

BAB I PENGENALAN KOMPUTER & BAHASA PEMROGRAMAN

DEFINISI KOMPUTER

Komputer berasal dari bahasa latin *computare* yang mengandung arti menghitung. Karena luasnya bidang garapan ilmu komputer, para pakar dan peneliti sedikit berbeda dalam mendefinisikan terminologi *komputer*.

Menurut *Hamacher*,

Komputer adalah mesin penghitung elektronik yang cepat dan dapat menerima informasi input digital, kemudian memprosesnya sesuai dengan program yang tersimpan di memorinya, dan menghasilkan output berupa informasi.

Menurut *Blissmer*,

komputer adalah suatu alat elektronik yang mampu melakukan beberapa tugas sebagai berikut :

- menerima input
- memproses input tadi sesuai dengan programnya
- menyimpan perintah-perintah dan hasil dari pengolahan
- menyediakan output dalam bentuk informasi

Menurut *Fuori*

Komputer adalah suatu pemroses data yang dapat melakukan perhitungan besar secara cepat, termasuk perhitungan aritmatika dan operasi logika, tanpa campur tangan dari manusia.

Untuk mewujudkan konsepsi komputer sebagai pengolah data untuk menghasilkan suatu informasi, maka diperlukan **sistem komputer** (computer system) yang elemennya terdiri dari

- *Hardware, Perangkat Keras*: peralatan yang secara fisik terlihat dan bisa dijamah.
- *Software* atau *Perangkat Lunak*: program yang berisi instruksi/perintah untuk melakukan pengolahan data.
- *Brainware*, manusia yang mengoperasikan dan mengendalikan sistem komputer.

Ketiga elemen sistem komputer tersebut harus saling berhubungan dan membentuk kesatuan. Hardware tidak akan berfungsi apabila tanpa software, demikian juga sebaliknya. Dan keduanya tiada bermanfaat apabila tidak ada manusia (brainware) yang mengoperasikan dan mengendalikannya.

PENGGOLONGAN KOMPUTER

Dibedakan atas beberapa jenis yaitu :

- **Berdasarkan Data Yang Diolah**
 - a. Komputer Analog
 - b. Komputer Digital
 - c. Komputer Hybrid
- **Berdasarkan Penggunaannya**
 - a. Komputer Untuk Tujuan Khusus (Special Purpose Computer)
 - b. Komputer Untuk Tujuan Umum (General Purpose Computer)
- **Berdasarkan Kapasitas dan Ukurannya**
 - a. Komputer Mikro (Micro Computer)
 - b. Komputer Mini (Mini Computer)
 - c. Komputer Kecil (Small Computer)
 - d. Komputer Menengah (Medium Computer)
 - e. Komputer Besar (Large Computer)
 - f. Komputer Super (Super Computer)
- **Berdasarkan Generasinya**

- a. **Komputer Generasi Pertama (1946-1959)**
- b. **Komputer Generasi Kedua (1959-1964)**
- c. **Komputer Generasi Ketiga (1964-1970)**
- d. **Komputer Generasi Keempat (1979-sekarang)**
- e. **Komputer Generasi Kelima**

BAHASA PEMROGRAMAN

Bahasa (language)

Adalah suatu sistim untuk berkomunikasi. Bahasa tertulis menggunakan simbol (yaitu huruf) untuk membentuk kata. Dalam ilmu komputer, bahasa manusia disebut bahasa alamiah (natural languages), dimana komputer tidak bisa memahaminya, sehingga diperlukan suatu bahasa komputer.

Bahasa pemrograman (programming language)

Program merupakan sekumpulan instruksi yang merupakan penyelesaian masalah. Program 'dimasukkan' kedalam komputer, komputer mengerjakan instruksi-instruksi di dalam program tersebut, lalu memberikan hasil atau keluaran yang diinginkan.

Agar program dapat dilaksanakan oleh komputer, program tersebut harus ditulis dalam suatu bahasa yang dimengerti oleh komputer. Karena komputer adalah mesin maka program harus ditulis dalam bahasa yang khusus dibuat untuk berkomunikasi dengan komputer. Bahasa komputer yang digunakan dalam menulis program dinamakan bahasa pemrograman.

Jadi bahasa pemrograman yaitu kumpulan perintah-perintah bermakna, berstruktur tertentu (syntax) yang dapat dimengerti komputer yang berguna didalam penyelesaian masalah.

Dalam pengertian luas pemrograman meliputi seluruh kegiatan yang tercakup dalam :

- Pembuatan program, termasuk analisis kebutuhan (requirement's analysis)
- Keseluruhan tahapan dalam perencanaan (planning) , perancangan (design) dan pewujudannya (implementation).

Dalam pengertian yang lebih sempit, pemrograman merupakan :

- Pengkodean (coding atau program writing = penulisan program)
- Pengujiannya (testing) berdasarkan rancangan tertentu.

Pemahaman yang lebih sempit ini sering digunakan dalam pembuatan program-program terapan komersial yang membedakan antara system analyst yang bertanggung jawab dalam menganalisa kebutuhan, perencanaan dan perancangan program dengan pemrogram (programmer) yang bertugas membuat kode program dan menguji kebenaran program.

GENERASI BAHASA PEMROGRAMAN

Generasi Pertama : BAHASA MESIN

Bahasa mesin adalah bahasa internal komputer yang mengeksekusi secara langsung tanpa terjemahan (translation).

Disebut generasi pertama karena merupakan jenis yang paling awal dikembangkan: tahun 1940-an dan awal 1950-an semua program harus dikodekan dalam bahasa mesin

Pemrograman dalam bahasa mesin :

- Akan menyita waktu dan kondusif untuk membuat kesalahan
- Berbeda untuk setiap jenis komputer, sehingga bergantung pada komputer dan tidak standar

Semua program harus ada dalam bahasa mesin agar dapat dieksekusi, sehingga bahasa lain yang ditulis programer perlu diterjemahkan oleh komputer ke bahasa mesin untuk eksekusi.

Generasi Kedua : BAHASA ASSEMBLY

Penggunaan komputer secara komersial tahun 1950-an mengakibatkan dikembangkannya bahasa assembly

Ciri-Ciri bahasa assembly :

- Kode ditandai dengan nama yang mudah diingat seperti ADD, SUB, dan MULT
- Alamat penyimpanan (*storage addresses*) nyata di mana data ditempatkan dapat didefinisikan dengan nama-nama seperti AMT1 dan AMT2 untuk memudahkan rujukan

Bahasa assembly sangat menyerupai bahasa mesin, sehingga untuk menjadi programmer bahasa assembly yang cakap kita harus memahami arsitektur mesin, yakni bagaimana mesin itu secara fisik memproses data.

Sama seperti bahasa mesin, bahasa assembly tergantung komputer (tidak portable). Untuk menerjemahkan kode-kode diperlukan program khusus yang disebut **ASSEMBLER**. Bahasa assembly masih digunakan karena begitu mirip dengan bahasa mesin dengan kode yang sangat efisien Untuk membuat system software lebih disukai menggunakan bahasa assembly karena sangat efisien dalam penggunaan komputer (butuh memori yang kecil)

Generasi Ketiga : Bahasa Tingkat Tinggi

Penggunaan komputer dalam bisnis berkembang sangat dramatis pada tahun 1950-an. Bahasa mesin dan assembly terlalu sulit, sehingga muncul third-generation languages (3GLs) yang lebih mudah untuk program dan portable.

Disebut tingkat tinggi karena mudah dipelajari & program tingkat-tinggi memerlukan proses penerjemahan oleh komputer yang sangat rumit yang disebut **COMPILER** atau **INTERPRETER**

Seperti generasi pendahulunya 1GL dan 2GL, 3GL disebut **bahasa prosedural** (4GL dan 5GL disebut **bahasa nonprosedural**), yakni program harus menentukan kumpulan instruksi yang tepat yang dibutuhkan untuk menyelesaikan tugas yang diberikan

Contoh bahasa tingkat-tinggi :

- FORTRAN (FORmula TRANslator)
- Cobol
- Pascal
- BASIC
- MODULA-2
- ADA
- Object-oriented programming language

Bahasa C disebut bahasa “tingkat-menengah” karena format instruksinya dengan bahasa tingkat-tinggi sekaligus bisa berinteraksi langsung dengan hardware

Generasi Keempat

Ciri-ciri :

- Mudah untuk dipelajari dan dipahami
- Tepat untuk pengaksesan database
- Memfokuskan pada memaksimalkan produktivitas manusia dari pada minimisasi waktu computer
- Nonprosedural
- Tersedia dalam software paket yang dapat digunakan untuk mengembangkan aplikasi yang diinginkan

Contoh:

- Query language seperti SQL (structured query language), QBE (query-by-example) dan INTELLECT
- Report generator

Generasi Kelima

Sering digunakan untuk akses database atau membuat sistem pakar (*expert system*) atau *knowledge-based system*

Dalam konsep, ditujukan untuk bahasa alami (*natural languages*) yang semirip mungkin dengan hubungan kemanusiaan

Contoh : LISP dan Prolog

Sekarang ini banyak sistem pakar dikodekan baik dalam LISP maupun Prolog, meski untuk hal yang sama bisa ditulis dalam C atau C++. Usaha yang sekarang dilakukan adalah memperbaiki bahasa AI (*artificial intellegence*) dengan mengkombinasikan kemampuan terbaik dari LISP dan Prolog.

PENERJEMAH BAHASA PEMROGRAMAN

Untuk menterjemahkan bahasa pemrograman yang kita tulis maka diperlukan Compiler & Interpreter.

Compiler adalah suatu program yang menterjemahkan bahasa program (source code) ke dalam bahasa objek (object code) secara keseluruhan program.

Interpreter berbeda dengan compiler, Interpreter menganalisis dan mengeksekusi setiap baris dari program tanpa melihat program secara keseluruhan. Keuntungan dari Interpreter adalah dalam eksekusi yang bisa dilakukan dengan segera. Tanpa melalui tahap kompilasi, untuk alasan ini interpreter digunakan pada saat pembuatan program berskala besar.

Compiler memerlukan waktu untuk membuat suatu program yang dapat dieksekusi oleh komputer. Tetapi, program yang diproduksi oleh Compiler bisa berjalan lebih cepat dibandingkan dengan yang diproduksi oleh Interpreter, dan bersifat independen.

PERBEDAAN COMPILER & INTERPRETER

Compiler	Interpreter
Menterjemahkan secara keseluruhan	Menterjemahkan intruksi per intruksi
Bila terjadi kesalahan kompilasi maka source program harus diperbaiki dan di kompilasi ulang	Bila terjadi kesalahan interpretasi dapat langsung diperbaiki
Dihasilkan object program	Tidak dihasilkan object program
Dihasilkan executable program	Tidak dihasilkan executable program
Proses pengerjaan program lebih cepat	Proses pengerjaan lebih lambat
Source program tidak di pergunakan hanya bila untuk perbaikan saja	Seource program terus dipergunakan
Keamanan dari program lebih terjamin	Keamanan dari program kurang terjamin

BAB II ALGORITMA

Kata *Algoritma* diambil dari nama ilmuwan muslim **Abu Ja'far Muhammad bin Musa Al-Khwarizmi** (780-846 M) yang banyak menghasilkan karya dalam bidang matematika, disamping karya-karyanya dalam bidang lainnya seperti geografi dan musik.

Algoritma adalah urutan langkah-langkah logis penyelesaian masalah yang disusun secara sistematis. Langkah-langkah tersebut harus logis ini berarti nilai kebenarannya harus dapat di tentukan, benar atau salah. Langkah-langkah yang tidak benar dapat memberikan hasil yang salah.

Algoritma merupakan jantung ilmu komputer atau informatika tetapi algoritma tidak selalu identik dengan ilmu komputer saja. Dalam kehidupan sehari-hari banyak terdapat proses yang digambarkan dalam suatu *algoritma*. Contohnya *resep masakan/membuat kue (resep kue), membuat pakaian (pola pakaian), merakit mobil (panduan merakit)*.

CIRI-CIRI ALGORITMA YANG BAIK.

- a. Tepat sasaran : memenuhi spesifikasi pekerjaan dan bekerja sesuai tujuan
- b. Flexible dan portable :
 - Flexible untuk dikembangkan lebih lanjut
 - Portable untuk digunakan pada berbagai sistem dan mesin
- c. Bersih dari kesalahan sistem ataupun logik
- d. Efektif : setiap langkah harus sederhana sehingga dapat dikerjakan dalam sejumlah waktu yang masuk akal.
- e. Murah:
 - Efisien dalam penggunaan piranti memori dan penyimpanan lainnya.
 - Cepat waktu pelaksanaannya.
- f. Didokumentasi dengan baik untuk pengoperasian, pemeliharaan dan pengembangan.
- g. *Algoritma* merupakan pemberian (description) pelaksanaan suatu proses.
- h. Tidak ambiguous : tidak bermakna ganda.
- i. Harus berhenti setelah mengerjakan sejumlah langkah terbatas.

ATURAN PENULISAN TEKS ALGORITMA

Ada dua cara penulisan *Algoritma*, yaitu :

- o Uraian deskriptif
- o Menggunakan bagan-bagan/symbol-simbol tertentu seperti *Diagram Alir* (Flowchart)
- o Menggunakan kata-kata atau kalimat yang mirip dengan bahasa pemrograman yaitu *Pseudo Code*

DIAGRAM ALIR

Flow-chart atau bagan alir adalah suatu skema/gambar yang memperlihatkan urutan intruksi/kegiatan dan hubungan antar proses beserta instruksinya.

Gambaran ini dinyatakan dengan symbol. Dengan demikian setiap symbol menggambarkan proses tertentu. Sedangkan antara proses digambarkan dengan garis penghubung.

Flow-chart dibedakan atas dua jenis, yaitu :

- o Diagram alir sistem (*System Flow-chart*)
Bagan yang memperlihatkan urutan prosedur dan proses dari beberapa file dalam media tertentu.

System flowchart menggambarkan :


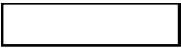
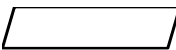






1. Hubungan antar suatu file dengan file lainnya
2. Media yang dipakai untuk setiap file

- o Diagram alir program (*Program Flow-chart*)
Bagan yang memperlihatkan urutan dan hubungan proses dalam suatu program.

Penulisan dengan menggunakan bagan alir sudah tidak banyak digunakan, dengan alasan :

- ❑ Hanya cocok untuk masalah kecil
- ❑ Memerlukan kemampuan menggambar yang baik, tetapi sangat bermanfaat jika digunakan untuk menggambarkan logika pemecahan masalah untuk pengajaran

Berikut merupakan simbol dari diagram alir program

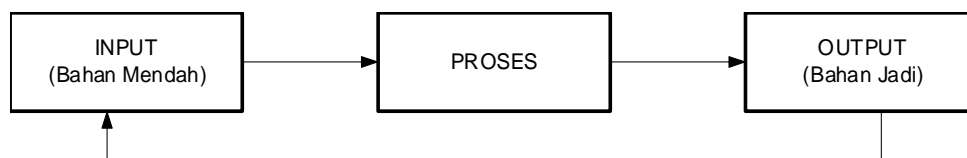
SIMBOL	ARTI
	<i>Terminal</i> yang menyatakan awal dan akhir
	<i>Process</i> yang melambangkan suatu pengolahan data
	Input / Output yang mewakili data input dan menuliskan outputnya
	<i>Inialisasi Awal</i> atau Loop (<i>For...Next</i>) yaitu untuk menginialisasi /memesan suatu variable atau menggunakan suatu perulangan
	<i>On-page Connector</i> , yaitu penghubung di satu halaman yang sama
	<i>Off-page Connector</i> , yaitu penghubung di halaman yang berbeda
	<i>Flow Lines</i> , menunjukkan arah arus/pekerjaan
	<i>Predefined</i> , yaitu sebuah program yang terpisah yang dapat dipanggil dari main program
	<i>Decision</i> , yaitu suatu perbandingan antara dua atau lebih nilai

KAJIDAH-KAJIDAH UMUM PEMBUATAN FLOWCHART

Dalam pembuatan flowchart tidak ada rumus atau patokan yang bersifat mutlak. Karena flowchart merupakan gambaran hasil pemikiran dalam menganalisa suatu masalah dengan komputer. Sehingga flowchart yang dihasilkan dapat bervariasi antara satu pemrogram dengan yang lainnya.

Namun secara garis besar setiap pengolahan selalu terdiri dari 3 bagian utama, yaitu:

- Input
- Proses Pengolahan dan
- Output



Untuk pengolahan data dengan komputer, urutan dasar pemecahan suatu masalah :

- START, berisi instruksi untuk persiapan peralatan yang diperlukan sebelum menangani pemecahan persoalan
- READ, berisi instruksi kegiatan untuk membaca data dari suatu peralatan input
- PROSES, berisi kegiatan yang berkaitan dengan pemecahan persoalan sesuai dengan data yang dibaca

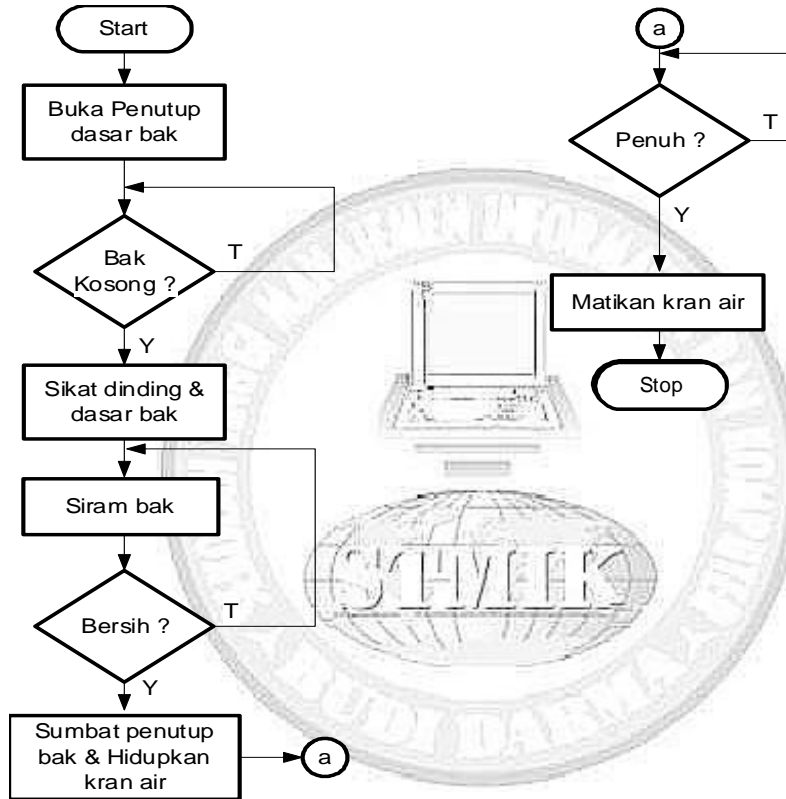
- WRITE, berisi instruksi untuk merekan hasil kegiatan ke peralatan output
- END, mengakhiri kegiatan pengolahan

Walaupun tidak ada kaidah-kaidah yang baku dalam penyusunan flowchart, namun ada beberapa anjuran:

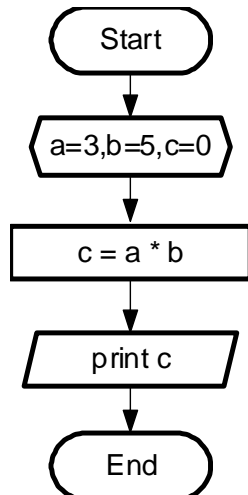
- Hindari pengulangan proses yang tidak perlu dan logika yang berbelit sehingga jalannya proses menjadi singkat
- Jalannya proses digambarkan dari atas ke bawah dan diberikan tanda panah untuk memperjelas
- Sebuah flowchart diawali dari satu titik **START** dan diakhiri dengan **END**

Contoh pemakaian flowchart

- o Suatu algoritma untuk membersihkan bak mandi



- o Untuk mencari hasil dari pertambahan dengan rumus $c = a * b$



PSEUDOCODE

Pseudocode banyak digunakan untuk mewakili urutan-urutan proses dari program. Pseudo berarti imitasi dan code dihubungkan dengan instruksi yang ditulis dalam bahasa computer.

Tujuan digunakan pseudocode yaitu untuk menjembatani jurang antara bahasa sehari-hari programmer dengan bahasa computer. Pseudocode disebut juga dengan bahasa inggris terstruktur. Karena merupakan kombinasi elemen-elemen dasar dari pemrograman terstruktur dengan menggunakan bahasa inggris.

Pseudocode dapat juga ditulis dengan bahasa Indonesia. Pseudocode merupakan suatu bahasa yang memungkinkan programmer untuk berpikir terhadap permasalahan yang harus dipecahkan tanpa harus memikirkan syntax dari bahasa pemrograman tertentu.

Jadi pseudocode digunakan untuk menggambarkan logika urutan-urutan dari program tanpa memandang bagaimana bahasa pemrogramannya.

Contoh suatu pseudocode :

```
Mulai
Nilai a=3 ; b=5 ; c=0
Hitung c = a * b
Tampilkan nilai c
End
```

Contoh Latihan

1. Buatlah algoritma sewaktu akan menelpon di telepon umum.

Jawab.

1. a. Angkat gagang telepon
b. Masukkan koin
c. Tekan nomor yang akan dihubungi
d. Bicara
e. letakkan gagang telepon.

Langkah tersebut diatas apabila dalam kondisi normal, tetapi dalam keadaan tidak normal maka, langkah langkah tersebut tidak akan memecahkan masalah. Sehingga algoritma diatas dapat dikembangkan lebih lanjut, seperti berikut ini :

1. Angkat gagang telepon
2. a. Jika terdengannya nada panggil, masukkan koin
b. Jika tidak terdengannya nada panggil, ke langkah v
3. Tekan nomor yang akan dihubungi
4. a. Jika telepon yang dihubungi ada yang mengangkat, bicara
b. Jika telepon yang dihubungi tidak ada yang mengangkat, ke langkah v
5. Letakkan gagang telepon

BAB III IDENTIFIER, TIPE DATA, OPERATOR

NAMA PENGENAL (IDENTIFIER)

Didalam algorithma, nama/pengenal dipakai sebagai pengidentifikasi 'sesuatu' dan program mengacu/memproses 'sesuatu' itu melalui namanya. Kata lain dari pengenal yaitu identifier yaitu suatu pengenal yang didefinisikan pemakai berguna untuk menampung data.

Didalam Algorithma 'sesuatu' yang diberi nama dapat berupa:

- Peubah (Variabel)
Adalah tempat penyimpanan data/ informasi yang isinya dapat diubah:
- Tetap (constant)
Tempat penyimpanan didalam memori yang isinya tetap selama pelaksanaan program dan tidak dapat diubah.
- Tipe Bentuk
Nama tipe bentuk di tentukan oleh pemrogram.
- Nama Fungsi
- Nama Procedure

Data yang disimpan dalam sebuah identifier bisa berubah-ubah. Hal ini terjadi karena adanya input, atau adanya proses dan penyimpanan hasil proses terhadap identifier itu sendiri. Ketentuan dalam mendefinisikan identifier :

- Gabungan huruf dan angka dengan karakter pertama harus berupa huruf.
- Tidak boleh ada spasi
- Tidak boleh ada simbol-simbol khusus, kecuali garis bawah.
- Panjang bebas.
- Mudah diingat dan dimengerti.

Contoh Identifier : Nama, Nm_3, Gaji dll.

Nilai/data yang tersimpan dalam suatu identifier bermacam-macam jenis atau tipenya, misalnya nilai numerik, karakter, string dan rekaman (record).

Suatu tipe menyatakan pola penyajian data dalam komputer. Tipe data dapat dikelompokkan menjadi atas dua macam yaitu :

- Tipe Data Dasar
Merupakan tipe data yang dapat langsung dipakai
- Tipe Terstruktur / bentuk.
Merupakan tipe yang dibentuk dari tipe dasar atau dari tipe bentuk lain yang sudah didefenisikan

TIPE DATA DASAR

Tipe Data dasar terdiri atas :

- ❖ Bilangan Bulat
Merupakan bilangan yang tidak mengandung bilangan pecahan desimal. Tipe untuk bilangan ini adalah Integer.
Terbagi atas beberapa kategori seperti terlihat dalam tabel yang menunjukkan jenis data, ukuran dalam memori dan rentang nilainya.

Tipe Data	Ukuran Tempat	Jangkauan
Byte	1 byte	0 s/d +255
Shortint	1 byte	-28 s/d +127
Integer	2 bytes	-32768 s/d 32767
Word	2 bytes	0 s/d 65535
Longint	4 bytes	2147483648 s/d 2147483647

Contoh bilangan integer adalah: 34 6458 -90 0 1112

Penggolongan tipe data integer tersebut dimaksudkan untuk membatasi alokasi memori yang dibutuhkan misalkan untuk suatu perhitungan dari suatu variabel bilangan diperkirakan nilai maksimumnya 32767 kita cukup mendeklarasikan variabel bilangan sebagai integer (2 byte), daripada sebagai longint (4 byte).

Tipe bilangan bulat adalah tipe yang memiliki keterurutan. Ini berarti, jika sebuah nilai bilangan bulat diketahui, nilai sebelumnya (*predecessor*) dan nilai sesudahnya (*successor*) dapat ditentukan.

Contoh predecessor dari 8 adalah 7 sedangkan successornya adalah 9.

❖ **Bilangan Real**

Merupakan bilangan yang mengandung pecahan desimal. Harus mengandung titik. Tipe bilangan riil ini adalah real. Dalam Turbo Pascal tipe real di representasikan ke dalam 4 macam tipe yaitu real, single, double dan extended. Rentang nilai positif untuk keempat macam tipe tersebut adalah :

Tipe Data	Ukuran Tempat	Jangkauan
Real	6 bytes	2.9 x 10 ⁻³⁹ s/d 1.7 x10 ³⁸
Single	4 bytes	1.5 x 10 ⁴⁵ s/d 3.4 x 10 ³⁸
Double	8 bytes	5.0 x 10 ⁻³²⁴ s/d 1.7 x 10 ³⁰⁸
Extended	10 bytes	3.4 x 10 ⁻⁴⁹³² s/d 1.1 x 10 ⁴⁹³²
Comp	8 bytes	-9.2x 10 ¹⁸ s/d 9.2x 10 ¹⁸

❖ **Karakter**

Tipe data ini menyimpan karakter yang diketikkan dari keyboard, memiliki 255 macam yang terdapat dalam tabel ASCII (American Standard Code for Information Interchange).

Untuk tipe data karakter penulisannya harus di apit oleh tanda petik tunggal.

❖ **Bilangan Logika (Boolean)**

Tipe data logika hanya mengenal dua buah nilai yaitu : **benar** (true) atau **salah** (false). Tipe boolean memakai memori paling kecil, sedangkan WordBool dan LongBool dipakai untuk menulis program yang sesuai dengan lingkungan Windows.

Tipe Variabel	Ukuran Tempat
Boolean	1 byte
WordBool	2 byte
Longbool	3 byte

Operasi-operasi yang dapat dilakukan untuk terhadap bilangan boolean dikenal dengan operasi logika operator yang dapat digunakan untuk operasi logika adalah NOT, AND, OR dan XOR.

Tabel Kebenaran

A	b	a AND b	a OR b	a XOR b
TRUE	TRUE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE	TRUE
FALSE	TRUE	FALSE	TRUE	TRUE
FALSE	FALSE	FALSE	FALSE	FALSE

TIPE DATA TERSTRUKTUR / BENTUKAN

Adalah tipe yang didefinisikan sendiri oleh pemrogram. Tipe bentuk disusun oleh satu atau lebih tipe dasar. Ada beberapa macam tipe bentuk :

- ❖ String
Merupakan suatu data yang menyimpan array (larik), sebagai contoh 'ABCDEF' merupakan sebuah konstanta string yang berisikan 6 byte karakter. Ukuran Tempat untuk tipe data ini adalah 2 s/d 256 byte, dengan jumlah elemen 1 s/d 255.

Pada Pascal string dideklarasikan dengan string [konstanta] atau string. Bila ukuran string tidak didefinisikan maka akan banyak memakan ruang, karena ukuran string menyesuaikan dengan defaultnya.

- ❖ Tipe Dasar yang diberi Nama Tipe Baru
Dalam pascal kita dapat memberi nama baru untuk tipe dasar dengan kata kunci TYPE.

Contoh :

Type Bil_INT : Integer ;

Bil_INT adalah tipe bilangan bulat yang sama dengan tipe integer. Apabila kita mempunyai variabel yang bernama P dan bertipe bilbulat, variabel P tersebut sama saja bertipe integer.

- ❖ Set
Sebuah set merupakan suatu himpunan yang berisi nilai (anggota). set merupakan Tipe data yang khusus untuk Pascal. Set dalam pemrograman sangat mirip dengan himpunan dalam ilmu matematika.

Contoh Day = (Sun, Mon, Tue, Wed, Thu, Fri, Sat)

- ❖ Pointer
Pointer merupakan variabel khusus yang berisi suatu address (alamat) di lokasi lain didalam memory. Suatu variabel yang points(menunjuk) ke sesuatu sehingga disebut pointer.

Ada dua macam pointer :

- Typed (tertentu) → merupakan pointer yang menunjuk pada tipe data tertentu pada variable.
- Generic(umum) → merupakan pointer yang tidak menunjuk pada tipe data tertentu pada variable.

Selain tipe data terstruktur diatas ada juga tipe data : Array, Rekaman, File yang akan dijelaskan pada bab-bab selanjutnya.

Tipe-tipe data diatas bisa dioperasikan dengan sejumlah operasi yaitu :

o **Operasi Arithmatika**

Operasi arithmatika yang berlaku pada bilangan bulat adalah

Operator	Arti	Priorotas
^, **	Pangkat	1
*	Kali	2
/	Bagi (bilangan Real)	2
\	Bagi (bilangn Bulat)	3
mod	Sisa hasil bagi	4
+	Tambah	5
-	Kurang	6

Jika dalam sebuah ungkapan terdapat lebih dari satu operator, perhitungan ditentukan berdasarkan prioritas.

o **Operasi Perbandingan**

Operasi perbandingan untuk bilangan bulat dengan salah satu operator relasional menghasilkan nilai boolean (true atau false). Operator perbandingan untuk bilangan bulat adalah :

Operator	Arti
=	Sama dengan
<>	Tidak sama dengan
>	Lebih besar dari
<	Lebih kecil dari
<=	Lebih kecil atau sama dengan
>=	Lebih besar atau sama dengan

o **Operasi STRING**

Digunakan untuk menggabungkan dua buah nilai data yang bertipe string. Hanya ada sebuah operator string, yaitu operator +.

PENGISIAN NILAI KE IDENTIFIER

Pengisian Nilai ke dalam variabel/konstanta bisa melalui :

- Pengisian secara langsung

Adalah memasukkan sebuah nilai ke dalam nama peubah langsung didalam teks *algorithma*. Nilai yang diberikan kepada nama peubah dapat berupa :

- Tetapan
- Nilai nama peubah lain
- Nilai sebuah ekspresi.

Nilai tersebut harus bertipe sama dengan tipe peubah. Akibat pengisian nilai kedalam nama peubah, nilai lain yang disimpan didalamnya hilang ditimpa dengan nama baru.

Nilai yang dikandung oleh nama peubah adalah nilai yang terakhir kali diisikan ke dalamnya.

Contoh :

M ← 5	M ←- 5	atau	K = 5	M = 5
Luas ← P * L	Luas ←- P * L	atau	Luas = P * L	Luas = P * L
Ketemu ← false		atau	Ketemu = false	
NH ← 'A'	NH ←- 'A'	atau	NH = 'A'	NH = 'A'
Nama ← "STMIK"		atau	Nama = "STMIK"	Nama = "STMIK"
B ← Luas	B ←-Luas	atau	B = Luas	B = Luas

- Dibaca dari piranti masukan

Mengisi nilai dari piranti masukan dinamakan operasi pembacaan data, misalnya dari keyboard

Didalam *algorithma*, instruksi pembacaan nilai untuk peubah dilakukan dengan notasi read

Contoh:

Read (P) Read P

Read (Nama) Read Nama

Read (nama1, nama2, ..., namaN)

Dengan syarat nama1, nama2,...,namaN adalah nama peubah yang sudah didefinisikan didalam deklarasi.

MENAMPILKAN NILAI KE PIRANTI KELUARAN

Nilai yang disimpan memori dapat ditampilkan ke piranti keluaran. Instuksi penulisan nilai dilakukan notasi write

Contoh:

Write (Nama) Write Nama

Write (Luas) Write Luas

Write (nama1, nama2,...,namaN)

Write (tetapan)

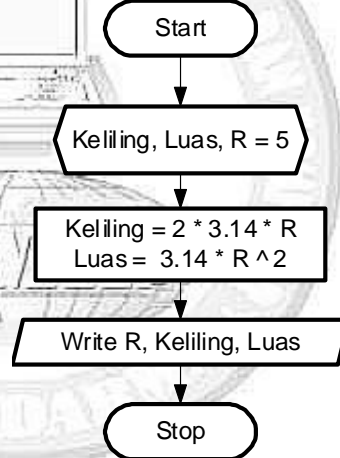
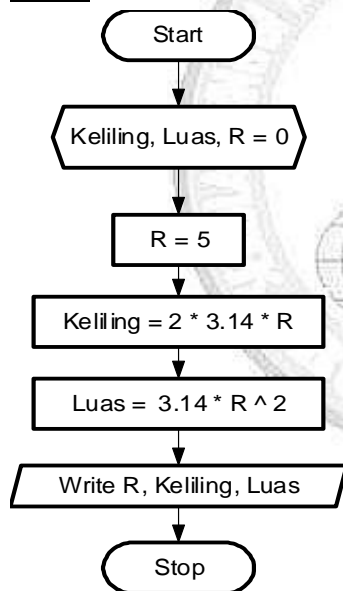
Write (nama, tetapan)

Dengan keterangan bahwa nama1, nama2, ...namaN dapat berupa nama peubah atau nama tetapan.

LATIHAN

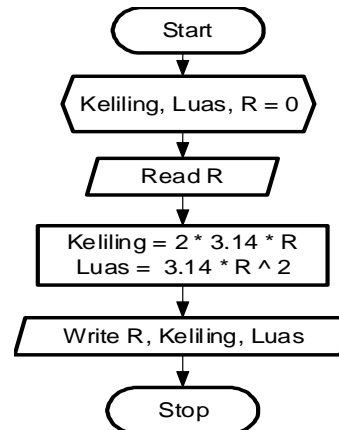
1. Buat diagram alir yang digunakan untuk menghitung dan mencetak Luas Lingkaran (rumus : $LUAS = 3.14 * r^2$) & Keliling lingkaran (rumus : $KELILING = 2 * 3.14 * r$) Dimana diketahui r bernilai 5

Jawab



Pada Flowchart diatas R telah diketahui bernilai 5 bagaimana jika R tidak diketahui (diinput dari keyboard)

Jawab



2. Buat diagram alir untuk proses dibawah ini :
- Merubah suhu dari Fahrenheit ke derajat celcius dan raemur (Celcius = $5/9 (f-32)$
Reamur = $4/9(f-32)$) f= Fahrenheit.
 - Menghitung Volume Kubus ($VolKubus = S * S * S$) S = Sisi
 - Menghitung Total Harga Photo Copy ($TotHarga = Jumlah\ lbr * Harga\ per\ lbr$)
 - Menghitung Penjualan Barang dgn Bentuk

INPUT PENJUALAN

Nama Barang : x(20)
Harga Rp. : 9(7)
Jumlah Jual : 99
Total Harga : 9(9)

Prosesnya $TotalHarga = Harga * Jumlah\ Jual.$

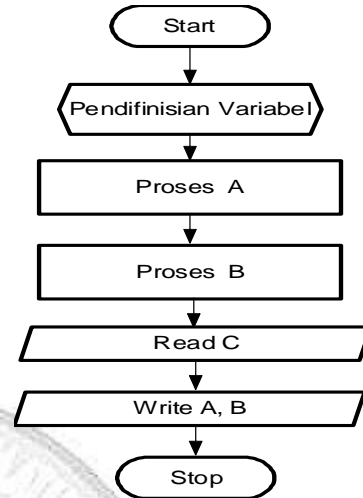


BAB IV STRUKTUR KENDALI

Di dalam tehnik pemrograman kita mengenal adanya '**structure programming**'. Pada dasarnya struktur programming disusun berdasarkan statement yang terstruktur pula.

Terdapat 3 proses eksekusi utama di dalam structure programming, yaitu :

- Proses secara 'Sequential' (berurutan)
- Proses secara 'Selection/Decision' (menggunakan statement kendali)
- Proses secara 'Repetition'



PROSES SEQUENTIAL (Berurutan)

Proses secara sequential terjadi bila proses berjalan dari atas ke bawah dan dikerjakan dari baris-perbaris. Proses sequential dapat digambarkan seperti dibawah ini. Proses A dikerjakan dahulu, setelah selesai dilanjutkan ke proses B. Dengan kata lain proses dikerjakan urut kebawah

PROSES DECISION (Kendali)

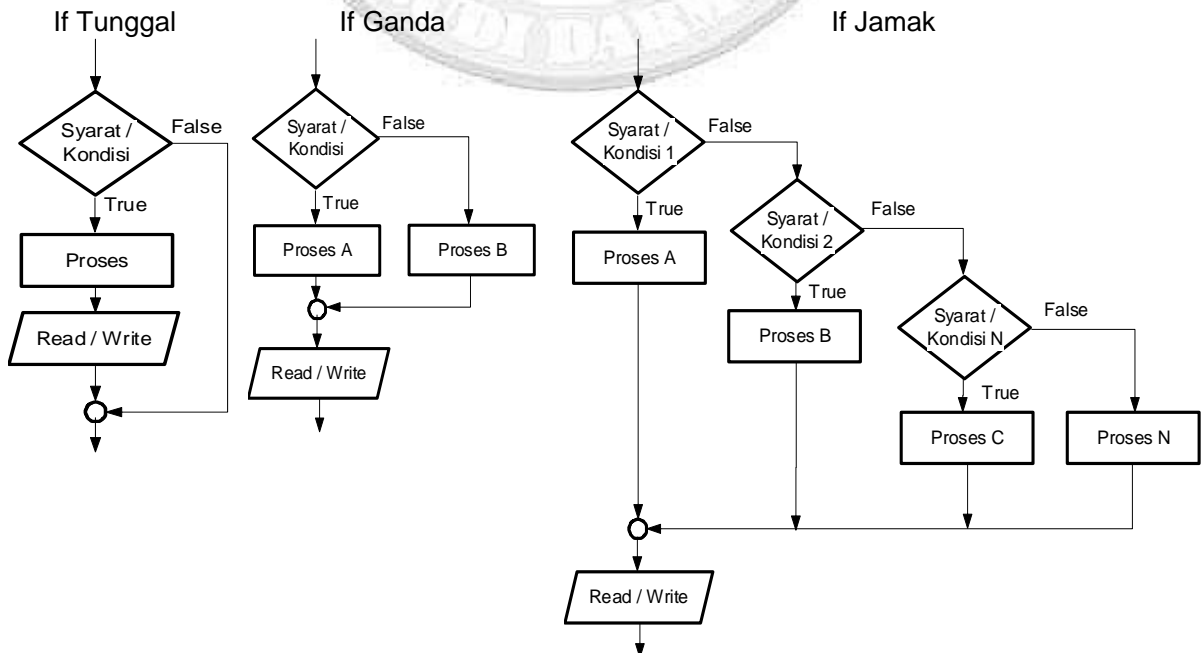
Statement kendali digunakan untuk proses pengambilan keputusan. Dimana proses akan dikerjakan bila kondisi yang disyaratkan sesuai (bernilai true/benar). Terdapat dua statemen kendali yaitu :

- IF
- CASE

❖ STATEMEN IF

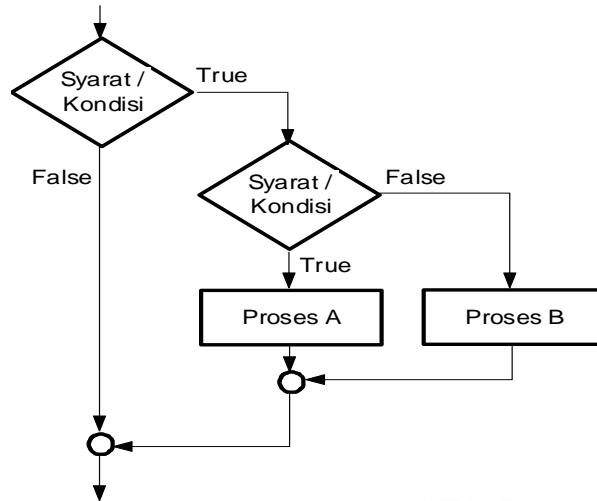
Struktur dari statemen If dapat berupa :

- **KONDISI TUNGGAL** yaitu Struktur If-Then
- **KONDISI GANDA** yaitu Struktur If-Then-Else
- **KONDISI JAMAK** yaitu Struktur If-Then Else If-Then-Else If-Then-Else



STATEMEN IF TERSARANG (NESTED IF)

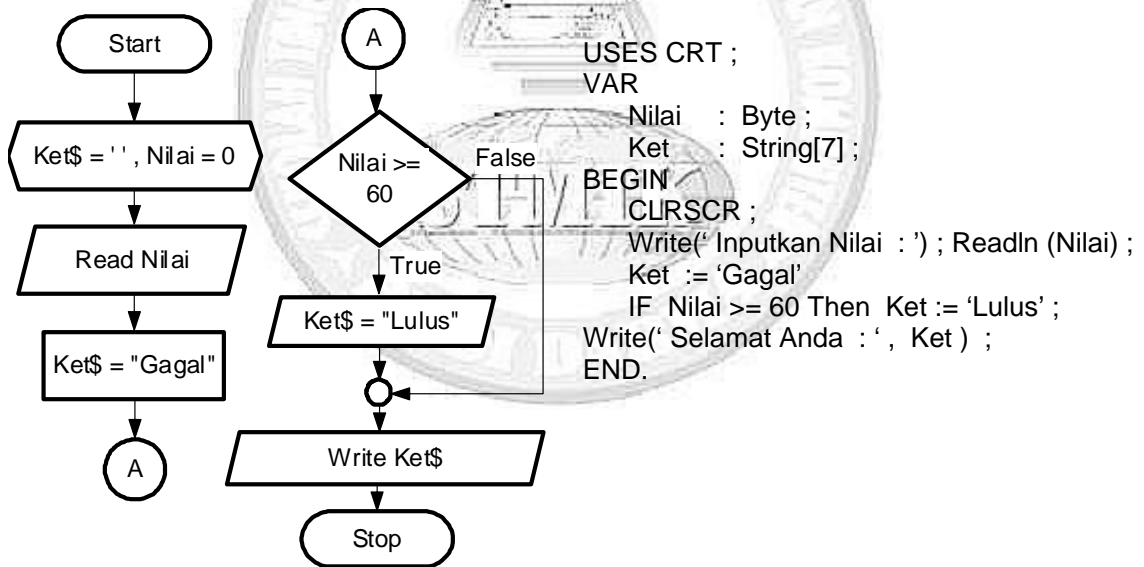
Yaitu IF yang berada di dalam statemen IF.



Contoh 1 :

Buatlah FlowChart & Program untuk menginputkan nilai mahasiswa. Jika Nilai ≥ 60 maka keterangan **Lulus**

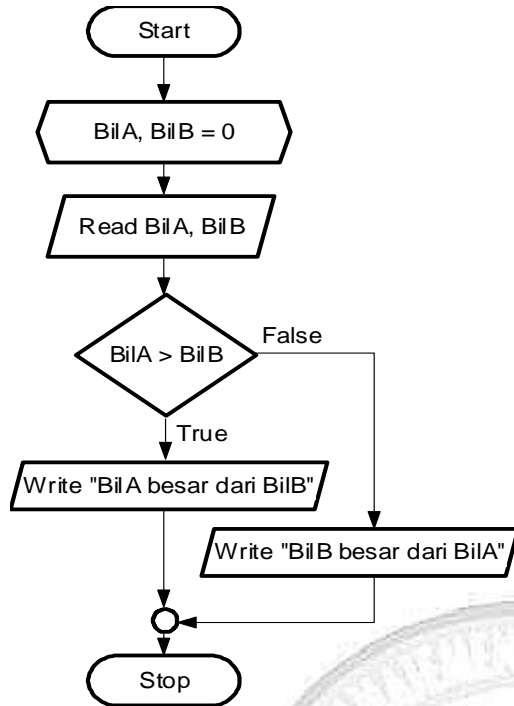
Jawab



Contoh 2 :

Diketahui 2 buah bilangan yaitu BilA, BilB. Buat FlowChart untuk mencetak bilangan yang terbesar diantara kedua bilangan tersebut!

Jawab



```

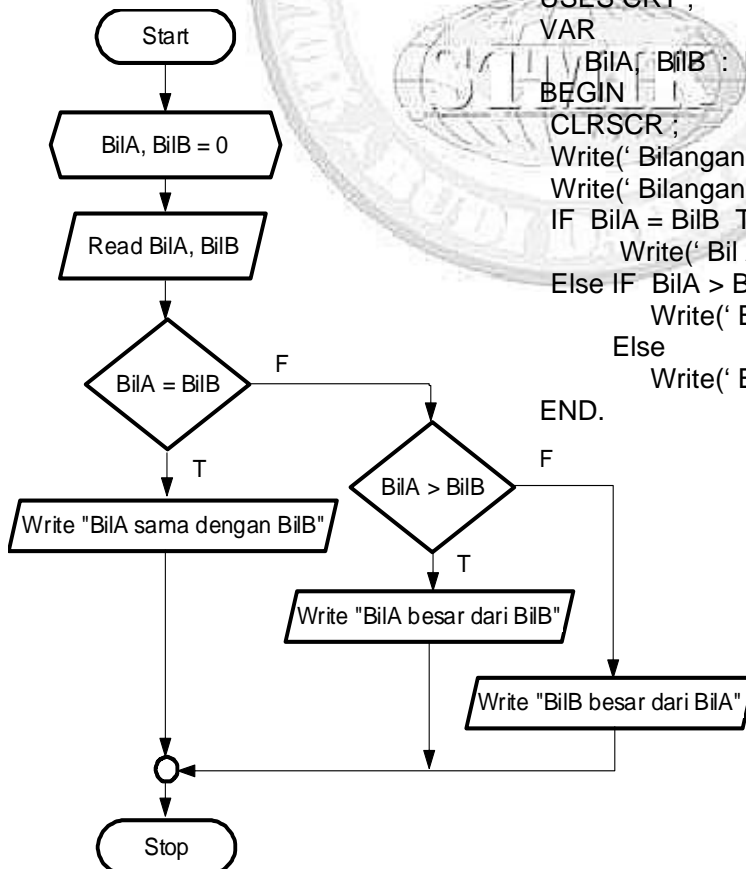
USES CRT ;
VAR
    BilA, BilB : Byte ;
BEGIN
    CLRSCR ;
    Write(' Bilangan A : ') ; Readln (BilA) ;
    Write(' Bilangan B : ') ; Readln (BilB) ;

    IF BilA > BilB Then
        Write(' Bil A Lebih Besar dari BilB')
    Else
        Write(' Bil B Lebih Besar dari BilA') ;
    END.
    
```

Contoh 3 :

Pada contoh 2 diatas, bagaimana jika Variabel BilA & BilB memiliki nilai yang sama ? Maka akan terjadi kesalahan logika yaitu : BilB besar dari BilA padahal bisa terjadi kemungkinan Bil B sama dengan BilA. Untuk mengatasi kendala tersebut maka digunakan IF Jamak.

Jawab



```

USES CRT ;
VAR
    BilA, BilB : Byte ;
BEGIN
    CLRSCR ;
    Write(' Bilangan A : ') ; Readln (BilA) ;
    Write(' Bilangan B : ') ; Readln (BilB) ;
    IF BilA = BilB Then
        Write(' Bil A sama dengan BilB')
    Else IF BilA > BilB Then
        Write(' Bil A Lebih Besar dari BilB')
    Else
        Write(' Bil B Lebih Besar dari BilA') ;
    END.
    
```

Contoh 4 :

Sebuah Toko penjualan menjual hanya tiga jenis barang. Yaitu :

Kode	Nama	Harga
SBN01	Sabun LUX	1500
RNS02	Rinso	11500
SMP03	Sunsilk	150

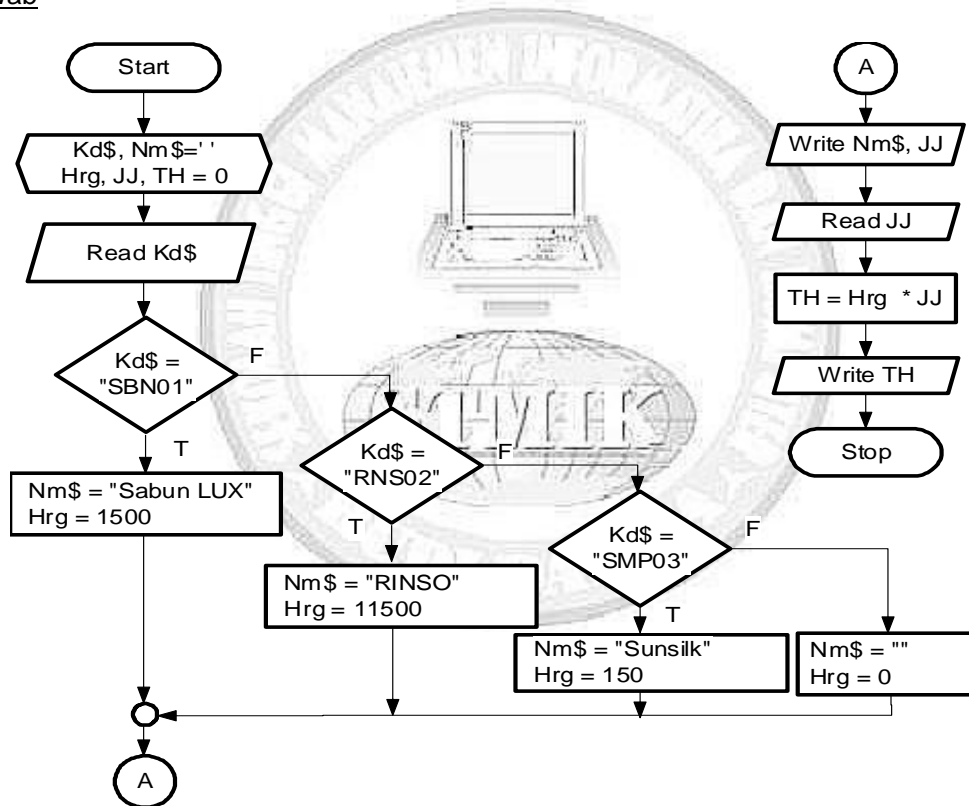
Buatlah FlowChart & Program dengan bentuk berikut :

DATA PENJUALAN BARANG

Kode Barang : x(5)
 Nama : x(30)
 Harga Rp : 9(8)
 Jumlah Jual : 99
Harga Jual Rp : 9(9)

Adapun proses digunakan untuk contoh diatas yaitu : mencari Nama, Harga barang berdasarkan Kodenya & Harga Jual = Harga * Jumlah Jual.

Jawab



Program

```

    USES CRT ;
    VAR
        Kd      : String[5] ;
        Nm      : String[15] ;
        JJ      : Byte ;
        Hrg, TH : Longint ;
    BEGIN
        CLRSCR ;
        Write(' Kode Barang : ') ; Readln (Kd) ;
        IF Kd='SBN01' Then
            Begin
    
```

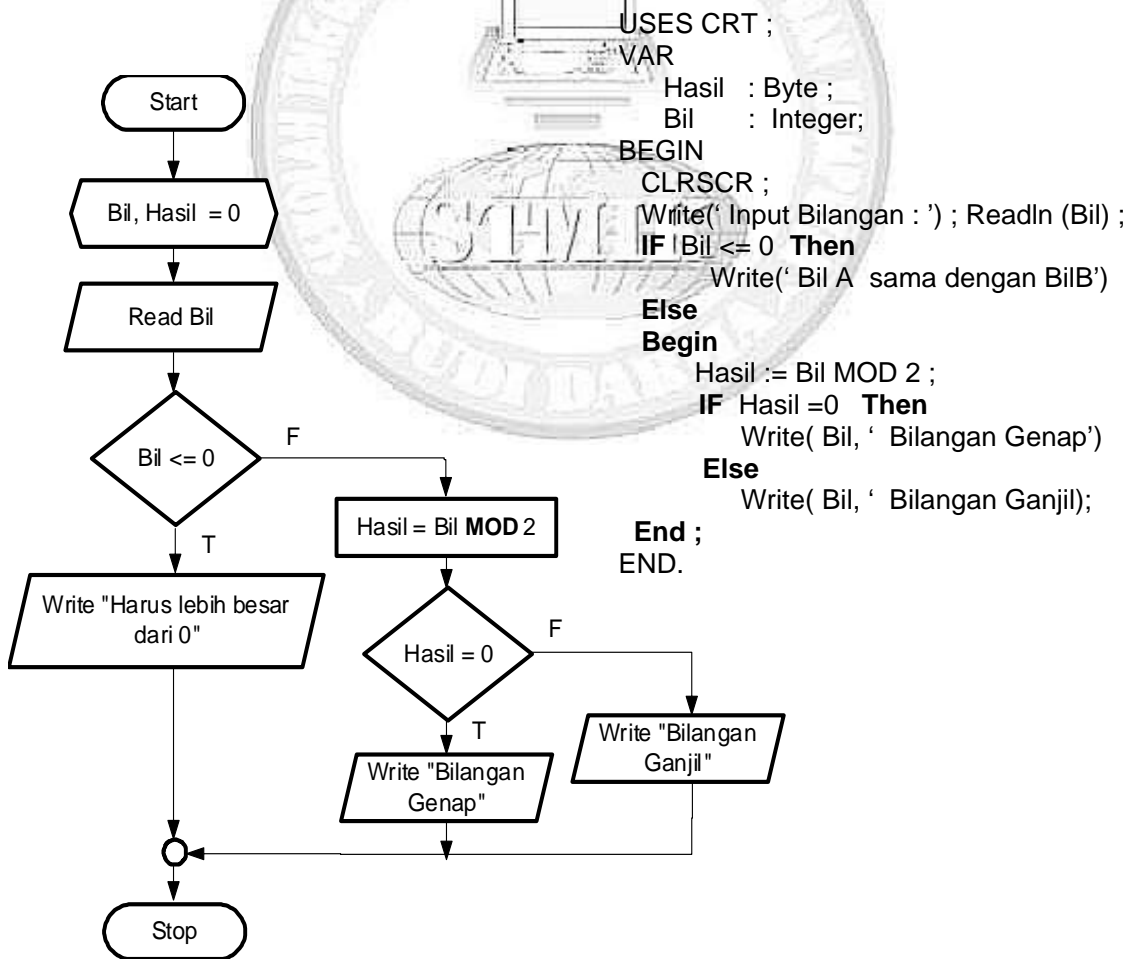
```

        Nm := 'Sabun LUX' ; Hrg := 1500 ;
    End
    ELSE IF Kd='RNS02' Then
        Begin
            Nm := 'Rinso' ; Hrg := 11500 ;
        End
        ELSE IF Kd='SMP03' Then
            Begin
                Nm := 'Sunsilk' ; Hrg := 150 ;
            End
        ELSE
            Begin
                Nm := '' ; Hrg := 0 ;
            End
        Write(' Jumlah Jual   :') ; Readln (JJ) ;
        TH := Hrg * JJ ;
        Write(' Harga Jual Rp. :', TH) ;
    END.
    
```

Contoh 5 :

Buatlah Flowchart & Program untuk mencari sebuah bilangan yang diinputkan apakah bilangan Genap atau bilangan Ganjil. Jika yang diinputkan 0 atau dibawah 0 maka akan tampil pesan bilangan harus diatas 0.

Jawab



LATIHAN :

1. Buatlah program untuk menghitung gaji harian pegawai bila diketahui ketentuan sebagai berikut :
 Gaji perjam = 500
 Bila jumlah jam kerja hari itu > 7 jam, maka kelebihanannya dihitung lembur yang besarnya 1.5 x gaji per jam.

Input : Jumlah Jam Kerja.
 Output : Gaji Harian Pegawai.

2. Sebuah perusahaan memberi komisi salesmannya berdasarkan ketentuan sebagai berikut :
 Bila salesman berhasil menjual barang seharga 500000 maka dia akan mendapat komisi sebesar 10%
 Bila lebih dari 500000, untuk 500000 pertama komisinya 10% sedangkan sisanya mendapat 15%
 Bila sebagai input adalah **jumlah penjualan**, buatlah program untuk menentukan komisi pegawai perusahaan tadi.

3. Buatlah program untuk mencari nilai akhir dengan bentuk output :

Data Nilai Ujian Algoritma & Pemrograman

 NPM : x (9)
 Nama : x (30)
 Nilai Mid : 999
 Nilai Semester: 999
 Nilai Akhir : 999
 Nilai Huruf : X

Proses

Mencari Nilai Akhir' = 40% * Nilai mid' + 60% * Nilai semester
 Mencari Nilai huruf dengan range dari Nilai Akhir.

Range Nilai Akhir	Nilai Huruf
80..100	A
70..79	B
55..69	C
40..54	D
0..39	E

4. Buatlah Flowchart beserta program untuk output berikut ini :
 PEMBAYARAN TAGIHAN LISTRIK PELANGGAN PT. PLN

 Nama Pelanggan : x (30)
 Kode : x (3)
 Jenis Pelanggan : x (15)
 Biaya Beban : 9(6)
 Harga per M³ Rp. : 9(9)
 Jumlah Pemakaian : 9999
 Total Harga Rp. : 9(8)
 Pajak 10% Rp : 9(7)
 Total Tagihan Rp. : 9(9)

Adapun Ketentuan untuk proses diatas yaitu :

- Nama Pelanggan, Jumlah Pemakaian n anda Input
- Untuk Jenis Pelanggan, Biaya Beban, Harga per M³ didapat berdasarkan Kode :

Kode	Jenis Pelanggan	Biaya Beban	Harga per M ³
L01	Pabrik	50000	2500
L02	Swalayan	35000	2000
L03	Toko	25000	1500
L04	Rumah	15000	750

- Total Harga = Harga per M³ * Jumlah Pemakaian + Biaya Beban
 - Pajak = 10 * Total Harga
 - Total Tagihan = Total Harga + Pajak
5. Untuk mendapatkan kredit pemilikan mobil, perlu dinilai penghasilan pemohon.
 Cara penilaian :
- pendapatan tetap/pokok dihitung penuh
 - pendapatan tambahan dihitung setengah
 - pendapatan keluarga (suami/istri) dihitung sepertiga.

Apabila jumlah pendapatan lebih besar atau sama dengan Rp. 1.000.000,- mendapat kredit SEDAN, kurang dari itu tetapi masih lebih besar dari Rp. 500.000,- mendapat kredit MINIBUS, selain itu tidak berhak mendapat kredit.

6. Mencari Sisa Uang Kuliah
 PEMBAYARAN UANG KULIAH

```

=====
Nama Mahasiswa : .....
NPM            : .....
Kode           : .....
Jurusan        : .....
Pembayaran Awal Rp: .....
Jumlah Cicilan : ...
Besar Cicilan Rp. : .....
Pembayaran Ke  : ...
Uang Kuliah Rp  : .....
Uang Kuliah Terbayar Rp : .....
Sisa Uang Kuliah Rp   : .....
=====
    
```

Ketentuan Program

- Nama, NIM, Kode & Pembayaran Ke diperoleh dari input
- Untuk mencari Jurusan, Pembayaran Awal, Jumlah Cicilan & Besar Cicilan dari Kode yaitu :

Kode	Jurusan	Pembayaran Awal Rp	Jumlah Cicilan	Besar Cicilan Rp.
SI	Sistem Informasi	2100000	7	300000
TI	Teknik Informatika	2500000	7	300000
KA	Komputer Akuntansi	1750000	6	200000
MI	Manajemen Informatika	1500000	6	250000

- Uang Kuliah = Pembayaran Awal + (Jumlah Cicilan * Besar Cicilan)
- Uang Kuliah Terbayar = Pembayaran Awal + (Pembayaran Ke * Besar Cicilan)
- Sisa Uang Kuliah = Uang Kuliah – Uang Kuliah Terbayar.

BAB V
STRUKTUR KENDALI II
(PERULANGAN – LOOP)

Di dalam pemrograman terdapat beberapa bentuk perulangan. Setiap pemrograman memiliki bentuk yang berbeda-beda. Yang akan dibahas disini yaitu pada perulangan pemrograman pascal yang menggunakan

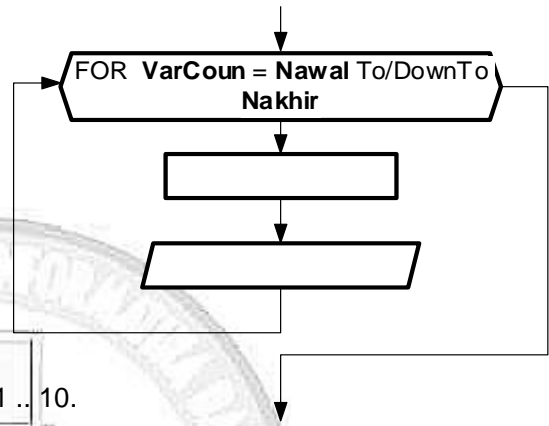
- For...Do
- While...Do dan
- Repeat...Until

FOR..DO

Digunakan untuk mengulang statemen atau satu blok statemen berulang kali sejumlah yang ditentukan tanpa suatu kondisi.

FOR...DO terdiri atas 2 jenis :

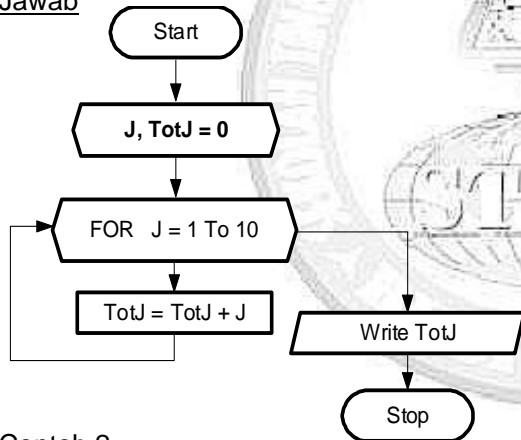
- For...Do Positif
Perulangan dimana Nawal lebih kecil dibanding NAKhir
- For...Do Negatif
Perulangan dimana Nawal lebih Besar dibanding NAKhir



Contoh 1

Hitunglah penjumlahan bilangan antara 1 .. 10.

Jawab



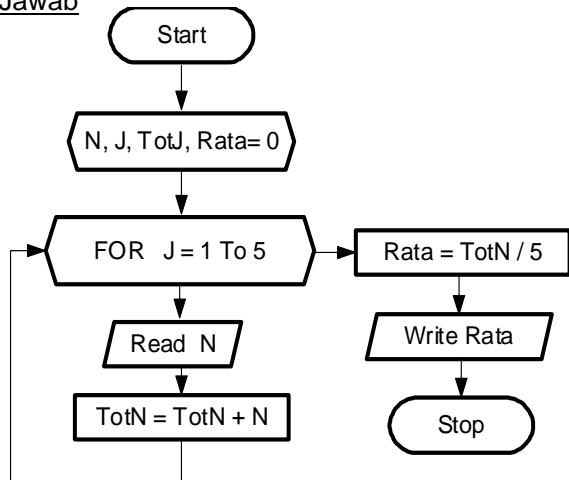
```

USES CRT ;
VAR
  J      : Byte ;
  TotJ   : Integer ;
BEGIN
  CLRSCR ;
  TotJ := 0 ;
  FOR J := 1 TO 10 DO
    TotJ := TotJ + J ;
  Write(' Total J : ', TotJ) ;
END.
    
```

Contoh 2

Hitunglah Jumlah Rata-Rata Nilai yang diinputkan sebanyak 5 kali.

Jawab



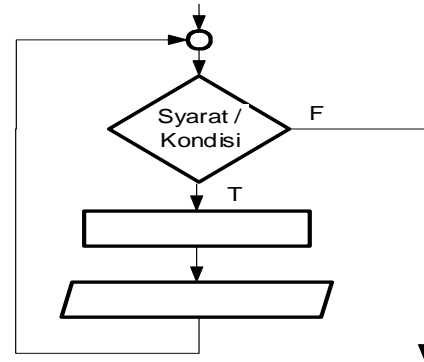
```

USES CRT ;
VAR
  J, N   : Byte   ;
  TotN   : Integer ;
  Rata   : Real   ;
BEGIN
  TotN := 0 ; Rata := 0 ;
  FOR J := 1 TO 5 DO
  Begin
    Write('Nilai Ke ', J, ' : ');
    Readln(N) ; TotN := TotN + N ;
  End ;
  Rata := TotN / 5 ;
  Write(' Nilai Rata-Rata : ', Rata:3:0) ;
END.
    
```

WHILE...DO

Jenis perulangan ini digunakan untuk mengulangi statemen atau satu blok statemen berulang kali yang jumlahnya belum bisa ditentukan, tergantung nilai kondisi yang terletak antara While...Do.

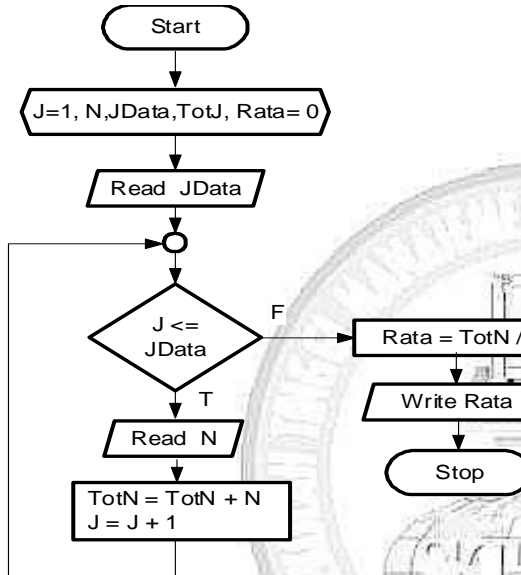
Perulangan akan dikerjakan bila nilai kondisinya bernilai benar dan akan berhenti jika kondisi bernilai salah.



Contoh 3

Hitunglah Jumlah Rata-Rata Nilai yang diinputkan sebanyak N kali.

Jawab



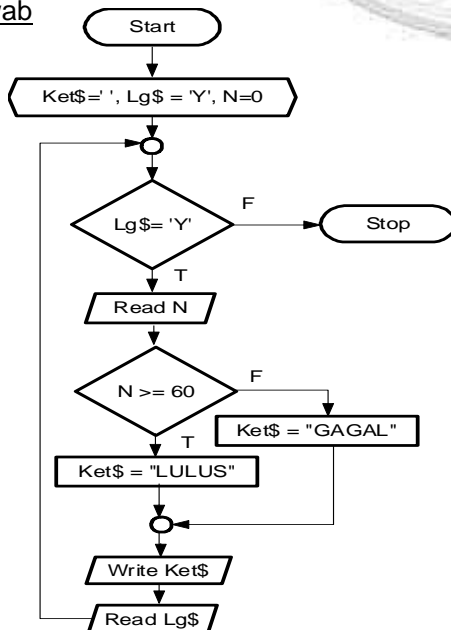
```

USES CRT ;
VAR
  JData,
  J, N : Byte ;
  TotN : Integer ;
  Rata : Real ;
BEGIN
  TotN := 0 ; Rata := 0 ; J := 1 ;
  Write ('Jumlah Data : '); Readln(JData);
  While J <= JData DO
  Begin
    Write('Nilai Ke ', J, ' : ');
    Readln(N);
    TotN := TotN + N;
    Inc(J);
  End ;
  Rata := TotN / JData ;
  Write(' Nilai Rata-Rata : ', Rata:3:0) ;
END.
    
```

Contoh 4

Buatlah program untuk menginputkan nilai dalam mencari Keterangan Lulus / Gagal. Jika nilai >= 60 maka keterangan Lulus atau Gagal. Program akan berjalan sampai **Menghitung lagi [Y/T]** dijawab 'T'

Jawab



```

USES CRT ;
VAR
  N : Byte ;
  Lg : Char ;
  Ket : String[8] ;
BEGIN
  Lg := 'Y';
  While Ucase(Lg='Y') DO
  Begin
    Write('Nilai Anda : '); Readln(N);
    IF N >= 60 Then
      Ket := 'LULUS'
    ELSE
      Ket := 'GAGAL';
    Write('Anda : ', Ket);
    Write('Menghitung Lagi [Y/T] : ');
    Readln(Lg);
  End ;
END.
    
```

Contoh 5 : Buatlah program untuk menginputkan gaji karyawan dengan ketentuan :

Input : Nama, Golongan, Jam Lembur
 Proses : Untuk mendapatkan gaji
 Jika Golongan = 'la' maka Gaji = 500000 & Uang Lembur = 2500
 Jika Golongan = 'lb' maka Gaji = 750000 & Uang Lembur = 3500
 Jika Golongan = 'lc' maka Gaji = 850000 & Uang Lembur = 5000

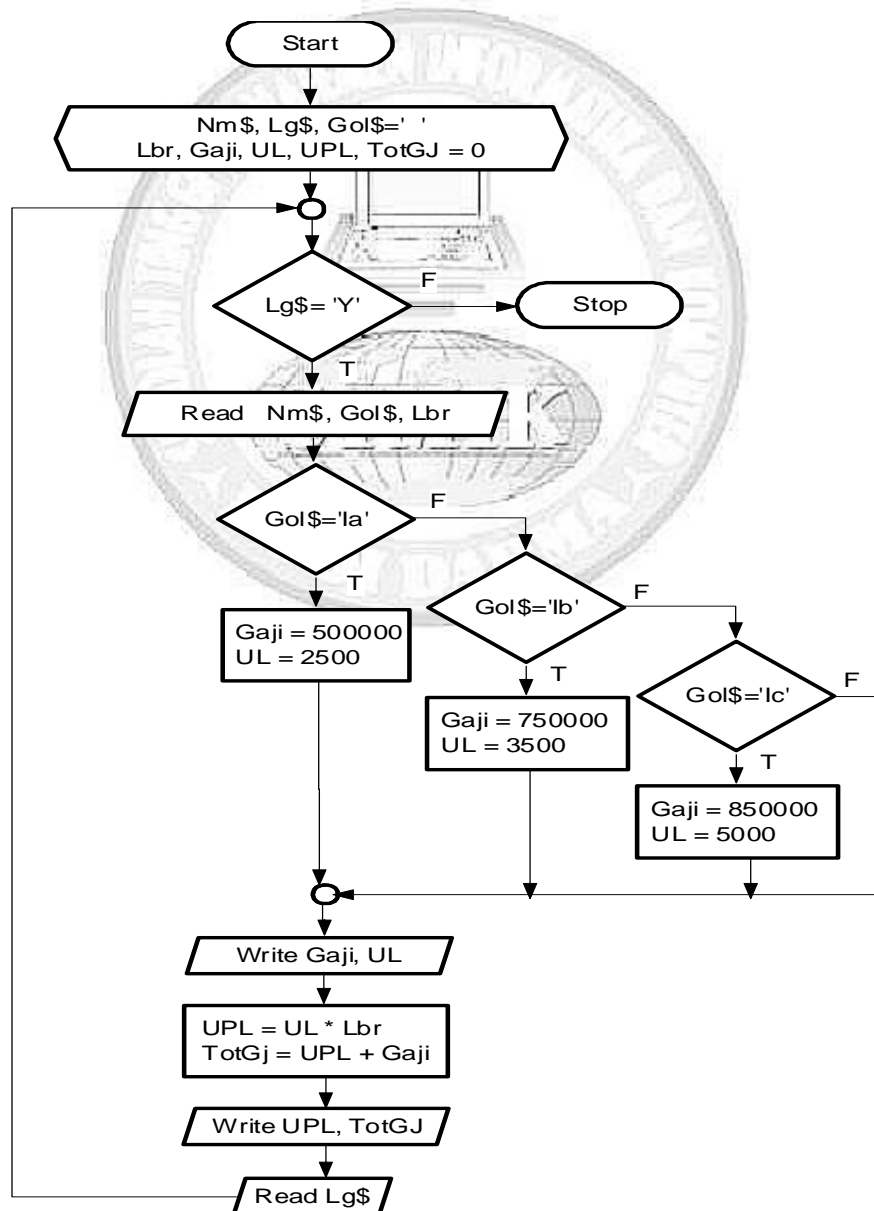
Untuk mendapatkan Upah Lembur
 Upah lembur = Jam Lembur * Uang Lembur

Untuk mendapatkan Total Gaji
 Total Gaji = Gaji + Upah Lembur

Output : Upah Lembur & Total Gaji

Ketentuan lain yaitu program menggunakan perulangan dan berjalan sampai **Menghitung lagi [Y/T]** dijawab 'T'

Jawab



Programnya :

USES CRT ;

VAR

```
Nm   : String[25] ;
Gol  : String[2] ;
Lg   : Char   ;
UPL, TotGj,
Gaji : Longint ;
Lbr, UL: Integer ;
```

BEGIN

Lg := 'Y' ;

While Upcase(Lg='Y') **DO**

Begin

{ Bagian Input Data }

```
Write('Nama Karyawan      : ' ) ; Readln( Nm ) ;
Write('Golongan [la, lb, lc] : ' ) ; Readln( Gol ) ;
Write('Jam Lembur          : ' ) ; Readln( Lbr ) ;
```

{ Bagian pemrosesan terhadap data input }

IF Gol='la' Then

Begin

Gaji := 500000 ; UL := 2500 ;

End

Else IF Gol='lb' Then

Begin

Gaji := 750000 ; UL := 3500 ;

End

Else IF Gol='lc' Then

Begin

Gaji := 850000 ; UL := 5000 ;

End;

{ Bagian Output / menampilkan hasil proses }

```
Write('Gaji                : ', Gaji) ;
```

```
Write('Uang Lembur / Jam   : ', UL) ;
```

```
UPL := UL * Lbr ;
```

```
TotGj := Gaji + UPL ;
```

```
Write('Upah Lembur Rp.    : ', UPL) ;
```

```
Write('Total Gaji Rp.     : ', TotGj) ;
```

```
Write('Menghitung Lagi [Y/T] : ' ) ; Readln( Lg ) ;
```

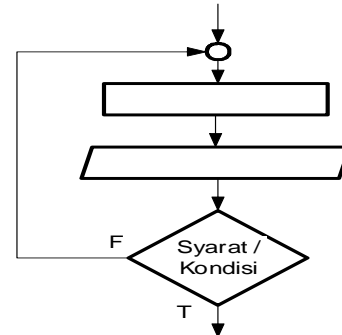
End ;

END.

REPEAT...UNTIL

Jenis perulangan ini digunakan untuk mengulang statemen atau satu blok statemen berulang kali, yang jumlahnya belum bisa ditentukan, tergantung nilai kondisi yang terletak setelah Until.

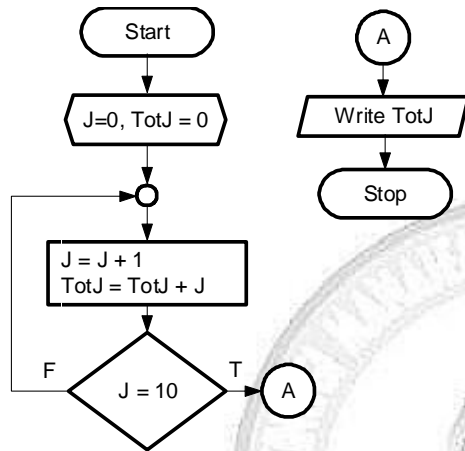
Perulangan akan dikerjakan bila nilai kondisinya bernilai salah dan akan berhenti jika kondisi bernilai benar.



Contoh 6

Hitunglah penjumlahan bilangan antara 1 .. 10 menggunakan Repeat...Until

Jawab



```

USES CRT ;
VAR
    J      : Byte ;
    TotJ   : Integer ;
BEGIN
    CLRSCR ;
    TotJ := 0 ; J := 0 ;
    REPEAT
        INC (J) ;
        TotJ := TotJ + J ;
    UNTIL J = 10 ;
    Write(' Total J : ', TotJ) ;
END.
    
```

LATIHAN

1. Buatlah FlowChart Program dibawah ini.

```

VAR i : Byte;
BEGIN
    CLRSCR ;
    Writeln(' I Keterangan ');
    Writeln('-----');
    FOR i:= 1 To 10 Do
    BEGIN
        Write (i);
        IF (i Mod 2)=0 then Writeln (' Genap') else Writeln (' Ganjil');
    END;
    Writeln('-----');
END.
    
```

2. Buatlah Flowchart & program untuk tampilan dibawah ini

```

2 STMIK
4 STMIK
6 STMIK
8 STMIK
10 STMIK
    
```

3. Buatlah FlowChart & Program untuk tampilan berikut ini .

```

5          10          15          20

25         30         35         40

45         50         55         60
    
```

4. Tampilkan Total Bilangan untuk 5 Bilangan Genap pertama.
5. Buatlah FlowChart beserta program untuk tampilan berikut :
 Bila **Ingin Menghitung Lagi [Y/T]** : kita isi dengan **Y** maka program akan menginputkan kembali dari awal

DAFTAR NILAI MAHASISWA
 JURUSAN INFORMATIKA

```

=====
Nama Mahasiswa      : x(20)
NIM Mahasiswa       : x(9)
Nilai Tugas         : 999
Nilai Mid Test      : 999
Nilai Final Test    : 999
Total Nilai         : 999
Nilai Huruf         : X
=====
    
```

Ingin Menghitung Lagi [Y/T] :

Ketentuan Program :

- a Nama, NIM, Nilai Tugas, Nilai Mid Test dan Nilai Final Test diperoleh dari input
- b Sedangkan Total Nilai diperoleh dari rumus:

$$\text{Total Nilai} = (20\% * \text{Nilai Tugas}) + (30\% * \text{Nilai Mid}) + (50\% * \text{Nilai Final})$$
- c Range dari Nilai Huruf yaitu :

Total Nilai	Nilai Huruf
80 – 100	A
79 – 70	B
69 – 51	C
50 – 41	D
40 – 0	E

6. Buatlah Flowchart & program untuk tampilan berikut ini
 Apabila diketahui harga beras per kilogram sebagai berikut:
 Jenis IR 64 = Rp. 2000,-
 Jenis C4 = Rp. 3000,-
 Jenis Delanggu = Rp. 3550,-
 Jenis Rajalele = Rp. 6000,-

**TABEL HARGA BERAS IR 64, C4, DELANGGU DAN RAJALELE
 TOKO "MANTEP" SALATIGA**

No	Jumlah (Kg)	IR64	C4	Delanggu	RajaLele
1	5
2	10
3	15
...
50	250

**BAB VI
ARRAY**

Sebuah variabel yang kita gunakan pada bab-bab sebelumnya hanya mampu menampung/menyimpan sebuah nilai data.

Contohnya : nama = 'Danur'

Pada saat variabel nama tersebut di atas yang telah tersimpan data dengan nama 'Danur' akan masukkan lagi data dengan : nama = 'NurAinun'. Maka yang terjadi adalah data 'NurAinun' lah yang tersimpan pada nama tersebut.

Bagaimana jika kita harus menyimpan 50 orang data mahasiswa, tetapi variabel yang kita gunakan tetap 1 (satu) mis Nama ?

Jawabannya adalah kita harus menggunakan suatu variabel dengan jenis Array. Dalam kegiatan pemrograman sekumpulan data yang bertipe sama perlu disimpan sementara didalam memori komputer untuk sewaktu-waktu dimanipulasi. Bila kumpulan data itu disimpan secara berurut didalam memori, maka tiap elemen data dapat diacu dengan menggunakan indeks. Indeks menyatakan posisi data relatif didalam kumpulannya. Struktur penyimpanan seperti ini dinamakan larik (array).

Larik adalah struktur data yang menyimpan sekumpulan elemen yang bertipe sama, setiap elemen diakses langsung dengan melalui indeksnya. Indeks larik haruslah tipe data yang menyatakan keterurutan atau karakter.

Jumlah elemen larik tidak dapat diubah, ditambah atau dikurang selama pelaksanaan program. Mendefenisikan banyaknya elemen larik berarti memesan sejumlah tempat dimemori. Memori mengalokasikan sejumlah lokasi memori sebanyak elemen larik yang bersangkutan. Tipe elemen larik dapat bertipe sederhana (integer, real, char, boolean, string), tipe terstruktur (tipe bentukan seperti record) atau bahkan bertipe larik.

Array dapat dibedakan atas beberapa macam:

- Array berdimensi satu
- Array berdimensi dua
- Array berdimensi banyak

PENDEKLARASIAN ARRAY DIMENSI SATU

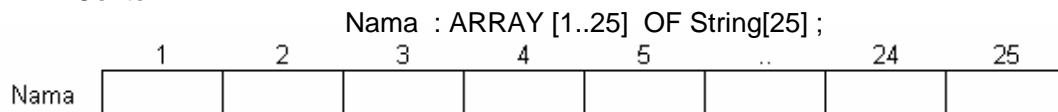
Seperti halnya variabel biasa, array juga harus didefinisikan sebelum dapat digunakan. Pendeklarasian array harus diletakkan pada bagian VAR. Bentuk umum pendeklarasian array dimensi satu yaitu :

VAR
nmVariabel : ARRAY [index] OF typeData

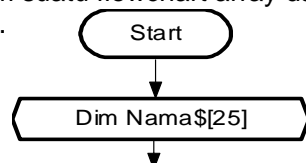
Keterangan :

- nmVariabel : Nama larik yang dideklarasikan
- index : Batasan/Jumlah Index (cacah elemen)
- typeData : Tipe data dari Larik

Contoh : VAR



Dalam suatu flowchart array dapat di deklarasikan dengan menggunakan pernyataan DIM. Contoh.



MENGAKSES ELEMEN ARRAY DIMENSI SATU

Setelah suatu array didefinisikan, elemen array dapat diakses dengan bentuk :

nmVarArray[subscript] := nilai data / variable ;

Sebagai contoh

Nama [2] := 'Danur' ;
 Nama [1] := 'NurAinun' ;

Dalam bentuk yang lebih umum

Nama [i] → jika diletakkan dalam perulangan.

Menampilkan data Array

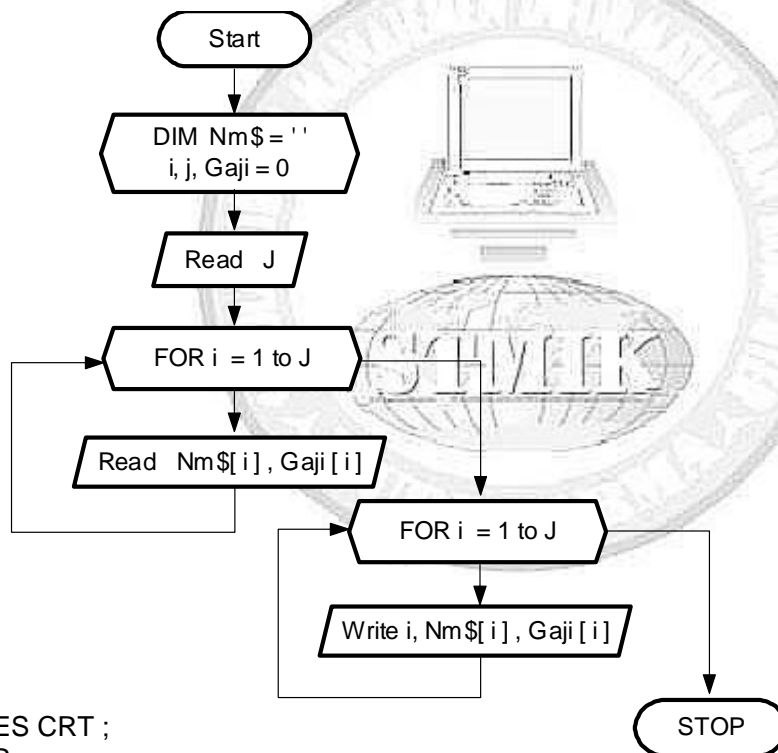
Write ('Nama : ' , nama[i]) ;

Menginputkan data Array

Readln ('Nama : ' , nama[i]) ;

Contoh 1.

Flowchart berikut ini berguna untuk menginputkan Data Karyawan yaitu Nama & Gaji.



USES CRT ;
 VAR

Nm : Array[1..30] of String[20] ;
 Gaji : Array[1..30] of Longint ;
 i, j : Byte ;

BEGIN

Clrscr

{Bagian Input Data}

Writeln('Inputkan Jumlah Data : ') ; Readln(J) ;

For i := 1 To J Do

Begin

Write('Inputkan Nama Ke ', i , ':') ; Readln(Nm[i]) ;

Write('Inputkan Gaji Ke ', i , ':') ; Readln(Gaji[i]) ;

End;

```

Clrscr ;
{Bagian Menampilkan Data yang telah diinputkan }
Writeln('-----');
Writeln('No.      Nama                Gaji');
Writeln('-----');

For i := 1 To J Do
Begin
    Writeln( i: 3, ' ', Nm [ i ]: 25, Gaji[i]:8 );
End ;

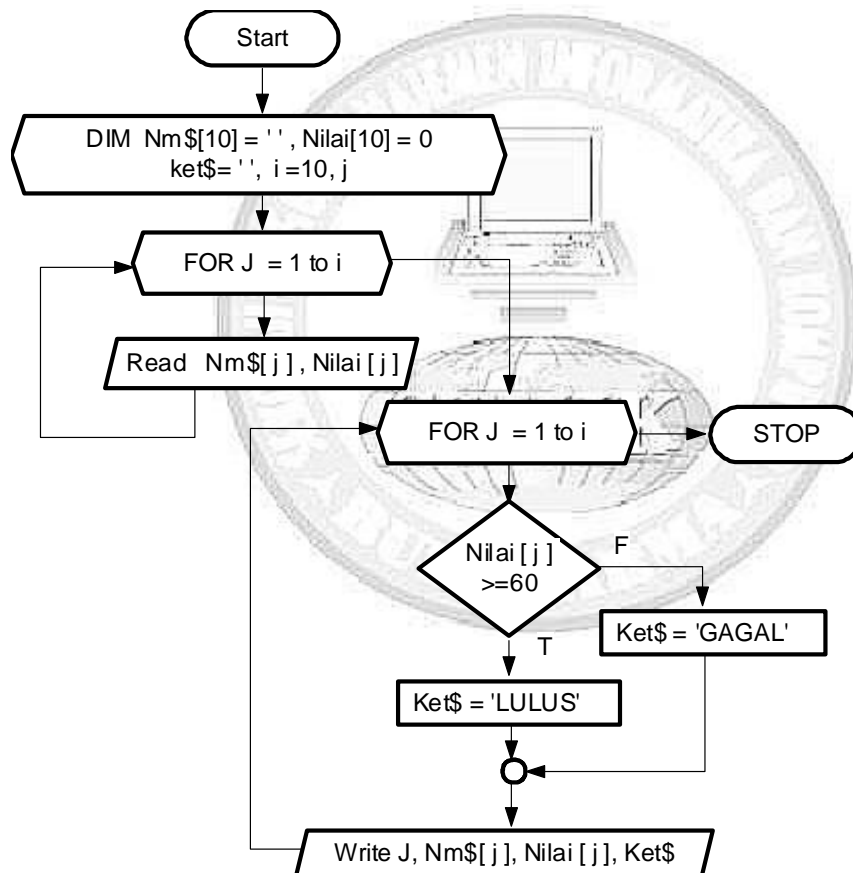
Writeln('-----')
Readln ;

```

END.

Contoh 2.

Buatlah FlowChart & Program untuk menginputkan nilai 10 orang mahasiswa. Jika Nilai ≥ 60 maka keterangan **Lulus**



USES CRT ;
VAR

```

Nm      : Array[1..10] of String[20] ;
Nilai   : Array[1..10] of Byte      ;
i, j    : Byte ;
Ket     : String[15] ;

```

BEGIN

```

Clrscr
{Bagian Input Data}

```

```

i := 10 ;
For J := 1 To i Do
Begin
    Writeln(' Data Ke :', J ) ;
    Write('Nama      : '); Readln( Nm[ J ] ) ;
    Write('Nilai [1..100] : '); Readln( Nilai[ J ] ) ;
End;

Clrscr ;
{Bagian Menampilkan Data yang telah diinputkan }
Writeln('-----');
Writeln('No.      Nama              Nilai ');
Writeln('-----');

For J := 1 To i Do
Begin
    IF Nilai [ J ] >= 60 THEN Ket := 'LULUS' ELSE Ket := 'GAGAL' ;
    Writeln( J: 3, ' ', Nm [ J ]: 25, Nilai [ J]:4, ' ', Ket ) ;
End ;

Writeln('-----')
Readln

```

END.

PENDEKLARASIAN ARRAY DIMENSI DUA

Array dimensi 2 atau 3 biasa digunakan untuk pemrosesan matrik berordo dua atau lebih. Bentuk umum pendeklarasian array dimensi dua yaitu :

```

VAR
    nmVariabel : ARRAY [index1, Index2 ] OF typeData ;

```

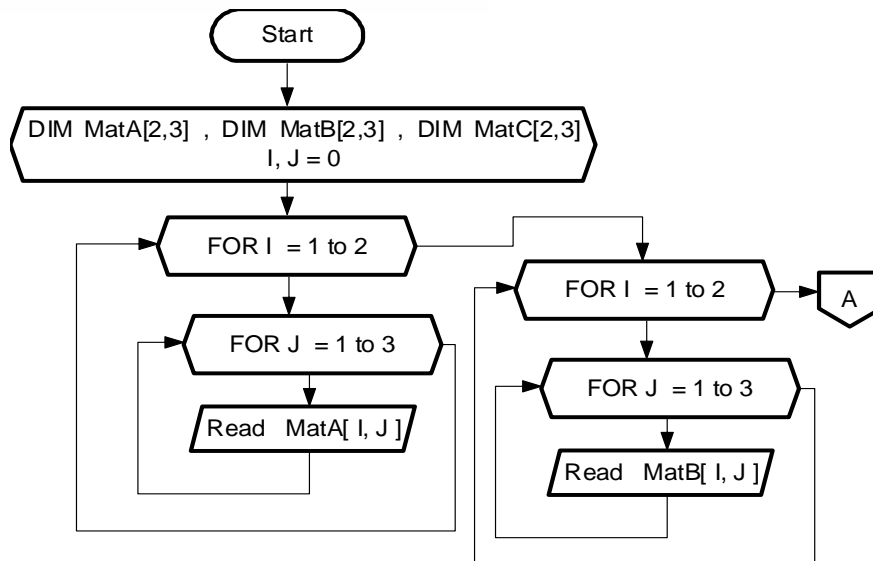
Contoh : VAR

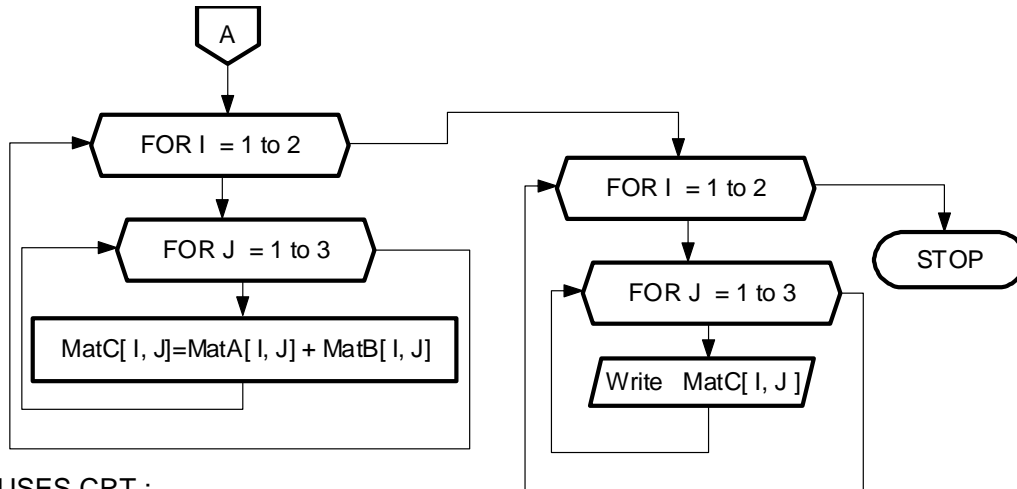
```

MatA : ARRAY [1..2, 1..3] OF Byte ;
MatB : ARRAY [1..2, 1..3] OF Byte ;

```

		Colom		
		1	2	3
Baris	1			
	2			





USES CRT ;

VAR

MatA : Array[1..2 , 1..3] of Byte ;

MatB : Array[1..2 , 1..3] of Byte ;

MatC : Array[1..2 , 1..3] of Byte ;

I, J : Byte ;

BEGIN

Clrscr ;

{Input Matrik A }

For I := 1 To 2 Do

Begin

For J := 1 To 3 Do

Begin

Write(' Data Matrik A [', I, ', ', J, ':') ; Readln(MatA [I, J]) ;

End;

End ;

{Input Matrik B }

For I := 1 To 2 Do

Begin

For J := 1 To 3 Do

Begin

Write(' Data Matrik B [', I, ', ', J, ':') ; Readln(MatB [I, J]) ;

End;

End ;

{Proses Nilai Matrik C }

For I := 1 To 2 Do

Begin

For J := 1 To 3 Do

Begin

C[I, J] = A[I, J] + B[I, J] ;

End;

End ;

{Tampilkan Hasil Matrik C }

For I := 1 To 2 Do

Begin

For J := 1 To 3 Do

Begin

Write(' Data Matrik C [', I, ', ', J, ':', MatC [I, J]) ;

End;

End ;

END;

LATIHAN

1. Buatlah Program untuk Menampilkan LayOut seperti berikut ini
Laporan Data Nilai Mahasiswa AMIK

No.	NIM	Nama	Mata Kuliah	Nilai Mid	Nilai Semester	Nilai Huruf	Ket
1.	0303001	Aldyan	Pascal I	90	90	A	Lulus
...

Ketentuan Proses

- NIM, Nama, Nilai Mid Test dan Nilai Sem diperoleh dari input
- Sedangkan Total Nilai diperoleh dari rumus:

$$\text{Total Nilai} = (40\% * \text{Nilai Mid}) + (60\% * \text{Nilai Sem})$$
- Range dari Nilai Huruf yaitu :

Total Nilai	Nilai Huruf
80 – 100	A
79 – 70	B
69 – 51	C
50 – 41	D
40 – 0	E

- Ket di dapat dari Nilai E yaitu Ket = 'GAGAL' selain itu Ket = 'LULUS'

2. Buatlah program untuk perkalian matrik berordo 3.

3. Buatlah program berikut ini

CV. "INTAN GANDINI" yang bergerak dalam konsultan teknik akan membuat laporan gaji semua karyawan yang berjumlah 100 orang, dengan data-data sebagai berikut :

- NoPeg (Nomer pegawai)
- Nama (Nama pegawai)
- Gol (Golongan)
- JJL (Jumlah jam lembur)

Laporan yang diinginkan dari data-data tersebut adalah

**LAPORAN GAJI PEGAWAI
 CV. "INTAN GANDINI"**

Jl. Waru Doyong No. 69 Semarang

No	Nomor Pegawai	Nama Pegawai	Gol	Gaji Pokok	Tunjangan	Jumlah Jam Lembur	Uang Lembur	Gaji Bersih
99	x(8)	x(25)	x	9(9)	9(9)	9999	9(9)	9(9)
99	x(8)	x(25)	x	9(9)	9(9)	9999	9(9)	9(9)

Ketentuan Proses

1. Mencari Gaji Pokok, Tunjangan, Uang Lembur

Gol	Gaji Pokok	Tunjangan	Uang Lembur/Jam
1	100000	50000	3500
2	200000	100000	4500
3	350000	200000	6000
4	500000	350000	Tidak Ada

- Uang Lembur = Jumlah Jam Lembur * Uang Lembur/Jam
- Gaji Bersih = Gaji Pokok + Tunjangan + Uang Lembur

BAB VII RECORD (REKAMAN)

Seperti halnya array(larik), record juga punya elemen, di sini elemennya disebut "Field". Tiap elemen bisa punya type data yang berbeda. Banyaknya field dapat bervariasi atau bisa juga tetap.

Record yang punya field yang bervariasi disebut 'Variant Record'. Deklarasi type data Record dapat ditetapkan di bagian deklarasi Var atau Type.

```
Syntax :
TYPE
    nmPengenal = RECORD
        Field1 : type1;
        Field2 : type2;
        :
        fieldn : typen;
    END;
VAR
    nmRec: nmPengenal ;
```

Keterangan :

nmPengenal : Nama Dari Record
 Field1, Field2, Fieldn : Nama dari Variabel/Field yang terdapat dalam Record
 Type1, Type2, Typen : Type Dari tiap-tiap Field yang dideklarasikan
 nmRec : Nama Dari Record yang di deklarasikan

Contoh

```
TYPE
    Mahasiswa = RECORD
        NIM : String[8] ;
        Nama : String[20] ;
        Alamat : String[30] ;
        Sex : Char ;
        Umur : Byte ;
    END;
VAR
    RecMhs : Mahasiswa ;
```

MEMBACA & MENULIS FIELD REKAMAN

Dapat dilaksanakan pada seluruh record atau sebagian record.

```
Syntax :
    nmRec.Field1 ;
```

Tanda titik harus ditulis

Notasi ini disebut Penandaan field ('field designator').

Berbeda dengan statement pemberian (assignment), maka untuk membaca dan menulis record selalu dilakukan menurut Field yang ada, tidak dilakukan secara keseluruhan record.

```
Contoh : Membaca    Readln(RecMhs.NIM );
                  Readln(RecMhs>Nama );

                  Menulis    Writeln(RecMhs.NIM );
                  Writeln(RecMhs>Nama );
```

Contoh 1 :

```

TYPE
    Mahasiswa = RECORD
        NIM    : String[8]    ;
        Nama   : String[20]   ;
        Umur   : Byte        ;
    END;
Var
    RecMhs : Mahasiswa    ;
Begin
    {Bagian Input Data}
    Writeln('NIM Anda  :'); readln( RecMhs.NIM );
    Writeln('Nama      :'); readln( RecMhs>Nama );
    Writeln('Umur     :'); readln( RecMhs.Umur );
    {Bagian Menampilkan Data yang telah diinputkan }
    Writeln('NIM Anda  :', RecMhs.NIM );
    Writeln('Nama      :', RecMhs>Nama );
    Writeln('Umur     :', RecMhs.Umur );
    Readln;
End.

```

Contoh 8.1 : Program dengan Penggunaan Record.

STATEMEN WITH

Digunakan untuk mengurangi penulisan yang berulang-ulang untuk nama recordnya. Maka dengan menggunakan statement WITH hanya cukup sekali saja menyebutkan nama recordnya.

Syntax :

```

WITH nmRecord DO
    {langsung dimasukkan/dicetak fieldnya}

```

Contoh 2 :

```

TYPE
    Mahasiswa = RECORD
        NIM    : String[8]    ;
        Nama   : String[20]   ;
        Umur   : Byte        ;
    END;
Var
    RecMhs : Mahasiswa    ;
Begin
    With RecMhs Do
    Begin
        {Bagian Input Data}
        Writeln('NIM Anda  :'); readln(NIM );
        Writeln('Nama      :'); readln>Nama );
        Writeln('Umur     :'); readln(Umur );
        {Bagian Menampilkan Data yang telah diinputkan }
        Writeln('NIM Anda  :', NIM );
        Writeln('Nama      :', Nama );
        Writeln('Umur     :', Umur );
    End;
    Readln;
End.

```

Contoh 8.2 : Penggunaan Statemen WITH pada RECORD.

LATIHAN1. Nama Program : **Latih81.Pas**

```

Program Latih81; {Program input data pada record}
Type
  Barang = Record
    Kode   : String[5] ;
    Nama   : String[20] ;
    Jumlah: Integer   ;
    Harsat : Longint   ;
  End;
Var
  RecBrg : Array [1..40] of Barang ;
  Lagi   : Char;
  i, J   : Integer;
Begin
  i:=0 ; J := 0 ;
  { Untuk Menginputkan Data Barang}
  Repeat
    Inc(i)
    Write('Kode Barang   :'); Readln(RecBrg[ i ].Kode);
    Write('Nama         :'); Readln(RecBrg[ i ].Nama);
    Write('Jumlah       :'); Readln(RecBrg[ i ].Jumlah);
    Write('Harga satuan  :'); Readln(RecBrg[ i ].Harsat);
    Write('Ada lagi [Y/T] :'); Lagi := Readkey ;
  Until ( Lagi<>'Y' ) or ( Lagi<>'y' ) { bisa ditulis Until Uppcase(lagi)='Y' }

  { Menampilkan Data Barang }
  Writeln('-----');
  Writeln('No. Kode   Nama           Jumlah   Harga ');
  Writeln('-----');
  For J := 1 To i Do
  Begin
    With RecBrg[ J ] Do
    Begin
      Writeln( i:3, Kode:6, Nama:25, Jumlah:4, Harsat:9 );
    End ;
  End ;
  Writeln('-----');
  Readln;
End.

```

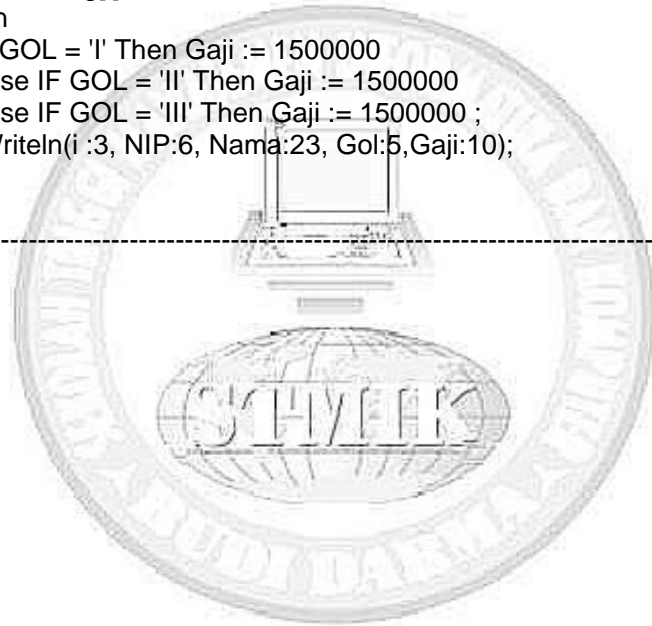
2. Nama Program : **Latih82.Pas**

```

Program Rekaman2 ;
TYPE
  PEGAWAI = RECORD
    NIP      : STRING[5] ;
    NAMA     : STRING[20];
    GOL      : STRING[3] ;
  END;
VAR
  RecPeg    : Array[1..20] Of Pegawai ;
  Gaji      : Longint   ;
  i, j      : Byte     ;
BEGIN
  {Bagian Input Data...}
  Write('Jumlah Data : '); Readln(J) ;

```

```
For i := 1 to j Do
Begin
  Writeln('INPUT DATA PEGAWAI');
  Writeln('-----');
  Writeln('Data Ke : ', i);
  With RecPeg[i] Do
  Begin
    Write('NIP      : '); Readln(NIP) ;
    Write('NAMA      : '); Readln>Nama) ;
    Write('GOL [I,II,III] : '); Readln(Gol) ;
  End;
End;
{Bagian Output...}
Writeln('-----');
Writeln('No  NIP      NAMA              GOL      GAJI ');
Writeln('-----');
For i := 1 to j do
Begin
  With RecPeg[i] Do
  Begin
    If GOL = 'I' Then Gaji := 1500000
    Else IF GOL = 'II' Then Gaji := 1500000
    Else IF GOL = 'III' Then Gaji := 1500000 ;
    Writeln(i :3, NIP:6, Nama:23, Gol:5,Gaji:10);
  End;
End;
Writeln('-----');
Readln ;
END.
```

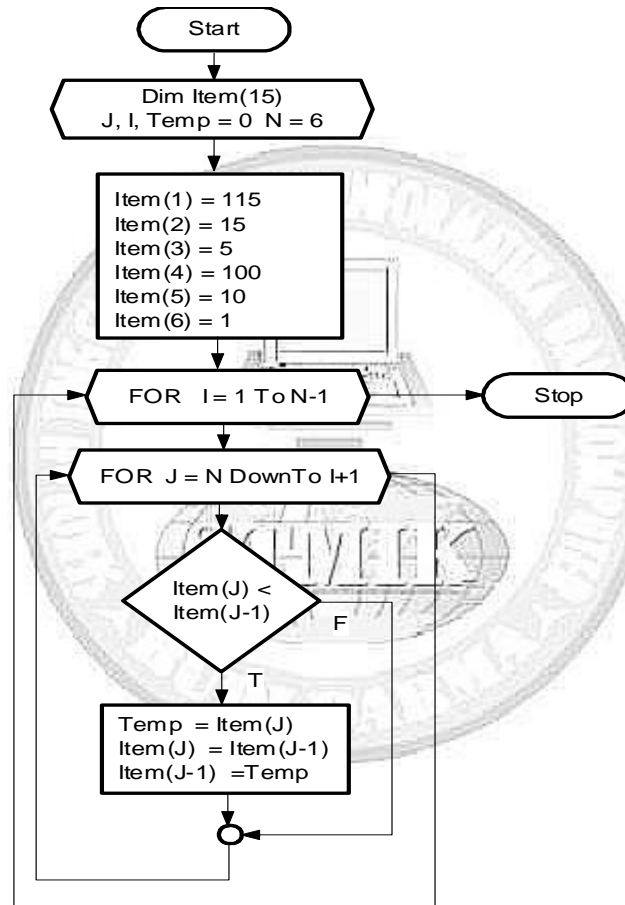


BAB VIII SORTING

Pengurutan adalah proses mengatur sejumlah objek menurut aturan atau susunan tertentu. Urutan itu dapat secara menaik (ascending) atau menurun (descending) Data yang sudah terurut memiliki beberapa keuntungan. Selain mempercepat pencarian dari data yang terurut juga dapat diketahui secara langsung harga maksimum dan harga minimum.

METODE BUBBLE SORT

Proses pengurutan data dilakukan dengan membandingkan 2 bilangan atau nilai yang berurutan dalam jajaran bilangan atau data yang akan diurutkan. Untuk memeriksa apakah urutan letak dari keduanya benar / ya, maka dilanjutkan dengan membandingkan 2 bilangan berikutnya. Sedangkan jika tidak benar (salah) maka letak kedua bilangan tersebut di pertukarkan dan baru dilanjutkan dengan membandingkan 2 bilangan berikutnya dalam jajaran tersebut.



Programnya.

```

VAR
  i, j, Temp, N, K: Integer ;
  Item           : Array[1..15] of Integer ;
BEGIN
  N := 6;
  Item[1] := 115 ; Item[2] := 15 ;
  Item[3] := 5   ; Item[4] := 100 ;
  Item[5] := 10  ; Item[5] := 1 ;
  Write ('Data Sebelum Diurut  : ');

  FOR i:= 1 TO N DO Write(Item[ i ] :4, ' ');
  Writeln ;
  
```

```

FOR I:= 1 to N-1 Do
Begin

    Write (' Pengurutan Ke : ', I ) ;
    For K:= 1 to N do Write( Item[K]:4, ' ' ) ;

    For J := N DownTo I+1 DO
    Begin
        IF Item[J] < Item[J-1] Then
        Begin
            Temp      := Item[J]      ;
            Item[J]   := Item[J-1]   ;
            Item[J-1] := Temp        ;
        End;
    End;
    Writeln;
End;

Write ('Data Setelah Diurut : ');

For i:= 1 to N do Write(Item[ I ]:4, ' ')
END.

```

Contoh 9.1 Program Sorting

LATIHAN

1. Buatlah program berikut ini :

```

Program Sort2 ;
TYPE
    PEGAWAI = RECORD
        NIP      : STRING[5];
        NAMA     : STRING[20];
        GOL      : STRING[3];
    END;

VAR
    RecPeg      : Array[1..20] Of Pegawai ;
    Temp        : Array[1..20] Of Pegawai ;
    Gaji        : Longint   ;
    i, j, k     : Byte      ;

BEGIN
    {Bagian Input Data...}
    Write('Jumlah Data : '); Readln(J) ;
    For i := 1 to j Do
    Begin
        Writeln('INPUT DATA PEGAWAI');
        Writeln('-----');
        Writeln('Data Ke : ', i) ;
        With RecPeg[i] Do
        Begin
            Write('NIP      : '); Readln(NIP) ;
            Write('NAMA     : '); Readln>Nama) ;
            Write('GOL [I,II,III] : '); Readln(Gol) ;
        End;
    End;

```

```

    End;
  End;
  {Bagian Pengurutan...}
  For k := 1 to j-1 Do
  Begin
    For i := j DownTo k+1 Do
    Begin
      IF RecPeg[ i ].NIP < RecPeg[ i-1 ].NIP Then
      Begin
        Temp[ i ] := RecPeg[ i ] ;
        RecPeg[ i ] := RecPeg[ i-1 ] ;
        RecPeg[ i-1 ] := Temp[ i ] ;
      End;
    End;
  End;
  End;
  {Bagian Output...}
  Writeln('-----');
  Writeln('No NIP NAMA GOL GAJI ');
  Writeln('-----');
  For i := 1 to j do
  begin
    With RecPeg[i] Do
    Begin
      If GOL = 'I' Then Gaji := 1500000
      Else IF GOL = 'II' Then Gaji := 1500000
      Else IF GOL = 'III' Then Gaji := 1500000
      Else Gaji := 0 ;
      Writeln(i :3, NIP:6, Nama:23, Gol:5,Gaji:10);
    End;
  End;
  Writeln('-----');
  Readln ;
END.

```

BAB IX EDITOR PASCAL

Untuk menggunakan Pascal kita harus menyediakan File-file pendukung pascal itu sendiri diantaranya :

- **TURBO.***
Turbo.Exe, Turbo.TP, Turbo.TPL, Turbo.HLP
Turbo.Exe → Merupakan file Editor pascal. Sehingga bila kita ingin menggunakan Pascal cukup Ketikkan :
C:\Pascal5**Turbo** [Enter]
Turbo.TP → file yang berguna untuk menyimpan seting dari Turbo Pascal kita.
Turbo.TPL → merupakan file yang sangat berguna untuk meLINK program atau mengcompile program sebelum di jalankan. Jika file ini tidak ada maka akan berakibat file tidak akan bisa di compile.
Turbo.HLP → merupakan file yang membantu kita mengenai penjelasan perintah-perintah tertentu (sebagai HELP)
- ***.CHR**
Merupakan file yang berekstensi .CHR yang berguna untuk pengaturan Character di dalam pemrograman yang berorientasi dengan grafik. Pascal menyediakan 10 jenis huruf dan tiap-tiap huruf tersebut terletak pada File-file CHR tersebut.
- ***.BGI**
Borland Grafik Interface → berguna untuk mengaktifkan mode berbentuk Grafik. Biasanya cukup satu file saja yang di butuhkan yaitu EGAVGA.BGI
- ***.TPU**
yaitu Turbo Pascal Unit yang berguna untuk unit pembantu misal pada saat kita menggunakan Mode Grafik maka kita memerlukan File Graph.TPU

MENGUNAKAN PASCAL

Pascal yang kita gunakan disini ada beberapa jenis bisa Pascal versi 5, atau Pascal versi 7 atau yang lainnya. Tetapi setiap versi tersebut nama file Executable tetap sama yaitu TURBO. Jadi bila kita menggunakan Pascal cukup cari File Turbo.EXE dan Double Click file tersebut (Windows). Jika kita menggunakan Dos maka ketikkan :

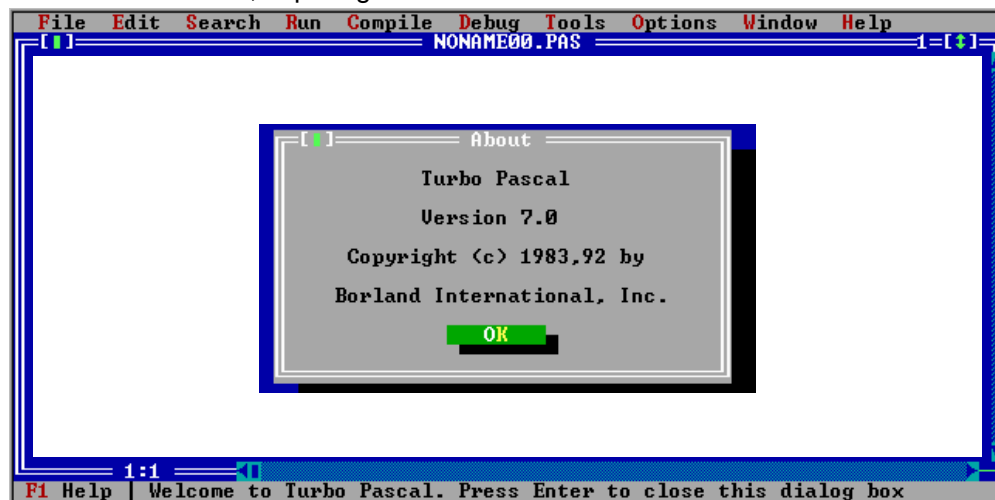
```
C:\TURBO [Enter]
```

Jika ada terdapat direktori maka anda harus masuk dahulu ke direktori tersebut.

```
Contoh C:\Pascal5\TURBO [Enter]
```

```
C:\TP\BIN\TURBO [Enter]
```

Setelah anda ketikkan perintah tersebut atau double click file Turbo.Exe maka akan muncul suatu Editor Pascal, seperti gambar di bawah ini.



Editor di atas merupakan tampilan dari Pascal Versi 7.0 untuk menghilangkan kotak About cukup tekan **ESC**. Untuk pascal versi 5 perintah-perintah menunya hampir sama.

Tetapi akan lebih menguntungkan jika kita menggunakan versi yang lebih tinggi, dikarenakan banyak menawarkan fasilitas-fasilitas yang sangat baik.

MEMBUAT PROGRAM BARU

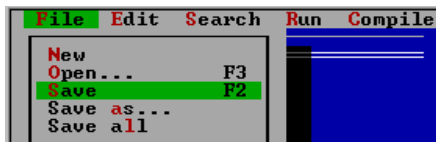
Pada memasuki pascal secara otomatis akan terbentuk file baru dengan Default Name yaitu : NONAME00.Pas. Untuk membuat program baru cukup pilih menu FILE dengan cara tekan *Alt + F*, pilih *New*.



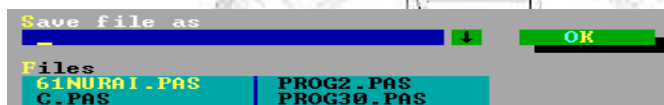
Sehingga muncul kembali editor Noname01.Pas (jika kita tidak menutup Noname00.Pas)

MENYIMPAN PROGRAM

Setelah program selesai diketik, kita bisa menyimpan program kita ke Disk atau Hardisk dengan memilih menu FILE dengan cara tekan *Alt + F*, pilih *Save F2*.



Atau kita bisa langsung dengan menekan **F2** tanpa memilih menu FILE. Setelah kita pilih *Save* atau tekan **F2** maka akan keluar dialog *Save*.



Untuk menyimpan ke Hardisk cukup ketikkan saja nama filenya Seperti : **Test** kemudian tekan [Enter]

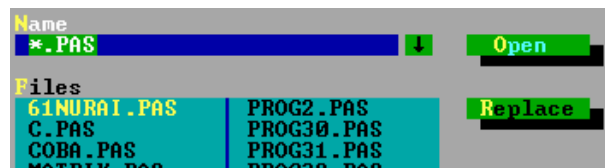
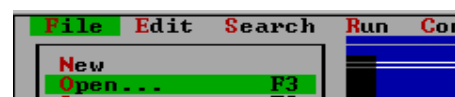
Jika ingin menyimpan ke Disk cukup ketikkan A: sebelum nama File seperti **A:Test** kemudian tekan [Enter]

Notes :

- Save As... → berguna untuk menyimpan dengan nama yang baru.
- Save All → berguna untuk menyimpan seluruh program yang telah terbuka (pada versi 7.0)

MEMBUKA PROGRAM

Untuk membuka program yang telah kita ketikkan sebelumnya yaitu dengan cara memilih menu FILE (*Alt+F*) kemudian pilih *Open...F3* atau cukup dengan menekan **F3** dan akan keluar dialog *Open*

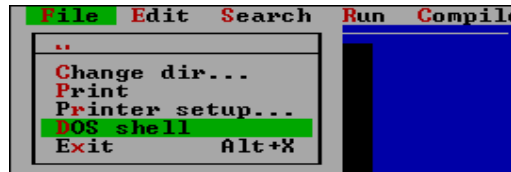


Untuk memilih file cukup tekan [Enter] pada dialog open dan kemudian kita bisa memilih dengan menekan tombol Panah. Atau kita bisa mengetikkan nama file jika kita mengetahui file yang akan kita buka.

Untuk membuka file yang berada di Disk cukup ketikkan **A: *.Pas** [Enter] (kita tinggal menyebutkan Nama Direktori)

KELUAR KE LAYAR DOS

Untuk keluar ke layar DOS cukup pilih DOS Shell pada menu FILE



Jika kita telah berada pada layar DOS maka untuk kembali ke Pascal cukup ketikkan Exit.

KELUAR DARI PASCAL

Untuk keluar dari pascal cukup tekan **Alt + X**. atau dengan menekan memilih *Exit* pada menu File. Pada Pascal 5 kita memilih Quit dari Menu(Alt + Q) .

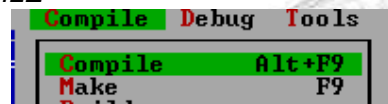
Notes :

- Change Dir → untuk mengganti direktory Aktif.
- Print → untuk mencetak program yang sedang aktif di layar Editor.
- Printer Setup → untuk mengatur setting Printer.

KOMPILASI PROGRAM

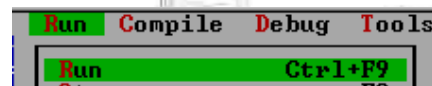
Setelah program di ketik, sebelum di jalankan maka kita harus melakukan Compile terhadap Program dengan cara :

Menekan **Alt + F9** (bisa **F9** saja) atau memilih perintah *Compile* pada menu **COMPILE**



MENJALANKAN PROGRAM

Jika pada saat kita mengcompile program tidak muncul pesan kesalahan, maka kita bisa menjalankan program dengan perintah *Run* pada menu *RUN*. Atau cukup menekan **Ctrl + F9**.



MEMBUAT EXECUTABLE

Untuk bisa menggunakan program pada media DOS atau Windows, maka kita perlu untuk membuat Executable program. Dengan cara.

1. Rubah Destination Memory menjadi Destination Disk
Caranya cukup pilih menu **COMPILE** kemudian pilih Destination Memory.
Dengan sendirinya akan berubah menjadi **DISK**.
2. Kemudian anda tinggal menekan **F9**. Perhatikan proses Compile Destination menjadi Disk.