RECORD

# Definisi Record

- Record adalah jenis tipe data terstruktur yang berisi beberapa data, yang masing-masing dapat berlainan tipe.
- Suatu tipe record dideklarasikan dengan bentuk sebagai berikut :

```
RECORD
Daftar_field_1 : tipe_1;
Daftar_field_2 : tipe_2;
. . .
daftar_field_n : tipe_n;
END
```

- Masing-masing *daftar_field* dapat berupa satu atau beberapa nama pengenal dan masing-masing dinamakan *field*. Bila *daftar_field* berisi lebih dari satu field , antar field perlu dipisahkan dengan koma. Masing-masing tipe dapat berupa tipe data apa saja termasuk *array*.

# Deklarasi Record

- Berikut contoh pendeklarasian record :

```
Type
    RecBarang = Record
                Nama     : String;
                Kualitas  : Char;
                Harga     : LongInt
                End;

Var
    Barang : RecBarang;
```

- Dengan mendeklarasikan seperti di samping, Barang akan mengandung tiga buah *field*, yaitu :
  - Nama,
  - Kualitas,
  - Harga.

# Cara Mengakses Field

- Field dari suatu *record* diakses dengan bentuk :

  *Variabel.field*

- Sebagai contoh :

  Barang.Nama

- Berarti "*field* Nama dari variabel record bernama Barang".

- Contoh penugasan nilai ke field tersebut :

  Barang.Nama := ’Ubin TISKA 20x20’;

- Dengan cara seperti di atas, field Nama dari record Barang berisi string ’Ubin TISKA 20x20’.

- Isi dari suatu field ditampilkan dengan menggunakan **Write** atau **Writeln**.
- Contoh :

  Writeln (Barang.Nama);

- Merupakan perintah untuk menampilkan isi field Nama dari record Barang.

Contoh program yang memberikan gambaran pendeklarasian record, pengisian terhadap field-field serta menampilkan isi masing-masing field dapat dilihat di bawah ini.

```
Program Rec1;
Uses crt;
Type
    RecBarang = Record
                Nama      : String[25];
                Kualitas  : Char;
                Harga     : LongInt
    End;
Var
    Barang : RecBarang;          {variabel bertipe record}
Begin
Clrscr;
{Penugasan nilai terhadap field-field}
Barang.Nama := 'Ubin TISKA 20x20';
Barang.Kualitas := 'A';
Barang.Harga := 14000;

{menampilkan isi field}
writeln ('Nama Barang   : ', Barang.Nama);
writeln ('Kualitas      : ', Barang.Kualitas);
writeln ('Harga         : ', Barang.Harga);
Readln
End.
```

- Hasil program :

  Nama Barang : Ubin TISKA 20x20
  Kualitas : A
  Harga : 14000

# Penugasan Antar Record

- Jika record R1 dan R2 bertipe sama dan masing-masing memiliki F1, F2, dan F3, maka penugasan :

  ```
  R1 := R2;
  ```

- diperkenankan. Pernyataan di atas merupakan penyederhanaan dari sederetan pernyataan berikut :

  ```
  R1.F1 := R2.F1;
  R1.F2 := R2.F2;
  R1.F3 := R2.F3;
  ```

- Untuk lebih jelasnya, tulislah program berikut dan cobalah menjalankannya.

# Penugasan Antar Record

```
Program Rec2;
Uses crt;
Type
        RecBarang = Record
                Nama      : string[25];
                Kualitas  : car;
                Harga     : longInt
        End;
Var
        Barang1, Barang2 : RecBarang;
        {variabel bertipe record}
Begin
Clrscr;
{penugasan nilai terhadap field-field}
Barang1.Nama := 'Ubin TISKA 20x20';
Barang1.Kualitas := 'A';
Barang1.Harga := 14000;

{menyalin record}
Barang2 := Barang1;

Menampilkan isi field}
Writeln ('Nama Barang          : ', Barang2.Nama);
Writeln ('Kualitas     : ', Barang.Kualitas);
Writeln ('Harga                : ', Barang.Harga);
Readln
End.
```

# Penugasan Antar Record

- Dengan adanya penugasan

  Barang2 := Barang1;

- maka semua field pada record Barang2 akan berisi recrod Barang1. Hasil dari program di atas :

  Nama Barang : Ubin TISKA 20x20

  Kualitas : A

  Harga : 14000

# Record Di Dalam Record

- Mungkin saja sebuah record berisi record. Sebagai gambaran hal ini, perhatikan deklarasi berikut :

```
RecTanggal = Record
                Tanggal,
                Bulan,
                Tahun     :Integer
        End;
RecPegawai = Record
                Nomor     : LongInt;
                Nama      : String [35];
                TglLahir  : RecTanggal;
                Gaji      : LongInt
        End;
```

# Record Di Dalam Record

- Tampak bahwa tipe record bernama RecPegawai berisi record yang lain (RecTanggal).
- Hal yang menarik yang perlu diperhatikan adalah cara mengakses field seperti Tanggal, Bulan dan Tahun. Notasi yang diperlukan adalah sebagai berikut.

  Nama_variabel.TglLahir.Tanggal

  Nama_variabel.TglLahir.Bulan

  Nama_variabel.TglLahir.Tahun

```
Program Rec3;
Uses crt;
Type
    RecTanggal = Record
                 Tanggal,
                 Bulan,
                 Tahun    : Integer
    End;
RecPegawai = Record
                 Nomor  : LongInt;
                 Nama   : string [35];
                 TglLahir : RecTanggal;
                 Gaji     : longInt
    End;
Var
    DataPeg : RecPegawai;   {variabel betipe
record}
Begin
Clrscr;
Penugasan nilai terhadap field-fiedl}
DataPeg.Nomor := 56789;
DataPeg.Nama   := 'Badu';
DataPeg.TglLahir.Tanggal := 24;
DataPeg.TglLahir.Bulan := 12;
DataPeg.TglLahir.Tahun := 1972;
DataPeg.Gaji := 750000;

{menampilkan isi field}
Writeln ('Nama Pegawai  : ', DataPeg.Nama);
Writeln ('Tanggal Lahir    : ',
DataPeg.TglLahir.Tanggal,
     '/', DataPeg.TglLahir.Bulan,
     '/', DataPeg.TglLahir.Tahun);
Readln
End.
```

- Hasil dari program di atas adalah sebagai berikut :

Nama Pegawai : Badu
Tanggal Lahir : 24 / 12 / 1972

FILE

# Definisi File

- Dalam kasus-kasus pemrograman tertentu kita sering disudutkan untuk menggunakan file sebagai media yang digunakan untuk menyimpan data-data, baik berupa data input (untuk masukan) maupun sebagai data output (untuk keluaran).
- Sebagai contoh, apabila kita menggunakan sistem operasi Microsoft Windows maka kita akan menemukan file sistem seperti *autoexec.bat*, *config.sys*, *system.ini* dan lainnya.
- File tersebut sebenarnya digunakan untuk menyimpan data-data secara terpisah sehingga nilainya dapat diubah dengan mudah sesuai dengan kebutuhan.

# Definisi File

- Berdasarkan prosedur yang ada, proses pengaksesan file di dalam bahasa Pascal terdiri dari empat tahap, yaitu :
  1. Menghubungkan file fisik yang akan dibuka atau dibuat dengan variabel file
  2. Membuka file
  3. Melakukan operasi file (membaca atau menulis)
  4. Menutup file

# Variabel File

- Sebelum file dapat dibuka atau dibuat oleh program, kita harus menghubungkannya dengan variabel file terlebih dahulu.
- Variabel file merupakan peralatan logik yang digunakan sebagai perantara dalm mentransfer atau membaca data dari atau ke sebuah file fisik yang tersimpan di dalam disk.
- Dalam bahasa Pascal, variabel seperti ini dianggap sebagai variabel biasa yang dideklarasikan dengan tipe file tertentu.
- Cara yang harus dilakukan untuk dapat menghubungkan file fisik dengan variabel file adalah dengan menggunakan prosedur **Assign**, yang memeiliki bentuk umum sebagai berikut.

  Procedure Assign (NamaVariabelFile, NamaFileFisik);

- **NamaVariabelFile** merupakan nama variabel yang sebelumnya harus dideklarasikan terlebih dahulu, sedangkan **NamaFileFisik** merupakan nama file yang akan dibuka atau dibuat. Perlu diperhatikan bahwa nama file tersebut harus dituliskan secara lengkap beserta lokasi atau path-nya.
- Sebagai contoh, apabila kita ingin membuka file dengan nama CONTOH.TXT yang terdapat di dalam direktori COBA di drive D, maka kode yang harus dituliskan adalah sebagai berikut.

```
Var
F:TextFile; {mendeklarasikan variabel file denagn nama F}
Begin
Assign (F, 'D:\COBA\CONTOH.TXT');
. . .
end.
```

- Sedangkan apabila file yang akan kita akses berada dalam satu direktori dengan program (file eksekusi) yang kita buat, maka kita tidak perlu menuliskan lokasi atau path-nya. Artinya kita hanya perlu untuk menuliskan nama file-nya saja seperti berikut.

```
Assign (F, ' CONTOH.TXT');
```

- Pada kode di atas, file CONTOH.TXT akan dihubungkan dengan variabel F. Selanjutnya yang akan dimanipulasi di dalam program adalah variabel F. Namun karena variabel tersbut sebenarnya menunjuk ke file fisik, maka setiap perubahan yang terjadi di dalam variabel F tentu akan mempengaruhi isi dari file CONTOH.TXT.

# Membuka File

- Setelah file tersebut dihubungkan dengan variabel file, maka langkah selanjutnya yang perlu dilakukan adalah membuka file. Dalam bahasa Pascal, terdapat tiga buah prosedur yang dapat digunakan untuk membuka file, yaitu **Rewrite**, **Reset** dan **Append**.
  a) Prosedur Rewrite

     Prosedur ini digunakan untuk membuka file yang sama sekali belum terdapat di dalam disk. Artinya, di sini kompiler akan melakukan proses pembuatan file baru sekaligus membuka file tersebut. Bentuk umum dari prosedur Rewrite adalah sebagai berikut.

```
Procedure Rewrite(NamaVariabelFile[:File;
                  UkuranRecord : Word]);
```

# Membuka File

- Bila file yang dibuka berupa file teks, maka file tersebut akan bersifat *writeonly* atau tidak dapat dibaca, artinya operasi yang diizinkan untuk file tersebut hanyalah operasi tulis.
- **UkuranRecord** merupakan ekspresi yang bersifat opsional yang hanya disertakan apabila file yang dibuka merupakan file tanpa tipe.
- Nilai tersebut akan digunakan sebagai ukuran record dalam transfer data. Apabila dihilangkan, maka UkuranRecord ini akan dianggap bernilai 128 byte.

# Membuka File

- Untuk lebih memahami penggunaan prosedur ini, perhatikan contoh program sederhana ini.

```
Program ContohRewrite;
Var
F : TextFile;
Begin
Assign (F, 'D:\CONTOH.TXT');
Rewrite (F);
End.
```

- Jalankan program tersebut dan lihatlah drive D pada komputer anda, maka di situ pasti akan tercipta file baru dengan nama CONTOH.TXT dimana isinya kosong.
- Apabila anda bereksperimen untuk membuka file yang sudah ada sebelumnya dengan menggunakan prosedur Rewrite maka isi dari file tersebut akan ditimpa dengan isi file baru.
- Satu hal lagi yang perlu diperhatikan adalah bahwa setelah pemanggilan prosedur Rewrite, fungsi EOF (*end-of-file*) akan selalu menghasilkan nilai true.

# Membuka File

b) Prosedur Reset

Prosedur ini digunakan untuk membuka file yang sebelumnya sudah ada di dalam disk. Hal ini bertujuan untuk membaca isi dari file tersebut untuk kemudian dimanipilasi sesuai dengan kebutuhan. Perlu sekali untuk diperhatikan bahwa apabila file yang dibuka merupakan file teks, maka variabel file akan bersifat *read only* (hanya dapat dibaca). Berikut ini prototype dari prosedur Reset.

```
Procedure Reset (NamaVariabelFile [: file; UkuranRecord: word]);
```

# Membuka File

- UkuranRecord merupakan ekspresi yang bersifat operaional yang hanya disertakan apabila file yang dibuka merupakan file tanpa tipe.
- UkuranRecord sendiri berfungsi untuk menentukan ukuran record dalam proses transfer data.
- Apabila nilai tersebut dihilangkan, maka nilainya akan dianggap 128 byte.
- Sebagai contoh, kita telah memiliki file SAMPLE.TXT yang disimpan di rive D dan isinya seperti berikut.

Teknik Pemrograman Pascal
Oleh : Budi Raharjo
Tahun : 2005
Penerbit : INFORMATIKA

# Membuka File

- Selanjutnya kita ingin membaca isi file tersebut dan ditampilkan di layar monitor, maka kita dapat menggunakan prosedur Reset seperti yang tampak pada kode berikut.

```
Program ContohReset;
Uses crt;
Var
F : TextFile;
Teks : string;
Begin
Assign (F, 'D :\SAMPLE.TXT');
Reset (F);
While not eof (F ) do begin
Readln (F, teks) ;      {membaca data dari file dan mengisikannya ke variabel teks}
Writeln (teks);         menampilkan variabel teks ke layar}
Close (F);
Readln;
End.
```

# Membuka File

- Sekarang coba jalankan program tersebut dan akan melihat bahwa seluruh isi dari file SAMPLE.TXT akan ditampilkan di layar monitor.
- Apabila kita menggunakan prosedur Reset pada file yang belum terdapat di dalam disk maka kompiler akan menampilkan pesan kesalahan pada saat program dijalankan.

c) Prosedur Append

Prosedur ini digunakan untuk menambahkan isi file yang sebelumnya sudah terdapat di dalam disk. Isi yang ditambahkan selalu berada pada bagian akhir dari isi file yang sudah ada sebelumnya. Dalam menggunakan prosedur ini kita tidak perlu memanggil prosedur Rewrite maupun reset karena prosedur Append secara otomatis akan membuka file yang telah dihubungkan dengan variabel file. Namun sebagai catatan bagi Anda bahwa prosedur Append hanya dapat digunakan apabila file yang dibuka merupakan file teks. Berikut ini bentuk umum dari prosedur Append.

```
Procedure Append (NamaVariabelFile: Text ) ;
```

- Apabila anda membuka file teks yang belum ada didalam diks dengan menggunakan prosedur Append, maka akan terjadi kesalahan pada saat program sedang berjalan (*run-time*).
- Setiap pemanggilan prosedur Append, file bersifat *write-only* (hanya dapat ditulis) dan posisi file akan diset pada bagian akhir baris file.
- Untuk dapat lebih memahaminya, buatlah file teks (*.TXT) dengan program teks editor yang tersedia, misalnya Notepad, kemudian isikan teks berikut ke dalamnya.

  Ini adalah data yang dituliskan pada baris pertama.

  Ini adalah data yang dituliskan pada baris pertama.

- Sebagai contoh, file tersebut disimpan dengan nama TEST.TXT pada drive D di dalan direktori COBA.
- Selanjutnya kita ingin menamnbahkan dua buah baris data ke dalam file di atas, maka kita dapat melakukannya melalui kode program dibawah ini.

```
Program ContohAppend;
Uses
Crt;
Var
F: TextFile;
Begin
Assign (F,'D:\COBA\TEST.TXT') ;
Append (F) ;   {Membuka file untuk di tambah isinya}

{Menambahkan data ke dalam file}
writeln (F,'Ini adalah data pertama yang ditambahkan') ;
writeln (F,'Ini adalah data kedua yang ditambahkan') ;

close(F) ;
end.
```

Hasil :

```
Ini adalah data yang dituliskan pada baris pertama
Ini adalah data yang dituliskan pada baris kedua
Ini adalah data pertama yang ditambahkan
Ini adalah data kedua yang ditambahkan
```