

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2
MODUL 7
STRUCK & ARRAY**



Oleh:

NAUFAL THORIQ MUZHAFAR

2311102078

IF-11-02

**S1 TEKNIK INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO**

2024

I. DASAR TEORI

Array

Array digunakan untuk menyimpan beberapa nilai bertipe sama dalam satu variabel, alih-alih mendeklarasikan variabel terpisah untuk setiap nilai.

Ada dua cara untuk mendeklarasikan array:

```
var array_name = [panjang]datatype{value} // panjangnya
terdefinisi

atau

var array_name = [...]datatype{value} // panjangnya
tidak terdefinisi

atau juga bisa

array_name := [panjang]datatype{value} // panjangnya
terdefinisi

atau

array_name := [...]datatype{value} // panjangnya tidak
terdefinisi
```

Kita dapat mengakses elemen array tertentu dengan merujuk ke nomor indeks. Indeks array dimulai dari 0. Itu berarti bahwa [0] adalah elemen pertama, [1] adalah elemen kedua, dst. Kita juga dapat mengubah nilai elemen array tertentu dengan merujuk pada nomor indeks.

```
func main() {
    prices := [3]int{10,20,30}

    prices[2] = 50 // mengubah array prices pada indeks
    ke 2 yang awalnya 30 menjadi 50
}
```

Jika array atau salah satu elemennya belum diinisialisasi dalam kode, array tersebut akan diberi nilai default dari tipenya. Nilai default untuk int adalah 0, dan nilai default untuk string adalah "".

```
func main() {
    arr1 := [5]int{} // tidak diinisialisasikan maka isi
    dari array nya 0 semua
    arr2 := [5]int{1,2} // hanya beberapa indeks yang
    diinialisasikan, hanya array kosong yang mempunyai
    nilai 0
    arr3 := [5]int{1,2,3,4,5} // semua indeks
    diinialisasikan
}
```

Dimungkinkan untuk menginisialisasi hanya elemen tertentu dalam suatu array.

Contoh

```
package main
import ("fmt")

func main() {
    arr1 := [5]int{1:10,2:40}

    fmt.Println(arr1)
}
```

Penjelasan

Array di atas memiliki 5 elemen.

- 1:10 berarti: tetapkan 10 pada indeks array indeks ke-1 (elemen kedua).
- 2:40 berarti: tetapkan 40 pada indeks array indeks ke-2 (elemen ketiga).

Kita dapat menggunakan fungsi **len()** untuk menemukan panjang array:

```
func main() {
    arr1 := [4]string{"Volvo", "BMW", "Ford", "Mazda"}
    arr2 := [...]int{1,2,3,4,5,6}

    fmt.Println(len(arr1))//maka akan menampilkan
    Panjang array dari arr1 yaitu 4
    fmt.Println(len(arr2))//walaupun panjang array tidak
    diinialisasikan jumlah pada Panjang array bergantung
    pada elemen di dalamnya maka Panjang dari arr2 yaitu 6
}
```

Slices

Slice mirip dengan array, tetapi lebih kuat dan fleksibel. Seperti array, slice juga digunakan untuk menyimpan beberapa nilai dengan tipe yang sama dalam satu variabel. Namun, tidak seperti array, panjang slice dapat bertambah dan berkurang sesuai keinginan Anda. Dalam Go, ada beberapa cara untuk membuat slice:

- Menggunakan format `[]datatype{value}`
- Membuat slice dari array
- Menggunakan fungsi **make()**

Membuat Slice Dengan `[]datatype{values}`

```
slice_name := []datatype{values}  
  
atau  
  
myslice : {} = []int
```

Kode di atas mendeklarasikan irisan kosong dengan panjang 0 dan kapasitas 0. Untuk menginisialisasi irisan selama deklarasi, gunakan ini:

```
myslice := []int{1,2,3}
```

Kode di atas mendeklarasikan irisan bilangan bulat dengan panjang 3 dan juga kapasitas 3. Dalam Go, ada dua fungsi yang dapat digunakan untuk mengembalikan panjang dan kapasitas slices:

- Fungsi `len()` - mengembalikan panjang irisan (jumlah elemen dalam irisan)
- Fungsi `cap()` - mengembalikan kapasitas irisan (jumlah elemen irisan dapat bertambah atau berkurang)

Membuat Slice Dari Array

Anda dapat membuat irisan dengan mengiris sebuah array:

```
var myarray = [length]datatype{values} // Sebuah array  
myslice := myarray[start:end] // Sebuah slice yang  
dibuat dari array
```

Contoh

```
func main() {  
    arr1 := [6]int{10, 11, 12, 13, 14, 15}  
    myslice := arr1[2:4]}
```

```
fmt.Printf("myslice = %v\n", myslice)
fmt.Printf("length = %d\n", len(myslice))
fmt.Printf("capacity = %d\n", cap(myslice))
}
```

Dalam contoh di atas `myslice` adalah irisan dengan panjang 2. Irisan ini dibuat dari `arr1` yang merupakan array dengan panjang 6.

Irisan dimulai dari elemen ketiga array yang memiliki nilai 12 (ingat bahwa indeks array dimulai dari 0. Itu berarti bahwa `[0]` adalah elemen pertama, `[1]` adalah elemen kedua, dst.). Irisan dapat bertambah hingga akhir array. Ini berarti bahwa kapasitas irisan adalah 4.

Jika `myslice` dimulai dari elemen 0, kapasitas irisan akan menjadi 6.

Membuat Slice dengan fungsi `make()`

Fungsi `make()` juga dapat digunakan untuk membuat irisan.

```
slice_name := make([]type, panjang, kapasitas)
```

Jika parameter kapasitas tidak ditentukan, maka akan sama dengan panjang.

Cara mengakses elemen dan mengganti nilai elemennya sama dengan array yaitu menggunakan indeks untuk mengakses elemen tertentu

Tambahkan Elemen ke Irisan

Kita dapat menambahkan elemen ke akhir irisan menggunakan fungsi `append()`:

```
slice_name =
append(slice_name, element1, element2, ...)
```

Tambahkan Satu Irisan Ke Irisan Lain

```
slice3 = append(slice1, slice2...)
```

'...' setelah `slice2` diperlukan saat menambahkan elemen dari satu slice ke slice lainnya.

Efisiensi Memori

Saat menggunakan slices, Go memuat semua elemen yang mendasarinya ke dalam memori.

Jika array berukuran besar dan kita hanya memerlukan beberapa elemen, sebaiknya salin elemen tersebut menggunakan fungsi `copy()`.

Fungsi `copy()` membuat array dasar baru dengan hanya elemen yang diperlukan untuk slices. Ini akan mengurangi memori yang digunakan untuk program.

```
copy(dest, src)
```

Fungsi `copy()` mengambil dua bagian `dest` dan `src`, dan menyalin data dari `src` ke `dest`. Fungsi ini mengembalikan jumlah elemen yang disalin.

Struct

Struct (kependekan dari structure) digunakan untuk membuat kumpulan anggota dengan tipe data berbeda, menjadi satu variabel.

Sementara array digunakan untuk menyimpan beberapa nilai dengan tipe data yang sama ke dalam satu variabel, struct digunakan untuk menyimpan beberapa nilai dengan tipe data berbeda ke dalam satu variabel.

Struktur dapat berguna untuk mengelompokkan data bersama-sama untuk membuat rekaman.

Mendeklarasikan Struktur

Untuk mendeklarasikan struktur di Go, gunakan kata kunci **type** dan **struct**:

```
type struct_name struct {  
    anggota1 datatype;  
    anggota2 datatype;  
    anggota3 datatype; ...  
}
```

Mengakses Anggota

Struktur Untuk mengakses anggota struktur mana pun, gunakan operator titik (.) di antara nama variabel struktur dan anggota struktur:

```
type Person struct {
    name string
    age int
    job string
    salary int
}

func main() {
    var pers1 Person
    var pers2 Person

    // Pers1 specification
    pers1.name = "Hege"
    pers1.age = 45
    pers1.job = "Teacher"
    pers1.salary = 6000

    // Pers2 specification
    pers2.name = "Cecilie"
    pers2.age = 24
    pers2.job = "Marketing"
    pers2.salary = 4500

    // Mengakses perl1 dan mengoutputkan perl1
    fmt.Println("Name: ", pers1.name)
    fmt.Println("Age: ", pers1.age)
    fmt.Println("Job: ", pers1.job)
    fmt.Println("Salary: ", pers1.salary)

    // Mengakses perl2 dan mengoutputkan perl2
    fmt.Println("Name: ", pers2.name)
    fmt.Println("Age: ", pers2.age)
    fmt.Println("Job: ", pers2.job)
    fmt.Println("Salary: ", pers2.salary)
}
```

Melewatkan Struct sebagai Argumen Fungsi

Kita juga dapat meneruskan struktur sebagai argumen fungsi, seperti ini:

```
type Person struct {
    name string
    age int
    job string
    salary int
}

func main() {
    var pers1 Person
    var pers2 Person

    // Pers1 specification
    pers1.name = "Hege"
    pers1.age = 45
    pers1.job = "Teacher"
    pers1.salary = 6000

    // Pers2 specification
    pers2.name = "Cecilie"
    pers2.age = 24
    pers2.job = "Marketing"
    pers2.salary = 4500

    // Cetak info Pers1 dengan memanggil fungsi
    printPerson(pers1)

    // Cetak info Pers2 dengan memanggil fungsi
    printPerson(pers2)
}

func printPerson(pers Person) {
    fmt.Println("Name: ", pers.name)
    fmt.Println("Age: ", pers.age)
    fmt.Println("Job: ", pers.job)
    fmt.Println("Salary: ", pers.salary)
}
```


Maps

Maps digunakan untuk menyimpan nilai data dalam pasangan kunci:nilai. Setiap elemen dalam Maps adalah pasangan kunci:nilai. Maps adalah koleksi yang tidak berurutan dan dapat diubah yang tidak memperbolehkan duplikat. Panjang Maps adalah jumlah elemennya. Anda dapat menemukannya menggunakan fungsi `len()`. Nilai default Maps adalah nol. Maps menyimpan referensi ke tabel hash yang mendasarinya. Go memiliki beberapa cara untuk membuat Maps.

Membuat Maps Menggunakan var dan :=

```
var a =  
map[KeyType]ValueType{key1:value1, key2:value2,...}  
b :=  
map[KeyType]ValueType{key1:value1, key2:value2,...}
```

Membuat Maps Menggunakan Fungsi `make()`:

```
var a = make(map[KeyType]ValueType)  
b := make(map[KeyType]ValueType)
```

Membuat Maps Kosong

Ada dua cara untuk membuat maps kosong. Salah satunya adalah dengan menggunakan fungsi **make()** dan yang lainnya adalah dengan menggunakan sintaks berikut.

```
var a map[KeyType]ValueType
```

Type Key yang Diizinkan

Kunci maps dapat berupa type data apa pun yang operator persamaan (`==`)-nya telah ditetapkan. Ini termasuk:

- Boolean
- Angka
- String
- Array
- Pointer
- Struktur

Interfaces (selama type dinamis mendukung persamaan)

Type key yang tidak valid adalah:

- Slices
- Maps
- Functions

Jenis ini tidak valid karena operator persamaan (==) tidak ditetapkan untuknya.

Type Nilai yang Diizinkan

Nilai Maps dapat berupa type apa pun.

Mengakses Elemen Maps

Kita dapat mengakses elemen maps dengan:

```
value = map_name[key]
```

Update dan Penambahan Elemen Maps

Pembaruan atau penambahan elemen dilakukan dengan cara:

```
map_name[key] = value
```

Hapus Elemen dari Maps

Penghapusan elemen dilakukan dengan menggunakan fungsi **delete()**.

```
delete(map_name, key)
```

Periksa Elemen Tertentu di Maps

Kamu dapat memeriksa apakah kunci tertentu ada di maps menggunakan:

```
val, ok :=map_name[key]
```

II. GUIDED GUIDED 1

Source Code

```
package main

import (
    "fmt"
    "sort"
)

// Struktur untuk menampung data mahasiswa
type Mahasiswa struct {
    Nama      string
    Matematika int
    Fisika    int
    Kimia     int
    RataRata  float64
}

// Fungsi untuk menghitung rata-rata nilai tiap mahasiswa
func hitungRataRata(m *Mahasiswa) {
    total := m.Matematika + m.Fisika + m.Kimia
    m.RataRata = float64(total) / 3.0
}

// Fungsi utama untuk mengelola dan mengurutkan data mahasiswa berdasarkan nilai rata-rata
func main() {
    // Array untuk menampung data mahasiswa
    mahasiswa := []Mahasiswa{
        {"Ali", 85, 90, 80, 0},
        {"Budi", 70, 75, 80, 0},
        {"Cici", 90, 85, 95, 0},
        {"Doni", 60, 65, 70, 0},
        {"Eka", 100, 95, 90, 0},
    }

    // Menghitung rata-rata nilai tiap mahasiswa
    for i := range mahasiswa {
        hitungRataRata(&mahasiswa[i])
    }
}
```

```

        // Mengurutkan mahasiswa berdasarkan nilai rata-
rata (descending)
        sort.Slice(mahasiswa, func(i, j int) bool {
            return      mahasiswa[i].RataRata      >
mahasiswa[j].RataRata
        })

        // Menampilkan hasil
        fmt.Println("Peringkat      mahasiswa      berdasarkan
rata-rata nilai:")
        for i, m := range mahasiswa {
            fmt.Printf("%d.      %s      -      Rata-rata:      %.2f
(Matematika: %d, Fisika: %d, Kimia: %d)\n",
                i+1, m>Nama, m.RataRata, m.Matematika,
m.Fisika, m.Kimia)
        }
    }
}

```

Screenshoot Output

```

[lauraneval@arco-kun laprak_6]$ go run guided_1.go
Peringkat mahasiswa berdasarkan rata-rata nilai:
1. Eka - Rata-rata: 95.00 (Matematika: 100, Fisika: 95, Kimia: 90)
2. Cici - Rata-rata: 90.00 (Matematika: 90, Fisika: 85, Kimia: 95)
3. Ali - Rata-rata: 85.00 (Matematika: 85, Fisika: 90, Kimia: 80)
4. Budi - Rata-rata: 75.00 (Matematika: 70, Fisika: 75, Kimia: 80)
5. Doni - Rata-rata: 65.00 (Matematika: 60, Fisika: 65, Kimia: 70)
[lauraneval@arco-kun laprak_6]$ _

```

Penjelasan Program

Program ini mengelola data mahasiswa dengan menyimpan nama dan nilai masing-masing dalam tiga mata pelajaran: Matematika, Fisika, dan Kimia. Menggunakan struktur data Mahasiswa, program menghitung rata-rata nilai untuk setiap mahasiswa melalui fungsi `hitungRataRata`, yang menjumlahkan nilai dari ketiga mata pelajaran dan membaginya dengan tiga. Dalam fungsi utama `main`, data mahasiswa disimpan dalam array, lalu nilai rata-rata tiap mahasiswa dihitung dan data mahasiswa diurutkan berdasarkan nilai rata-rata secara menurun menggunakan `sort.Slice`. Terakhir, program menampilkan daftar mahasiswa beserta peringkatnya, berdasarkan nilai rata-rata tertinggi hingga terendah.

GUDED 2

Source Code

```
package main

import "fmt"

func main() {

    // Membuat map dengan NIM sebagai kunci dan Nama
    sebagai nilai

    mahasiswa := map[string]string{

        "20231001": "Andi",

        "20231002": "Budi",

        "20231003": "Cici",

    }

    // Menambahkan data baru ke map

    mahasiswa["20231004"] = "Dedi"

    // Menampilkan seluruh isi map dalam format kolom
    dan baris

    fmt.Println("Daftar Mahasiswa:")

    fmt.Println("NIM\t\tNama")

    fmt.Println("-----")

    for nim, nama := range mahasiswa {

        fmt.Printf("%s\t%s\n", nim, nama)

    }

    // Mengakses data berdasarkan NIM
```

```

        nim := "20231002"

        fmt.Println("\nNama Mahasiswa dengan NIM", nim,
"adalah", mahasiswa[nim])

        // Menghapus data berdasarkan NIM

        delete(mahasiswa, "20231003")

        // Menampilkan isi map setelah data dihapus dalam
format kolom dan baris

        fmt.Println("\nDaftar          Mahasiswa          setelah
dihapus:")

        fmt.Println("NIM\t\t\tNama")

        fmt.Println("-----")

        for nim, nama := range mahasiswa {

                fmt.Printf("%s\t%s\n", nim, nama)

        }

}

```

Screenshoot Output

```

[lauraneval@arco-kun laprak_6]$ go run guided_2.go
Daftar Mahasiswa:
NIM          Nama
-----
20231001     Andi
20231002     Budi
20231003     Cici
20231004     Dedi

Nama Mahasiswa dengan NIM 20231002 adalah Budi

Daftar Mahasiswa setelah dihapus:
NIM          Nama
-----
20231001     Andi
20231002     Budi
20231004     Dedi
[lauraneval@arco-kun laprak_6]$

```

Penjelasan Program

Program ini mengelola data mahasiswa menggunakan struktur data map dengan NIM sebagai kunci dan Nama sebagai nilai. Program diawali dengan inisialisasi map berisi tiga data mahasiswa, lalu menambahkan satu data baru. Selanjutnya, program menampilkan seluruh isi map dalam format tabel dengan NIM dan Nama. Program juga dapat mengakses nama mahasiswa berdasarkan NIM tertentu dan menampilkan hasilnya. Setelah itu, satu data dihapus dari map berdasarkan NIM, dan program kembali menampilkan daftar mahasiswa yang telah diperbarui dalam format tabel.

III. UNGUIDED UNGUIDED 1

Suatu lingkaran didefinisikan dengan koordinat titik pusat (cx, cy) dengan radius r . Apabila diberikan dua buah lingkaran, maka tentukan posisi sebuah titik sembarang (x, y) berdasarkan dua lingkaran tersebut. **Gunakan tipe bentukan titik untuk menyimpan koordinat, dan tipe bentukan lingkaran untuk menyimpan titik pusat lingkaran dan radiusnya.**

Masukan terdiri dari beberapa tiga baris. Baris pertama dan kedua adalah koordinat titik pusat dan radius dari lingkaran 1 dan lingkaran 2, sedangkan baris ketiga adalah koordinat titik sembarang. Asumsi sumbu x dan y dari semua titik dan juga radius direpresentasikan dengan bilangan bulat.

Keluaran berupa string yang menyatakan posisi titik **"Titik di dalam lingkaran 1 dan 2"**, **"Titik di dalam lingkaran 1"**, **"Titik di dalam lingkaran 2"**, atau **"Titik di luar lingkaran 1 dan 2"**.

Contoh

No	Masukan	Keluaran
1	1 1 5 8 8 4 2 2	Titik di dalam lingkaran 1
2	1 2 3 4 5 6 7 8	Titik di dalam lingkaran 2
3	5 10 15 -15 4 20 0 0	Titik di dalam lingkaran 1 dan 2
4	1 1 5 8 8 4 15 20	Titik di luar lingkaran 1 dan 2

Fungsi untuk menghitung jarak titik (a, b) dan (c, d) dimana rumus jarak adalah:

$$\text{jarak} = \sqrt{(a - c)^2 + (b - d)^2}$$

dan juga fungsi untuk menentukan posisi sebuah titik sembarang berada di dalam suatu lingkaran atau tidak.

```
function jarak(p, q : titik) -> real
{Mengembalikan jarak antara titik p(x,y) dan titik q(x,y)}

function didalam(c:lingkaran, p:titik) -> boolean
{Mengembalikan true apabila titik p(x,y) berada di dalam lingkaran c yang
memiliki titik pusat (cx,cy) dan radius r}
```

Catatan: Lihat paket **math** dalam lampiran untuk menggunakan fungsi **math.Sqrt()** untuk menghitung akar kuadrat.

Source Code

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var ling1 Lingkaran
    var ling2 Lingkaran
    var ttk Titik

    fmt.Scan(&ling1.cx, &ling1.cy, &ling1.r)
    fmt.Scan(&ling2.cx, &ling2.cy, &ling2.r)
    fmt.Scan(&ttk.x, &ttk.y)

    lingkaran1 := ling1
    lingkaran2 := ling2
    titik := ttk

    hasil := cekTitik(lingkaran1, lingkaran2, titik)
    fmt.Println(hasil)
}

type Lingkaran struct {
    cx, cy, r float64
}

type Titik struct {
```

```

        x, y float64
    }

func jarak(p, q Titik) float64 {
    return math.Sqrt((p.x-q.x)(p.x-q.x) + (p.y-
q.y)(p.y-q.y))
}

func didalam(c Lingkaran, p Titik) bool {
    return jarak(Titik{c.cx, c.cy}, p) <= c.r
}

func cekTitik(c1, c2 Lingkaran, p Titik) string {
    inCircle1 := didalam(c1, p)
    inCircle2 := didalam(c2, p)

    if inCircle1 && inCircle2 {
        return "Titik di dalam lingkaran 1 dan 2"
    } else if inCircle1 {
        return "Titik di dalam lingkaran 1"
    } else if inCircle2 {
        return "Titik di dalam lingkaran 2"
    } else {
        return "Titik di luar lingkaran 1 dan 2"
    }
}

```

Screenshot Output

```
[lauraneval@arco-kun laprak_6]$ go run unguided1.go
5 10 15
-15 4 20
0 0
Titik di dalam lingkaran 1 dan 2
[lauraneval@arco-kun laprak_6]$
```

Deskripsi Program

Program ini menentukan posisi sebuah titik terhadap dua lingkaran, apakah titik tersebut berada di dalam salah satu atau kedua lingkaran, atau di luar keduanya. Program memanfaatkan struct `Lingkaran` untuk menyimpan koordinat pusat dan radius lingkaran, serta struct `Titik` untuk menyimpan koordinat titik. Fungsi `jarak` menghitung jarak antara dua titik, sedangkan fungsi `didalam` memeriksa apakah titik berada di dalam lingkaran berdasarkan jarak tersebut. Fungsi `cekTitik` kemudian memeriksa posisi titik terhadap kedua lingkaran dan mengembalikan pesan yang sesuai, misalnya "Titik di dalam lingkaran 1 dan 2" jika titik berada di dalam kedua lingkaran, atau "Titik di luar lingkaran 1 dan 2" jika titik berada di luar keduanya.

UNGUIDED 2

Sebuah array digunakan untuk menampung sekumpulan bilangan bulat. Buatlah program yang digunakan untuk mengisi array tersebut sebanyak N elemen nilai. Asumsikan array memiliki kapasitas penyimpanan data sejumlah elemen tertentu. Program dapat menampilkan beberapa informasi berikut:

- a. Menampilkan keseluruhan isi dari array.
- b. Menampilkan elemen-elemen array dengan indeks ganjil saja.
- c. Menampilkan elemen-elemen array dengan indeks genap saja (asumsi indeks ke-0 adalah genap).
- d. Menampilkan elemen-elemen array dengan indeks kelipatan bilangan x. x bisa diperoleh dari masukan pengguna.
- e. Menghapus elemen array pada indeks tertentu, asumsi indeks yang hapus selalu valid. Tampilkan keseluruhan isi dari arraynya, pastikan data yang dihapus tidak tampil
- f. Menampilkan rata-rata dari bilangan yang ada di dalam array.
- g. Menampilkan standar deviasi atau simpangan baku dari bilangan yang ada di dalam array tersebut.
- h. Menampilkan frekuensi dari suatu bilangan tertentu di dalam array yang telah diisi tersebut.

Source Code

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var n, x, index, value int

    fmt.Print("Masukkan jumlah elemen dalam array: ")

    fmt.Scan(&n)
```

```

array := make([]int, n)

fmt.Println("Masukkan elemen-elemen array:")

for i := 0; i < n; i++ {
    fmt.Scan(&array[i])
}

fmt.Println("a. Seluruh isi array:", array)
fmt.Print("b. Elemen dengan indeks ganjil: ")
for i := 1; i < n; i += 2 {
    fmt.Print(array[i], " ")
}

fmt.Println()
fmt.Print("c. Elemen dengan indeks genap: ")
for i := 0; i < n; i += 2 {
    fmt.Print(array[i], " ")
}

fmt.Println()

fmt.Print("Masukkan nilai x untuk kelipatan
indeks: ")

fmt.Scan(&x)

fmt.Print("d. Elemen dengan indeks kelipatan ", x,
": ")

for i := 0; i < n; i += x {
    fmt.Print(array[i], " ")
}

```

```

        fmt.Println()

        fmt.Print("Masukkan indeks untuk menghapus elemen:
")

        fmt.Scan(&index)

        if index >= 0 && index < n {

            array          =          append(array[:index],
array[index+1:]...)

            fmt.Println("e. Array setelah penghapusan:",
array)

        } else {

            fmt.Println("Indeks tidak valid.")

        }

        sum := 0

        for _, num := range array {

            sum += num

        }

        avg := float64(sum) / float64(len(array))

        fmt.Printf("f. Rata-rata: %.2f\n", avg)

        varianceSum := 0.0

        for _, num := range array {

            varianceSum += math.Pow(float64(num)-avg, 2)

        }

        stdDev          :=          math.Sqrt(varianceSum          /
float64(len(array)))

        fmt.Printf("g. Standar deviasi: %.2f\n", stdDev)

```

```

        fmt.Print("Masukkan nilai untuk mencari frekuensi:
")

        fmt.Scan(&value)

        frequency := 0

        for _, num := range array {
            if num == value {
                frequency++
            }
        }

        fmt.Printf("h.      Frekuensi      %d:      %d      kali\n",
value, frequency)
    }

```

Sreenshoot Output

```

[lauraneval@arco-kun laprak_6]$ go run unguided2.go
Masukkan jumlah elemen dalam array: 5
Masukkan elemen-elemen array:
1 2 3 4 5
a. Seluruh isi array: [1 2 3 4 5]
b. Elemen dengan indeks ganjil: 2 4
c. Elemen dengan indeks genap: 1 3 5
Masukkan nilai x untuk kelipatan indeks: 1
d. Elemen dengan indeks kelipatan 1: 1 2 3 4 5
Masukkan indeks untuk menghapus elemen: 4
e. Array setelah penghapusan: [1 2 3 4]
f. Rata-rata: 2.50
g. Standar deviasi: 1.12
Masukkan nilai untuk mencari frekuensi: 2
h. Frekuensi 2: 1 kali
[lauraneval@arco-kun laprak_6]$

```

Penjelasan Program

Program ini melakukan berbagai operasi pada array integer berdasarkan input pengguna, seperti menampilkan elemen-elemen array sesuai kriteria tertentu,

menghitung rata-rata dan standar deviasi, serta menghitung frekuensi kemunculan nilai tertentu. Program dimulai dengan meminta jumlah elemen array, kemudian mengisi array dengan elemen-elemen yang dimasukkan pengguna. Selanjutnya, program menampilkan seluruh isi array, elemen dengan indeks ganjil dan genap, serta elemen pada indeks kelipatan nilai tertentu. Pengguna juga dapat memilih indeks untuk menghapus elemen dari array, dan program menampilkan array yang diperbarui. Program kemudian menghitung rata-rata dan standar deviasi dari elemen-elemen yang tersisa dalam array. Akhirnya, program meminta nilai tertentu dari pengguna dan menghitung frekuensi kemunculannya dalam array, menampilkan hasilnya.

UNGUIDED 3

Sebuah program digunakan untuk menyimpan dan menampilkan nama-nama klub yang memenangkan pertandingan bola pada suatu grup pertandingan. Buatlah program yang digunakan untuk merekap skor pertandingan bola 2 buah klub bola yang berlawanan.

Pertama-tama program meminta masukan nama-nama klub yang bertanding, kemudian program meminta masukan skor hasil pertandingan kedua klub tersebut. Yang disimpan dalam array adalah nama-nama klub yang menang saja.

Proses input skor berhenti ketika skor salah satu atau kedua klub tidak valid (negatif). Di akhir program, tampilkan daftar klub yang memenangkan pertandingan.

Perhatikan sesi interaksi pada contoh berikut ini (teks bergaris bawah adalah input/read)

```
Klub A : MU
Klub B : Inter
Pertandingan 1 : 2 0           // MU = 2 sedangkan Inter = 0
Pertandingan 2 : 1 2
Pertandingan 3 : 2 2
Pertandingan 4 : 0 1
Pertandingan 5 : 3 2
Pertandingan 6 : 1 0
Pertandingan 7 : 5 2
Pertandingan 8 : 2 3
Pertandingan 9 : -1 2
Hasil 1 : MU
Hasil 2 : Inter
Hasil 3 : Draw
Hasil 4 : Inter
Hasil 5 : MU
Hasil 6 : MU
Hasil 7 : MU
Hasil 8 : Inter
Pertandingan selesai
```

Source Code

```
package main

import "fmt"

func main() {

    var clubA, clubB string
```

```
var scoreA, scoreB int
var results []string

fmt.Print("Klub A: ")
fmt.Scanln(&clubA)
fmt.Print("Klub B: ")
fmt.Scanln(&clubB)

matchCount := 1

for {
    fmt.Printf("Pertandingan %d: ", matchCount)
    fmt.Scan(&scoreA, &scoreB)

    if scoreA < 0 || scoreB < 0 {
        break
    }

    if scoreA > scoreB {
        results = append(results,
fmt.Sprintf("Hasil %d: %s", matchCount, clubA))
    } else if scoreB > scoreA {
        results = append(results,
fmt.Sprintf("Hasil %d: %s", matchCount, clubB))
    } else {
        results = append(results,
fmt.Sprintf("Hasil %d: Draw", matchCount))
    }
}
```

```

        matchCount++

    }

    for _, result := range results {

        fmt.Println(result)

    }

    fmt.Println("Pertandingan selesai")
}

```

Screenshoot Output

```

[lauraneval@arco-kun laprak_6]$ go run unguided3.go
Klub A: MU
Klub B: Inter
Pertandingan 1: 2 0
Pertandingan 2: 1 2
Pertandingan 3: 2 2
Pertandingan 4: 0 1
Pertandingan 5: 3 2
Pertandingan 6: 1 0
Pertandingan 7: 5 2
Pertandingan 8: 2 3
Pertandingan 9: -1 2
Hasil 1: MU
Hasil 2: Inter
Hasil 3: Draw
Hasil 4: Inter
Hasil 5: MU
Hasil 6: MU
Hasil 7: MU
Hasil 8: Inter
Pertandingan selesai
[lauraneval@arco-kun laprak_6]$ _

```

Penjelasan Program

Program ini mencatat hasil pertandingan antara dua klub sepak bola, clubA dan clubB, berdasarkan skor yang dimasukkan pengguna. Program pertama-tama meminta nama kedua klub, lalu memasukkan skor untuk setiap pertandingan berturut-turut. Jika skor yang dimasukkan negatif, program berhenti menerima input. Setiap pertandingan dicatat dengan penentuan pemenang berdasarkan skor: jika scoreA lebih besar, clubA menang; jika scoreB lebih besar, clubB menang; dan jika sama, hasilnya "Draw." Hasil setiap pertandingan disimpan dalam slice results, dan setelah selesai, program mencetak seluruh hasil pertandingan serta pesan "Pertandingan selesai."

UNGUIDED 4

Sebuah array digunakan untuk menampung sekumpulan karakter, Anda diminta untuk membuat sebuah subprogram untuk melakukan membalikkan urutan isi array dan memeriksa apakah membentuk palindrom.

Lengkapi potongan algoritma berikut ini!

```
package main
import "fmt"
const NMAX int = 127
type tabel [NMAX]rune
    tab : tabel
    m : integer

func isiArray(t *tabel, n *int)
/*I.S. Data tersedia dalam piranti masukan
F.S. Array t berisi sejumlah n karakter yang dimasukkan user,
Proses input selama karakter bukanlah TITIK dan n <= NMAX */
```

```
func cetakArray(t tabel, n int)
/*I.S. Terdefinisi array t yang berisi sejumlah n karakter
F.S. n karakter dalam array muncul di layar */

func balikanArray(t *tabel, n int)
/*I.S. Terdefinisi array t yang berisi sejumlah n karakter
F.S. Urutan isi array t terbalik */

func main(){
    var tab tabel
    var m int
    // si array tab dengan memanggil prosedur isiArray

    // Balikian isi array tab dengan memanggil balikanArray

    // Cetak isi array tab
}
```

Perhatikan sesi interaksi pada contoh berikut ini (**teks bergaris bawah adalah Input/read**)

```
Teks      : S E N A N G .
Reverse teks : G N A N E S

Teks      : K A T A K .
Reverse teks : K A T A K
```

Modifikasi program tersebut dengan menambahkan fungsi palindrom. Tambahkan instruksi untuk memanggil fungsi tersebut dan menampilkan hasilnya pada program utama.

***Palindrom adalah teks yang dibaca dari awal atau akhir adalah sama, contoh: KATAK, APA, KASUR_RUSAK.**

```
func palindrom(t tabel, n int) bool
/* Mengembalikan true apabila susunan karakter di dalam t membentuk palindrom,
dan false apabila sebaliknya. Petunjuk: Manfaatkan prosedur balikanArray */
```

Perhatikan sesi interaksi pada contoh berikut ini (**teks bergaris bawah adalah Input/read**)

```
Teks      : K A T A K
Palindrom : ? true

Teks      : S E N A N G
Palindrom : ? false
```

Source Code

```
package main

import "fmt"

const NMAX int = 127

type tabel struct {
    tab [NMAX]rune
    m    int
}

func isiArray(t *tabel, n *int) {
    fmt.Println("Masukkan teks (akhiri dengan TITIK):")
    for {
        var input rune
        fmt.Scanf("%c", &input)
        if input == '.' || *n >= NMAX {
            break
        }

        if input != ' ' {
            t.tab[*n] = input
            *n++
        }
    }

    t.m = *n
}
```

```

}

func cetakArray(t tabel, n int) {
    for i := 0; i < n; i++ {
        fmt.Print(string(t.tab[i]))
    }
    fmt.Println()
}

func balikkanArray(t *tabel, n int) {
    for i := 0; i < n/2; i++ {
        t.tab[i], t.tab[n-i-1] = t.tab[n-i-1],
t.tab[i]
    }
}

func palindrome(t tabel, n int) bool {
    for i := 0; i < n/2; i++ {
        if t.tab[i] != t.tab[n-i-1] {
            return false
        }
    }
    return true
}

func main() {
    var t tabel

```

```

var n int

isiArray(&t, &n)

fmt.Print("Teks: ")

cetakArray(t, n)

fmt.Print("Reverse teks: ")

balikkanArray(&t, n)

cetakArray(t, n)

if palindrome(t, n) {
    fmt.Println("Palindrome: true")
} else {
    fmt.Println("Palindrome: false")
}
}

```

Sreenchoot Output

```

[lauraneval@arco-kun laprak_6]$ go run unguided4.go
Masukkan teks (akhiri dengan TITIK):
K A T A K .
Teks: KATAK
Reverse teks: KATAK
Palindrome: true
[lauraneval@arco-kun laprak_6]$

```

Penjelasan Program

Program ini membaca teks yang dimasukkan pengguna hingga karakter titik (.), menyimpannya dalam array, kemudian membalik dan memeriksa apakah teks tersebut adalah palindrome (teks yang sama ketika dibaca dari depan atau belakang). Program menggunakan struct tabel dengan array tab untuk menyimpan karakter-karakter teks dan m untuk mencatat jumlah karakter yang dimasukkan. Fungsi isiArray mengisi array dengan karakter yang dimasukkan, mengabaikan spasi. Fungsi cetakArray menampilkan isi array, dan fungsi balikkanArray

membalik urutan karakter dalam array. Fungsi palindrome memeriksa apakah teks dalam array adalah palindrome. Di dalam main, program membaca teks, mencetaknya, membalik urutannya, mencetak teks yang sudah dibalik, dan kemudian menentukan apakah teks tersebut palindrome, menampilkan hasilnya.

UNGUIDED 5

Buatlah program yang mengimplementasikan rekursif untuk menampilkan barisan bilangan ganjil.

Masukan terdiri dari sebuah bilangan bulat positif N.

Keluaran terdiri dari barisan bilangan ganjil dari 1 hingga N.

Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	5	1 3 5
2	20	1 3 5 7 9 11 13 15 17 19

Source Code

```
package main

import "fmt"

func cetakGanjil(n, current int) {
    if current > n {
        return
    }

    fmt.Print(current, " ")
    cetakGanjil(n, current+2)
}

func main() {
    var n int

    fmt.Print("Masukkan bilangan bulat positif N: ")
    fmt.Scan(&n)

    cetakGanjil(n, 1)
}
```

Screenshoot Output

```
[lauraneval@arco-kun laprak_5]$ go run unguided_5.go
Masukkan bilangan bulat positif N: 20
1 3 5 7 9 11 13 15 17 19 [lauraneval@arco-kun laprak_5]$ S
```

Penjelasan Program

Program meminta pengguna untuk memasukkan sebuah bilangan bulat positif n . Nilai n ini menentukan batas akhir dari deret bilangan ganjil yang akan dicetak.

Fungsi Rekursif cetakGanjil:

- Fungsi cetakGanjil(n , $current$) bertujuan untuk mencetak bilangan ganjil secara rekursif.
-
- Base Case: Jika $current$ lebih besar dari n , rekursi akan berhenti (mengakhiri proses pencetakan).
- Rekursif Call: Program mencetak nilai $current$ (bilangan ganjil), lalu memanggil fungsi cetakGanjil lagi dengan nilai $current + 2$. Ini memastikan setiap angka yang dicetak adalah ganjil karena mulai dari 1 dan bertambah 2 setiap kali.

Setelah pengguna memasukkan nilai n , program memanggil fungsi cetakGanjil dengan $current$ dimulai dari 1. Fungsi akan mencetak semua bilangan ganjil dari 1 hingga batas yang kurang atau sama dengan n .

UNGUIDED 6

Buatlah program yang mengimplementasikan rekursif untuk mencari hasil pangkat dari dua buah bilangan.

Masukan terdiri dari bilangan bulat x dan y.

Keluaran terdiri dari hasil x dipangkatkan y.

Catatan: diperbolehkan menggunakan asterik "*", tapi dilarang menggunakan import "math".

Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	2 2	4
2	5 3	125

Source Code

```
package main

import "fmt"

func power(x, y int) int {
    if y == 0 {
        return 1
    }
    return x * power(x, y-1)
}

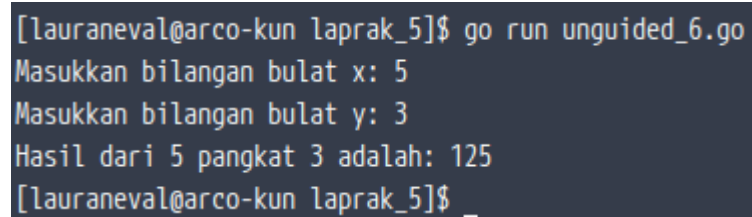
func main() {
    var x, y int
    fmt.Print("Masukkan bilangan bulat x: ")
    fmt.Scan(&x)
    fmt.Print("Masukkan bilangan bulat y: ")
    fmt.Scan(&y)
```

```
    result := power(x, y)

    fmt.Printf("Hasil      dari      %d      pangkat      %d
adalah: %d\n", x, y, result)

}
```

Screenshoot Output



```
[lauraneval@arco-kun laprak_5]$ go run unguided_6.go
Masukkan bilangan bulat x: 5
Masukkan bilangan bulat y: 3
Hasil dari 5 pangkat 3 adalah: 125
[lauraneval@arco-kun laprak_5]$ _
```

Penjelasan Program

Program meminta pengguna untuk memasukkan dua bilangan bulat: x (basis) dan y (eksponen). Nilai x dan y ini akan digunakan untuk menghitung x pangkat y.

Fungsi Rekursif power:

- Fungsi power(x, y) bertujuan untuk menghitung hasil dari x pangkat y secara rekursif.
- Base Case: Jika y sama dengan 0, fungsi mengembalikan 1, karena setiap bilangan yang dipangkatkan 0 hasilnya adalah 1.
- Rekursif Call: Jika y lebih besar dari 0, fungsi mengalikan x dengan hasil power(x, y-1). Ini berarti fungsi terus memanggil dirinya sendiri dengan mengurangi y hingga mencapai 0.

Setelah pengguna memasukkan nilai x dan y, program memanggil fungsi power untuk menghitung x pangkat y. Hasilnya disimpan dalam variabel result, dan program akan mencetak hasil perpangkatan tersebut.