Tugas Kecil 1 IF2211
Strategi Algoritma
Semester II tahun 2025/2026
Penyelesaian Permainan Queens Linkedin

Disusun oleh:
Daniel Putra Rywandi S
13524143

Laboratorium Ilmu dan Rekayasa Komputasi
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

I. Deskripsi Permasalahan

Permainan Queens LinkedIn merupakan variasi dari permasalahan N-Queens dengan batasan tambahan. Diberikan papan berukuran n × n yang terbagi dalam beberapa daerah (region), tujuan permainan adalah menempatkan tepat satu queen pada setiap baris dengan memenuhi syarat:

1. Tidak ada dua queen pada kolom yang sama.
2. Tidak ada dua queen pada region yang sama.
3. Tidak ada dua queen yang saling bersebelahan (horizontal, vertikal, maupun diagonal).

II. Metode Penyelesaian

Metode yang digunakan adalah brute force. Brute force adalah strategi pencarian sederhana dan komprehensif yang secara sistematis mengeksplorasi setiap opsi hingga jawaban suatu masalah ditemukan, yang berarti algoritma yang digunakan mencoba seluruh kemungkinan konfigurasi penempatan queen.

III. Implementasi

1. Algoritma bacapapan

Algoritma bacapapan ini bertanggung jawab untuk membaca isi file .txt dan dan menghitung ukuran papan. Penghitungan di lakukan dengan cara menerima jumlah input hurud di file .txt lalu mengakar kannya. Logika ini digunakan karena papan yang digunakan selalu berbentuk persegi (n x n) sehingga tidak memerlukan validasi.

```
function bacapapan(scanner):
    // inisialisasi list kosong
    totalhuruf ← empty list

    // loop untuk membaca setiap huruf
    while scanner.hasnextline():
        line ← scanner.nextline()
        if length(line) > 0 then
            totalhuruf ← totalhuruf + list of characters in line
```

```
        end if
    end while

    // hitung jumlah total karakter
    total ← length(totalhuruf)
    n ← integer square root of total
    b ← array of n rows, each row is array of n elements
    index ← 0

    for i from 0 to n-1:
        for j from 0 to n-1:
            b[i][j] ← totalhuruf[index]
            index ← index + 1
        end for
    end for

    return b
end function
```

2. Algoritma solve

Algoritma solve bertujuan untuk menyelesaikan permasalahan pada permainan queens linkedin. Fungsi ini menggunakan 2 fungsi tambahan yaitu permutasi dan valid. Algoritma ini mencoba seluruh kemungkinan penempatan ratu yang memenuhi syarat dasar

```
function solve():
    // buat permutasi awal
    perm ← array of size n
    for i from 0 to n-1:
        perm[i] ← i

    // loop sampai semua permutasi diperiksa
    while true:
        count ← count + 1
        if valid(perm) then
            queenpos ← copy of perm
            return true
        end if
```

```
        if not permutasi(perm) then
            break
        end if
    end while

    return false
end function
```

3. Algoritma valid

Algoritma valid bertujuan untuk memverifikasi apakah sebuah konfigurasi penempatan ratu memenuhi seluruh aturan permainan Queens LinkedIn

```
function valid(perm):
    usedregion ← empty map

    for i from 0 to n-1:
        row ← i
        col ← perm[i]
        region ← papan[row][col]

        // cek apakah region sudah dipakai
        if region in usedregion then
            return false
        end if
        usedregion[region] ← true


        for j from 0 to i-1:
            if abs(row - j) <= 1 and abs(col - perm[j]) <= 1 then
                return false
            end if
        end for
    end for

    return true
```

```
end function
```

4. Algoritma permutasi

Algoritma permutasi bertugas untuk mengubah susunan array saat ini menjadi susunan permutasi berikutnya. Fungsi ini mengembalikan nilai true jika permutasi berikutnya berhasil dibentuk, dan false jika susunan saat ini sudah merupakan permutasi terakhir .

```
function permutasi(a):
    i ← length(a) - 2
    while i >= 0 and a[i] >= a[i+1]:
        i ← i - 1
    end while

    if i < 0 then
        return false
    end if

    j ← length(a) - 1
    while a[j] <= a[i]:
        j ← j - 1
    end while

    swap a[i] with a[j]
    reverse(a from i+1 to end)
    return true
end function
```

5. Algoritma reverse

algoritma reverse digunakan dalam proses pembentukan permutasi berikutnya.

```
function reverse(subarray):
```

```
    left ← 0
    right ← length(subarray) - 1
    while left < right:
        swap subarray[left] with subarray[right]
        left ← left + 1
        right ← right - 1
    end while
end function
```

6. Algoritma printpapan

algoritma printpapan digunakan untuk memvisualisasikan status papan permainan saat ini ke layar terminal

```
function printpapan():
    for i from 0 to n-1:
        for j from 0 to n-1:
            if queenpos[i] == j then
                print "#"
            else
                print papan[i][j]  // tampilkan karakter asli
            end if
        end for
        print newline
    end for
end function
```

6. Algoritma simpan

Algoritma simpan bertugas untuk menuliskan hasil akhir pencarian solusi,

```
function simpan(found, waktu):
```

```
    buka file "../test/output.txt" untuk menulis sebagai writer

    if found then
        for i from 0 to n-1:
            for j from 0 to n-1:
                if queenpos[i] == j then
                    tulis "#" ke writer
                else
                    tulis papan[i][j] ke writer
                end if
            end for
            tulis newline ke writer
        end for
    else
        tulis "tidak ada solusi" ke writer
    end if

    tulis "waktu pencarian: " + waktu ke writer
    tulis "banyak kasus yang ditinjau: " + count ke writer
    simpan dan tutup writer
    print "solusi berhasil disimpan ke ../test/output.txt"
end function
```

6. Algoritma mutlak

Algoritma Mutlak digunakan untuk proses validasi diagonal

```
function mutlak(x):
    if x < 0 then
        return -x
    else
        return x
    end if
end function
```

IV. Source Program

Program menggunakan bahasa golang

```
package main

import (
	"bufio"
	"fmt"
	"math"
	"os"
	"strings"
	"time"
)

var papan [][]byte
var n int
var queenpos []int
var count int
var anim bool

func main() {
	fmt.Print("Masukkan nama file : ")
	scannerInput := bufio.NewScanner(os.Stdin)
	if scannerInput.Scan() {
		filename := scannerInput.Text()
		file, err := os.Open(filename)
		if err != nil {
			fmt.Println("Gagal membuka file:", err)
			return
		}
		defer file.Close()

		scanner := bufio.NewScanner(file)
		papan = bacapapan(scanner)
	}

	fmt.Print("Tampilkan animasi visualisasi? (Ya/Tidak): ")
	animScanner := bufio.NewScanner(os.Stdin)
```

```
    if animScanner.Scan() {
        animJawaban := strings.TrimSpace(animScanner.Text())
        if strings.EqualFold(animJawaban, "Ya") {
            anim = true
        } else {
            anim = false
        }
    }

    start := time.Now()
    found := solve()
    end := time.Since(start)

    if anim {
        fmt.Print("\033[H\033[2J")
    }

    printpapan()
    if found {
        fmt.Printf("Waktu pencarian: %v\n", end)
        fmt.Printf("Banyak kasus yang ditinjau: %d kasus\n", count)
    } else {
        fmt.Println("Tidak ada solusi")
        fmt.Printf("Waktu pencarian: %v\n", end)
        fmt.Printf("Banyak kasus yang ditinjau: %d kasus\n", count)
    }

    fmt.Print("Apakah Anda ingin menyimpan solusi? (Ya/Tidak) ")
    reader := bufio.NewReader(os.Stdin)
    jawaban, _ := reader.ReadString('\n')
    jawaban = strings.TrimSpace(jawaban)

    if strings.EqualFold(jawaban, "Ya") {
        fmt.Print("Masukkan nama file output : ")
        outScanner := bufio.NewScanner(os.Stdin)
        if outScanner.Scan() {
            outName := outScanner.Text()
            simpan(found, end, outName)
        }
```

```
	}
}

func bacapapan(scanner *bufio.Scanner) [][]byte {
	var totalhuruf []byte

	for scanner.Scan() {
		line := scanner.Text()
		line = strings.TrimSpace(line)
		if len(line) > 0 {
			totalhuruf = append(totalhuruf, []byte(line)...)
		}
	}

	total := len(totalhuruf)
	root := int(math.Sqrt(float64(total)))

	if root < 4 {
		fmt.Println("Tidak valid: Ukuran papan minimal 4x4.")
		os.Exit(1)
	}

	n = root
	queenpos = make([]int, n)

	b := make([][]byte, n)
	index := 0
	for i := 0; i < n; i++ {
		b[i] = make([]byte, n)
		for j := 0; j < n; j++ {
			b[i][j] = totalhuruf[index]
			index++
		}
	}

	return b
}

func solve() bool {
```

```
	perm := make([]int, n)
	for i := 0; i < n; i++ {
		perm[i] = i
	}

	for {
		count++
		if anim {
			copy(queenpos, perm)
			printLive()
		}

		if valid(perm) {
			copy(queenpos, perm)
			return true
		}
		if !permutasi(perm) {
			break
		}
	}

	return false
}

func valid(perm []int) bool {
	used := make(map[byte]bool)

	for i := 0; i < n; i++ {
		row := i
		col := perm[i]

		region := papan[row][col]
		if used[region] {
			return false
		}
		used[region] = true

		for j := 0; j < i; j++ {
			if mutlak(row-j) <= 1 && mutlak(col-perm[j]) <= 1 {
```

```
                                return false
                        }
                }
        }

        return true
}

func permutasi(a []int) bool {
        i := len(a) - 2
        for i >= 0 && a[i] >= a[i+1] {
                i--
        }
        if i < 0 {
                return false
        }

        j := len(a) - 1
        for a[j] <= a[i] {
                j--
        }

        a[i], a[j] = a[j], a[i]
        reverse(a[i+1:])
        return true
}

func reverse(a []int) {
        for i, j := 0, len(a)-1; i < j; i, j = i+1, j-1 {
                a[i], a[j] = a[j], a[i]
        }
}

func printLive() {
        fmt.Print("\033[H\033[2J")
        printpapan()
        fmt.Printf("Mencoba konfigurasi ke: %d\n", count)
        time.Sleep(10 * time.Millisecond)
}
```

```
func printpapan() {
	for i := 0; i < n; i++ {
		for j := 0; j < n; j++ {
			if queenpos[i] == j {
				fmt.Print("#")
			} else {
				fmt.Printf("%c", papan[i][j])
			}
		}
		fmt.Println()
	}
}

func simpan(found bool, end time.Duration, filename string) {
	f, err := os.Create(filename)
	if err != nil {
		fmt.Println("Gagal menyimpan file:", err)
		return
	}
	defer f.Close()

	writer := bufio.NewWriter(f)

	if found {
		for i := 0; i < n; i++ {
			for j := 0; j < n; j++ {
				if queenpos[i] == j {
					fmt.Fprint(writer, "#")
				} else {
					fmt.Fprintf(writer, "%c", papan[i][j])
				}
			}
			fmt.Fprintln(writer)
		}
	} else {
		fmt.Fprintln(writer, "Tidak ada solusi")
	}
```

```
        fmt.Fprintf(writer, "Waktu pencarian: %v\n", end)
        fmt.Fprintf(writer, "Banyak kasus yang ditinjau: %d kasus\n",
count)
        writer.Flush()

        fmt.Printf("Solusi berhasil disimpan ke %s\n", filename)
}

func mutlak(x int) int {
        if x < 0 {
                return -x
        }
        return x
}
```

V. Tangkapan Layar

Input 1 :

```
AAABBCCCD
ABBBBCECD
ABBBDCECD
AAABDCCCD
BBBBDDDDD
FGGGDDHDD
FGIGDDHDD
FGIGDDHDD
FGGGDDHHH
```

```
PS C:\smthshit\Kuliah\tucil1\src> go run main.go
AAABBCC#D
ABBB#CECD
ABBBDC#CD
A#ABDCCCD
BBBBD#DDD
FGG#DDHDD
#GIGDDHDD
FG#GDDHDD
FGGGDDHH#
Waktu pencarian: 18.5192ms
Banyak kasus yang ditinjau: 306205 kasus
Apakah Anda ingin menyimpan solusi? (Ya/Tidak) ya
Solusi berhasil disimpan ke ../test/output.txt
PS C:\smthshit\Kuliah\tucil1\src>
```

Input 2 :

```
AAABB
AAABB
CCDEE
CCDEE
DDDEE
```

```
PS C:\smthshit\Kuliah\tucil1\src> go run main.go
#AABB
AAA#B
C#DEE
CCDE#
DD#EE
Waktu pencarian: 0s
Banyak kasus yang ditinjau: 14 kasus
Apakah Anda ingin menyimpan solusi? (Ya/Tidak)
```

Input 3 :

```
AAABB
AAABB
CDDDB
CDDEE
CCEEE
```

```
PS C:\smthshit\Kuliah\tucil1\src> go run main.go
A#ABB
AAA#B
#DDDB
CD#EE
CCEE#
Waktu pencarian: 0s
Banyak kasus yang ditinjau: 37 kasus
Apakah Anda ingin menyimpan solusi? (Ya/Tidak)
```

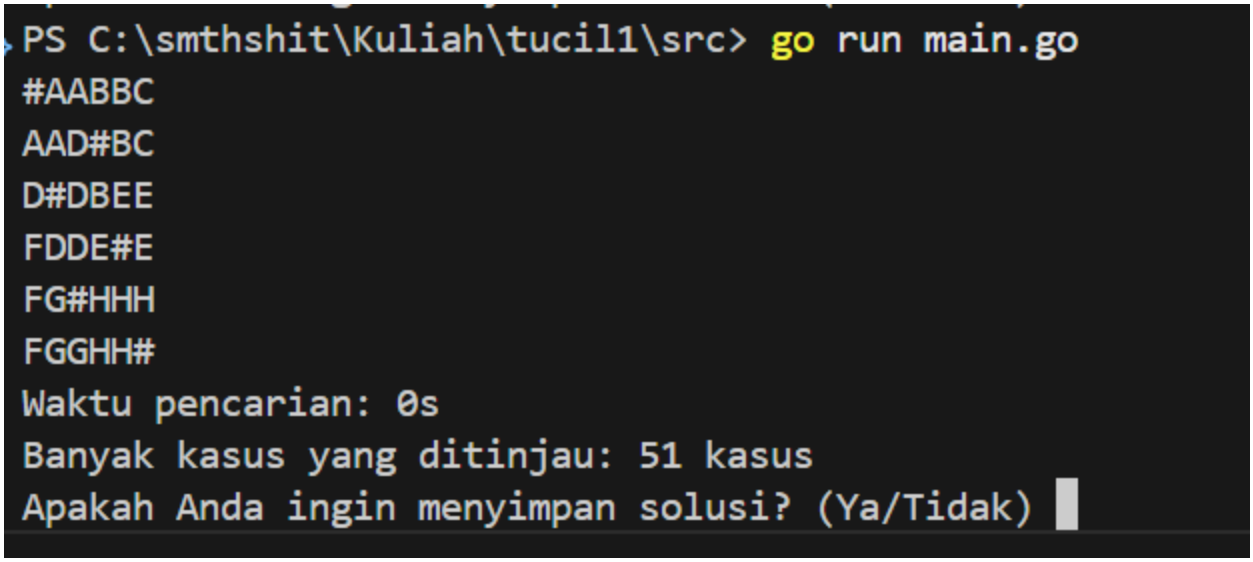
Input 4 :

```
AAABBC
AADBBC
DDDBEE
FDDEEE
FGGHHH
FGGHHH
```

```
PS C:\smthshit\Kuliah\tucil1\src> go run main.go
#AABBC
AAD#BC
D#DBEE
FDDE#E
FG#HHH
FGGHH#
Waktu pencarian: 0s
Banyak kasus yang ditinjau: 51 kasus
Apakah Anda ingin menyimpan solusi? (Ya/Tidak)
```

Input 5 :

```
AAAABBBB
ACCABBBB
ACCDDEEE
FFGDDEEE
FFGGGHHH
IIIGGHHH
IIIJJJKK
LLLLJJKK
```



VI. Pranala Ke Repository

https://github.com/ItsMeD4N/Tucil1_13524143

Tugas ini disusun sepenuhnya tanpa bantuan kecerdasan buatan (Generative AI), melainkan hasil pemikiran dan analisis mandiri.

Daniel

Daniel Putra Rywandi S