Nama;Bintang Putra Angkasa

Nim: 2311102255

Kelas :IF 11 06

Soal 1

1.package main

import "fmt"

type set [2022]int

// Memeriksa apakah sebuah bilangan sudah ada di dalam array

func exist(T set, n int, val int) bool {

for i := 0; i < n; i++ {

if T[i] == val {

return true

}

}

return false

}

// Memasukkan data ke dalam set

func inputSet2311102255(T \*set, n \*int) {

\*n = 0

var val int

fmt.Println("Masukkan angka (akhiri dengan duplikat):")

for {

fmt.Scan(&val)

if exist(\*T, \*n, val) {

fmt.Println("Duplikat ditemukan, berhenti menerima input.")

return

}

T[\*n] = val

\*n++

}

}

// Mencari irisan dua himpunan

func findIntersection(T1, T2 set, n, m int, T3 \*set, h \*int) {

\*h = 0

for i := 0; i < n; i++ {

if exist(T2, m, T1[i]) && !exist(\*T3, \*h, T1[i]) {

T3[\*h] = T1[i]

\*h++

}

}

}

// Menampilkan isi set

func printSet(T set, n int) {

for i := 0; i < n; i++ {

fmt.Print(T[i], " ")

}

fmt.Println()

}

func main() {

var s1, s2, s3 set

var n1, n2, n3 int

// Input untuk himpunan pertama

fmt.Println("Input untuk himpunan pertama:")

inputSet2311102255(&s1, &n1)

// Input untuk himpunan kedua

fmt.Println("Input untuk himpunan kedua:")

inputSet2311102255(&s2, &n2)

// Mencari irisan

findIntersection(s1, s2, n1, n2, &s3, &n3)

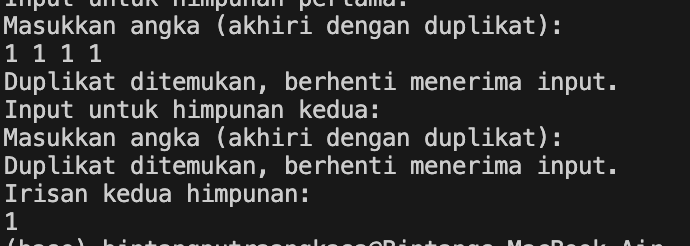
// Menampilkan hasil irisan

fmt.Println("Irisan kedua himpunan:")

printSet(s3, n3)

}

Output



Soal 2

2.package main

import (

"fmt"

)

const nMax = 51

type Mahasiswa struct {

NIM string

Nama string

Nilai int

}

type ArrayMahasiswa [nMax]Mahasiswa

var dataMahasiswa ArrayMahasiswa

var jumlahMahasiswa2311102255 int

// Fungsi untuk menambahkan data mahasiswa

func tambahMahasiswa(nim, nama string, nilai int) {

if jumlahMahasiswa2311102255 < nMax {

dataMahasiswa[jumlahMahasiswa2311102255] = Mahasiswa{NIM: nim, Nama: nama, Nilai: nilai}

jumlahMahasiswa2311102255++

} else {

fmt.Println("Data mahasiswa sudah penuh.")

}

}

// Fungsi untuk mencari nilai pertama seorang mahasiswa dengan NIM tertentu

func cariNilaiPertama(nim string) (int, bool) {

for i := 0; i < jumlahMahasiswa2311102255; i++ {

if dataMahasiswa[i].NIM == nim {

return dataMahasiswa[i].Nilai, true

}

}

return 0, false

}

// Fungsi untuk mencari nilai terbesar seorang mahasiswa dengan NIM tertentu

func cariNilaiTerbesar(nim string) (int, bool) {

terbesar := -1

found := false

for i := 0; i < jumlahMahasiswa2311102255; i++ {

if dataMahasiswa[i].NIM == nim {

if dataMahasiswa[i].Nilai > terbesar {

terbesar = dataMahasiswa[i].Nilai

found = true

}

}

}

return terbesar, found

}

func main() {

var N int

fmt.Print("Masukkan jumlah data mahasiswa: ")

fmt.Scan(&N)

for i := 0; i < N; i++ {

var nim, nama string

var nilai int

fmt.Print("Masukkan NIM: ")

fmt.Scan(&nim)

fmt.Print("Masukkan Nama: ")

fmt.Scan(&nama)

fmt.Print("Masukkan Nilai: ")

fmt.Scan(&nilai)

tambahMahasiswa(nim, nama, nilai)

}

var nimCari string

fmt.Print("Masukkan NIM yang ingin dicari: ")

fmt.Scan(&nimCari)

// Mencari nilai pertama

nilaiPertama, ditemukanPertama := cariNilaiPertama(nimCari)

if ditemukanPertama {

fmt.Printf("Nilai pertama mahasiswa dengan NIM %s adalah: %d\n", nimCari, nilaiPertama)

} else {

fmt.Printf("Mahasiswa dengan NIM %s tidak ditemukan.\n", nimCari)

}

// Mencari nilai terbesar

nilaiTerbesar, ditemukanTerbesar := cariNilaiTerbesar(nimCari)

if ditemukanTerbesar {

fmt.Printf("Nilai terbesar mahasiswa dengan NIM %s adalah: %d\n", nimCari, nilaiTerbesar)

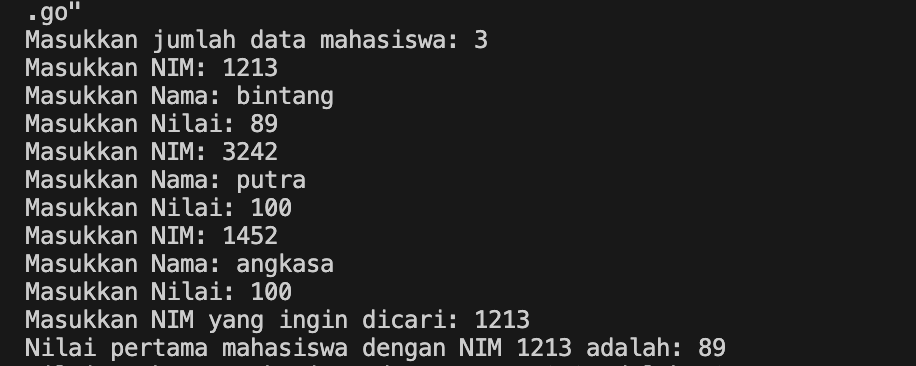
} else {

fmt.Printf("Mahasiswa dengan NIM %s tidak ditemukan.\n", nimCari)

}

}

Output



3package main

import (

"fmt"

"math"

)

const nProv = 34 // Jumlah provinsi yang ditangani program

// Struct untuk menyimpan data tentang provinsi

type Provinsi struct {

nama string

populasi int

tumbuh float64

}

func main() {

var prov [nProv]Provinsi

var namaDicari string

// Input data provinsi

for i := 0; i < nProv; i++ {

fmt.Printf("Masukkan data provinsi ke-%d (nama populasi tingkat\_pertumbuhan): ", i+1)

fmt.Scan(&prov[i].nama, &prov[i].populasi, &prov[i].tumbuh)

}

fmt.Print("Masukkan nama provinsi yang ingin dicari: ")

fmt.Scan(&namaDicari)

// Menampilkan nama provinsi dengan angka pertumbuhan tercepat

indeksTercepat := provinsiTercepat2311102255(prov)

fmt.Printf("Provinsi dengan pertumbuhan tercepat: %s\n", prov[indeksTercepat].nama)

// Menampilkan indeks provinsi yang dicari

indeksDicari := indeksProvinsi(prov, namaDicari)

if indeksDicari != -1 {

fmt.Printf("Indeks provinsi %s: %d\n", namaDicari, indeksDicari)

} else {

fmt.Printf("Provinsi %s tidak ditemukan.\n", namaDicari)

}

// Menampilkan prediksi jumlah penduduk untuk provinsi dengan pertumbuhan di atas 2%

fmt.Println("Prediksi populasi untuk provinsi dengan pertumbuhan di atas 2%:")

prediksi(prov)

}

// Fungsi untuk menemukan indeks provinsi dengan tingkat pertumbuhan tercepat

func provinsiTercepat2311102255(prov [nProv]Provinsi) int {

maxIndex := 0

for i := 1; i < nProv; i++ {

if prov[i].tumbuh > prov[maxIndex].tumbuh {

maxIndex = i

}

}

return maxIndex

}

// Fungsi untuk mencari indeks provinsi berdasarkan nama

func indeksProvinsi(prov [nProv]Provinsi, nama string) int {

for i := 0; i < nProv; i++ {

if prov[i].nama == nama {

return i

}

}

return -1

}

// Fungsi untuk menampilkan prediksi populasi untuk provinsi dengan tingkat pertumbuhan di atas 2%

func prediksi(prov [nProv]Provinsi) {

for i := 0; i < nProv; i++ {

if prov[i].tumbuh > 0.02 { // Cek jika tingkat pertumbuhan lebih dari 2%

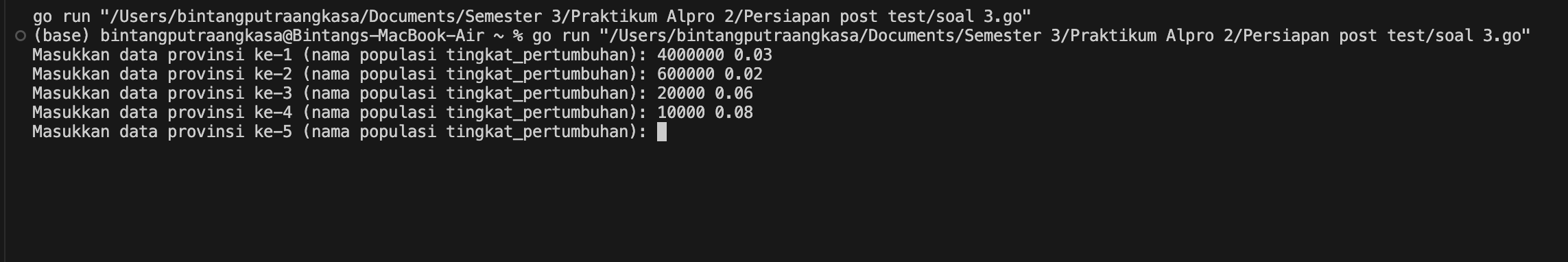
prediksiPopulasi := int(math.Round(float64(prov[i].populasi) \* (1 + prov[i].tumbuh)))

fmt.Printf("%s: %d\n", prov[i].nama, prediksiPopulasi)

}

}

}



Soal no 4

4.package main

import "fmt"

// Mendefinisikan konstanta untuk ukuran maksimum array

const maxArraySize = 1000000

// Fungsi untuk menghitung median dari data yang sudah diurutkan

func calculateMedian2311102255(data []int) int {

n := len(data)

// Jika jumlah elemen ganjil, kembalikan elemen tengah

if n%2 == 1 {

return data[n/2]

}

// Jika jumlah elemen genap, kembalikan rata-rata dari dua elemen tengah

return (data[n/2-1] + data[n/2]) / 2

}

// Fungsi untuk mengurutkan data menggunakan algoritma Insertion Sort

func insertionSort(data []int) []int {

for i := 1; i < len(data); i++ {

key := data[i]

j := i - 1

// Pindahkan elemen yang lebih besar dari key ke satu posisi di depan

for j >= 0 && data[j] > key {

data[j+1] = data[j]

j--

}

data[j+1] = key

}

return data

}

// Fungsi utama

func main() {

var data []int // Slice untuk menyimpan data input

var result []int // Slice untuk menyimpan hasil median

var num int // Variabel untuk menyimpan input pengguna

fmt.Println("Masukkan data (akhiri dengan -5313):")

for {

fmt.Scan(&num) // Membaca input dari pengguna

// Jika input adalah -5313, cetak semua hasil median dan keluar

if num == -5313 {

for \_, r := range result {

fmt.Println(r)

}

return

} else if num == 0 {

// Jika input adalah 0, urutkan data dan hitung median

data = insertionSort(data)

median := calculateMedian2311102255(data)

result = append(result, median) // Simpan median ke dalam hasil

} else {

// Jika input bukan 0 atau -5313, tambahkan ke data

if len(data) >= maxArraySize {

fmt.Println("Error: Jumlah data melebihi batas maksimum 1.000.000 elemen.")

continue // Jika melebihi batas, tampilkan pesan dan lanjutkan

}

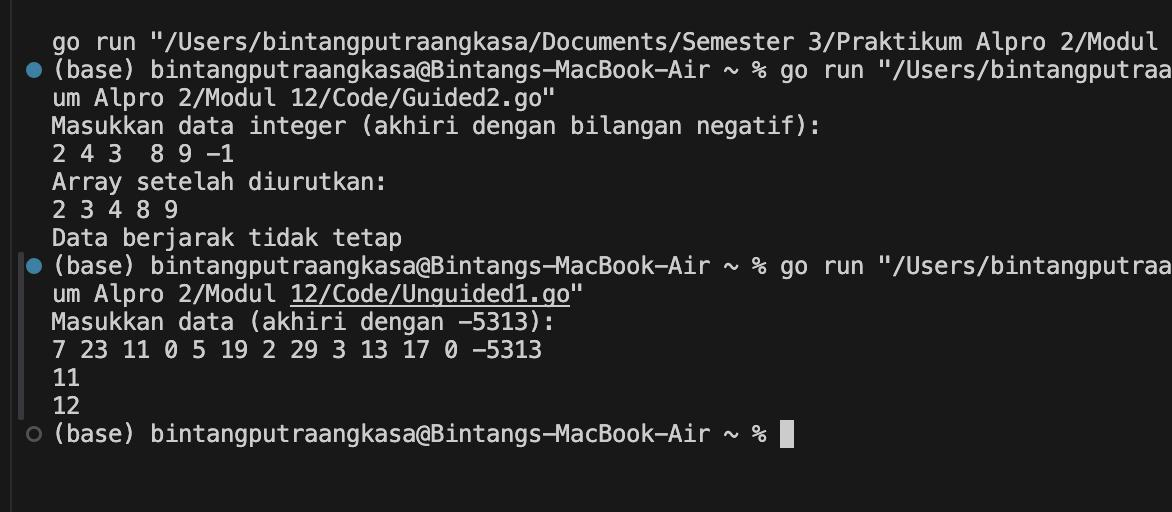
data = append(data, num) // Tambahkan input ke slice data

}

}

}

Output



Soal no 5

5.package main

import (

"fmt"

"sort"

)

const NMAX = 1000000

// Struct untuk menyimpan data partai

type Partai struct {

ID int

Suara int

}

// Tipe untuk array partai

type TabPartai []Partai

func main() {

var tabPartai\_2311102255 TabPartai // Mengganti nama variabel

var suara int

partaiMap := make(map[int]int)

fmt.Println("Masukkan perolehan suara untuk setiap partai (akhiri dengan -1):")

// Input perolehan suara

for {

fmt.Scan(&suara)

if suara == -1 {

break

}

partaiMap[suara]++

}

// Memindahkan data dari map ke slice

for id, totalSuara := range partaiMap {

tabPartai\_2311102255 = append(tabPartai\_2311102255, Partai{ID: id, Suara: totalSuara})

}

// Mengurutkan secara descending berdasarkan jumlah suara

sort.Slice(tabPartai\_2311102255, func(i, j int) bool {

if tabPartai\_2311102255[i].Suara == tabPartai\_2311102255[j].Suara {

return tabPartai\_2311102255[i].ID < tabPartai\_2311102255[j].ID // Jika suara sama, urutkan berdasarkan ID partai

}

return tabPartai\_2311102255[i].Suara > tabPartai\_2311102255[j].Suara

})

// Menampilkan hasil

fmt.Println("\nHasil perolehan suara:")

for \_, partai := range tabPartai\_2311102255 {

fmt.Printf("(%d:%d) ", partai.ID, partai.Suara)

}

fmt.Println()

}

