

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL VII
STRUCK & ARRAY**



Disusun Oleh :

Erwin Rivaldo Silaban/2311102248

IF-11-06

Dosen Pengampu :

Abednego Dwi Septiadi, S.Kom., M.Kom

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Dalam pemrograman komputer, struktur data merupakan komponen fundamental yang memungkinkan pengorganisasian dan penyimpanan data secara efisien. Di antara berbagai struktur data yang ada, array dan struct merupakan dua tipe yang sangat penting dan sering digunakan. Tipe-tipe ini dapat dibedakan menjadi beberapa jenis utama yang memiliki karakteristik dan kegunaan yang berbeda.

Array merupakan struktur data yang terdiri dari kumpulan elemen dengan tipe data yang sama, tersusun secara berurutan dalam memori komputer. Setiap elemen dalam array dapat diakses melalui indeks yang dimulai dari 0. Array memiliki ukuran yang tetap (fixed size) setelah dideklarasikan, yang berarti jumlah elemennya tidak dapat diubah selama program berjalan. Karakteristik ini membuat array sangat efisien untuk penggunaan memori, namun kurang fleksibel ketika ukuran data perlu diubah.

Untuk mengatasi keterbatasan array konvensional, bahasa pemrograman modern memperkenalkan konsep slice atau array dinamik. Slice merupakan referensi ke array yang dapat berubah ukurannya secara dinamis selama program berjalan. Berbeda dengan array biasa, slice memiliki fleksibilitas dalam hal ukuran dan dapat diperbesar atau diperkecil sesuai kebutuhan menggunakan fungsi seperti `append()`. Slice juga memiliki dua properti penting yaitu `length` (panjang) dan `capacity` (kapasitas), yang memungkinkan pengelolaan memori yang lebih efisien.

Tipe alias (type alias) merupakan cara untuk memberikan nama alternatif pada tipe data yang sudah ada. Konsep ini sangat berguna untuk meningkatkan keterbacaan kode dan memberikan konteks yang lebih jelas tentang penggunaan suatu tipe data. Misalnya, kita dapat membuat alias "Celsius" untuk tipe data `float64` yang digunakan untuk menyimpan suhu, sehingga maksud penggunaan variabel tersebut menjadi lebih jelas.

Struct atau record merupakan tipe data yang memungkinkan penggabungan beberapa tipe data berbeda dalam satu kesatuan. Berbeda dengan array yang hanya dapat menyimpan data dengan tipe yang sama, struct dapat menampung berbagai tipe data dalam field-field yang berbeda. Setiap field dalam struct memiliki nama dan tipe data tersendiri, memungkinkan organisasi data yang lebih terstruktur dan bermakna. Struct sangat berguna ketika kita perlu merepresentasikan entitas dengan berbagai atribut yang berbeda, seperti data person yang memiliki nama (string), umur (integer), dan alamat (string).

Map merupakan struktur data yang menyimpan pasangan key-value, di mana setiap key harus unik dalam map tersebut. Berbeda dengan array atau slice yang

menggunakan indeks numerik, map menggunakan key untuk mengakses value-nya. Key dalam map dapat berupa tipe data comparable (yang dapat dibandingkan), sementara value dapat berupa tipe data apapun. Map sangat efektif untuk implementasi kamus, cache, atau situasi di mana kita perlu mengasosiasikan suatu nilai dengan identifier unik.

Setiap struktur data ini memiliki kelebihan dan kegunaan yang berbeda. Array dan slice ideal untuk data berurutan dengan tipe yang sama, struct cocok untuk merepresentasikan entitas dengan berbagai atribut, sedangkan map sangat berguna untuk asosiasi key-value. Pemilihan struktur data yang tepat tergantung pada kebutuhan spesifik aplikasi, seperti efisiensi memori, fleksibilitas, dan kemudahan akses data. Pemahaman yang baik tentang karakteristik masing-masing struktur data ini akan membantu dalam merancang program yang efisien dan mudah dipelihara.

II. GUIDED

Soal Studi Case

XXXXXXXXXXXXXXXXXX

Sourcecode

```
package main
// Erwin Rivaldo Silaban
// 2311102248

import (
    "fmt"
    "math"
)

type Titik struct {
    x int
    y int
}

type Lingkaran struct {
    pusat Titik
    radius int
}

func hitungJarak(a, b Titik) float64 {
    return math.Sqrt(float64((a.x-b.x)*(a.x-
b.x) + (a.y-b.y)*(a.y-b.y)))
}

func titikDalamLingkaran(t Titik, l Lingkaran)
bool {
    jarak := hitungJarak(t, l.pusat)
    return jarak <= float64(l.radius)
}

func main() {
    var cx1, cy1, r1 int
    fmt.Print("Masukkan Koordinat Pusat dan
Radius Lingkaran1 (cx1 cy1 r1): ")
    fmt.Scanln(&cx1, &cy1, &r1)
    lingkaran1 := Lingkaran{pusat: Titik{x:
cx1, y: cy1}, radius: r1}
```

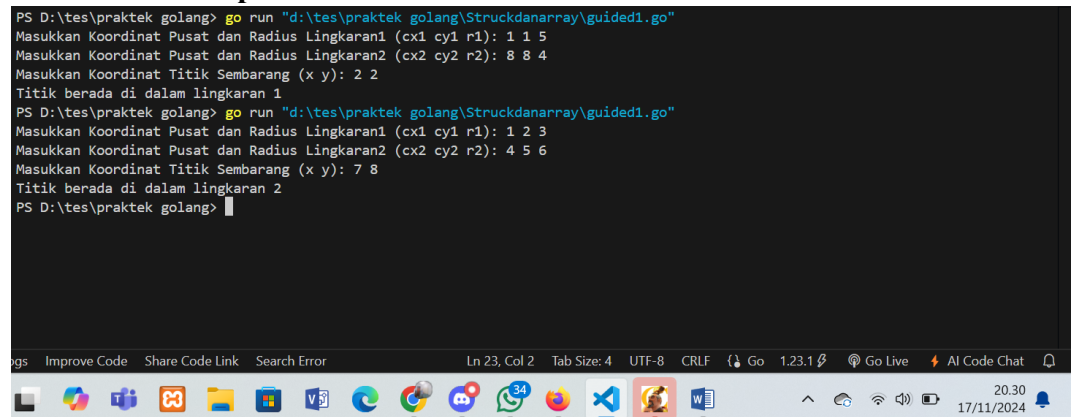
```
    var cx2, cy2, r2 int
    fmt.Print("Masukkan Koordinat Pusat dan
Radius Lingkaran2 (cx2 cy2 r2): ")
    fmt.Scanln(&cx2, &cy2, &r2)
    lingkaran2 := Lingkaran{pusat: Titik{x:
cx2, y: cy2}, radius: r2}

    var x, y int
    fmt.Print("Masukkan Koordinat Titik
Sembarang (x y): ")
    fmt.Scanln(&x, &y)
    titik := Titik{x: x, y: y}

    diDalam1 := titikDalamLingkaran(titik,
lingkaran1)
    diDalam2 := titikDalamLingkaran(titik,
lingkaran2)

    if diDalam1 && diDalam2 {
        fmt.Println("Titik berada di dalam
lingkaran 1 dan 2")
    } else if diDalam1 {
        fmt.Println("Titik berada di dalam
lingkaran 1")
    } else if diDalam2 {
        fmt.Println("Titik berada di dalam
lingkaran 2")
    } else {
        fmt.Println("Titik tidak berada di
dalam kedua lingkaran")
    }
}
```

Screenshoot Output

A screenshot of a Windows terminal window running a Go program. The terminal shows the execution of a program that prompts for coordinates and radii of two circles, and then checks if a point lies within either circle. The program is run from a directory path 'd:\tes\praktek golang'. The output shows two test cases. In the first, a point (2, 2) is inside a circle with center (1, 1) and radius 5. In the second, a point (7, 8) is inside a circle with center (4, 5) and radius 6. The terminal window has a dark background and a status bar at the bottom showing file encoding (UTF-8) and line endings (CRLF). The Windows taskbar is visible at the bottom of the screen.

```
PS D:\tes\praktek golang> go run "d:\tes\praktek golang\Struckdanarray\guided1.go"
Masukkan Koordinat Pusat dan Radius Lingkaran1 (cx1 cy1 r1): 1 1 5
Masukkan Koordinat Pusat dan Radius Lingkaran2 (cx2 cy2 r2): 8 8 4
Masukkan Koordinat Titik Sembarang (x y): 2 2
Titik berada di dalam lingkaran 1
PS D:\tes\praktek golang> go run "d:\tes\praktek golang\Struckdanarray\guided1.go"
Masukkan Koordinat Pusat dan Radius Lingkaran1 (cx1 cy1 r1): 1 2 3
Masukkan Koordinat Pusat dan Radius Lingkaran2 (cx2 cy2 r2): 4 5 6
Masukkan Koordinat Titik Sembarang (x y): 7 8
Titik berada di dalam lingkaran 2
PS D:\tes\praktek golang>
```

Deskripsi Program

Program di awali dengan mendefinisikan dua struktur data, yaitu Titik yang merepresentasikan koordinat titik dengan atribut x dan y, serta Lingkaran yang memiliki pusat bertipe Titik dan radius bertipe integer. Fungsi `hitungJarak` digunakan untuk menghitung jarak antara dua titik menggunakan rumus jarak Euclidean. Selanjutnya, fungsi `titikDalamLingkaran` memeriksa apakah suatu titik berada di dalam lingkaran dengan membandingkan jarak titik tersebut dari pusat lingkaran dengan radius lingkaran itu sendiri.

Di dalam fungsi main, program meminta pengguna untuk memasukkan koordinat pusat dan radius dari dua lingkaran, serta koordinat dari titik sembarang. Setelah data dimasukkan, program akan mengevaluasi apakah titik tersebut berada di dalam salah satu atau kedua lingkaran dengan memanggil fungsi `titikDalamLingkaran`. Hasil evaluasi kemudian ditampilkan ke layar, memberikan informasi apakah titik tersebut berada di dalam lingkaran pertama, kedua, atau tidak berada di dalam kedua lingkaran tersebut.

III. UNGUIDED

1. Soal Studi Case

XXXXXXXXXXXXXXXXXX

Sourcecode

```
package main
// Erwin Rivaldo Silaban
// 2311102248
import (
    "fmt"
    "math"
)

func main() {
    var erwin int
    fmt.Println("Masukkan Jumlah Elemen Dalam Array: ")
    fmt.Scanln(&erwin)

    numbers := make([]float64, erwin)

    for i := 0; i < erwin; i++ {
        fmt.Printf("Masukkan elemen ke-%d: ", i)
        fmt.Scan(&numbers[i])
    }

    for {
        fmt.Println("\nMenu")
        fmt.Println("1. Tampilkan Seluruh Isi Elemen
Array")
        fmt.Println("2. Tampilkan Elemen Indeks Ganjil")
        fmt.Println("3. Tampilkan Indeks Genap")
        fmt.Println("4. Menampilkan Elemen Array
Berdasarkan Kelipatan Indeks")
        fmt.Println("5. Menghapus Elemen Array Pada
Indeks Tertentu")
        fmt.Println("6. Menampilkan Rata-rata")
        fmt.Println("7. Tampilkan Standar Deviasi")
        fmt.Println("8. Menampilkan Frekuensi")
        fmt.Println("9. Keluar")

        var pilihan int
        fmt.Println("Masukkan Pilihan 1-9: ")
        fmt.Scan(&pilihan)

        switch pilihan {
        case 1:
            for i, num := range numbers {
                if num != math.MaxFloat64 {
                    fmt.Printf("Index %d: %.2f\n", i,
num)
                }
            }
        }
    }
}
```

```

    }

    case 2:
        for i, num := range numbers {
            if i%2 != 0 && num != math.MaxFloat64 {
                fmt.Printf("Index %d: %.2f\n", i,
num)
            }
        }

    case 3:
        for i, num := range numbers {
            if i%2 == 0 && num != math.MaxFloat64 {
                fmt.Printf("Index %d: %.2f\n", i,
num)
            }
        }

    case 4:
        var x int
        fmt.Print("Masukkan Nilai X: ")
        fmt.Scan(&x)
        fmt.Printf("\nElemen dengan indeks kelipatan
%d:\n", x)
        for i, num := range numbers {
            if i%x == 0 && num != math.MaxFloat64 {
                fmt.Printf("Index %d: %.2f\n", i,
num)
            }
        }

    case 5:
        var idx int
        fmt.Print("Masukkan indeks yang akan
dihapus: ")
        fmt.Scan(&idx)
        if idx >= 0 && idx < len(numbers) {
            numbers[idx] = math.MaxFloat64
            fmt.Printf("Elemen pada indeks %d telah
dihapus\n", idx)
        } else {
            fmt.Println("Indeks tidak valid")
        }

    case 6:
        sum := 0.0
        count := 0
        for _, num := range numbers {
            if num != math.MaxFloat64 {
                sum += num
                count++
            }
        }

```



```

    }
    if count > 0 {
        fmt.Printf("Rata-rata: %.2f\n",
sum/float64(count))
    } else {
        fmt.Println("Array kosong")
    }

case 7 :
    var mean, sumSquares float64
    count := 0
    for _, num := range numbers {
        if num != math.MaxFloat64 {
            mean += num
            count++
        }
    }
    if count > 0 {
        mean /= float64(count)
        for _, num := range numbers {
            if num != math.MaxFloat64 {
                sumSquares += math.Pow(num-mean,
2)

            }
        }

        stdDev := math.Sqrt(sumSquares /
float64(count))
        fmt.Printf("Standar Deviasi: %.2f\n",
stdDev)
    } else {
        fmt.Println("Array kosong")
    }

case 8:
    var target float64
    fmt.Print("Masukkan bilangan yang ingin
dihitung frekuensinya: ")
    fmt.Scan(&target)

    freq := 0
    for _, num := range numbers {
        if num == target {
            freq++
        }
    }
    fmt.Printf("Frekuensi %.2f dalam array:
%d\n", target, freq)

case 9:
    fmt.Println("Program selesai")
    return

```

```

        default:
            fmt.Println(" Pilihan Tidak Tersedia ")
        }
    }
}

```

Screenshoot Output

- Menampilkan seluruh isi array

```

PROBLEMS 26 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR Code
PS D:\tes\praktek golang> go run "d:\tes\praktek golang\Struckdanarray\guided2.go"
Masukkan Jumlah Elemen Dalam Array:
5
Masukkan elemen ke-0: 1
Masukkan elemen ke-1: 2
Masukkan elemen ke-2: 3
Masukkan elemen ke-3: 4
Masukkan elemen ke-4: 5

Menu
1. Tampilkan Seluruh Isi Elemen Array
2. Tampilkan Elemen Indeks Ganjil
3. Tampilkan Indeks Genap
4. Menampilkan Elemen Array Berdasarkan Kelipatan Indeks
5. Menghapus Elemen Array Pada Indeks Tertentu
6. Menampilkan Rata-rata
7. Tampilkan Standar Deviasi
8. Menampilkan Frekuensi
9. Keluar
Masukkan Pilihan 1-9:
1
Index 0: 1.00
Index 1: 2.00
Index 2: 3.00
Index 3: 4.00
Index 4: 5.00

```

- Menampilkan elemen dengan indeks ganjil

```

Masukkan Pilihan 1-9:
2
Elemen pada indeks ganjil:
Index 1: 2.00
Index 3: 4.00

```

- Menampilkan elemen pada indeks genap

```
Masukkan Pilihan 1-9:  
3  
Elemen pada indeks genap:  
Index 0: 1.00  
Index 2: 3.00  
Index 4: 5.00
```

- Menampilkan elemen array dengan kelipatan x

```
Masukkan Pilihan 1-9:  
4  
Masukkan Nilai X: 2  
  
Elemen dengan indeks kelipatan 2:  
Index 0: 1.00  
Index 2: 3.00  
Index 4: 5.00
```

- Menghapus elemen array pada indeks tertentu

```
Masukkan Pilihan 1-9:  
5  
Masukkan indeks yang akan dihapus: 3  
Elemen pada indeks 3 telah dihapus  
  
Menu  
1. Tampilkan Seluruh Isi Elemen Array  
2. Tampilkan Elemen Indeks Ganjil  
3. Tampilkan Elemen Indeks Genap  
4. Menampilkan Elemen Array Berdasarkan Kelipatan Indeks  
5. Menghapus Elemen Array Pada Indeks Tertentu  
6. Menampilkan Rata-rata  
7. Tampilkan Standar Deviasi  
8. Menampilkan Frekuensi  
9. Keluar  
Masukkan Pilihan 1-9:  
1  
Index 0: 1.00  
Index 1: 2.00  
Index 2: 3.00  
Index 3: 5.00
```

- Menampilkan rata rat array

```
Masukkan Pilihan 1-9:
6
Rata-rata: 2.75
Menu
```

- Menampilkan standar deviasi

```
Masukkan Pilihan 1-9:
7
Standar Deviasi: 1.48
```

- Menampilkan frekuensi

```
Masukkan Pilihan 1-9:
8
Masukkan bilangan yang ingin dihitung frekuensinya: 5
Frekuensi 5.00 dalam array: 1
```

Deskripsi Program

pengguna diminta untuk menentukan jumlah elemen dalam array dan mengisi elemen-elemen tersebut. Setelah itu, program menyediakan menu interaktif dengan berbagai opsi, termasuk menampilkan seluruh isi array, menampilkan elemen pada indeks ganjil atau genap, menampilkan elemen berdasarkan kelipatan indeks tertentu, menghapus elemen pada indeks yang ditentukan, serta menghitung rata-rata, standar deviasi, dan frekuensi kemunculan suatu bilangan dalam array. Program akan terus berjalan hingga pengguna memilih untuk keluar dengan memasukkan pilihan 9. Dengan demikian, program ini memungkinkan pengguna untuk melakukan berbagai operasi analisis data sederhana pada array yang telah diinput.

2. Studi Case

XXXXXXXXXXXXX

Source code

```
package main

// Erwin Rivaldo Silaban
// 2311102248

import (
```

```

        "fmt"
    )

    func main() {
        var klubA, klubB string
        var skorA, skorB int
        var pemenang []string

        fmt.Print("Masukkan nama Klub A: ")
        fmt.Scanln(&klubA)
        fmt.Print("Masukkan nama Klub B: ")
        fmt.Scanln(&klubB)

        for i := 1; ; i++ {
            fmt.Printf("Pertandingan %d - Masukkan skor %v dan %v : ", i, klubA, klubB)
            fmt.Scan(&skorA, &skorB)

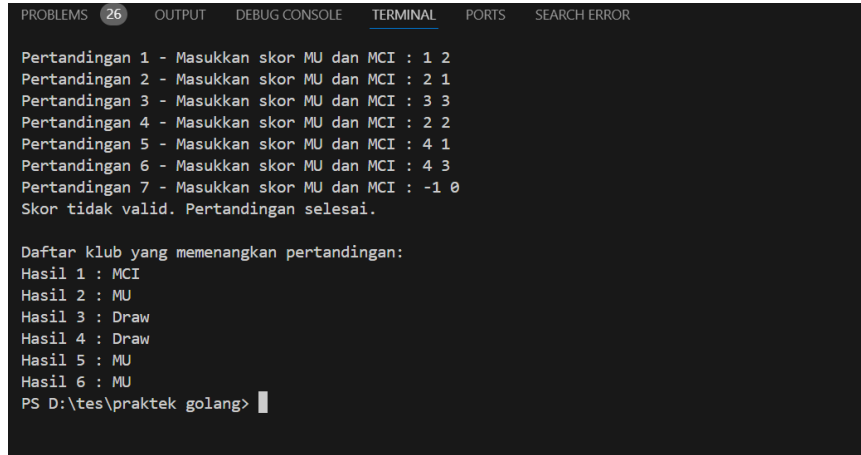
            if skorA < 0 || skorB < 0 {
                fmt.Println("Skor tidak valid. Pertandingan selesai.")
                break
            }

            if skorA > skorB {
                pemenang = append(pemenang, klubA)
            } else if skorB > skorA {
                pemenang = append(pemenang, klubB)
            } else {
                pemenang = append(pemenang,
                "Draw")
            }
        }

        fmt.Println("\nDaftar klub yang memenangkan pertandingan:")
        for i := 1; i <= len(pemenang); i++ {
            fmt.Printf("Hasil %v : %v\n", i,
            pemenang[i-1])
        }
    }

```

Screenshot Output



```
PROBLEMS 26 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

Pertandingan 1 - Masukkan skor MU dan MCI : 1 2
Pertandingan 2 - Masukkan skor MU dan MCI : 2 1
Pertandingan 3 - Masukkan skor MU dan MCI : 3 3
Pertandingan 4 - Masukkan skor MU dan MCI : 2 2
Pertandingan 5 - Masukkan skor MU dan MCI : 4 1
Pertandingan 6 - Masukkan skor MU dan MCI : 4 3
Pertandingan 7 - Masukkan skor MU dan MCI : -1 0
Skor tidak valid. Pertandingan selesai.

Daftar klub yang memenangkan pertandingan:
Hasil 1 : MCI
Hasil 2 : MU
Hasil 3 : Draw
Hasil 4 : Draw
Hasil 5 : MU
Hasil 6 : MU
PS D:\tes\praktek golang>
```

Deskripsi Program

Program ini meminta pengguna untuk memasukkan nama dua klub sepak bola, yaitu klubA dan klubB. Setelah itu, program memasuki loop yang meminta pengguna untuk memasukkan skor untuk setiap pertandingan antara kedua klub. Loop ini akan terus berlanjut hingga pengguna memasukkan skor negatif, yang menandakan bahwa input telah selesai. Setelah menerima input skor, program memvalidasi apakah skor tersebut valid (tidak negatif). Jika salah satu skor negatif, loop akan berhenti. Program kemudian membandingkan skor: jika skorA lebih tinggi, klubA dicatat sebagai pemenang; jika skorB lebih tinggi, klubB yang dicatat sebagai pemenang; dan jika kedua skor sama, hasilnya dicatat sebagai "Draw". Setelah loop selesai, program mencetak daftar hasil pertandingan yang menunjukkan pemenang untuk setiap pertandingan. Dengan demikian, program ini memungkinkan pengguna untuk secara interaktif mencatat dan melihat hasil pertandingan antara dua klub sepak bola.

3. Studi Case

XXXXXXXXXXXX

Source Code

```
package main
// Erwin Rivaldo Silaban
```

```

// 2311102248
import "fmt"

const NMAX int = 127

type tabel [NMAX]rune

func isiArray(t *tabel, n *int) {
    var kar rune
    fmt.Print("Masukkan Teks (akhiri dengan
    '.'): ")
    for i := 0; i < NMAX; i++ {
        fmt.Scanf("%c", &kar)
        if kar == '.' {
            break
        }
        t[i] = kar
        *n++
    }
}

func cetakArray(t tabel, n int) {
    for i := 0; i < n; i++ {
        fmt.Printf("%c ", t[i])
    }
    fmt.Print("\n")
}

func balikkanArray(t *tabel, n int) {
    for i := 0; i < n/2; i++ {
        temp := t[i]
        t[i] = t[n-1-i]
        t[n-1-i] = temp
    }
    fmt.Print("Teks setelah dibalik: ")
    for i := 0; i < n; i++ {
        fmt.Printf("%c ", t[i])
    }
    fmt.Print("\n")
}

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

```

```

        return true
    }

    func main() {
        var tab tabel
        var m int

        isiArray(&tab, &m)

        fmt.Print("Teks asli: ")
        cetakArray(tab, m)

        balikkanArray(&tab, m)

        if palindrom(tab, m) {
            fmt.Println("Apakah teks ini
palindrom? true")
        } else {
            fmt.Println("Apakah teks ini
palindrom? false")
        }
    }
}

```

Screenshot Output

```

PS D:\tes\praktek golang> go run "d:\tes\praktek golang\Struckdanarray\unguided4.go"
Masukkan Teks : K A T A K.
Teks : K   A   T   A   K
Reverse teks: K   A   T   A   K
Palindrom ? true
PS D:\tes\praktek golang> go run "d:\tes\praktek golang\Struckdanarray\unguided4.go"
Masukkan Teks : K U D A.
Teks : K   U   D   A
Reverse teks: A   D   U   K
Palindrom ? flase
PS D:\tes\praktek golang>

```

Deskripsi Program

Program menggunakan fungsi `isiArray` untuk mengisi array dengan karakter hingga pengguna mengetikkan titik ('.'). Fungsi `cetakArray` menampilkan isi array, sedangkan `balikkanArray` membalik urutan elemen dalam array dan menampilkan hasilnya. Fungsi `palindrom` memeriksa apakah teks yang dimasukkan adalah palindrom. Dalam fungsi `main`, program mengatur alur kerja dengan memanggil fungsi-fungsi tersebut secara berurutan: mengisi array, mencetak teks asli, membalikkan teks, dan memeriksa apakah teks tersebut adalah palindrom.