

MODUL PRAKTIKUM KOMPUTASI DAN OPTIMASI



Disusun Oleh:

Ir. Taufik Nur, ST, MT, IPM

Dimas Primadian N, ST

KATA PENGANTAR

Alhamdulillah, puji syukur penulis haturkan kehadiran Allah SWT, yang telah memberikan nikmat kesehatan dan kekuatan serta kesempatan sehingga penyusunan Modul Praktikum ini dapat terselesaikan dengan baik.

Bahasa pemrograman (pemrograman komputer) merupakan salah satu tools yang banyak dipakai didalam penyelesaian kalkulasi keteknikan. Pemrograman komputer mulai berkembang sejak diperkenalkannya metode penyelesaian secara numerik. Metode numerik banyak dipakai untuk menyelesaikan kasus-kasus yang kompleks namun kekurangan metode ini adalah membutuhkan waktu yang lama dalam proses perhitungannya. Oleh sebab itu metode numerik sering menyertakan pemrograman komputer sebagai solusi penyelesaiannya.

Penyusunan modul ini dimaksudkan sebagai salah satu acuan didalam mempelajari teknik-teknik pemrograman. Karena keterbatasan waktu dan luasnya cakupan dari materi yang ada, sehingga tidak semua hal-hal yang berhubungan dengan materi tersebut disajikan dalam modul ini. Didalam modul hanya dibahas beberapa aplikasi yang umum digunakan didalam suatu industri kimia. Saran dan kritikan dari para pembaca tentunya sangat penulis harapkan demi kesempurnaan modul ini dimasa yang akan datang.

Makassar, April 2016

Penulis

BAB I

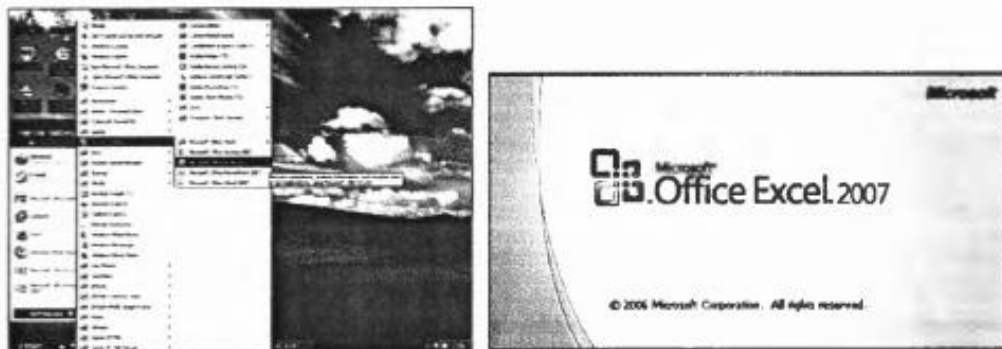
MICROSOFT EXCEL 2007

1.1 Mengoperasikan Excel 2007

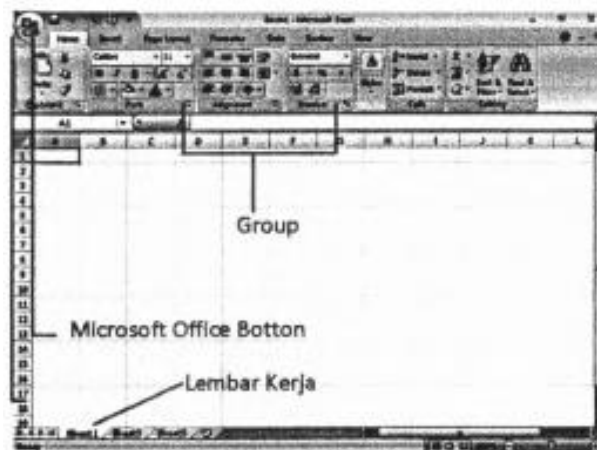
Agar Excel 2007 bisa bekerja secara optimal, sebaiknya system operasi yang digunakan di komputer Anda adalah Microsoft Windows XP Service Pack (SP) 2, Microsoft Windows Server 2003 atau Microsoft Windows Vista.

Untuk memulai mengoperasikan Excel 2007, langkah-langkahnya sebagai berikut,

1. Hidupkan komputer Anda, tunggu proses booting selesai sampai di Layar muncul desktop atau area kerja Windows.
2. Klik tombol Start yang terdapat pada toolbar.
3. Pilih dan klik menu Start >All Programs > Microsoft Office > Microsoft Office Excel 2007,



Selanjutnya jendela Microsoft Excel 2007 akan tampil seperti dibawah ini:



1.2 Data Masukan Pada Microsoft Excel

1. Data Teks

Data teks terdiri dari gabungan atau kombinasi antara huruf, angka dan tanda baca, pada intinya tidak dapat digunakan dalam operasi matematik. Ciri data teks adalah tulisanya rata kiri.

2. Data Angka

Data angka terdiri dari angka 0-9 dan beberapa karakter khusus seperti = + ., () % / dan lain-lain

3. Data Tanggal dan Waktu

Data tanggal dan waktu dalam Excel diketik menggunakan beberapa format, berikut bentuk dan contoh format tanggal dan waktu:

DATA TANGGAL		DATA WAKTU	
FORMAT	HASIL	FORMAT	HASIL
D/M	14/3	HH:MM	1:30
DD-MM	14-03	HH:MM:SS	1:30:42
DD-MM-YY	14-03-98	HH:MM:SS AM/PM	1:30:45 PM

4. Data Rumus/Formula Operator

Operator Aritmetik	Fungsi	Contoh
+ (tanda plus)	Penjumlahan	3+3
- (tanda minus)	Pengurangan	3-1
* (asterik)	Perkalian	3*3
/ (garis miring)	Pembagian	6/2
% (tanda persen)	Persen	20%
^ (caret)	Pangkat	3^2

Operator Pembandingan	Fungsi	Contoh
=	Sama dengan	A1=B1
>	Lebih besar dari	A1>B1
<	Lebih kecil dari	A1<B1
>=	Lebih besar atau sama dengan	A1>=B1
<>	Tidak sama dengan	A1<>B1

Operator Teks	Fungsi	Contoh
& (tanda dan)	Menggabungkan dua buah teks atau lebih untuk menghasilkan teks tunggal menjadi "Nortwind"	"North" & "Wind"

Operator Referensi	Fungsi	Contoh
: (Titik dua)	Sebagai batasan (range) dari beberapa sel yang menjadi acuan	B5:B15
(Spasi tunggal)	Perpotongan antara dua buah range yang bersilangan	Sum(B5:B15 A7:D7)
,/; (Koma atau titik koma)	Menghubungkan beberapa range menjadi suatu acuan	Sum(B5:B15;D5:D15)

5. Fungsi Rumus/Formula Statistik

- Average : Berfungsi untuk menghitung nilai rata-rata pada suatu range.
=Average(Range)
- Min : Berfungsi untuk mencari nilai minimum/terkecil pada suatu range.
=Min(Range)
- Max : Berfungsi untuk mencari nilai maksimum/terbesar pada suatu range.
=Max(Range)
- Sum : Berfungsi untuk menjumlahkan data pada suatu range.
=Sum(Range)
- Count : Berfungsi untuk menghitung data pada suatu range.
=Count(Range)
- Stdev : Menghitung deviasi baku data argumen =Stdev(Range)
- Var : Menghitung nilai Variance argumen =Var(Range)

6. Fungsi Rumus/Formula Matematik

- ABS : Berfungsi untuk mencari absolut/positive dari suatu bilangan.
- EVEN : Berfungsi untuk membulatkan bilangan kebilangan terdekat genap.
=Even(Bilangan)
- ODD : Berfungsi untuk membulatkan bilangan terdekat ganjil.
=Odd(Bilangan)

7. Fungsi Rumus/Formula Teks

- Left : Berfungsi untuk mengambil karakter teks paling kiri sebanyak n.
=Left(Teks,n)
- Right : berfungsi untuk mengambil karakter teks paling kanan sebanyak n.
=Right(Teks,n)

Mid : Berfungsi untuk mengambil karakter teks dari posisi tertentu sebanyak n. =Mid(Teks, Posisi, n)

Lower : Berfungsi untuk merubah teks menjadi huruf kecil. =Lower(teks)

Upper : Berfungsi untuk merubah teks menjadi huruf kapital. =Upper(Teks)

Proper : Berfungsi untuk merubah awal kata menjadi huruf kapital. =Proper(Teks)

8. Fungsi Logika (IF)

Fungsi ini bekerja berdasarkan perumusan logika yang memiliki dua kemungkinan jawaban, yaitu BENAR dan SALAH atau YA dan TIDAK atau TRUE dan FLASE.

Fungsi IF tunggal : =If(logika;hasil1;hasil2)

Fungsi IF Ganda : =If(Logika;hasil1;if(logika;hasil2;if()))

9. Fungsi Lookup

Choose : Untuk memilih data Value1, Value2 dan seterusnya yang ada dalam formulanya berdasarkan urutan angka Index_num.

=Choose(Index_num;Value1;Value2....)

Vlookup : Pencari data dimulai dari kolom pertama tabel yang dibaca, data bisa angka, sel referensi atau teks Table_array (table tabel yang akan dibaca dan harus dalam keadaan absolut, agar kalau dicopy tidak berubah) Col_Index_Num adalah posisi kolom data yang dibaca.

=Vlookup(Lookup_value;Table_array;Col_index_num)

Hlookup : Pencari data dimulai dari baris pertama tabel yang yang dibaca, data bisa angka, sel referensi atau teks Table_array (table tabel yang akan dibaca dan harus dalam keadaan absolut, agar kalau dicopy tidak berubah) Col_Index_Num adalah posisi kolom data yang dibaca.

=Hlookup(Lookup_value;Table_array;Col_index_num)

10. Sel

Dalam pengolahan data dalam Excel ada beberapa macam Sel yaitu:

- **Sel Relatif**

Adalah sel yang jika disalin atau dicopy akan menyesuaikan dengan sel yang baru

- **Sel Semi Absolut**

Adalah sel yang jika disalin salah satu nilainya tidak berubah. Sel Semi Absolut dilambangkan dengan tanda \$ (dollar)

Contoh:

\$A1 : Maka yang di Absolutkan adalah kolom A, maka nilai kolom A tidak akan berubah, sedangkan barisnya (1) menyesuaikan dengan baris yang baru.

A\$1 : Maka yang di Absolutkan adalah baris 1 , maka nilai baris 1 tidak akan berubah, sedangkan nilai kolom A menyesuaikan dengan kolom yang baru.

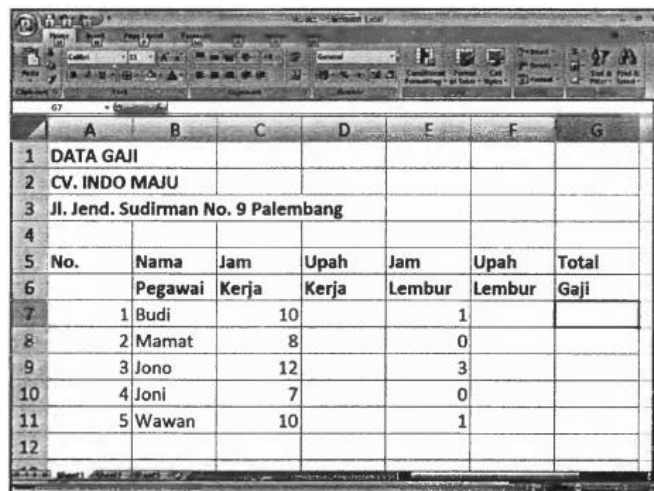
- **Sel Absolut**

Jika Sel Semi Absolut menetapkan salah satu baris atau kolom, maka Sel Absolut akan menetapkan nilai keduanya (baris dan kolom).

Contoh:

\$A\$1 : Maka nilai baris dan kolom tetap nilainya walaupun disalin ke sel yang lain

1.3 Latihan 1: Menggunakan fungsi Aritmetik



The screenshot shows an Excel spreadsheet with a table containing payroll data. The table has columns for employee number, name, working hours, overtime hours, and total salary. The data is as follows:

No.	Nama	Jam Kerja	Upah Kerja	Jam Lembur	Upah Lembur	Total Gaji
1	Budi	10		1		
2	Mamat	8		0		
3	Jono	12		3		
4	Joni	7		0		
5	Wawan	10		1		

Keterangan:

Upah/jam Rp. 2500

Upah lembur/jam Rp. 1000

Ketentuan Soal:

1. Upah kerja = Jam kerja x Upah/jam
2. Upah lembur = Jam lembur x Upah lembur/jam
3. Total Gaji = Upah kerja + Upah lembur

1.4 Mengcopy Rumus

- a. Mengcopy dengan menggunakan menu
 - Letakan Pointer pada sel D7 (yang memiliki rumus)
 - Klik menu Home lalu pilih Copy atau tekan tombol Ctrl-C pada keyboard
 - Sorot range D7:D11
 - Klik menu Home- Paste atau tekan Enter.
- b. Mengcopy dengan Fasilitas Autofill
 - Klik sel D7 (pada sel yang ada rumusnya)

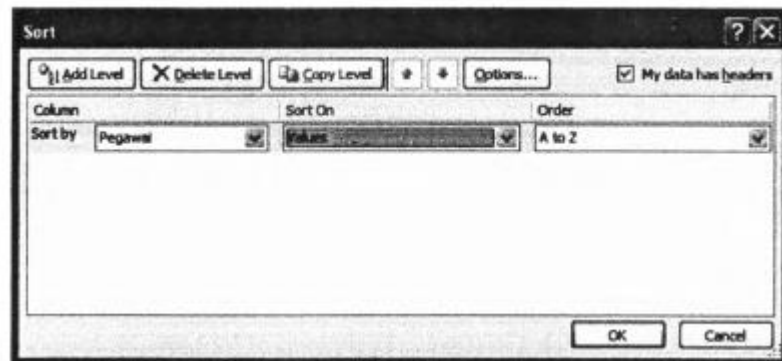
Jam Kerja	Upah Kerja	Jam Lembur
10	25000	1
8		0
12		3
7		0
10		1

- Arahkan Pinter pada Fill Handle, sampai pointer berubah menjadi tanda plus kecil (+), klik mouse dan tahan, geser mouse sampai D11 lalu lepaskan tombol mouse.

1.5 Membuat Border

1. Blok Range A5:G11, pada tab Home dalam group Font, seperti gambar dibawah ini:

2. Klik Sort, maka akan tampil gambar seperti dibawah ini:



3. Pada Sort By pilih Pegawai > Ok.

1.7 Latihan 2: Fungsi Teks

	A	B	C	D	E	F
1	DAFTAR PEGAWAI					
2						
3	NO	KODE	NAMA	JENIS KELAMIN	GOLONGAN	STATUS
4	1	P1N	HADI			
5	2	P3B	WIJAYA			
6	3	P2N	HENDRA			
7	4	W1N	DESI			
8	5	W2B	WATI			
9						

Ketentuan:

1. JENIS KELAMIN = di ambil dari 1 karakter pertama kode
2. GOLONGAN = di ambil dari karakter ke 2 dari kode
3. STATUS = diambil dari 1 karakter terakhir dari kode

1.8 Latihan 3: Menggunakan Fungsi Absolut dan Fungsi Statistik

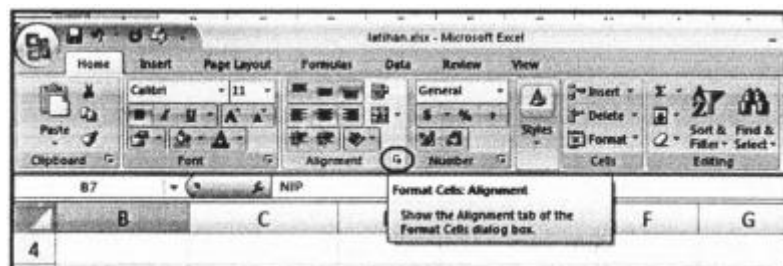
	A	B	C	D	E	F	G
1	DAFTAR PENJUALAN BUKU						
2					Diskon	25%	
3							
4	NO	JUDUL BUKU	HARGA BUKU	JUMLAH	DISKON	TOTAL	
5				BUKU			
6	1	Corel Draw X3	50000	2			
7	2	Adobe Photoshop CS3	35000	5			
8	3	Visual Basic	27500	7			
9	4	PHP	70000	8			
10	5	Microsof Office 2007	85000	15			
11	Total Seluruh (SUM)						
12	Data Tertinggi (MAX)						
13	Data Terendah (MIN)						
14	Rata-rata (AVERAGE)						
15	Jumlah Data (COUNT)						
16							

Ketentuan:

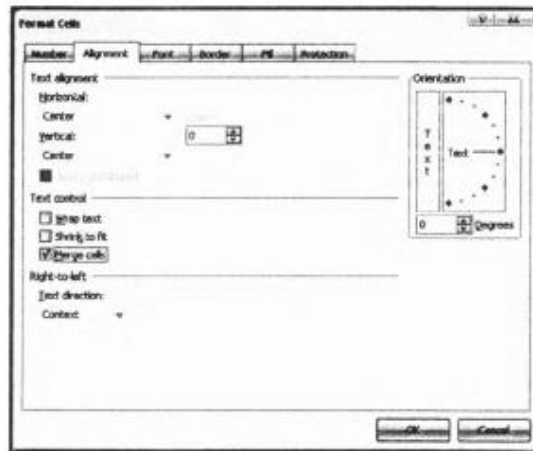
1. Diskon = Harga Buku x besarnya persentase Diskon
2. Total = (harga buku x jumlah buku) – Diskon

1.9 Mengatur Alignment

1. Blok Range A4:A5, Pada tab **Home** dalam group **Alignment**, seperti gambar dibawah ini:



2. Klik **Format Cells: Alignment**, maka akan tampil gambar seperti dibawah ini:



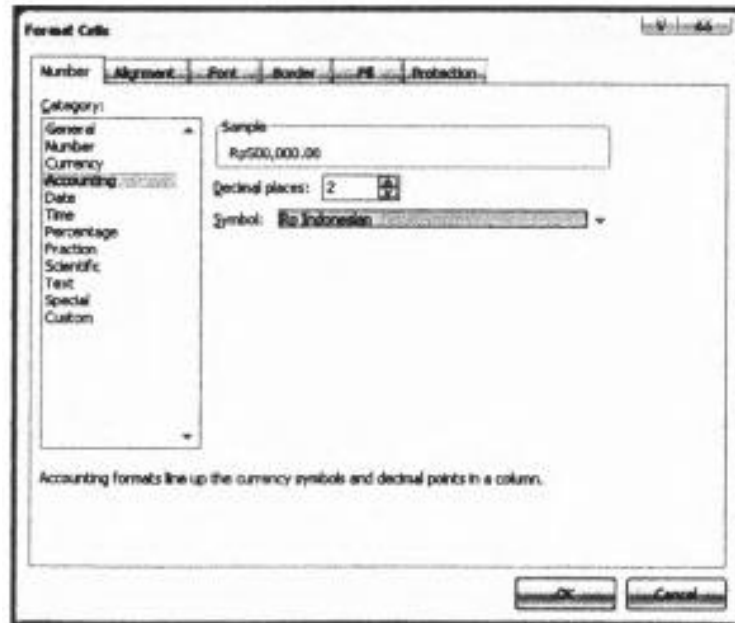
3. Pada kotak Item **Alignment**, pilih center pada Horizontal dan begitu juga pada Vertikal.
4. Pada Tex Control pilih **Merge Cells (menggabungkan sel)**.
5. Kemudian pada kotak **Orientation Degrees**, tentukan 45 Degrees. Bila perlu.
6. Klik **Ok**.

1.10 Format Mata Uang

1. Blok Range C6:C10, Pada tab **Home** dalam group **Number**, seperti gambar dibawah ini:



2. Klik menu **More Number Format >More >Number > Accounting > Symbol** : Pilih Rp Indonesia seperti gambar dibawah ini > **OK**



1.11 Latihan 3. Fungsi Logika (IF) dan Pembanding

	A	B	C	D	E	F	G
1	DAFTAR NILAI						
2							
3	NO	NAMA	KODE	GENDER	NILAI	HASIL	PREDIKAT
4	1	TONO	P		A		
5	2	DINI	W		C		
6	3	JIMMY	P		B		
7	4	DESI	W		D		
8	5	ZIDAN	P		A		
9							
10							

Ketentuan:

1. GENDER = jika kode "P" maka "Pria", Jika Kode "W" maka "Wanita"
2. HASIL = jika kode "A" maka 100, jika kode "B" maka 80 jika kode "C" maka 75, selain itu 60.
3. PREDIKAT = jika hasil ≥ 85 maka "Baik", Jika hasil ≥ 75 maka "cukup", jika hasil ≥ 60 maka "Biasa Saja".

1.12 Latihan 4. Menggunakan Fungsi Lookup:

	A	B	C	D	E	F	G	H	I
1	KARTU HASIL STUDI MASISWA								
2									
3	NAMA	HADIWIJAYA					Semester = 2		
4	NIM	10122234							
5	JURUSAN	Program Pendidikan Komputer 1 Tahun							
6									
7	MATAKULIAH	SKS	NILAI	BOBOT	TOTAL NILAI		NILAI	BOBOT	
8	PHP	2	A				A	4	
9	DREAMWAVER	3	B				B	3	
10	VISUAL BASIC	3	A				C	2	
11	NETWORKING	2	B				D	1	
12	COREL DRAW	2	A						
13	PHOTOSHOP	2	C						
14	JUMLAH	14							
15	INDEKS PRESTASI								
16									

Ketentuan:

1. Isi sel Bobot (D8), dengan mengambil data pada Vlookup
2. Isi sel Total Nilai (E8), merupakan perkalian antara SKS dengan Bobot
3. Isi sel Indek Prestasi (E15), merupakan hasil pembagian Jumlah Total Nilai dengan Jumlah SKS

Bentuk Tabel Lookup:

Nilai	Bobot
A	4
B	3
C	2
D	0

TABEL VLOOKUP

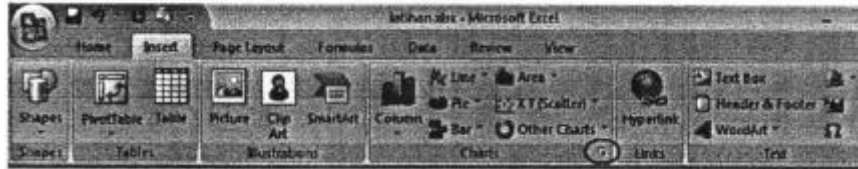
Nilai	A	B	C	D
Bobot	4	3	2	0

TABEL HLOOKUP

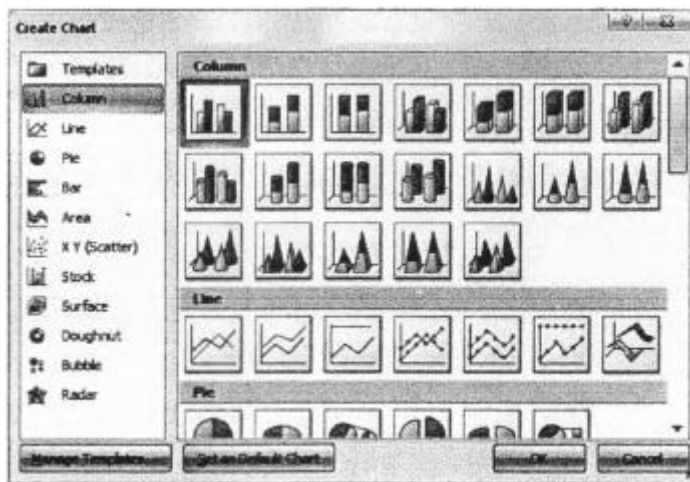
1.13 Latihan 5: Membuat Grafik

	A	B	C	D	E	F
1	PalComTech					
2	PROGRAM PAKET SHORT COURSE					
3	BULAN MARET 2008					
4						
5	PROGRAM	KOMP. OFFICE	TEKNIS KOMPUTER	KOMP. PROGRAMER	KOMP. AKUNTANSI	KOMP. DESAIN GRAFIS
6	JUMLAH MAHASISWA	200	110	99	40	60
7						

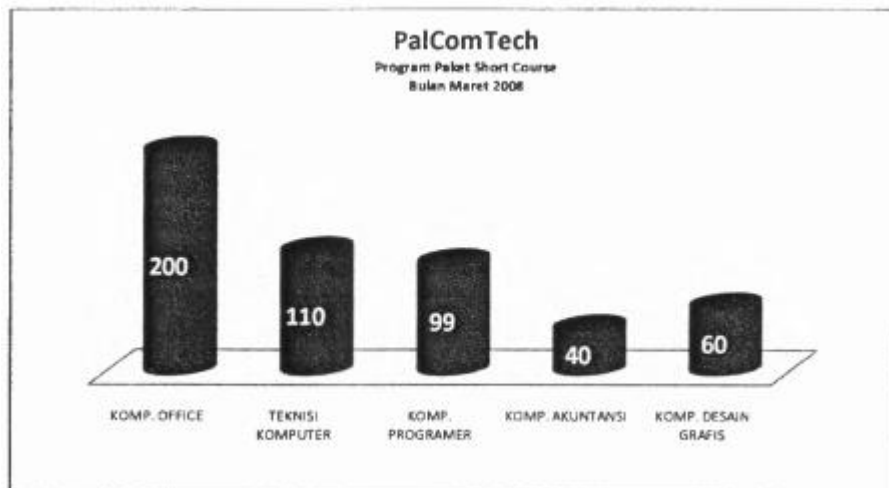
1. Blok Range A5:F6, Pada tab **Insert** dalam group **Charts** > klik tanda panah di sebelah kanan bawah, seperti gambar dibawah ini:



2. Klik tanda panah di sebelah kanan bawah, maka akan tampil kotak dialog seperti gambar dibawah ini:



3. Pilih salah satu jenis grapiknya > **Ok**, maka akan tampil gambar seperti dibawah ini:



1.14 Latihan Soal:

PT. Hadiwijaya
Jl. Jend. A Yani No. 14 Palembang

Tanggal tutup buku : **31-Dec-07**

NO	NIP	NAMA	J.K	STS	JMLH ANAK	G O L	RUANG	TGL MASUK	MASA BAKTI	GAPOK	TUNJANG				GATOR	PAJAK	GASIH
											NIKAH	ISTRI	ANAK	Ms. BAKTI			
1	LB04101	Hendra						17-Jun-87									
2	LB03302	Deddy						05-May-02									
3	LN24203	Hadi						10-Aug-00									
4	LN11310	Budi						15-Jun-95									
5	LB02211	Santoso						12-Apr-05									
6	LN72304	Kaka						11-Mar-94									
7	LB03113	Iwan						04-Aug-04									
8	PN13205	Sari						18-Feb-05									
9	PN12208	Lily						20-Jul-02									
10	LB04130	Badu						19-Jan-05									
11	PB02121	Fauziah						25-Sep-04									
12	PB01107	Tassya						02-Dec-93									
13	PN31309	Mery						30-Oct-92									

Ketentuan soal:

1. JK (Jenis Kelamin) diisi berdasarkan 1 karakter pertama dari NIP jika "L" maka "Laki-laki" dan jika "P" maka "Perempuan"
2. STS (Status) diisi berdasarkan karakter ke 2 dari NIP jika "B" maka "Belum Nikah", jika "N", maka "Nikah"
3. Jumlah Anak diisi berdasarkan karakter ke 3 dari NIP dan jadikan dalam bentuk Value
4. Isilah kolom Golongan dengan ketentuan jika karakter ke 5 dari NIP adalah 1, maka golongan I. Jika karakter ke 5 dari NIP adalah 2, maka golongan II, dan jika karakter ke 5 dari NIP adalah 3, maka golongan III
5. Kolom Ruang di isi jika karakter ke 4 NIP adalah 1, maka ruang A, jika karakter ke 4 NIP adalah 2, maka ruang B, jika karakter ke 4 NIP adalah 3, maka ruang C, dan jika karakter ke 4 NIP adalah 4, maka ruang D

6. Masa Bakti, diperhitungkan dari tanggal masuk pegawai sampai dengan tanggal tutup buku [format dalam tahun dengan dua angka desimal)
7. Kolom gapok merupakan jumlah gaji pokok dan honor (Gapok = Gapok + Honor) dan dimasukkan dengan tabel Lookup di bawah ini:

GOL	GAPOK
I	2.500.000
II	2.000.000
III	1.500.000

RUANG	A	B	C	D
HONOR	750.000	600.000	545.000	200.000

8. Kolom Tunjang, di isi dengan tunjangan yang terdiri dari:
 - a. Tunjangan Nikah sebesar 10% dari Gapok untuk semua pegawai yang telah menikah
 - b. Tunjangan Istri, diberikan kepada laki-laki yang sudah menikah sebesar Rp. 50.000,-
 - c. Tunjangan Anak, sebesar 5% dari Gapok untuk setiap anak dan terbatas sampai anak ke 2 saja
 - d. Tunjangan Masa Bakti, ditentukan dengan aturan :
 - 15% dari Gapok jika masa bakti lebih dari 5 tahun
 - 10% dari Gapok jika masa bakti antara 3-5 tahun
 - 5% dari Gapok jika masa bakti antara 1-3 tahun 1% dari Gapok jika masa bakti kurang dari 1 tahun
9. Gator, merupakan jumlah dari semua pendapatan yang diperoleh pegawai
10. Pajak di isi sesuai dengan ketentuan berikut:
 - a. 15% dari Gator untuk gator lebih dari 3.000.000,-
 - b. 10% dari Gator untuk gator antara 2.000.000,- sampai dengan 3.000.000, 5% dari Gator untuk gator antara 1.500.000,- sampai dengan 2.000.000 1% dari Gator untuk gator yang kurang dari 1.500.000,-
11. Isilah Gaji Bersih merupakan pengurangan Gaji kotor dengan pajak, serta formatlah dengan mata uang Rp. dengan dua angka decimal.
12. Simpanlah di Folder Anda

BAB II

LINGO

2.1. Pengenalan *Software* LINGO

LINGO adalah alat bantu yang didesain sangat luas untuk menyelesaikan permasalahan-permasalahan riset operasi seperti program linier dan non linier, kuadratik, *quadratically constrained*, stokastik dan optimasi model integer dengan lebih cepat, mudah dan efisien. LINGO menyediakan paket integrasi lengkap yang termasuk di dalamnya yaitu bahasa untuk optimasi model yang mudah dipahami.

Terdapat 5 menu di dalam *software* LINGO yaitu *File*, *Edit*, *LINGO*, *Window*, dan *Help*.



Dalam menu *File* terdapat beberapa perintah sebagai berikut:

File	Edit	Solver	Window	Help
New				F2
Open...				Ctrl+O
Save				Ctrl+S
Save As...				F5
Close				F6
Print...				F7
Print Setup...				F8
Print Preview				Shift+F8
Log Output...				F9
Take Commands...				F11
Export File...				▶
License...				
Database User Info...				
Recent File				
Exit				F10

Tabel 2.1 Fungsi dari masing-masing Submenu File

Submenu	Fungsi
---------	--------

<i>New (F2)</i>	Membuka jendela baru
<i>Open (Ctrl+O)</i>	Membuka <i>file</i> yang tersimpan
<i>Save (Ctrl+S)</i>	Menyimpan model yang sedang terbuka atau aktif
<i>Save as (F5)</i>	Menyimpan model yang sedang terbuka atau aktif dengan nama yang berbeda
<i>Close (F6)</i>	Menutup model yang sedang terbuka atau aktif
<i>Print (F7)</i>	Mencetak isi dari jendela yang sedang terbuka atau aktif
<i>Print Setup (F8)</i>	Mengkonfigurasi <i>printer preferences</i>
<i>Print Preview (Shift+F8)</i>	Menampilkan isi dari jendela <i>file</i> yang akan dicetak
<i>Log Output (F9)</i>	Membuka <i>log file</i> untuk <i>log output</i> pada command window
<i>Take Commands (F11)</i>	Menjalankan <i>command script</i> yang terdapat di suatu file
<i>Import Lindo File (F12)</i>	Mengkonversi Lindo <i>file</i> ke LINGO model
<i>Export File</i>	Ekspor model kedalam format MPS atau MPI
<i>License</i>	Rujukan bagi pengguna untuk <i>upgrade system</i>
<i>Database User Info</i>	Rujukan bagi pengguna untuk melihat id pengguna dan <i>password</i> untuk mengakses <i>database</i> melalui fungsi @ODBC()
<i>Exit (F10)</i>	Menutup LINGO

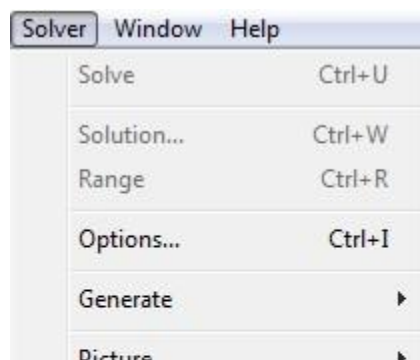
Menu *Edit*, digunakan untuk mengubah tipe *font*, teks perintah, dan lain-lain yang berkaitan dengan konten atau isi.



Tabel 2.2 Fungsi dari masing-masing Submenu Edit

Submenu	Fungsi
<i>Undo (Ctrl+Z)</i>	Meng- <i>undo</i> kegiatan terakhir
<i>Redo (Ctrl+Y)</i>	Me- <i>redo</i> kegiatan <i>undo</i> yang terakhir
<i>Cut (Ctrl+X)</i>	Menyalin dan menghapus kata/ kalimat yang diinginkan
<i>Copy (Ctrl+C)</i>	Menyalinkata/ kalimat yang diinginkan ke <i>clipboard</i>
<i>Paste (Ctrl+V)</i>	Menempel konten <i>clipboard</i> ke dalam dokumen
<i>Paste Special</i>	Menempel konten <i>clipboard</i> ke dalam dokumen dengan keadaan tertentu
<i>Select All (Ctrl+A)</i>	Memilih semua konten yang ada di dalam dokumen
<i>Find (Ctrl+F)</i>	Mencari kata di dalam suatu dokumen
<i>Find Next (Ctrl+N)</i>	Mencari kata di dalam suatu dokumen untuk kejadian selanjutnya
<i>Replace (Ctrl+H)</i>	Mengganti suatu kata dengan kata yang lain
<i>Go To Line (Ctrl+T)</i>	Memindahkan kursor ke suatu baris tertentu
<i>Match Parenthesis (Ctrl+P)</i>	Menemukan pasangan <i>parenthesis</i> yang terpilih
<i>Paste Function</i>	Menempel suatu fungsi spesifik yang terdapat di <i>software</i> LINGO
<i>Select Font (Ctrl+J)</i>	Mengkonfigurasi tipe huruf untuk sebagian teks yang dipilih
<i>Insert New Object</i>	Memasukkan objek dengan ekstensi OLE ke dalam dokumen
<i>Links</i>	Membuat link ke objek eksternal
<i>Object Properties</i>	Mendefinisikan sifat atau ketentuan lain dari gambar yang dipilih

Menu *Solver*, digunakan untuk memecahkan model, membuat laporan solusi, analisis dan formulasi, serta memunculkan grafik, mengidentifikasi *error*, dan sebagainya.



Tabel 2.3 Fungsi dari masing-masing Submenu Solver

Submenu	Fungsi
<i>Solve (Ctrl+U)</i>	Memecahkan model di jendela yang sedang dibuka/ aktif
<i>Solution Report (Ctrl+W)</i>	Membuka jendela <i>solution report</i> pada model yang sedang dibuka/aktif
<i>Range (Ctrl+R)</i>	Membuka <i>range analysis report</i> pada jendela yang sedang dibuka/aktif
<i>Options (Ctrl+I)</i>	Menetapkan pilihan sistem
<i>Generate</i>	Membuka representasi aljabar untuk model yang sedang dibuka/ aktif
<i>Picture</i>	Menampilkan gambaran grafis dari model matriks
<i>Debug (Ctrl+D)</i>	Melacak kesalahan formulasi di dalam kasus program linier yang <i>infeasible</i> dan <i>unbounded</i>

Menu *Window*, digunakan untuk mengatur tampilan jendela yang sedang terbuka atau aktif.



Tabel 2.4 Fungsi dari masing-masing Submenu Window

Submenu	Fungsi
Command Window (Ctrl+1)	Membuka jendela perintah untuk pengoperasian baris perintah di <i>software</i> LINGO
Status Window (Ctrl+2)	Membuka jendela <i>solver's status</i>
Close All (Ctrl+3)	Menutup semua jendela yang terbuka
Tile (Ctrl+4)	Menyusun semua jendela yang terbuka dengan pola <i>tile</i>
Cascade (Ctrl+5)	Menyusun semua jendela yang terbuka dengan pola <i>cascade</i>
Next	Menampilkan jendela berikutnya ke bagian depan dokumen
Previous (Ctrl+B)	Menampilkan jendela sebelumnya ke bagian depan dokumen

Menu *Help*, digunakan untuk membuka LINGO's *manual book* dan informasi-informasi mengenai *software* LINGO.



Tabel 2.5 Fungsi dari masing-masing Submenu Help

Help Topics	Fungsi
Register	Mendaftarkan <i>software</i> LINGO yang dimiliki, secara online
AutoUpdate	Memeriksa ketersediaan versi terbaru <i>software</i> LINGO
About Lingo	Menampilkan informasi tentang versi dan ukuran <i>software</i> LINGO yang dimiliki dan bagaimana cara menghubungi pihak perusahaan yaitu LINDO

Adapun beberapa manfaat atau keunggulan *software* LINGO adalah sebagai berikut:

1. Pengekspresian Model yang Mudah

LINGO dapat membuat formula untuk permasalahan linier, non linier dan integer secara cepat dengan bentuk yang sangat mudah untuk dibaca dan dipahami. Bahasa permodelan LINGO dapat membuat model yang sangat mirip dengan model matematik yang sering dibuat manual di atas kertas.

2. Pilihan Data Tidak Menyusahkan

Data yang akan diolah melalui software LINGO bisa merupakan data yang sebelumnya ditulis dalam sebuah database dan spreadsheets. Begitu pula dengan output solusi bisa dikeluarkan dalam bentuk database atau spreadsheet, sehingga pengguna bisa lebih mudah dalam pembuatan laporan sesuai dengan keinginan pengguna.

3. Solver yang Baik

Dengan menggunakan LINGO, pengguna tidak perlu menentukan atau memisahkan solver, karena LINGO akan membaca formulasi yang diberikan dan secara otomatis memilih solver yang tepat.

4. Model yang Interaktif

Pengguna dapat memanggil software LINGO langsung dari Excel macro atau aplikasi database lainnya. Untuk kasus building turn-key solutions, LINGO memiliki fungsi DLL dan OLE interfaces yang memungkinkan untuk dapat dipanggil dari aplikasi tertulis yang dimiliki pengguna.

5. Adanya Dokumentasi dan Bantuan

LINGO menyediakan semua alat bantu yang mungkin akan dibutuhkan untuk pembuatan dan running dari suatu model, sebagai contoh LINGO menyediakan teks diskusi dari kelas-kelas utama seperti optimasi program linier, non linier dan integer. LINGO juga menyediakan beberapa contoh model dasar untuk dimodifikasi dan dikembangkan.

2.2. Pembuatan Model LINGO

Sebuah optimasi terdiri dari tiga bagian utamayaitu:

1. Fungsi Tujuan

Sebuah formula yang mendeskripsikan apa yang harus dioptimalkan dalam suatu model. Sebagai contoh, fungsi tujuan dari suatu model adalah maksimasi keuntungan.

2. Variabel

Adalah kuantitas yang bisa diubah untuk mengeluarkan hasil yang optimal dari fungsi tujuan.

3. Batasan

Formula yang didefinisikan sebagai nilai pembatas dari suatu variabel.

Adapun hal-hal lain yang perlu diperhatikan dalam pembuatan model di LINGO adalah sebagai berikut:

1. Untuk comment dalam model diinisiasi dengan tanda seru (!) dan akan berwarna hijau.
2. LINGO menetapkan teks operator dan functions muncul dengan warna biru. Untuk tulisan lainnya akan dimunculkan dengan warna hitam.
3. Setiap statement di LINGO harus diakhiri dengan semi-colon (;)
4. Untuk nama variabel harus diawali dengan huruf (A-Z) dan karakter selanjutnya dapat berupa huruf, angka (0-9), atau underscore (_). Panjang dari nama variabel dapat mencapai hingga 32 karakter.

2.3. Penggunaan SETS pada LINGO

SETS adalah sekelompok objek yang berhubungan, digunakan untuk mendefinisikan suatu objek atau variabel. Adapun atribut yang dimiliki oleh *SETS* bisa lebih dari satu contoh:

Perusahaan/ P1 P2 P3/ : Kapasitas, Lokasi, Penitipan;

Adapun format penulisan *SETS* adalah sebagai berikut:

Nama_*SETS*/ Anggota/ : Atribut;

2.4. Fungsi Set Looping

Set Looping berfungsi untuk menerapkan semua operasi pada semua anggota SET dengan menggunakan satu statement. Adapun beberapa fungsi yang bisa diterapkan adalah sebagai berikut:

Tabel 2.6 Set Looping

Fungsi	Penggunaan
@FOR	Digunakan untuk membangkitkan pembatas ke seluruh anggota <i>SET</i> . Bisa juga digunakan dalam perhitungan untuk seluruh anggota <i>SET</i> .

@SUM	Untuk menghitung penjumlahan dari sebuah ekspresi untuk seluruh anggota <i>SET</i> .
@MIN	Untuk menghitung jumlah paling minimum dari sebuah ekspresi untuk seluruh anggota <i>SET</i> .
@MAX	Untuk menghitung jumlah paling minimum dari sebuah ekspresi untuk seluruh anggota <i>SET</i> .
@PROD	Untuk menghitung produk dari sebuah ekspresi untuk seluruh anggota <i>SET</i> .

2.5. LINGO Operators dan Functions

Terdapat tiga tipe operator yang digunakan di LINGO, yaitu aritmatika, logika, dan relational. Untuk operator aritmatika adalah sebagai berikut:

Tabel 2.7 Operator Aritmatika

Perpangkatan	\wedge
Perkalian	$*$
Pembagian	$/$
Penjumlahan	$+$
Pengurangan	$-$

Untuk *relational operator* yang sering digunakan dalam pendefinisian batasan pada model adalah sebagai berikut:

=	sama dengan
<=	ekspresi di sebelah kiri kurang dari atau sama dengan ekspresi di sebelah kanan
>=	ekspresi di sebelah kiri lebih dari atau sama dengan ekspresi di sebelah kanan

Untuk
logical

operator yang dapat dibaca oleh Lingo adaah sebagai berikut:


#EQ#	Sama dengan
#NE#	Tidak sama dengan
#GE#	Lebih besar dari atau sama dengan
#GT#	Lebih besar dari
#LE#	Kurang dari atau sama dengan
#LT#	Kurang dari

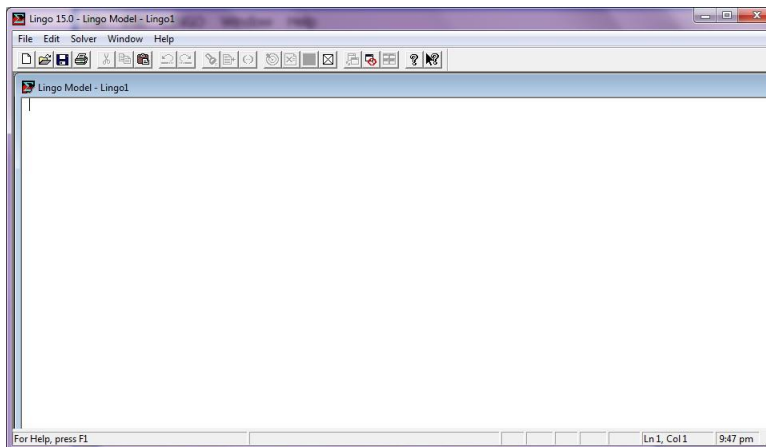
Studi Kasus (Transportation)

Seorang pengusaha peternakan lele, mempunyai tiga tempat khusus untuk memelihara lele. Tempat pemeliharaan lele tersebut meliputi daerah Yogyakarta, Magelang, dan Surakarta. Lele yang dihasilkan dari tiga daerah pemeliharaan tersebut akan didistribusikan ke tiga rumah makan besar yang ada di daerah Purwokerto, Semarang, dan Madiun. Kapasitas produksi lele adalah 4000 kg untuk daerah Yogyakarta, 5000 kg untuk daerah Magelang, dan

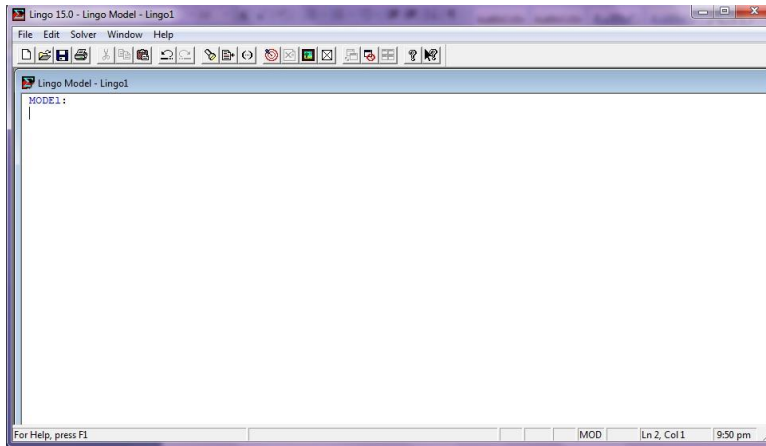
6000 untuk daerah Surakarta. Permintaan jumlah lele dari masing – masing rumah makan tersebut adalah 5000 kg untuk rumah makan di Purwokerto, 3000 kg untuk rumah makan di daerah Semarang, dan 5500 kg untuk rumah makan di Madiun. Jarak dari Yogyakarta ke rumah makan di Purwokerto, Semarang, dan Madiun adalah 40 km, 50 km, dan 70 km. Jarak dari Magelang ke rumah makan di Purwokerto, Semarang, dan Madiun adalah 60 km, 30 km, dan 80 km. Jarak dari Surakarta ke rumah makan di Purwokerto, Semarang, dan Madiun adalah 50 km, 20 km, dan 30 km. Biaya transportasi per kg per km adalah Rp.100. Tentukan kebijakan pengiriman lele dari tempat pemeliharaan tersebut ke rumah makan agar total biaya minimum.

Langkah-langkah penyelesaian:

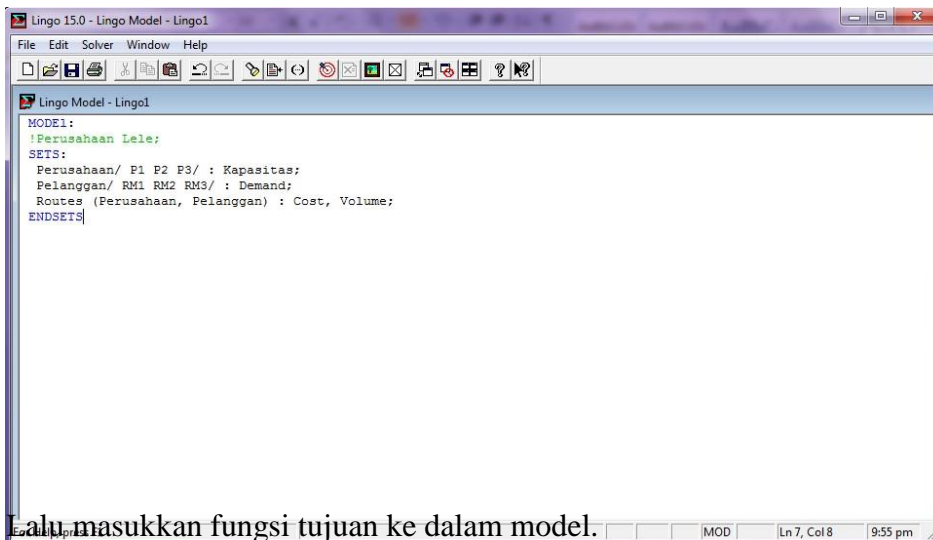
1. Buka *Software Lingo 15.0* dengan cara *double-click icon*  di *desktop*.
2. Setelah membuka maka akan muncul tampilan sebagai berikut:



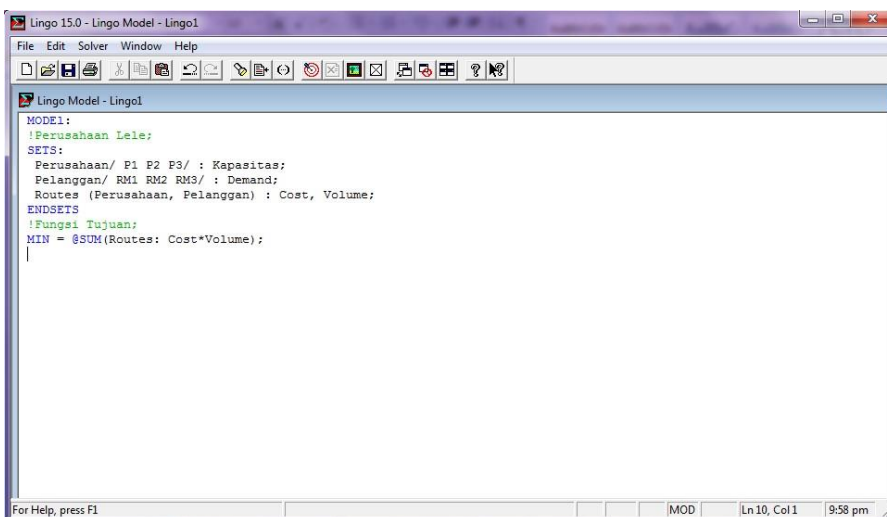
3. Lalu mulai membuat model.



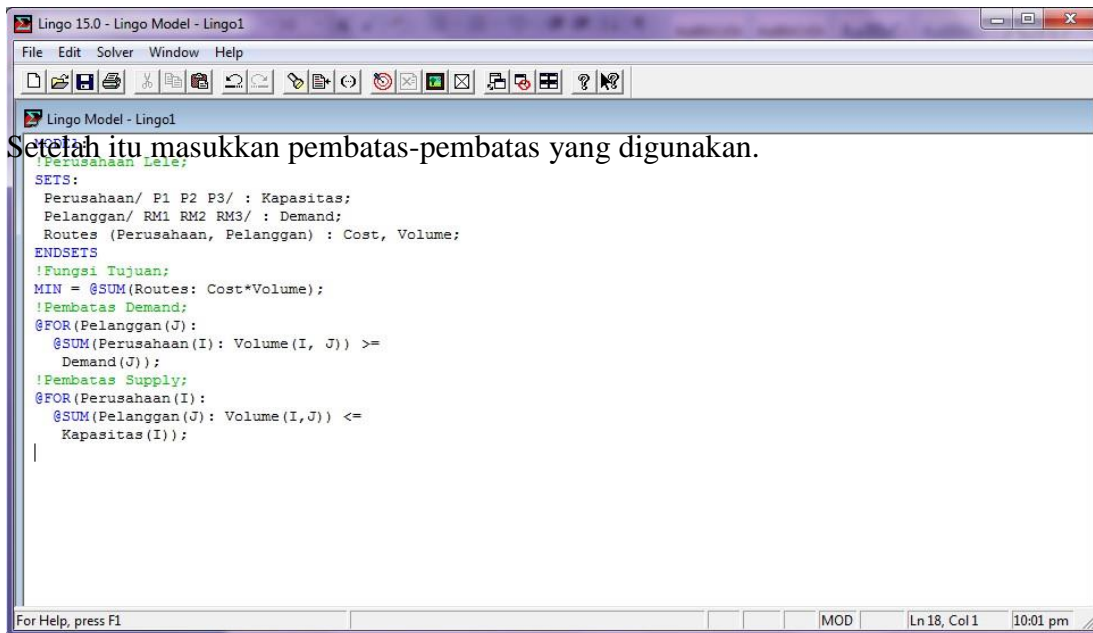
4. Pertama-tama buat *SETS* terlebih dahulu untuk mendefinisikan variabel yang akan kita cantumkan dalam model.



5. Lalu masukkan fungsi tujuan ke dalam model.

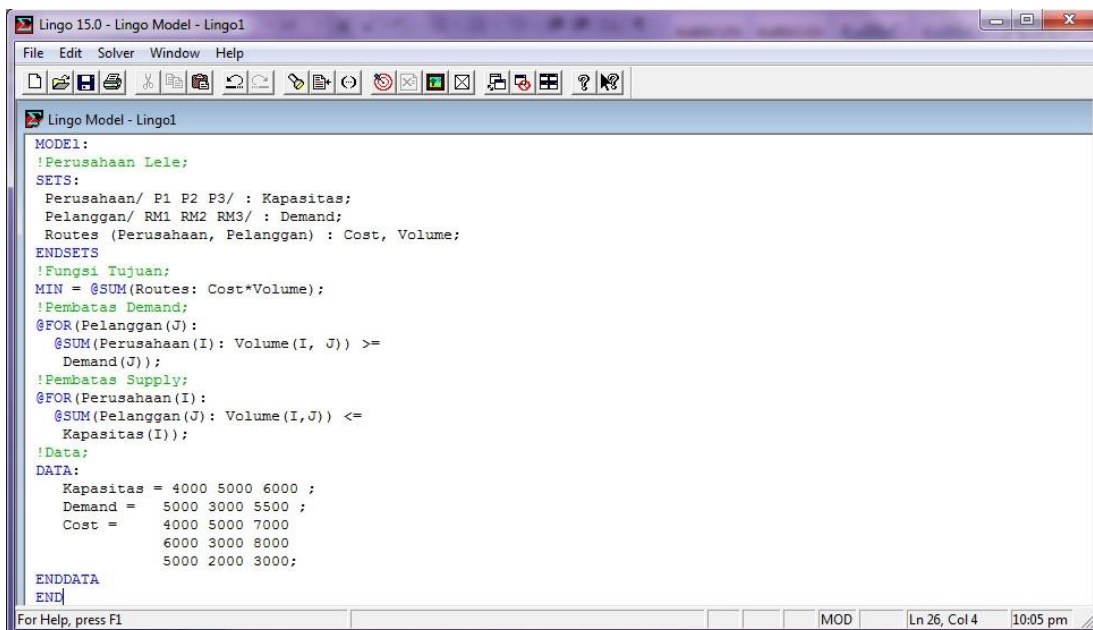


6. Setelah itu masukkan pembatas-pembatas yang digunakan.



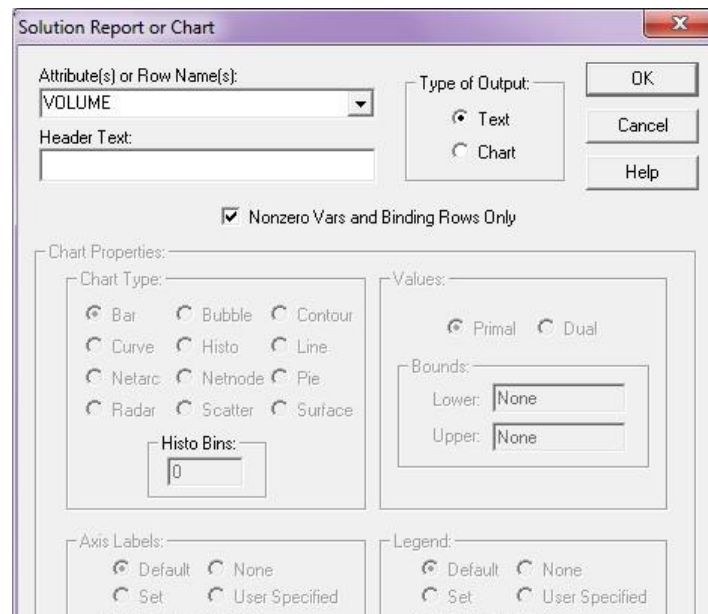
```
MODEL:
!Perusahaan Lele;
SETS:
Perusahaan/ P1 P2 P3/ : Kapasitas;
Pelanggan/ RM1 RM2 RM3/ : Demand;
Routes (Perusahaan, Pelanggan) : Cost, Volume;
ENDSETS
!Fungsi Tujuan;
MIN = @SUM(Routes: Cost*Volume);
!Pembatas Demand;
@FOR(Pelanggan(J):
@SUM(Perusahaan(I): Volume(I, J)) >=
Demand(J));
!Pembatas Supply;
@FOR(Perusahaan(I):
@SUM(Pelanggan(J): Volume(I, J)) <=
Kapasitas(I));
```

7. Lalu masukkan data yang digunakan, biasanya berupa biaya transportasi.



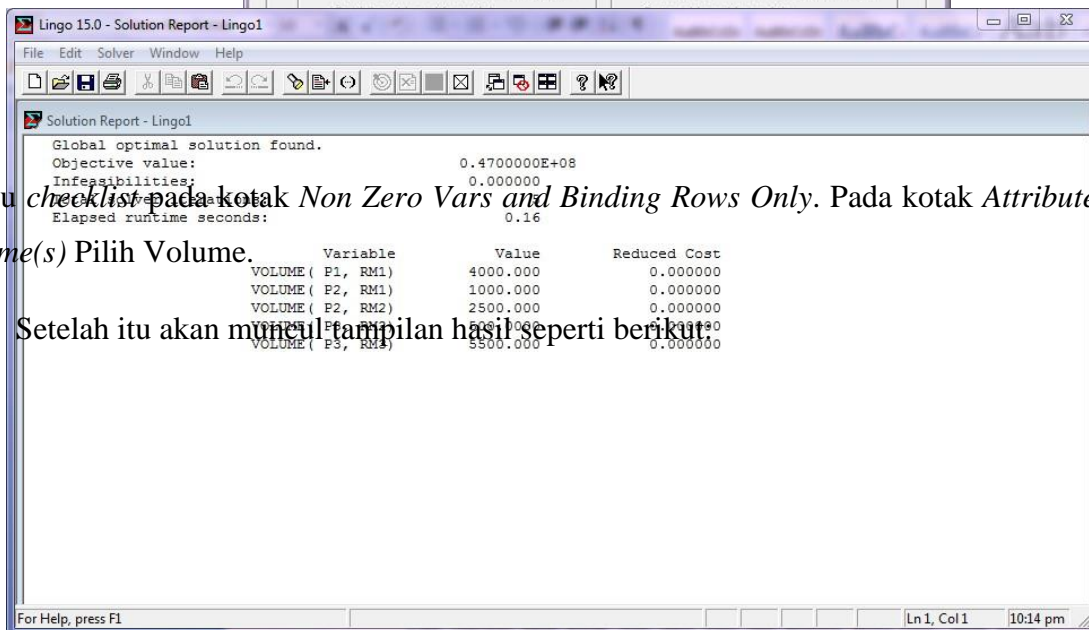
```
MODEL:
!Perusahaan Lele;
SETS:
Perusahaan/ P1 P2 P3/ : Kapasitas;
Pelanggan/ RM1 RM2 RM3/ : Demand;
Routes (Perusahaan, Pelanggan) : Cost, Volume;
ENDSETS
!Fungsi Tujuan;
MIN = @SUM(Routes: Cost*Volume);
!Pembatas Demand;
@FOR(Pelanggan(J):
@SUM(Perusahaan(I): Volume(I, J)) >=
Demand(J));
!Pembatas Supply;
@FOR(Perusahaan(I):
@SUM(Pelanggan(J): Volume(I, J)) <=
Kapasitas(I));
!Data;
DATA:
Kapasitas = 4000 5000 6000 ;
Demand = 5000 3000 5500 ;
Cost = 4000 5000 7000
6000 3000 8000
5000 2000 3000;
ENDDATA
END
```

- Setelah pembuatan model selesai , klik *Solution* pada Menu *Solver* dan akan muncul tampilan seperti berikut:



Lalu *checklist* pada kotak *Non Zero Vars and Binding Rows Only*. Pada kotak *Attribute(s) or Row Name(s)* Pilih *Volume*.

- Setelah itu akan muncul tampilan hasil seperti berikut:



10. Tahap terakhir adalah interpretasikan hasil.

Studi Kasus (Penugasan)

Dalam sebuah usaha peternakan lele terdapat empat orang pekerja yaitu Joko, Jono, Joni dan Jodi. Pemilik ingin menempatkan masing-masing pekerja dengan pekerjaan yang tersedia seperti pemeliharaan, pengepakan, distribusi dan penjualan. Adapun biaya yang harus dikeluarkan per jam untuk masing-masing pekerja dalam pekerjaan tertentu berturut-turut adalah:

Joko: Rp10.000, Rp9.000, Rp7.000, Rp8.000


Jono: Rp5.000, Rp8.000, Rp7.000, Rp7.000

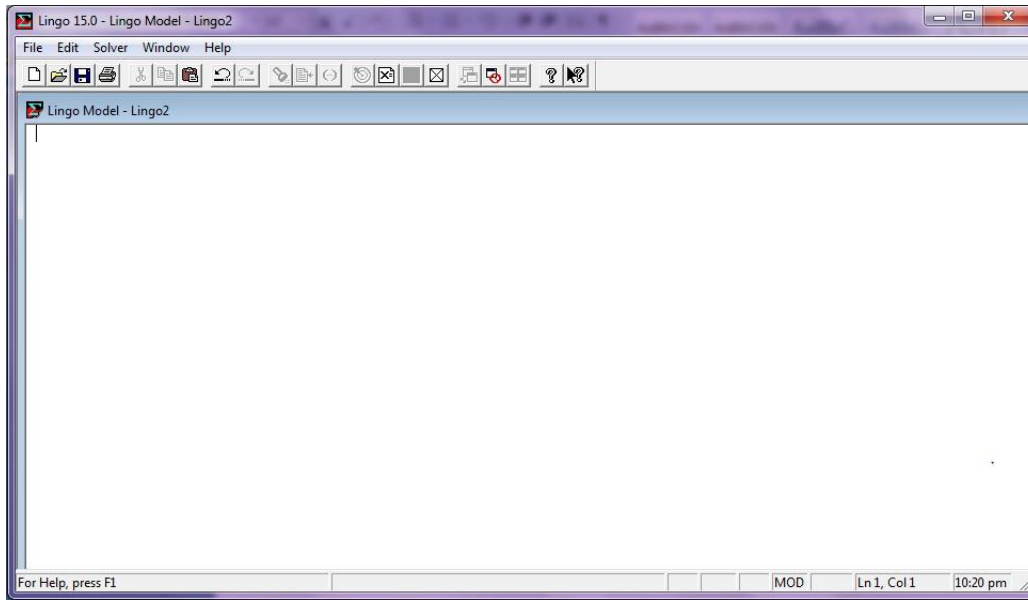
Joni: Rp5.000, Rp4.000, Rp6.000, Rp5.000

Jodi: Rp2.000, Rp3.000, Rp4.000, Rp5.000

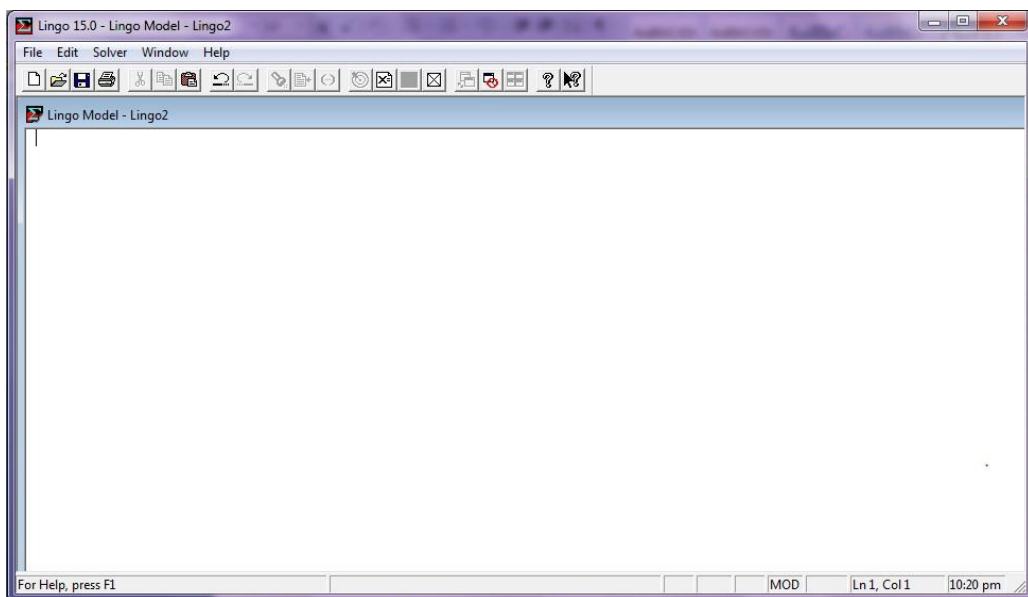
Bantulah pemilik usaha lele untuk menentukan pekerjaan yang tepat untuk masing-masing pekerja agar upah yang dikeluarkan minimum.

Langkah-langkah penyelesaian:

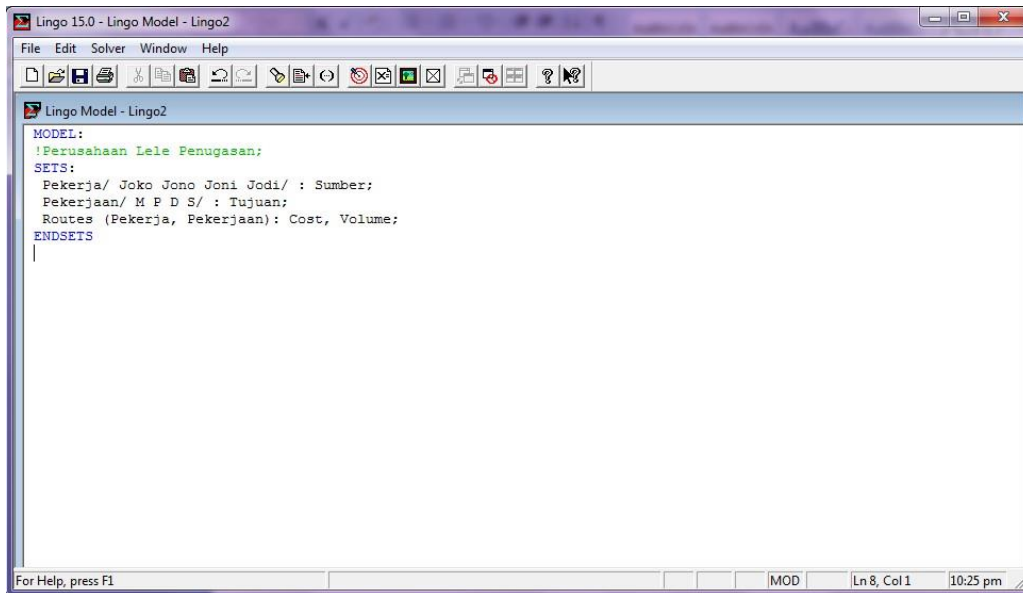
1. Buka jendela kerja baru di *Software Lingo 15.0* dengan cara klik *New* pada Menu *File* atau icon .
2. Setelah membuka maka akan muncul tampilan sebagai berikut:



3. Lalu mulai membuat model.



4. Pertama-tama buat *SETS* terlebih dahulu untuk mendefinisikan variabel yang akan kita cantumkan dalam model.

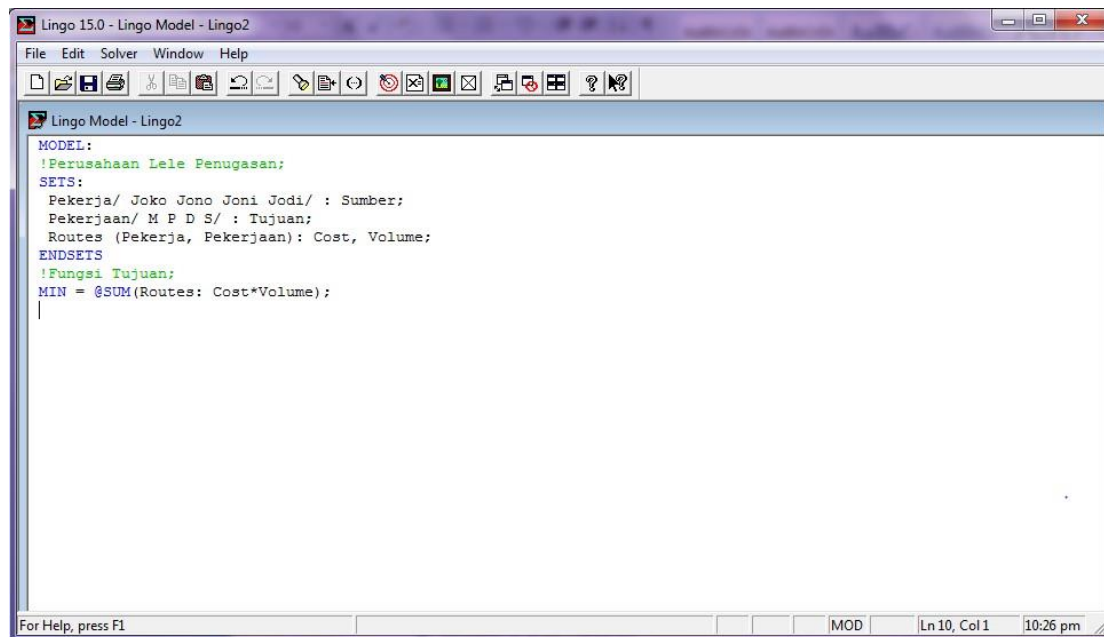


The screenshot shows the Lingo 15.0 - Lingo Model - Lingo2 window. The menu bar includes File, Edit, Solver, Window, and Help. The toolbar contains various icons for file operations and solving. The main text area contains the following code:

```
MODEL:
!Perusahaan Lele Penugasan;
SETS:
Pekerja/ Joko Jono Joni Jodi/ : Sumber;
Pekerjaan/ M P D S/ : Tujuan;
Routes (Pekerja, Pekerjaan): Cost, Volume;
ENDSETS
```

The status bar at the bottom indicates "For Help, press F1", "MOD", "Ln 8, Col 1", and "10:25 pm".

5. Lalu masukkan fungsi tujuan ke dalam model.

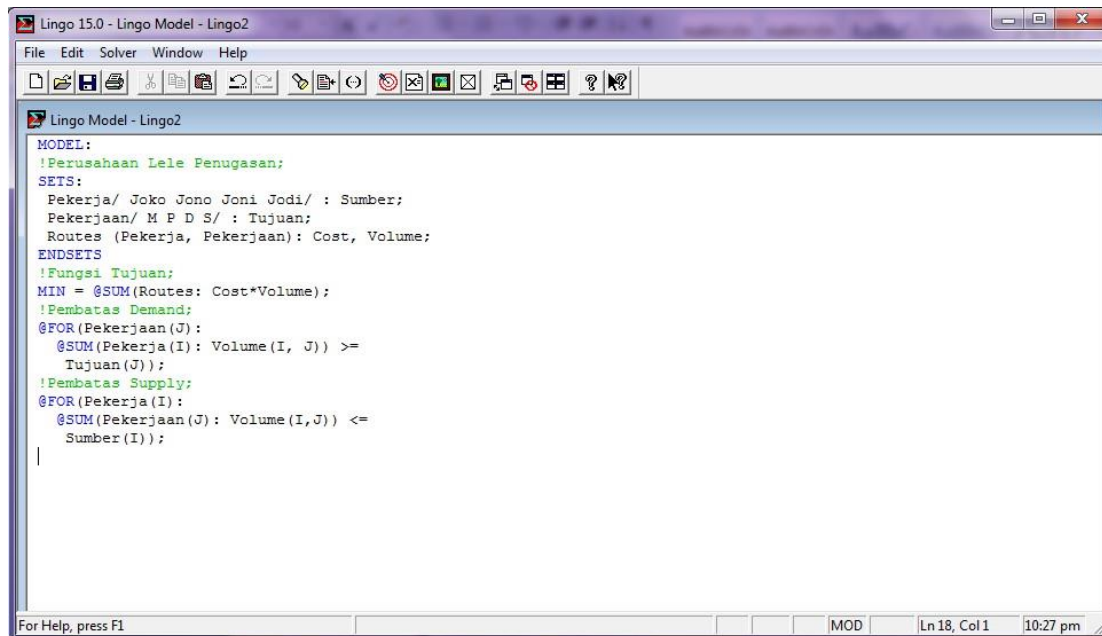


The screenshot shows the Lingo 15.0 - Lingo Model - Lingo2 window with the objective function added. The code in the main text area is now:

```
MODEL:
!Perusahaan Lele Penugasan;
SETS:
Pekerja/ Joko Jono Joni Jodi/ : Sumber;
Pekerjaan/ M P D S/ : Tujuan;
Routes (Pekerja, Pekerjaan): Cost, Volume;
ENDSETS
!Fungsi Tujuan;
MIN = @SUM(Routes: Cost*Volume);
```

The status bar at the bottom indicates "For Help, press F1", "MOD", "Ln 10, Col 1", and "10:26 pm".

6. Setelah itu masukkan pembatas-pembatas yang digunakan.

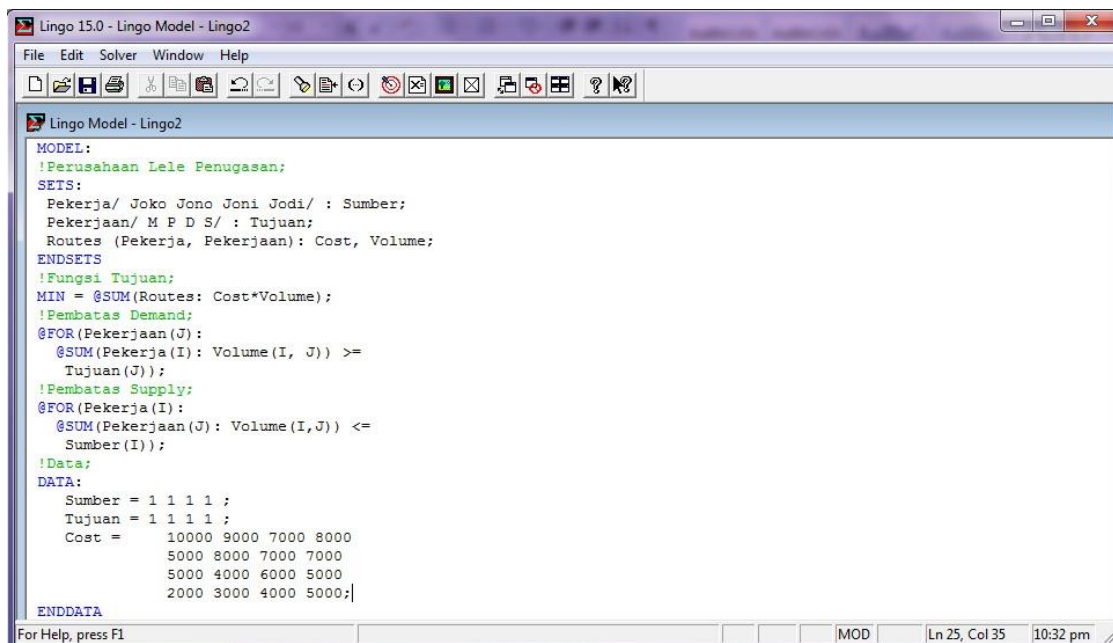


```
MODEL:
!Perusahaan Lele Penugasan;
SETS:
Pekerja/ Joko Jono Joni Jodi/ : Sumber;
Pekerjaan/ M P D S/ : Tujuan;
Routes (Pekerja, Pekerjaan): Cost, Volume;
ENDSETS
!Fungsi Tujuan;
MIN = @SUM(Routes: Cost*Volume);
!Pembatas Demand;
@FOR(Pekerja(J):
@SUM(Pekerja(I): Volume(I, J)) >=
Tujuan(J));
!Pembatas Supply;
@FOR(Pekerja(I):
@SUM(Pekerjaan(J): Volume(I, J)) <=
Sumber(I));
|
```

For Help, press F1

MOD Ln 18, Col 1 10:27 pm

7. Lalu masukkan data yang digunakan, biasanya berupa biaya penugasan.



```
MODEL:
!Perusahaan Lele Penugasan;
SETS:
Pekerja/ Joko Jono Joni Jodi/ : Sumber;
Pekerjaan/ M P D S/ : Tujuan;
Routes (Pekerja, Pekerjaan): Cost, Volume;
ENDSETS
!Fungsi Tujuan;
MIN = @SUM(Routes: Cost*Volume);
!Pembatas Demand;
@FOR(Pekerja(J):
@SUM(Pekerja(I): Volume(I, J)) >=
Tujuan(J));
!Pembatas Supply;
@FOR(Pekerja(I):
@SUM(Pekerjaan(J): Volume(I, J)) <=
Sumber(I));
!Data;
DATA:
Sumber = 1 1 1 1 ;
Tujuan = 1 1 1 1 ;
Cost =
10000 9000 7000 8000
5000 8000 7000 7000
5000 4000 6000 5000
2000 3000 4000 5000;
ENDDATA
```

For Help, press F1

MOD Ln 25, Col 35 10:32 pm

8. Setelah pembuatan model selesai , klik *Solution* pada Menu *Solver* dan akan muncul tampilan seperti berikut:

Attribute(s) or Row Name(s):
VOLUME

Header Text:

Type of Output:
☒ Text
☐ Chart

☒ Nonzero Vars and Binding Rows Only

Chart Properties:

Chart Type:
☒ Bar ☐ Bubble ☐ Contour
☐ Curve ☐ Histo ☐ Line
☐ Netarc ☐ Netnode ☐ Pie
☐ Radar ☐ Scatter ☐ Surface

Histo Bins:
0

Values:
☒ Primal ☐ Dual

Bounds:
Lower: None
Upper: None

Axis Labels:
☒ Default ☐ None
☐ Set ☐ User Specified

Set(s) or User Name(s):

Legend:
☒ Default ☐ None
☐ Set ☐ User Specified

Set or User Name(s):

☒ Use 3D and Shading

OK
Cancel
Help

9. Setelah itu akan muncul tampilan hasil seperti berikut:

Lingo 15.0 - Solution Report - Lingo2

File Edit Solver Window Help

Solution Report - Lingo2

Global optimal solution found.

Objective value:	20000.00
Infeasibilities:	0.000000
Total solver iterations:	7
Elapsed runtime seconds:	0.06

Variable	Value	Reduced Cost
VOLUME (JOKO, D)	1.000000	0.000000
VOLUME (JONO, M)	1.000000	0.000000
VOLUME (JONI, S)	1.000000	0.000000
VOLUME (JODI, F)	1.000000	0.000000

For Help, press F1

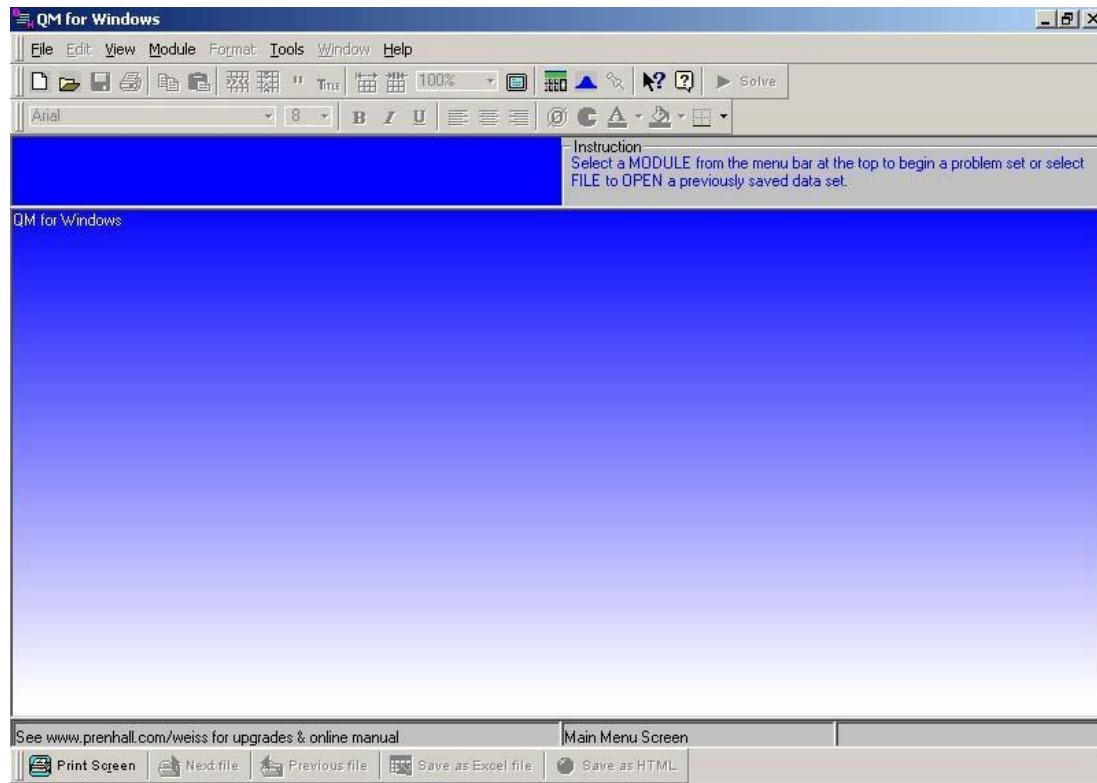
Ln 1, Col 1 10:35 pm

10. Tahap terakhir adalah interpretasikan hasil.

BAB III

POM-QM for Windows

QM adalah kepanjangan dari quantitative method yang merupakan perangkat lunak dan menyertai buku-buku teks seputar manajemen operasi. QM for windows merupakan gabungan dari program terdahulu DS dan POM for windows, jadi jika dibandingkan dengan program POM for windows modul-modul yang tersedia pada QM for windows lebih banyak. Namun ada modul-modul yang hanya tersedia pada program POM for windows, atau hanya tersedia di program DS for windows dan tidak tersedia di QM for windows. Berikut ini adalah contoh tampilan awal pada saat QM for windows dijalankan.



3.1. Linier Programming

Linear Programming (LP) adalah salah satu metode untuk menyelesaikan masalah optimasi. Masalah optimalisasi produksi menjadi salah satu masalah yang paling populer diselesaikan

dengan LP. Tujuan yang ingin dicapai biasanya memaksimumkan keuntungan dan meminimasi biaya produksi.

Studi Kasus

Perusahaan mebel “RAPI”, membuat meja dan kursi dari kayu. Setiap meja membutuhkan pekerjaan tukang kayu rata-rata selama 4 jam dan pengecatan rata-rata 2 jam; setiap kursi membutuhkan pekerjaan tukang kayu rata-rata 3 jam dan pengecatan rata-rata 1 jam. Dalam satu minggu tersedia 240 jam kerja untuk tukang kayu dan 100 jam kerja untuk pengecatan. Jika dijual, setiap meja menghasilkan keuntungan rata-rata \$7 dan setiap kursi \$5. Ringkasan data mengenai meja dan kursi ada pada Tabel dibawah ini.

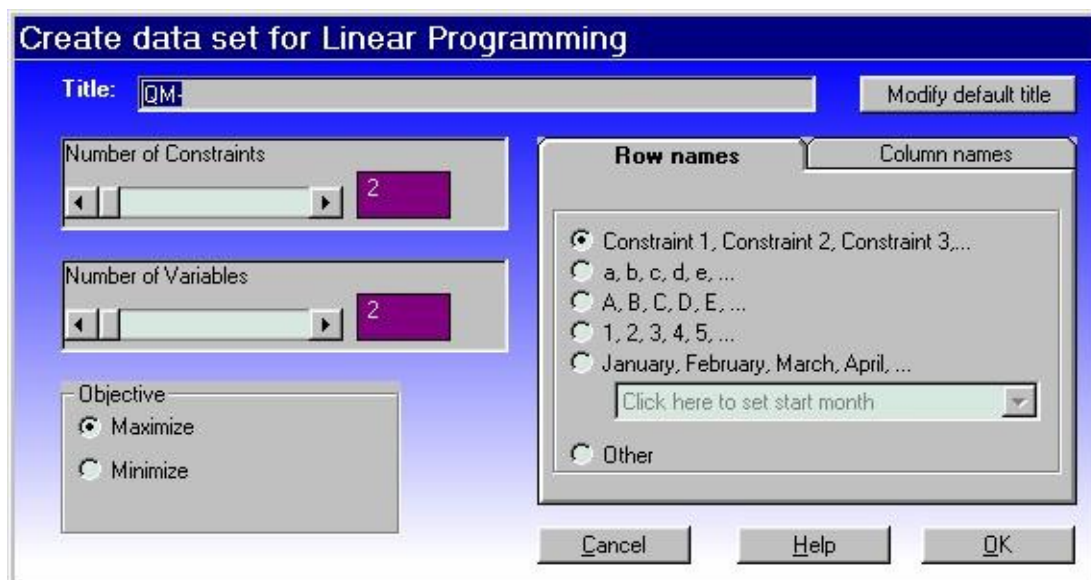
Pekerjaan	Jam yang dibutuhkan		Jam kerja tersedia perminggu kerja
	Meja	Kursi	
Tukang kayu	4	3	240
Pegecatan	2	1	100
Profit per unit	\$7	\$5	



Pertanyaan:

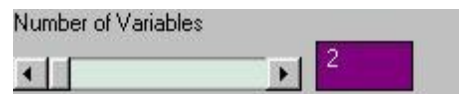
Berapa seharusnya produksi meja dan kursi dalam satu minggu kerja agar profit total perusahaan “RAPI” maksimal?

Langkah Penyelesaian

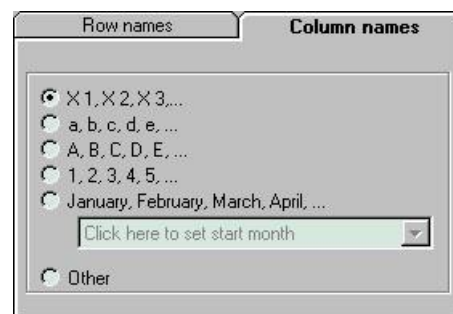
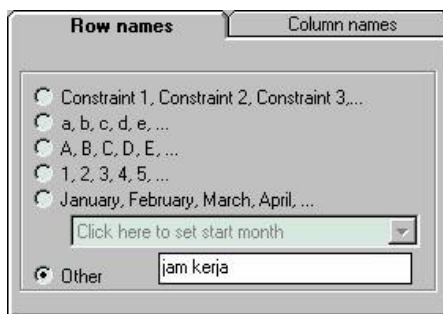
1. Jalankan program QM for Windows, pilih Module – Linear Programming.
2. Pilih menu File - New, sehingga muncul tampilan seperti Gambar dibawah ini



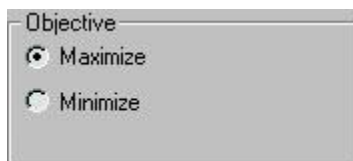
3. Buat judul penyelesaian soal ini dengan mengisi bagian Title: “CONTOH SOAL LP”. Jika Title tidak diisi, program QM for Windows akan membuat judul sendiri sesuai default (patokan)-nya. Default Title ini dapat dirubah dengan meng-klik modify default title. Judul dapat diubah/edit dengan meng-klik ikon title.
4. Isikan (set) jumlah kendala dengan 2, dengan cara meng-klik tanda  pada kotak Number of Constraints (dalam program QM for Windows, tidak perlu memasukkan kendala non negatif)
5. Isikan (set) jumlah variabel dengan 2, dengan cara meng-klik tanda  pada kotak Number of Variables



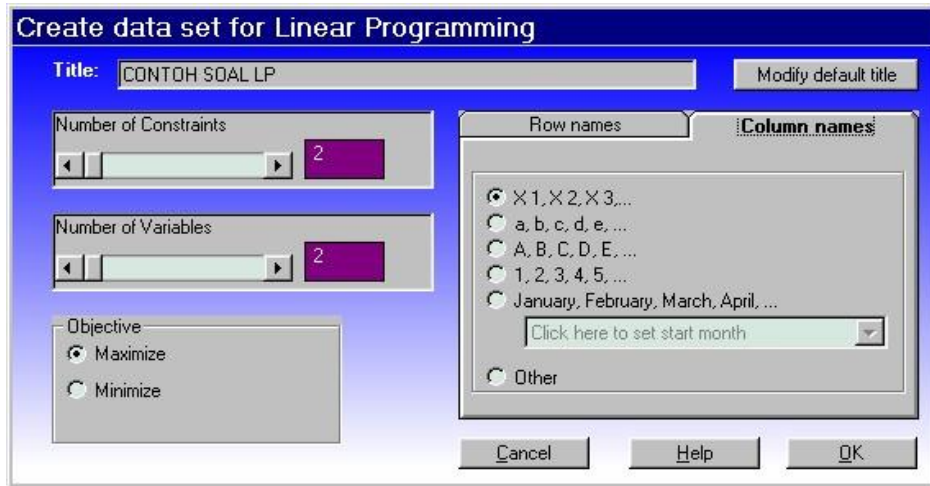
6. Pilih ☒ Other pada bagian Row names, kemudian isi dengan nama “jam kerja”
7. Pilih ☒ Other pada bagian Column names,



8. Biarkan pada bagian Objective, tetap pada pilihan Maximize




9. Sekarang tampilan akan seperti pada Gambar dibawah ini, lanjutkan dengan meng-klik tombol Ok.



	X1	X2		
Maximize	0	0		
jam kerja 1	0	0	<=	0
jam kerja 2	0	0	<=	0

10. Isikan angka-angka pada kotak-kotak yang bersesuaian antara jam kerja dan variabel (X_1 = meja; X_2 = kursi), yaitu

	X1	X2		
Maximize	7	5		
jam kerja 1	4	3	<=	240
jam kerja 2	2	1	<=	100

11. Selesaikan Contoh Soal ini dengan meng-klik tombol  pada toolbar atau dari menu File – Solve, atau dengan menekan tombol F9 pada keyboard
12. Jika ternyata ada data soal yang perlu diperbaiki, klik tombol edit pada toolbar atau dari menu File – Edit
13. Jangan lupa simpan (save) file kerja ini dengan menu File – Save (atau menekan tombol Ctrl+S). Pilihan untuk menyimpan file dengan format Excel (.xls) dan html (.html) juga disediakan.

3.2. Metode Transportasi

Studi Kasus

Ada tiga pabrik mebel A, B dan C masing masing memiliki kapasitas produksi maksimal dalam satu periode waktu tertentu 100, 300, dan 300 unit mebel. Ada tiga gudang D, E, dan F yang masing

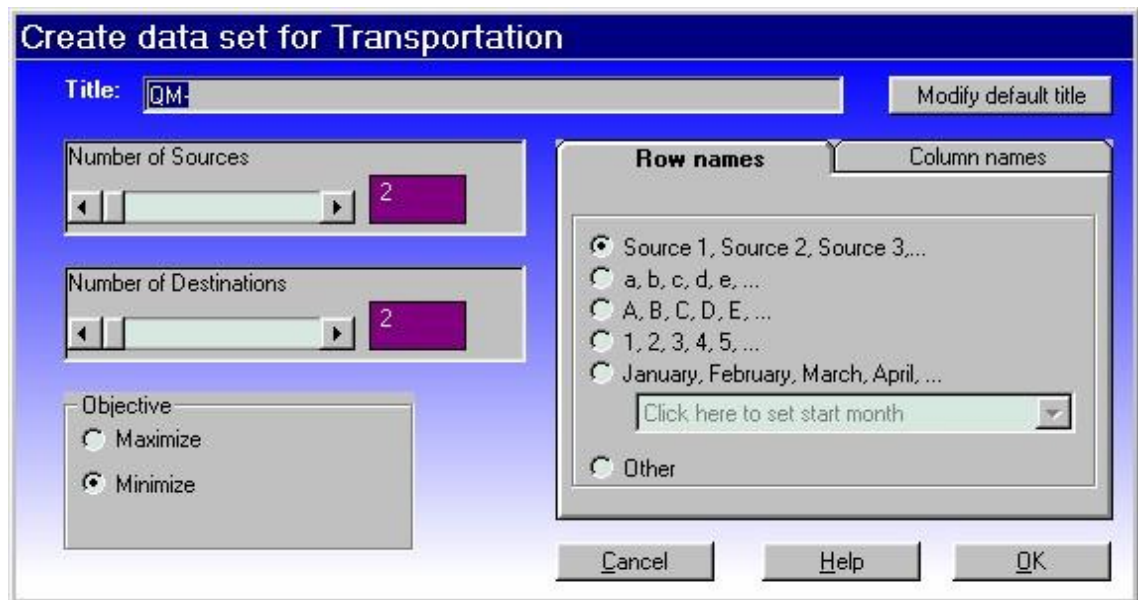
masing dapat menampung maksimal 300, 200 dan 200 unit mebel. Rata-rata biaya angkut per unit mebel dari masing-masing pabrik ke masing-masing gudang disajikan dalam Tabel dibawah ini



	Gudang D	Gudang E	Gudang F
Pabrik A	\$5	\$4	\$3
Pabrik B	\$8	\$4	\$3
Pabrik C	\$9	\$7	\$3

Pertanyaan: Berapa unit mebel harus diangkut dari masing-masing pabrik ke tiap-tiap gudang sehingga biaya transportasi total minimum?



Langkah Penyelesaian

- ✓ Jalankan program QM for Windows, pilih *Module – Transportation*
- ✓ Pilih *menu File - New*, sehingga muncul tampilan seperti Gambar dibawah ini.



- Buat judul penyelesaian soal ini dengan mengisi bagian *Title*: **“CONTOH SOAL TRANSPORTASI”** . Jika *Title* tidak diisi, program QM for Windows akan membuat judul sendiri sesuai default (patokan)- nya. Default *Title* ini dapat dirubah dengan meng-klik . Judul dapat diubah/edit dengan meng-klik ikon
- Isikan (set) jumlah sumber dengan 3, dengan cara meng-klik tanda  pada kotak *Number of Sources*
- Isikan (set) jumlah tujuan dengan 3, dengan cara meng-klik tanda  pada kotak *Number of Destinations*



- ☐ Pilih  pada bagian *Row names*, kemudian isi dengan nama **“Pabrik”**
- ☐ Pilih  pada bagian *Column names*, kemudian isi dengan nama **“Gudang”**

Row names **Column names**

☐ Source 1, Source 2, Source 3, ...
☐ a, b, c, d, e, ...
☐ A, B, C, D, E, ...
☐ 1, 2, 3, 4, 5, ...
☐ January, February, March, April, ...

☒ Other

Row names **Column names**

☐ Destination 1, Destination 2, Destination 3, ...
☐ a, b, c, d, e, ...
☐ A, B, C, D, E, ...
☐ 1, 2, 3, 4, 5, ...
☐ January, February, March, April, ...

☒ Other

- Biarkan pada bagian *Objective*, tetap pada pilihan *Minimize*

Objective

☐ Maximize
☒ Minimize

- Sekarang tampilan akan seperti pada Gambar, lanjutkan dengan meng-klik tombol **OK** hingga akan muncul tampilan seperti pada

Create data set for Transportation

Title:

Number of Sources

Number of Destinations

Objective

☐ Maximize
☒ Minimize

Row names **Column names**



☐ Destination 1, Destination 2, Destination 3, ...
☐ a, b, c, d, e, ...
☐ A, B, C, D, E, ...
☐ 1, 2, 3, 4, 5, ...
☐ January, February, March, April, ...

☒ Other

	Gudang 1	Gudang 2	Gudang 3	SUPPLY
Pabrik 1	0	0	0	0
Pabrik 2	0	0	0	0
Pabrik 3	0	0	0	0
DEMAND	0	0	0	

- Isikan angka-angka yang sesuai pada kotak-kotak yang bersesuaian antara Pabrik dan Gudang, yaitu

	Gudang 1	Gudang 2	Gudang 3	SUPPLY
Pabrik 1	5	4	3	100
Pabrik 2	8	4	3	300
Pabrik 3	9	7	5	300
DEMAND	300	200	200	

- Selesaikan Contoh Soal ini dengan meng-klik tombol  pada *toolbar* atau darimenu *File – Solve*, atau dengan menekan tombol F9 pada keyboard.
- Jika ternyata ada data soal yang perlu diperbaiki, klik tombol  pada *toolbar* atau dari menu *File – Edit*
- Jangan lupa simpan (save) file kerja ini dengan menu *File – Save* (atau menekan tombol Ctrl+S. Pilihan untuk menyimpan file dengan format Excel (.xls) dan html (.html) juga disediakan.

BAB IV

PRAKTEK MODEL PERSAMAAN STRUKTURAL (SEM) MELALUI PROGRAM AMOS

Banyak orang yang menghindari melakukan penelitian dengan menggunakan pendekatan Model Persamaan Struktural (SEM) dengan alasan kompleksitas prosedur analisis SEM. Analisis dengan menggunakan SEM memang sangat kompleks karena SEM merupakan analisis multivariat dengan banyak variabel. Namun dengan menggunakan AMOS, analisis SEM menjadi menarik dan menantang. AMOS menyediakan kanvas di dalam programnya agar peneliti menuangkan modelnya dalam bentuk gambar di dalam kanvas tersebut. Analisis menjadi semakin mudah karena dengan satu kali klik, gambar model yang dituangkan di dalam kanvas langsung dianalisis dengan lengkap. Makalah ini akan menyajikan prosedur analisis SEM melalui AMOS yang dilengkapi dengan beberapa informasi mengenai dasar-dasar SEM.

4.1. BAGIAN – BAGIAN SEM

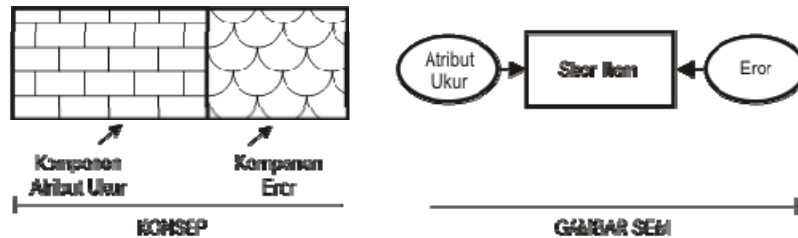
A. SUB MODEL SEM

SEM adalah penggabungan antara dua konsep statistika, yaitu konsep analisis faktor yang masuk pada model pengukuran (measurement model) dan konsep regresi melalui model struktural (structural model). Model pengukuran menjelaskan hubungan antara variabel dengan indikator-indikatornya dan model struktural menjelaskan hubungan antar variabel. Model pengukuran merupakan kajian dari psikometrika sedangkan model struktural merupakan kajian dari statistika.

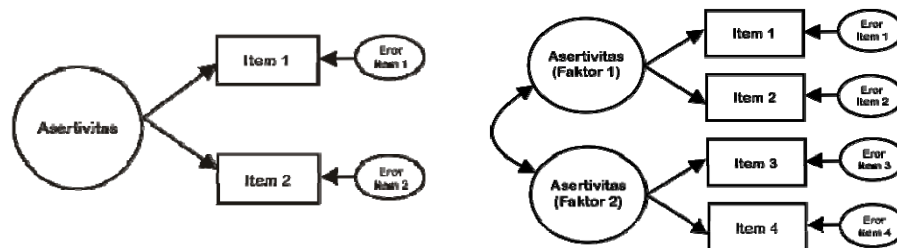
1. SUB MODEL PENGUKURAN

Di dalam sebuah skor hasil pengukuran (skor tampak), didalamnya terkandung dua komponen, yaitu a) komponen yang menjelaskan atribut yang diukur dan b) komponen yang terkait dengan atribut lain yang tidak diukur (error). Dengan kata lain, di dalam skor tampak didalamnya terkandung komponen yang menunjukkan atribut ukur dan error. Dalam gambar dengan pendekatan SEM konsep ini dijabarkan menjadi

gambar yang menunjukkan skor sebuah item yang dibangun dari dua komponen, yaitu atribut ukur dan eror (lihat Gambar dibawah ini).

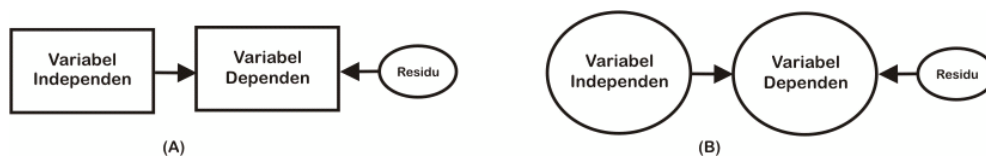


Model pengukuran menggambarkan hubungan antara item dengan konstruk yang diukur. Model pengukuran memiliki ketepatan model yang memuaskan ketika item-item yang dilibatkan mampu menjadi indikator dari konstruk yang diukur yang dibuktikan dengan nilai eror pengukuran yang rendah dan nilai komponen asertivitas yang tinggi.



2. SUB MODEL STRUKTURAL

Model struktural menggambarkan hubungan satu variabel dengan variabel lainnya. Hubungan tersebut dapat berupa korelasi maupun pengaruh. Korelasi antar variabel ditunjukkan dengan garis dengan berpanah di kedua ujungnya sedangkan pengaruh ditandai dengan satu ujung berpanah. Gambar diatas menunjukkan peranan variabel independen terhadap variabel dependen. Pada gambar tersebut terlihat ada dua jenis model struktural. Gambar dibawah ini menunjukkan hubungan antar dua konstruk terukur dan Gambar 4.b menunjukkan hubungan konstruk laten.

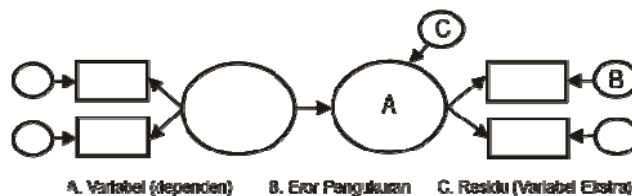


B. KONSTRUK

Konstrak adalah atribut yang menunjukkan variabel. Konstrak di dalam SEM terdiri dari dua jenis, yaitu konstrak empirik dan konstrak laten.



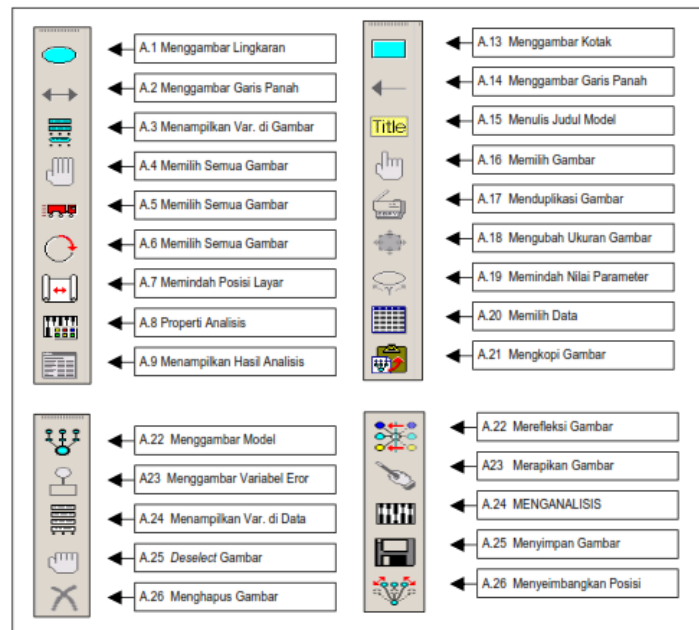
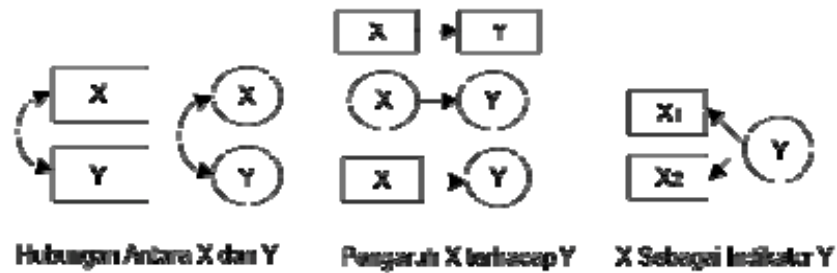
Konstrak Empirik. Merupakan konstrak yang terukur (observed). Dinamakan terukur karena kita dapat mengetahui besarnya konstrak ini secara empirik, misalnya dari item tunggal atau skor total item-item hasil pengukuran. Konstrak empirik disimbolkan dengan gambar kotak.



Konstrak Laten. Konstrak laten adalah konstrak yang tidak terukur (unobserved). Dinamakan tidak terukur karena tidak ada data empirik yang menunjukkan besarnya konstrak ini. Konstrak laten dapat berupa a) common factor yang menunjukkan domain yang diukur oleh seperangkat indikator/item dan b) unique factor (error) yang merupakan error pengukuran. Konstrak ini disimbolkan dengan gambar lingkaran dan c) residu yaitu faktor-faktor lain yang mempengaruhi variabel dependen selain variabel independen.

C. JALUR

Jalur (path) adalah informasi yang menunjukkan keterkaitan antara satu konstrak dengan konstrak lainnya. Jalur di dalam SEM terbagi menjadi dua jenis yaitu jalur hubungan kausal dan non kausal. Jalur kausal digambarkan dengan garis dengan panah salah satu ujungnya (J) dan jalur hubungan non kausal ditandai dengan gambar garis dengan dua panah di ujungnya (Q). Namun demikian, meski bentuk garis sama, akan tetapi jika jenis konstrak yang dihubungkan adalah berbeda makna garis berbentuk sama tersebut dapat bermakna berbeda. Selengkapnya jenis-jenis jalur dapat dilihat pada Gambar dibawah ini.



Gambar Fungsi dari Tolls Amos

Goodness of Fit

GOFI	Ukuran Kecocokan Yang Baik
p-value of X^2	≥ 0.05
RMSEA	≤ 0.08
NFI	≥ 0.90
NNFI	≥ 0.90
RFI	≥ 0.90
CFI	≥ 0.90
IFI	≥ 0.90
Standardized RMR	≤ 0.05
GFI	≥ 0.90
AGFI	≥ 0.90

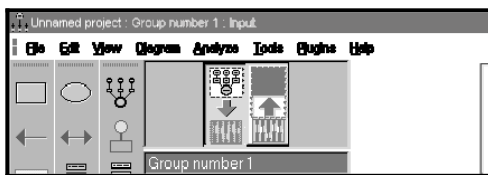
Contoh Kasus

1. Menyiapkan data.

Data yang kita pakai adalah data SPSS.

2. Membuka Program AMOS

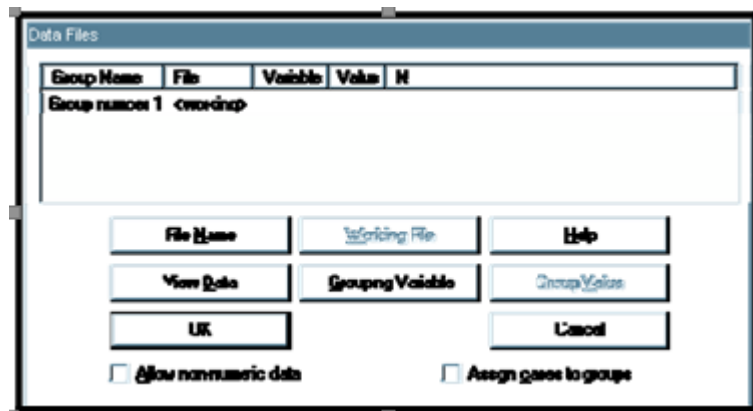
Buka program AMOS dengan membuka Program AMOS GRAPHICS



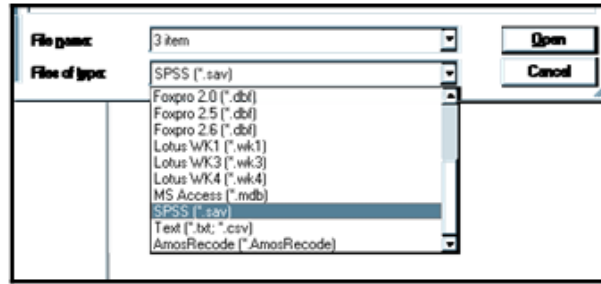
3. Membuat hubungan antara AMOS dan SPSS

Data kita terletak di SPSS sedangkan model kita terletak di AMOS. Langkah ini akan membuat kedua program tersebut menjadi terhubung. Caranya adalah sebagai berikut.

- Di Program AMOS tekan DATA FILES, lalu akan muncul menu di bawah ini.

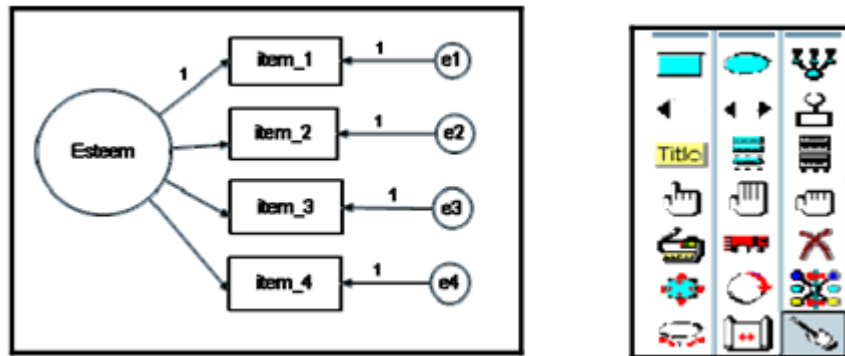


- Tekan FILE NAME lalu pilih NAMA FILE yang berisi data. Pada jendela di bawah ini carilah nama file yang berisi data anda kemudian KLIK file tersebut. Nama file yang muncul di dalam jendela tergantung dari FILE OF TYPES yang dimunculkan. Jika file anda adalah SPSS maka pada FILE OF TYPES pilihlah data berbentuk SPSS. Tekan OK



4. Menggambar Model

Gambarlah model sesuai dengan konsep yang anda kembangkan. Dalam hal ini kita sedang melakukan analisis faktor terhadap pada skala Harga Diri yang terdiri dari 4 item.

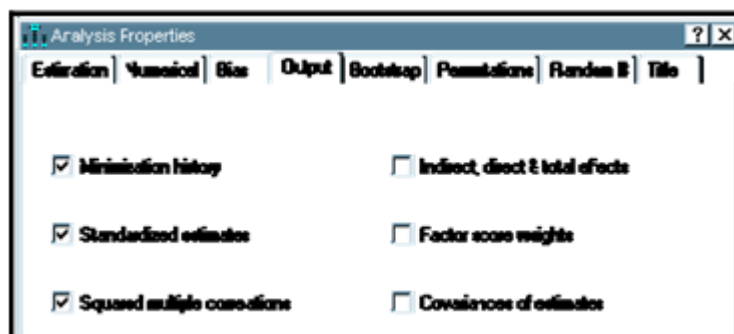


AMOS memfasilitasi anda untuk menggambar model dengan berbagai fitur yang menarik.

5. Memilih Keluaran Analisis

Klik VIEW – ANALYSIS PROPERTIS – lalu pilih OUTPUT.

Langkah ini bertujuan untuk memerintahkan AMOS mengeluarkan informasi hasil analisis. Centang informasi mengenai Standardized Estimates, Square Multiple Correlation dan Modification Indices.



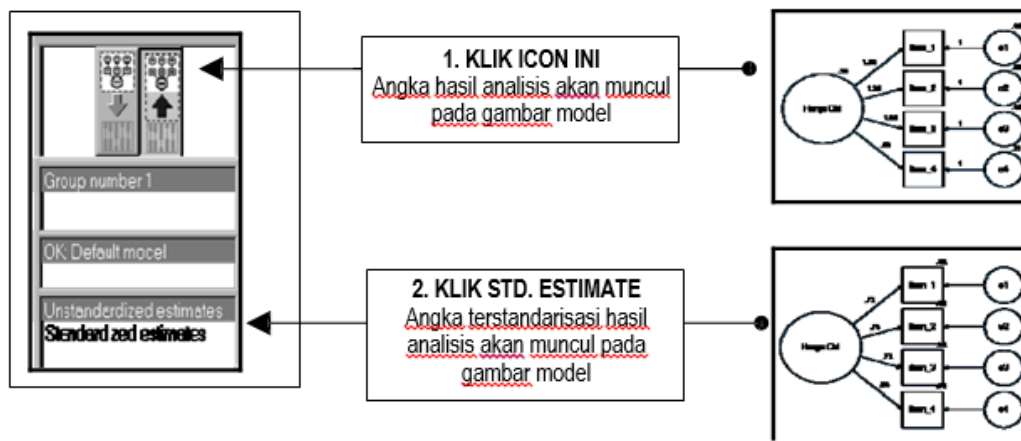
Mencentang Standardized Estimates akan mengeluarkan statistik yang terstandarisasi, Square Multiple Correlation mengeluarkan informasi sumbangan efektif dan Modification Indices mengeluarkan informasi pertimbangan dalam melakukan modifikasi model.

6. Melakukan Analisis

Klik ANALYZE – CALCULATE ESTIMATES atau ikon bergambar piano  untuk menganalisis model anda.

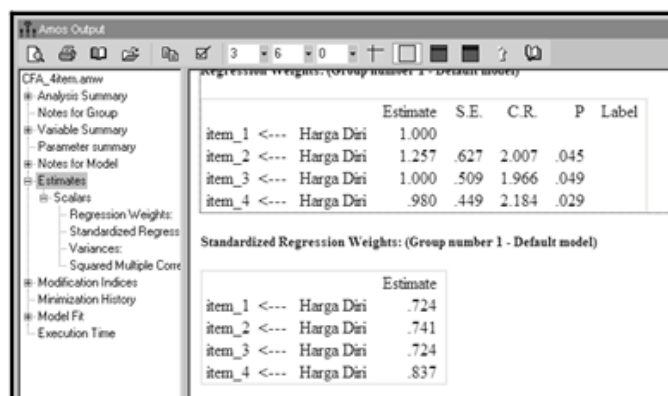
7. Menampilkan Gambar Hasil Analisis

Klik Ikon untuk menampilkan angka-angka hasil analisis di dalam model.



8. Menampilkan Tabel Hasil Analisis

Klik VIEW – TEXT OUTPUT atau langsung tekan F10 untuk menampilkan jendela hasil analisis. Anda tinggal memilih mana informasi yang anda inginkan dengan cara mengarahkan cursor mouse anda pada menu yang tersedia, misalnya ESTIMATE, MODIFICATION INDICES atau MODEL FIT.



	Estimate	S.E.	C.R.	P	Label
item_1 <--- Harga Diri	1.000				
item_2 <--- Harga Diri	1.257	.627	2.007	.045	
item_3 <--- Harga Diri	1.000	.509	1.966	.049	
item_4 <--- Harga Diri	.980	.449	2.184	.029	

	Estimate
item_1 <--- Harga Diri	.724
item_2 <--- Harga Diri	.741
item_3 <--- Harga Diri	.724
item_4 <--- Harga Diri	.837

9. Tinggal Membaca Outputnya

BAB I PENGENALAN

C/C++

1.1 Algoritma & Pemrograman

Algoritma adalah urutan aksi-aksi yang dinyatakan dengan jelas dan tidak rancu untuk memecahkan suatu masalah dalam rentang waktu tertentu. Setiap aksi harus dapat dikerjakan dan mempunyai efek tertentu. Algoritma merupakan logika, metode dan tahapan (urutan) sistematis yang digunakan untuk memecahkan suatu permasalahan. Algoritma dapat dituliskan dengan banyak cara, mulai dari menggunakan bahasa alami yang digunakan sehari-hari, simbol grafik bagan alir (flowchart), sampai menggunakan bahasa pemrograman seperti bahasa C atau C++.

Program adalah kumpulan instruksi komputer, sedangkan metode dan tahapan sistematis dalam program adalah algoritma. Program ini ditulis dengan menggunakan bahasa pemrograman. Jadi bisa kita sebut bahwa program adalah suatu implementasi dari bahasa pemrograman.

Beberapa pakar memberi formula bahwa:

program = struktur data + algoritma

Bagaimanapun juga struktur data dan algoritma berhubungan sangat erat pada sebuah program. Algoritma yang baik tanpa pemilihan struktur data yang tepat akan membuat program menjadi kurang baik, semikian juga sebaliknya. Struktur data disini bisa berupa *list*, *tree*, *graph*, dsb.

1.2 Sejarah C++

C++ adalah pengembangan dari bahasa C, yang merupakan pengembangan dari dua bahasa pemrograman generasi sebelumnya, yaitu BCPL dan B. BCPL dibuat pada tahun 1967 oleh Martin Richards sebagai bahasa untuk menulis sistem operasi dan *compiler*. Ken Thompson membuat banyak fitur pada bahasa B yang dibuatnya dan menggunakan B untuk membuat versi awal dari sistem operasi UNIX di Bell Laboratories pada tahun 1970 pada komputer DEC PDP-7.

Bahasa C dikembangkan dari bahasa B oleh Dennis Ritchie di Bell Laboratories dan pada awalnya diimplementasi pada komputer DEC PDP-11 pada

tahun 1972. C menggunakan banyak konsep penting dari BCPL dan B sekaligus ada tambahan jenis-jenis data dan fitur lainnya. C kemudian dikenal sebagai bahasa pengembang sistem operasi UNIX. Pada masa sekarang, kebanyakan sistem operasi ditulis dengan menggunakan C dan/atau C++. C tersedia untuk hampir semua komputer.

Pada akhir dekade 1970 an, C telah berkembang dengan menjadi sesuatu yang sekarang disebut “C tradisional”, “C klasik”, atau “C Kernighan dan Ritchie”.

C++ adalah penambahan dari C, dikembangkan oleh Bjarne Stroustrup pada awal dekade 1980 an di Bell Laboratories. C++ memberikan tambahan fitur yang meningkatkan kekuatan bahasa C, dan yang lebih penting lagi, kemampuan untuk pemrograman berbasis object (Object Oriented Programming).

1.2 Kelebihan dan Kekurangan

TM Kelebihan Bahasa C/C++

- Bahasa C++ tersedia hampir di semua jenis computer.
- Kode bahasa C/C++ sifatnya adalah portable dan fleksibel untuk semua jenis komputer.
- Proses executable program bahasa C/C++ lebih cepat
- Dukungan pustaka yang banyak.
- C adalah bahasa yang terstruktur.
- C++ Sudah mendukung OOP (Object Oriented Programming).

TM Kekurangan Bahasa C/C++

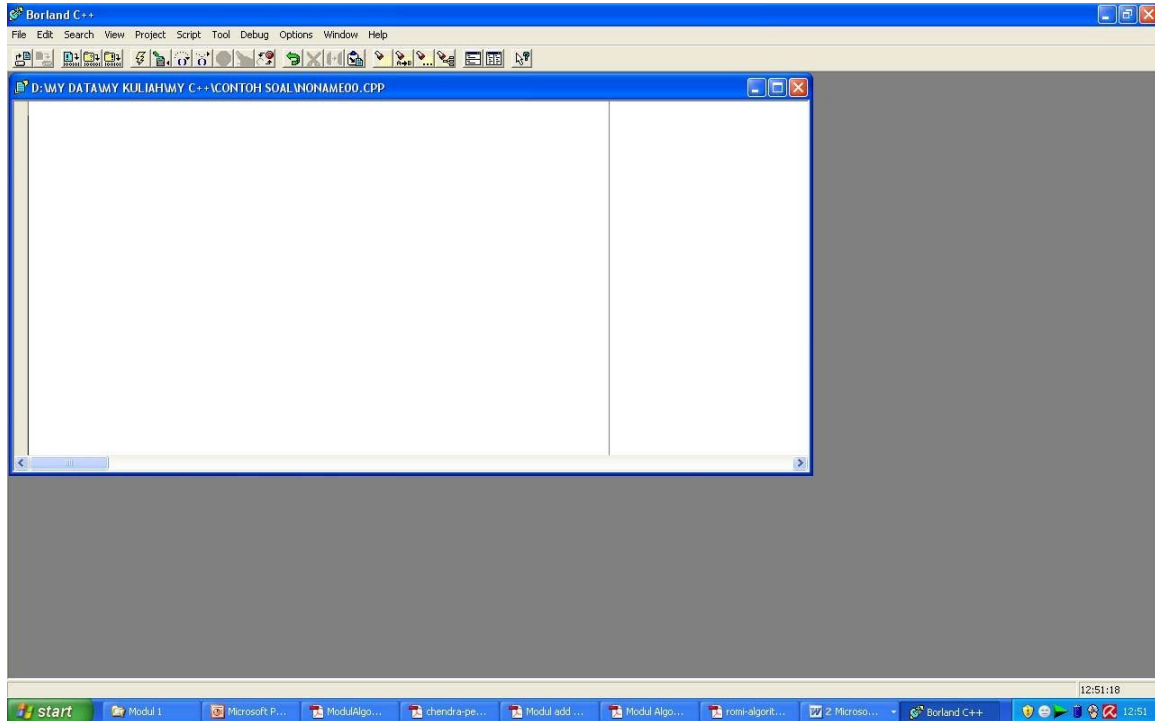
- Banyaknya Operator serta fleksibilitas penulisan program kadang-kadang membingungkan pemakai.
- Bagi pemula pada umumnya akan kesulitan menggunakan pointer dan penerapan konsep OOP.

1.3 Editor Bahasa C/C++

Untuk memulai membuat program, tersedia berbagai editor yang dapat digunakan diantaranya : Turbo C++, Borland C++, C++ Builder, Microsoft Visual C++,
dlsb.

Note :

Seluruh sourcecode program yang ada di tutorial ini 100% dibuat dan telah diuji coba menggunakan Borland C++ 5.02



User Interface Borland C++ 5.02

1.4 Langkah-langkah menuliskan program dalam Borland C++

1. **Bukalah Editor Borland C++ melalui START menu.** Tampilan awal Borland C++ tampak seperti gambar di atas.
2. **Source Code program C/C++ dapat ditulis di text editor Borland C++**

File Æ New Æ Text Edit

3. **Untuk menyimpan project, Pilih menu Save As atau Save (ctrl K + ctrl S)**

4. **Kompile file dengan (ALT + F9 atau pilih submenu Compile)**

compiler dijalankan untuk mengubah source code menjadi sebuah program. *Compile* adalah suatu proses di mana mengubah bahasa pemrograman menjadi instruksi-instruksi yang dikenali oleh komputer. Setelah source code tercompile, terbentuklah sebuah file objek dengan ekstension “ .obj “. File “ .obj “ ini belum merupakan sebuah program executable.

5. Jalankan Program dengan (CTRL+F9 atau pilih submenu Run)

Setelah kita compile file yang berisi source code, maka sebagai hasil kompilasi tersebut kita akan mendapatkan suatu file yang bisa dijalankan (*executable file*). Menjalankan program yang kita buat berarti menjalankan file hasil proses kompilasi tersebut.

Note :

Sebelum mulai melakukan coding program, sebaiknya diingat bahwa bahasa C/C++ bersifat “*case sensitive*”, yang artinya huruf besar dan huruf kecil dibedakan 😊 😊.

BAB II

Struktur Bahasa C/C++

Program Bahasa C/C++ tidak mengenal aturan penulisan di kolom/baris tertentu, jadi bisa dimulai dari kolom/baris manapun. Namun demikian, untuk mempermudah pembacaan program dan untuk keperluan dokumentasi, sebaiknya penulisan program di bahasa C/C++ diatur sedemikian rupa sehingga mudah dan enak dibaca.

Berikut contoh penulisan Program Bahasa C/C++

```
#include <header>
void main()
{
    deklarasi variabel;
    deklarasi konstanta;
    perintah - perintah;
    //komentar
}
```

Cara terbaik untuk belajar bahasa pemrograman adalah dengan langsung mempraktikannya. Cobalah contoh program berikut :

```
#include <iostream.h>
#include <conio.h>
void main()
{
    clrscr();
    cout<<"Hello World"<<endl;
    cout<<"Selamat Belajar C/C++ ";
    cout<<"enter my World";
    getch();
}
```

Penjelasan :

1. include

Adalah salah satu Pengarah Preprosesor (*preprocessor directive*) yang tersedia pada C++. Preprocessor selalu dijalankan terlebih dahulu pada saat proses kompilasi terjadi. Bentuk umumnya :

```
# include <nama_file>
```

tidak diakhiri dengan tanda semicolon (;), karena bentuk tersebut bukanlah suatu bentuk pernyataan, tetapi merupakan preprocessor directive. Baris tersebut menginstruksikan kepada kompiler untuk menyisipkan file lain dalam hal ini file yang berakhiran .h (file header) yaitu file yang berisi C++ *standard library*. contohnya:

- #include <iostream.h> : diperlukan pada program yang melibatkan objek **cout** dan **cin**
- #include <conio.h> : diperlukan bila melibatkan clrscr(), yaitu perintah untuk membersihkan layar dan fungsi getch() untuk menerima sembarang input keyboard dari user.
- #include <iomanip.h> : diperlukan bila melibatkan setw() yang bermanfaat untuk mengatur lebar dari suatu tampilan data.
- #include <math.h> : diperlukan pada program yang menggunakan operasi sqrt() yang bermanfaat untuk operasi matematika kuadrat.

2.Fungsi main ()

Program C++ terdiri dari satu atau lebih fungsi, dan di antara salah satunya harus ada fungsi **main** dan hanya boleh ada satu **main** pada tiap program C++. Setiap program C++ akan dan pasti akan memulai eksekusi programnya pada fungsi **main** ini, meskipun main bukan fungsi yang pertama ditulis di program.

Melihat bentuk seperti itu dapat kita ambil kesimpulan bahwa batang tubuh program utama berada didalam fungsi **main()**. Berarti dalam setiap pembuatan program utama, maka dapat dipastikan seorang pemrogram menggunakan minimal sebuah fungsi.

Tanda { dan pada akhir program terdapat tanda }. Tanda { harus ada pada setiap awal dari sebuah fungsi dan tentu saja harus diakhiri dengan tanda }. Tanda

ini digunakan untuk menunjukkan cakupan(*scope*) dari sebuah fungsi, dimana untuk menunjukkan fungsi ini dimulai dan berakhir.

3. Komentar

Komentar tidak pernah dicompile oleh compiler. Dalam C++ terdapat 2 jenis komentar, yaitu:

Jenis 1 : /* Komentar anda diletakkan di dalam ini

Bisa mengapit lebih dari satu baris */

Jenis 2 : // Komentar anda diletakkan disini (hanya bisa sebaris)

Programmer sering sekali memasukkan komentar di dalam code agar program lebih mudah dibaca. Komentar juga membantu orang lain untuk membaca dan mengerti isi dari code. Komentar tidak menyebabkan komputer melakukan suatu instruksi ketika program dijalankan.

4. Tanda Semicolon

Tanda semicolon “ ; ” digunakan untuk mengakhiri sebuah pernyataan. Setiap pernyataan harus diakhiri dengan sebuah tanda semicolon.

5. Mengenal Input/Output

Pernyataan `cout` (dibaca C out) merupakan sebuah objek di dalam C++, yang digunakan untuk mengarahkan data ke dalam standar output (cetak pada layar). Sedangkan untuk menginputkan data, dapat digunakan `cin` (dibaca C in).

Berikutnya adalah operator `<<` Operator ini digunakan sebagai penghubung antara stream dengan kalimat. Operator ini disesuaikan dengan fungsional dari `cout`. Untuk sementara bayangkan saja operator `<<` sebagai arah dari aliran data. Jadi karena kita ingin mencetak kalimat ke layar, dan yang menghubungkan program kita dengan layar dengan `cout`, otomatis kita harus mengirimkan kalimat ke `cout`. Maka operator `<<` digunakan, yang berarti kalimat dialirkan ke arah `cout`, dan `cout` akan mencetaknya ke layar.

Sintaks yang digunakan :

`cout << daftar_keluaran`

`cin >> daftar_masukan`

endl merupakan suatu fungsi manipulator yang digunakan untuk menyisipkan karakter NewLine atau mengatur pindah baris. Fungsi ini sangat berguna untuk piranti keluaran berupa file di disk. File header yang harus disertakan adalah file header **iostream.h**

Fungsi **getch()** (get character and echo) dipakai untuk membaca sebuah karakter dengan sifat karakter yang dimasukkan tidak perlu diakhiri dengan menekan tombol ENTER, dan karakter yang dimasukan tidak akan ditampilkan di layar. File header yang harus disertakan adalah **conio.h**

Latihan Soal 1.

```
#include"iostream.h" //preprocessor
#include<conio.h>
void main() //ada 3; void main(), main() & int main()
{
cout<<"Hello world\n"; //cout untuk menampilkan ke layar
//cout<<"Hello world"<<endl;
getch();
}
```

Latihan Soal 2.

```
// programku
#include <iostream.h>
int main ()
{
cout << "Selamat Belajar C++";
return 0;
}
```

BAB III

Identifier, Tipe Data, Variabel, Konstanta, Operator

3.1 Identifier

Aturan pemberian nama suatu pengenalan/identifier :

- nama pengenalan harus dimulai dengan karakter berupa huruf (a...z, A...Z) atau karakter garis bawah (_)
- karakter berikutnya dapat berupa huruf, angka (0...9) atau karakter garis bawah (_)
- tidak boleh sama dengan reserved word (nama – nama yang sudah digunakan dalam bahasa C++) seperti char, int, float, double, void, dll.
- panjang karakter maksimum adalah 32 karakter
- bersifat case sensitive (huruf besar dan huruf kecil dibedakan)

Pada bahasa C, yang termasuk reserved words antara lain:

break	case	char	const	continue	default	do
double	else	enum	float	for	goto	if
inline	int	long	return	short	signed	sizeof
static	struct	switch	typedef	union	unsigned	void
while						

3.2 Tipe Data

Tipe data merupakan bagian program yang paling penting karena tipe data mempengaruhi setiap instruksi yang akan dilaksanakan oleh computer. Misalnya saja 5 dibagi 2 bisa saja menghasilkan hasil yang berbeda tergantung tipe datanya. Jika 5 dan 2 bertipe integer maka akan menghasilkan nilai 2, namun jika keduanya bertipe float maka akan menghasilkan nilai 2,5. Pemilihan tipe data yang tepat akan membuat proses operasi data menjadi lebih efisien dan efektif.

Tipe	Ukuran (bits)	Range
unsigned char	8	0 s/d 255
char	8	-128 s/d 127
short int	16	-32,768 s/d 32,767
unsigned int	32	0 s/d 4,294,967,295
int	32	-2,147,483,648 s/d 2,147,483,647
unsigned long	32	0 s/d 4,294,697,295
long	32	-2,147,483,648 s/d 2,147,483,647
float	32	3.4 e-38 s/d 1.7 E +38
double	64	1.7 E-308 s/d 3.4 E + 308
long double	80	3.4 E-4932 s/d 1.1 E + 4932

Type	Keterangan
bool	isi bilangan Boolean (True dan False)
wchar_t	wide character

3.3 Konstanta

Konstanta merupakan suatu nilai yang tidak dapat diubah selama proses program berlangsung. Konstanta nilainya selalu tetap. Konstanta harus didefinisikan terlebih dahulu di awal program. Konstanta dapat bernilai integer, pecahan, karakter dan string.

Pendeklarasian konstanta dapat dilakukan dengan 2 cara :

1. menggunakan (**#define**)

deklarasi konstanta dengan cara ini, lebih gampang dilakukan karena akan menyertakan **#define** sebagai preprocessor directive. Dan sintaknya diletakkan bersama – sama dengan pernyataan **#include** (di atas **main()**).

Format penulisannya adalah :

```
#define pengenalan nilai
```

Contoh penggunaan :

```
#define phi 3.14159265
```

pendeklarasian dengan **#define** tanpa diperlukan adanya tanda = untuk memasukkan nilai ke dalam pengenalan dan juga tanpa diakhiri dengan tanda semicolon(;

2. menggunakan (**const**)

Sedangkan dengan kata kunci `const`, pendeklarasian konstanta mirip dengan deklarasi variable yang ditambah kata depan `const`

Contoh :

```
const int lebar = 100;
const char tab = '\t';
```

3.4 Variabel

Variabel adalah suatu pengenalan (identifier) yang digunakan untuk mewakili suatu nilai tertentu di dalam proses program. Berbeda dengan konstanta yang nilainya selalu tetap, nilai dari suatu variable bisa diubah-ubah sesuai kebutuhan. Bentuk umum pendeklarasian suatu variable adalah :

```
tipe_data nama_variabel;
```

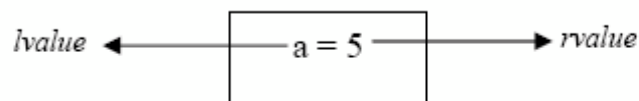
Contoh :

```
int x; // Deklarasi x bertipe integer
char y, huruf, nim[10]; // Deklarasi variable bertipe char
float nilai1; // Deklarasi variable bertipe float
double beta; // Deklarasi variable bertipe double
int array[5][4]; // Deklarasi array bertipe integer
char *p; // Deklarasi pointer p bertipe char
```

3.5 Operator

1. Operator Assign (=)

Operator (=), akan memberikan nilai ke dalam suatu variable.



artinya memberikan nilai 5 ke dalam variable a. Sebelah kiri tanda = dalam pernyataan di atas, dikenal dengan ***lvalue*** (left value) dan di sebelah kanan tanda = dikenal dengan ***rvalue*** (right value). lvalue harus selalu berupa variable, sedangkan rvalue dapat berupa variable, nilai, konstanta, hasil operasi ataupun kombinasinya.

2. Operator Majemuk (+=, -=, *=, /=, %=, <<=, >>=, &=, |=)

Dalam C++, operasi aritmatika dapat disederhanakan penulisannya dengan format penulisan operator majemuk.

Misalnya :

a += 5 sama artinya dengan menuliskan a = a+5
a *= 5 sama artinya dengan menuliskan a = a*5
a /= 5 sama artinya dengan menuliskan a = a/5
a %= 5 sama artinya dengan menuliskan a = a % 5

Assignment Operator	Operasi aritmatik biasa	Assignment operator	Hasil
+=	i = i + 2	i += 2	i bernilai 3
-=	j = j - 3	j -= 3	j bernilai 0
*=	k = k * 4	k *= 4	k bernilai 12
/=	l = l / 4	l /= 4	l bernilai 1
%=	m = m % 2	m %= 2	m bernilai 1

3. Operator Penaikan dan Penurunan (++ dan --)

Operator penaikan (++) akan menaikkan atau menambahkan 1 nilai variable. Sedangkan operator (--) akan menurunkan atau mengurangi 1 nilai variable. Misalnya :

```
a++;
a+=1;
a=a+1;
```

untuk ketiga pernyataan tersebut, memiliki arti yang sama yaitu menaikkan 1 nilai. Karakteristik dari operator ini adalah dapat dipakai di awal (++a) atau di akhir (--a) variable. Untuk penggunaan biasa, mungkin tidak akan ditemui perbedaan hasil dari cara penulisannya. Namun untuk beberapa operasi nantinya harus diperhatikan cara peletakan operator ini, karena akan berpengaruh terhadap hasil.

Contoh 1 :

```
B=3;
A=++B;
//hasil A= 4, B=4
```

Contoh 2:

```
B=3;
A=B++;
//hasil A=3, B=4
```

Dari contoh1, nilai B dinaikkan sebelum dikopi ke variable A. Sedangkan pada contoh2, nilai B dikopi terlebih dahulu ke variable A baru kemudian dinaikkan.

Beda dari operator --/++ di sebelah kiri variabel dengan --/++ di sebelah kanan variabel bisa dilihat dari contoh berikut ini:

```
int i = 10;
cout << --i << endl;
cout << i << endl;
```

hasil output:

9
9

```
int i = 10;
cout << i-- << endl;
cout << i << endl;
```

hasil output:

10
9

Jadi bisa diambil kesimpulan, dengan operator --/++ (--i) di sebelah kiri variabel maka operator tersebut akan mempunyai prioritas lebih tinggi untuk dikerjakan terlebih dahulu. Jadi i akan dikurangi terlebih baru dicetak oleh cout. Sebaliknya dengan operator --/++ (i--) di sebelah kanan variabel maka operator tersebut akan mempunyai prioritas lebih rendah untuk dikerjakan. Maka i akan dicetak terlebih dahulu, baru dikurangi.

4. **Operator Relasional** (==, !=, >, <, >=, <=)

Yang dihasilkan dari operator ini bukan berupa sebuah nilai, namun berupa bilangan bool yaitu benar atau salah.

Operator	Keterangan
==	Sama dengan
!=	Tidak sama dengan
>	Lebih besar dari
<	Kurang dari

>=	Lebih besar dari atau sama dengan
<=	Kurang dari atau sama dengan

Contoh :

(7==5) hasilnya adalah **false**

(5>4) hasilnya adalah **true**

(5<5) hasilnya adalah **false**

5. Operator Logika (!, &&, ||)

Operator logika juga digunakan untuk memberikan nilai atau kondisi **true** dan **false**. Biasanya operator logika dipakai untuk membandingkan dua kondisi. Misalnya:

((5==5) && (3>6)) mengembalikan nilai **false**, karena (true && false) untuk logika NOT (!), contohnya !(5==5) akan mengembalikan nilai **false**, karena **!(true)**.

Latihan Soal 1.

```
#include <conio.h>
#include <iostream.h>
void main()
{
    const float phi = 3.141592;
    float jari_jari, keliling, luas;
    jari_jari = 7.2;
    luas = phi * jari_jari * jari_jari;
    keliling = 2 * phi * jari_jari;
    cout<<"Luas lingkaran adalah " << luas << " satuan luas "<<endl;
    cout<<"Keliling lingkaran adalah " << keliling << " satuan panjang";
    getch();
}
```

Latihan Soal 2.

```
#include"iostream.h"
#include"conio.h"
//#include"stdio.h"
void main()
{
    float data1,data2,tambah,kurang,kali,bagi;
    cout<<"Operasi aritmatika Dasar"<<endl;
    cout<<"Masukkan data1: ";
    cin>>data1;
    cout<<"Masukkan data2: ";
    cin>>data2;
```

```

        tambah=data1+data2;
        kurang=data1-data2;
        kali=data1*data2;
        bagi=data1/data2;
        cout<<endl;
        cout<<data1<<" + " <<data2<<" = "<<tambah<<endl;
        cout<<data1<<" - " <<data2<<" = "<<kurang<<endl;
        cout<<data1<<" * " <<data2<<" = "<<kali<<endl;
        cout<<data1<<" : " <<data2<<" = "<<bagi;
        //printf("%6.2f",bagi);
        getch();
    }

```

Latihan Soal 3.

Buatlah sebuah program untuk menghitung jumlah dan rata-rata dari 5 buah bilangan bulat positif.

Latihan Soal 4.

Suatu ember berbentuk tabung dengan tutupnya terbuka berisi air penuh. Jari- jari alas ember adalah 10.5 cm, dan tingginya 5 cm. Kemudian sebuah kerucut dengan jari-jari alas yang berbentuk lingkaran adalah 4 cm dan tingginya 4.7 cm dimasukkan ke dalam ember. Akibatnya sebagian air dalam ember tumpah. Dengan menggunakan program C++ hitunglah berapa **liter** air yang tumpah?

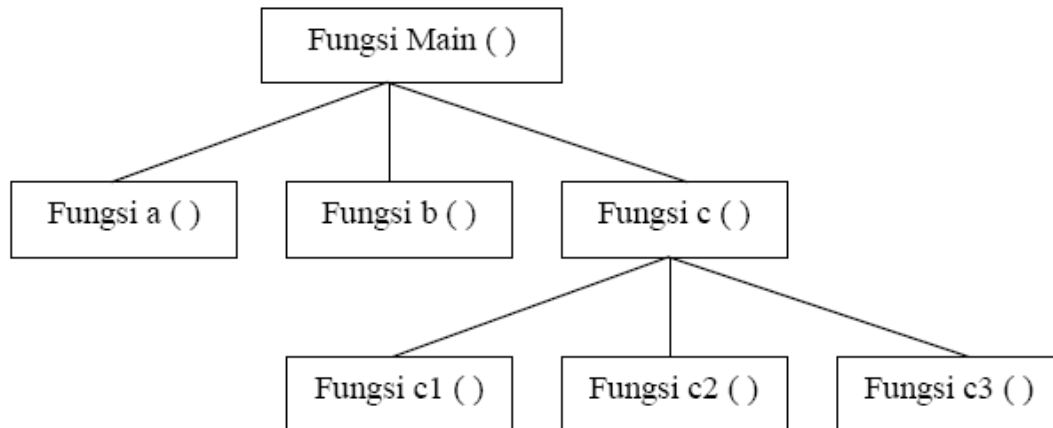
BAB IV Function dan Procedure

Procedure dan Function disebut juga subroutine, merupakan blok statement yang dapat dipanggil dari lokasi yang berbeda di dalam program. Yang membedakan antara function dan procedure yaitu: suatu function jika dijalankan/dipanggil akan mengembalikan suatu nilai.

Dalam PASCAL dikenal istilah procedure dan function, dalam Basic dikenal sub dan function, sedangkan dalam C++, Java, PHP, dan keturunan C lainnya dikenal hanya istilah function. Apabila kita ingin membuat subroutine yang tidak mengembalikan nilai, kita dapat memberi nilai kembalian berupa **void**.

Fungsi (Function) merupakan blok dari kode yang dirancang untuk melaksanakan tugas khusus. Pada intinya fungsi berguna untuk :

- Mengurangi pengulangan penulisan program yang berulang atau sama.
- Dapat melakukan pendekatan top-down dan divide-and-conquer: program besar dapat dipisah menjadi program-program kecil.
- Program menjadi terstruktur, sehingga mudah dipahami dan dikembangkan.
- Kemudahan dalam mencari kesalahan-kesalahan karena alur logika jelas dan kesalahan dapat dilokalisasi dalam suatu modul tertentu saja.
- Modifikasi program dapat dilakukan pada suatu modul tertentu saja tanpa mengganggu program keseluruhan.
- Mempermudah dokumentasi.
- Reusability: Suatu fungsi dapat digunakan kembali oleh program atau fungsi lain



Kategori Function dalam C/C++

1. Standard Library Function

Yaitu fungsi-fungsi yang telah disediakan oleh C/C++ dalam file-file header atau librarynya. Misalnya: `clrscr()`, `printf()`, `getch()`

Untuk function ini kita harus mendeklarasikan terlebih dahulu library yang akan digunakan, yaitu dengan menggunakan preprosesor direktif.

2. Programmer-Defined Function

Adalah function yang dibuat oleh programmer sendiri. Function ini memiliki nama tertentu yang unik dalam program, letaknya terpisah dari program utama, dan bisa dijadikan satu ke dalam suatu library buatan programmer itu sendiri yang kemudian juga di-include-kan untuk penggunaannya.

Struktur Function

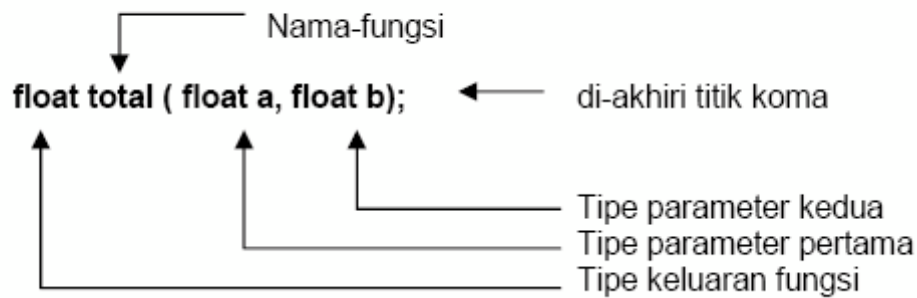
1. Function Prototype

Prototipe fungsi digunakan untuk menjelaskan kepada kompiler mengenai :

- Tipe keluaran fungsi.
- Jumlah parameter.
- Tipe dari masing-masing parameter.

Salah satu keuntungan pemakai prototipe, kompiler akan melakukan konversi antara tipe parameter dalam definisi dan parameter saat pemanggilan fungsi tidak sama atau akan menunjukkan kesalahan jika jumlah parameter dalam definisi dan saat pemanggilan berbeda.

Contoh prototipe fungsi :



Contoh lain :

```
long kuadrat (long l) ;
```

Pada contoh pertama, fungsi `kuadrat ()` mempunyai argumen/parameter bertipe `long` dan nilai balik bertipe `long`.

```
void garis ( ) ;
```

Pada contoh kedua, fungsi `garis ()` tidak memiliki argumen/parameter dan nilai baliknya tidak ada (`void`).

```
double maks (double x, double y)
```

Pada contoh ketiga, fungsi `maks()` mempunyai dua buah argumen/parameter, dengan masing-masing argumen bertipe `double`.

2. Function Definition

```
tipe_data nama_fungsi(arguman 1, argument 2, argument ...)  
{  
    Variabel_lokal;  
    Statement_1;  
    Statement_2;  
    ...  
    return (variabel);  
}
```

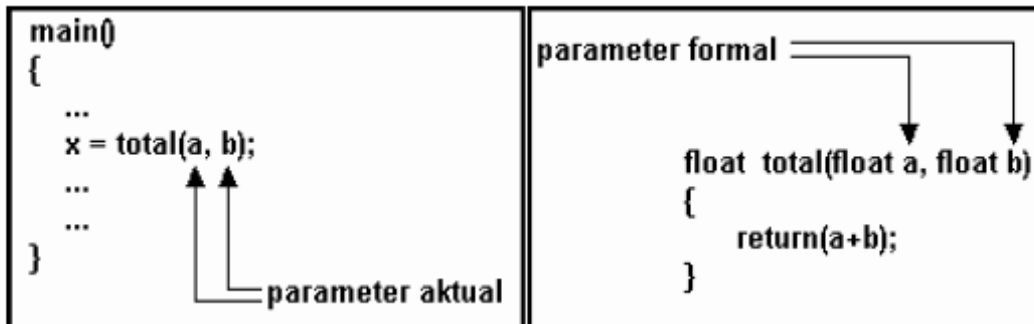
3. Parameter Fungsi

Terdapat dua macam parameter fungsi, yaitu :

1. **Parameter formal** adalah variabel yang ada pada daftar parameter dalam definisi fungsi.

2. **Parameter Aktual** adalah variabel yang dipakai dalam pemanggilan fungsi.

Bentuk penulisan Parameter Formal dan Parameter Aktual.



4. Contoh Penerapan

Contoh 1.

```
/*Contoh penerapan function dg prototype tapi tidak memberikan
nilai balik (void) */
#include<iostream.h>
#include<conio.h>

void gaya(double m, double a); //prototype function

void main()    //main function
{
    double m,a;
    cout<<"Massa      : "; cin>>m;
    cout<<"percepatan : "; cin>>a;
    cout<<"F : ";
    gaya(m,a);    //parameter aktual
    getch();
}

void gaya(double m, double a)    //function definition
{
    double hasil;
    hasil=m*a;
    cout<<hasil;
}
```

Selain menggunakan prototype, penerapan fungsi juga dimungkinkan tanpa penggunaan prototype. Tetapi definisi fungsi harus diletakkan diatas fungsi main().

Contoh 2.

```
/*Contoh penerapan function tanpa prototype tapi memberikan nilai
balik */
#include<iostream.h>
#include<conio.h>
```

```
double gaya(double m, double a) //tanpa prototype
{
    double hasil;
    hasil=m*a;
    return (hasil); //memberikan nilai balik bertipe data double
}

void main()
{
    double m,a,f;
    cout<<"Massa      : "; cin>>m;
    cout<<"percepatan : "; cin>>a;
    f=gaya(m,a);
    cout<<"F : "<<f;
    getch();
}
```

5. Cara pelewatan argumen fungsi

a. Pass by value

Pemanggilan dengan nilai merupakan cara yang dipakai untuk seluruh fungsi buatan yang telah dibahas didepan. Pada pemanggilan dengan nilai, nilai dari parameter aktual akan ditulis keparameter formal. Dengan cara ini nilai parameter aktual tidak bisa berubah, walaupun nilai parameter formal berubah.

b. Pas by Reference

Pemanggilan dengan reference merupakan upaya untuk melewatkan alamat dari suatu variabel kedalam fungsi. Cara ini dapat dipakai untuk mengubah isi suatu variabel diluar fungsi dengan melaksanakan pengubahan dilakukan didalam fungsi.

Contoh 3.

```
/* ----- */
/* Penggunaan Pass By Reference */
/* Program Pertukaran Nilai      */
/* ----- */
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
void tukar(int *x, int *y);
void main()
{
    int a, b;
    a = 88; b
    = 77;
    clrscr();
    cout<<"Nilai Sebelum Pemanggilan Fungsi"<<endl;
    cout<<"a = "<<a<<" b = "<<b<<endl;
    tukar(&a,&b);
    cout<<endl;
```

```

        cout<<"Nilai Setelah Pemanggilan Fungsi"<<endl;
        cout<<"a = "<<a<<" b = "<<b<<endl;
        getch();
    }
    void tukar(int *x, int *y)
    {
        int z;
        z = *x;
        *x = *y;
        *y = z;
        cout<<endl;
        cout<<"Nilai di Akhir Fungsi Tukar() "<<endl;
        cout<<"x = "<<*x<<" y = "<<*y<<endl;
    }

```

Contoh 4.

```

#include <iostream.h>
#include <conio.h>
void Ubah(int *a)
{
    *a = 10;
    cout<<"Ubah menjadi= "<<*a<<endl;
}
/*main program*/
void main()
{
    int bil;
    bil = 1;
    cout<<"Bil sebelum = "<<bil<<endl;
    Ubah(&bil);
    cout<<"Bil sesudah = "<<bil<<endl;
    getch();
}

```

- Pemanggilan secara Referensi merupakan upaya untuk melewatkan alamat dari suatu variabel ke dalam fungsi.
- Yang dikirimkan ke fungsi adalah alamat letak dari nilai datanya, bukan nilai datanya.
- Fungsi yang menerima kiriman alamat ini akan menggunakan alamat yang sama untuk mendapatkan nilai datanya.
- Perubahan nilai di fungsi akan merubah nilai asli di bagian program yang memanggil fungsi.
- Pengiriman parameter secara referensi adalah pengiriman dua arah, yaitu dari fungsi pemanggil ke fungsi yang dipanggil dan juga sebaliknya.
- Pengiriman secara acuan tidak dapat dilakukan untuk suatu ungkapan.

6. Fungsi Rekursif

Fungsi rekursif adalah suatu fungsi yang memanggil dirinya sendiri, artinya fungsi tersebut dipanggil di dalam tubuh fungsi itu sendiri. Salah satu kelemahan fungsi rekursif adalah waktu komputasi yang lebih lama.

```
#include<iostream.h>
#include<conio.h>

long factorial (long a)
{
    if (a>1)
        return (a* factorial (a-1));
    else
        return (1);
}

int main()
{
    long l;
    cout<<"tuliskan bilangan : ";
    cin>>l;
    cout<<"!"<<l<<" = "<<factorial(l);
    return 0;
}
```

Hasil :

Tuliskan bilangan : 9

!9 = 362880

Latihan Soal.

Buatlah Program (dg function) untuk menghitung jarak maksimum (xmax) dan ketinggian maksimum (hmax) dari sebuah peluru yang ditembakkan dengan sudut elevasi A. Anggap $g = 10 \text{ m/s}^2$ (Gunakan fungsi $\sin()$ dan $\cos()$)

BAB V

FILE dan STREAM

Operasi dasar file pada prinsipnya terbagi menjadi 3 tahap, yaitu:

- a. membuka atau mengaktifkan file
- b. melaksanakan pemrosesan file
- c. menutup file

5.1 Membuka file

Sebelum suatu file dapat diproses, file harus dibuka terlebih dahulu. Sebelum file dibuka, terlebih dahulu obyek file harus didefinisikan. Sintaksnya:

```
ofstream nama_obyek;
```

perintah ofstream dapat dijalankan dengan menyertakan file header **fstream.h**. Setelah itu, suatu file dapat dibuka dengan perintah

```
nama_obyek.open("nama file dan path");
```

5.2 Menulis ke File

Salah satu jenis pemrosesan pada file adalah menulis atau merekam data ke file. Sintaknya:

```
nama_obyek << ... ;
```

5.3 Menutup File

Setelah pemrosesan file selesai, file dapat ditutup menggunakan perintah

```
nama_obyek.close();
```

Contoh 1. Program berikut ini untuk menulis teks ke dalam file

```
#include<iostream.h>
#include<fstream.h>
void main()
{
    ofstream fileteks;
    fileteks.open("C:/algo.txt");
```



```

fileteks<<"Untuk mencapai tujuan yg besar, maka tujuan itu "
<<endl;
fileteks << "harus dibagi-bagi menjadi tujuan kecil"<< endl;
fileteks << "sampai tujuan itu merupakan tujuan yg dapat "
<< "dicapai" << endl;
fileteks << "berdasarkan kondisi dan potensi yg dimiliki saat "
<< itu " << endl;
fileteks.close();
}

```

perintah `fileteks.open("C:/algo.txt");` akan membuka file `algo.txt` yang ada di `C:\`. Apabila file tersebut belum ada maka akan dibuat secara otomatis, dan apabila sudah ada isi file `algo.txt` akan terhapus.

5.4 Menambah Data pada File

Suatu file yang sudah ada sebelumnya dapat ditambah data yang baru (tidak menghapus data lama). Caranya dengan menambahkan perintah **`ios::app`** pada `open()`.

```

nama_objek.open("nama file", ios::app);

```

Contoh 2.

```

#include<iostream.h>
#include<fstream.h>
void main()
{
    ofstream fileteks;
    fileteks.open("C:/algo.txt", ios::app);
    fileteks << endl;
    fileteks << "Oleh: Al Khowarizmi << endl;
    fileteks.close();
}

```

5.5 Memeriksa Keberhasilan Operasi File

Tidak selamanya jalan yang mulus ditemui. Ada kemungkinan terjadi saat file dibuka, ternyata file tidak ada. Dalam C++ tersedia function untuk memeriksa kondisi-kondisi pada operasi file, sehingga kesalahan saat eksekusi dapat dikendalikan. Function yang dimaksud adalah **`fail()`**.

Contoh 3:

```

#include<iostream.h>
#include<fstream.h>
void main()

```

```

{
ifstream fileteks; { ifstream digunakan u/ membaca file }
fileteks.open("C:/algo.txt");
if (fileteks.fail()) cout << "Maaf file takdapat dibuka/"
<< "tidak ditemukan";
fileteks.close();
}

```

5.6. Operasi Berbasis Karakter

Operasi file dapat dilakukan dalam bentuk karakter. Misalnya proses penyimpanan data ke file dilakukan setiap karakter, atau membaca data file karakter per karakter. Operasi ini didukung oleh function **put()** dan **get()**. Contoh 4. Program untuk menyimpan data karakter per karakter ke dalam file.

```

#include<iostream.h>
#include<fstream.h>
void main()
{
ofstream fileteks;
fileteks.open("C:/contoh.txt");
fileteks.put('A');
fileteks.put('B');
fileteks.put('C');
fileteks.close();
}

```

Contoh 5. Program untuk membaca file karakter per karakter

```

#include<iostream.h>
#include<fstream.h>
void main()
{
char karakter;
ifstream fileteks; {}
fileteks.open("C:/contoh.txt");
while(!fileteks.eof())
{
fileteks.get(karakter);
cout << karakter;
}
fileteks.close();
}

```

Latihan Soal.

1. Buatlah program C++ untuk menghitung jumlah karakter dalam suatu file. Inputnya adalah nama file dan pathnya. Jangan lupa error handling!
2. Buatlah program C++ untuk menghitung jumlah karakter tertentu, misalnya karakter 'A'. Input berupa nama file dan karakter yang akan dihitung. Jangan lupa error handling!
3. Misalkan suatu file teks berisi listing program C++. Buatlah program untuk menghitung pasangan kurung kurawal yang ada pada file teks tersebut. Jangan lupa error handling!

BAB VI

Struktur Kontrol Kondisional & Perulangan

6.1 Struktur Kondisional

6.1.1 Statement IF

Pernyataan Percabangan digunakan untuk memecahkan persoalan dalam mengambil suatu keputusan diantara sekian kondisi yang ada.

Syntax nya

```
if (kondisi)
{
    pernyataan;
    .....
}
```

Pernyataan IF diatas mempunyai pengertian, “ *Jika kondisi bernilai benar, maka perintah/pernyataan akan dikerjakan dan jika tidak memenuhi syarat maka akan diabaikan*”.

Jika 'pernyataan' yang dijalankan hanya sebaris, maka tanda {} **boleh ditiadakan**. Statement 'kondisi' harus merupakan statement Relasional ataupun logika! (Baca kembali BAB 3.5.4 & 3.5.5)

Contoh 1:

```
#include <conio.h>
#include <iostream.h>
void main()
{
    int usia;
    clrscr();
    cout << "Berapa usia Anda : ";
    cin >> usia;
    if (usia < 17)
    cout << "Anda tidak boleh menonton bioskop";
    getch();
}
```

Contoh 2:

```
#include <conio.h>
#include <iostream.h>
void main()
{
    int usia;
    clrscr();
    cout << "Berapa usia Anda : ";
    cin >> usia;
```

```

if (usia < 17)
{
    cout << "Anda tidak boleh menonton bioskop"<<endl;
    cout << "Kerjakan PR anda...";
}
getch();
}

```

Statement IF juga dapat ditambahkan ELSE sebagai konsekuensi alternatif jika kondisi tidak dipenuhi (FALSE). Sintaksnya:

```

if (kondisi)
    perintah-1;
else
    perintah-2;

```

Perintah-1 dan perintah-2 dapat berupa sebuah pernyataan tunggal, pernyataan majemuk atau pernyataan kosong. Jika pemakaian if-else diikuti dengan pernyataan majemuk, bentuk penulisannya sebagai berikut :

```

if (kondisi)
{
    perintah-1;
    ...
}
else
{
    perintah-2;
    ...
}

```

Pernyataan if diatas mempunyai pengertian, “ *Jika kondisi bernilai benar, maka perintah-1 akan dikerjakan dan jika tidak memenuhi syarat maka perintah-2 yang akan dikerjakan*”.

Contoh 3:

```

#include <conio.h>
#include <iostream.h>
void main()
{
    int usia;

```

```

clrscr();
cout << "Berapa usia Anda : ";
cin >> usia;
if (usia < 17)
{
    cout << "Anda tidak boleh menonton bioskop"<<endl;
    cout << "Kerjakan PR anda...";
}
else
{
    cout << "Anda Boleh ke Bioskop."<<endl;
    cout << "Belikan 1 Tiket Buat ASDOS";
}
getch();
}

```

Selain format penulisan statement IF diatas, berikut adalah beberapa format penulisan statement IF lainnya:

- IF Else Majemuk

```

if (syarat)
{
    ... perintah;
    ... perintah;
}
else if (syarat)
{
    ... perintah;
    ... perintah;
}
else
{
    ... perintah;
    ... perintah;
}

```

- Nested IF (IF Bersarang)

Nested if merupakan pernyataan if berada didalam pernyataan if yang lainnya.

Bentuk penulisan pernyataan Nested if adalah :

```

if(syarat)
    if(syarat)
        ... perintah;
    else
        ... perintah;
else
    if(syarat)
        ... perintah;
    else
        ... perintah;

```

6.1.2 Statement SWITCH – CASE

Pernyataan **switch** adalah pernyataan yang digunakan untuk menjalankan salah satu pernyataan dari beberapa kemungkinan pernyataan, berdasarkan nilai dari sebuah ungkapan dan nilai penyeleksian.

Syntaksnya :

```

switch (ekspresi)
{
    case konstanta1 :
        pernyataan1 ;
        break ;
    case konstanta2 :
        pernyataan2 ;
        break ;
    case konstanta3 :
        pernyataan3 ;
        break ;
    :
    :
    case konstantaN :
        pernyataanN ;
        break ;
    default :
        pernyataanlain;
}

```

Hal – hal yang perlu diperhatikan adalah :

1. Dibelakang keyword case harus diikuti oleh sebuah konstanta, tidak boleh diikuti oleh kondisional.
2. Konstanta yang digunakan bertipe int atau char
3. Jika bentuknya seperti diatas maka apabila *ekspresi* sesuai dengan konstanta2 maka pernyataan2, pernyataan3 sampai dengan pernyataanlain dieksekusi. Untuk mencegah hal tersebut, gunakan keyword **break**;. Jika keyword **break** digunakan maka setelah pernyataan2 dieksekusi program langsung keluar dari pernyataan **switch**. Selain digunakan dalam **switch**, keyword *break* banyak digunakan untuk keluar dari pernyataan yang berulang (looping).
4. pernyataan 'default' dieksekusi jika konstanta1 sampai konstantaN tidak ada yang memenuhi *ekspresi*.

Contoh 4:

```

#include <iostream.h>
#include <conio.h>
void main()

```



```

{
int bil;
clrscr();
cout << "Masukkan bilangan : ";
cin >> bil;
    switch (bil)
    {
        case 1 : cout << "Anda memasukkan bil. satu";
                break;
        case 2 : cout << "Anda memasukkan bil. dua";
                break;
        case 3 : cout << "Anda memasukkan bil. tiga";
                break;
        default: cout << "Anda memasukkan bil selain 1, 2, dan 3";
                break;
    }
    getch();
}

```

Note :

Tidak setiap IF bisa dijadikan Switch. Tapi semua Switch dapat dijadikan IF



Contoh 5:

```

#include<conio.h>
#include<iostream.h>
void main()
{
    char kode;
    clrscr();
    cout<<"Masukkan Kode Barang [A..C] : ";
    cin>>kode;
    switch(kode)
    {
        case 'A' :
        case 'a' :
            cout<<"Alat Olah Raga";
            break;
        case 'B' :
        case 'b' :
            cout<<"Alat Elelektronik";
            break;
        case 'C' :
        case 'c' :
            cout<<"Alat Masak";
            break;
        default:
            cout<<"Anda Salah Memasukan kode";
            break;
    }
    getch();
}

```

Latihan Soal.

1. Buatlah program untuk mengetahui bilangan tersebut genap atau ganjil!
2. Buatlah program untuk menentukan banyaknya uang pecahan yang dibutuhkan, urut dari pecahan terbesar!
Input: jumlah uang dalam rupiah
Proses: ratusanribu = jml_uang dibagi 100000
 sisa = jml_uang – (ratusanribu*100000)
 limaplhribu = sisa dibagi 50000
 sisa = sisa – (limaplhribu*50000)
 dan seterusnya.
3. Buatlah program konversi angka ke nilai huruf:
100 >= nilai > 80 A
80 >= nilai > 60 B
60 >= nilai > 40 C
40 >= nilai > 20 D
20 >= nilai > 0 E

6.2 Struktur Kontrol Perulangan

Perulangan digunakan untuk mengerjakan suatu perintah secara berulang-ulang sesuai dengan yang diinginkan.

Struktur pengulangan terdiri atas dua bagian :

1. Kondisi pengulangan yaitu ekspresi boolean yang harus dipenuhi untuk melaksanakan pengulangan
2. Isi atau badan pengulangan yaitu satu atau lebih pernyataan (aksi) yang akan diulang.

Perintah atau notasi dalam struktur pengulangan adalah :

1. Pernyataan **For**
2. Pernyataan **while**
3. Pernyataan **do..while**
4. Pernyataan **continue dan break**
5. Pernyataan **go to**

6.2.1 Statement FOR

Pernyataan for digunakan untuk melakukan looping. Pada umumnya looping yang dilakukan oleh for telah diketahui batas awal, syarat looping dan perubahannya. Selama *kondisi* terpenuhi, maka pernyataan akan terus dieksekusi.

```
for ( inisialisasi; syarat pengulangan; pengubah nilai pencacah )
```

- Inisialisasi merupakan pemberian nilai awal.

- Syarat Pengulangan : memegang kontrol terhadap pengulangan, karena bagian ini yang akan menentukan suatu perulangan diteruskan atau dihentikan.
- Pengubah nilai pencacah merupakan statement control untuk perulangan. Umumnya mengatur kenaikan atau penurunan nilai pencacah. Bila pernyataan didalam for lebih dari satu maka pernyataan-pernyataan tersebut harus diletakan didalam tanda kurung.

```
for ( inisialisasi; syarat pengulangan; pengubah nilai pencacah )
{
    pernyataan / perintah;
    pernyataan / perintah;
    pernyataan / perintah;
}
```

Contoh 1.

```
#include<conio.h>
#include<iostream.h>
void main()
{
    int a;
    clrscr();
    for(a = 0; a <= 10; a++)
        cout<<a<<" ";
    getch();
}
```

Contoh 2.

```
# include <conio.h>
# include <iostream.h>
void main()
{
    clrscr();
    for(int a = 20; a >= 1; a-=2)
        cout<<a<<endl;
    getch();
}
```

Selain berupa angka, pencacah perulangan juga dapat berupa karakter.

Contoh 3.

```
for (huruf = 'Z'; huruf >= 'A'; huruf--)
{
    cout << "Huruf abjad= " << huruf << "\n";
}
```

Nested For

Perulangan bertumpuk secara sederhana dapat diartikan : terdapat satu atau lebih loop di dalam sebuah loop. Banyaknya tingkatan perulangan, tergantung dari kebutuhan. Biasanya, nested loops digunakan untuk membuat aplikasi matematika yang menggunakan baris dan kolom. Loop luar, biasanya digunakan untuk mendefinisikan baris. Sedangkan loop dalam, digunakan untuk mendefinisikan kolom.

```
for(int baris = 1; baris <= 4; baris++)
{
    for (int kolom = 1; kolom <= 5; kolom++)
    {
        cout<<kolom<<" ";
    }
    cout<<endl;
}
```

Penjelasan program:

Perulangan akan menghasilkan nilai sebagai berikut :

baris =1 ; kolom = 1; cetak 1
 kolom = 2; cetak 2
 kolom = 3; cetak 3
 kolom = 4; cetak 4
 kolom = 5 ; cetak 5

ganti baris !

baris =2 ; kolom = 1; cetak 1
 kolom = 2; cetak 2
 kolom = 3; cetak 3
 kolom = 4; cetak 4
 kolom = 5 ; cetak 5

ganti baris !

baris =3 ; kolom = 1; cetak 1
 kolom = 2; cetak 2
 kolom = 3; cetak 3
 kolom = 4; cetak 4
 kolom = 5 ; cetak 5

ganti baris !

```

baris =4 ;   kolom = 1; cetak 1
              kolom = 2; cetak 2
              kolom = 3; cetak 3
              kolom = 4; cetak 4
              kolom = 5 ; cetak 5

```

ganti baris !

selesai.

Dan di layar akan muncul hasil dengan bentuk matrik sebagai berikut:

```

1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5

```

Contoh 5.

```

#include<conio.h>
#include<iostream.h>
void main()
{
    int bil;
    clrscr();
    cout<<"Inputkan Jumlah Bintang = "; cin>>bil;
    for (int i=1; i<=bil; i++)
    {
        for (int j=1; j<=i; j++)
        {
            cout<<"*";
        }
        cout<<endl;
    }
    getch();
}

```

Why?? Jelaskan!

6.2.2 Statement While

Pernyataan **while** merupakan salah satu pernyataan yang berguna untuk memproses suatu pernyataan atau beberapa pernyataan beberapa kali. Pernyataan **while** memungkinkan statemen-statemen yang ada didalamnya tidak dilakukan sama sekali.

```

while (kondisi)
{
    Pernyataan ;
}

```

Karakteristik while() adalah:

1. Dilakukan pengecekan kondisi terlebih dahulu sebelum dilakukan perulangan. Jika kondisi yang dicek bernilai benar (true) maka perulangan akan dilakukan.
2. Blok statement tidak harus ada. Struktur tanpa statement akan tetap dilakukan selama kondisi masih true.

Penting!!!

Jika Anda menggunakan WHILE, pastikan bahwa suatu saat bagian kondisi sampai bernilai FALSE. Apabila tidak, proses perulangan akan terus berjalan selamanya. ♂♀

Contoh 6.

```

#include<iostream.h>
#include<conio.h>
void main()
{
    cout<<"Program Konversi angka ke huruf"<<endl;
    cout<<endl;
    cout<<"Masukkan Angka : ";
    int angka,i;
    char huruf;
    cin>>angka;
    cout<<endl;

    i=1;
    huruf='A';

    while (i<=angka)
    {
        cout<<i<<" --> "<<huruf<<endl;
        i++;
        huruf++;
    }
    getch();
}

```

6.2.3 Statement do ... while

Karakteristik do ... while() adalah:

1. Perulangan akan dilakukan minimal 1x terlebih dahulu, kemudian baru dilakukan pengecekan terhadap kondisi, jika kondisi **benar** maka perulangan masih akan tetap dilakukan.
2. Perulangan dengan do...while() akan dilakukan sampai kondisi **false**.

Bentuk Umumnya :

```
do
{
    pernyataan ;
} while(kondisi);
```

Perbedaan dengan WHILE sebelumnya yaitu bahwa pada DO WHILE statement perulangannya dilakukan terlebih dahulu baru kemudian di cek kondisinya. Sedangkan WHILE kondisi dicek dulu baru kemudian statement perulangannya dijalankan. Akibat dari hal ini adalah dalam DO WHILE minimal terdapat 1x perulangan. Sedangkan WHILE dimungkinkan perulangan tidak pernah terjadi yaitu ketika kondisinya langsung bernilai FALSE.

Contoh 7.

```
#include<iostream.h>
#include<conio.h>
void main()
{
    int angka=0;
    do
    {
        angka++;
        if (angka % 2 ==0)
            cout<<angka<<" ";
    }
    while (angka<30);
    getch();
}
```

6.3 Struktur Kontrol Lompatan

6.3.1 PERNYATAAN *continue* dan *break*

Pernyataan *break* akan selalu terlihat digunakan bila menggunakan pernyataan *switch*. Pernyataan ini juga digunakan dalam loop. Bila pernyataan ini dieksekusi, maka akan mengakhiri loop dan akan menghentikan iterasi pada saat tersebut. Pernyataan *continue* digunakan untuk pergi ke bagian awal dari blok loop untuk memulai iterasi berikutnya. Dengan kata lain, perintah *continue* akan melewati satu iterasi yang sesuai dengan syarat tertentu, dan melanjutkan ke iterasi berikutnya.

Contoh 8.

```
# include <iostream.h>
void main ()
{
    int i;
    for (i=0; i<10; i++)
    {
        if (i==4) continue;
        cout << " Bilangan " << i <<endl;
        if (i==6) break;
    }
}
```

Output :

```
Bilangan 0
Bilangan 1
Bilangan 2
Bilangan 3
Bilangan 5
Bilangan 6
```

Penjelasan :

Dari program diatas, dapat dilihat perulangan dari suatu bilangan sebanyak 10 kali. Tetapi, pada perulangan $i=4$, ada perintah *continue*. Dengan perintah ini, maka program langsung meloncat ke loop berikutnya dan ketika sampai perulangan $i = 6$, ada perintah *break*. Otomatis program akan berhenti dan tidak sampai ke $i=10$. Dan program akan mencetak bilangan 0, bilangan 1, bilangan 2, bilangan 3, bilangan 5, bilangan 6.

6.3.2 PERNYATAAN *goto*

Pernyataan **goto**, diperlukan untuk melakukan suatu lompatan ke suatu pernyataan berlabel yang ditandai dengan tanda “:”

Contoh 9.

```
# include <iostream.h>

void main ()
{
    cout << "Tes go to " << endl;
    goto selesai;

    cout << "Hai, saya kok tidak disapa" << endl;

    selesai :
    cout << "Selesai... " << endl;
}
```

Outputnya :

```
Tes go to
Selesai...
```

Latihan Soal 1.

Buatlah simulasi menu program dengan tampilan di bawah ini menggunakan Do ... WHILE.

MENU PILIHAN

1. Baca Data
2. Ubah Data
3. Hapus Data
4. Exit

Pilihan Anda (1/2/3/4) ? ...

Apabila dipilih menu no 1, maka akan tampil teks “Anda memilih menu 1”. Demikian pula untuk menu 2 dan 3. Kemudian setelah itu muncul teks “Tekan ENTER untuk kembali ke menu utama”. Artinya begitu kita tekan ENTER menu

pilihan akan muncul kembali, dst. Akan tetapi bila yang dipilih menu 4 (EXIT), program langsung berhenti.

Latihan Soal 2.

Buatlah program dengan looping untuk menampilkan hasil seperti berikut:

P	Q	P or Q	P and Q	Not P	P xor Q
=====					
1	1	1	1	0	0
1	0	1	0	0	1
0	1	1	0	1	1
0	0	0	0	1	0
=====					

Latihan Soal 3.

Buatlah program untuk menampilkan deret 'Gossip Girls':

```

Untuk n = 5
X O X O X
X O X O
X O X
X O
X

```

Latihan Soal 4.

Buatlah program untuk menampilkan bilangan fibonacci pada deret ke-n!

Bilangan fibonacci adalah bilangan seperti: 1 1 2 3 5 8 13 ... dst

Jadi jika inputan n = 7, maka hasil adalah 13

Latihan Soal 5.

Buatlah program untuk menampilkan bilangan prima sampai deret ke-n!

Bilang prima adalah bilangan yang hanya habis dibagi 1 dan bilangan itu sendiri.

Jika inputan = 10, maka hasil 2 3 5 7

BAB VII ARRAY dan STRING

7.1 Array

Ilustrasi

- Selama ini kita menggunakan satu variabel untuk menyimpan 1 buah nilai dengan tipe data tertentu. Misalnya :

```
int a1, a2, a3, a4, a5;
```

Deklarasi variabel diatas digunakan untuk menyimpan 5 data integer dimana masingmasing variabel diberi nama a1, a2, a3, a4, dan a5.

- Jika kita memiliki 10 data, 100 data integer bahkan mungkin data yang ingin kita proses tidak kita ketahui atau bersifat dinamis? Kita tidak mungkin menggunakan variabel seperti diatas.
- Di dalam C dan pemrograman yang lain, terdapat suatu fasilitas untuk menyimpan data-data yang bertipe data sama dengan suatu nama tertentu.

Solusi?? Æ Array

Array merupakan kumpulan dari nilai-nilai data yang bertipe sama dalam urutan tertentu yang menggunakan nama yang sama. Dengan menggunakan array, sejumlah variabel dapat memakai nama yang sama. Letak atau posisi dari elemen array ditunjukkan oleh suatu index. Dilihat dari dimensinya array dapat dibagi menjadi Array dimensi satu, array dimensi dua dan array multi-dimensi.

Bentuk Umum pendeklarasian array :

Tipe-Data Nama_Variabel[Ukuran]

Contoh :

```
int nil[5];
```

Nilai suatu variabel array dapat juga diinisialisasi secara langsung pada saat deklarasi, misalnya:

```
int nil[5] = { 1,3,6,12,24 };
```

Maka di penyimpanan ke dalam array dapat digambarkan sebagai berikut:

	0	1	2	3	4
nil	1	3	6	12	24

Mengakses nilai array

Untuk mengakses nilai yang terdapat dalam array, mempergunakan sintak:

```
nama_array[index];
```

Pada contoh di atas, variabel nil memiliki 5 buah elemen yang masing-masing berisi data. Pengaksesan tiap-tiap elemen data adalah:

	nil[0]	nil[1]	nil[2]	nil[3]	nil[4]
nil					

Misal, untuk memberikan nilai 75 pada elemen ke 3, maka pernyataannya adalah:

```
nil[2] = 75;
```

atau jika akan memberikan nilai array kepada sebuah variabel a, dapat ditulis:

```
a = nil[2];
```

Contoh Penerapan:

Misalkan kita memiliki sekumpulan data *ujian* seorang siswa, *ujian* pertama bernilai 90, kemudian 95,78,85. Sekarang kita ingin menyusunnya sebagai suatu data kumpulan *ujian* seorang siswa. Dalam array kita menyusunnya sebagai berikut:

```

ujian[0] = 90;
ujian[1] = 95;
ujian[2] = 78;
ujian[3] = 85;

```

Empat pernyataan diatas memberikan nilai kepada array *ujian*. Tetapi sebelum kita memberikan nilai kepada array, kita harus mendeklarasikannya terlebih dahulu, yaitu :

```
int ujian[4];
```

Perhatikan bahwa nilai 4 yang berada didalam tanda kurung menunjukkan jumlah elemen larik, bukan menunjukkan elemen larik yang ke-4. Jadi elemen larik *ujian* dimulai dari angka 0 sampai 3. Pemrogram juga dapat menginisialisasi larik sekaligus mendeklarasikannya, sebagai contoh :

```
int ujian[4] = {90,95,78,85};
```

Contoh 1.

```

#include <iostream.h>
#include <conio.h>
void main()
{
    //inisialisasi array
    // int ujian[5]= {90,95,78,85};
    int ujian[5];
    //input data ke array
    for (int k=0;k<5;k++)
    {
        cout<<"masukkan data nilai ujian["<<k<<" ] = ";
        cin>>ujian[k];
    }
    //tampil data array
    for (int j=0;j<5;j++)
    {
        cout<<"data nilai ujian["<<j<<" ] = "<<ujian[j]<<endl;
    }
    getch();
}

```

Contoh 2.

```
#include <iostream.h>
#include <conio.h>

void main()
{
    float data[5];
    float rata, total = 0;
    //input data ke array
    for (int k=0;k<5;k++)
    {
        cout<<"masukkan data["<<k<<"] = ";
        cin>>data[k];
    }
    //menghitung total nilai pada array
    for (int j=0;j<5;j++)
    {
        total = total + data[j];
    }
    //menghitung rata - rata
    rata = total / 5;
    cout<<"rata - rata data pada array = "<<rata<<endl;
    getch();
}
```

Contoh 3.

```
#include<iostream.h>
#include<conio.h>
void main()
{
    int data[10] = {4, 1, 0, -9, 8, 5, -1, 2, 3, -7};
    int elemen, ketemu, x;
    cout << "Data yang dicari : ";
    cin >> x;
    ketemu = 0;
    for(elemen=0; elemen<= 9; elemen++)
    {
        if (data[elemen] == x)
        {
            ketemu = !ketemu;
            break;
        }
    }
    if (ketemu == 0)
        cout << "Data tidak ditemukan ";
    else
        cout << "Data ada di elemen : " << elemen;
    getch();
}
```

Modifikasi program diatas sehingga user dapat menentukan/menginputkan sendiri jumlah dan nilai datanya!

Array Dua Dimensi

Struktur array yang dibahas di atas, mempunyai satu dimensi, sehingga variabelnya disebut dengan variabel array berdimensi satu. Pada bagian ini, ditunjukkan array berdimensi lebih dari satu, yang sering disebut dengan array berdimensi dua.

Tipe-Data Nama_Variabel[index-1][index-2]

Sering kali digambarkan/dianalogikan sebagai sebuah matriks. dimana indeks pertama menunjukan baris dan indeks kedua menunjukan kolom.

ILUSTRASI ARRAY 2 DIMENSI

Gambar array berdimensi (baris x kolom = 3 x 4):

	0	1	2	3
0	5	20	1	11
1	4	7	67	-9
2	9	0	45	3

Contoh 4.

```
#include <iostream.h>
#include <conio.h>
void main()
{
    int matrix[3][4] = {{5,10,1,11},{4,7,67,-9},{9,0,45,3}};
    for (int i = 0; i<3; i++)
    {
        for (int j=0;j<4; j++)
        {
            cout<<matrix[i][j]<<" ";
        }
        cout<<endl;
    }
}
```

```

    }
    getch();
}

```

Contoh 5.

```

#include<conio.h>
#include<iostream.h>
void main()
{
    int i,j,kola,kolb,bara,barb;
    int data1[25][25],data2[25][25],hasil[25][25];
    char jawab;
do
{
    do
    {
        clrscr();
        cout<<"Program Penjumlahan Matrix"<<endl;
        cout<<"======"<<endl;
        cout<<endl;
        cout<<"Input Matrix A "<<endl;
        cout<<"Jml baris Matrix A: "; cin>>bara;
        cout<<"Jml kolom Matrix A: "; cin>>kola;
        cout<<endl;
        cout<<"Input Matrix B "<<endl;
        cout<<"Jml baris Matrix B: "; cin>>barb;
        cout<<"Jml kolom Matrix B: "; cin>>kolb;
    }
    while ((kola!=kolb) || (bara!=barb));

    cout<<endl;
    for (i=1; i<=bara; i++)
    {
        for (j=1; j<=kola; j++)
        {
            cout<<"Data A ["<<i<<","<<j<<"]: "; cin>>data1[i][j];
        }
    }
    cout<<endl;
    for (i=1; i<=bara; i++)
    {
        for (j=1; j<=kola; j++)
        {
            cout<<"Data B ["<<i<<","<<j<<"]: "; cin>>data2[i][j];
        }
    }
    for (i=1; i<=bara; i++)
    {
        for (j=1; j<=kola; j++)
        {
            hasil[i][j]=data1[i][j] + data2[i][j];
        }
    }
}

```



```

        cout<<endl;
        cout<<"Hasil Penjumlahan Matrix A + Matrix B: "<<endl;
for (i=1; i<=bara; i++)
{
    for (j=1; j<=kola; j++)
    {
        cout<<hasil[i][j]<<" ";
    }
    cout<<endl;
}
getch();
cout<<endl;
cout<<"Mau Melakukan Perhitungan Lagi [Y/T] = "; cin>>jawab;
}
while ((jawab == 'y') || (jawab == 'Y'));
}

```

Latihan Soal 1.

Modifikasi program penjumlahan array diatas dengan menambahkan fasilitas pengurangan dan perkalian matrix! Perhatikan error handling nya!

Latihan Soal 2.

Buatlah sebuah program untuk menentukan nilai maximum dan minimum dari nilai elemen matrix.

7.2 String

Dalam pemrograman, string merupakan kumpulan dari beberapa karakter-karakter. Untuk membedakan string dengan karakter, dalam C++ dibedakan penulisannya. Suatu nilai merupakan string apabila diapit dengan tanda petik ganda "...", misalnya "SAYA". Sedangkan karakter (char) diapit dengan tanda petik tunggal, misal 's'. Lantas bagaimana dengan "s"?? Dalam hal ini "s" juga merupakan string, meskipun karakter penyusunnya terlihat hanya satu. Akan tetapi pada kenyataannya, "s" disusun tidak hanya karakter 's' saja, melainkan terdapat pula karakter NULL atau '\0', yang berfungsi sebagai tanda akhir dari string.

Simbol karakter ASCII:

0 :	18 :↑	37 :%	56 :8	75 :K	94 :^	113 :q
1 :☺	19 :!!	38 :&	57 :9	76 :L	95 :_	114 :r
2 :☼	20 :¶	39 :'	58 ::	77 :M	96 :`	115 :s
3 :♥	21 :\$	40 :(59 :;	78 :N	97 :a	116 :t
4 :♦	22 :—	41 :)	60 :<	79 :O	98 :b	117 :u
5 :♣	23 :↑	42 :*	61 :=	80 :P	99 :c	118 :v
6 :♠	24 :↑	43 :+	62 :>	81 :Q	100 :d	119 :w
7 :	25 :↓	44 :,	63 :?	82 :R	101 :e	120 :x
8 :	26 :→	45 :-	64 :@	83 :S	102 :f	121 :y
9 :	27 :←	46 :.	65 :A	84 :T	103 :g	122 :z
10 :	28 :L	47 :/	66 :B	85 :U	104 :h	123 :{
	29 :↔	48 :0	67 :C	86 :V	105 :i	124 :
11 :♂	30 :▲	49 :1	68 :D	87 :W	106 :j	125 :}
12 :♀	31 :▼	50 :2	69 :E	88 :X	107 :k	126 :~
13 :	32 :	51 :3	70 :F	89 :Y	108 :l	127 :△
14 :♪	33 :!	52 :4	71 :G	90 :Z	109 :m	
15 :☀	34 :"	53 :5	72 :H	91 :[110 :n	
16 :▶	35 :#	54 :6	73 :I	92 :\	111 :o	
17 :◀	36 :\$	55 :7	74 :J	93 :]	112 :p	

Untuk mendeklarasikan suatu variabel merupakan string, maka perintahnya:

```
char variabel[maks_karakter];
```

contoh:

```
char teks[20];
```

Perintah di atas bermakna bahwa teks merupakan variabel string dengan jumlah karakter yang dapat disimpan maksimal adalah 20 (sudah termasuk karakter NULL). Misalkan suatu variabel string katakanlah kalimat[30] akan diberi nilai "SAYA BELAJAR C++", maka perintahnya:

```
char kalimat[30] = "SAYA BELAJAR C++";
```

Contoh 1.

```
#include <iostream.h>
#include <conio.h>
void main()
{
    int a;
    a = 20;
    char kalimat[30] = "SAYA BELAJAR C++";
    cout << "Nilai a = " << a << endl;
    cout << "Nilai kalimat = " << kalimat << endl;
    getch();
}
```

Selanjutnya bagaimana cara membaca string yang berasal dari keyboard?

```
#include<iostream.h>
#include<conio.h>
void main()
{
char nama[20];
char alamat[30];
cout << "Masukkan nama Anda : ";
cin.getline(nama, sizeof(nama));
cout << "Masukkan alamat Anda : ";
cin.getline(alamat, sizeof(alamat));
cout << "Nama Anda : " << nama << endl;
cout << "Alamat Anda : " << alamat << endl;
getch();
}
```

™ Beberapa Function untuk Operasi String.

Meng-Copy String

`strcpy(kata2, kata1);` // mengcopy isi dari kata1 ke kata2

Mengetahui panjang string dengan strlen()

`strlen(string);`

akan mereturn bilangan bulat yang menyatakan panjang string.

Menggabungkan string dengan strcat()

`strcat(string1, string2)`

menambahkan string2 ke string1.

Mengkonversi ke huruf kapital denganstrupr()

`strupr(string)`

Mengubah huruf kecil dari string ke huruf kapital.

Contoh;

```
char string1[30] = "aBcDefgHIJKLmno";
strupr(string1); //string1 menjadi "ABCDEFGHJKLMNO"
```

Mengkonversi ke huruf kecil dengan strlwr()

`strlwr(string)`

Function ini kebalikan daristrupr().

Mencari Substring dengan strstr()

Misalkan diberikan suatu string “JAKARTA KOTA METROPOLITAN”. Apakah string “METRO” terdapat dalam string tersebut?

Untuk mengetahui hal ini dengan C++, kita dapat menggunakan function **strstr()**.

Sintaks:

```
strstr(string1, string2);
```

Function tersebut akan mereturn nilai 1 jika string2 merupakan substring dari string1, dan akan mereturn 0 jika tidak.

Contoh:

```
if (strstr("JAKARTA KOTA METROPOLITAN", "METRO") == 1)
    cout << "Merupakan substring";
else cout << "Bukan merupakan substring";
```

Membalik string dengan strrev()

Bagaimana cara membalik string “C++” supaya diperoleh “++C”? Berikut ini perintah dalam C++,

sintaks:

```
strrev(string);
```

Contoh:

```
char kata[10] = "C++";
strrev(kata);
cout << kata;
```

Latihan Soal.

Buatlah program yang meminta inputan data karakter dari user yang disimpan ke dalam array 1 dimensi. Kemudian buatlah menu dan program untuk menu seperti berikut:

1. Input karakter
2. Cari karakter
3. Hapus karakter
4. Ubah karakter tertentu
5. Tampilkan karakter-karakter tersebut
6. Statistik karakter (jumlah vokal dan konsonan)
7. Exit

Note :

Gunakan Contoh program berikut sebagai referensi! ☺ ☹ ☺

```
#include <iostream.h>
#include <conio.h>
void main()
{
    char string [20]="contoh program";
    char *hasil;
    hasil=strchr(string,'m');
    *hasil = 'L';
    cout<<string;
    getch();
}
```

BAB VIII

POINTER

Pointer sesungguhnya berisi alamat dari suatu data, bukan data sebagaimana pada variable yang sudah anda kenal. **Pointer** (variabel penunjuk) adalah suatu variabel yang berisi alamat memori dari suatu variabel lain. Alamat ini merupakan lokasi dari obyek lain (biasanya variabel lain) di dalam memori. Contoh, jika sebuah variabel berisi alamat dari variabel lain, variabel pertama dikatakan menunjuk ke variabel kedua.

Operator Pointer ada dua, yaitu :

1. Operator & (***Dereference Operator***)

- Operator & menghasilkan alamat dari operandnya.
- Setiap variabel yang dideklarasikan, disimpan dalam sebuah lokasi memori dan pengguna biasanya tidak mengetahui di alamat mana data tersebut disimpan. Dalam C++, untuk mengetahui alamat tempat penyimpanan data, dapat digunakan tanda ampersand(&) yang dapat diartikan “alamat”. Contoh :

```
Bil1 = &Bil2;
```

dibaca: isi variabel bil1 sama dengan alamat bil2

2. Operator * (***Reference Operator***)

- Operator * menghasilkan nilai yang berada pada sebuah alamat.
- Penggunaan operator ini, berarti mengakses nilai sebuah alamat yang ditunjuk oleh variabel pointer. Contoh :

```
Bil1 = *Bil2;
```

dibaca: bil1 sama dengan nilai yang ditunjuk oleh bil2

Mendefinisikan Variabel Pointer

Suatu variabel pointer didefinisikan dengan bentuk sebagai berikut:

```
tipe_data *nama_variabel;
```

- *tipe_data* dapat berupa sebarang tipe seperti halnya pada pendefinisian variabel non-pointer.
- *nama_variabel* adalah nama variabel pointer.

Contoh :

```
int * pint ;    // pointer ke int
char *pch;    // pointer ke char
char *pch1, *pch2;
```

Guided 1.

```
//contoh program menggunakan pointer
#include<iostream.h>
#include<conio.h>
void main()
{

    int x, y; // x dan y bertipe int
    int *px; // px pointer yang menunjuk objek
    clrscr();

    x = 87;
    px = &x; // px berisi alamat dari x
    y = *px; // y berisi nilai yang ditunjuk px

    cout<<"Alamat x pd Memori = "<<&x<<endl;
    cout<<"Isi px    = "<<px<<endl;
    cout<<"Isi x      = "<<x<<endl;
    cout<<"Nilai yang ditunjuk oleh px = "<<*px<<endl;
    cout<<"Alamat y pd Memori = "<<&y<<endl;
    cout<<"Nilai y    = "<<y<<endl;
    getch();
}
```

Guided 2.

```
//mengubah nilai melalui suatu pointer
#include<conio.h>
#include<iostream.h>
void main()
{
    int vint = 55; //variabel bukan pointer
    int *pint;     //variabel pointer

    clrscr();
    cout<<"vint semula = "<<vint<<endl;
```

```

pint = &vint; //pointer menunjuk ke vint
*pint = 69;    //Nilai yang ditunjuk diubah menjadi 69

cout<<"vint sekarang = "<<vint<<endl;
getch();
}

```

Penjelasan :

Pernyataan `*pint = 69;`

Menyebabkan yang ditunjuk oleh *pint* (yaitu *vint*) berubah menjadi 69

Guided 3.

```

//operasi aritmatika pada pointer
#include<iostream.h>
#include<conio.h>
void main()
{ int nilai[3], *penunjuk;
  clrscr();
  nilai[0] = 125;
  nilai[1] = 345;
  nilai[2] = 750;
  petunjuk = &nilai[0];
  cout<<"Nilai      "<<*petunjuk<<"      ada      di      alamat      memori
  "<<petunjuk<<endl;
  cout<<"Nilai      "<<*(petunjuk+1)<<"      ada      di      alamat      memori
  "<<(petunjuk+1)<<endl;
  cout<<"Nilai      "<<*(petunjuk+2)<<"      ada      di      alamat      memori
  "<<(petunjuk+2)<<endl;
  getch();
}

```

1. Pointer dan Array

Pointer dan array mempunyai hubungan yang dekat. Secara internal array juga menyatakan alamat. Misalnya didefinisikan :

```
char data1[] = {"A", "I", "U", "E", "O"};
```

dan :

```
char *pdata;
```

agar pdata menunjuk ke array, diperlukan pernyataan berupa :

```
pdata = data1;
```


Perhatikan dengan seksama program diatas. Tidak ada tanda & di depan data1. padahal kita ketahui untuk mengakses pointer memerlukan format berikut:

```
Ptr = &variabel;
```

Ini disebabkan array sebenarnya sudah menyatakan alamat. Oleh karena itu tanda & tidak diperlukan. Tanda & digunakan jika variabel tidak berupa array.

```
//guided 1  
//mengakses elemen array via pointer  
#include <conio.h>  
#include <iostream.h>  
  
void main()  
{  
    char data1[] = {'A', 'I', 'U', 'E', 'O'};  
    clrscr();  
    char *pdata;  
    pdata = data1; // pdata menunjuk ke array  
    for (int i=0; i<5; i++)  
    {  
        cout<<*(pdata + i)<<" ";  
    }  
    getch();  
}
```

Seluruh elemen array data1 dapat ditampilkan juga melalui pernyataan :

```
for (int i=0; i<5; i++)  
{  
    cout<<data1[i]<<" ";  
}
```

Memberi nilai pada array

```
//memberi nilai suatu data array
#include<iostream.h>
#include<conio.h>
int main()
{
    int x[5], *p, k;

    clrscr();

    p = x;

    x[0] = 5;           // x[0] diisi dengan 5 sehingga x[0] = 5
    x[1] = x[0];        // x[1] diisi dengan x[0] sehingga x[1] =
    5
    x[2] = *p + 2;      // x[2] diisi dengan x[0] + 2 sehingga
    x[2] = 7
    x[3] = *(p+1)-3;    // x[3] diisi dengan x[1] - 3 sehingga
    x[3] = 2
    x[4] = *(x + 2);    // x[4] diisi dengan x[2] sehingga x[4] =
    7

    cout<<"Array Stelah diisi = "<<endl;
    cout<<endl;
    for(k=0; k<5; k++)
    {
        cout<<"x["<<k<<"] = "<<x[k]<<endl;
    }
    getch();
}
```

2. Pointer dan String

```
//pointer menunjuk ke string
#include<conio.h>
#include<iostream.h>
void main()
{
    clrscr();
    char *ptr = "STIKOM";
    cout<<ptr<<endl;
    getch();
}
```

Pada contoh diatas :

```
char *ptr = "STIKOM";
```

akan menyebabkan C++ :

- Mengalokasikan ptr sebagai variable pointer yang menunjuk ke data bertipe char dan menempatkan konstanta string “STIKOM” ke suatu lokasi di memori komputer
- Kemudian ptr akan menunjuk ke lokasi string “STIKOM”

Pernyataan di atas menyerupai pernyataan :

```
char vptr[] = "STIKOM";
```

Perbedaanya :

- ptr adalah pointer yang dengan mudah dapat diatur agar menunjuk ke data string
- vptr adalah array yang menyatakan alamat yang konstan, tidak dapat dirubah. Yang dapat diubah adalah elemen array-nya.

Perbedaan ini ditunjukkan oleh program berikut :

```
//pointer menunjuk ke string
#include<conio.h>
#include<iostream.h>
void main()
{
    clrscr();
    char *ptr = "STIKOM";
    char vptr[] = "STIKOM";

    cout<<"vptr = "<<vptr<<endl;
    cout<<"ptr = "<<ptr<<endl;

    //tokoh++;      //tidak diperkenankan!!!!!!

    ptr = ptr + 3;      //diperkenankan!!!

    cout<<"ptr sekarang = "<<ptr;
    getch();
}
```

3. Pointer dan Function

```
#include <iostream.h>
#include <conio.h>
void proses(char *);

void main()
{
    char string[] = "characters";
```

```

clrscr();
cout<<"String sebelum proses adalah "<<string<<endl;
proses(string);
cout<<"String setelah proses adalah "<<string<<endl;
getch();
}

```

```

void proses(char *s)
{
while ( *s != ' ' )    //' ' sama denga '\0' yg artinya
null
{
if ( *s >= 'a' && *s <= 'z' )
*s -= 32;
++s;
}
}

```

DAFTAR PUSTAKA

Kadir, Abdul. 2003. *Pemrograman C++*. Yogyakarta : Penerbit ANDI Yogyakarta

Sanjaya, Dwi. 2003. *Asyiknya Belajar Struktur Data di Planet C++*. Jakarta : PT. Elex Media Komputindo

Utami, Erna & Sukrisno. 2005. *10 Langkah Belajar Logika dan Algoritma, Menggunakan Bahasa C dan C++ di GNU/Linux*. Yogyakarta : Penerbit ANDI Yogyakarta

Wahid, Fathul. 2004. *Dasar-dasar Algoritma & Pemrograman*. Yogyakarta : Penerbit ANDI Yogyakarta