LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL 10

MATERI

Oleh:

MUHAMMAD ZAKY MUBAROK

103112400073

KELAS

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

I. DASAR TEORI

MODUL 10. PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA

II. GUIDED

```go
package main

import "fmt"

func main() {
    var n int
    fmt.Print("Masukkan Jumlah Tanaman: ")
    fmt.Scan(&n)

    var tinggiTanaman [500]float64
    for i := 0; i < n; i++ {
        fmt.Printf("Masukkan Tinggi Tanaman ke-%d(cm) :", i+1)
        fmt.Scan(&tinggiTanaman[i])
    }

    min, max := tinggiTanaman[0], tinggiTanaman[0]
    for i := 1; i < n; i++ {
        if tinggiTanaman[i] < min {
            min = tinggiTanaman[i]
        }
        if tinggiTanaman[i] > max {
            max = tinggiTanaman[i]
        }
    }

    fmt.Printf("Tinggi Tanaman Terpendek: %.2f cm\n", min)
    fmt.Printf("Tinggi Tanaman Tertinggi: %.2f cm\n", max)
}
```

```
PS D:\ALPRO 2\103112400073_MODUL10> go
Masukkan Jumlah Tanaman: 4
Masukkan Tinggi Tanaman ke-1(cm) :12
Masukkan Tinggi Tanaman ke-2(cm) :14
Masukkan Tinggi Tanaman ke-3(cm) :15
Masukkan Tinggi Tanaman ke-4(cm) :17
Tinggi Tanaman Terpendek: 12.00 cm
Tinggi Tanaman Tertinggi: 17.00 cm
```

Penjelasan :

Kode ini adalah program untuk menemukan tinggi tanaman terpendek dan tertinggi dari data yang dimasukkan pengguna dalam bahasa Go. Berikut penjelasan singkatnya:

1. **Input Jumlah Tanaman**:
   Pengguna diminta memasukkan jumlah tanaman (n).

2. **Input Tinggi Tanaman**:
   Program meminta pengguna untuk memasukkan tinggi masing-masing tanaman dalam bentuk array dengan maksimum 500 elemen.

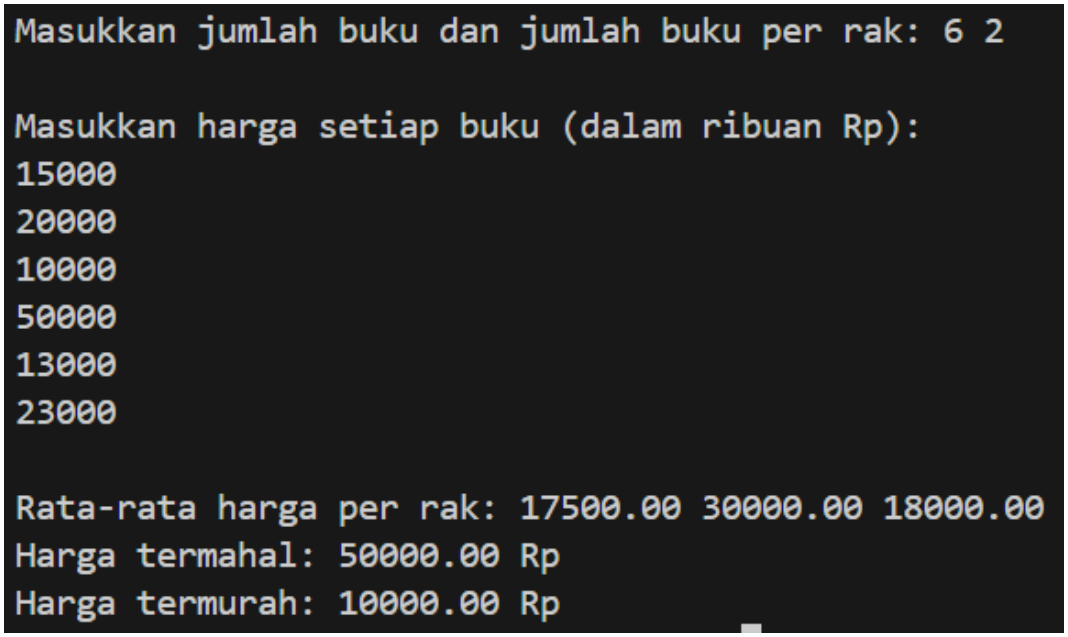
3. **Mencari Nilai Minimum dan Maksimum**:

   - Program menggunakan perulangan untuk membandingkan setiap nilai dalam array.
   - Nilai minimum (min) akan diperbarui jika ditemukan tinggi yang lebih kecil.
   - Nilai maksimum (max) akan diperbarui jika ditemukan tinggi yang lebih besar.

4. **Output Hasil**:
   Program mencetak tinggi tanaman terpendek dan tertinggi dalam format dua desimal (%.2f cm).

III. GUIDED

```go
package main

import "fmt"

func main() {
    var x, y int
    fmt.Print("Masukkan jumlah buku dan jumlah buku per rak: ")
    fmt.Scan(&x, &y)

    var hargaBuku [500]float64
    fmt.Println("\nMasukkan harga setiap buku (dalam ribuan Rp):")
    for i := 0; i < x; i++ {
        fmt.Scan(&hargaBuku[i])
    }

    var hargaRataRata []float64
    for i := 0; i < x; i += y {
        total := 0.0
        for j := i; j < i+y && j < x; j++ {
            total += hargaBuku[j]
        }
        hargaRataRata = append(hargaRataRata, total/float64(y))
    }

    min, max := hargaRataRata[0], hargaRataRata[0]
    for _, harga := range hargaBuku [:x] {
        if harga < min {
            min = harga
        }
        if harga > max {
            max = harga
        }
    }
    fmt.Printf("\nRata-rata harga per rak: ")
    for _, avg := range hargaRataRata {
        fmt.Printf("%.2f ", avg)
    }
    fmt.Printf("\nHarga termahal: %.2f Rp\n", max)
    fmt.Printf("Harga termurah: %.2f Rp\n", min)
}
```



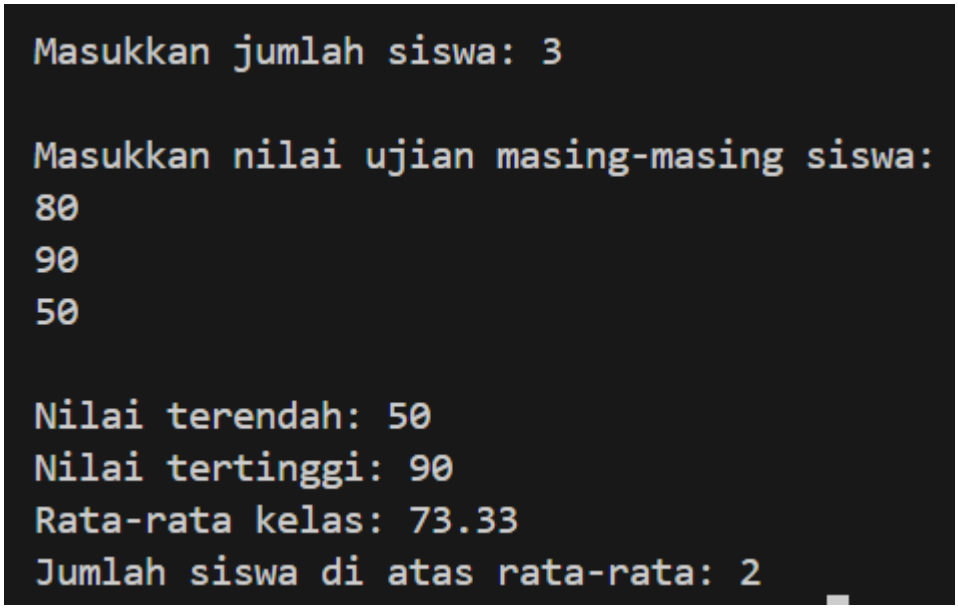
Penjelasan :

Program ini adalah aplikasi untuk menghitung rata-rata harga buku per rak, harga buku termurah, dan harga buku termahal. Berikut penjelasan singkatnya:

1. **Input Jumlah Buku dan Buku per Rak**:
   - Pengguna diminta memasukkan jumlah total buku (`x`) dan jumlah buku per rak (`y`).
2. **Input Harga Buku**:
   - Pengguna memasukkan harga setiap buku (maksimum hingga 500 buku).
3. **Menghitung Rata-rata Harga per Rak**:
   - Program menggunakan array `hargaRataRata` untuk menyimpan rata-rata harga buku pada setiap rak. Perulangan dilakukan untuk menghitung total harga buku dalam rak, kemudian rata-rata dihitung dengan membagi total harga dengan jumlah buku per rak.
4. **Menghitung Harga Buku Termahal dan Termurah**:
   - Program membandingkan harga setiap buku untuk menentukan nilai minimum (`min`) dan maksimum (`max`) dari array `hargaBuku`.
5. **Output Hasil**:
   - Program mencetak rata-rata harga buku untuk setiap rak, harga buku termahal, dan harga buku termurah dalam format dua desimal (`%.2f Rp`).

IV. GUIDED

```
package main

import "fmt"

func main() {
    var n int
    fmt.Print("Masukkan jumlah siswa: ")
    fmt.Scan(&n)

    var nilaiSiswa [200]float64
    var totalNilai float64 = 0

    fmt.Println("\nMasukkan nilai ujian masing-masing siswa:")
    for i := 0; i < n; i++ {
        fmt.Scan(&nilaiSiswa[i])
        totalNilai += nilaiSiswa[i]
    }

    rataRata := totalNilai / float64(n)

    min, max := nilaiSiswa[0], nilaiSiswa[0]
    var diAtasRataRata int = 0
    for _, nilai := range nilaiSiswa[:n] {
        if nilai < min {
            min = nilai
        }
        if nilai > max {
            max = nilai
        }
        if nilai > rataRata {
            diAtasRataRata++
        }
    }
    fmt.Printf("\nNilai terendah: %.0f\n", min)
    fmt.Printf("Nilai tertinggi: %.0f\n", max)
    fmt.Printf("Rata-rata kelas: %.2f\n", rataRata)
    fmt.Printf("Jumlah siswa di atas rata-rata: %d\n",
diAtasRataRata)
}
```



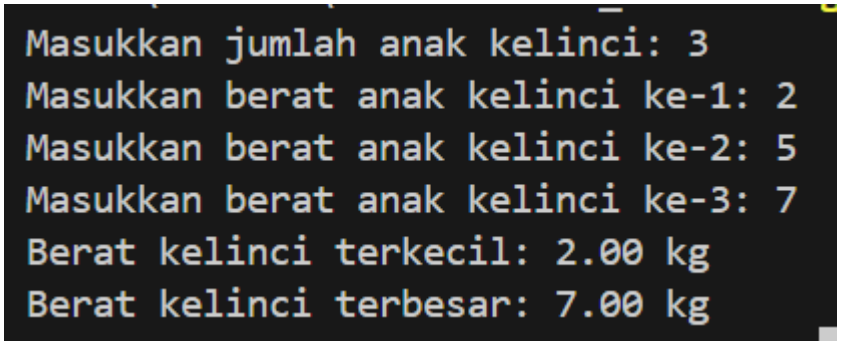
Penjelasan :

Program ini adalah aplikasi untuk menganalisis nilai ujian siswa dalam bahasa Go. Berikut penjelasan singkatnya:

1. **Input Jumlah Siswa**:
   Program meminta pengguna memasukkan jumlah siswa (`n`).
2. **Input Nilai Siswa**:
   Pengguna memasukkan nilai ujian masing-masing siswa. Nilai-nilai tersebut disimpan dalam array `nilaiSiswa`.
3. **Menghitung Rata-rata Nilai**:
   Total nilai dijumlahkan, lalu rata-rata nilai kelas dihitung dengan membagi total nilai dengan jumlah siswa.
4. **Menghitung Nilai Tertinggi dan Terendah**:
   Program mencari nilai tertinggi (`max`) dan nilai terendah (`min`) dengan membandingkan setiap nilai dalam array.
5. **Menghitung Jumlah Siswa di Atas Rata-rata**:
   Program menghitung jumlah siswa yang nilai ujiannya lebih tinggi dari rata-rata kelas.
6. **Output Hasil**:
   Program mencetak nilai terendah, nilai tertinggi, rata-rata kelas, dan jumlah siswa dengan nilai di atas rata-rata.

V. UNGUIDED I

```go
//Muhammad Zaky Mubarok
package main

import (
    "fmt"
)

func hitungMinMax(daftarBerat []float64) (float64, float64) {
    if len(daftarBerat) == 0 {
        return 0, 0
    }
    beratTerkecil := daftarBerat[0]
    beratTerbesar := daftarBerat[0]

    for _, berat := range daftarBerat[1:] {
        if berat < beratTerkecil {
            beratTerkecil = berat
        }
        if berat > beratTerbesar {
            beratTerbesar = berat
        }
    }
    return beratTerkecil, beratTerbesar
}

func main() {
    var jumlahKelinci int
    fmt.Print("Masukkan jumlah anak kelinci: ")
    fmt.Scan(&jumlahKelinci)

    daftarBerat := make([]float64, jumlahKelinci)

    for i := 0; i < jumlahKelinci; i++ {
        fmt.Printf("Masukkan berat anak kelinci ke-%d: ", i+1)
        fmt.Scan(&daftarBerat[i])
    }

    beratTerkecil, beratTerbesar := hitungMinMax(daftarBerat)

    fmt.Printf("Berat kelinci terkecil: %.2f kg\n", beratTerkecil)
    fmt.Printf("Berat kelinci terbesar: %.2f kg\n", beratTerbesar)
}
```



Penjelasan :

Program ini dalam bahasa Go digunakan untuk menganalisis berat anak kelinci dengan mencari berat terkecil dan terbesar. Berikut penjelasan singkatnya:

1. **Fungsi hitungMinMax**:
    - Mengambil sebuah daftar berat (daftarBerat) dan mengembalikan berat terkecil serta terbesar.
    - Fungsi membandingkan setiap berat dalam array untuk memperbarui nilai minimum (beratTerkecil) dan maksimum (beratTerbesar).
2. **Input Jumlah dan Berat Kelinci**:
    - Program meminta pengguna untuk memasukkan jumlah anak kelinci (jumlahKelinci).
    - Pengguna kemudian mengisi berat setiap kelinci, yang disimpan dalam array daftarBerat.
3. **Output Hasil**:
    - Memanggil fungsi hitungMinMax untuk menemukan berat terkecil dan terbesar dari daftar berat.
    - Menampilkan berat kelinci terkecil dan terbesar dalam format dua desimal (%.2f kg).

VI. UNGUIDED II

```go
//Muhammad Zaky Mubarok
package main

import (
    "fmt"
)

func hitungTotalPerWadah(daftarBerat []float64, jumlahWadah int)
[]float64 {
    totalPerWadah := make([]float64, jumlahWadah)
    for i, berat := range daftarBerat {
        indeksWadah := i % jumlahWadah
        totalPerWadah[indeksWadah] += berat
    }
    return totalPerWadah
}

func hitungRataRata(daftar []float64) float64 {
    var total float64
    for _, nilai := range daftar {
        total += nilai
    }
    return total / float64(len(daftar))
}

func main() {
    var jumlahIkan, jumlahWadah int
    fmt.Print("Masukkan jumlah ikan yang akan dijual (x): ")
    fmt.Scan(&jumlahIkan)
    fmt.Print("Masukkan jumlah ikan per wadah (y): ")
    fmt.Scan(&jumlahWadah)

    daftarBerat := make([]float64, jumlahIkan)
    for i := 0; i < jumlahIkan; i++ {
        fmt.Printf("Masukkan berat ikan ke-%d: ", i+1)
        fmt.Scan(&daftarBerat[i])
    }

    totalBeratPerWadah := hitungTotalPerWadah(daftarBerat,
jumlahWadah)
    rataRataBeratWadah := hitungRataRata(totalBeratPerWadah)
```

```
    fmt.Println("Total berat ikan per wadah:")
    for i, berat := range totalBeratPerWadah {
        fmt.Printf("Wadah %d: %.2f kg\n", i+1, berat)
    }

    fmt.Printf("Rata-rata berat ikan per wadah: %.2f kg\n",
rataRataBeratWadah)
}
```

```
Masukkan jumlah ikan yang akan dijual (x): 4
Masukkan jumlah ikan per wadah (y): 2
Masukkan berat ikan ke-1: 12
Masukkan berat ikan ke-2: 5
Masukkan berat ikan ke-3: 13
Masukkan berat ikan ke-4: 7
Total berat ikan per wadah:
Wadah 1: 25.00 kg
Wadah 2: 12.00 kg
Rata-rata berat ikan per wadah: 18.50 kg
```

Penjelasan :

Program ini dalam bahasa Go digunakan untuk menghitung berat ikan yang didistribusikan ke beberapa wadah dan rata-rata berat per wadah. Berikut penjelasannya:

1. **Fungsi hitungTotalPerWadah:**
    - **Menghitung total berat ikan di setiap wadah berdasarkan pembagian indeks.**
    - **Berat ikan ditambahkan ke wadah berdasarkan indeks modulo jumlah wadah.**
2. **Fungsi hitungRataRata:**
    - **Menghitung rata-rata berat dengan menjumlahkan semua elemen dalam daftar dan membaginya dengan jumlah elemen.**
3. **Input Data:**

- **Pengguna memasukkan jumlah ikan (jumlahIkan) dan jumlah wadah (jumlahWadah).**
- **Berat ikan per ekor dimasukkan ke dalam array daftarBerat.**

4. **Output:**
   - **Total berat ikan di masing-masing wadah dicetak.**
   - **Rata-rata berat ikan per wadah juga ditampilkan dalam format dua desimal (%.2f kg).**

VII. UNGUIDED III

```go
//Muhammad Zaky Mubarok
package main

import (
    "fmt"
)

func hitungMinMax(beratBalita []float64, min *float64, max *float64)
{
    if len(beratBalita) == 0 {
        return
    }
    *min = beratBalita[0]
    *max = beratBalita[0]
    for _, berat := range beratBalita[1:] {
        if berat < *min {
            *min = berat
        }
        if berat > *max {
            *max = berat
        }
    }
}

func hitungRerata(beratBalita []float64) float64 {
    if len(beratBalita) == 0 {
        return 0
    }
    var total float64
    for _, berat := range beratBalita {
        total += berat
    }
    return total / float64(len(beratBalita))
}

func main() {
    var jumlah int
    fmt.Print("Masukan banyak data berat balita: ")
    fmt.Scan(&jumlah)

    beratBalita := make([]float64, jumlah)
```

```go
    for i := 0; i < jumlah; i++ {
        fmt.Printf("Masukan berat balita ke-%d: ", i+1)
        fmt.Scan(&beratBalita[i])
    }

    var beratMinimum, beratMaksimum float64
    hitungMinMax(beratBalita, &beratMinimum, &beratMaksimum)
    rerata := hitungRerata(beratBalita)

    fmt.Printf("Berat balita minimum: %.2f kg\n", beratMinimum)
    fmt.Printf("Berat balita maksimum: %.2f kg\n", beratMaksimum)
    fmt.Printf("Rerata berat balita: %.2f kg\n", rerata)
}
```

```
Masukkan jumlah ikan yang akan dijual (x): 4
Masukkan jumlah ikan per wadah (y): 2
Masukkan berat ikan ke-1: 12
Masukkan berat ikan ke-2: 5
Masukkan berat ikan ke-3: 13
Masukkan berat ikan ke-4: 7
Total berat ikan per wadah:
Wadah 1: 25.00 kg
Wadah 2: 12.00 kg
Rata-rata berat ikan per wadah: 18.50 kg
PS D:\ALPRO 2\103112400073_MODUL10> go run "d
Masukan banyak data berat balita: 4
Masukan berat balita ke-1: 23
Masukan berat balita ke-2: 50
Masukan berat balita ke-3: 12
Masukan berat balita ke-4: 15
Berat balita minimum: 12.00 kg
Berat balita maksimum: 50.00 kg
Rerata berat balita: 25.00 kg
```

Penjelasan :

Program ini digunakan untuk menganalisis berat badan balita dengan menghitung berat minimum, berat maksimum, dan rata-rata. Berikut penjelasan singkatnya:

1. **Fungsi `hitungMinMax`**:
    - Mengambil array berat balita dan menentukan nilai minimum serta maksimum.
    - Memperbarui variabel `min` dan `max` melalui pointer berdasarkan perbandingan berat.
2. **Fungsi `hitungRerata`**:
    - Menghitung rata-rata berat balita dengan menjumlahkan semua nilai dalam array dan membaginya dengan jumlah balita.
3. **Input Data**:
    - Pengguna diminta memasukkan jumlah balita.
    - Berat masing-masing balita dimasukkan ke dalam array `beratBalita`.
4. **Output**:
    - Menggunakan fungsi `hitungMinMax` untuk menemukan berat minimum dan maksimum.
    - Menggunakan fungsi `hitungRerata` untuk menghitung rata-rata.
    - Menampilkan hasil analisis dalam format dua desimal (`%.2f kg`).

VIII. KESIMPULAN

MODUL 10. PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA

Pencarian nilai maksimum atau minimum (nilai ekstrim) merupakan proses penting dalam pengolahan data yang sering ditemui dalam berbagai konteks kehidupan sehari-hari. Algoritma pencarian nilai ekstrim bekerja secara sederhana dan efisien dengan memproses data secara sekuensial. Langkah utamanya adalah menetapkan nilai awal sebagai nilai ekstrim, lalu membandingkannya dengan setiap data berikutnya. Jika ditemukan nilai yang lebih ekstrim, maka nilai tersebut diperbarui. Setelah seluruh data diperiksa, nilai ekstrim yang tersisa merupakan hasil yang dicari. Algoritma ini efektif karena hanya memerlukan satu kali iterasi terhadap seluruh data.

REFERENSI

MODUL 10. PENCARIAN NILAI EKSTRIM PADA HIMPUNAN DATA