENTARIS :	1509/K/97 - 10 (2)
KLASIFIKASI :	001.642 R12 1

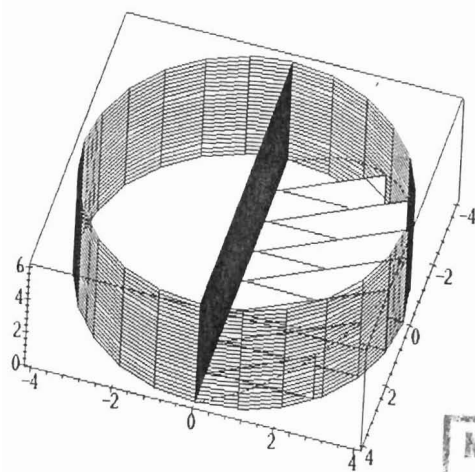
Oleh:
Drs. Yusmet Rizal

JURUSAN PENDIDIKAN MATEMATIKA
FAKULTAS PENDIDIKAN MATEMATIKA DAN IPA
INSTITUT KEGURUAN DAN ILMU PENDIDIKAN PADANG
1997

*) Diseminarkan pada Pertemuan Mingguan Jurusan Pendidikan Matematika
FPMIPA IKIP Padang Tanggal 15 Januari 1997



MAKALAH
PROGRAM KOMPUTER UNTUK
MANIPULASI MATEMATIKA *)



Oleh:
Drs. Yusmet Rizal

MILIK PERPUSTAKAAN IKIP PADANG	
DITERIMA TGL. :	_____
SUMBER / HARGA :	_____ / _____
KOLEKSI :	_____
NO. INVENTARIS :	_____
NO. KOPISI :	_____

JURUSAN PENDIDIKAN MATEMATIKA
FAKULTAS PENDIDIKAN MATEMATIKA DAN IPA
INSTITUT KEGURUAN DAN ILMU PENDIDIKAN PADANG
1997

*) Diseminarkan pada Pertemuan Mingguan Jurusan Pendidikan Matematika
FPMIPA IKIP Padang Tanggal 15 Januari 1997

MILIK UPT PERPUSTAKAAN
IKIP PADANG

BERKENALAN DENGAN MAPLE

1. Pendahuluan

Pada umumnya program komputer hanya dapat mengolah bilangan dan karakter saja. Kalaupun ada program yang mampu mengolah bilangan dan karakter dalam berbagai operasi, tetapi itupun kemampuannya sangat terbatas sekali. Maple adalah suatu program komputer yang mampu mengkombinasikan bilangan dan karakter dalam berbagai bentuk serta mengoperasikannya, dari bentuk yang sederhana sampai kepada bentuk yang rumit. Maple adalah manipulation mathematical language, yaitu merupakan suatu program komputer yang sangat ampuh dan cocok sekali untuk melakukan berbagai bentuk manipulasi matematik. Maple mampu untuk melakukan berbagai perhitungan numerik, perhitungan secara simbolik (manipulasi aljabar), grafik, dan pemograman (programming).

Pada makalah ini penulis tidak mungkin rasanya untuk menjelaskan semua kemampuan dari program Maple tersebut secara lebih terperinci, karena keterbatasan waktu atau keterbatasan kemampuan dari penulis sendiri. Karena program Maple merupakan suatu program komputer yang cukup besar yang dapat menjangkau lebih banyak permasalahan matematika. Penulis hanya akan menjelaskan beberapa contoh sederhana berkenaan dengan berbagai operasi atau pengolahan data pada Maple. Penulis merasa penjelasan pada makalah ini sudah cukup untuk membuka wawasan dan memperkenalkan program Maple sebagai program komputer yang mempunyai kemampuan yang komplit, khususnya di bidang matematika.

Pada makalah ini dianggap bahwa program Maple diinstall di windows, minimal versi 3.1, dan versi dari Maple yang digunakan adalah adalah versi 4. Diharapkan pemakai telah mampu untuk menjalankan program di lingkungan windows 3.1, minimal bisa membukanya, hingga masuk ke program Maple. Sebenarnya nama dari program ini adalah Maple V, tapi disini penulis hanya menggunakan kata Maple saja.

Waktu kita membuka program windows pilihlah ikon yang berbentuk

$$\int x$$

dengan jalan menggerakkan mouse hingga pemandu (cursor) berada pada ikon tersebut. Kemudian dengan cepat tekan tombol kiri dari mouse dua kali berturut-turut. Setelah beberapa saat kemudian akan muncul di layar tanda " $>$ " (*prompt*) per-tanda bahwa kita telah berada pada program Maple, dan berarti pula Maple telah siap untuk melaksanakan berbagai perintah yang diberikan.

Setiap kita setelah selesai menuliskan suatu ekspresi, kita akan selalu mengakhirinya dengan tanda ":" dan/atau ";". Tanda titik dua menyebabkan hasil perhitungan hanya disimpan dalam memori, sedangkan tanda titik koma menyebabkan hasil perhitungan ditampilkan di layar.

2. Jenis Data

Data matematika di Maple terdiri atas bilangan, ekspresi aljabar, persamaan dan lain-lain. Masing-masing data ini masih terbagi lagi hingga lebih terinci, misalnya data bilangan terbagi atas bilangan bulat (integer), rasional, dan bilangan real (floating point). Ekspresi aljabar dapat berupa "+, *, ^, and, or", dan lain-lain. Untuk mengetahui jenis data ini kita dapat menggunakan perintah *whattype*.

Contoh:

```
> whattype(21046875);
```

integer

```
> m := 2104.6875;
```

m := 2104.6875

```
> whattype(m);
```

float

```
> whattype(nm);
string

> whattype(m + n);
+

> whattype(a and b);
and
```

3. Ekspresi, Variabel dan Fungsi

Ekspresi data Maple dapat berupa angka (numerik), karakter, ataupun bentuk-bentuk persamaan matematika.

Contoh

```
>13^26+135;
91733330193268616658399616144
```

```
>'Berkenalan_dengan_Maple';
Berkenalan_dengan_Maple
```

```
>x^2-3*x+4;
x2 - 3x + 4
```

Ekspresi dapat disimpan dalam memori pada suatu variabel. Nama variabel dapat terdiri dari 524275 karakter. Huruf besar dan kecil dianggap sebagai karakter yang berbeda. Nama variabel tidak boleh menggunakan kata-kata cadangan (reserved words) tertentu yang mempunyai arti khusus bagi Maple, seperti *evalf*, *solve*, *plot*, dan lain-lain. Disamping itu Maple juga mempunyai variabel khusus seperti *Pi* untuk bilangan π , *Digits* untuk banyaknya angka keberartian, *e* untuk bilangan pokok logaritma natural, dan lain-lain. Berikut ini adalah contoh untuk pengisian variabel.

```
>a:=68; # contoh 1
```

```
b:=75;
```

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

```
a := 68
```

```
b := 75
```

```
c := 143
```

```
>c:=a+b: # contoh 2
```

```
a:=68;
```

```
b:=75;
```

```
c:=c;
```

```
a := 68
```

```
b := 75
```

```
c := 143
```

Pada contoh pertama Maple mencatat isi dari variabel **c** sebagai ekspresi numerik, yaitu sebagai jumlah dari bilangan **68** dan **75**, sehingga kalau nilai variabel **a** atau **b** diganti maka nilai **c** akan tetap **143**. Tetapi pada contoh kedua Maple mencatat isi dari variabel **c** sebagai ekspresi dari variabel, sehingga kalau nilai dari variabel **a** atau **b** dirubah maka nilai dari variabel **c** juga berubah.

Isi dari suatu variabel dapat dihapuskan sehingga variabel tersebut dapat dipergunakan sebagai variabel matematik. Misalkan kita ingin mengosongkan isi dari variabel **a**, maka perintah yang kita berikan adalah:

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

```
a;
```

```
a
```

Contoh

```
>p:=2*x^4-5*x^2+3;
```

```
x:=2;
p;
x:='x';
p;
```

$$p := 2x^4 - 5x^2 + 3$$

$$x := 2$$

$$15$$

$$x := x$$

$$2x^4 - 5x^2 + 3$$

Cara lain untuk mengganti isi dari suatu variabel adalah dengan menggunakan perintah *subs*. Dari contoh sebelumnya berikan perintah berikut:

```
>subs(x=2,");
p;
x;
```

$$p := 15 .$$

$$2x^4 - 5x^2 + 3$$

$$x$$

Fungsi Baku

Seperti yang telah disinggung sebelumnya, Maple juga telah siap dengan fungsi baku seperti *sin*, *cos*, *tan*, *ln*, dan lain-lain. Misalkan kita akan menghitung nilai $\sin(\frac{\pi}{3})$. Pada mulanya Maple akan memberikan nilai eksak, disamping itu kita dapat juga menampilkan nilai penghampirannya sampai angka keberartian tertentu. Perhatikan proses kerja berikut:

```
>x:=Pi/3;
sin(x);
evalf("");
```

$$x := \frac{1}{3}\pi$$

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

$$.8660254040$$

Mendefinisikan Fungsi

Dengan program Maple kita dapat mendefinisikan fungsi secara mudah. Misalkan kita akan mendefinisikan $f(x) = x^2 \sin(x)$, dan menghitung nilai fungsi di $x = \frac{\pi}{3}$. Bentuk perintah yang dapat kita berikan adalah:

```
>f:=x->x^2*sin(x);
f(Pi/3);
```

$$f := x \rightarrow x^2 \sin(x)$$

$$\frac{1}{18}\pi^2 \sqrt{3}$$

Dengan perintah ini variabel x merupakan variabel dummy, artinya fungsi yang didefinisikan secara

```
>f:=y->y^2*sin(y)
```

akan memberikan hasil yang sama. Perhatikan contoh berikut:

```
>f(x)+f(y)+f(z);
```

$$x^2 \sin(x) + y^2 \sin(y) + z^2 \sin(z)$$

Cara lain untuk mendefinisikan fungsi adalah dengan menggunakan perintah *unapply* dan perintah *piecewise*. Perintah *unapply*, yaitu mendefinisikan fungsi yang hasilnya seperti cara yang di atas, sedangkan dengan perintah *piecewise* fungsi didefinisikan sepotong-sepotong.

Contoh

```
>f:=x->piecewise(x>4 and x<8,x^2);
```

```
f(5);
```

```
f(1);
```

$$f := x \rightarrow \text{piecewise}(4 < x \text{ and } x < 8, x^2)$$

$$25$$

$$0$$

```
>f:=unapply(x^2*sin(x),x);
```

```
f(Pi/3);
```

$$f := x \rightarrow x^2 \sin(x)$$

$$\frac{1}{18}\pi^2\sqrt{3}$$

4. Mencari Informasi Ekspresi Data

Kadang-kadang kita ingin menggunakan sebagian informasi yang ada pada suatu ekspresi untuk menghasilkan bentuk ekspresi yang lain. Jika informasi tersebut cukup pendek dan sederhana, kita dengan mudah dapat menuliskannya kembali. Tapi dalam hal yang panjang dan rumit, kita dapat menghindari penulisan kembali. Perintah yang digunakan untuk ini adalah *op* dan *nops*, yang masing-masing adalah operan dan banyaknya operan. Bentuk $+$, $*$, $^$, dan lain-lain merupakan operan dalam Maple. Dengan operan ini kita dapat menyusun suatu ekspresi dari tipe yang sederhana menjadi suatu ekspresi yang cukup rumit. Dengan perintah *op*(operant) kita dapat meminta data yang terhubung, sedangkan dengan perintah *nops*(number operants) kita dapat menghitung banyaknya kemungkinan operan atau banyaknya data yang terhubung pada suatu ekspresi. Misalkan kita mempunyai ekspresi $2x^5 + 1$

$x^3 - 3x^2 - 7$. Perhatikan proses berikut

```
>p:= 2*x^5+x^3-3*x^2-7;
```

$$p := 2x^5 + x^3 - 3x^2 - 7$$

```
> whattype(p);
```

+

Perintah *whattype* pada ekspresi di atas, yaitu berupa operan terakhir yang ada pada ekspresi p . Bentuk-bentuk operan dari ekspresi p adalah $2x^5$, x^3 , $-3x^2$, -7 . Perhatikan contoh berikut yang menghasilkan banyaknya operan (data), beberapa operan, dan jenis dari suatu operan pada ekspresi p .

```
> nops(p); # Banyaknya operan pada p
```

4

```
> op(1,p); # operan pertama pada p
```

$2x^5$

```
> op(3,p); # operan ketiga pada p
```

$-3x^2$

```
> whattype(op(3,p)); # tipe dari operan ketiga pada p
```

*

```
> op(2,op(3,p)); # operan kedua dari operan ketiga pada p
```

x^2

Berikut ini adalah contoh yang berkenaan dengan membangun suatu ekspresi dengan menggunakan sebagian informasi pada ekspresi yang sudah ada.

```
>p:= (2*x+7)*(x^2-3*x+8);
```

$$p := (2x + 7)(x^2 - 3x + 8)$$

```
>q:=op(2,p)*(2*x^3- 2*x-1);
```

$$q := (x^2 - 3x + 8)(2x^3 - 2x - 1)$$

5. Struktur Data

Selain data tunggal, Maple juga mampu menangani data yang berbentuk kumpulan data seperti himpunan (*set*), barisan (*sequent*), daftar (*list*), tabel (*table*), dan larik (*array*).

Himpunan (set)

Maple mampu melaksanakan tugas-tugas yang berkenaan dengan himpunan mulai dari menampilkan himpunan itu sendiri sampai kepada operasi- operasi pada himpunan.

Contoh

```
> A := {e, t, n, i, m, n, o};
```

$$A := \{e, t, n, i, m, n, o\}$$

```
>B:={seq(2*x+3,x=1..5)};
```

$$B := \{5, 7, 9, 11, 13\}$$

```
> P := {m, i, s, s, i, s, s, i, p, p, i} :
```

```
Q := {s, u, n, g, a, i} :
```

```
Q_gabung_P := Q union P;
```

```
P_iris_Q := P intersect Q;
```

$$Q_gabung_P := \{m, i, s, p, u, n, g, a\}$$

$$P_iris_Q := \{i, s\}$$

```
> whattype(A);
```

```
set
```

Barisan (sequent)

Pada himpunan urutan elemen-elemennya tidak dipentingkan, dan kalau ada elemen yang sama hanya ditulis satu kali. Tetapi pada barisan elemen-elemennya terurut, dan semua elemennya ditampilkan sesuai dengan urutan yang kita ketikkan.

Contoh

```
> m := 21, 1, 19, 68;
```

```
  n := 4, 4, m, 75;
```

```
      m := 21, 1, 19, 68
```

```
      n := 4, 4, 21, 1, 19, 68, 75
```

Pada contoh di atas kita menampilkan barisan dengan langsung mengetikkan semua elemen-elemennya. Disamping cara ini kita dapat pula mendefinisikan suatu barisan dengan menggunakan perintah *seq*. Bentuk perintah tersebut adalah:

```
> seq(ekspresi, jelajahvariabel)
```

Pada barisan masing-masing datanya mempunyai nomor urut, dan kita dapat memanggil data yang kita inginkan. Dengan perintah ini kita dapat menampilkan operan-operan dari suatu ekspresi yang tersusun seperti halnya barisan.

Contoh

```
> S:=seq(4*x-1,x=1..5);
```

```
      S := 3, 7, 11, 15, 19
```

```
> Suku[4] := S[4];
```

```
      Suku4 := 15
```

```
>Ekspresi:= 2*x^5+x^3-3*x^2-7;
  brs_op:=seq(op(i,Ekspresi),i=1..nops(Ekspresi));
```

$$Ekspresi := 2x^5 + x^3 - 3x^2 - 7$$

$$brs_op := 2x^5, x^3, -3x^2, -7$$

Daftar (list)

Seperti halnya barisan, jenis data ini juga terurut. Bedanya data yang berupa daftar dibatasi oleh tanda kurung siku. Tipe ini dipakai jika operasi yang ada pada himpunan atau barisan tidak ingin dilakukan. Tetapi kita juga dapat meminta data nomor tertentu, dan dapat pula merubah tipenya menjadi himpunan dan/atau barisan. Disamping itu kita juga dapat mengganti beberapa elemen dari kumpulan data yang bertipe daftar. Perhatikan contoh berikut:

```
>a:=[21,4,68,75];
  a_3:=a[3];
  brs_a:=seq(a[i],i=1..4);
  him_a:={brs_a};
```

$$a := [21, 4, 68, 75]$$

$$a_3 := 68$$

$$brs_a := 21, 4, 68, 75$$

$$him_a := \{68, 4, 21, 75\}$$

```
>b:=seq(1/i^2,i=1..10);
  daf_b:=[b];
  daf_c:=subsop(6=2104,daf_b);
```

$$b := 1, \frac{1}{4}, \frac{1}{9}, \frac{1}{16}, \frac{1}{25}, \frac{1}{36}, \frac{1}{49}, \frac{1}{64}, \frac{1}{81}, \frac{1}{100}$$



$$daf_b := \left[1, \frac{1}{4}, \frac{1}{9}, \frac{1}{16}, \frac{1}{25}, \frac{1}{36}, \frac{1}{49}, \frac{1}{64}, \frac{1}{81}, \frac{1}{100} \right]$$

$$daf_c := \left[1, \frac{1}{4}, \frac{1}{9}, \frac{1}{16}, \frac{1}{25}, 2104, \frac{1}{49}, \frac{1}{64}, \frac{1}{81}, \frac{1}{100} \right]$$

Tabel(table)

Tabel adalah koleksi data di Maple yang diindeks tanpa harus menggunakan bilangan asli. Misalkan kita ingin membuat tabel nilai dari beberapa orang siswa seperti pada contoh berikut:

```
>Nilai:=table([Budi=30,Ani=45,Anton=68, Ali=85]);
```

```
Nilai:=table({
```

```
Budi=30
```

```
Ani=45
```

```
Anton=68
```

```
Ali=85
```

```
})
```

```
>Nilai_Ali:=Nilai[Ali];
```

Nilai_Ali := 85

Bentuk perintah berikut masing-masing adalah menampilkan barisan nama atau indeks yang ada dan barisan nilainya atau isi dari tabel nilai.

```
>Nama_siswa:=indices(Nilai);
```

```
Nilai_siswa:=entries(Nilai);
```

Nama_siswa := [Budi], [Ani], [Anton], [Ali]

Nilai_siswa := [30], [45], [68], [85]

Larik(array)

Array adalah bentuk umum dari kumpulan data di Maple. Indeks dari array dapat lebih dari satu dimensi, dalam hal indeks array satu dimensi data ini mirip

dengan *list*. Sebagai contoh perhatikan kumpulan data pada contoh *list* sebelumnya, yaitu $\mathbf{a}:=\mathbf{[21,4,68,75]}$. Kita menganggap kumpulan data itu sebagai data yang terdiri dari satu baris dan empat kolom, atau array berdimensi satu dengan panjang empat. Bentuk perintah yang kita berikan adalah:

```
>m:=array(1..4);
```

```
m := array(1..4, [ ])
```

```
>m[1]:=21 :m[2]:=4 :m[3]:=68 :m[4]:=75;
```

```
m1 := 21
```

```
m2 := 4
```

```
m3 := 68
```

```
m4 := 75
```

```
>print(m);
```

```
[21, 4, 68, 75]
```

Dengan perintah *array* kita dapat menghasilkan bentuk tampilan matriks, dimana kita menganggapnya sebagai *array* berdimensi dua, sebab array berdimensi dua akan terlihat berupa baris dan kolom sebagaimana halnya matriks. Misalkan kita akan menampilkan matriks dengan ordo 2x3, cara pendefinisian arraynya adalah sebagai berikut:

```
>A:=array(1..2,1..3);
```

```
print(A);
```

```
A := array(1..2, 1..3, [ ])
```

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

```
>whattype(");
```

Kita dapat mengisi entri-entri dari matriks tersebut dengan elemen-elemen yang diinginkan, seperti contoh berikut:

```
>A[1,1]:=2: A[1,2]:=1: A[1,3]:=-3:
A[2,1]:=4: A[2,2]:=0: A[2,3]:=7:
print(A);
```

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

Bentuk perintah lain untuk menampilkan matriks ini adalah:

```
>A:=array(1..2,1..3,[[2,1,-3],[4,0,7]]):
print(A);
```

$$A = \begin{bmatrix} 2 & 1 & -3 \\ 4 & 0 & 7 \end{bmatrix}$$

```
>A:=matrix(2,3,[2,1,-3,4,0,7]);
```

$$A = \begin{bmatrix} 2 & 1 & -3 \\ 4 & 0 & 7 \end{bmatrix}$$

```
>A[2,3]:=x^2-5*x+1:
print(A);
```

$$\begin{bmatrix} 2 & 1 & -3 \\ 4 & 0 & x^2 - 5x + 1 \end{bmatrix}$$

Disamping mampu menampilkan matriks, Maple juga dapat melakukan berbagai operasi pada matriks seperti penjumlahan, perkalian, invers, determinan, dan lain-lain. Kita dapat pula memperoleh entri-entri untuk mengisi array yang bergantung kepada suatu ekspresi. Misalnya matriks 3x2 dengan masing-masing elemennya ditentukan oleh aturan, yaitu elemen baris ke- i dan kolom ke- j isinya adalah $2i - j^2$. Bentuk perintah yang kita berikan adalah:

```
>A:= array(1..3,1..2,[seq([seq(2*i-j^2, j=1..2)],i=1..3)]);
```


$$A = \begin{bmatrix} 1 & -2 \\ 3 & 0 \\ 5 & 2 \end{bmatrix}$$

Maple juga mempunyai perintah khusus untuk menampilkan matriks identitas, yaitu dengan perintah *identity*. Misalkan kita akan menampilkan matriks identitas dengan ordo 4x4. Maka bentuk perintah yang kita berikan adalah:

```
>a:=array(1..4,1..4,identity);
```

```
a:= array(identity, 1..4, 1..4, [ ])
```

```
>print(a);
```

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

Atau

```
>i:=matrix(4,4,a);
```

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

6. Konversi Jenis Data

Dengan Maple kita dapat merubah suatu jenis data ke data lain. Bentuk perintah yang kita berikan adalah:

```
> convert(ekspresi, jenis)
```

Contoh

```
>Nilai:=[54,48,72,86,54,72];
```

```
Jenis_data:=whattype(Nilai);
```

```
Nilai :=[54, 48, 72, 86, 54, 72]
```

```
Jenis_data := list
```

```
>Jumlah_nilai:=convert(Nilai,'+');
```

```
Jenis_data:=whattype(Jumlah_nilai);
```

```
rata_rata:=Jumlah_nilai/nops(Nilai);
```

```
Jumlah_nilai :=386
```

```
Jenis_data := integer
```

```
Rata_rata :=  $\frac{193}{3}$ 
```

```
>him_Nilai:=convert(Nilai,set);
```

```
him_Nilai :={48, 54, 72, 86}
```

7. Perintah Map dan Zip

Program Maple dapat menampilkan peta atau nilai dari suatu fungsi, baik fungsi dengan satu peubah ataupun lebih. Perintah yang digunakan untuk keperluan ini adalah perintah **map** dan **zip**. Dengan perintah **map** kita dapat mengevaluasi nilai fungsi pada sekumpulan data, sedangkan dengan perintah **zip** kita dapat mengoperasikan atau mengevaluasi nilai fungsi pada beberapa kumpulan data. Bentuk perintahnya adalah:

```
>map(nama_fungsi,data)
```

```
>zip(nama_fungsi,data1,data2)
```

1509/K/97 (2)

K/001.642
R.12
19

Yusmet /MN-Maple/ hal- 17

Contoh

```
>f:=x->x^2;
```

```
Data:=[21,4,68,75];
```

```
Nilai_fungsi:=map(f,Data);
```

$$f := x \rightarrow x^2$$

$$Data := [21, 4, 68, 75]$$

$$Nilai_fungsi := [441, 16, 4624, 5625]$$

```
>g:=(x,y)->x+y;
```

```
Data_1:=[a,b,c];
```

```
Data_2:=[p,q,r];
```

```
Data_jumlah:=zip(g,Data_1,Data_2);
```

$$g := (x, y) \rightarrow x + y$$

$$Data_1 = [a, b, c]$$

$$Data_2 := [p, q, r]$$

$$Data_jumlah := [a + p, b + q, c + r]$$

Dengan perintah-perintah di atas kita dapat mengolah dua kelompok data yang berpasangan, misalnya kita akan mencari nilai-nilai dari $\sum x$, $\sum y$, $\sum x^2$, $\sum y^2$, $\sum xy$ dari kelompok data berikut:

	1	2	3	4	5	6	7	8	9	10
X	112	125	94	98	120	115	110	116	95	90
Y	7	9	6	6	8	7	7	7	6	5

Kita misalkan variabel-variabel Jum_x , Jum_y , Jum_x2 , Jum_y2 , dan Jum_xy masing-masing menyatakan besaran-besaran di atas. Maka bentuk perintah yang dapat kita berikan adalah:



```

>f:=x->x^2:
g:=(x,y)->x*y:
x:=[112,125,94,98,120,115,110,116,95,90]:
y:=[7,9,6,6,8,7,7,7,6,5]:
x2:=map(f,x):
y2:=map(f,y):
xy:=zip(g,x,y):
Jum_x:=convert(x,'+');
Jum_y:=convert(y,'+');
Jum_x2:=convert(x2,'+');
Jum_y2:=convert(y2,'+');
Jum_xy:=convert(xy,'+');

```

Jum_x :=1075

Jum_y :=68

Jum_x2 :=116915

Jum_y2 :=474

Jum_xy :=7428

Jadi dari hasil di atas kita peroleh $\sum x=1075$, $\sum y=68$, $\sum x^2=116915$, $\sum y^2=474$,
 $\sum xy=7428$.

Penutup

Pengembangan perangkat lunak komputer sebagai alat bantu dalam proses belajar mengajar makin terasa manfaatnya untuk meningkatkan mutu pendidikan secara umum. Pada saat ini sudah cukup banyak beredar perangkat lunak komputer yang dapat dimanfaatkan untuk memperlancar proses belajar mengajar. Namun kemajuan ini sepertinya tidak begitu berarti karena tidak digunakan seoptimal mungkin, seolah-olah perangkat lunak ini dikembangkan hanya sebagai alat komersial saja atau untuk keperluan penulisan, seperti artikel-artikel, buku, dan hal-hal lain yang bersifat sama.

Program Maple adalah suatu perangkat lunak komputer yang dapat mengolah bilangan dan karakter baik secara numerik, aljabar, dan grafik, bahkan juga untuk pemrograman. Karena itu Maple dapat menjangkau lebih banyak permasalahan matematika. Dengan kemampuan tersebut program Maple dapat dimanfaatkan sebagai alat bantu untuk memperlancar dan menimbulkan daya tarik dalam kegiatan belajar mengajar.

Demikianlah yang dapat penulis jelaskan pada makalah ini untuk memperkenalkan salah satu perangkat lunak komputer yang mampu melakukan berbagai bentuk manipulasi matematika. Bagi pembaca yang berminat untuk menambah wawasannya mengenai program ini dapat dipelajari sendiri pada buku yang penulis cantumkan pada daftar kepustakaan.



Daftar Kepustakaan

1. Char, Bruce W,.... Maple V Language Reference Manual. Springer-Verlag.
New York, Berlin, Heidelberg,.... 1991.
2. Char, Bruce W,.... Maple V Library Reference Manual. Springer-Verlag.
New York, Berlin, Heidelberg,.... 1991.
3. Redfern, Darren. Maple Handbook. Springer-Verlag.
New York, Berlin, Heidelberg,.... 1991.