Modul Panduan Struktur Data Menggunakan MATLAB

Yang Tidak Dipublikasikan

Yusuf Lestanto, ST., MSc., MBA

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK DAN ILMU KOMPUTER
UNIVERSITAS BAKRIE
Semester Genap 2022/2023

LEMBAR PENGESAHAN MODUL PANDUAN YANG TIDAK DIPUBLIKASIKAN

Judul	:	Modul Panduan Struktur Data Menggunakan MATLAB
Penulis		
a. Nama lengkap	:	Yusuf Lestanto, ST., MSc., MBA.
b. Jenis kelamin	:	Laki-laki
c. NIDN	:	0302057105
d. Bidang Keahlian	:	Teknik Informatika
e. Program Studi	:	Teknik Informatika
f. E-mail	:	yusuf.lestanto@bakrie.ac.id
Jangka waktu penulisan	:	1 Mei 2023 - 25 Juni 2023

Jakarta, 28 Juli 2023

Menyetujui	Penulis
Ardiansyah, S.TP., M.Si., Ph.D.	Yusuf Lestanto, S.T., MSc., MBA
NIDN: 0318107501	NIDN: 0302057105

Modul Panduan Struktur Data Menggunakan MATLAB

Yusuf Lestanto

31 Juli 2023

Struktur di MATLAB memiliki beberapa keuntungan dibandingkan dengan tipe data lainnya, termasuk:

- Efisiensi
  Struktur dapat menyimpan data yang berbeda dalam satu variabel, yang dapat menghemat memori dan waktu pemrosesan.
- Kerapian
  Struktur dapat digunakan untuk menyimpan data yang terkait, yang dapat membuat kode lebih rapi dan mudah dibaca.
- Fleksibilitas
  Struktur dapat digunakan untuk menyimpan berbagai jenis data, yang dapat membuatnya lebih fleksibel daripada tipe data lainnya.

Berikut adalah beberapa contoh penggunaan struktur di MATLAB:

- Menyimpan data pelanggan
  Struktur dapat digunakan untuk menyimpan data mahasiswa, seperti nama, alamat, dan nomor telepon.
- Menyimpan data produk
  Struktur dapat digunakan untuk menyimpan data produk, seperti nama, harga, dan stok.
- Menyimpan hasil simulasi
  Struktur dapat digunakan untuk menyimpan hasil simulasi, seperti data input, data output, dan waktu eksekusi.

Secara keseluruhan, struktur adalah tipe data yang kuat dan serbaguna yang dapat digunakan untuk berbagai tugas di MATLAB.

1 Membuat Struktur

Struktur adalah tipe data yang dapat digunakan untuk menyimpan data yang berbeda dalam satu variabel. Struktur sering digunakan untuk menyimpan data yang terkait, seperti data mahasiswa atau data produk.

Untuk membuat struktur di MATLAB, Anda dapat menggunakan sintaksis berikut:

```
struct_name = struct('field1', value1, 'field2', value2, ...);
```

Misalnya, untuk membuat struktur dengan dua bidang, *field1* dan *field2*, Anda dapat menggunakan kode berikut:

```
my_struct = struct('field1', 1, 'field2', 2);
```

Kemudian dapat mengakses *field* dari struktur menggunakan titik, seperti berikut:

```
my_struct.field1
```

Statement di atas akan mencetak nilai 1.

Selain ini juga dapat menggunakan sintaksis berikut untuk membuat struktur:

```
my_struct = struct('field1', {1, 2, 3}, 'field2', {4, 5, 6});
```

Ini akan membuat struktur dengan dua *field*, *field1* dan *field2*, dan masing-masing *field* berisi vektor.

2 Mengambil data dari struktur

Ada beberapa cara untuk mengambil data dari struktur MATLAB. Salah satu cara adalah menggunakan titik. Misalnya, jika struktur memiliki bernama *my_struct* dengan bidang (*field*) bernama *name* dan *age*, untuk dapat mengakses bidang 'name' dengan menggunakan sintaks berikut:

```
my_struct.name
```

Ini akan mengembalikan nilai bidang 'name'.

Cara lain untuk mengambil data dari struktur MATLAB adalah menggunakan fungsi *get*(). Fungsi tersebut memiliki sintaks berikut:

```
get(my_struct, 'field_name')
```

Fungsi *get*() akan mengembalikan nilai bidang *field_name* dari struktur *my_struct*. Selain itu juga dapat menggunakan fungsi *fieldnames*() untuk mendapatkan daftar nama bidang dari struktur. Fungsi *fieldnames*() memiliki sintaks berikut:

```
fieldnames(my_struct)
```

Fungsi *fieldnames*() akan mengembalikan vektor berisi nama bidang dari struktur *my_struct*.

3 Menggabungkan struktur

Untuk menggabungkan struktur di MATLAB dapat menggunakan fungsi *cat*(). Fungsi tersebut memiliki sintaks berikut:

```
cat(dim, array1, array2, ..., arrayN)
```

Dimensi (*dim*) adalah arah di mana struktur akan digabungkan, array1, array2, ..., arrayN adalah struktur yang akan digabungkan.

Misalnya, untuk menggabungkan dua struktur **A** dan **B** bersama-sama di sepanjang dimensi pertama, dapat menggunakan kode berikut:

```
C = cat(1, A, B);
```

Fungsi *cat*() akan mengembalikan struktur baru dengan bidang yang sama dari **A** dan **B**. Bidang dari **A** dan **B** akan digabungkan bersama-sama di sepanjang dimensi pertama.

Selain itu, juga bisa menggunakan fungsi *vertcat*() dan *horzcat*() untuk menggabungkan struktur di MATLAB. Fungsi *vertcat*() menggabungkan struktur bersama-sama di sepanjang dimensi kedua, sedangkan fungsi *horzcat*() menggabungkan struktur bersama-sama di sepanjang dimensi pertama.

Berikut adalah beberapa contoh penggunaan fungsi *cat*() untuk menggabungkan struktur:

```
A = struct('field1', 1, 'field2', 2);
B = struct('field3', 3, 'field4', 4);
C = cat(1, A, B);
D = vertcat(A, B);
E = horzcat(A, B);
```

Variabel **C** akan memiliki bidang *field1*, *field2*, *field3*, dan *field4*. Variabel **D** akan memiliki bidang *field1* dan *field2* dari **A** di atas bidang *field3* dan *field4* dari **B**. Variabel **E** akan memiliki bidang *field1* dari **A** di sebelah kiri bidang *field2* dari **B**.

4 Menyimpan data dari struktur ke variabel

Di MATLAB, Anda dapat menyimpan data dari struktur ke dalam variabel dengan cara mengakses anggota struktur (*field*) menggunakan titik (.) dan menyimpan nilainya ke dalam variabel baru. Berikut adalah contoh bagaimana melakukannya:

```
% Membuat struktur data (struct) sebagai contoh
data_mahasiswa.nama = 'John Doe';
data_mahasiswa.usia = 25;
data_mahasiswa.jurusan = 'Informatika';
data_mahasiswa.nilai = 85.5;

% Menyimpan data dari struktur ke variabel
nama_mahasiswa = data_mahasiswa.nama;
usia_mahasiswa = data_mahasiswa.usia;
jurusan_mahasiswa = data_mahasiswa.jurusan;
nilai_mahasiswa = data_mahasiswa.nilai;
```

```
% Cetak hasilnya
disp(['Nama: ', nama_mahasiswa]);
disp(['Usia: ', num2str(usia_mahasiswa)]);
disp(['Jurusan: ', jurusan_mahasiswa]);
disp(['Nilai: ', num2str(nilai_mahasiswa)]);
```

Dalam contoh di atas, dibuat struktur data *data_mahasiswa* yang berisi beberapa field seperti *nama*, *usia*, *jurusan*, dan *nilai*. Selanjutnya, mengakses nilai dari setiap field menggunakan titik (.) dan menyimpannya ke dalam variabel-variabel baru yaitu *nama_mahasiswa*, *usia_mahasiswa*, *jurusan_mahasiswa*, dan *nilai_mahasiswa*. Kemudian, mencetak hasilnya dengan menggunakan *disp* atau *fprintf* jika ingin format lebih kompleks.

5 Operasi-operasi lain pada struktur

Dalam MATLAB, struktur (struct) adalah tipe data yang dapat menyimpan beberapa nilai dengan berbagai jenis tipe data dalam satu variabel. Struktur biasanya digunakan untuk mengorganisasi data yang terkait menjadi satu kesatuan. Di bawah ini, penjelasan beberapa operasi umum yang dapat dilakukan pada struktur dalam MATLAB:

1. Membuat Struktur

   Untuk membuat struktur baru digunakan kurung kurawal {} dan memberikan nama field dan nilai yang sesuai. Contoh:

   ```
   % Membuat struktur data baru
   mahasiswa.nama = 'John Doe';
   mahasiswa.usia = 25;
   mahasiswa.jurusan = 'Informatika';
   mahasiswa.nilai = 85.5;
   ```

2. Akses Nilai dalam Struktur

   Untuk dapat mengakses nilai dalam struktur dengan menggunakan titik (.) dan nama field-nya. Contoh:

   ```
   % Akses nilai dalam struktur
   disp(['Nama: ', mahasiswa.nama]);
   disp(['Usia: ', num2str(mahasiswa.usia)]);
   disp(['Jurusan: ', mahasiswa.jurusan]);
   disp(['Nilai: ', num2str(mahasiswa.nilai)]);
   ```

3. Mengubah Nilai dalam Struktur

   Untuk dapat mengubah nilai field dalam struktur dengan cara mengaksesnya menggunakan titik (.) dan memberikan nilai baru. Contoh:

```
% Mengubah nilai dalam struktur
mahasiswa.usia = 26; % Mengubah usia menjadi 26
mahasiswa.nilai = 90.0; % Mengubah nilai menjadi 90.0
```

4. Menambahkan Field Baru

   Untuk dapat menambahkan field baru ke dalam struktur dengan cara langsung memberikan nilai baru ke field yang belum ada. Contoh:

```
% Menambahkan field baru ke dalam struktur
mahasiswa.alamat = 'Jl. Contoh No. 123'; % Menambahkan field '
    alamat'
```

5. Menghapus Field dari Struktur

   Untuk menghapus field dari struktur, dapat menggunakan fungsi *rmfield*. Contoh:

```
% Menghapus field dari struktur
mahasiswa = rmfield(mahasiswa, 'nilai'); % Menghapus field 'nilai'
```

6. Menggabungkan Dua Struktur

   Untuk menggabungkan dua struktur menggunakan operator *struct* atau fungsi *fieldnames*. Contoh:

```
% Menggabungkan dua struktur
struktur1.nama = 'John';
struktur1.umur = 25;

struktur2.jurusan = 'Informatika';
struktur2.nilai = 85.5;

gabungan = struct('data1', struktur1, 'data2', struktur2);
% atau
gabungan = struct('data1', struktur1);
gabungan.data2 = struktur2;
```

7. Iterasi pada Struktur

   Untuk dapat melakukan iterasi pada field-field dalam struktur menggunakan loop *for* atau fungsi *fieldnames*. Contoh:

```
% Iterasi pada struktur menggunakan loop for
fieldNames = fieldnames(mahasiswa);
for i = 1:numel(fieldNames)
  fieldName = fieldNames{i};
```

```
    value = mahasiswa.(fieldName);
    disp([fieldName, ': ', num2str(value)]);
end

% Iterasi pada struktur menggunakan fungsi fieldnames
fieldNames = fieldnames(mahasiswa);
for i = 1:numel(fieldNames)
    fieldName = fieldNames{i};
    value = getfield(mahasiswa, fieldName);
    disp([fieldName, ': ', num2str(value)]);
end
```

Struktur adalah alat yang berguna untuk mengorganisasi dan menyimpan data yang terkait secara terstruktur dalam satu variabel.

6 Membuat *Map Container*

Wadah peta (*map container*) dalam MATLAB adalah sebuah struktur data yang memungkinkan untuk menyimpan dan mengakses data berdasarkan pasangan kunci-nilai (*key-value pairs*). Wadah peta ini sering disebut juga sebagai struktur data *map*, *dictionary*, atau *associative array* pada bahasa pemrograman lainnya. Dengan menggunakan wadah peta, dapat menghubungkan nilai-nilai tertentu dengan kunci yang unik untuk setiap nilai tersebut.

Wadah peta di MATLAB diwakili oleh kelas *containers.Map*. wadah peta baru dapa dibuat dan dikelola dengan berbagai operasi seperti menambahkan pasangan kunci-nilai, mengakses nilai berdasarkan kunci, menghapus pasangan kunci-nilai, memeriksa keberadaan kunci, dan banyak lagi.

Berikut adalah contoh penggunaan wadah peta dalam MATLAB:

1. Membuat wadah peta dan menambahkan pasangan kunci-nilai.

```
% Membuat wadah peta baru
mapObj = containers.Map;

% Menambahkan pasangan kunci-nilai
mapObj('nama') = 'John Doe';
mapObj('usia') = 30;
mapObj('pekerjaan') = 'Insinyur';

% Mengakses nilai berdasarkan kunci
disp(['Nama:', mapObj('nama')]);
disp(['Usia:', num2str(mapObj('usia'))]);
```

```
disp(['Pekerjaan:', mapObj('pekerjaan')]);
```

2. Membuat wadah peta dengan inisialisasi pasangan kunci-nilai.

```
% Membuat wadah peta dengan pasangan kunci-nilai awal
keys = {'nama', 'usia', 'pekerjaan'};
values = {'Jane Smith', 25, 'Data Scientist'};
mapObj = containers.Map(keys, values);

% Mengakses nilai berdasarkan kunci
disp(['Nama:', mapObj('nama')]);
disp(['Usia:', num2str(mapObj('usia'))]);
disp(['Pekerjaan:', mapObj('pekerjaan')]);
```

3. Memeriksa keberadaan kunci dalam wadah peta dan menghapus pasangan kunci-nilai.

```
% Memeriksa keberadaan kunci dalam wadah peta
kunciCek = 'usia';
if isKey(mapObj, kunciCek)
  disp([kunciCek, ' ada dalam wadah peta.']);
else
  disp([kunciCek, ' tidak ada dalam wadah peta.']);
end

% Menghapus pasangan kunci-nilai dari wadah peta
kunciHapus = 'usia';
if isKey(mapObj, kunciHapus)
  remove(mapObj, kunciHapus);
  disp([kunciHapus, ' telah dihapus dari wadah peta.']);
else
  disp([kunciHapus, ' tidak ada dalam wadah peta.']);
end
```

Wadah peta dalam MATLAB sangat bermanfaat ketika ingin menghubungkan data yang terkait dengan kunci yang unik, sehingga memudahkan akses dan manipulasi data berdasarkan kunci tersebut.

7 Penggabungan dan beberapa operasi pada wadah peta (*Map Container*)

Berikut adalah beberapa operasi lanjutan dan penggabungan pada wadah peta (*map container*) dalam MATLAB:

1. Penggabungan Dua Wadah Peta

   Untuk menggabungkan dua wadah peta, dapat menggunakan fungsi *containers.Map* dan menyediakan wadah peta pertama sebagai argumen pertama, lalu menambahkan pasangan kunci-nilai dari wadah peta kedua ke wadah peta pertama. Jika ada kunci yang sama dalam kedua wadah peta, nilai dari kunci tersebut akan diambil dari wadah peta kedua. Contoh:

```
% Membuat dua wadah peta
mapObj1 = containers.Map({'nama', 'usia'}, {'John Doe', 30});
mapObj2 = containers.Map({'pekerjaan', 'usia'}, {'Insinyur', 35});

% Menggabungkan dua wadah peta
mapMerged = [mapObj1; mapObj2];

% Cetak hasilnya
disp(mapMerged('nama'));
disp(mapMerged('usia'));
disp(mapMerged('pekerjaan'));
```

2. Iterasi pada Wadah Peta

   Untuk dapat melakukan iterasi pada pasangan kunci-nilai dalam wadah peta menggunakan loop *for*. Fungsi *keys* dan *values* juga dapat membantu untuk mendapatkan daftar kunci dan nilai dalam wadah peta. Contoh:

```
mapObj = containers.Map({'nama', 'usia', 'pekerjaan'}, {'John Doe'
    , 30, 'Insinyur'});

% Iterasi menggunakan loop for
keysList = keys(mapObj);
for i = 1:length(keysList)
  key = keysList{i};
  value = mapObj(key);
  disp([key, ': ', num2str(value)]);
end
```

```
% Iterasi menggunakan loop for dan fungsi values
valuesList = values(mapObj);
for i = 1:length(valuesList)
  value = valuesList{i};
  disp(['Nilai: ', num2str(value)]);
end
```

3. Penghapusan Seluruh Pasangan Kunci-nilai dalam Wadah Peta

   Untuk dapat menggunakan fungsi *remove* untuk menghapus seluruh pasangan kunci-nilai dalam wadah peta. Contoh:

```
mapObj = containers.Map({'nama', 'usia', 'pekerjaan'}, {'John Doe'
    , 30, 'Insinyur'});

% Menghapus seluruh pasangan kunci-nilai
keysList = keys(mapObj);
for i = 1:length(keysList)
  remove(mapObj, keysList{i});
end
```

4. Mendapatkan Daftar Kunci atau Nilai dalam Wadah Peta

   Fungsi *keys* dan *values* dapat digunakan untuk mendapatkan daftar kunci atau nilai dalam wadah peta. Contoh:

```
mapObj = containers.Map({'nama', 'usia', 'pekerjaan'}, {'John Doe'
    , 30, 'Insinyur'});

% Mendapatkan daftar kunci dalam wadah peta
keysList = keys(mapObj);
disp(keysList);

% Mendapatkan daftar nilai dalam wadah peta
valuesList = values(mapObj);
disp(valuesList);
```

5. Mengetahui Jumlah Pasangan Kunci-nilai dalam Wadah Peta

   Fungsi *length* dapat digunakan untuk mengetahui berapa jumlah pasangan kunci-nilai dalam wadah peta. Contoh:

```
mapObj = containers.Map({'nama', 'usia', 'pekerjaan'}, {'John Doe'
    , 30, 'Insinyur'});
```

```
disp(['Jumlah pasangan kunci-nilai: ', num2str(length(mapObj))]);
```