Nama : Muhammad Sidqi Nabhan

NIM : 1103200179

Analisa Tugas Week 14

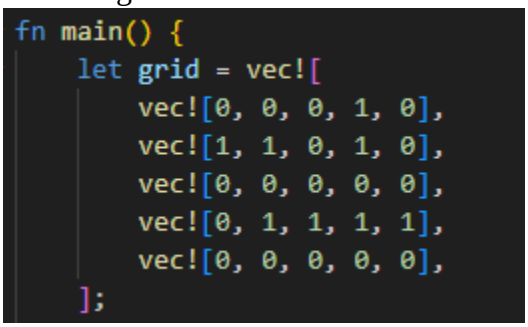
Source code

```
use std::collections::VecDeque;

fn find_path(grid: &Vec<Vec<i32>>, start: (usize, usize), end: (usize, usize)) ->
Option<Vec<(usize, usize)>> {
    let rows = grid.len();
    let cols = grid[0].len();
    let directions = [(0, 1), (1, 0), (0, -1), (-1, 0)];
    let mut visited = vec![vec![false; cols]; rows];
    let mut queue = VecDeque::new();
    let mut parent = vec![vec![None; cols]; rows];

    // Start BFS
    queue.push_back(start);
    visited[start.0][start.1] = true;

    while let Some((x, y)) = queue.pop_front() {
        if (x, y) == end {
            // Reconstruct path
            let mut path = vec![];
            let mut current = Some((x, y));
            while let Some(pos) = current {
                path.push(pos);
                current = parent[pos.0][pos.1];
            }
            path.reverse();
            return Some(path);
        }

        for &(dx, dy) in &directions {
            let nx = x as isize + dx;
            let ny = y as isize + dy;
            if nx >= 0 && nx < rows as isize && ny >= 0 && ny < cols as isize {
                let (nx, ny) = (nx as usize, ny as usize);
                if !visited[nx][ny] && grid[nx][ny] == 0 {
                    visited[nx][ny] = true;
                    queue.push_back((nx, ny));
                    parent[nx][ny] = Some((x, y));
                }
            }
```

```
        }
    }

    None // No path found
}

fn main() {
    let grid = vec![
        vec![0, 0, 0, 1, 0],
        vec![1, 1, 0, 1, 0],
        vec![0, 0, 0, 0, 0],
        vec![0, 1, 1, 1, 1],
        vec![0, 0, 0, 0, 0],
    ];

    let start = (0, 0);
    let end = (4, 4);

    match find_path(&grid, start, end) {
        Some(path) => {
            println!("Path found:");
            for (x, y) in path {
                println!("({}, {})", x, y);
            }
        }
        None => {
            println!("No path found!");
        }
    }
}
```

Analisa kode

1. Struktur grid



a. Grid : matriks 2D mempresentasikan peta
   0 : jalan bebas yang dapat dilalui robot
   1 : rintangan yang harus dihindari
b. Ukuran grid adalah 5x5 dengan titik awal (0,0) dan tujuan (4,4)

2. Algoritma Pencarian Jalur: Breadth-First Search (BFS)

```
while let Some((x, y)) = queue.pop_front() {
    if (x, y) == end {
        // Reconstruct path
        let mut path = vec![];
        let mut current = Some((x, y));
        while let Some(pos) = current {
            path.push(pos);
            current = parent[pos.0][pos.1];
        }
        path.reverse();
        return Some(path);
    }

    for &(dx, dy) in &directions {
        let nx = x as isize + dx;
        let ny = y as isize + dy;
        if nx >= 0 && nx < rows as isize && ny >= 0 && ny < cols as isize {
            let (nx, ny) = (nx as usize, ny as usize);
            if !visited[nx][ny] && grid[nx][ny] == 0 {
                visited[nx][ny] = true;
                queue.push_back((nx, ny));
                parent[nx][ny] = Some((x, y));
```

a. BFS adalah algoritma yang bekerja dengan menjelajah seluruh tetangga terdekat terlebih dahulu sebelum melangkah lebih jauh. Ini memastikan jalur terpendek ditemukan
b. Queue (antrian) digunakan untuk melacak titik yang akan diproses
c. Visited (dikunjungi) digunakan untuk memastikan titik tidak diproses lebih dari sekali

3. Rekonstruksis jalur

```
let mut current = Some((x, y));
while let Some(pos) = current {
    path.push(pos);
    current = parent[pos.0][pos.1];
}
```

a. Setelah mencapai tujuan (4,4) program menelusuri Kembali dari titik tujuan ke titik awal menggunakan array parent
b. Parent: menyimpan hubungan dari titik sebelumnya, sehingga jalur dapat direkonstruksi

4. output

```rust
let mut current = Some((x, y));
while let Some(pos) = current {
    path.push(pos);
    current = parent[pos.0][pos.1];
}
```

   a. program mencetak semua Langkah yang membentuk jalur dari awal (0,0) ke tujuan (4,4)

Analisa Running Kode

```
E:\Tugas\LAST SEMESTER\Robotika Dan Sistem Cerdas\Tugas 14\week14>rustc robot.rs

E:\Tugas\LAST SEMESTER\Robotika Dan Sistem Cerdas\Tugas 14\week14>robot.exe
Path found:
(0, 0)
(0, 1)
(0, 2)
(1, 2)
(2, 2)
(2, 1)
(2, 0)
(3, 0)
(4, 0)
(4, 1)
(4, 2)
(4, 3)
(4, 4)
```

Hasil running kode menunjukkan bahwa program berhasil menemukan jalur terpendek dari titik awal (0, 0) ke tujuan (4, 4) menggunakan algoritma Breadth-First Search (BFS), dengan output berupa koordinat langkah-langkah yang diambil robot untuk mencapai tujuan. Robot mampu menghindari rintangan yang ditandai dengan 1 pada grid dan hanya melewati jalur yang aman 0. Jalur yang dihasilkan optimal dan sesuai dengan struktur grid, membuktikan bahwa implementasi algoritma berjalan dengan benar. Program juga mencetak jalur dalam urutan langkah yang jelas, merepresentasikan perjalanan dari awal ke akhir tanpa kesalahan.