

## MODUL 11

### RECORD

#### TUJUAN

1. Praktikan dapat mengerti dan memahami penggunaan *record/struct* dalam bahasa pemrograman C.
2. Praktikan dapat mengerti kegunaan *record/struct* dalam efisiensi pengkodean.
3. Praktikan dapat mengerti cara mengakses dan memodifikasi *record/struct*.
4. Praktikan dapat mengerti cara membuat relasi antara satu *record/struct* dan *record/struct* lain.
5. Praktikan dapat mengerti perbedaan pembuatan suatu *record/struct* dengan menggunakan *typedef* dan tanpa menggunakan *typedef*.

#### DASAR TEORI

*Record* atau *struct* merupakan sebuah tipe data bentukan yang berisi kumpulan atribut atau tipe data – baik berbeda-beda maupun sama – dalam suatu nama yang sama, dan memiliki kaitan antara satu sama lain, berbeda dengan *array* yang hanya dapat diisi oleh satu tipe data yang sama. Dengan menggunakan *record/struct*, kita dapat menyimpan beberapa atribut dengan tipe data yang berbeda di bawah suatu nama.

Dalam penerapannya, masing-masing item di dalam *record* disebut *field* yang berbeda tipe data. Jadi, *record/struct* terdiri dari kumpulan *field* yang dapat berbeda tipe datanya. Biasanya suatu *record* berisi beberapa *field* untuk sebuah subjek tertentu. Setiap *field* dapat menyimpan tipe data dasar (*int*, *float*, *char*) maupun tipe data bentukan (*struct/record* lain maupun tipe data yang sudah didefinisikan sebelumnya).

#### Mengapa kita membutuhkan *record* atau *struct*?

*Contoh kasus: Menyimpan data mahasiswa*

```
string nama;  
float ipk;  
int umur;
```

Untuk menyimpan data 1 mahasiswa, hal ini bisa dilakukan, tetapi kode kita terlihat kurang elegan karena variabel-variabel tersebut tidak dapat diketahui dengan pasti “pemiliknya”. Apakah variabel “nama” di samping adalah nama orang tua mahasiswa? “umur” siapakah yang dimaksud di samping?

```
string nama_mhs;  
float ipk_mhs;  
int umur_mhs;
```

```
string nama_mhs_1;  
float ipk_mhs_1;  
int umur_mhs_1;  
  
string nama_mhs_2;  
float ipk_mhs_2;  
int umur_mhs_2;  
  
string nama_mhs_3;  
float ipk_mhs_3;  
int umur_mhs_3;
```

Alternatifnya, kita bisa menambahkan *suffix* “\_mhs” pada setiap variabel agar dapat dikategorikan dengan lebih mudah, tetapi dengan demikian nama variabel menjadi semakin panjang, kemudian bagaimana cara kita menyimpan data lebih dari 1 mahasiswa? Solusi *naïve* untuk menyimpan data lebih dari 1 mahasiswa adalah seperti di samping, namun semakin banyak mahasiswa yang kita tambahkan, semakin banyak variabel yang dipakai, akhirnya kode kita menjadi semakin panjang, tidak efektif, dan tidak efisien.

Bagaimana jika semua data di atas kita satukan menggunakan *record/struct*?

```
typedef struct {  
    string nama;  
    float ipk;  
    int umur;  
} Mahasiswa;  
  
void main() {  
    Mahasiswa M1, M2, M3;  
}
```

Dari potongan kode di atas, kita sudah mengatasi masalah-masalah yang dipaparkan di awal.

Kita telah membuat suatu record bernama “Mahasiswa”, di dalamnya berisikan *field-field* nama, IPK, dan umur – dan tidak perlu lagi kita menambahkan suffix “\_mhs” di belakang karena kita sudah mengetahui bahwa *field* tersebut adalah “kepunyaan” dari “Mahasiswa”.

Kemudian untuk menambahkan 3 orang mahasiswa, kita cukup mendeklarasikan tipe data “Mahasiswa” di dalam fungsi/prosedur yang kita inginkan, kemudian memberi nama variabel-variabel tersebut, sama seperti tipe data lain.

Bagaimana cara mengakses dan memodifikasi *record* yang telah dibuat?

Field-field di dalam *record* dapat kita akses dengan memanggil variabel *record*, diikuti tanda titik '.', diakhiri dengan nama field di dalam record tersebut.

```
TipeDataRecord record;  
record.namaField;
```

Beberapa cara mengakses field dalam record “Mahasiswa” yang telah kita buat tadi adalah:

```
// Mencetak isi field `nama` dari struct `Mahasiswa` bernama `M1`:  
printf("%s", M1.nama);  
  
// Mencetak isi field `ipk` dari struct `Mahasiswa` bernama `M2`:  
printf("%.2f", M2.ipk);  
  
// Mencetak isi field `umur` dari struct `Mahasiswa` bernama `M3`:  
printf("%d", M3.umur);
```

Beberapa cara memodifikasi/mengubah isi field dalam record “Mahasiswa” yang telah kita buat tadi adalah:

```
// Mengubah isi field `nama` dari struct `Mahasiswa` bernama `M1`:  
strcpy(M1.nama, "Nama Baru");  
  
// Mengubah isi field `ipk` dari struct `Mahasiswa` bernama `M2`:  
M2.ipk = 3.81;  
  
// Mengubah isi field `umur` dari struct `Mahasiswa` bernama `M3`:  
M3.umur = 21;
```

Beberapa operasi lain yang bisa dilakukan antara lain:

```
// Menggunakan if-else berdasarkan konten `nama` dari struct  
`Mahasiswa` bernama `M1`:  
if(strcmp(M1.nama, "Jolly") == 0) {  
    printf("Pemegang Modul 11 - Record");  
} else if(strcmpi(M1.nama, "Henry") == 0) {  
    printf("Pemegang Modul 3 - Tipe Data, Penamaan, dan Sekuens");  
} else if(strcmpi(M1.nama, "Vano") == 0) {  
    printf("Pemegang Modul 5 - Perulangan 1");  
} else {  
    printf("Oops... nama belum diinput!");  
}
```

```
// Periksa apakah seorang mahasiswa termasuk cumlaude atau
tidak:
if(M2.ipk > 3.99) {
    printf("Mahasiswa Summa Cumlaude");
} else if(M2.ipk > 3.79) {
    printf("Magna Cumlaude");
} else if(M2.ipk > 3.50) {
    printf("Cumlaude");
} else {
    printf("Tidak Cumlaude");
}

// Periksa (perkiraan kasar) tahun lahir mahasiswa
int tahunLahir = 0;
tahunLahir = 2022 - M3.umur;
```

Bagaimana kalau ada *struct* di dalam *struct*?

Contohnya di sini: kita akan membuat record Mahasiswa dan DosenPembimbing.

```
// KONTEKS:
// 1 mahasiswa sudah pasti memiliki 1 dan hanya 1 dosen
pembimbing.

// Terlebih dahulu, buatlah record/struct DosenPembimbing - karena
perlu diketahui compiler sebelum dipanggil di dalam Mahasiswa.
typedef struct {
    string nama;
    string nidn;
} DosenPembimbing;

// Kemudian buatlah record Mahasiswa, jangan lupa menyertakan
record DosenPembimbing yang sudah dibuatkan sebelumnya.
typedef struct {
    string nama;
    float ipk;
    int umur;
    DosenPembimbing DP;
} Mahasiswa;

void main() {
    Mahasiswa M;
}
```

Dari contoh di atas, maka cara mengakses struct DosenPembimbing tersebut adalah:

```
// KONTEKS
// Sudah ada record `Mahasiswa` dengan nama variabel `M` (seperti
gambar sebelumnya)

// Membuat salinan dari `M`.`DP` (field DosenPembimbing dari
Mahasiswa tersebut) di variabel baru bertipe `DosenPembimbing`
dengan nama `dosbimMhs`
DosenPembimbing dosbingMhs = M.DP;

// Memodifikasi nilai dari variabel `dosbimMhs` tanpa mengubah nilai
yang ada di `M`.`DP`
strcpy(dosbimMhs.nidn, "09.08.82");

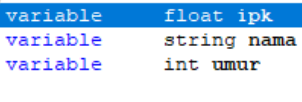
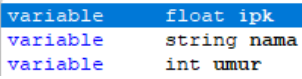
// Memodifikasi nama dosen pembimbing secara langsung:
strcpy(M.DP.nama, "Nama Baru");

// Mendapatkan nama dosen pembimbing yang dimiliki `Mahasiswa`
dengan nama variabel `M`:
string namaDosbim;
strcpy(namaDosbim, M.DP.nama);
printf("%s", namaDosbim);
```

Kenapa kita menggunakan **typedef**? Apakah bisa kita membuat record/struct tanpa typedef?

Tentu bisa, mari perhatikan beberapa contoh di bawah:

Tindakan	Tanpa typedef	Dengan typedef
Pen- definisian struct	<pre>struct {     string nama;     float ipk;     int umur; } Mahasiswa;</pre>	<pre>typedef struct {     string nama;     float ipk;     int umur; } Mahasiswa;</pre>
	Membuat variabel bernama "Mahasiswa" dengan tipe data <i>struct</i> .	Membuat tipe data buatan bernama "Mahasiswa" bertipe <i>struct/record</i> .

Pen-deklarasian struct	<pre>// ... Mahasiswa.umur = 21; strcpy(     Mahasiswa.nama,     "Jolly Hans Frankle" );</pre> <p>Setiap variabel struct hanya dapat dipakai sekali saja.</p>	<pre>Mahasiswa M1, M2, M3; M1.umur = 21; strcpy(     M2.nama,     "Jolly Hans Frankle" );</pre> <p>Dikarenakan Mahasiswa merupakan tipe data buatan, kita dapat menggunakannya lebih dari satu kali.</p>
Pemang-gilan field	<pre>struct {     string nama;     float ipk;     int umur; } Mahasiswa;  void main() {     Mahasiswa.</pre>  <p>Dikarenakan Mahasiswa merupakan sebuah variabel, sehingga kita dapat langsung menggunakannya. Variabel tersebut sudah terdeklarasi di bagian header program.</p>	<pre>typedef struct {     string nama;     float ipk;     int umur; } Mahasiswa;  void main() {     Mahasiswa M1, M2, M3;     M1.</pre>  <p>Dikarenakan Mahasiswa merupakan sebuah tipe data, maka kita harus mendeklarasikan variabel M1, M2, dan/atau M3 terlebih dahulu.</p>

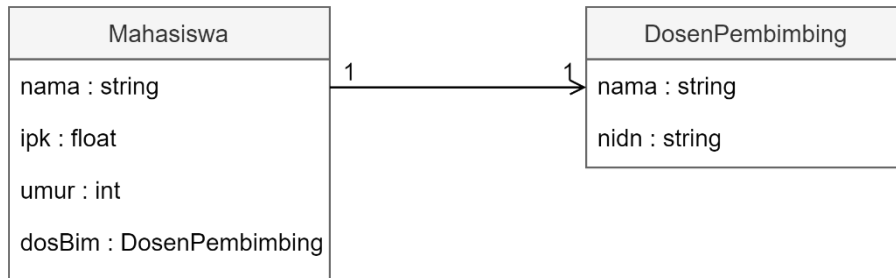
*Sebuah record hanya dapat menampung 1 data struct saja. Jika ingin menyimpan record sebagai array, maka kita dapat menggunakan array of record.*

*Di modul ini, kita belum bisa menggunakan array of record, karena akan dipelajari di modul 12.*

## GUIDED

Universitas Dummy Raya ingin melakukan pengelolaan data mahasiswa dan dosen pembimbingnya. Buatlah sistemnya dengan ketentuan sebagai berikut:

1. Buatlah dua buah tipe data bentukan (record) dengan ketentuan sebagai berikut:



2. Inisialisasilah struct tersebut dengan data kosong sebelum digunakan (inisialisasi dilakukan sebelum memasuki menu program, tidak perlu menu khusus untuk inisialisasi).

*Contoh:*

```
Mahasiswa.nama = "" ; (empty string)
Mahasiswa.ipk = 0;
Mahasiswa.umur = 0;
Mahasiswa.DosenPembimbing.nama = "";
Mahasiswa.DosenPembimbing.nidn = "";
```

3. Buatlah menu 1 yang berfungsi untuk menginput data mahasiswa dan/atau dosen pembimbing.

**Menu ini hanya dapat diakses apabila data mahasiswa kosong/tidak diisi.**

User dapat menginput data mahasiswa sesuai dengan urutan field pada diagram di atas. Setelah mengisi data mahasiswa (**sebelum mengisi data dosen pembimbing**), tambahkanlah konfirmasi “Apakah sudah ada dosen pembimbing? [Y/N]”. Apabila user menjawab “Y”, maka lanjutkanlah dengan pengisian data dosen pembimbing.

*Pastikan ada error handling apabila nama mahasiswa atau dosen pembimbing kosong.*

4. Buatlah menu 2 yang berfungsi untuk menampilkan data mahasiswa.

**Menu ini hanya dapat diakses apabila data mahasiswa sudah diisi.**

Tampilkanlah data mahasiswa dengan urutan yang sesuai pada diagram di atas. Tampilkanlah terlebih dahulu data mahasiswa, sebelum menampilkan data dosen pembimbing apabila ada.

5. Buatlah menu 3 yang berfungsi untuk mengubah data mahasiswa/dosen pembimbing.

*Menu ini hanya dapat diakses apabila data mahasiswa sudah diisi.*

Di dalam menu ini, tambahkan dua buah submenu dengan ketentuan:

- 1) Menu 1: mengubah data mahasiswa

Gunakan menu ini untuk mengubah **hanya data mahasiswa, bukan dosen pembimbing**.

- 2) Menu 2: mengubah data dosen pembimbing

Gunakan menu ini untuk mengubah data dosen pembimbing. Apabila data dosen pembimbing belum diisi sebelumnya, menu ini berfungsi untuk menambah data dosen pembimbing.

Setelah mengisi salah satu data di atas, kembalilah ke menu utama.

*Pastikan ada error handling apabila nama mahasiswa atau dosen pembimbing kosong.*

6. Buatlah menu 4 yang berfungsi untuk menghapus data mahasiswa.

*Menu ini hanya dapat diakses apabila data mahasiswa sudah diisi.*

Sebelum menghapus, tambahkanlah konfirmasi apakah user benar-benar ingin menghapus data. Apabila iya, kosongkanlah data mahasiswa dan dosen pembimbing seperti pada inisialisasi awal.

7. Tambahkan menu keluar yang berfungsi untuk keluar program. Tambahkan identitas kamu dengan format “<Nama> – <NPM lengkap> – Praktikum Dasar Pemrograman <Kelas>”.

*Menu ini dapat diakses entah data mahasiswa sudah diisi maupun belum diisi.*

Contoh identitas: “Jolly Hans Frankle – 200710932 – Praktikum Dasar Pemrograman D”

*Beberapa prosedur dan fungsi yang digunakan di jawaban GD:*

```
void initMahasiswa(Mahasiswa *M);
void inputMahasiswa(Mahasiswa *M);
void inputDosenPembimbing(DosenPembimbing *DP);
int tahunLahirMahasiswa(int umur);
int isMahasiswaEmpty(Mahasiswa M);
int isDosenPembimbingEmpty(DosenPembimbing DP);
void tampilMahasiswa(Mahasiswa M);
```

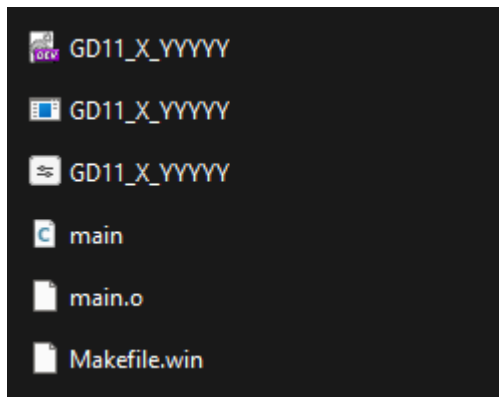


### **Ketentuan Pengerjaan:**

Dikerjakan secara mandiri, tanpa copy-paste dari teman, karena pemahaman akan guided akan sangat berguna di unguided, tugas, bahkan sampai mata kuliah Informasi dan Struktur Data (semester 3).

Guided dikumpulkan ke situs kuliah dengan nama: **GD11\_X\_YYYY**

Dikumpulkan secara lengkap di dalam satu folder:



Kemudian folder tersebut di-ZIP (**ingat, di-ZIP, bukan di-RAR**)!

*Keterangan format penamaan:*

**X** = kelas

**Y** = 5 digit terakhir NPM

Nilai GD dikurangi 10 apabila format penamaan salah atau isi folder tidak sesuai (minimal ada .dev dan .c, dan bisa dibuka dengan baik).

### **Persiapan Unguided:**

1. Pelajari fungsi dan prosedur yang ada di guided, karena beberapa prosedur/fungsi tersebut dapat digunakan kembali di unguided dan tugas.
2. Pelajari kembali fungsi dan prosedur secara umum.
3. Pelajari kembali array dan penggunaan perulangan untuk iterasi array.
4. Unguided wajib dikumpulkan menggunakan project Dev-C++.

*Eits... jangan dulu ke jawaban guided! Coba dulu mandiri!*

### Jawaban Guided:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <stdbool.h>
5 typedef char string[50];
6
7 typedef struct {
8     string nama;
9     string nidn;
10 } DosenPembimbing;
11
12 typedef struct {
13     string nama;
14     float ipk;
15     int umur;
16     DosenPembimbing dosBim;
17 } Mahasiswa;
18
19 void initMahasiswa(Mahasiswa *M);
20 void inputMahasiswa(Mahasiswa *M);
21 void inputDosenPembimbing(DosenPembimbing *DP);
22 int tahunLahirMahasiswa(int umur);
23 int isMahasiswaEmpty(Mahasiswa M);
24 int isDosBimEmpty(DosenPembimbing DP);
25 void tampilMahasiswa(Mahasiswa M);
26
27 void main() {
28     Mahasiswa M;
29
30     int menu;
31
32     // (1) Inisialisasi data mahasiswa dan dosen pembimbing
33     initMahasiswa(&M);
34
35     do {
36         system("cls");
37         printf("--- Guided Record: Mahasiswa x Dosen Pembimbing ---");
38         printf("\n[1] Input Data");
39         printf("\n[2] Tampilkan Data");
40         printf("\n[3] Edit Data");
41         printf("\n[4] Hapus Data");
42         printf("\n[0] Exit");
43         printf("\n>> "); scanf("%d", &menu);
44
45         switch(menu) {
46             case 1:
47                 if(isMahasiswaEmpty(M)) {
48                     printf("\n\t--- Input Data ---\n");
49
50                     // Apabila data mahasiswa kosong/tidak diisi, menu ini dapat diakses
51                     // Input data mahasiswa menggunakan prosedur inputMahasiswa(...)
52                     inputMahasiswa(&M);
53
54                     // Setelah input, tanyakan apakah user ingin menginput data dosen pembimbing
55                     string confirm;
56                     printf("\n\tApakah sudah ada dosen pembimbing? [Y/N]: "); fflush(stdin); gets(confirm);
57                     if(strncmp(confirm, "Y") == 0) {
58                         // Tambahkan dosen pembimbing
59                         inputDosenPembimbing(&M.dosBim);
60                         printf("\n\t[+] Berhasil menginput data mahasiswa dan dosen pembimbing!");
61                     } else {
62                         printf("\n\t[+] Berhasil menginput data mahasiswa tanpa dosen pembimbing!");
63                     }
64                 } else {
65                     // Data sudah diisi
66                     printf("\n\t[-] Data mahasiswa sudah diisi!");
67                 }
68                 break;
69
70             case 2:
71                 printf("\n\t--- Tampil Data ---");
72                 if(!isMahasiswaEmpty(M)) {
73                     tampilMahasiswa(M);
74                 } else {
75                     printf("\n\t[-] Data mahasiswa belum diisi!");
76                 }
77                 break;
78
79             default:
80                 break;
81         }
82     } while(menu != 0);
83 }
```

```

78
79     case 3:
80         if(!isMahasiswaEmpty(M)) {
81             int submenu;
82             // Submenu apakah user ingin menginput dosen pembimbing atau mahasiswa
83             printf("\n\t--- Ubah Data ---");
84             printf("\n\t[1] Ubah data Mahasiswa");
85             printf("\n\t[2] Ubah data Dosen Pembimbing");
86             printf("\n\t>> "); scanf("%d", &submenu);
87
88             switch(submenu) {
89                 case 1:
90                     printf("\n\t--- Ubah Data Mahasiswa ---\n");
91                     inputMahasiswa(&M);
92                     printf("\n\t[+] Berhasil menginput data mahasiswa!");
93                     break;
94                 case 2:
95                     printf("\n\t--- Ubah Data Dosen Pembimbing ---\n");
96                     inputDosenPembimbing(&M.dosBim);
97                     printf("\n\t[+] Berhasil menginput data dosen pembimbing!");
98                     break;
99                 default:
100                     printf("\n\t[-] Menu tidak tersedia!");
101                     break;
102             }
103         } else {
104             printf("\n\t[-] Data mahasiswa belum diisi!");
105         }
106         break;
107
108     case 4:
109         if(!isMahasiswaEmpty(M)) {
110             string confirm;
111             printf("\n\tApakah Anda yakin benar-benar ingin menghapus data? [Y/N]: "); fflush(stdin); gets(confirm);
112             if(strcmp(confirm, "Y") == 0) {
113                 initMahasiswa(&M);
114                 printf("\n\t[+] Berhasil menghapus data mahasiswa dan dosen pembimbing!");
115             } else {
116                 printf("\n\t[-] Batal menghapus data mahasiswa dan dosen pembimbing!");
117             }
118         } else {
119             printf("\n\t[-] Data mahasiswa belum diisi!");
120         }
121         break;
122
123     case 0:
124         printf("\n\t[+] Terima kasih.");
125         printf("\n\t<Nama> - <NPM lengkap> - Praktikum Dasar Pemrograman <Kelas>");
126         break;
127
128     default:
129         printf("\n\t[-] Menu tidak tersedia!");
130         break;
131 }
132 getch();
133
134 } while (menu != 0);
135 }
136
137 void initMahasiswa(Mahasiswa *M) {
138     /* Di prosedur ini, kita akan mengeset semua field pada Mahasiswa menjadi kosong
139     * Dapat kita lakukan dengan cara mengeset field string menjadi "" (string kosong)
140     * atau mengeset field int/float menjadi 0
141     */
142     // Inisialisasi data Mahasiswa
143     strcpy((*M).nama, "");
144     (*M).ipk = 0;
145     (*M).umur = 0;
146
147     // Inisialisasi data DosenPembimbing
148     strcpy((*M).dosBim.nama, "");
149     strcpy((*M).dosBim.nidn, "");
150 }
151
152 void inputMahasiswa(Mahasiswa *M) {
153     // Kita bisa menggunakan variabel pembantu agar memudahkan kita dalam mencari nama field
154     // Coba bandingkan: apakah menulis `(*M).` akan memunculkan list nama field dalam record?
155     // Bagaimana dengan menulis `input.` , list nama field dalam record Mahasiswa akan tampil.
156     Mahasiswa input = (*M);
157
158     while(true) {
159         // Harus ada error handling apabila nama mahasiswa kosong,
160         // karena nama mahasiswa digunakan untuk mengecek record kosong atau tidak.
161         printf("\tNama mahasiswa : "); fflush(stdin); gets(input.nama);
162         if(strlen(input.nama) == 0) {
163             printf("\t[-] Nama mahasiswa tidak boleh kosong!\n");
164         } else {
165             // keluar dari loop apabila nama mahasiswa tidak kosong.
166             break;
167         }
168     }
169 }

```

```

169
170 // contoh error handling yang bisa diimplementasi pada field IPK:
171 // IPK < 0 OR IPK > 4
172 printf("\tIPK mahasiswa : "); scanf("%f", &input.ipk);
173
174 // contoh error handling yang bisa diimplementasi pada field umur:
175 // umur < 0
176 printf("\tUmur mahasiswa : "); scanf("%d", &input.umur);
177
178 // "Pindahkan" inputan user ke dalam Mahasiswa *M
179 (*M) = input;
180 }
181
182 void inputDosenPembimbing(DosenPembimbing *DP) {
183 // Kembali menggunakan variabel pembantu
184 DosenPembimbing input = (*DP);
185
186 while(true) {
187 printf("\tNama dosen pembimbing : "); fflush(stdin); gets(input.nama);
188 if(strlen(input.nama) == 0) {
189 printf("\t[-] Nama dosen pembimbing tidak boleh kosong!\n");
190 } else {
191 break;
192 }
193 }
194
195 // contoh error handling yang bisa diimplementasi pada field NIDN:
196 // length of NIDN < 10 OR length of NIDN > 20
197 printf("\tNIDN dosen pembimbing : "); fflush(stdin); gets(input.nidn);
198
199 // "Pindahkan" inputan user ke dalam DosenPembimbing *DP
200 (*DP) = input;
201 }
202
203 int tahunLahirMahasiswa(int umur) {
204 /* Fungsi ini mengembalikan tahun lahir mahasiswa (secara kasar)
205 * terhadap tahun 2022
206 */
207 return 2022 - umur;
208 }
209
210 int isMahasiswaEmpty(Mahasiswa M) {
211 /* Di fungsi ini, kita akan memeriksa apakah data mahasiswa tersebut kosong,
212 * hanya dengan memeriksa apakah nama mahasiswa tersebut adalah empty string.
213 */
214 return (strcmp(M.nama, "") == 0);
215 }
216
217 int isDosBimEmpty(DosenPembimbing DP) {
218 /* Di fungsi ini, kita akan memeriksa apakah data dosen pembimbing kosong,
219 * hanya dengan memeriksa apakah nama dosen pembimbing adalah empty string.
220 */
221 return (strcmp(DP.nama, "") == 0);
222 }
223
224 void tampilMahasiswa(Mahasiswa M) {
225 /* Prosedur ini menampilkan data Mahasiswa dan DosenPembimbing (apabila ada)
226 */
227 printf("\n\t[*] Data Mahasiswa:");
228 printf("\n\tNama mahasiswa : %s", M.nama);
229 printf("\n\tIPK mahasiswa : %.2f", M.ipk);
230 printf("\n\tUmur mahasiswa : %d", M.umur);
231 printf("\n\tTahun lahir : %d\n", tahunLahirMahasiswa(M.umur));
232
233 if(!isDosBimEmpty(M.dosBim)) {
234 // Tampilkan data DosenPembimbing
235 printf("\n\t\t[*] Dosen Pembimbing:");
236 printf("\n\t\tNama dosen pembimbing : %s", M.dosBim.nama);
237 printf("\n\t\tNIDN dosen pembimbing : %s\n", M.dosBim.nidn);
238 }
239 }

```

## MINI DEMO

### INPUT DATA:

*Dengan dosen pembimbing:*

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 1

    --- Input Data ---
    Nama mahasiswa   : Jolly Hans Frankle
    IPK mahasiswa    : 3.81
    Umur mahasiswa   : 19

    Apakah sudah ada dosen pembimbing? [Y/N]: Y
    Nama dosen pembimbing : W. J. Lala Mentik
    NIDN dosen pembimbing : 09.08.1982

    [+] Berhasil menginput data mahasiswa dan dosen pembimbing!
```

*Tampilan data:*

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 2

    --- Tampil Data ---
    [*] Data Mahasiswa:
    Nama mahasiswa   : Jolly Hans Frankle
    IPK mahasiswa    : 3.81
    Umur mahasiswa   : 19
    Tahun lahir      : 2003

    [*] Dosen Pembimbing:
    Nama dosen pembimbing : W. J. Lala Mentik
    NIDN dosen pembimbing : 09.08.1982
```

*Tanpa dosen pembimbing:*

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 1

    --- Input Data ---
    Nama mahasiswa   : Jolly Hans Frankle
    IPK mahasiswa    : 3.98
    Umur mahasiswa   : 19

    Apakah sudah ada dosen pembimbing? [Y/N]: N

    [+] Berhasil menginput data mahasiswa tanpa dosen pembimbing!_
```

*Tampilan data:*

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 2

    --- Tampil Data ---
    [*] Data Mahasiswa:
    Nama mahasiswa   : Jolly Hans Frankle
    IPK mahasiswa    : 3.98
    Umur mahasiswa   : 19
    Tahun lahir      : 2003
```

## MENGEDIT DATA:

*Mengedit data mahasiswa:*

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 3

    --- Ubah Data ---
    [1] Ubah data Mahasiswa
    [2] Ubah data Dosen Pembimbing
    >> 1

    --- Ubah Data Mahasiswa ---
    Nama mahasiswa   : Calvin Andrean Suhedy
    IPK mahasiswa    : 3.99
    Umur mahasiswa   : 20

    [+] Berhasil menginput data mahasiswa!
```

*Tampilan data (apabila tidak ada dosen pembimbing)*

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 2

    --- Tampil Data ---
    [*] Data Mahasiswa:
    Nama mahasiswa   : Calvin Andrean Suhedy
    IPK mahasiswa    : 3.99
    Umur mahasiswa   : 20
    Tahun lahir      : 2002
```

Menambahkan data dosen pembimbing melalui menu "Edit data":

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 3

    --- Ubah Data ---
    [1] Ubah data Mahasiswa
    [2] Ubah data Dosen Pembimbing
    >> 2

    --- Ubah Data Dosen Pembimbing ---
    Nama dosen pembimbing : Fransiskus Xaverius Seda
    NIDN dosen pembimbing : 07.19.1877

    [+] Berhasil menginput data dosen pembimbing!
```

Tampilan data:

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 2

    --- Tampil Data ---
    [*] Data Mahasiswa:
    Nama mahasiswa : Calvin Andrean Suhedy
    IPK mahasiswa : 3.99
    Umur mahasiswa : 20
    Tahun lahir : 2002

    [*] Dosen Pembimbing:
    Nama dosen pembimbing : Fransiskus Xaverius Seda
    NIDN dosen pembimbing : 07.19.1877
```



## HAPUS DATA

*Yakin menghapus data:*

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 4

    Apakah Anda yakin benar-benar ingin menghapus data? [Y/N]: Y

    [+] Berhasil menghapus data mahasiswa dan dosen pembimbing!
```

*Batal menghapus data:*

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 4

    Apakah Anda yakin benar-benar ingin menghapus data? [Y/N]: N

    [-] Batal menghapus data mahasiswa dan dosen pembimbing!
```

## ERROR HANDLING:

*Mencoba menginput data padahal data sudah terinput:*

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 1

    [-] Data mahasiswa sudah diisi!
```

*Mencoba menampilkan data padahal data belum diinput:*

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 2

    --- Tampil Data ---
    [-] Data mahasiswa belum diisi!
```

#### **KELUAR PROGRAM:**

```
C:\Users\jolly\OneDrive - Universitas Atma Jaya Yogyakarta\MATERI KULIAH\1.Asistens...
--- Guided Record: Mahasiswa x Dosen Pembimbing ---
[1] Input Data
[2] Tampilkan Data
[3] Edit Data
[4] Hapus Data
[0] Exit
>> 0

    [+] Terima kasih.
    <Nama> - <NPM lengkap> - Praktikum Dasar Pemrograman <Kelas>
-----
Process exited after 682.6 seconds with return value 0
Press any key to continue . . .
```