

**LAPORAN TUGAS KECIL 3**  
**MATA KULIAH IF2211 STRATEGI ALGORITMA**  
**PENYELESAIAN PUZZLE RUSH HOUR MENGGUNAKAN**  
**ALGORITMA PATHFINDING**



**Dosen Pengampu:**  
Dr. Nur Ulfa Maulidevi, S.T, M.Sc.

**Disusun Oleh:**  
Nadhif Radityo Nugroho      13523045

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**  
**INSTITUT TEKNOLOGI BANDUNG**  
**MEI 2025**

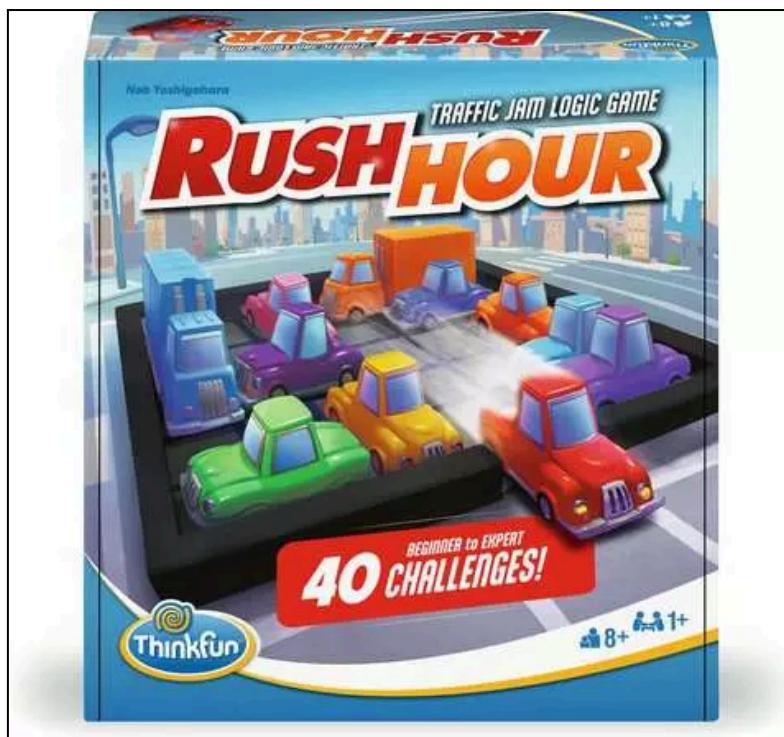
## DAFTAR ISI

<b>DAFTAR ISI.....</b>	<b>2</b>
<b>BAB I LATAR BELAKANG.....</b>	<b>4</b>
1.1. Permainan Rush Hour.....	4
1.2. Uniform Cost Search (UCS).....	5
1.3. Greedy Best First Search (GBFS).....	6
1.4. A-Star (A*).....	7
1.5. Iterative Deepening A-Star (IDA*).....	8
<b>BAB II DESKRIPSI DAN IMPLEMENTASI ALGORITMA.....</b>	<b>9</b>
2.1. Permainan Rush Hour.....	9
2.2. Algoritma Uniform Cost Search.....	10
2.3. Algoritma Greedy Best First Search.....	11
2.4. Algoritma A-Star.....	11
2.5. Algoritma Iterative Deepening A-Star.....	12
2.6. Branching Factor.....	13
2.7. Heuristik Car Distance.....	14
2.8. Heuristik Car Blocked.....	15
2.9. Heuristik Car Blocked Recursive.....	16
<b>BAB III SEKILAS KODE PROGRAM.....</b>	<b>17</b>
3.1. Struktur Program.....	17
3.2. Implementasi Program.....	17
3.3. Optimasi Program - Iterating Cars for State Expansion.....	22
3.4. Optimasi Program - Fast Queue Insertion.....	23
3.5. Optimasi Program - Stack Solver (IDA*) Fasttrack Approximation.....	25
3.6. Optimasi Program - Data Structure, Caching, and Fulfilled Assumptions.....	26
3.7. Bonus - Algoritma Alternatif.....	29
3.8. Bonus - Heuristic Alternatif.....	30
3.9. Bonus - GUI.....	31
3.10. Bonus - Deployment.....	31
3.11. Bonus - Video Animation Output.....	32
<b>BAB IV PERCOBAAN PROGRAM.....</b>	<b>33</b>
4.1. Jam-1.....	33
4.2. Jam-2.....	34
4.3. Jam-3.....	35
4.4. Jam-4.....	36
4.5. Jam-5.....	37
4.6. Jam-6.....	38
4.7. Jam-7.....	39
4.8. Jam-8.....	40
4.9. Jam-9.....	41

4.10. Jam-10.....	42
4.11. Jam-11.....	43
4.12. Jam-12.....	44
4.13. Jam-13.....	45
4.14. Jam-14.....	46
4.15. Jam-15.....	47
4.16. Jam-16.....	48
4.17. Jam-17.....	49
4.18. Jam-18.....	50
4.19. Jam-19.....	51
4.20. Jam-20.....	52
4.21. Jam-21.....	53
4.22. Jam-22.....	54
4.23. Jam-23.....	55
4.24. Jam-24.....	56
4.25. Jam-25.....	57
4.26. Jam-26.....	58
4.27. Jam-27.....	59
4.28. Jam-28.....	60
4.29. Jam-29.....	61
4.30. Jam-30.....	62
4.31. Jam-31.....	63
4.32. Jam-32.....	64
4.33. Jam-33.....	65
4.34. Jam-34.....	66
4.35. Jam-35.....	67
4.36. Jam-36.....	68
4.37. Jam-37.....	69
4.38. Jam-38.....	70
4.39. Jam-39.....	71
4.40. Jam-40.....	72
4.41. Invalid Input.....	73
<b>BAB V ANALISIS.....</b>	<b>79</b>
5.1. Analisis.....	79
5.2. Kesimpulan.....	80
5.3. Saran.....	80
<b>LAMPIRAN.....</b>	<b>81</b>
Tautan.....	81
Tabel Laporan.....	81

## BAB I LATAR BELAKANG

### 1.1. Permainan Rush Hour



Gambar 1. Rush Hour *Puzzle*

(Sumber: <https://www.thinkfun.com/en-US/products/educational-games/rush-hour-76582>)

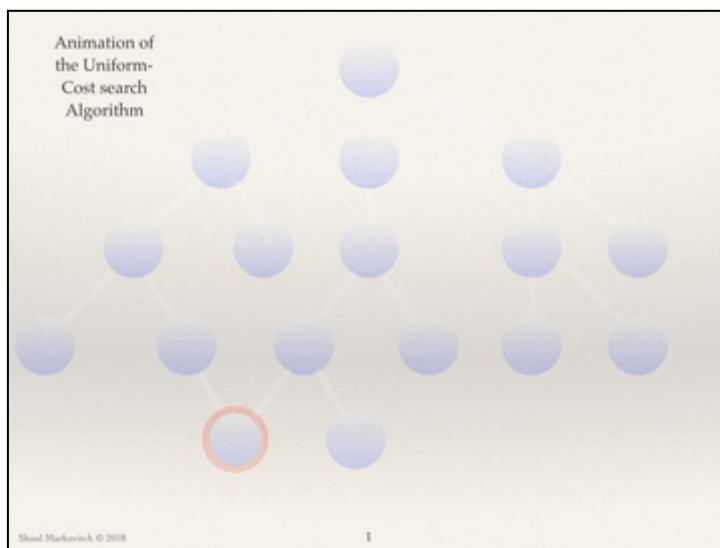
Rush Hour adalah sebuah permainan *puzzle* logika berbasis grid yang menantang pemain untuk menggeser kendaraan di dalam sebuah kotak (biasanya berukuran 6x6) agar mobil utama (biasanya berwarna merah) dapat keluar dari kemacetan melalui pintu keluar di sisi papan. Setiap kendaraan hanya bisa bergerak lurus ke depan atau ke belakang sesuai dengan orientasinya (horizontal atau vertikal), dan tidak dapat berputar. Tujuan utama dari permainan ini adalah memindahkan mobil merah ke pintu keluar dengan jumlah langkah seminimal mungkin.

Komponen penting dari permainan Rush Hour terdiri dari:

1. Papan — Papan merupakan tempat permainan dimainkan. Papan terdiri atas cell, yaitu sebuah singular point dari papan. Sebuah piece akan menempati cell-cell pada papan. Ketika permainan dimulai, semua piece telah diletakkan di dalam papan dengan konfigurasi tertentu berupa lokasi piece dan orientasi, antara horizontal atau vertikal. Hanya primary piece yang dapat digerakkan keluar papan melewati pintu keluar. Piece yang bukan primary piece tidak dapat digerakkan keluar papan. Papan memiliki satu pintu keluar yang pasti berada di dinding papan dan sejajar dengan orientasi primary piece.

2. Piece — Piece adalah sebuah kendaraan di dalam papan. Setiap piece memiliki posisi, ukuran, dan orientasi. Orientasi sebuah piece hanya dapat berupa horizontal atau vertikal—tidak mungkin diagonal. Piece dapat memiliki beragam ukuran, yaitu jumlah cell yang ditempati oleh piece. Secara standar, variasi ukuran sebuah piece adalah 2-piece (menempati 2 cell) atau 3-piece (menempati 3 cell). Suatu piece tidak dapat digerakkan melewati/menembus piece yang lain.
3. Primary Piece — Primary piece adalah kendaraan utama yang harus dikeluarkan dari papan (biasanya berwarna merah). Hanya boleh terdapat satu primary piece.
4. Pintu Keluar — Pintu keluar adalah tempat primary piece dapat digerakkan keluar untuk menyelesaikan permainan.
5. Gerakan — Gerakan yang dimaksudkan adalah pergeseran piece di dalam permainan. Piece hanya dapat bergerak/bergeser lurus sesuai orientasinya (atas-bawah jika vertikal dan kiri-kanan jika horizontal). Suatu piece tidak dapat digerakkan melewati/menembus piece yang lain.

## 1.2. Uniform Cost Search (UCS)



**Gambar 2.** Animasi Pencarian Algoritma UCS

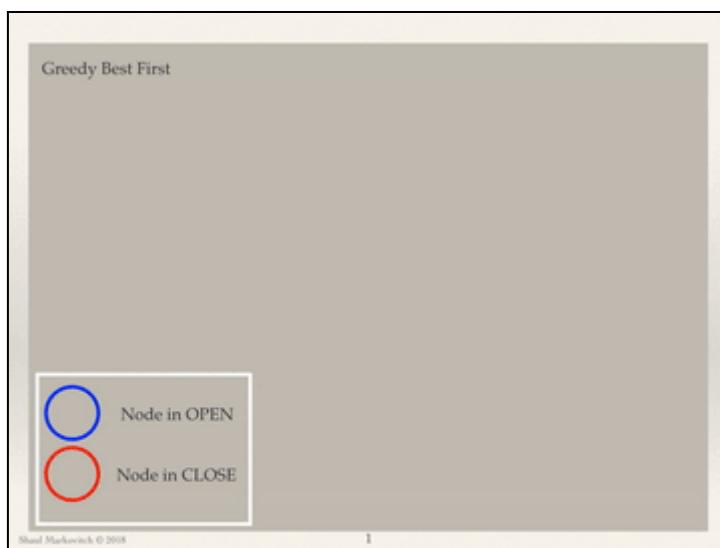
(Sumber:<https://www.youtube.com/watch?v=XyoucHYKYSE>)

*Uniform Cost Search* (UCS) adalah algoritma pencarian yang termasuk dalam kategori *uninformed search*, di mana pencarian dilakukan tanpa pengetahuan tambahan tentang tujuan akhir selain struktur graf. UCS bekerja dengan prinsip eksplorasi jalur berdasarkan biaya terkecil dari titik awal, menggunakan antrian prioritas untuk memilih simpul dengan total biaya terendah. Dalam konteks permainan Rush Hour, UCS akan mengevaluasi semua kemungkinan gerakan kendaraan berdasarkan total langkah yang telah dilakukan, tanpa mempertimbangkan seberapa dekat posisi tersebut dengan solusi akhir. Hal ini membuat UCS

bersifat optimal—selalu menemukan solusi dengan biaya terkecil jika biaya tiap langkah dianggap seragam—namun bisa sangat lambat karena mengevaluasi banyak kemungkinan.

Kelebihan utama UCS dalam Rush Hour adalah kemampuannya menemukan solusi optimal, yaitu jalur dengan jumlah gerakan paling sedikit yang diperlukan untuk mengeluarkan mobil utama dari papan. Namun, karena UCS tidak mempertimbangkan arah atau jarak menuju tujuan secara eksplisit, algoritma ini cenderung mengeksplorasi banyak jalur yang sebenarnya tidak efisien. Hal ini dapat menyebabkan waktu pencarian yang lama, terutama pada konfigurasi papan yang kompleks dengan banyak kendaraan penghalang. Meskipun demikian, UCS tetap menjadi tolok ukur penting dalam membandingkan performa algoritma pencarian lain seperti *Greedy Best-First Search*, A\*, dan IDA\*, yang menggunakan pendekatan heuristik untuk mengarahkan pencarian ke solusi lebih cepat.

### 1.3. *Greedy Best First Search (GBFS)*



**Gambar 3.** Animasi Pencarian Algoritma GBFS

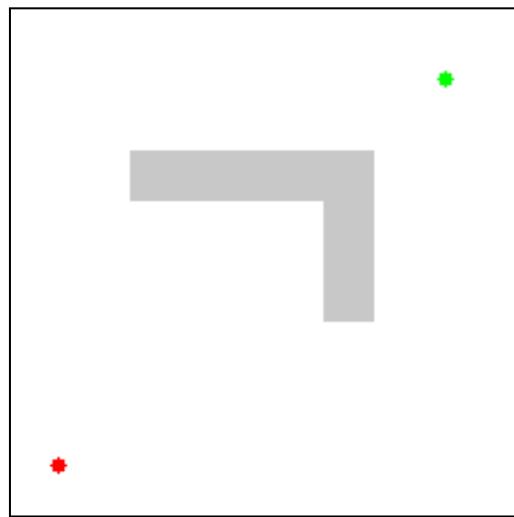
(Sumber: <https://www.youtube.com/watch?v=A8pmud1Uh0Q>)

*Greedy Best-First Search* (GBFS) adalah algoritma pencarian berbasis heuristik yang berfokus pada mempercepat proses pencarian solusi dengan memprioritaskan perluasan simpul yang tampak paling dekat dengan tujuan. Tidak seperti *Uniform Cost Search* yang mempertimbangkan biaya total dari titik awal, GBFS hanya mempertimbangkan nilai heuristik dari simpul saat ini untuk menentukan jalur mana yang akan dieksplorasi terlebih dahulu. Dalam permainan Rush Hour, GBFS akan memilih konfigurasi papan yang menurut fungsi heuristik tampak paling menjanjikan untuk mendekatkan mobil utama ke pintu keluar, tanpa memperhitungkan jumlah langkah yang sudah dilakukan.

Keunggulan utama GBFS adalah kecepatannya dalam menemukan solusi dibandingkan algoritma yang mempertimbangkan seluruh biaya jalur, karena hanya berfokus pada arah menuju tujuan. Namun, kelemahannya terletak pada sifatnya yang tidak optimal dan tidak

lengkap. Algoritma ini bisa saja terjebak pada jalur yang tampak menjanjikan namun sebenarnya membawa ke solusi tidak efisien. Dalam konteks Rush Hour, hal ini berarti GBFS dapat dengan cepat menemukan solusi, tetapi solusi tersebut belum tentu merupakan solusi dengan langkah paling sedikit. Oleh karena itu, meskipun cepat, GBFS lebih cocok digunakan ketika kecepatan lebih diutamakan daripada optimalitas.

#### 1.4. A-Star (A\*)



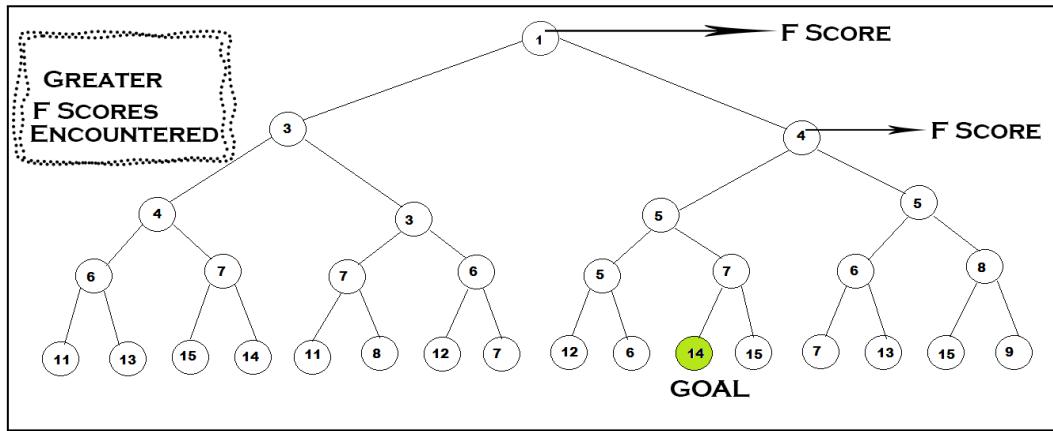
**Gambar 4.** Animasi Pencarian Algoritma A\*

(Sumber: [https://en.m.wikipedia.org/wiki/A\\*\\_search\\_algorithm](https://en.m.wikipedia.org/wiki/A*_search_algorithm))

A\* (*A-Star*) adalah algoritma pencarian yang menggabungkan kelebihan dari *Uniform Cost Search* dan *Greedy Best-First Search* dengan menggunakan fungsi evaluasi gabungan, yaitu  $f(n) = g(n) + h(n)$ . Di mana  $g(n)$  adalah biaya dari titik awal ke simpul saat ini, dan  $h(n)$  adalah estimasi biaya dari simpul tersebut ke tujuan akhir berdasarkan fungsi heuristik. Dalam permainan Rush Hour, A\* mempertimbangkan baik jumlah langkah yang sudah dilakukan maupun seberapa dekat posisi mobil utama dengan pintu keluar, sehingga pencarian solusi menjadi lebih terarah namun tetap optimal.

Kelebihan utama A\* adalah kemampuannya menemukan solusi yang tidak hanya cepat, tetapi juga optimal selama heuristik yang digunakan bersifat *admissible* (tidak melebih-lebihkan jarak ke tujuan). Dalam kasus Rush Hour, ini berarti A\* dapat menemukan jalur dengan jumlah langkah minimum sambil tetap menghindari eksplorasi konfigurasi papan yang tidak menjanjikan. Meskipun A\* umumnya lebih efisien daripada UCS karena menggunakan heuristik sebagai panduan, performanya tetap sangat bergantung pada kualitas heuristik yang digunakan. Jika heuristik dirancang dengan baik, A\* menjadi salah satu algoritma paling efektif untuk menyelesaikan *puzzle* seperti Rush Hour.

## 1.5. Iterative Deepening A-Star (IDA\*)



Gambar 5. Animasi Pencarian Algoritma IDA\*

(Sumber: <https://algorithmsinsight.wordpress.com/graph-theory-2/ida-star-algorithm-in-general/>)

*Iterative Deepening A-Star (IDA\*)* adalah algoritma pencarian yang menggabungkan prinsip dari A\* dan *Iterative Deepening Depth-First Search* (IDDFS). IDA\* menggunakan pendekatan pencarian berbasis kedalaman dengan batasan biaya (*cost threshold*) yang diturunkan dari fungsi evaluasi A\*, yaitu  $f(n) = g(n) + h(n)$ . Pada setiap iterasi, IDA\* melakukan pencarian secara mendalam hingga batas  $f(n)$  tertentu, dan jika tidak menemukan solusi, batas ini akan dinaikkan secara bertahap berdasarkan nilai terkecil yang melebihi ambang sebelumnya. Dalam konteks permainan Rush Hour, IDA\* mencoba menjelajahi solusi dengan lebih sedikit konsumsi memori dibandingkan A\*, karena tidak menyimpan seluruh node dalam antrian prioritas.

Kelebihan IDA\* terletak pada efisiensi memorinya yang sangat rendah, menjadikannya cocok untuk masalah dengan ruang status besar seperti Rush Hour yang kompleks. Meskipun bisa lebih lambat dibandingkan A\* karena mengulangi pencarian pada iterasi sebelumnya, IDA\* tetap menjamin optimalitas solusi jika heuristik yang digunakan bersifat admissible. Dalam aplikasi pada Rush Hour, IDA\* mampu menyeimbangkan antara kebutuhan memori dan kualitas solusi, menjadikannya alternatif menarik untuk menangani konfigurasi *puzzle* yang memerlukan banyak langkah atau memiliki banyak kendaraan penghalang.

## BAB II DESKRIPSI DAN IMPLEMENTASI ALGORITMA

### 2.1. Permainan Rush Hour

Dalam representasi komputer untuk masalah *puzzle* Rush Hour, class State berperan sebagai model yang menggambarkan setiap konfigurasi papan permainan pada suatu titik waktu tertentu. Setiap objek State menyimpan informasi lengkap mengenai posisi mobil-mobil yang ada, dinding, serta posisi pintu keluar pada papan dengan ukuran tertentu. Dengan mengorganisasi kondisi permainan secara hierarkis—dari keadaan awal (*root state*) hingga keadaan turunan yang merepresentasikan perubahan posisi mobil—class ini memungkinkan simulasi dan eksplorasi langkah-langkah solusi secara efisien. Selain itu, dengan menggunakan teknik pencatatan perbedaan posisi antar-state dan mekanisme *hashing* untuk mempercepat pengecekan kondisi yang telah pernah dijelajahi, representasi ini mendukung algoritma pencarian dalam menemukan urutan langkah yang membawa mobil utama ke pintu keluar, sehingga menyelesaikan *puzzle* Rush Hour.

```
class State:
    // Fields (private)
    parent          // Parent State (nullable)
    depth           // Depth in search tree
    width, height   // Board dimensions
    isDifference    // Flag: true if this state is a difference from parent
    allCars         // List of all cars
    carIds          // IDs of cars moved in this state
    carPositions    // Positions of cars (encoded)
    primaryCarPosition // Position of primary car (goal car)
    walls           // Positions of walls on the board
    exitPosition    // Position of exit on the board
    fields          // Array representing the board fields (car IDs or empty/wall)
    hashCode         // Cached hash code for this state
    cachedComputedCarPositions // Cache computed car positions
    cachedStepDescription // Cache of step description string
    cachedMoveDescription // Cache of move description string

    // Static factory methods
    // Create root state with full board setup
    static new_root(width, height, allCars, carPositions, walls, exitPosition)

    // Create new state from parent by updating positions of given cars
    static new_isDifference(parent, carIds, carPositions)

    // Create new state by applying steps (car moves) from a string
    static new_fromSteps(parent, stepsString)

    // Return full array of car positions, combining parent's positions and differences
    getComputedCarPositions()

    // Return a string describing the moves from root to current state
    getStepDescription()

    // Return a human-readable description of the last move
    getMoveDescription()

    // Check if car can be placed in given position/direction without collisions
    canFillField(direction, position, size)

    // Return list of valid move steps (positive and negative) for a car
    getCarMoveOptions(direction, position, size)
```

```

// Check if primary car occupies the exit position
isSolved()

// Return cached hash code of the state
hashCode()

// Return string representation of board for debugging
toString()

// Internal helper methods like #fillField(), #calculateHashCode(), etc.```

```

- Field internal menyimpan dimensi papan, posisi mobil, posisi dinding, posisi pintu keluar, serta representasi grid papan secara keseluruhan.
- Ada konsep *state root* sebagai kondisi awal lengkap, dan *state difference* yang hanya menyimpan perubahan posisi mobil relatif ke *parent state* untuk menghemat memori dan mempercepat perhitungan.
- Metode `canFillField` memeriksa apakah sebuah mobil bisa ditempatkan pada posisi tertentu tanpa bertabrakan.
- Metode `getCarMoveOptions` menghasilkan semua langkah valid yang dapat diambil sebuah mobil pada kondisi saat ini.
- Fungsi `isSolved` memeriksa apakah mobil utama sudah sampai di pintu keluar, menandai *puzzle* selesai.
- Kode juga menggunakan `hashCode` untuk mempercepat pencarian dan pengecekan status yang sudah pernah dijelajahi.
- Deskripsi langkah dan gerakan disediakan untuk membantu memvisualisasikan solusi langkah demi langkah.

## 2.2. Algoritma Uniform Cost Search

```

function UCS(start, goal):
    frontier ← priority queue ordered by path_cost
    frontier.push(start, path_cost=0)
    explored ← empty set

    while frontier is not empty:
        node ← frontier.pop()
        if node.state == goal:
            return solution(node)

        explored.add(node.state)
        for each child in expand(node):
            if child.state not in explored and child not in frontier:
                frontier.push(child, path_cost=child.g)
            else if child in frontier with higher path_cost:
                frontier.update(child, new lower path_cost)
    return failure

```

Algoritma *Uniform Cost Search* (UCS) bekerja dengan mengeksplorasi simpul berdasarkan biaya total terkecil dari titik awal ke simpul tersebut. Dalam pseudocode di atas,

UCS memulai dengan menempatkan simpul awal ke dalam *frontier* (antrian prioritas) dengan biaya 0. Pada setiap iterasi, simpul dengan biaya terkecil diambil dari antrian, dan jika simpul tersebut adalah tujuan, maka solusi ditemukan. Jika belum, simpul tersebut diekspansi dan semua anaknya diperiksa. Jika anak simpul belum pernah dieksplorasi dan tidak ada dalam *frontier*, maka ia ditambahkan. Namun, jika anak tersebut sudah ada dalam *frontier* tetapi memiliki jalur baru dengan biaya lebih rendah, maka nilai biayanya diperbarui. Proses ini berlanjut sampai solusi ditemukan atau semua kemungkinan telah dicoba. UCS menjamin solusi optimal selama biaya langkah antar simpul bersifat non-negatif.

### 2.3. Algoritma Greedy Best First Search

```

function GBFS(start, goal):
    frontier  $\leftarrow$  priority queue ordered by heuristic h(n)
    frontier.push(start)
    explored  $\leftarrow$  empty set

    while frontier is not empty:
        node  $\leftarrow$  frontier.pop()
        if node.state  $=$  goal:
            return solution(node)

        explored.add(node.state)
        for each child in expand(node):
            if child.state not in explored and child not in frontier:
                frontier.push(child)
    return failure

```

*Greedy Best-First Search* (GBFS) adalah algoritma pencarian yang berfokus pada mempercepat pencarian solusi dengan menggunakan fungsi heuristik  $h(n)$  untuk memperkirakan seberapa dekat suatu simpul dengan tujuan. Dalam pseudocode di atas, simpul awal dimasukkan ke dalam *frontier* yang diatur sebagai antrian prioritas berdasarkan nilai heuristiknya. Pada setiap langkah, simpul dengan nilai  $h(n)$  terendah diambil untuk dievaluasi. Jika simpul tersebut adalah tujuan, maka solusi dikembalikan. Jika tidak, simpul diekspansi dan semua anak yang belum dieksplorasi maupun belum ada di *frontier* ditambahkan ke dalam *frontier*. Karena hanya mempertimbangkan heuristik tanpa memperhitungkan jalur yang telah ditempuh, GBFS dapat bekerja lebih cepat, namun tidak menjamin solusi yang ditemukan adalah yang paling optimal.

### 2.4. Algoritma A-Star

```

function A*(start, goal):
    frontier  $\leftarrow$  priority queue ordered by  $f(n) = g(n) + h(n)$ 
    start.g  $\leftarrow$  0
    frontier.push(start, f(start))
    explored  $\leftarrow$  empty set

    while frontier is not empty:
        node  $\leftarrow$  frontier.pop()

```

```

    if node.state == goal:
        return solution(node)

    explored.add(node.state)
    for each child in expand(node):
        child.g <- node.g + cost(node, child)
        if child.state not in explored and child not in frontier:
            frontier.push(child, f(child) = child.g + h(child))
        else if child in frontier with higher f(n):
            frontier.update(child, f(child))
    return failure

```

A\* adalah algoritma pencarian yang menggabungkan dua elemen penting: biaya dari awal ke simpul saat ini ( $g(n)$ ) dan estimasi biaya dari simpul tersebut ke tujuan menggunakan heuristik ( $h(n)$ ). Dalam pseudocode di atas, simpul awal dimasukkan ke dalam *frontier* dengan nilai  $f(n)$  awal. Di setiap iterasi, simpul dengan nilai  $f(n)$  terkecil diambil dari antrian prioritas. Jika simpul tersebut merupakan tujuan, maka solusi dikembalikan. Jika belum, simpul tersebut diekspansi, dan anak-anaknya dihitung nilai  $g(n)$ -nya. Jika anak belum pernah dieksplorasi atau ada jalur baru dengan nilai  $f(n)$  yang lebih baik, simpul anak dimasukkan atau diperbarui dalam *frontier*. Karena A\* mempertimbangkan jalur yang telah ditempuh dan estimasi menuju tujuan, ia mampu menemukan solusi optimal dengan lebih efisien dibandingkan UCS, selama heuristik yang digunakan bersifat admissible.

## 2.5. Algoritma Iterative Deepening A-Star

```

function IDA*(start, goal):
    threshold <- f(start) = g(start) + h(start)

    while True:
        temp <- search(start, 0, threshold)
        if temp == FOUND:
            return solution
        if temp == infinity:
            return failure
        threshold <- temp

    function search(node, g, threshold):
        f <- g + h(node)
        if f > threshold:
            return f
        if node.state == goal:
            return FOUND
        min <- infinity
        for each child in expand(node):
            temp <- search(child, g + cost(node, child), threshold)
            if temp == FOUND:
                return FOUND
            if temp < min:
                min <- temp
        return min

```

IDA\* (*Iterative Deepening A-Star*) merupakan algoritma pencarian yang memadukan prinsip dari A\* dan teknik *iterative deepening*. Daripada menyimpan semua simpul dalam memori seperti A\*, IDA\* melakukan pencarian dengan batas threshold berdasarkan fungsi evaluasi  $f(n) = g(n) + h(n)$ , yang merupakan gabungan dari biaya aktual dan estimasi heuristik menuju tujuan. Pencarian dilakukan secara rekursif dan mendalam, dan jika dalam satu iterasi tidak ditemukan solusi, batas threshold akan diperbarui ke nilai  $f(n)$  terkecil yang melebihi ambang sebelumnya. Proses ini diulang hingga solusi ditemukan atau semua kemungkinan telah dieksplorasi. Kelebihan utama IDA\* adalah efisiensi memorinya, karena hanya menyimpan jejak jalur saat ini, menjadikannya cocok untuk masalah besar seperti Rush Hour meskipun berisiko melakukan eksplorasi berulang.

## 2.6. Branching Factor

```
function estimate_branching_factor(N, d):
    if d == 0:
        return 0

    low ← 1.0
    high ← N
    epsilon ← 0.0001

    while (high - low) > epsilon:
        mid ← (low + high) / 2
        total ← geometric_sum(mid, d)

        if total < N:
            low ← mid
        else:
            high ← mid

    return (low + high) / 2

function geometric_sum(b, d):
    return (b^(d + 1) - 1) / (b - 1)
```

*Branching factor* adalah jumlah rata-rata cabang atau kemungkinan tindakan yang dapat diambil dari setiap simpul dalam pohon pencarian. Dalam konteks algoritma pencarian seperti UCS, GBFS, A\*, dan IDA\*, *branching factor* berperan penting dalam menentukan kompleksitas waktu dan ruang pencarian. Semakin besar *branching factor*, semakin banyak simpul yang perlu dieksplorasi oleh algoritma, yang berarti waktu pencarian akan meningkat secara eksponensial terhadap kedalaman solusi. Dalam permainan Rush Hour, *branching factor* ditentukan oleh jumlah kendaraan yang dapat bergerak dan jumlah arah gerak yang tersedia (maju atau mundur), tergantung pada posisi kendaraan lain di papan.

Memahami *branching factor* membantu dalam memperkirakan seberapa besar ruang pencarian dan seberapa efisien suatu algoritma dapat menjelajahi solusi. Misalnya, jika suatu konfigurasi Rush Hour memiliki banyak kendaraan dengan ruang gerak bebas, maka

*branching factor*-nya akan tinggi dan pencarian akan menjadi lebih sulit serta lambat. Oleh karena itu, algoritma seperti A\* dan GBFS yang menggunakan heuristik lebih efektif dalam membatasi eksplorasi cabang yang tidak relevan. Dengan menganalisis *branching factor*, maka dapat memilih atau merancang algoritma yang paling sesuai untuk menyelesaikan suatu masalah dengan mempertimbangkan keseimbangan antara kecepatan, memori, dan optimalitas solusi.

## 2.7. Heuristik *Car Distance*

```
function heuristicCarDistance(oldBound, state):
    if state.isSolved():
        return oldBound

    width = state.getWidth()
    primaryCar = state.getAllCars()[0]
    primaryCarPosition = state.getPrimaryCarPosition()
    primaryCarPositionX = primaryCarPosition mod width
    primaryCarPositionY = primaryCarPosition div width
    exitPosition = state.getExitPosition()
    exitPositionX = exitPosition mod width
    exitPositionY = exitPosition div width

    if primaryCar.direction == HORIZONTAL:
        if primaryCarPositionX > exitPositionX:
            return oldBound + (primaryCarPositionX - exitPositionX)
        else:
            return oldBound + (exitPositionX - primaryCarPositionX -
primaryCar.size)

    if primaryCar.direction == VERTICAL:
        if primaryCarPositionY > exitPositionY:
            return oldBound + (primaryCarPositionY - exitPositionY)
        else:
            return oldBound + (exitPositionY - primaryCarPositionY -
primaryCar.size)

    return Infinity
```

Heuristik `heuristicCarDistance` menghitung estimasi jarak minimal yang harus ditempuh mobil utama untuk mencapai posisi pintu keluar. Dalam konteks pencarian jalur seperti A\*, nilai fungsi  $f(n)$  didefinisikan sebagai jumlah dari  $g(n)$  dan  $h(n)$ , di mana  $g(n)$  adalah biaya aktual yang sudah dikeluarkan untuk mencapai *state n* (dalam kode ini direpresentasikan oleh `oldBound`), sedangkan  $h(n)$  adalah estimasi biaya sisa dari *n* menuju solusi. Heuristik ini mengambil jarak horizontal atau vertikal antara posisi depan mobil utama dan pintu keluar, dikurangi ukuran mobil, sehingga memberikan perkiraan yang konsisten dan admissible untuk mempercepat pencarian solusi *puzzle Rush Hour*.

## 2.8. Heuristik *Car Blocked*

```
function heuristicCarBlocked(oldBound, state):
    if state.isSolved():
        return oldBound

    width = state.getWidth()
    height = state.getHeight()
    fields = state.getFields()
    primaryCar = state.getAllCars()[0]
    primaryCarPosition = state.getPrimaryCarPosition()
    primaryCarPositionX = primaryCarPosition mod width
    primaryCarPositionY = primaryCarPosition div width
    exitPosition = state.getExitPosition()
    exitPositionX = exitPosition mod width
    exitPositionY = exitPosition div width

    carBlocked = 0

    if primaryCar.direction == HORIZONTAL:
        if primaryCarPositionX > exitPositionX:
            for i from primaryCarPositionX - 1 down to 0:
                index = primaryCarPositionY * width + i
                if fields[index] != FIELD_EMPTY:
                    carBlocked += 1
        else:
            for i from primaryCarPositionX + primaryCar.size up to
width - 1:
                index = primaryCarPositionY * width + i
                if fields[index] != FIELD_EMPTY:
                    carBlocked += 1

    if primaryCar.direction == VERTICAL:
        if primaryCarPositionY > exitPositionY:
            for i from primaryCarPositionY - 1 down to 0:
                index = i * width + primaryCarPositionX
                if fields[index] != FIELD_EMPTY:
                    carBlocked += 1
        else:
            for i from primaryCarPositionY + primaryCar.size up to
height - 1:
                index = i * width + primaryCarPositionX
                if fields[index] != FIELD_EMPTY:
                    carBlocked += 1

    return oldBound + carBlocked
```

Heuristik `heuristicCarBlocked` memperkirakan kesulitan mencapai solusi dengan menghitung jumlah mobil atau objek lain yang menghalangi jalur mobil utama menuju pintu keluar. Dalam konteks pencarian seperti algoritma A\*, fungsi  $f(n)$  adalah total estimasi biaya untuk mencapai solusi dari node  $n$ , yang merupakan penjumlahan dari  $g(n)$  dan  $h(n)$ .  $g(n)$  merepresentasikan biaya aktual yang sudah dikeluarkan untuk mencapai *state* saat ini (di sini `oldBound`), sedangkan  $h(n)$  adalah estimasi tambahan biaya yang diperlukan agar mobil

utama tidak terhalang kendaraan lain. Dengan memasukkan jumlah mobil yang menghalangi jalur sebagai bagian dari heuristik, algoritma dapat lebih efektif memprioritaskan jalur yang lebih mungkin menuju solusi optimal.

## 2.9. Heuristik *Car Blocked Recursive*

Heuristik `heuristicCarBlockedRecursive` adalah pendekatan estimasi yang bertujuan untuk menentukan seberapa sulit mencapai solusi dalam permainan Rush Hour, dengan menghitung jumlah mobil yang menghalangi jalur mobil utama menuju pintu keluar. Dalam konteks algoritma pencarian seperti A\*, fungsi  $f(n) = g(n) + h(n)$ , di mana  $g(n)$  adalah biaya aktual yang telah dikeluarkan sejauh ini (`oldBound`), dan  $h(n)$  adalah estimasi biaya ke depan, yang dalam hal ini didasarkan pada jumlah mobil penghalang. Semakin banyak mobil yang menghalangi, semakin besar estimasi biaya tersebut.

Versi dasar heuristik ini hanya menghitung mobil yang secara langsung menghalangi mobil utama. Namun, sebuah pendekatan lanjutan mengembangkan ide ini dengan menghitung juga mobil-mobil yang menghalangi mobil-mobil penghalang, dan seterusnya, secara rekursif. Pendekatan ini memberikan estimasi yang lebih akurat terhadap kompleksitas *state* saat ini, namun dengan trade-off berupa waktu komputasi yang lebih tinggi. Tantangan dalam pendekatan ini termasuk penghitungan ruang yang diperlukan untuk memindahkan mobil, pengaruh tembok sebagai penghalang, dan mencegah perhitungan ganda terhadap mobil yang sama.

Implementasi lengkap pendekatan heuristik lanjutan ini dapat ditemukan dalam repositori proyek Rush Hour di GitHub: <https://github.com/saschazar21/rushhour>. Pendekatan tersebut ditulis dalam Java dan tersedia di direktori `Heuristics/AdvancedHeuristic.java`.

## BAB III SEKILAS KODE PROGRAM

### 3.1. Struktur Program

Berikut merupakan struktur file dari program:

```
└── .gitignore
└── .vscode/
    └── launch.json
└── index.mjs                      # Entry point backend server
└── index.route-esm.mjs             # Router caching file ESM frontend
└── logic.mjs                       # Implementasi utama algoritma puzzle
└── package-lock.json
└── package.json
└── postcss.config.mjs
└── protocols.mjs                  # Definisi protokol komunikasi worker
└── docs/
    ├── Spesifikasi Tugas Kecil 3 Stima 2024_2025.pdf
    └── Tucil3_13523045.pdf          # Dokumen laporan
└── public/
    ├── BoardView.mjs                # Komponen merender papan
    ├── CarBlock.mjs                 # Komponen visualisasi mobil
    └── assets/
        ├── car-1.png               # Gambar mobil untuk visualisasi
        ├── car-2.png
        ├── car-3.png
        ├── car-4.png
        ├── car-5.png
        ├── car-6.png
        ├── exit.png                 # Gambar pintu keluar
        └── wall.png                 # Gambar dinding
    ├── index.mjs                     # Entry point frontend
    ├── main-template.css            # Template stylesheet dasar
    ├── main.css                      # Stylesheet aplikasi frontend
    ├── protocols.mjs               # Definisi protokol komunikasi
    ├── shared.mjs                   # Utilitas frontend
    └── worker.mjs                  # RPC frontend via websocket
└── test/
    ├── perftest.mjs                # Pengujian performa algoritma
    ├── record-format.txt           # Format data recording puzzle
    ├── test-worker.mjs              # Entry point test worker
    ├── test.mjs                     # Pengujian umum aplikasi
    └── test_output.log              # Hasil keluaran pengujian otomatis
└── views/
    ├── basic-head.ejs              # Template EJS HTML
    └── index.ejs                   # Template utama halaman
└── worker.mjs                     # Entry point worker
```

### 3.2. Implementasi Program

logic.mjs cd35cff86fe00fe2c1220c1814e0a6ca41bed66

```
export class QueueSolver {
    ... snip ...
    tick() {
        if(this.#solution != null || this.#stateQueue.length == 0)
            return false;
        /** @type {[number, State]} */
        const [oldBound, state] = this.#stateQueue.shift();
        if(state.isSolved()) {
            this.#solution = state;
            return false;
        }
        /** @type {[number, State][]} */
        const toAddStateQueue = [];
        const width = state.getWidth();
        const allCars = state.getAllCars();
        const fields = state.getFields();
        // Optimisation: Do not compute car positions. Iterating the fields and
        storing the visited car is faster.
        // Optimisation: Check the visited cars by bit fields because `Set` is
        slower. This puts restriction to the max number of cars to 52.
        let visitedCars = 0;
        for(let i = 0; i < fields.length; i++) {
            const fieldValue = fields[i];
            if(fieldValue == FIELD_EMPTY || fieldValue == FIELD_WALL)
                continue;
            const carId = fieldValue - 1;
            if(visitedCars & (1 << carId)) continue;
            visitedCars |= (1 << carId);
            const car = allCars[carId];
            const x = i % width;
            const y = Math.floor(i / width);
            const moves = state.getCarMoveOptions(car.direction, i, car.size);
            if(moves.length == 0) continue;
            this.#searchCount += moves.length;
            for(const move of moves) {
                const newPosition = car.direction == HORIZONTAL ? y * width
                + (x + move) : (y + move) * width + x;
                const expandedState = State.new_isDifference(state,
                [car.id], [(((i + 1) << 16) | (newPosition & 0xFFFF)) >>> 0]);
                const hashCode = expandedState.hashCode();
                if(this.#closedHashes.has(hashCode)) continue;
                this.#closedHashes.add(hashCode);
                toAddStateQueue.push([this.#heuristicCalculator(oldBound,
                expandedState), expandedState]);
            }
        }
        // Optimisation: Sort the newly added states, and merge sort to the
        queue.
```

```

        // The premise is, the newly added states will always be smaller than
        // the existing states.
        // Because inserting to queue first and sort the queue, will
        // unnecessarily compare sorted entries.
        toAddStateQueue.sort((a, b) => a[0] - b[0]);
        QueueSolver.#mergeSortedArraysInPlace(this.#stateQueue,
toAddStateQueue, (a, b) => a[0] - b[0]);
        return this.#stateQueue.length != 0;
    }
    ...
}

export class StackSolver {
    ...
    tick() {
        if(this.#solution != null || this.#bound == Infinity)
            return false;
        const initialStateHashCode = this.#initialState.hashCode();
        const initialStateBound = this.#heuristicCalculator(0,
this.#initialState);
        this.#closedHashes.add(initialStateHashCode);
        const newBound = this.#search(this.#initialState, initialStateBound);
        this.#closedHashes.delete(initialStateHashCode);
        if(newBound instanceof State) {
            this.#solution = newBound;
            return false;
        }
        this.#bound = newBound;
        if(newBound == Infinity)
            return false;
        return true;
    }
    ...
}

...
/** @type {HeuristicCalculator} */
export const heuristicDepth = (_, state) => state.getDepth();
/** @type {HeuristicCalculator} */
export const heuristicConstant = () => 0;
/** @type {HeuristicCalculator} */
export const heuristicCarDistance = (_, state) => {
    if(state.isSolved()) return 0;
    const width = state.getWidth();
    const primaryCar = state.getAllCars()[0];
    const primaryCarPosition = state.getPrimaryCarPosition();
    const primaryCarPositionX = primaryCarPosition % width;
    const primaryCarPositionY = Math.floor(primaryCarPosition / width);
    const exitPosition = state.getExitPosition();
    const exitPositionX = exitPosition % width;
}

```

```

const exitPositionY = Math.floor(exitPosition / width);
if(primaryCar.direction == HORIZONTAL) {
    if(primaryCarPositionX > exitPositionX)
        return primaryCarPositionX - exitPositionX;
    else
        return exitPositionX - primaryCarPositionX - primaryCar.size;
}
if(primaryCar.direction == VERTICAL) {
    if(primaryCarPositionY > exitPositionY)
        return primaryCarPositionY - exitPositionY;
    else
        return exitPositionY - primaryCarPositionY - primaryCar.size;
}
return Infinity;
};

/** @type {HeuristicCalculator} */
export const heuristicCarBlocked = (_, state) => {
    if(state.isSolved()) return 0;
    const width = state.getWidth();
    const height = state.getHeight();
    const fields = state.getFields();
    const primaryCar = state.getAllCars()[0];
    const primaryCarPosition = state.getPrimaryCarPosition();
    const primaryCarPositionX = primaryCarPosition % width;
    const primaryCarPositionY = Math.floor(primaryCarPosition / width);
    const exitPosition = state.getExitPosition();
    const exitPositionX = exitPosition % width;
    const exitPositionY = Math.floor(exitPosition / width);
    let carBlocked = 0;
    if(primaryCar.direction == HORIZONTAL) {
        if(primaryCarPositionX > exitPositionX) {
            for(let i = primaryCarPositionX - 1; i >= 0; i--) {
                const index = primaryCarPositionY * width + i;
                if(fields[index] == FIELD_EMPTY) continue;
                carBlocked++;
            }
        } else {
            for(let i = primaryCarPositionX + primaryCar.size; i < width; i++) {
                const index = primaryCarPositionY * width + i;
                if(fields[index] == FIELD_EMPTY) continue;
                carBlocked++;
            }
        }
    }
    if(primaryCar.direction == VERTICAL) {
        if(primaryCarPositionY > exitPositionY) {
            for(let i = primaryCarPositionY - 1; i >= 0; i--) {
                const index = i * width + primaryCarPositionX;
                if(fields[index] == FIELD_EMPTY) continue;
                carBlocked++;
            }
        }
    }
}

```

```

        if(fields[index] == FIELD_EMPTY) continue;
        carBlocked++;
    }
} else {
    for(let i = primaryCarPositionY + primaryCar.size; i < height;
i++) {
        const index = i * width + primaryCarPositionX;
        if(fields[index] == FIELD_EMPTY) continue;
        carBlocked++;
    }
}
return carBlocked;
};

... snip ...

/**
 * @typedef {(oldBound: number, state: State) => number} HeuristicCalculator
 */

/**
 * @typedef {Object} Car
 * @property {number} id
 * @property {number} direction
 * @property {number} size
 */

export const FIELD_EMPTY = 0;
export const FIELD_WALL = 255;
export const FIELD_PRIMARY_CAR = 1;
export const HORIZONTAL = 0;
export const VERTICAL = 1;

// `State` is effectively constant. You should cache the computation over `State`
whenever possible.
export class State {
    ... snip ...

    /** @type {State?} */
    #parent;
    /** @type {number} */
    #depth;
    /** @type {number} */
    #width; // Optimisation: Positions are encoded as (y * width + x). Applies
for carPositions, walls, and exitPosition.
    /** @type {number} */
    #height;
    /** @type {boolean} */
    #isDifference; // Optimisation: Store the difference instead of copying all
}

```

```

objects.

    /** @type {Car[]} */
    #allCars; // Optimisation: Avoid traversing up to the root node.
    /** @type {number[]} */
    #carIds;
    /** @type {number[]} */
    #carPositions;
    /** @type {number} */
    #primaryCarPosition; // Optimisation: Heuristic needs this. Instead of
searching the fields, just cache when constructing state.
    /** @type {number[]} */
    #walls;
    /** @type {number} */
    #exitPosition;
    /** @type {Uint8Array} */
    #fields;
    /** @type {number} */
    #hashCode;
    ... snip ...
}

/**
 * @param {string} inputText
 * @returns {{
 *   width: number,
 *   height: number,
 *   cars: Car[],
 *   carPositions: number[],
 *   walls: number[],
 *   exitPosition: number
 *   }
 * }
export function parseBoardInput(inputText) {
    ... snip ...
}

```

### 3.3. Optimasi Program - Iterating Cars for State Expansion

Optimasi ini ditujukan untuk meningkatkan performa pencarian *state* baru dalam algoritma pemecahan *puzzle* dengan menghindari perhitungan posisi mobil secara eksplisit. Dari pada menghitung posisi tiap mobil dari objek mobil itu sendiri, algoritma ini langsung memindai seluruh grid (*fields*) dan mencatat mobil yang telah diproses menggunakan *bit field*, yang jauh lebih cepat dibandingkan struktur data *Set*. Setiap mobil hanya diproses satu kali, dengan identifikasi menggunakan *bit shifting* (maksimum 52 mobil karena keterbatasan representasi integer di IEEE754). Pendekatan ini mengurangi overhead alokasi objek dan operasi pencarian, sekaligus mempercepat deteksi mobil dan perluasan *state* baru. Strategi ini

secara signifikan mengurangi waktu komputasi selama eksplorasi *state* pada *puzzle* yang kompleks.

```
logic.mjs cd35cff86fe00fe2c1220c1814e0a6ca41bed66

/** @type {[number, State][]} */
const toAddStateQueue = [];
const width = state.getWidth();
const allCars = state.getAllCars();
const fields = state.getFields();
// Optimisation: Do not compute car positions. Iterating the
fields and storing the visited car is faster.
// Optimisation: Check the visited cars by bit fields because
`Set` is slower. This puts restriction to the max number of cars
to 52.
let visitedCars = 0;
for(let i = 0; i < fields.length; i++) {
    const fieldValue = fields[i];
    if(fieldValue === FIELD_EMPTY || fieldValue === FIELD_WALL)
        continue;
    const carId = fieldValue - 1;
    if(visitedCars & (1 << carId)) continue;
    visitedCars |= (1 << carId);
    const car = allCars[carId];
    const x = i % width;
    const y = Math.floor(i / width);
    const moves = state.getCarMoveOptions(car.direction, i,
    car.size);
    if(moves.length === 0) continue;
    this.#searchCount += moves.length;
    for(const move of moves) {
        const newPosition = car.direction === HORIZONTAL ? y
        * width + (x + move) : (y + move) * width + x;
        const expandedState = State.new_isDifference(state,
        [car.id], [(((i + 1) << 16) | (newPosition & 0xFFFF)) >>> 0]);
        const hashCode = expandedState.hashCode();
        if(this.#closedHashes.has(hashCode)) continue;
        this.#closedHashes.add(hashCode);

        toAddStateQueue.push([this.#heuristicCalculator(oldBound,
        expandedState), expandedState]);
    }
}
```

### 3.4. Optimasi Program - Fast Queue Insertion

Optimasi ini bertujuan mempercepat penyisipan *state* baru ke dalam antrian prioritas dengan memanfaatkan asumsi bahwa semua *state* baru (*toAddStateQueue*) jumlahnya mungkin lebih kecil dibandingkan *state* yang sudah ada. Daripada langsung memasukkan dan

menyortir seluruh antrian, *state* baru terlebih dahulu diurutkan, lalu digabungkan secara efisien ke antrean utama menggunakan *in-place merge sort*. Proses penggabungan ini dioptimalkan lebih lanjut dengan pencarian posisi *insert* menggunakan binary search, sehingga meminimalkan jumlah perbandingan yang diperlukan. Strategi ini mengurangi kompleksitas dari operasi *insert* dan menjaga antrian tetap terurut tanpa penalti performa dari penyortiran penuh setiap iterasi.

```
logic.mjs cd35cff86fe00fe2c1220c1814e0a6ca41bed66

// Optimisation: Sort the newly added states, and merge sort to
// the queue.
// The premise is, the newly added states will always be smaller
// than the existing states.
// Because inserting to queue first and sort the queue, will
// unnecessarily compare sorted entries.
toAddStateQueue.sort((a, b) => a[0] - b[0]);
QueueSolver.#mergeSortedArraysInPlace(this.#stateQueue,
toAddStateQueue, (a, b) => a[0] - b[0]);
return this.#stateQueue.length != 0;

/**
 * @template T
 * @param {T[]} aArray
 * @param {T[]} bArray
 * @param {(a: T, b: T) => number} compareFn
 */
static #mergeSortedArraysInPlace(aArray, bArray, compareFn) {
    let aIndex = 0;
    for(let bIndex = 0; bIndex < bArray.length; bIndex++) {
        const item = bArray[bIndex];
        // Optimisation: Since aArray and bArray both
sorted, we can optimize
        // the insertion further by searching the aIndex
using binary search.
        let high = aArray.length;
        while(aIndex < high) {
            const mid = (aIndex + high) >> 1;
            if(compareFn(aArray[mid], item) <= 0)
                aIndex = mid + 1;
            else
                high = mid;
        }
        aArray.splice(aIndex, 0, item);
        aIndex++;
    }
}
```

### 3.5. Optimasi Program - *Stack Solver (IDA\*) Fasttrack Approximation*

Optimasi pada `StackSolverApprox` bertujuan mempercepat pencarian solusi tanpa mengorbankan optimalitas hasil secara signifikan. Berbeda dari pendekatan `StackSolver` yang menghindari siklus dengan memeriksa jejak (*path*) dalam tumpukan secara lokal, versi ini menggunakan pendekatan global melalui `closedHashes`, yaitu *dictionary hash* yang menyimpan bound terkecil untuk setiap *state* yang telah diekspansi. Suatu *state* hanya akan diproses kembali jika nilai bound barunya lebih kecil dari yang tercatat sebelumnya, memungkinkan eksplorasi ulang hanya bila menjanjikan hasil lebih baik. Teknik ini berpotensi mengurangi jumlah ekspansi tidak perlu dan menyimpan informasi antar iterasi untuk mempercepat tick berikutnya. Meskipun belum diverifikasi secara keseluruhan, pendekatan ini menunjukkan performa menjanjikan dalam beberapa kasus uji, dengan kemungkinan untuk dioptimalkan lebih lanjut melalui analisis karakteristik *state* yang sering menghasilkan perluasan tidak optimal.

```
logic.mjs cd35cff86fe00fe2c1220c1814e0a6ca41bed66
```

```
// This solver is an optimization over StackSolver. It may be faster for some test cases,
// but it should outputs optimal solution like StackSolver (though unverified for all cases).
// The main difference is how both solvers skip the closedHashes. In the "proper" implementation,
// it behaves like a hash path, where it skips the node if it's in the current stack trace.
// In contrast, this approx solver tries to use "global" closedHashes, where it still compares
// if the state bound is less than the last bound within the same hash. If it is, it'll probably
// resulting in fewer steps. Though, optimization can be further analyzed, since it has new
// information every tick to speed up the next tick. For example, skipping the state that
// always result in worst expansion.
export class StackSolverApprox {
    ...snip...
    tick() {
        if(this.#finished)
            return false;
        const initialStateBound = this.#heuristicCalculator(0, this.#initialState);
        this.#closedHashes.clear();
        this.#closedHashes.set(this.#initialState.hashCode(), initialStateBound);
        const newBound = this.#search(this.#initialState, initialStateBound);
        if(newBound instanceof State) {
            this.#finished = true;
            this.#solution = newBound;
            return false;
        }
        if(newBound == Infinity) {
            this.#finished = true;
            return false;
        }
        this.#bound = newBound;
        return true;
    }
    /**
     * @param {State} state
     * @param {number} bound
     * @returns {number | State}
     */
    #search(state, bound) {
```

```

        if(bound > this.#bound)
            return bound;
        if(state.isSolved())
            return state;
        /** @type {State?} */
        let solvedState = null;
        let minBound = Infinity;
        const width = state.getWidth();
        const allCars = state.getAllCars();
        const fields = state.getFields();
        let visitedCars = 0;
        for(let i = 0; i < fields.length; i++) {
            const fieldValue = fields[i];
            if(fieldValue == FIELD_EMPTY || fieldValue == FIELD_WALL) continue;
            const carId = fieldValue - 1;
            if(visitedCars & (1 << carId)) continue;
            visitedCars |= (1 << carId);
            const car = allCars[carId];
            const x = i % width;
            const y = Math.floor(i / width);
            const moves = state.getCarMoveOptions(car.direction, i, car.size);
            if(moves.length == 0) continue;
            this.#searchCount += moves.length;
            for(const move of moves) {
                const newPosition = car.direction == HORIZONTAL ? y * width + (x + move)
: (y + move) * width + x;
                const expandedState = State.new_isDifference(state, [car.id], [((i + 1)
<< 16) | (newPosition & 0xFFFF) >>> 0]);
                const expandedStateBound = this.#heuristicCalculator(bound,
expandedState);
                const hashCode = expandedState.hashCode();
                const closedHashBound = this.#closedHashes.get(hashCode);
                if(closedHashBound != null && closedHashBound <= expandedStateBound)
continue;
                this.#closedHashes.set(hashCode, expandedStateBound);
                const result = this.#search(expandedState, expandedStateBound);
                if(result instanceof State) {
                    if(expandedStateBound < minBound) {
                        solvedState = result;
                        minBound = expandedStateBound;
                    }
                    continue;
                }
                if(result < minBound) {
                    solvedState = null;
                    minBound = result;
                }
            }
        }
        if(solvedState != null)
            return solvedState;
        return minBound;
    }
}

```

### 3.6. Optimasi Program - Data Structure, Caching, and Fulfilled Assumptions

Optimasi ini berfokus pada efisiensi struktur data, *caching*, dan asumsi validitas untuk mempercepat proses pembentukan dan evaluasi *state* dalam algoritma pemecahan *puzzle*.

Representasi posisi (termasuk mobil, dinding, dan pintu keluar) dikompresi dalam bentuk  $y * width + x$ , mengurangi kebutuhan pemrosesan *array* dua dimensi. Untuk mempercepat pembuatan *state* turunan, pendekatan *difference-based* diterapkan, hanya menyimpan perbedaan terhadap *parent* daripada menyalin keseluruhan data. Informasi penting seperti posisi mobil utama langsung disimpan saat *state* dibuat, menghindari pencarian di *field*. Beberapa data turunan seperti posisi seluruh mobil, deskripsi langkah, dan deskripsi pergerakan dikalkulasi satu kali dan disimpan menggunakan *lazy caching*. Proses pengisian *field* tidak melakukan validasi ulang dengan asumsi bahwa *parent* valid dan langkah pergerakan telah diverifikasi sebelumnya. Selain itu, *hashCode* *state* dihitung secara efisien dengan teknik XOR dan perkalian sederhana, meskipun tidak sepenuhnya akurat, karena mengorbankan akurasi demi kecepatan, selama tidak ada mobil berukuran satu. Keseluruhan optimasi ini secara signifikan mengurangi overhead per iterasi, mempercepat traversal dan penyimpanan *state* dalam pencarian solusi.

```
logic.mjs cd35cff86fe00fe2c1220c1814e0a6ca41bed66
```

```
/** @type {State?} */
#parent;
/** @type {number} */
#depth;
/** @type {number} */
#width; // Optimisation: Positions are encoded as (y * width +
x). Applies for carPositions, walls, and exitPosition.
/** @type {number} */
#height;
/** @type {boolean} */
#isDifference; // Optimisation: Store the difference instead of
copying all objects.
/** @type {Car[]} */
#allCars; // Optimisation: Avoid traversing up to the root node.
/** @type {number[]} */
#carIds;
/** @type {number[]} */
#carPositions;
/** @type {number} */
#primaryCarPosition; // Optimisation: Heuristic needs this.
Instead of searching the fields, just cache when constructing
state.
/** @type {number[]} */
#walls;
/** @type {number} */
#exitPosition;
/** @type {Uint8Array} */
#fields;
/** @type {number} */
```

```

hashCode;

for(let i = 0; i < this.#carIds.length; i++) {
    const car = this.#allCars[this.#carIds[i]];
    const carPosition = this.#carPositions[i];
    const newPosition = carPosition & 0xFFFF;
    // Optimisation: Assume valid. Premise: Parent is valid
    // and getCarMovePositions returns valid moves.
    // if(!this.canFillField(car.direction, newPosition,
    car.size)) {
        //     const x = newPosition % this.#width;
        //     const y = Math.floor(newPosition / this.#width);
        //     throw new Error(`Cannot place Car#${car.id} at
position ${x}, ${y}) with direction ${car.direction} and size
${car.size}.\\n${this.toString()}`);
        //
    this.#fillField(car.direction, newPosition, car.size,
    car.id + 1);
    if(car.id == 0)
        this.#primaryCarPosition = newPosition;
}

```

```

calculateHashCode() {
    let hash = 0x811c9dc5;
    // Optimisation: This hashCode is not 100% correct. But
    // that's what we sacrifice to half the computation.
    // Optimisation: Compute the hashCode over checkered board
    // pattern to reduce clashes.
    // This hash is safe as long as there's no one-sized car.
    // for(let y = 0; y < this.#height; y++) {
    //     for(let x = y % 2 == 0 ? 0 : 1; x < this.#width;
    x++) {
        //
        const i = y * this.#width + x;
        //
        hash ^= this.#fields[i];
        //
        hash = (hash * 0x010193) >>> 0;
        //
    }
    for(let i = 0; i < this.#fields.length; i++) {
        hash ^= this.#fields[i];
        hash = (hash * 0x1193) >>> 0;
    }
    this.#hashCode = hash;
}

```

```

/** @type {number[]} */
#_cachedComputedCarPositions = null; // Optimisation: Cache.
getComputedCarPositions() {
    if(this.#_cachedComputedCarPositions != null)
        return this.#_cachedComputedCarPositions;
    ... snip ...
}

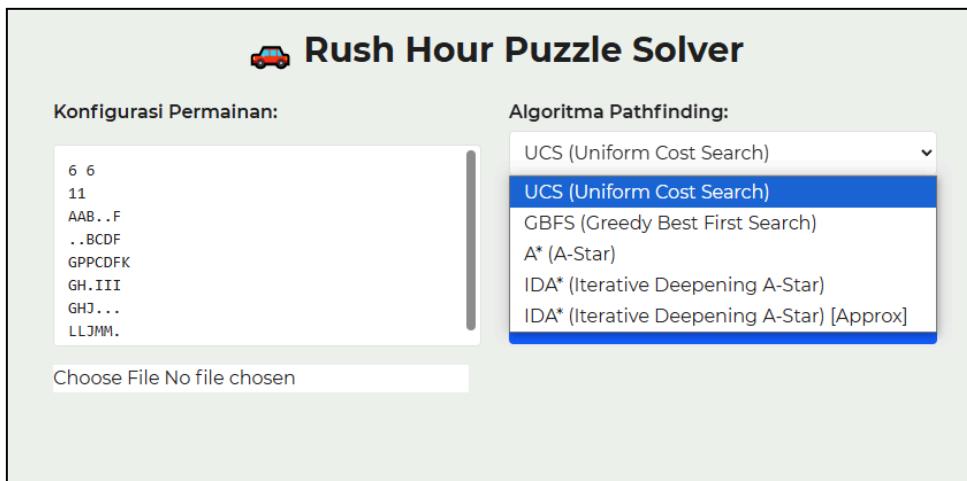
```

```

        return this.#__cachedComputedCarPositions = computedCars;
    }
    /** @type {string?} */
    #__cachedStepDescription = null; // Optimisation: Cache.
    getStepDescription() {
        if(this.#__cachedStepDescription != null)
            return this.#__cachedStepDescription;
        ... snip ...
        return this.#__cachedStepDescription = descriptions.join(
            ""
        );
    }
    /** @type {string?} */
    #__cachedMoveDescription = null; // Optimisation: Cache.
    getMoveDescription() {
        if(this.#__cachedMoveDescription != null)
            return this.#__cachedMoveDescription;
        ... snip ...
        return this.#__cachedMoveDescription = descriptions.length
            == 0 ? "No change in car positions" : descriptions.join("\n");
    }
}

```

### 3.7. Bonus - Algoritma Alternatif

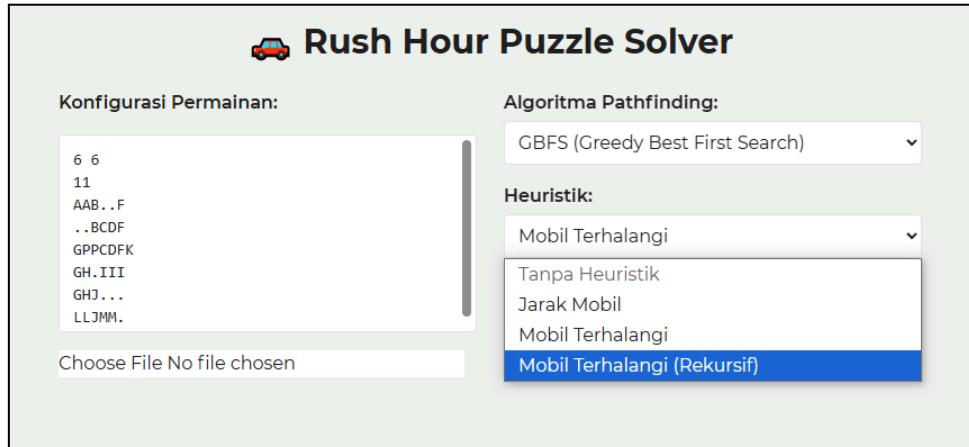


**Gambar X.** Macam-Macam Algoritma yang Ditawarkan

Sebagai bagian dari bonus, program ini mengimplementasikan dua varian algoritma pencarian: IDA\* (*Iterative Deepening A-Star*) dan IDA\* Approx. Keduanya dirancang untuk menemukan solusi optimal dengan memori yang jauh lebih rendah dibanding A\*, dengan melakukan pencarian berbasis batas heuristik yang meningkat secara bertahap. IDA\* menjaga akurasi dengan hanya melewati *node* yang berada dalam *stack trace* untuk menghindari siklus. Sementara itu, IDA\* Approx memperkenalkan pendekatan heuristik yang lebih agresif dengan menggunakan *closedHashes* global, memungkinkan pengabaian *state* yang dianggap tidak efisien berdasarkan bound sebelumnya. Pendekatan ini tidak menjamin

optimalitas sempurna, namun seringkali memberikan hasil lebih cepat pada kasus tertentu dengan jumlah langkah eksplorasi yang lebih sedikit. Kedua varian ini menunjukkan eksplorasi arah optimasi yang lebih eksperimental namun berpotensi menjanjikan dalam konteks efisiensi.

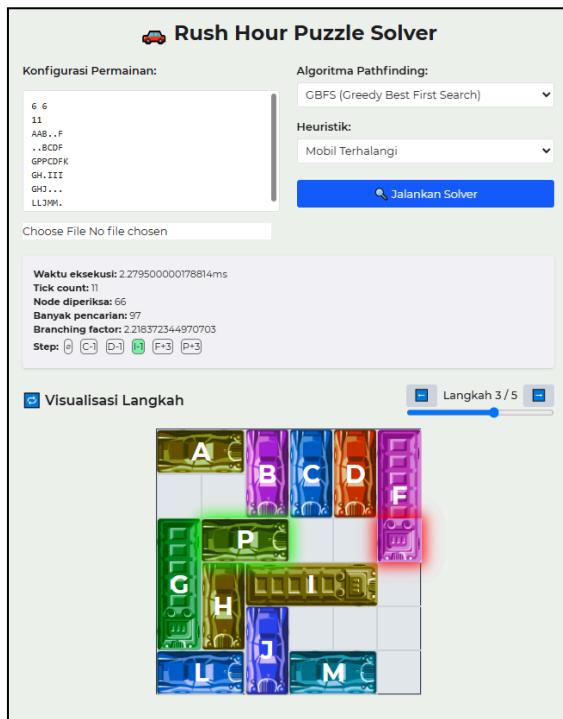
### 3.8. Bonus - Heuristic Alternatif



**Gambar X.** Macam-Macam Heuristik yang Ditawarkan

Sebagai fitur bonus, program ini menyediakan tiga alternatif fungsi heuristik untuk meningkatkan efisiensi pencarian solusi: *car distance*, *car blocked*, dan *car blocked recursive*. Heuristik *car distance* menghitung jarak dari mobil utama ke pintu keluar, memberikan estimasi dasar namun cepat. Heuristik *car blocked* menambahkan pertimbangan jumlah mobil yang menghalangi jalur mobil utama, sehingga lebih informatif dalam memperkirakan tingkat kesulitan. Sementara itu, *car blocked recursive* memperluas konsep ini dengan menghitung hambatan secara rekursif—yakni mempertimbangkan mobil yang menghalangi mobil penghalang—sehingga memberikan estimasi lebih akurat terhadap kompleksitas konfigurasi papan. Ketiga pendekatan ini menunjukkan fleksibilitas sistem dalam mengeksplorasi strategi evaluasi yang berbeda, dari yang ringan hingga kompleks, untuk menyesuaikan performa dan akurasi pencarian.

### 3.9. Bonus - GUI



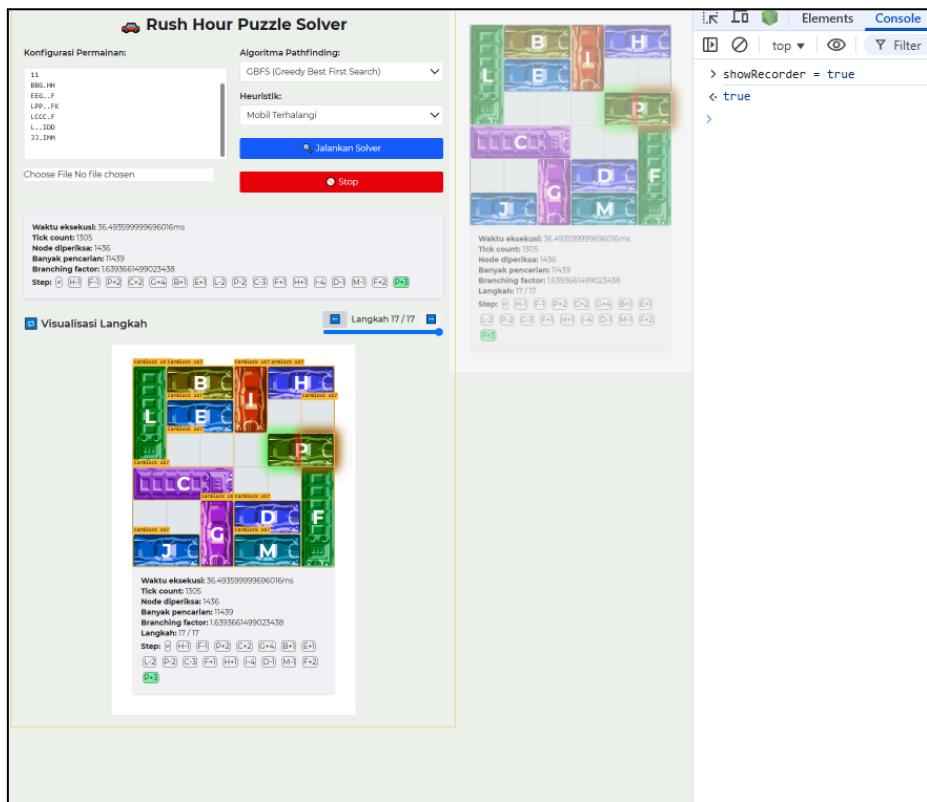
**Gambar X.** Tampilan GUI Program

Sebagai fitur bonus, program ini dilengkapi dengan antarmuka grafis (GUI) berbasis web yang memudahkan pengguna dalam berinteraksi dengan sistem. Pada halaman utama, pengguna dapat memasukkan *puzzle* dalam bentuk *string* melalui *form input*, atau memilih *file puzzle* langsung melalui tombol unggah. Selain itu, tersedia *selector* untuk memilih algoritma pencarian dan fungsi heuristik yang ingin digunakan, memberikan fleksibilitas dalam eksperimen dan evaluasi. Setelah tombol “Solve” ditekan dan solusi ditemukan, GUI akan menampilkan visualisasi papan permainan beserta animasi langkah-langkah penyelesaiannya, memungkinkan pengguna untuk mengikuti jalur solusi dengan jelas dan intuitif.

### 3.10. Bonus - Deployment

Sebagai bagian dari bonus, website ini telah di-*deploy* secara publik untuk dapat diakses oleh pengguna melalui domain pribadi di alamat <https://rushhour.bwks.link>. Proses *deployment* dilakukan pada sebuah VM server *on-premise* yang dikelola secara mandiri, menjalankan sistem operasi Windows Server 2022 Datacenter dengan spesifikasi 4 core CPU, 4 GB RAM, dan kapasitas penyimpanan sebesar 92 GB. Akses ke server diamankan dan di-rutekan menggunakan layanan Cloudflare Tunnel, dengan DNS yang juga dikelola melalui Cloudflare untuk memastikan kestabilan dan kemudahan pengelolaan domain. Domain yang digunakan merupakan milik pribadi, dan layanan ini direncanakan memiliki uptime selama kurang lebih tiga bulan sebagai periode demonstrasi dan pengujian publik.

### 3.11. Bonus - Video Animation Output



Gambar X. Merekam Animasi Solusi *Puzzle*

Sebagai fitur bonus tambahan, aplikasi ini menyediakan kemampuan untuk merekam animasi solusi dalam bentuk video. Pada tampilan antarmuka, terdapat tombol “Record” yang dapat muncul dengan mengaktifkan pengaturan `showRecorder = true` melalui *console inspector* browser. Setelah diaktifkan, tombol ini memungkinkan pengguna merekam animasi pergerakan solusi *puzzle* secara otomatis. Selama proses rekaman, animasi dijalankan dari awal hingga akhir tanpa perlu interaksi manual, dan hasil video disimpan ke file system. Fitur ini sangat berguna untuk menghasilkan dokumentasi visual solusi dari setiap *puzzle* secara massal dan terstruktur.

## BAB IV PERCOBAAN PROGRAM

*Semua gambar di bawah adalah animasi. Buka link docs untuk melihatnya secara langsung.*

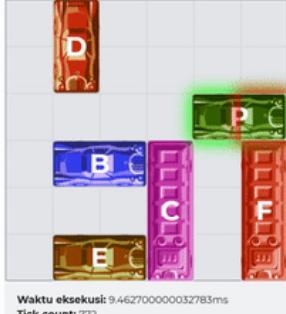
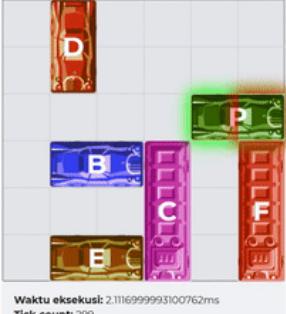
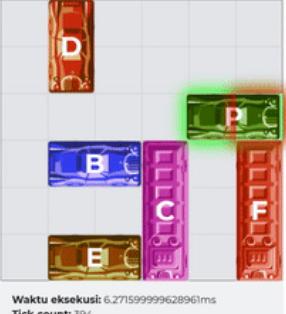
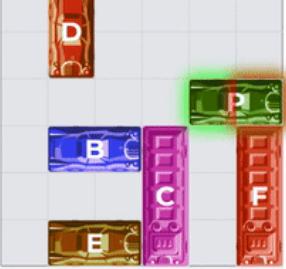
### 4.1. Jam-1

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-1</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>8</td><td>7</td></tr> <tr> <td>Step Optimal</td><td>:</td><td><math>\emptyset \text{ C+1 B-1 F-1 G-3}</math> <math>\text{H+3 E-2 D+2 P+3}</math></td><td>CC ... H B .. D.H BPPD.HK B .. D .. F ... GG F.EEE.</td></tr> </tbody> </table>				Nama	:	Jam-1	6 6	Banyak Step Optimal	:	8	7	Step Optimal	:	$\emptyset \text{ C+1 B-1 F-1 G-3}$ $\text{H+3 E-2 D+2 P+3}$	CC ... H B .. D.H BPPD.HK B .. D .. F ... GG F.EEE.
Nama	:	Jam-1	6 6												
Banyak Step Optimal	:	8	7												
Step Optimal	:	$\emptyset \text{ C+1 B-1 F-1 G-3}$ $\text{H+3 E-2 D+2 P+3}$	CC ... H B .. D.H BPPD.HK B .. D .. F ... GG F.EEE.												
UCS	GBFS Car Distance	GBFS Car Blocked													
<p>Waktu eksekusi: 22.68099999986589ms Tick count: 1057 Node diperiksa: 1079 Banyak pencarian: 11570 Branching factor: 3.0653724670410156 Langkah: 8 / 8 Step: <math>\emptyset \text{ C+1 B-1 F-1 G-3 H+3 E-2 D+2 P+3}</math></p>	<p>Waktu eksekusi: 37.242800000123568ms Tick count: 1222 Node diperiksa: 1031 Banyak pencarian: 9095 Branching factor: 2.6094627380371094 Langkah: 9 / 9 Step: <math>\emptyset \text{ C+1 B-1 F-1 G-3 E-2 D+2 P+2}</math></p>	<p>Waktu eksekusi: 4.180699999444187ms Tick count: 141 Node diperiksa: 580 Banyak pencarian: 1493 Branching factor: 2.096132278442383 Langkah: 0 / 9 Step: <math>\emptyset \text{ G+1 H+3 C+1 B+1 F+1 E+2 D+2}</math></p>													
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive													
<p>Waktu eksekusi: 27.000500000081956ms Tick count: 1063 Node diperiksa: 1079 Banyak pencarian: 18167 Branching factor: 2.6866703033447266 Langkah: 0 / 9 Step: <math>\emptyset \text{ C+1 B-1 F-1 G-3 E-2 D+2 P+2}</math></p>	<p>Waktu eksekusi: 47.23999999968708ms Tick count: 790 Node diperiksa: 1035 Banyak pencarian: 8784 Branching factor: 2.5648668725569398 Langkah: 0 / 8 Step: <math>\emptyset \text{ C+1 B-1 F-1 G-3 E-2 D+2}</math></p>	<p>Waktu eksekusi: 79.39229999948293ms Tick count: 615 Node diperiksa: 906 Banyak pencarian: 6916 Branching factor: 2.861856460571289 Langkah: 8 / 8 Step: <math>\emptyset \text{ G+1 H+3 C+1 B+1 F+1 E+2 D+2}</math></p>													

## 4.2. Jam-2

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-2</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>8</td><td>10</td></tr> <tr> <td>Step Optimal</td><td>:</td><td><math>\emptyset P+1 B+1 F-3 G-1</math> <math>H-2 I-1 L+2 P+3</math></td><td>B .. FFF B .. G..L PP.GHLK CCC.HL .. E.II DDEJJ.</td></tr> </tbody> </table>				Nama	:	Jam-2	6 6	Banyak Step Optimal	:	8	10	Step Optimal	:	$\emptyset P+1 B+1 F-3 G-1$ $H-2 I-1 L+2 P+3$	B .. FFF B .. G..L PP.GHLK CCC.HL .. E.II DDEJJ.
Nama	:	Jam-2	6 6												
Banyak Step Optimal	:	8	10												
Step Optimal	:	$\emptyset P+1 B+1 F-3 G-1$ $H-2 I-1 L+2 P+3$	B .. FFF B .. G..L PP.GHLK CCC.HL .. E.II DDEJJ.												
<table border="1"> <thead> <tr> <th>UCS</th> <th>GBFS Car Distance</th> <th>GBFS Car Blocked</th> </tr> </thead> <tbody> <tr> <td>  <p>Waktu eksekusi: 86.57409999985248ms Tick count: 2949 Node diperiksa: 4380 Banyak pencarian: 32515 Branching factor: 3.514249801635742 Langkah: 8 / 8 Step: <math>\emptyset P+1 B+1 F-3 G-1 H-2 I-1 L+2 P+3</math></p> </td><td>  <p>Waktu eksekusi: 16.556199999526143ms Tick count: 856 Node diperiksa: 1367 Banyak pencarian: 7916 Branching factor: 2.31927490234375 Langkah: 10 / 10 Step: <math>\emptyset P+1 B+1 F-3 G-1 H-2 I-1 L+2 P+3</math></p> </td><td>  <p>Waktu eksekusi: 2.019099999219179ms Tick count: 89 Node diperiksa: 354 Banyak pencarian: 638 Branching factor: 1.8069981079101562 Langkah: 10 / 10 Step: <math>\emptyset G+2 F-2 H-2 P+1 B+1 F-1 G-3</math></p> </td></tr> </tbody> </table>		UCS	GBFS Car Distance	GBFS Car Blocked	 <p>Waktu eksekusi: 86.57409999985248ms Tick count: 2949 Node diperiksa: 4380 Banyak pencarian: 32515 Branching factor: 3.514249801635742 Langkah: 8 / 8 Step: <math>\emptyset P+1 B+1 F-3 G-1 H-2 I-1 L+2 P+3</math></p>	 <p>Waktu eksekusi: 16.556199999526143ms Tick count: 856 Node diperiksa: 1367 Banyak pencarian: 7916 Branching factor: 2.31927490234375 Langkah: 10 / 10 Step: <math>\emptyset P+1 B+1 F-3 G-1 H-2 I-1 L+2 P+3</math></p>	 <p>Waktu eksekusi: 2.019099999219179ms Tick count: 89 Node diperiksa: 354 Banyak pencarian: 638 Branching factor: 1.8069981079101562 Langkah: 10 / 10 Step: <math>\emptyset G+2 F-2 H-2 P+1 B+1 F-1 G-3</math></p>								
UCS	GBFS Car Distance	GBFS Car Blocked													
 <p>Waktu eksekusi: 86.57409999985248ms Tick count: 2949 Node diperiksa: 4380 Banyak pencarian: 32515 Branching factor: 3.514249801635742 Langkah: 8 / 8 Step: <math>\emptyset P+1 B+1 F-3 G-1 H-2 I-1 L+2 P+3</math></p>	 <p>Waktu eksekusi: 16.556199999526143ms Tick count: 856 Node diperiksa: 1367 Banyak pencarian: 7916 Branching factor: 2.31927490234375 Langkah: 10 / 10 Step: <math>\emptyset P+1 B+1 F-3 G-1 H-2 I-1 L+2 P+3</math></p>	 <p>Waktu eksekusi: 2.019099999219179ms Tick count: 89 Node diperiksa: 354 Banyak pencarian: 638 Branching factor: 1.8069981079101562 Langkah: 10 / 10 Step: <math>\emptyset G+2 F-2 H-2 P+1 B+1 F-1 G-3</math></p>													
<table border="1"> <thead> <tr> <th>A* Car Distance</th> <th>A* Car Blocked</th> <th>A* Car Blocked Recursive</th> </tr> </thead> <tbody> <tr> <td>  <p>Waktu eksekusi: 41.451200000001043ms Tick count: 1477 Node diperiksa: 2321 Banyak pencarian: 15260 Branching factor: 2.775104522705078 Langkah: 9 / 9 Step: <math>\emptyset P+1 B+1 F-3 G-1 H-2 P+2 H-1 L+2 P+3</math></p> </td><td>  <p>Waktu eksekusi: 56.483500000008941ms Tick count: 1771 Node diperiksa: 1898 Banyak pencarian: 11877 Branching factor: 3.0760669708251953 Langkah: 8 / 8 Step: <math>\emptyset L+2 P+1 B+1 F-3 G-1 H-2 P+3</math></p> </td><td>  <p>Waktu eksekusi: 34.0848000000302494ms Tick count: 816 Node diperiksa: 1374 Banyak pencarian: 8323 Branching factor: 2.933656692504883 Langkah: 8 / 8 Step: <math>\emptyset H-1 L+2 P+1 B+1 F-3 G-1 H-2 P+3</math></p> </td></tr> </tbody> </table>		A* Car Distance	A* Car Blocked	A* Car Blocked Recursive	 <p>Waktu eksekusi: 41.451200000001043ms Tick count: 1477 Node diperiksa: 2321 Banyak pencarian: 15260 Branching factor: 2.775104522705078 Langkah: 9 / 9 Step: <math>\emptyset P+1 B+1 F-3 G-1 H-2 P+2 H-1 L+2 P+3</math></p>	 <p>Waktu eksekusi: 56.483500000008941ms Tick count: 1771 Node diperiksa: 1898 Banyak pencarian: 11877 Branching factor: 3.0760669708251953 Langkah: 8 / 8 Step: <math>\emptyset L+2 P+1 B+1 F-3 G-1 H-2 P+3</math></p>	 <p>Waktu eksekusi: 34.0848000000302494ms Tick count: 816 Node diperiksa: 1374 Banyak pencarian: 8323 Branching factor: 2.933656692504883 Langkah: 8 / 8 Step: <math>\emptyset H-1 L+2 P+1 B+1 F-3 G-1 H-2 P+3</math></p>								
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive													
 <p>Waktu eksekusi: 41.451200000001043ms Tick count: 1477 Node diperiksa: 2321 Banyak pencarian: 15260 Branching factor: 2.775104522705078 Langkah: 9 / 9 Step: <math>\emptyset P+1 B+1 F-3 G-1 H-2 P+2 H-1 L+2 P+3</math></p>	 <p>Waktu eksekusi: 56.483500000008941ms Tick count: 1771 Node diperiksa: 1898 Banyak pencarian: 11877 Branching factor: 3.0760669708251953 Langkah: 8 / 8 Step: <math>\emptyset L+2 P+1 B+1 F-3 G-1 H-2 P+3</math></p>	 <p>Waktu eksekusi: 34.0848000000302494ms Tick count: 816 Node diperiksa: 1374 Banyak pencarian: 8323 Branching factor: 2.933656692504883 Langkah: 8 / 8 Step: <math>\emptyset H-1 L+2 P+1 B+1 F-3 G-1 H-2 P+3</math></p>													

### 4.3. Jam-3

Nama : Jam-3 Banyak Step Optimal : 14 Step Optimal : $\emptyset$ C-2 F-3 B+3 D-1 E-1 C+3 P+1 D-3 P-1 C-3 B-3 C+3 F+3 P+3		6 6 5 ..... ..... .PPC..K .BBC.F .D.C.F .DEE.F
UCS	GBFS Car Distance	GBFS Car Blocked
 <p>Waktu eksekusi: 9.462700000032783ms          Tick count: 772          Node diperiksa: 816          Banyak pencarian: 7778          Branching factor: 1.7886276245117188          Langkah: 14 / 14          Step: <math>\emptyset</math> C-2 F-3 B+3 D-1 E-1 C+3 P+1                D-3 P-1 C-3 B-3 C+3 F+3 P+3</p>	 <p>Waktu eksekusi: 2.11169999993100762ms          Tick count: 299          Node diperiksa: 647          Banyak pencarian: 2669          Branching factor: 1.583160400390625          Langkah: 15 / 15          Step: <math>\emptyset</math> C-2 F-3 B+3 D-1 E-1 C+3 P+2                D-3 P-2 C-3 B-3 C+3 P+2 F+3 P+3</p>	 <p>Waktu eksekusi: 6.271599999628961ms          Tick count: 394          Node diperiksa: 472          Banyak pencarian: 3537          Branching factor: 1.5635147094726562          Langkah: 16 / 16          Step: <math>\emptyset</math> C-2 B-1 D-1 E-1 F-3 B+2 C+3 P+3</p>
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive
 <p>Waktu eksekusi: 6.407200000248849ms          Tick count: 657          Node diperiksa: 826          Banyak pencarian: 6364          Branching factor: 1.6889877319335938          Langkah: 15 / 15          Step: <math>\emptyset</math> C-2 F-3 B+3 E+2 C+3 P+2 D-4                P-2 C-3 B-3 E-3 C-3 P-2 F-3 P+3</p>	 <p>Waktu eksekusi: 23.245499999262393ms          Tick count: 704          Node diperiksa: 812          Banyak pencarian: 7006          Branching factor: 1.7739944458007812          Langkah: 14 / 14          Step: <math>\emptyset</math> C-2 F-3 B+3 E+2 C+3 P+1 D-4                P-1 C-3 B-3 E-3 C-3 F-3 P+3</p>	 <p>Waktu eksekusi: 29.9964000005275ms          Tick count: 574          Node diperiksa: 758          Banyak pencarian: 5642          Branching factor: 1.7439651489257812          Langkah: 14 / 14          Step: <math>\emptyset</math> C-2 F-3 B+3 D-1 E-1 C+3 P+1                D-3 P-1 C-3 B-3 C+3 F+3 P+3</p>

#### 4.4. Jam-4

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-4</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>9</td><td>6</td></tr> <tr> <td>Step Optimal</td><td>:</td><td><math>\emptyset \text{ D+3 P-1 B-3 P+1}</math> <math>\text{D-3 C-3 G-2 F+3 P+3}</math></td><td>D .. F .. D .. F .. DPPF .. K .. BCCC .. B .. E .. GGGE</td></tr> </tbody> </table>				Nama	:	Jam-4	6 6	Banyak Step Optimal	:	9	6	Step Optimal	:	$\emptyset \text{ D+3 P-1 B-3 P+1}$ $\text{D-3 C-3 G-2 F+3 P+3}$	D .. F .. D .. F .. DPPF .. K .. BCCC .. B .. E .. GGGE
Nama	:	Jam-4	6 6												
Banyak Step Optimal	:	9	6												
Step Optimal	:	$\emptyset \text{ D+3 P-1 B-3 P+1}$ $\text{D-3 C-3 G-2 F+3 P+3}$	D .. F .. D .. F .. DPPF .. K .. BCCC .. B .. E .. GGGE												
<table border="1"> <thead> <tr> <th>UCS</th> <th>GBFS Car Distance</th> <th>GBFS Car Blocked</th> </tr> </thead> <tbody> <tr> <td> <p>Waktu eksekusi: 4.712600000202656ms Tick count: 366 Node diperiksa: 410 Banyak pencarian: 3476 Branching factor: 2.324453353881836 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p> </td><td> <p>Waktu eksekusi: 1.8874000003561378ms Tick count: 212 Node diperiksa: 304 Banyak pencarian: 2186 Branching factor: 2.1967544555664062 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p> </td><td> <p>Waktu eksekusi: 0.9607999995350838ms Tick count: 91 Node diperiksa: 259 Banyak pencarian: 836 Branching factor: 1.9500732421875 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p> </td></tr> </tbody> </table>		UCS	GBFS Car Distance	GBFS Car Blocked	<p>Waktu eksekusi: 4.712600000202656ms Tick count: 366 Node diperiksa: 410 Banyak pencarian: 3476 Branching factor: 2.324453353881836 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p>	<p>Waktu eksekusi: 1.8874000003561378ms Tick count: 212 Node diperiksa: 304 Banyak pencarian: 2186 Branching factor: 2.1967544555664062 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p>	<p>Waktu eksekusi: 0.9607999995350838ms Tick count: 91 Node diperiksa: 259 Banyak pencarian: 836 Branching factor: 1.9500732421875 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p>								
UCS	GBFS Car Distance	GBFS Car Blocked													
<p>Waktu eksekusi: 4.712600000202656ms Tick count: 366 Node diperiksa: 410 Banyak pencarian: 3476 Branching factor: 2.324453353881836 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p>	<p>Waktu eksekusi: 1.8874000003561378ms Tick count: 212 Node diperiksa: 304 Banyak pencarian: 2186 Branching factor: 2.1967544555664062 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p>	<p>Waktu eksekusi: 0.9607999995350838ms Tick count: 91 Node diperiksa: 259 Banyak pencarian: 836 Branching factor: 1.9500732421875 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p>													
<table border="1"> <thead> <tr> <th>A* Car Distance</th> <th>A* Car Blocked</th> <th>A* Car Blocked Recursive</th> </tr> </thead> <tbody> <tr> <td> <p>Waktu eksekusi: 1.689100000075996ms Tick count: 291 Node diperiksa: 356 Banyak pencarian: 2855 Branching factor: 2.269533157349633 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p> </td><td> <p>Waktu eksekusi: 1.4650999996811152ms Tick count: 233 Node diperiksa: 353 Banyak pencarian: 2244 Branching factor: 2.2038097381591797 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p> </td><td> <p>Waktu eksekusi: 5.942999999970198ms Tick count: 155 Node diperiksa: 295 Banyak pencarian: 1415 Branching factor: 2.0822677612304688 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p> </td></tr> </tbody> </table>		A* Car Distance	A* Car Blocked	A* Car Blocked Recursive	<p>Waktu eksekusi: 1.689100000075996ms Tick count: 291 Node diperiksa: 356 Banyak pencarian: 2855 Branching factor: 2.269533157349633 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p>	<p>Waktu eksekusi: 1.4650999996811152ms Tick count: 233 Node diperiksa: 353 Banyak pencarian: 2244 Branching factor: 2.2038097381591797 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p>	<p>Waktu eksekusi: 5.942999999970198ms Tick count: 155 Node diperiksa: 295 Banyak pencarian: 1415 Branching factor: 2.0822677612304688 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p>								
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive													
<p>Waktu eksekusi: 1.689100000075996ms Tick count: 291 Node diperiksa: 356 Banyak pencarian: 2855 Branching factor: 2.269533157349633 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p>	<p>Waktu eksekusi: 1.4650999996811152ms Tick count: 233 Node diperiksa: 353 Banyak pencarian: 2244 Branching factor: 2.2038097381591797 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p>	<p>Waktu eksekusi: 5.942999999970198ms Tick count: 155 Node diperiksa: 295 Banyak pencarian: 1415 Branching factor: 2.0822677612304688 Langkah: 9 / 9 Step: <math>\emptyset \text{ D+3 P-1 B-3 P+1 D-3 C-3 G-2}</math> <math>\text{F+3 P+3}</math></p>													

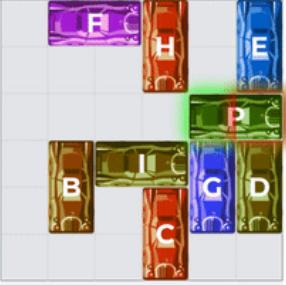
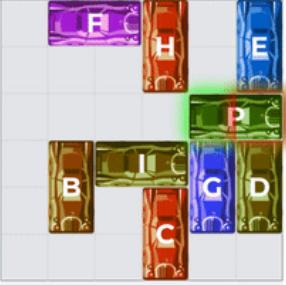
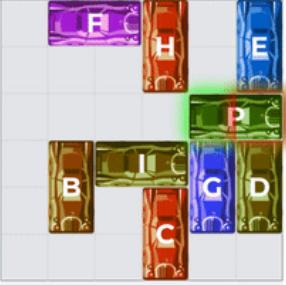
#### 4.5. Jam-5

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-5</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>9</td><td>10</td></tr> <tr> <td>Step Optimal</td><td>:</td><td><math>\emptyset</math> B+1 L-1 H-1 I-3 J+1 D-3 F+3 C+2 P+3</td><td>BB.F.G L..FCG LPPFCJK LHHHCJ E ... II E ... DD</td></tr> </tbody> </table>				Nama	:	Jam-5	6 6	Banyak Step Optimal	:	9	10	Step Optimal	:	$\emptyset$ B+1 L-1 H-1 I-3 J+1 D-3 F+3 C+2 P+3	BB.F.G L..FCG LPPFCJK LHHHCJ E ... II E ... DD
Nama	:	Jam-5	6 6												
Banyak Step Optimal	:	9	10												
Step Optimal	:	$\emptyset$ B+1 L-1 H-1 I-3 J+1 D-3 F+3 C+2 P+3	BB.F.G L..FCG LPPFCJK LHHHCJ E ... II E ... DD												
<p>Waktu eksekusi: 68.45220000017434ms Tick count: 2237 Node diperiksa: 2504 Banyak pencarian: 24622 Branching factor: 2.936534881591797 Langkah: 9 / 9 Step: <math>\emptyset</math> B+1 L-1 I-3 J+1 D-3 F+3 C+2 P+3</p>		<p>Waktu eksekusi: 12.954400000162423ms Tick count: 805 Node diperiksa: 1379 Banyak pencarian: 8513 Branching factor: 2.1507606506347656 Langkah: 11 / 11 Step: <math>\emptyset</math> B+1 L-1 I-3 J+1 D-3 F+3 C+2 P+3</p>	<p>Waktu eksekusi: 0.9499000003561378ms Tick count: 75 Node diperiksa: 437 Banyak pencarian: 788 Branching factor: 1.6078567504882812 Langkah: 12 / 12 Step: <math>\emptyset</math> I-3 J+1 D-2 C+2 B+1 L-1 H-1 P+2 F+3</p>												
<p>Waktu eksekusi: 23.222999999299645ms Tick count: 1893 Node diperiksa: 2309 Banyak pencarian: 20758 Branching factor: 2.5724239349365234 Langkah: 10 / 10 Step: <math>\emptyset</math> B+1 L-1 I-3 D-3 F+3 C+2 P+2 J+1</p>		<p>Waktu eksekusi: 16.032999999821186ms Tick count: 814 Node diperiksa: 1416 Banyak pencarian: 8901 Branching factor: 2.6027565002441406 Langkah: 9 / 9 Step: <math>\emptyset</math> I-3 J+1 D-3 C+2 B+1 L-1 H-1 F+3 P+3</p>	<p>Waktu eksekusi: 30.061700000427663ms Tick count: 718 Node diperiksa: 1243 Banyak pencarian: 7819 Branching factor: 2.5627593994140625 Langkah: 9 / 9 Step: <math>\emptyset</math> I-3 J+1 D-3 C+2 B+1 L-1 H-1 F+3 P+3</p>												

#### 4.6. Jam-6

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-6</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>9</td><td>10</td></tr> <tr> <td>Step Optimal</td><td>:</td><td><math>\emptyset P-1 I-3 E+1 D-1</math> <math>J-3 F+2 L+2 C+1 P+4</math></td><td>BB.G.. HH.GFL .PPCFLK EEICFL D.IC.. D .. JJJ</td></tr> </tbody> </table>				Nama	:	Jam-6	6 6	Banyak Step Optimal	:	9	10	Step Optimal	:	$\emptyset P-1 I-3 E+1 D-1$ $J-3 F+2 L+2 C+1 P+4$	BB.G.. HH.GFL .PPCFLK EEICFL D.IC.. D .. JJJ
Nama	:	Jam-6	6 6												
Banyak Step Optimal	:	9	10												
Step Optimal	:	$\emptyset P-1 I-3 E+1 D-1$ $J-3 F+2 L+2 C+1 P+4$	BB.G.. HH.GFL .PPCFLK EEICFL D.IC.. D .. JJJ												
<table border="1"> <thead> <tr> <th>UCS</th> <th>GBFS Car Distance</th> <th>GBFS Car Blocked</th> </tr> </thead> <tbody> <tr> <td>  <p>Waktu eksekusi: 30.51370000001043ms Tick count: 1649 Node diperiksa: 1864 Banyak pencarian: 16965 Branching factor: 2.8101348876953125 Langkah: 9 / 9 Step: <math>\emptyset P-1 I-3 E+1 D-1 J-3 F+2 L+2 C+1</math> <math>P+4</math></p> </td><td>  <p>Waktu eksekusi: 8.552000000141561ms Tick count: 408 Node diperiksa: 806 Banyak pencarian: 3904 Branching factor: 1.7722702026367188 Langkah: 13 / 13 Step: <math>\emptyset I+1 H+1 E+1 D-1 C-1 J-3 C+1</math> <math>P+1 F+2 P+1 L+2 D+1</math></p> </td><td>  <p>Waktu eksekusi: 0.35829999949783087ms Tick count: 37 Node diperiksa: 239 Banyak pencarian: 372 Branching factor: 1.56128699262695312 Langkah: 11 / 11 Step: <math>\emptyset J-1 L+2 I-1 F+2 D-1 I-3 E+1 D-1</math> <math>J+1 C+1 P+4</math></p> </td></tr> </tbody> </table>		UCS	GBFS Car Distance	GBFS Car Blocked	 <p>Waktu eksekusi: 30.51370000001043ms Tick count: 1649 Node diperiksa: 1864 Banyak pencarian: 16965 Branching factor: 2.8101348876953125 Langkah: 9 / 9 Step: <math>\emptyset P-1 I-3 E+1 D-1 J-3 F+2 L+2 C+1</math> <math>P+4</math></p>	 <p>Waktu eksekusi: 8.552000000141561ms Tick count: 408 Node diperiksa: 806 Banyak pencarian: 3904 Branching factor: 1.7722702026367188 Langkah: 13 / 13 Step: <math>\emptyset I+1 H+1 E+1 D-1 C-1 J-3 C+1</math> <math>P+1 F+2 P+1 L+2 D+1</math></p>	 <p>Waktu eksekusi: 0.35829999949783087ms Tick count: 37 Node diperiksa: 239 Banyak pencarian: 372 Branching factor: 1.56128699262695312 Langkah: 11 / 11 Step: <math>\emptyset J-1 L+2 I-1 F+2 D-1 I-3 E+1 D-1</math> <math>J+1 C+1 P+4</math></p>								
UCS	GBFS Car Distance	GBFS Car Blocked													
 <p>Waktu eksekusi: 30.51370000001043ms Tick count: 1649 Node diperiksa: 1864 Banyak pencarian: 16965 Branching factor: 2.8101348876953125 Langkah: 9 / 9 Step: <math>\emptyset P-1 I-3 E+1 D-1 J-3 F+2 L+2 C+1</math> <math>P+4</math></p>	 <p>Waktu eksekusi: 8.552000000141561ms Tick count: 408 Node diperiksa: 806 Banyak pencarian: 3904 Branching factor: 1.7722702026367188 Langkah: 13 / 13 Step: <math>\emptyset I+1 H+1 E+1 D-1 C-1 J-3 C+1</math> <math>P+1 F+2 P+1 L+2 D+1</math></p>	 <p>Waktu eksekusi: 0.35829999949783087ms Tick count: 37 Node diperiksa: 239 Banyak pencarian: 372 Branching factor: 1.56128699262695312 Langkah: 11 / 11 Step: <math>\emptyset J-1 L+2 I-1 F+2 D-1 I-3 E+1 D-1</math> <math>J+1 C+1 P+4</math></p>													
<table border="1"> <thead> <tr> <th>A* Car Distance</th> <th>A* Car Blocked</th> <th>A* Car Blocked Recursive</th> </tr> </thead> <tbody> <tr> <td>  <p>Waktu eksekusi: 18.358800000051111ms Tick count: 1582 Node diperiksa: 1810 Banyak pencarian: 16226 Branching factor: 2.2914371490478516 Langkah: 11 / 11 Step: <math>\emptyset P-1 I-3 P+1 E+1 D-1 J-3 F+2 C+1</math> <math>P+2 L+2 P+4</math></p> </td><td>  <p>Waktu eksekusi: 15.85950000025332ms Tick count: 796 Node diperiksa: 1121 Banyak pencarian: 8420 Branching factor: 2.334810256958008 Langkah: 10 / 10 Step: <math>\emptyset J-2 F+2 L+2 D-1 I-3 E+1 D-1 J-1</math> <math>C+1 P+4</math></p> </td><td>  <p>Waktu eksekusi: 36.480900000078082ms Tick count: 1033 Node diperiksa: 1288 Banyak pencarian: 10629 Branching factor: 2.399057388305664 Langkah: 10 / 10 Step: <math>\emptyset J-2 F+2 L+2 P-1 I-3 E+1 D-1 J-1</math> <math>C+1 P+4</math></p> </td></tr> </tbody> </table>		A* Car Distance	A* Car Blocked	A* Car Blocked Recursive	 <p>Waktu eksekusi: 18.358800000051111ms Tick count: 1582 Node diperiksa: 1810 Banyak pencarian: 16226 Branching factor: 2.2914371490478516 Langkah: 11 / 11 Step: <math>\emptyset P-1 I-3 P+1 E+1 D-1 J-3 F+2 C+1</math> <math>P+2 L+2 P+4</math></p>	 <p>Waktu eksekusi: 15.85950000025332ms Tick count: 796 Node diperiksa: 1121 Banyak pencarian: 8420 Branching factor: 2.334810256958008 Langkah: 10 / 10 Step: <math>\emptyset J-2 F+2 L+2 D-1 I-3 E+1 D-1 J-1</math> <math>C+1 P+4</math></p>	 <p>Waktu eksekusi: 36.480900000078082ms Tick count: 1033 Node diperiksa: 1288 Banyak pencarian: 10629 Branching factor: 2.399057388305664 Langkah: 10 / 10 Step: <math>\emptyset J-2 F+2 L+2 P-1 I-3 E+1 D-1 J-1</math> <math>C+1 P+4</math></p>								
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive													
 <p>Waktu eksekusi: 18.358800000051111ms Tick count: 1582 Node diperiksa: 1810 Banyak pencarian: 16226 Branching factor: 2.2914371490478516 Langkah: 11 / 11 Step: <math>\emptyset P-1 I-3 P+1 E+1 D-1 J-3 F+2 C+1</math> <math>P+2 L+2 P+4</math></p>	 <p>Waktu eksekusi: 15.85950000025332ms Tick count: 796 Node diperiksa: 1121 Banyak pencarian: 8420 Branching factor: 2.334810256958008 Langkah: 10 / 10 Step: <math>\emptyset J-2 F+2 L+2 D-1 I-3 E+1 D-1 J-1</math> <math>C+1 P+4</math></p>	 <p>Waktu eksekusi: 36.480900000078082ms Tick count: 1033 Node diperiksa: 1288 Banyak pencarian: 10629 Branching factor: 2.399057388305664 Langkah: 10 / 10 Step: <math>\emptyset J-2 F+2 L+2 P-1 I-3 E+1 D-1 J-1</math> <math>C+1 P+4</math></p>													

#### 4.7. Jam-7

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-7</td><td>6</td><td>6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>13</td><td>8</td><td></td></tr> <tr> <td>Step Optimal</td><td>:</td><td><math>\emptyset</math> G+3 D+1 E+1 F+2 H-1 P+1 B+3 P-1 H+1 F-3 H-1 E-1 P+3</td><td>.BFFGE .B.HGE .PPH.DK .II.D ... C .. ... C ..</td><td></td></tr> </tbody> </table>					Nama	:	Jam-7	6	6	Banyak Step Optimal	:	13	8		Step Optimal	:	$\emptyset$ G+3 D+1 E+1 F+2 H-1 P+1 B+3 P-1 H+1 F-3 H-1 E-1 P+3	.BFFGE .B.HGE .PPH.DK .II.D ... C .. ... C ..	
Nama	:	Jam-7	6	6															
Banyak Step Optimal	:	13	8																
Step Optimal	:	$\emptyset$ G+3 D+1 E+1 F+2 H-1 P+1 B+3 P-1 H+1 F-3 H-1 E-1 P+3	.BFFGE .B.HGE .PPH.DK .II.D ... C .. ... C ..																
<table border="1"> <thead> <tr> <th>UCS</th> <th>GBFS Car Distance</th> <th>GBFS Car Blocked</th> </tr> </thead> <tbody> <tr> <td>  <p>Waktu eksekusi: 79.7105000000447ms Tick count: 4943 Node diperiksa: 6011 Banyak pencarian: 60951 Branching factor: 2.2294158935546875 Langkah: 13 / 13 Step: <math>\emptyset</math> G+3 D+1 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+3</p> </td><td>  <p>Waktu eksekusi: 57.26919999998063ms Tick count: 1753 Node diperiksa: 3823 Banyak pencarian: 20048 Branching factor: 1.7625350952148438 Langkah: 16 / 16 Step: <math>\emptyset</math> G+3 D+1 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+2 G+2 P+1 E-1 P+3</p> </td><td>  <p>Waktu eksekusi: 20.5518000000481308ms Tick count: 1570 Node diperiksa: 2636 Banyak pencarian: 16834 Branching factor: 1.8997716088867188 Langkah: 14 / 14 Step: <math>\emptyset</math> D+1 G+3 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+2 H-2 P+3</p> </td></tr> </tbody> </table>			UCS	GBFS Car Distance	GBFS Car Blocked	 <p>Waktu eksekusi: 79.7105000000447ms Tick count: 4943 Node diperiksa: 6011 Banyak pencarian: 60951 Branching factor: 2.2294158935546875 Langkah: 13 / 13 Step: <math>\emptyset</math> G+3 D+1 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+3</p>	 <p>Waktu eksekusi: 57.26919999998063ms Tick count: 1753 Node diperiksa: 3823 Banyak pencarian: 20048 Branching factor: 1.7625350952148438 Langkah: 16 / 16 Step: <math>\emptyset</math> G+3 D+1 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+2 G+2 P+1 E-1 P+3</p>	 <p>Waktu eksekusi: 20.5518000000481308ms Tick count: 1570 Node diperiksa: 2636 Banyak pencarian: 16834 Branching factor: 1.8997716088867188 Langkah: 14 / 14 Step: <math>\emptyset</math> D+1 G+3 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+2 H-2 P+3</p>											
UCS	GBFS Car Distance	GBFS Car Blocked																	
 <p>Waktu eksekusi: 79.7105000000447ms Tick count: 4943 Node diperiksa: 6011 Banyak pencarian: 60951 Branching factor: 2.2294158935546875 Langkah: 13 / 13 Step: <math>\emptyset</math> G+3 D+1 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+3</p>	 <p>Waktu eksekusi: 57.26919999998063ms Tick count: 1753 Node diperiksa: 3823 Banyak pencarian: 20048 Branching factor: 1.7625350952148438 Langkah: 16 / 16 Step: <math>\emptyset</math> G+3 D+1 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+2 G+2 P+1 E-1 P+3</p>	 <p>Waktu eksekusi: 20.5518000000481308ms Tick count: 1570 Node diperiksa: 2636 Banyak pencarian: 16834 Branching factor: 1.8997716088867188 Langkah: 14 / 14 Step: <math>\emptyset</math> D+1 G+3 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+2 H-2 P+3</p>																	
<table border="1"> <thead> <tr> <th>A* Car Distance</th> <th>A* Car Blocked</th> <th>A* Car Blocked Recursive</th> </tr> </thead> <tbody> <tr> <td>  <p>Waktu eksekusi: 36.270099999938309ms Tick count: 3597 Node diperiksa: 5117 Banyak pencarian: 42532 Branching factor: 2.040468215942383 Langkah: 14 / 14 Step: <math>\emptyset</math> G+3 D+1 E+1 F+2 H-1 P+2 B+3 P+2 H+1 F-3 H-1 P+2 E-1 P+3</p> </td><td>  <p>Waktu eksekusi: 70.79330000001937ms Tick count: 3219 Node diperiksa: 4866 Banyak pencarian: 38091 Branching factor: 2.144855499267578 Langkah: 13 / 13 Step: <math>\emptyset</math> D+1 G+3 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+3</p> </td><td>  <p>Waktu eksekusi: 78.386800000044405ms Tick count: 3217 Node diperiksa: 4857 Banyak pencarian: 38064 Branching factor: 2.1447296142578125 Langkah: 13 / 13 Step: <math>\emptyset</math> D+1 G+3 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+3</p> </td></tr> </tbody> </table>			A* Car Distance	A* Car Blocked	A* Car Blocked Recursive	 <p>Waktu eksekusi: 36.270099999938309ms Tick count: 3597 Node diperiksa: 5117 Banyak pencarian: 42532 Branching factor: 2.040468215942383 Langkah: 14 / 14 Step: <math>\emptyset</math> G+3 D+1 E+1 F+2 H-1 P+2 B+3 P+2 H+1 F-3 H-1 P+2 E-1 P+3</p>	 <p>Waktu eksekusi: 70.79330000001937ms Tick count: 3219 Node diperiksa: 4866 Banyak pencarian: 38091 Branching factor: 2.144855499267578 Langkah: 13 / 13 Step: <math>\emptyset</math> D+1 G+3 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+3</p>	 <p>Waktu eksekusi: 78.386800000044405ms Tick count: 3217 Node diperiksa: 4857 Banyak pencarian: 38064 Branching factor: 2.1447296142578125 Langkah: 13 / 13 Step: <math>\emptyset</math> D+1 G+3 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+3</p>											
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive																	
 <p>Waktu eksekusi: 36.270099999938309ms Tick count: 3597 Node diperiksa: 5117 Banyak pencarian: 42532 Branching factor: 2.040468215942383 Langkah: 14 / 14 Step: <math>\emptyset</math> G+3 D+1 E+1 F+2 H-1 P+2 B+3 P+2 H+1 F-3 H-1 P+2 E-1 P+3</p>	 <p>Waktu eksekusi: 70.79330000001937ms Tick count: 3219 Node diperiksa: 4866 Banyak pencarian: 38091 Branching factor: 2.144855499267578 Langkah: 13 / 13 Step: <math>\emptyset</math> D+1 G+3 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+3</p>	 <p>Waktu eksekusi: 78.386800000044405ms Tick count: 3217 Node diperiksa: 4857 Banyak pencarian: 38064 Branching factor: 2.1447296142578125 Langkah: 13 / 13 Step: <math>\emptyset</math> D+1 G+3 E+1 F+2 H-1 P+1 B+3 P+1 H+1 F-3 H-1 E-1 P+3</p>																	

#### 4.8. Jam-8

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-8</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>12</td><td>13</td></tr> <tr> <td>Step Optimal</td><td>:</td><td><math>\emptyset</math> B-3 G-2 H-1 E-2 I-2 P+3 J-1 N-2 L-1 C-1 F+3 P+1</td><td>... BBF .. GGHF PPEIHFK DDEIJJ MMNLLL OONCCC</td></tr> </tbody> </table>				Nama	:	Jam-8	6 6	Banyak Step Optimal	:	12	13	Step Optimal	:	$\emptyset$ B-3 G-2 H-1 E-2 I-2 P+3 J-1 N-2 L-1 C-1 F+3 P+1	... BBF .. GGHF PPEIHFK DDEIJJ MMNLLL OONCCC
Nama	:	Jam-8	6 6												
Banyak Step Optimal	:	12	13												
Step Optimal	:	$\emptyset$ B-3 G-2 H-1 E-2 I-2 P+3 J-1 N-2 L-1 C-1 F+3 P+1	... BBF .. GGHF PPEIHFK DDEIJJ MMNLLL OONCCC												
UCS	GBFS Car Distance	GBFS Car Blocked													
<p>Waktu eksekusi: 10.763900000602007ms Tick count: 950 Node diperiksa: 951 Banyak pencarian: 6465 Branching factor: 1.9573516845703125 Langkah: 12 / 12 Step: <math>\emptyset</math> B-3 G-2 H-1 E-2 I-2 P+3 J-1 N-2 L-1 C-1 F+3 P+1</p>	<p>Waktu eksekusi: 10.769799999692712ms Tick count: 186 Node diperiksa: 344 Banyak pencarian: 1029 Branching factor: 1.5204544067382812 Langkah: 14 / 14 Step: <math>\emptyset</math> B-3 G-2 E-2 P+1 I-2 P+2 H-1 P+3 J-1 N-2 L-1 C-1 F+3 P+1</p>	<p>Waktu eksekusi: 0.9004000006243587ms Tick count: 149 Node diperiksa: 325 Banyak pencarian: 1000 Branching factor: 1.46893310546875 Langkah: 15 / 15 Step: <math>\emptyset</math> B-3 H-1 B-1 G-1 I-2 B-1 G-1 E-2 P+3 J-1 N-2 L-1 C-1 F+3 P+1</p>													
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive													
<p>Waktu eksekusi: 11.397499999962747ms Tick count: 613 Node diperiksa: 767 Banyak pencarian: 4282 Branching factor: 1.884857177734375 Langkah: 12 / 12 Step: <math>\emptyset</math> B-3 G-2 H-1 E-2 I-2 P+3 J-1 N-2 L-1 C-1 F+3 P+1</p>	<p>Waktu eksekusi: 11.53830000013113ms Tick count: 737 Node diperiksa: 861 Banyak pencarian: 5221 Branching factor: 1.9194793701171875 Langkah: 12 / 12 Step: <math>\emptyset</math> B-3 H-1 G-2 E-2 I-2 P+3 J-1 N-2 L-1 C-1 F+3 P+1</p>	<p>Waktu eksekusi: 30.89530000090599ms Tick count: 837 Node diperiksa: 935 Banyak pencarian: 5901 Branching factor: 1.94110107421875 Langkah: 12 / 12 Step: <math>\emptyset</math> B-3 H-1 G-2 E-2 I-2 J-1 P+3 N-2 L-1 C-1 F+3 P+1</p>													

#### 4.9. Jam-9

Nama : Jam-9		6 6
Banyak Step Optimal : 12		11
Step Optimal : $\emptyset$ P+1 F+1 L-3 C-1 E+2 I-1 P+1 B+1 G-1 H-1 D-2 P+2		.BGGHH .B.EII PP.EFDK LCCCFD L.J.FM L.J..M
UCS	GBFS Car Distance	GBFS Car Blocked
 <p>Waktu eksekusi: 3.9005999993532896ms Tick count: 680 Node diperiksa: 875 Banyak pencarian: 5897 Branching factor: 1.94097900390625 Langkah: 12 / 12 Step: <math>\emptyset</math> P+1 F+1 L-3 C-1 E+2 I-1 P+1 B+1 G-1 H-1 D-2 P+2</p>	 <p>Waktu eksekusi: 3.028500000014901ms Tick count: 116 Node diperiksa: 324 Banyak pencarian: 894 Branching factor: 1.5589218139648438 Langkah: 13 / 13 Step: <math>\emptyset</math> P+1 F+1 L-3 C-1 E+2 P+1 F+1 P+2 B+1 G-1 H-1 D-2 P+2</p>	 <p>Waktu eksekusi: 1.1398000000044703ms Tick count: 156 Node diperiksa: 394 Banyak pencarian: 1330 Branching factor: 1.6902313232421875 Langkah: 12 / 12 Step: <math>\emptyset</math> F+1 P+1 L-3 C-1 E+2 I-1 P+1 B+1 G-1 H-1 D-2 P+2</p>
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive
 <p>Waktu eksekusi: 3.8479000004008412ms Tick count: 429 Node diperiksa: 646 Banyak pencarian: 3640 Branching factor: 1.8568649291992188 Langkah: 12 / 12 Step: <math>\emptyset</math> P+1 F+1 L-3 C-1 E+2 P+2 B+1 G-1 H-1 I-1 D-2 P+1</p>	 <p>Waktu eksekusi: 8.31400000024587ms Tick count: 483 Node diperiksa: 737 Banyak pencarian: 4155 Branching factor: 1.8796463012695312 Langkah: 12 / 12 Step: <math>\emptyset</math> F+1 P+1 L-3 C-1 E+2 H-1 P+1 B+1 G-1 H-1 D-2 P+2</p>	 <p>Waktu eksekusi: 12.182200000621378ms Tick count: 486 Node diperiksa: 737 Banyak pencarian: 471 Branching factor: 1.88031005859375 Langkah: 12 / 12 Step: <math>\emptyset</math> F+1 P+1 L-3 C-1 E+2 I-1 P+1 B+1 G-1 H-1 D-2 P+2</p>

#### 4.10. Jam-10

Nama : Jam-10		6 6
Banyak Step Optimal : 17		11
Step Optimal : $\emptyset$ H-1 F-1 P+2 C+2 G+4 B+1 E+1 L-2 P-2 C-3 F+1 H+1 I-4 D-1 M-1 F+2 P+3		BBG.HH EEG..F LPP..FK LCCC.F L..IDD JJ.IMM
<p style="text-align: center;">UCS</p>		<p style="text-align: center;">GBFS Car Distance</p>
<p style="text-align: center;">GBFS Car Blocked</p>		
<p style="text-align: center;">A* Car Distance</p>		<p style="text-align: center;">A* Car Blocked</p>
<p style="text-align: center;">A* Car Blocked Recursive</p>		

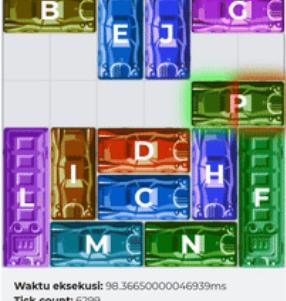
#### 4.11. Jam-11

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-11</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>25</td><td>7</td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> E+3 P-1 F-2 C-1            G-3 C+1 D+1 F+3 P+1            E-3 C-3 H+1 B+3 H-1            C+3 E+3 P-1 F-4 P+1            E-3 C-3 G+2 D-3 H+3            P+3         </td><td>           E .. H ..            EPPH .. K            .. FCCC            .. F .. G            .. DDDG         </td></tr> </tbody> </table>	Nama	:	Jam-11	6 6	Banyak Step Optimal	:	25	7	Step Optimal	:	$\emptyset$ E+3 P-1 F-2 C-1 G-3 C+1 D+1 F+3 P+1 E-3 C-3 H+1 B+3 H-1 C+3 E+3 P-1 F-4 P+1 E-3 C-3 G+2 D-3 H+3 P+3	E .. H .. EPPH .. K .. FCCC .. F .. G .. DDDG	
Nama	:	Jam-11	6 6										
Banyak Step Optimal	:	25	7										
Step Optimal	:	$\emptyset$ E+3 P-1 F-2 C-1 G-3 C+1 D+1 F+3 P+1 E-3 C-3 H+1 B+3 H-1 C+3 E+3 P-1 F-4 P+1 E-3 C-3 G+2 D-3 H+3 P+3	E .. H .. EPPH .. K .. FCCC .. F .. G .. DDDG										
<p style="text-align: center;">UCS</p>  <p>           Waktu eksekusi: 10.87360000051558ms            Tick count: 828            Node diperiksa: 847            Banyak pencarian: 6879            Branching factor: 1.349029541015625            Langkah: 25 / 25            Step: <math>\emptyset</math> E+3 P-1 F-2 C-1 G+1 D+1            F+3 P+1 E-3 C-3 H+1 B+3 H-1 C+3 E+3            P-1 F-4 P+1 E-3 C-3 G+2 D-3 H+3 P+3         </p>	<p style="text-align: center;">GBFS Car Distance</p>  <p>           Waktu eksekusi: 11.749900000169873ms            Tick count: 700            Node diperiksa: 808            Banyak pencarian: 5887            Branching factor: 1.25714111328125            Langkah: 31 / 31            Step: <math>\emptyset</math> E+3 B-1 P-1 F-3 P+1 C-1 G-3            C+1 P-1 F+3 P+1 B+1 E-3 D+1 F+1 C-3            H+1 B+3 H-1 C+3 E-3 P-1 F-4 P+1 E-3            C-3 D-3 H+3 P+2 G+2 P+3         </p>	<p style="text-align: center;">GBFS Car Blocked</p>  <p>           Waktu eksekusi: 8.482699999585748ms            Tick count: 554            Node diperiksa: 754            Banyak pencarian: 4492            Branching factor: 1.24462890625            Langkah: 31 / 31            Step: <math>\emptyset</math> E+3 B-1 P-1 F-3 C-1 G-4 H+1            D+1 F+4 B+1 P+1 E-3 C-3 H+1 G+3 B+3            H-1 C+1 E+3 P-1 G-2 C-2 F-4 C-1 G+2            P+1 E-3 C-2 D-3 H+3 P+3         </p>											
<p style="text-align: center;">A* Car Distance</p>  <p>           Waktu eksekusi: 4.101400000043213ms            Tick count: 801            Node diperiksa: 860            Banyak pencarian: 6679            Branching factor: 1.3299179077148438            Langkah: 26 / 26            Step: <math>\emptyset</math> E+3 P-1 F-2 C-1 G-3 C+1 D+1            F+3 P+1 E-3 C-3 H+1 B+3 H-1 C+3 E+3            P-1 F-4 P+1 E-3 C-3 D-3 H+3 P+2 G+2            P+3         </p>	<p style="text-align: center;">A* Car Blocked</p>  <p>           Waktu eksekusi: 4.88990000076592ms            Tick count: 769            Node diperiksa: 820            Banyak pencarian: 6392            Branching factor: 1.29740142882226562            Langkah: 28 / 28            Step: <math>\emptyset</math> E+3 B-1 P-1 F-3 C-1 G-4 C+1            D+1 F+4 B+1 P+1 E-3 C-3 H+1 G+1 B+3            H-1 C+3 E+3 P-1 F-4 P+1 E-3 C-3 G+2            D-3 H+3 P+3         </p>	<p style="text-align: center;">A* Car Blocked Recursive</p>  <p>           Waktu eksekusi: 21.37100000027567ms            Tick count: 753            Node diperiksa: 811            Banyak pencarian: 6283            Branching factor: 1.326385498046875            Langkah: 26 / 26            Step: <math>\emptyset</math> E+3 P-1 F-2 C-1 G-4 C+1 D+1            F+3 P+1 E-3 C-3 H+1 G+1 B+3 H-1 C+3            E-3 P-1 F-4 P+1 E-3 C-3 G+2 D-3 H+3            P+3         </p>											

#### 4.12. Jam-12

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-12</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>17</td><td>7</td></tr> <tr> <td>Step Optimal</td><td>:</td><td><math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 F-1 G-1 P+3 B+1 F-2 H-3 C-1 D-1 E+3 P+1</td><td>BFF .. E B.H .. E PPH .. EK .. HCCC . . . G. DDD.G.</td></tr> </tbody> </table>				Nama	:	Jam-12	6 6	Banyak Step Optimal	:	17	7	Step Optimal	:	$\emptyset$ F+2 H-1 C-2 G-3 C+2 D+3 H+3 F-1 G-1 P+3 B+1 F-2 H-3 C-1 D-1 E+3 P+1	BFF .. E B.H .. E PPH .. EK .. HCCC . . . G. DDD.G.
Nama	:	Jam-12	6 6												
Banyak Step Optimal	:	17	7												
Step Optimal	:	$\emptyset$ F+2 H-1 C-2 G-3 C+2 D+3 H+3 F-1 G-1 P+3 B+1 F-2 H-3 C-1 D-1 E+3 P+1	BFF .. E B.H .. E PPH .. EK .. HCCC . . . G. DDD.G.												
<table border="1"> <tbody> <tr> <td colspan="2">UCS</td></tr> <tr> <td colspan="2">  </td></tr> <tr> <td colspan="2"> <p>Waktu eksekusi: 11.206600000150502ms Tick count: 1267 Node diperiksa: 1330 Banyak pencarian: 11649 Branching factor: 1.6413040161132812 Langkah: 17 / 17</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+3 B+1 F-2 H-3 C-1 D-1 E+3 P+1</p> </td></tr> </tbody> </table>		UCS				<p>Waktu eksekusi: 11.206600000150502ms Tick count: 1267 Node diperiksa: 1330 Banyak pencarian: 11649 Branching factor: 1.6413040161132812 Langkah: 17 / 17</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+3 B+1 F-2 H-3 C-1 D-1 E+3 P+1</p>		<table border="1"> <tbody> <tr> <td colspan="2">GBFS Car Distance</td></tr> <tr> <td colspan="2">  </td></tr> <tr> <td colspan="2"> <p>Waktu eksekusi: 3.0080000003799796ms Tick count: 316 Node diperiksa: 655 Banyak pencarian: 4871 Branching factor: 1.5086729858398438 Langkah: 18 / 18</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+2 G-1 P+1 B+1 F-2 H-3 C-1 D-1 E+3 P+1</p> </td></tr> </tbody> </table>		GBFS Car Distance				<p>Waktu eksekusi: 3.0080000003799796ms Tick count: 316 Node diperiksa: 655 Banyak pencarian: 4871 Branching factor: 1.5086729858398438 Langkah: 18 / 18</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+2 G-1 P+1 B+1 F-2 H-3 C-1 D-1 E+3 P+1</p>	
UCS															
															
<p>Waktu eksekusi: 11.206600000150502ms Tick count: 1267 Node diperiksa: 1330 Banyak pencarian: 11649 Branching factor: 1.6413040161132812 Langkah: 17 / 17</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+3 B+1 F-2 H-3 C-1 D-1 E+3 P+1</p>															
GBFS Car Distance															
															
<p>Waktu eksekusi: 3.0080000003799796ms Tick count: 316 Node diperiksa: 655 Banyak pencarian: 4871 Branching factor: 1.5086729858398438 Langkah: 18 / 18</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+2 G-1 P+1 B+1 F-2 H-3 C-1 D-1 E+3 P+1</p>															
<table border="1"> <tbody> <tr> <td colspan="2">GBFS Car Blocked</td></tr> <tr> <td colspan="2">  </td></tr> <tr> <td colspan="2"> <p>Waktu eksekusi: 6.51589999999851ms Tick count: 432 Node diperiksa: 570 Banyak pencarian: 4060 Branching factor: 1.3993072509765625 Langkah: 21 / 21</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-1 E+3 C-1 G-1 E+3 D+3 G-2 C-2 H+3 F-1 C-1 P+3 B+1 F-2 H-3 C-1 E+3 P+1</p> </td></tr> </tbody> </table>			GBFS Car Blocked				<p>Waktu eksekusi: 6.51589999999851ms Tick count: 432 Node diperiksa: 570 Banyak pencarian: 4060 Branching factor: 1.3993072509765625 Langkah: 21 / 21</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-1 E+3 C-1 G-1 E+3 D+3 G-2 C-2 H+3 F-1 C-1 P+3 B+1 F-2 H-3 C-1 E+3 P+1</p>								
GBFS Car Blocked															
															
<p>Waktu eksekusi: 6.51589999999851ms Tick count: 432 Node diperiksa: 570 Banyak pencarian: 4060 Branching factor: 1.3993072509765625 Langkah: 21 / 21</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-1 E+3 C-1 G-1 E+3 D+3 G-2 C-2 H+3 F-1 C-1 P+3 B+1 F-2 H-3 C-1 E+3 P+1</p>															
<table border="1"> <tbody> <tr> <td colspan="2">A* Car Distance</td></tr> <tr> <td colspan="2">  </td></tr> <tr> <td colspan="2"> <p>Waktu eksekusi: 10.87390000000596ms Tick count: 660 Node diperiksa: 761 Banyak pencarian: 6109 Branching factor: 1.5737380981445312 Langkah: 17 / 17</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+2 B+1 F+3 G-1 P+1 H-3 C-1 D-1 E+3 P+1</p> </td></tr> </tbody> </table>		A* Car Distance				<p>Waktu eksekusi: 10.87390000000596ms Tick count: 660 Node diperiksa: 761 Banyak pencarian: 6109 Branching factor: 1.5737380981445312 Langkah: 17 / 17</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+2 B+1 F+3 G-1 P+1 H-3 C-1 D-1 E+3 P+1</p>		<table border="1"> <tbody> <tr> <td colspan="2">A* Car Blocked</td></tr> <tr> <td colspan="2">  </td></tr> <tr> <td colspan="2"> <p>Waktu eksekusi: 14.568199999630451ms Tick count: 734 Node diperiksa: 857 Banyak pencarian: 6729 Branching factor: 1.5837326049804688 Langkah: 17 / 17</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+2 G-1 P+1 B+1 F-2 H-3 C-1 D-1 E+3 P+1</p> </td></tr> </tbody> </table>		A* Car Blocked				<p>Waktu eksekusi: 14.568199999630451ms Tick count: 734 Node diperiksa: 857 Banyak pencarian: 6729 Branching factor: 1.5837326049804688 Langkah: 17 / 17</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+2 G-1 P+1 B+1 F-2 H-3 C-1 D-1 E+3 P+1</p>	
A* Car Distance															
															
<p>Waktu eksekusi: 10.87390000000596ms Tick count: 660 Node diperiksa: 761 Banyak pencarian: 6109 Branching factor: 1.5737380981445312 Langkah: 17 / 17</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+2 B+1 F+3 G-1 P+1 H-3 C-1 D-1 E+3 P+1</p>															
A* Car Blocked															
															
<p>Waktu eksekusi: 14.568199999630451ms Tick count: 734 Node diperiksa: 857 Banyak pencarian: 6729 Branching factor: 1.5837326049804688 Langkah: 17 / 17</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+2 G-1 P+1 B+1 F-2 H-3 C-1 D-1 E+3 P+1</p>															
<table border="1"> <tbody> <tr> <td colspan="2">A* Car Blocked Recursive</td></tr> <tr> <td colspan="2">  </td></tr> <tr> <td colspan="2"> <p>Waktu eksekusi: 11.70990000013262ms Tick count: 802 Node diperiksa: 955 Banyak pencarian: 7343 Branching factor: 1.5927886962890625 Langkah: 17 / 17</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+2 G-1 P+3 B+1 F-2 H-3 C-1 D-1 E+3 P+1</p> </td></tr> </tbody> </table>			A* Car Blocked Recursive				<p>Waktu eksekusi: 11.70990000013262ms Tick count: 802 Node diperiksa: 955 Banyak pencarian: 7343 Branching factor: 1.5927886962890625 Langkah: 17 / 17</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+2 G-1 P+3 B+1 F-2 H-3 C-1 D-1 E+3 P+1</p>								
A* Car Blocked Recursive															
															
<p>Waktu eksekusi: 11.70990000013262ms Tick count: 802 Node diperiksa: 955 Banyak pencarian: 7343 Branching factor: 1.5927886962890625 Langkah: 17 / 17</p> <p>Step: <math>\emptyset</math> F+2 H-1 C-2 G-3 C+2 D+3 H+3 P+2 G-1 P+3 B+1 F-2 H-3 C-1 D-1 E+3 P+1</p>															

#### 4.13. Jam-13

Nama : Jam-13		6 6
Banyak Step Optimal : 16		12
Step Optimal : $\emptyset E+2 F-1 I+1 P-3 E-2 D+1 J-3 D-2 C-2 H+3 N-1 F+3 G+2 E-1 J-1 P+4$		BBGGH.. ..E.HF .IEPPFK LI.DDF L..JCC LMMJNN
UCS	GBFS Car Distance	GBFS Car Blocked
 <p>Waktu eksekusi: 98.3665000004679ms      Tick count: 8209      Node diperiksa: 9344      Banyak pencarian: 97539      Branching factor: 19273605346679688      Langkah: 16 / 16      Step: <math>\emptyset E+2 F-1 H+1 P-3 E-2 D+1 J-3 D-2 C+1 H+2 N-1 F+3 G+2 E-1 J-1 P+4</math></p>	 <p>Waktu eksekusi: 98.3665000004679ms      Tick count: 6299      Node diperiksa: 8024      Banyak pencarian: 56419      Branching factor: 1507568359375      Langkah: 24 / 24      Step: <math>\emptyset E+2 F-1 H+1 P-3 E-2 D+1 J-3 D-2 C+1 H+2 N-1 F+3 G+2 E-1 J-1 P+4</math></p>	 <p>Waktu eksekusi: 4012820000015199ms      Tick count: 2110      Node diperiksa: 3095      Banyak pencarian: 10097      Branching factor: 15501556396484375      Langkah: 20 / 20      Step: <math>\emptyset E+2 F-1 H+1 P-2 D+1 J-3 N-1 D-2 D-3 C-1 F+3 H+1 G+2 J-1 P-1 E-3 D-1 C-1 H+2 P+6</math></p>
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive
 <p>Waktu eksekusi: 120.8525000003725ms      Tick count: 9754      Node diperiksa: 10614      Banyak pencarian: 91063      Branching factor: 16926345825195312      Langkah: 20 / 20      Step: <math>\emptyset E+2 F-1 H+1 D-1 J-3 N-1 P-1 H+1 G+1 P-2 E-3 P-1 J-2 D-2 C-2 F-3 G+1 J-1 H+2 P+3</math></p>	 <p>Waktu eksekusi: 113.25719999987632ms      Tick count: 5722      Node diperiksa: 7312      Banyak pencarian: 51171      Branching factor: 174188232421875      Langkah: 18 / 18      Step: <math>\emptyset E+2 F-1 H+1 P-3 D+1 J-3 D-1 C-1 N-1 F+3 H+1 G+2 J-1 E-3 D-1 C-1 H+2 P+4</math></p>	 <p>Waktu eksekusi: 178.35939999949187ms      Tick count: 5937      Node diperiksa: 7311      Banyak pencarian: 53166      Branching factor: 1.8089141845703125      Langkah: 17 / 17      Step: <math>\emptyset E+2 H+1 P-3 H+1 G+1 E-3 F-1 D-1 J-3 N-1 D-2 C-2 F-3 H+2 G+1 J-1 P+6</math></p>

#### 4.14. Jam-14

Nama : Jam-14		6 6
Banyak Step Optimal : 17		11
Step Optimal : $\emptyset E+1 I+1 P-2 G+1$ $B+3 G-1 P+2 E-3 I-3$ $P-2 M-2 L-2 C-4 L+1$ $D+1 J+1 P+4$		BBG ... .. G.HH EIPPDJK EIMMDJ .. L.CC FFL ...
UCS	GBFS Car Distance	GBFS Car Blocked
 <p>Waktu eksekusi: 114.8128000004217ms Tick count: 9656 Node diperiksa: 13452 Banyak pencarian: 11343 Branching factor: 1.8973312377929688 Langkah: 17 / 17 Step: <math>\emptyset E I G B H C M L A F P D J</math> <math>E+3 I+2 P+2 M+2 L+2 C+4 L+1 D+1 J+1</math> <math>P+4</math></p>	 <p>Waktu eksekusi: 72.6324000004679ms Tick count: 6246 Node diperiksa: 9954 Banyak pencarian: 95739 Branching factor: 1.7489471435546875 Langkah: 19 / 19 Step: <math>\emptyset E I G B H C M L A F P D J</math> <math>E+3 H+2 M+2 P+2 G+2 H+4 G-2 P+2</math> <math>D-2 P+1 I-2 P+4</math></p>	 <p>Waktu eksekusi: 118.12199999950826ms Tick count: 4493 Node diperiksa: 8333 Banyak pencarian: 49779 Branching factor: 1.6850433349609375 Langkah: 19 / 19 Step: <math>\emptyset E I G B H C M L A F P D J</math> <math>M+2 E+3 H+3 P+2 M+2 G+2 H+1 D-2</math> <math>H-2 G-2 P+4</math></p>
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive
 <p>Waktu eksekusi: 107.20429999940097ms Tick count: 7939 Node diperiksa: 10657 Banyak pencarian: 91767 Branching factor: 1.803863525390625 Langkah: 18 / 18 Step: <math>\emptyset E I G B H C M L A F P D J</math> <math>E+3 H+2 M+2 P+2 G+2 H+4 G-2 P+2</math> <math>D-2 J-2 P+4</math></p>	 <p>Waktu eksekusi: 99.95299999974668ms Tick count: 5618 Node diperiksa: 8507 Banyak pencarian: 62077 Branching factor: 1.7621841430664062 Langkah: 18 / 18 Step: <math>\emptyset E I G B H C M L A F P D J</math> <math>M+2 E+3 H+3 P+2 M+2 G+2 H-3 G-2</math> <math>D-2 P+4</math></p>	 <p>Waktu eksekusi: 109.16589999943972ms Tick count: 5846 Node diperiksa: 8680 Banyak pencarian: 64544 Branching factor: 1.7663040161132812 Langkah: 18 / 18 Step: <math>\emptyset E I G B H C M L A F P D J</math> <math>M+2 E+3 H+3 P+2 M+2 G+2 H-3 G-2</math> <math>D-2 P+4</math></p>

#### 4.15. Jam-15

<table border="1"> <tbody> <tr> <td>Nama : Jam-15</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal : 23</td><td>13</td></tr> <tr> <td>Step Optimal :</td><td>.BBGG. HHEEFL CMPPFLK CMIDFL CMIDJJ .NNOO.</td></tr> </tbody> </table>	Nama : Jam-15	6 6	Banyak Step Optimal : 23	13	Step Optimal :	.BBGG. HHEEFL CMPPFLK CMIDFL CMIDJJ .NNOO.	
Nama : Jam-15	6 6						
Banyak Step Optimal : 23	13						
Step Optimal :	.BBGG. HHEEFL CMPPFLK CMIDFL CMIDJJ .NNOO.						
UCS	GBFS Car Distance						
 <p>Waktu eksekusi: 2.60450000036507784ms Tick count: 524 Node diperiksa: 530 Banyak pencarian: 3187 Branching factor: 1.3376541137695312 Langkah: 23 / 23 Step: ① B-1 G+1 C+1 O+1 N+1 M+1 P-2 ② D-1 J-2 F+1 L+1 E+2 I-2 D-2 P+2 C-1 M-1 N-2 O-2 F+1 L+1 P+2 ③ C-1 M-1 N-2 O-2 F+1 L+1 P+2</p>	 <p>Waktu eksekusi: 2.3774999994784594ms Tick count: 446 Node diperiksa: 521 Banyak pencarian: 2753 Branching factor: 1.2656402587890625 Langkah: 27 / 27 Step: ① B-1 G+1 C+1 O+1 N+1 M+1 P-2 H-1 ② D-1 J-2 F+1 L+1 E+2 P+1 G+1 L+1 E+1 ③ D-2 P+1 C-1 M-1 N-2 O-2 F+1 P+1 L+1 P+2</p>						
A* Car Distance	A* Car Blocked						
 <p>Waktu eksekusi: 6.9738999999633431ms Tick count: 524 Node diperiksa: 530 Banyak pencarian: 3187 Branching factor: 1.3376541137695312 Langkah: 23 / 23 Step: ① B-1 G+1 C+1 O+1 N+1 M+1 P-2 ② D-1 J-2 F+1 L+1 E+2 D-2 P+2 ③ C-1 M-1 N-2 O-2 F+1 L+1 P+2</p>	 <p>Waktu eksekusi: 2.4930000007152557ms Tick count: 522 Node diperiksa: 527 Banyak pencarian: 3178 Branching factor: 1.33746337890625 Langkah: 23 / 23 Step: ① B-1 G+1 C+1 O+1 N+1 M+1 P-2 ② D-1 J-2 F+1 L+1 E+2 D-2 P+2 ③ C-1 M-1 N-2 O-2 F+1 L+1 P+2</p>						
A* Car Blocked Recursive							
 <p>Waktu eksekusi: 9.339599999959159ms Tick count: 520 Node diperiksa: 525 Banyak pencarian: 3170 Branching factor: 1.3372955322265625 Langkah: 23 / 23 Step: ① B-1 G+1 C+1 O+1 N+1 M+1 P-2 ② D-1 J-2 F+1 L+1 E+2 D-2 P+2 ③ C-1 M-1 N-2 O-2 F+1 L+1 P+2</p>							

#### 4.16. Jam-16

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-16</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>21</td><td>10</td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> E+2 I-2 D+1 L+1            P-3 H+1 G+1 L-3 J+2            D+1 C-2 H+3 C+2 D-1            J-2 L+3 P+3 L-3 C-1            F+3 P+1         </td><td>           BBGGHF            E.IIHF            EDLPPFK            .DLCCC            .. L ...            JJ....         </td></tr> </tbody> </table>	Nama	:	Jam-16	6 6	Banyak Step Optimal	:	21	10	Step Optimal	:	$\emptyset$ E+2 I-2 D+1 L+1 P-3 H+1 G+1 L-3 J+2 D+1 C-2 H+3 C+2 D-1 J-2 L+3 P+3 L-3 C-1 F+3 P+1	BBGGHF E.IIHF EDLPPFK .DLCCC .. L ... JJ....			
Nama	:	Jam-16	6 6												
Banyak Step Optimal	:	21	10												
Step Optimal	:	$\emptyset$ E+2 I-2 D+1 L+1 P-3 H+1 G+1 L-3 J+2 D+1 C-2 H+3 C+2 D-1 J-2 L+3 P+3 L-3 C-1 F+3 P+1	BBGGHF E.IIHF EDLPPFK .DLCCC .. L ... JJ....												
<p style="text-align: center;"><b>UCS</b></p>  <p> <b>Waktu eksekusi:</b> 17.394400000572205ms  <b>Tick count:</b> 2518  <b>Node diperiksa:</b> 2898  <b>Banyak pencarian:</b> 23220  <b>Branching factor:</b> 153497314453125  <b>Langkah:</b> 21 / 21  <b>Step:</b> <math>\emptyset</math> E+2 I-2 D+1 L+1 P+3 H+1            L-3 D-2 C-2 H+3 C-2 D-1 J-2 L-5            P-3 L-3 C-1 F+3 P+1         </p>	<p style="text-align: center;"><b>GBFS Car Distance</b></p>  <p> <b>Waktu eksekusi:</b> 12.385999999807775ms  <b>Tick count:</b> 2056  <b>Node diperiksa:</b> 2572  <b>Banyak pencarian:</b> 1967  <b>Branching factor:</b> 148992999921875  <b>Langkah:</b> 22 / 22  <b>Step:</b> <math>\emptyset</math> E+2 I-2 J-3 D+2 L+1 P-3 H+1            G+1 P-2 L-3 C-2 J-1 H+3 C-2 D-1 J-2            L-3 P-3 L-3 C-1 F+3 P+1         </p>	<p style="text-align: center;"><b>GBFS Car Blocked</b></p>  <p> <b>Waktu eksekusi:</b> 11.504499999806285ms  <b>Tick count:</b> 1530  <b>Node diperiksa:</b> 1944  <b>Banyak pencarian:</b> 14547  <b>Branching factor:</b> 13950729370117188  <b>Langkah:</b> 25 / 25  <b>Step:</b> <math>\emptyset</math> E+2 I-2 D+1 L+1 P-3 H+1            L-3 C-1 F+3 J-2 D-1 C-1 H+3 C-1 D-1            J-2 E-3 C-1 L-3 P-3 L-3 C-1 F+3 D-1         </p>													
<p style="text-align: center;"><b>A* Car Distance</b></p>  <p> <b>Waktu eksekusi:</b> 22.7441999996081ms  <b>Tick count:</b> 2716  <b>Node diperiksa:</b> 2646  <b>Banyak pencarian:</b> 20547  <b>Branching factor:</b> 14933853149414062  <b>Langkah:</b> 22 / 22  <b>Step:</b> <math>\emptyset</math> E+2 I-2 J-3 D+2 L+1 P-1 H+1            G+1 P-2 L-3 C-2 J-1 H+3 C-2 D-1 J-2            L-3 P-3 L-3 C-1 F+3 P+1         </p>	<p style="text-align: center;"><b>A* Car Blocked</b></p>  <p> <b>Waktu eksekusi:</b> 42.76310000009835ms  <b>Tick count:</b> 2095  <b>Node diperiksa:</b> 2439  <b>Banyak pencarian:</b> 19405  <b>Branching factor:</b> 1520622534179688  <b>Langkah:</b> 21 / 21  <b>Step:</b> <math>\emptyset</math> E+2 I-2 J-3 D+2 L+1 P-3 H+1            G+1 L-3 C-2 J-1 H+3 C-2 D-1 J-2 L-3            P-3 L-3 C-1 F+3 P+1         </p>	<p style="text-align: center;"><b>A* Car Blocked Recursive</b></p>  <p> <b>Waktu eksekusi:</b> 48.815200000070035ms  <b>Tick count:</b> 1869  <b>Node diperiksa:</b> 2109  <b>Banyak pencarian:</b> 17509  <b>Branching factor:</b> 1515280151367188  <b>Langkah:</b> 21 / 21  <b>Step:</b> <math>\emptyset</math> E+2 I-2 J-3 D+2 L+1 P-3 H+1            G+1 L-3 C-2 J-1 H+3 C-2 D-1 J-2 L-3            P-3 L-3 C-1 F+3 P+1         </p>													

#### 4.17. Jam-17

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-17</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>24</td><td>11</td></tr> <tr> <td>Step Optimal</td><td>:</td><td><math>\emptyset \text{ F+2 G-1 L-2 D-2}</math> <math>\text{J-2 C+3 M+3 E+2 P+1}</math> <math>\text{I+1 B+4 G-1 P-1 I-1}</math> <math>\text{E-4 P+1 I+1 B-2 C-3}</math> <math>\text{D+1 J+1 M-3 L+2 P+3}</math></td><td>BFFF.. B.GGHH PPE ... K IIEL.. CCCLDJ MMMLDJ</td></tr> </tbody> </table>	Nama	:	Jam-17	6 6	Banyak Step Optimal	:	24	11	Step Optimal	:	$\emptyset \text{ F+2 G-1 L-2 D-2}$ $\text{J-2 C+3 M+3 E+2 P+1}$ $\text{I+1 B+4 G-1 P-1 I-1}$ $\text{E-4 P+1 I+1 B-2 C-3}$ $\text{D+1 J+1 M-3 L+2 P+3}$	BFFF.. B.GGHH PPE ... K IIEL.. CCCLDJ MMMLDJ			
Nama	:	Jam-17	6 6												
Banyak Step Optimal	:	24	11												
Step Optimal	:	$\emptyset \text{ F+2 G-1 L-2 D-2}$ $\text{J-2 C+3 M+3 E+2 P+1}$ $\text{I+1 B+4 G-1 P-1 I-1}$ $\text{E-4 P+1 I+1 B-2 C-3}$ $\text{D+1 J+1 M-3 L+2 P+3}$	BFFF.. B.GGHH PPE ... K IIEL.. CCCLDJ MMMLDJ												
UCS	GBFS Car Distance	GBFS Car Blocked													
<p>Waktu eksekusi: 9.926200000569224ms Tick count: 2107 Node diperiksa: 2152 Banyak pencarian: 19664 Branching factor: 1.4366607666015625 Langkah: 24 / 24 Step: <math>\emptyset \text{ F+2 G-1 L-2 D-2 C+3 M+3}</math> <math>\text{E+2 P+1 H-1 B+4 G-1 P-1 I-1 E-4 P+1}</math> <math>\text{H-2 B-2 C-3 D+1 J+1 M-3 L+2 P+3}</math> <math>\text{J+1 P+3}</math></p>	<p>Waktu eksekusi: 9.635999999940395ms Tick count: 993 Node diperiksa: 1246 Banyak pencarian: 9078 Branching factor: 1.331233520507812 Langkah: 27 / 27 Step: <math>\emptyset \text{ G-1 L-2 D-2 J-2 C+3 E+1 P+2}</math> <math>\text{F+2 M+3 E+1 H-1 B+4 G-1 P-1 I-1 E-4}</math> <math>\text{P+1 H-2 B-2 C-3 M-3 L+2 P+1 D+1 P+1}</math> <math>\text{J+1 P+3}</math></p>	<p>Waktu eksekusi: 6.762299999594688ms Tick count: 864 Node diperiksa: 1098 Banyak pencarian: 7951 Branching factor: 1.3237762451171875 Langkah: 27 / 27 Step: <math>\emptyset \text{ G-1 L-2 D-2 J-2 C+3 E+1 F+2}</math> <math>\text{P+1 H-2 B-2 C-3 M-3 L+2 P+1 D+1 P+1}</math> <math>\text{L+1 P+3}</math></p>													
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive													
<p>Waktu eksekusi: 11.697900000028312ms Tick count: 2069 Node diperiksa: 2155 Banyak pencarian: 19406 Branching factor: 1.4130325317382812 Langkah: 25 / 25 Step: <math>\emptyset \text{ G-1 L-2 D-2 J-2 C+3 M+3 E+2}</math> <math>\text{P+1 F+2 H-1 B+4 G-1 I-1 P-1 E-4 P+1}</math> <math>\text{H-2 B-2 C-3 D+1 M-3 L+2 P+2 J+1 P+3}</math></p>	<p>Waktu eksekusi: 13.386600000783801ms Tick count: 1783 Node diperiksa: 1971 Banyak pencarian: 16898 Branching factor: 1.404388427734375 Langkah: 25 / 25 Step: <math>\emptyset \text{ F+2 G-1 L-2 D-2 J-2 C+3 M+3}</math> <math>\text{E+2 P+1 H-1 B+4 G-1 P-1 I-1 E-4 C-2}</math> <math>\text{D+1 J+1 P+1 H-2 B-2 C-1 M-3 L+2 P+3}</math></p>	<p>Waktu eksekusi: 50.556200000457466ms Tick count: 1973 Node diperiksa: 2041 Banyak pencarian: 18656 Branching factor: 1.4105682373046875 Langkah: 25 / 25 Step: <math>\emptyset \text{ F+2 G-1 L-2 D-2 J-2 C+3 M+3}</math> <math>\text{E+2 P+1 H-1 B+4 G-1 P-1 I-1 E-4 C-2}</math> <math>\text{D+1 J+1 P+1 H-2 B-2 C-1 M-3 L+2 P+3}</math></p>													

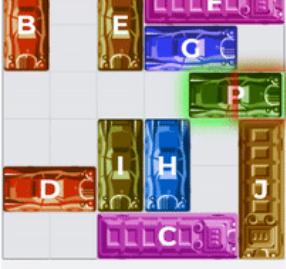
#### 4.18. Jam-18

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-18</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>25</td><td>8</td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> C+2 E+3 D+3 I+1            P-1 G+4 B+1 H+1 P+1            I-3 C-3 F+2 B+3 H+3            F-2 C+3 I+3 P-1 G-4            P+1 I-3 C-3 D-3 F+3            P+3         </td><td>           BBGF ..            HHGF ..            IPPF .. K            ICCC ..            IEE ...            DDD ...         </td></tr> </tbody> </table>	Nama	:	Jam-18	6 6	Banyak Step Optimal	:	25	8	Step Optimal	:	$\emptyset$ C+2 E+3 D+3 I+1 P-1 G+4 B+1 H+1 P+1 I-3 C-3 F+2 B+3 H+3 F-2 C+3 I+3 P-1 G-4 P+1 I-3 C-3 D-3 F+3 P+3	BBGF .. HHGF .. IPPF .. K ICCC .. IEE ... DDD ...			
Nama	:	Jam-18	6 6												
Banyak Step Optimal	:	25	8												
Step Optimal	:	$\emptyset$ C+2 E+3 D+3 I+1 P-1 G+4 B+1 H+1 P+1 I-3 C-3 F+2 B+3 H+3 F-2 C+3 I+3 P-1 G-4 P+1 I-3 C-3 D-3 F+3 P+3	BBGF .. HHGF .. IPPF .. K ICCC .. IEE ... DDD ...												
UCS	GBFS Car Distance	GBFS Car Blocked													
 <p>Waktu eksekusi: 5.207899999804795ms            Tick count: 1586            Node diperiksa: 1634            Banyak pencarian: 13905            Branching factor: 1.39227294921875            Langkah: 25 / 25            Step: <math>\emptyset</math> C+2 E+3 D+3 I+1 P-1 G+4 B+1            H+1 P+1 I-3 C-3 F+2 B+3 H+3 F-2 C+3            I-3 P-1 G-4 P+1 I-3 C-3 D-3 F+3 P+3</p>	 <p>Waktu eksekusi: 4.5848999999201298ms            Tick count: 1102            Node diperiksa: 1288            Banyak pencarian: 9485            Branching factor: 1.3182449340820312            Langkah: 28 / 28            Step: <math>\emptyset</math> C+2 E+2 D+1 H+1 P-1 G+3 P+1            B+1 H+1 I-3 E+1 D+2 C+1 C-3 F+2 B+3            H+3 F-2 C+3 I+3 P-1 G-4 P+1 I-3 C-3            D-3 F+3 P+3</p>	 <p>Waktu eksekusi: 13.497999999672174ms            Tick count: 994            Node diperiksa: 1428            Banyak pencarian: 8375            Branching factor: 1.3610382080078125            Langkah: 25 / 25            Step: <math>\emptyset</math> C+2 E+3 D+3 I+1 P-1 G+4 B+1            H+1 P+1 I-3 C-3 F+2 B+3 H+3 F-2 C+3            I-3 D-3 G-4 P+1 I-3 C-3 D-3 F+3 P+3</p>													
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive													
 <p>Waktu eksekusi: 10.472900000400841ms            Tick count: 1545            Node diperiksa: 1590            Banyak pencarian: 13575            Branching factor: 1.390777587890625            Langkah: 25 / 25            Step: <math>\emptyset</math> C+2 E+3 D+3 I+1 P-1 G+4 B+1            B+1 H+1 I-3 C-3 F+2 B+3 H+3 F-2 C+3            I-3 P-1 G-4 P+1 I-3 C-3 D-3 F+3 P+3</p>	 <p>Waktu eksekusi: 29.884499999694526ms            Tick count: 1548            Node diperiksa: 1595            Banyak pencarian: 13593            Branching factor: 1.3908615112304688            Langkah: 25 / 25            Step: <math>\emptyset</math> C+2 E+3 D+3 I+1 P-1 G+4 B+1            H+1 P+1 I-3 C-3 F+2 B+3 H+3 F-2 C+3            I-3 P-1 G-4 P+1 I-3 C-3 D-3 F+3 P+3</p>	 <p>Waktu eksekusi: 46.510999999940395ms            Tick count: 1287            Node diperiksa: 1520            Banyak pencarian: 10695            Branching factor: 1.377197265625            Langkah: 25 / 25            Step: <math>\emptyset</math> C+2 E+3 D+3 I+1 P-1 G+4 B+1            H+1 P+1 I-3 C-3 F+2 B+3 H+3 F-2 C+3            I-3 P-1 G-4 P+1 I-3 C-3 D-3 F+3 P+3</p>													

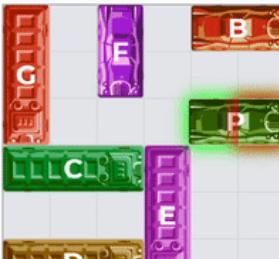
#### 4.19. Jam-19

Nama : Jam-19		6 6
Banyak Step Optimal : 22		7
Step Optimal :	$\emptyset$ D-2 P-2 B+1 F-1 G-1 H-2 C-1 E+2 B+3 P+2 H+2 D+4 F-2 P-2 H-2 B-4 P+2 H+2 D-2 E-2 C+1 P+2	.. BFF. .. B.G. .DPPG.K .DHHC. .EEE.C. .....
UCS	GBFS Car Distance	GBFS Car Blocked
<p>Waktu eksekusi: 16.455699999816716ms Tick count: 484 Node diperiksa: 527 Banyak pencarian: 3572 Branching factor: 1.3662185668945312 Langkah: 22 / 22 Step: <math>\emptyset</math> D-2 P-2 B+1 F-1 G-1 C-1 E-2 B+3 P+2 H+2 D+4 F-2 P-2 H-2 B-4 P+2 H+2 D-2 E-2 C+1 P+2</p>	<p>Waktu eksekusi: 1.720100000500679ms Tick count: 431 Node diperiksa: 477 Banyak pencarian: 3250 Branching factor: 1.338958740234375 Langkah: 23 / 23 Step: <math>\emptyset</math> D-2 H-2 P-2 B+1 F-1 G-1 C-1 E-2 B+3 P+2 H+2 D+4 F-2 P-2 H-2 B-4 P+2 H+2 D-2 E-2 C+1 P+2</p>	<p>Waktu eksekusi: 14.769399999640882ms Tick count: 434 Node diperiksa: 475 Banyak pencarian: 3258 Branching factor: 1.359710693359375 Langkah: 22 / 22 Step: <math>\emptyset</math> D-2 P-2 B+1 F-1 G-1 H-2 C-1 E-2 B+3 P+2 H+2 D+4 F-2 P-2 H-2 B-4 P+2 H+2 D-2 E-2 C+1 P+2</p>
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive
<p>Waktu eksekusi: 1.6309000002220273ms Tick count: 480 Node diperiksa: 527 Banyak pencarian: 3554 Branching factor: 1.3658599853515625 Langkah: 22 / 22 Step: <math>\emptyset</math> D-2 H-2 P-2 B+1 F-1 G-1 C-1 E-2 B+3 P+2 H+2 D+4 F-2 H-2 P-2 B-4 P+2 H+2 D-2 E-2 C+1 P+2</p>	<p>Waktu eksekusi: 7.223900000564754ms Tick count: 483 Node diperiksa: 527 Banyak pencarian: 3568 Branching factor: 1.3661422729492188 Langkah: 22 / 22 Step: <math>\emptyset</math> D-2 P-2 H-2 B+1 F-1 G-1 C-1 E-2 B+3 P+2 H+2 D+4 F-2 P-2 H-2 B-4 P+2 H+2 D-2 E-2 C+1 P+2</p>	<p>Waktu eksekusi: 13.499100000597537ms Tick count: 482 Node diperiksa: 526 Banyak pencarian: 3562 Branching factor: 1.36660202026367188 Langkah: 22 / 22 Step: <math>\emptyset</math> D-2 P-2 H-2 B+1 F-1 G-1 C-1 E-2 B+3 P+2 H+2 D+4 F-2 P-2 H-2 B-4 P+2 H+2 D-2 E-2 C+1 P+2</p>

#### 4.20. Jam-20

Nama : Jam-20		6 6
Banyak Step Optimal : 10		9
Step Optimal : $\emptyset H+1 G+2 E-2 I-2 D-3 I+1 H+1 C-1 J+1 P+4$		B .. FFF BGGH .. PPEH.JK .. E .. J .. IDDJ .. ICCC
UCS	GBFS Car Distance	GBFS Car Blocked
 <p>         Waktu eksekusi: 36.40940000023693ms          Tick count: 1660          Node diperiksa: 2376          Banyak pencarian: 14939          Branching factor: 2.4834518432617188          Langkah: 10 / 10          Step: <math>\emptyset H+1 G+2 E-2 I-2 D-3 H+1 H+1</math>  <math>\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow</math> </p>	 <p>         Waktu eksekusi: 12.500700000673532ms          Tick count: 1691          Node diperiksa: 2908          Banyak pencarian: 15382          Branching factor: 2.1173152923583984          Langkah: 12 / 12          Step: <math>\emptyset H+1 G+2 E-2 I-2 D-3 H+1 H+1</math>  <math>\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow</math> </p>	 <p>         Waktu eksekusi: 5.352699999697506ms          Tick count: 1191          Node diperiksa: 1893          Banyak pencarian: 10622          Branching factor: 2.0477066040039062          Langkah: 12 / 12          Step: <math>\emptyset H+1 G+2 E-2 I-2 C-1 H+1 D-2</math>  <math>\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow</math> </p>
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive
 <p>         Waktu eksekusi: 9.816899999976158ms          Tick count: 1659          Node diperiksa: 2444          Banyak pencarian: 14940          Branching factor: 2.27300077163086          Langkah: 11 / 11          Step: <math>\emptyset H+1 G+2 E-2 I-2 D-3 H+1 H+1</math>  <math>\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow</math> </p>	 <p>         Waktu eksekusi: 32.0832000002265ms          Tick count: 1299          Node diperiksa: 1974          Banyak pencarian: 11715          Branching factor: 2.4194507598876953          Langkah: 10 / 10          Step: <math>\emptyset H+1 G+2 E-2 I-2 D-3 H+1 H+1</math>  <math>\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow</math> </p>	 <p>         Waktu eksekusi: 44.6211000001058ms          Tick count: 779          Node diperiksa: 1283          Banyak pencarian: 6770          Branching factor: 2.2803192138671875          Langkah: 10 / 10          Step: <math>\emptyset H+1 G+2 E-2 I-2 C-1 H+1 D-3</math>  <math>\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow</math> </p>

#### 4.21. Jam-21

Nama : Jam-21		6 6
Banyak Step Optimal : 21		6
Step Optimal : $\emptyset$ G+2 P-1 C+2 F+4 B+1 P+1 G-3 C-3 E+1 B+3 E-1 C+3 G+3 P-1 F-4 P+1 G-3 C-3 D-3 E+3 P+3		BBFE.. G.FE.. GPPE.. K GCCC.. ..... ... DDD
<p style="text-align: center;">UCS</p>  <p>Waktu eksekusi: 0.93809999999120831ms Tick count: 258 Node diperiksa: 263 Banyak pencarian: 1683 Branching factor: 1.3335113525390625 Langkah: 21 / 21 Step: <math>\emptyset</math> G+2 P-1 C+2 F+4 B+1 P+1 G+3 C-3 E-1 B+3 E-1 C+3 G+3 P-1 F-4 P+1 G-3 C-3 D-3 E+3 P+3</p>		<p style="text-align: center;">GBFS Car Distance</p>  <p>Waktu eksekusi: 1.15730000000789762ms Tick count: 232 Node diperiksa: 257 Banyak pencarian: 1535 Branching factor: 1.30696866889648438 Langkah: 22 / 22 Step: <math>\emptyset</math> G+2 P-1 C+2 F+4 B+1 P+1 G+3 F-1 C-3 E-1 B-3 E-1 C+3 G+3 P-1 F-4 P+1 G-3 C-3 D-3 E+3 P+3</p>
 <p>Waktu eksekusi: 0.832400000654161ms Tick count: 208 Node diperiksa: 253 Banyak pencarian: 1387 Branching factor: 1.319244384765625 Langkah: 21 / 21 Step: <math>\emptyset</math> G+2 P-1 C+2 F+4 B+1 P+1 G+3 C-3 E-1 B+3 E-1 C+3 G+3 P-1 F-4 P+1 G-3 C-3 D-3 E+3 P+3</p>		<p style="text-align: center;">GBFS Car Blocked</p>  <p>Waktu eksekusi: 0.832400000654161ms Tick count: 208 Node diperiksa: 253 Banyak pencarian: 1387 Branching factor: 1.319244384765625 Langkah: 21 / 21 Step: <math>\emptyset</math> G+2 P-1 C+2 F+4 B+1 P+1 G+3 C-3 E-1 B+3 E-1 C+3 G+3 P-1 F-4 P+1 G-3 C-3 D-3 E+3 P+3</p>
<p style="text-align: center;">A* Car Distance</p>  <p>Waktu eksekusi: 0.9602999994531274ms Tick count: 250 Node diperiksa: 261 Banyak pencarian: 1641 Branching factor: 1.3316497802734375 Langkah: 21 / 21 Step: <math>\emptyset</math> G+2 P-1 C+2 F+4 P+1 B+1 G+3 C-3 E-1 B+3 E-1 C+3 G+3 P-1 F-4 P+1 G-3 C-3 D-3 E+3 P+3</p>		<p style="text-align: center;">A* Car Blocked</p>  <p>Waktu eksekusi: 2.4153000004589558ms Tick count: 252 Node diperiksa: 263 Banyak pencarian: 1651 Branching factor: 1.33209228515625 Langkah: 21 / 21 Step: <math>\emptyset</math> G+2 P-1 C+2 F+4 B+1 P+1 G+3 C-3 E-1 B+3 E-1 C+3 G+3 P-1 F-4 P+1 G-3 C-3 D-3 E+3 P+3</p>
 <p>Waktu eksekusi: 2.7028999999165535ms Tick count: 243 Node diperiksa: 255 Banyak pencarian: 1602 Branching factor: 1.3298721313476562 Langkah: 21 / 21 Step: <math>\emptyset</math> G+2 P-1 C+2 F+4 B+1 P+1 G+3 C-3 E-1 B+3 E-1 C+3 G+3 P-1 F-4 P+1 G-3 C-3 D-3 E+3 P+3</p>		<p style="text-align: center;">A* Car Blocked Recursive</p>  <p>Waktu eksekusi: 2.7028999999165535ms Tick count: 243 Node diperiksa: 255 Banyak pencarian: 1602 Branching factor: 1.3298721313476562 Langkah: 21 / 21 Step: <math>\emptyset</math> G+2 P-1 C+2 F+4 B+1 P+1 G+3 C-3 E-1 B+3 E-1 C+3 G+3 P-1 F-4 P+1 G-3 C-3 D-3 E+3 P+3</p>

## 4.22. Jam-22

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-22</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>26</td><td>11</td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> G-1 P-1 D-1 J+1            B+3 P+1 G+1 F-3 L-1            I-1 M-2 J+1 C+2 E+1            B+1 I-2 L+2 H-3 L-2            M-2 I+3 E-1 B-1 C-3            L+3 P+3         </td><td>           .. BFFF            G.BLHH            GPPL .. K            .E.LII            DEJJ.M            DCCC.M         </td></tr> </tbody> </table>	Nama	:	Jam-22	6 6	Banyak Step Optimal	:	26	11	Step Optimal	:	$\emptyset$ G-1 P-1 D-1 J+1 B+3 P+1 G+1 F-3 L-1 I-1 M-2 J+1 C+2 E+1 B+1 I-2 L+2 H-3 L-2 M-2 I+3 E-1 B-1 C-3 L+3 P+3	.. BFFF G.BLHH GPPL .. K .E.LII DEJJ.M DCCC.M			
Nama	:	Jam-22	6 6												
Banyak Step Optimal	:	26	11												
Step Optimal	:	$\emptyset$ G-1 P-1 D-1 J+1 B+3 P+1 G+1 F-3 L-1 I-1 M-2 J+1 C+2 E+1 B+1 I-2 L+2 H-3 L-2 M-2 I+3 E-1 B-1 C-3 L+3 P+3	.. BFFF G.BLHH GPPL .. K .E.LII DEJJ.M DCCC.M												
UCS	GBFS Car Distance	GBFS Car Blocked													
 <p>           Waktu eksekusi: 73.51949999994701ms            Tick count: 3602            Node diperiksa: 4139            Banyak pencarian: 32777            Branching factor: 1.4236373901367188            Langkah: 26 / 26            Step: <math>\emptyset</math> G-1 P-1 D-1 J+1 B+3 P+1 G+1 F-3 L-1 I-1 M-2 J+1 C+2 E+1 B-1 I-2 L+2 H-3 L-2 M-2 I-3 E-1 B-1 C-3 L+3 P+3         </p>	 <p>           Waktu eksekusi: 20.057599999941885ms            Tick count: 2175            Node diperiksa: 3097            Banyak pencarian: 19962            Branching factor: 1.358001708984375            Langkah: 28 / 28            Step: <math>\emptyset</math> G-1 J+1 P-1 B+3 P+1 G+1 F-3 L-1 I-1 D-1 M-2 J+1 C+2 E+1 B-1 I-2 L+2 H-3 L-2 M-2 I-3 E-1 B-1 C-3 L+3 P+3         </p>	 <p>           Waktu eksekusi: 41.793899999931455ms            Tick count: 1427            Node diperiksa: 2697            Banyak pencarian: 11497            Branching factor: 1.3284683227539062            Langkah: 28 / 28            Step: <math>\emptyset</math> G-1 P-1 J-1 B+3 P+1 G+1 F-3 L-1 I-1 M-1 C-2 E-1 B-1 I-2 M-1 J-1 L+2 H-3 M-2 L-2 I-3 D-1 E-1 B-1 C-3 L+3 P+3         </p>													
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive													
 <p>           Waktu eksekusi: 42.3179999999702ms            Tick count: 3352            Node diperiksa: 3886            Banyak pencarian: 30277            Branching factor: 1.48975830078125            Langkah: 26 / 26            Step: <math>\emptyset</math> G-1 D-1 J+1 P-1 B+3 P+1 G+1 F-3 L-1 I-1 M-2 J+1 C+2 E+1 B-1 I-2 L+2 H-3 L-2 M-2 I-3 E-1 B-1 C-3 L+3 P+3         </p>	 <p>           Waktu eksekusi: 62.582300000824034ms            Tick count: 2797            Node diperiksa: 3501            Banyak pencarian: 24624            Branching factor: 1.38712301079101562            Langkah: 27 / 27            Step: <math>\emptyset</math> G-1 D-1 J-1 B+3 P+1 G+1 F-3 L-1 I-1 M-1 C-2 E-1 B-1 I-2 M-1 J-1 L-2 H-3 M-2 L-2 I-3 E-1 B-1 C-3 L+3 P+3         </p>	 <p>           Waktu eksekusi: 43.440099999308586ms            Tick count: 2033            Node diperiksa: 2582            Banyak pencarian: 17310            Branching factor: 1.3677951293945312            Langkah: 27 / 27            Step: <math>\emptyset</math> D-1 C-1 G-1 P-1 J-1 B+3 P+1 G+1 F-3 L-1 I-1 M-1 C-2 E-1 B-1 I-2 M-1 J-1 L-2 H-3 M-2 L-2 I-3 E-1 B-1 C-3 L+3 P+3         </p>													

#### 4.23. Jam-23

<p>Nama : Jam-23</p> <p>Banyak Step Optimal : 29</p> <p>Step Optimal :</p> <pre><math>\emptyset \text{ C-2 E+1 I-1 J+1}</math> <math>\text{F+1 B-1 P-3 B+1 F-1}</math> <math>\text{J-1 I+1 E-1 C+3 H+1}</math> <math>\text{B+1 G-3 B-1 E-2 I-4}</math> <math>\text{J+1 F+1 B-1 H-2 D-4}</math> <math>\text{E+2 H+1 C-1 J+2 P+4}</math></pre>	<p>6 6</p> <p>9</p> <p>.. FFFF</p> <p>.. BGJJ</p> <p>.. BPPJK</p> <p>.. HEII</p> <p>.. HEDD</p> <p>.. CCC.</p>
<p>UCS</p> <p>Waktu eksekusi: 5814419999998063ms Tick count: 2306 Node diperiksa: 2747 Banyak pencarian: 19552 Branching factor: 1.3411331176757812 Langkah: 29 / 29 Step: <math>\emptyset \text{ C-2 E+1 F-1 J+1 P-3 B-1 G-3 B-1}</math> <math>\text{E-2 I-4 J+1 F+1 B-1 H-2 D-4 E+2 H+1}</math> <math>\text{C-1 J+2 P+4}</math></p>	<p>GBFS Car Distance</p> <p>Waktu eksekusi: 9.0514000000229478ms Tick count: 1515 Node diperiksa: 2216 Banyak pencarian: 1929 Branching factor: 1.2400283813476562 Langkah: 36 / 36 Step: <math>\emptyset \text{ C-1 H+1 B-1 G-3 B-1 H-1 C-3}</math> <math>\text{E-1 I-1 J+1 F+1 B-1 P-3 B-1 F-1 J-1 H-1}</math> <math>\text{E-5 I-1 J+1 F+1 B-1 H-1 D-4 H-1 P-1}</math> <math>\text{C-5 H-1 E-2 P-2 I-1 H-1 C-1 J-2 P-1}</math> <math>\text{P+4}</math></p>
<p>A* Car Distance</p> <p>Waktu eksekusi: 32.41650000028312ms Tick count: 2033 Node diperiksa: 2405 Banyak pencarian: 16742 Branching factor: 1.29383087715820312 Langkah: 32 / 32 Step: <math>\emptyset \text{ C-1 H+1 B-1 G-3 B-1 H-1 C-3}</math> <math>\text{E-1 I-1 J+1 F+1 B-1 P-3 B-1 F-1 J-1 H-1}</math> <math>\text{E-3 C-5 H-1 I-4 J+1 F+1 B-1 H-2 D-4}</math> <math>\text{E+2 H+1 P-3 C-1 J-2 P+4}</math></p>	<p>A* Car Blocked</p> <p>Waktu eksekusi: 12.077100000232458ms Tick count: 1855 Node diperiksa: 2170 Banyak pencarian: 15220 Branching factor: 1.30145263671875 Langkah: 31 / 31 Step: <math>\emptyset \text{ C-2 E+1 F-1 J+1 P-3 B-1}</math> <math>\text{F-3 I-1 H-1 E-1 C-3 H-1 B-1 G-3 B-1}</math> <math>\text{E-3 I-4 H-1 D-1 C-1 J-3 E-1 F-3 B-1}</math> <math>\text{H-1 D-3 E-2 H-1 P-4}</math></p>

#### 4.24. Jam-24

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-24</td><td>6</td><td>6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>25</td><td>9</td><td></td></tr> <tr> <td>Step Optimal</td><td>:</td><td><math>\emptyset \text{ H-1 E-2 P-2 B+1}</math> <math>\text{G-1 I-2 D-1 J-2 F+3}</math> <math>\text{C+3 B+3 P+2 D+2 E+4}</math> <math>\text{H+4 G-2 P-2 D-2 B-4}</math> <math>\text{P+2 D+2 H-2 F-2 J+1}</math> <math>\text{P+2}</math></td><td><math>\dots \text{BGG.}</math> <math>\text{.HB ...}</math> <math>\text{EHPPI.K}</math> <math>\text{EDD.I.}</math> <math>\text{FFF.J.}</math> <math>\text{CC .. J.}</math></td></tr> </tbody> </table>	Nama	:	Jam-24	6	6	Banyak Step Optimal	:	25	9		Step Optimal	:	$\emptyset \text{ H-1 E-2 P-2 B+1}$ $\text{G-1 I-2 D-1 J-2 F+3}$ $\text{C+3 B+3 P+2 D+2 E+4}$ $\text{H+4 G-2 P-2 D-2 B-4}$ $\text{P+2 D+2 H-2 F-2 J+1}$ $\text{P+2}$	$\dots \text{BGG.}$ $\text{.HB ...}$ $\text{EHPPI.K}$ $\text{EDD.I.}$ $\text{FFF.J.}$ $\text{CC .. J.}$				
Nama	:	Jam-24	6	6														
Banyak Step Optimal	:	25	9															
Step Optimal	:	$\emptyset \text{ H-1 E-2 P-2 B+1}$ $\text{G-1 I-2 D-1 J-2 F+3}$ $\text{C+3 B+3 P+2 D+2 E+4}$ $\text{H+4 G-2 P-2 D-2 B-4}$ $\text{P+2 D+2 H-2 F-2 J+1}$ $\text{P+2}$	$\dots \text{BGG.}$ $\text{.HB ...}$ $\text{EHPPI.K}$ $\text{EDD.I.}$ $\text{FFF.J.}$ $\text{CC .. J.}$															
<p style="text-align: center;">UCS</p>  <p>         Waktu eksekusi: 79.97499999962747ms          Tick count: 4326          Node diperiksa: 4512          Banyak pencarian: 46154          Branching factor: 1.4679336547851562          Langkah: 25 / 25          Step: <math>\emptyset \text{ H-1 E-2 P-2 B+1 C-1 I-2 D-1 J-2}</math>  <math>\text{F+3 C+3 B+3 P+2 D+2 E+4 H+4 G-2}</math>  <math>\text{P-2 D-2 B-4 P+2 D+2 H-2 F-2 J+1}</math> <math>\text{P+2}</math> </p>	<p style="text-align: center;">GBFS Car Distance</p>  <p>         Waktu eksekusi: 65.09059999976307ms          Tick count: 3825          Node diperiksa: 4146          Banyak pencarian: 41526          Branching factor: 1.4380416870117188          Langkah: 26 / 26          Step: <math>\emptyset \text{ H-1 E-2 D-1 P-2 B+1 C-1 I-2 J-2}</math>  <math>\text{F+3 B+2 P+2 C+3 B+1 D+2 E+4 H+4}</math>  <math>\text{G-2 D-2 P-2 B-4 P+2 D+2 H-2 F-2 J+1}</math> <math>\text{P+2}</math> </p>	<p style="text-align: center;">GBFS Car Blocked</p>  <p>         Waktu eksekusi: 60.44899999909103ms          Tick count: 3721          Node diperiksa: 4037          Banyak pencarian: 40444          Branching factor: 1.4364395141601562          Langkah: 26 / 26          Step: <math>\emptyset \text{ H-1 E-2 P-2 B+1 C-1 I-2 D-1 J-2}</math>  <math>\text{F+3 B+2 P+2 C+3 B+1 D+2 E+4 H+4}</math>  <math>\text{G-2 P-2 B-4 P+2 D+2 H-2 F-2 J+1}</math> <math>\text{P+2}</math> </p>																
<p style="text-align: center;">A* Car Distance</p>  <p>         Waktu eksekusi: 33.264500000051409ms          Tick count: 4102          Node diperiksa: 4343          Banyak pencarian: 43937          Branching factor: 1.4647750854492188          Langkah: 25 / 25          Step: <math>\emptyset \text{ H-1 E-2 D-1 P-2 B+1 C-1 I-2 J-2}</math>  <math>\text{F+3 C+3 B+3 P+2 D+2 E+4 H+4 G-2}</math>  <math>\text{D-2 P-2 B-4 P+2 D+2 H-2 F-2 J+1}</math> <math>\text{P+2}</math> </p>	<p style="text-align: center;">A* Car Blocked</p>  <p>         Waktu eksekusi: 74.1688999993145ms          Tick count: 4185          Node diperiksa: 4377          Banyak pencarian: 44663          Branching factor: 1.4658279418945312          Langkah: 25 / 25          Step: <math>\emptyset \text{ H-1 E-2 P-2 D-1 B+1 C-1 I-2 J-2}</math>  <math>\text{F+3 C+3 B+3 P+2 D+2 E+4 H+4 G-2}</math>  <math>\text{P-2 D-2 B-4 P+2 D+2 H-2 F-2 J+1}</math> <math>\text{P+2}</math> </p>	<p style="text-align: center;">A* Car Blocked Recursive</p>  <p>         Waktu eksekusi: 51.600199999287724ms          Tick count: 4196          Node diperiksa: 4374          Banyak pencarian: 44744          Branching factor: 1.4659423828125          Langkah: 25 / 25          Step: <math>\emptyset \text{ H-1 E-2 P-2 D-1 B+1 C-1 I-2 J-2}</math>  <math>\text{F+3 C+3 B+3 P+2 D+2 E+4 H+4 G-2}</math>  <math>\text{P-2 D-2 B-4 P+2 D+2 H-2 F-2 J+1}</math> <math>\text{P+2}</math> </p>																

#### 4.25. Jam-25

<p>Nama : Jam-25</p> <p>Banyak Step Optimal : 27</p> <p>Step Optimal :</p> <ul style="list-style-type: none"> <li><math>\emptyset</math> H-1 F-1 L+1 P-1</li> <li>I-1 C+2 G+4 B+1 E+1</li> <li>P+1 L-3 C-3 J-3 C+3</li> <li>L+3 B-1 E-1 P-1 G-4</li> <li>C-2 M-2 I+2 N-1 F+3</li> <li>H+1 J-1 P+4</li> </ul>	<p>6 6 12 BBG.HH EEG .. F LPP.IFK LCCCIF LD.JMM .D.JNN</p>
<p>UCS</p>  <p>Waktu eksekusi: 114.4922000002116ms Tick count: 8621 Node diperiksa: 8824 Banyak pencarian: 82834 Branching factor: 1.4571685791015625 Langkah: 27 / 27</p> <p>Step: <math>\emptyset</math> H-1 F-1 L+1 P-1 I-1 C+2 G+4 B-1 E-1 P-1 L-3 C-3 J-3 C+3 L+3 B-1 E-1 P-1 G-4 C-2 M-2 I-2 N-1 F+3 H+1 J-1 P+4</p>	<p>GBFS Car Distance</p>  <p>Waktu eksekusi: 90.97599999979138ms Tick count: 6327 Node diperiksa: 6945 Banyak pencarian: 59622 Branching factor: 1.3142776489257812 Langkah: 35 / 35</p> <p>Step: <math>\emptyset</math> P+1 H-1 F-1 L+1 P-1 I-1 C+2 G+4 G-3 P+2 B-1 E-1 L-3 G+1 C-3 D-1 J-3 C+3 L-3 B-1 E-1 P-1 G-4 P-1 C-1 F-1 H+1 J-1 P-1 C-1 M-2 I-2 P-1 N-1 F+2 P+1</p>
<p>A* Car Distance</p>  <p>Waktu eksekusi: 139.88970000017434ms Tick count: 8546 Node diperiksa: 8804 Banyak pencarian: 82184 Branching factor: 1.3984603881835938 Langkah: 30 / 30</p> <p>Step: <math>\emptyset</math> P+1 H-1 F-1 L+1 P-1 I-1 C+2 G+4 G-4 P-2 B-1 E-1 L-3 C-3 P-1 J-3 C-3 L-3 B-1 E-1 N-1 P-1 G-4 P-1 C-2 M-2 F+3 H+1 J-1 I-2 P+3</p>	<p>A* Car Blocked</p>  <p>Waktu eksekusi: 53.928299999795854ms Tick count: 7161 Node diperiksa: 7858 Banyak pencarian: 68436 Branching factor: 1.44598388671875 Langkah: 27 / 27</p> <p>Step: <math>\emptyset</math> H-1 P-1 F-1 L+1 P-1 I-1 C+2 G+4 B-1 E-1 P-1 L-3 C-3 J-3 C-3 L+3 B-1 E-1 P-1 G-4 C-2 M-2 I-2 N-1 F+3 H+1 J-1 P+4</p>
	<p>A* Car Blocked Recursive</p>  <p>Waktu eksekusi: 94.33930000010878ms Tick count: 7054 Node diperiksa: 7738 Banyak pencarian: 67464 Branching factor: 1.424591064453125 Langkah: 28 / 28</p> <p>Step: <math>\emptyset</math> H-1 F-1 L+1 P-1 I-1 C+2 G+4 B-1 E-1 P-1 L-3 C-3 J-3 M-1 N-1 C-3 L-3 B-1 E-1 P-1 G-4 C-2 F+3 H+1 J-1 M-1 I-2 P+4</p>

#### 4.26. Jam-26

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-26</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>28</td><td>11</td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> F-1 E-2 I+1 L-1            J-1 C+1 H+3 P+1 B+1            F-2 M-1 L+2 B+3 P-1            L-2 H-4 P+1 C-1 J+1            L+3 I-1 B-3 D-1 C-3            D+1 L-2 M+3 P+2         </td><td>           .B.FFF            GB.HM.            GPPHMEK            ILLLME            I.D..J            ..DCCJ         </td></tr> </tbody> </table>	Nama	:	Jam-26	6 6	Banyak Step Optimal	:	28	11	Step Optimal	:	$\emptyset$ F-1 E-2 I+1 L-1 J-1 C+1 H+3 P+1 B+1 F-2 M-1 L+2 B+3 P-1 L-2 H-4 P+1 C-1 J+1 L+3 I-1 B-3 D-1 C-3 D+1 L-2 M+3 P+2	.B.FFF GB.HM. GPPHMEK ILLLME I.D..J ..DCCJ			
Nama	:	Jam-26	6 6												
Banyak Step Optimal	:	28	11												
Step Optimal	:	$\emptyset$ F-1 E-2 I+1 L-1 J-1 C+1 H+3 P+1 B+1 F-2 M-1 L+2 B+3 P-1 L-2 H-4 P+1 C-1 J+1 L+3 I-1 B-3 D-1 C-3 D+1 L-2 M+3 P+2	.B.FFF GB.HM. GPPHMEK ILLLME I.D..J ..DCCJ												
UCS	GBFS Car Distance	GBFS Car Blocked													
 <p>Waktu eksekusi: 41.85869999974966ms            Tick count: 4699            Node diperiksa: 4816            Banyak pencarian: 40081            Branching factor: 1.3958663940429688            Langkah: 28 / 28            Step: <math>\emptyset</math> F-1 E-2 I+1 L-1 C+1 H+3 P+1            B+1 F-2 M-1 L+2 B+3 P-1 L-2 H-4 P+1            C-1 H-3 L+3 B-3 D-1 C-3 D+1 L-2            M+3 P+2</p>	 <p>Waktu eksekusi: 26.945700000040233ms            Tick count: 1989            Node diperiksa: 2775            Banyak pencarian: 16541            Branching factor: 1.3184127807677188            Langkah: 30 / 30            Step: <math>\emptyset</math> H-1 L-2 H+2 P+2 B+1 F-3 M-1            E-1 J-1 C-1 H-1 L-2 B-3 L-2 P-1 H-4            P+1 C-1 J-1 L-3 B-3 D-1 C-3 D+1            L-2 M+3 E-1 P+1</p>	 <p>Waktu eksekusi: 4.421499999240041ms            Tick count: 688            Node diperiksa: 2241            Banyak pencarian: 6124            Branching factor: 1.2820968627929688            Langkah: 29 / 29            Step: <math>\emptyset</math> F-1 E-2 I+1 L-1 H+2 P+1 B+1            F-2 M-1 L-1 C+1 H+1 L-2 B+3 P-1 L-2            H-4 P-1 C-1 L-3 L-3 B-3 D-1 C-3            D+1 L-2 M+3 P+2</p>													
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive													
 <p>Waktu eksekusi: 65.205900000040829ms            Tick count: 3750            Node diperiksa: 3851            Banyak pencarian: 32130            Branching factor: 1.3837890625            Langkah: 28 / 28            Step: <math>\emptyset</math> H-1 L-2 H+2 P+1 B+1 F-3 M-1            E-2 J-1 C-1 H+1 L-2 B+3 L-2 P-1 H-4            P-1 C-1 J-1 L-3 B-3 D-1 C-3 D+1            L-2 M+3 P+2</p>	 <p>Waktu eksekusi: 63.273400000029653ms            Tick count: 3690            Node diperiksa: 3825            Banyak pencarian: 31694            Branching factor: 1.38304138018359375            Langkah: 28 / 28            Step: <math>\emptyset</math> F-1 E-2 H-1 L-1 J-1 C-1 H+3 P+1            B+1 F-2 M-1 L-2 B+3 P-1 L-2 H-4 P+1            C-1 H-3 L-3 B-3 D-1 C-3 D+1 L-2            M+3 P+2</p>	 <p>Waktu eksekusi: 79.25980000011623ms            Tick count: 3920            Node diperiksa: 4191            Banyak pencarian: 33776            Branching factor: 1.38641357421875            Langkah: 28 / 28            Step: <math>\emptyset</math> F-1 E-2 H-1 J-1 C-1 L-1 H+3 P+1            B+1 F-2 M-1 L-2 B+3 P-1 L-2 H-4 P+1            C-1 H-3 L-3 B-3 D-1 C-3 D+1 L-2            M+3 P+2</p>													

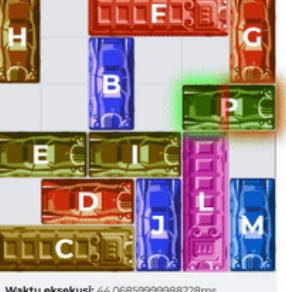
#### 4.27. Jam-27

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-27</td><td>6</td><td>6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>28</td><td>9</td><td></td></tr> <tr> <td>Step Optimal</td><td>:</td><td><math>\emptyset J-2 I+1 F+2 G+2</math> <math>H+2 E-2 P+1 B+4 P-1</math> <math>E+2 G-3 H-3 E-1 F-2</math> <math>I-4 F+2 E+1 H+3 E-2</math> <math>P+1 I+1 B-3 I-1 D-1</math> <math>C-3 J+3 F+1 P+3</math></td><td>BGGF .. BHHF .. PPEF..JK ..EIIJ ..D..J ..DCCC</td><td></td></tr> </tbody> </table>	Nama	:	Jam-27	6	6	Banyak Step Optimal	:	28	9		Step Optimal	:	$\emptyset J-2 I+1 F+2 G+2$ $H+2 E-2 P+1 B+4 P-1$ $E+2 G-3 H-3 E-1 F-2$ $I-4 F+2 E+1 H+3 E-2$ $P+1 I+1 B-3 I-1 D-1$ $C-3 J+3 F+1 P+3$	BGGF .. BHHF .. PPEF..JK ..EIIJ ..D..J ..DCCC					
Nama	:	Jam-27	6	6															
Banyak Step Optimal	:	28	9																
Step Optimal	:	$\emptyset J-2 I+1 F+2 G+2$ $H+2 E-2 P+1 B+4 P-1$ $E+2 G-3 H-3 E-1 F-2$ $I-4 F+2 E+1 H+3 E-2$ $P+1 I+1 B-3 I-1 D-1$ $C-3 J+3 F+1 P+3$	BGGF .. BHHF .. PPEF..JK ..EIIJ ..D..J ..DCCC																
UCS	GBFS Car Distance	GBFS Car Blocked																	
<p>Waktu eksekusi: 13.527499999850988ms Tick count: 2743 Node diperiksa: 2936 Banyak pencarian: 21347 Branching factor: 1.3616180419921875 Langkah: 28 / 28 Step: <math>\emptyset J-2 I+1 F+2 G+2 H+2 E-2 P+1</math> <math>B+4 P-1 E+2 G-3 H-3 E-1 F-2 I-4 F+2</math> <math>E+1 H+3 E-2 P+1 H B-3 I-1 D-1 C-3</math> <math>J-3 F+1 P+3</math></p>	<p>Waktu eksekusi: 10.59279999975115ms Tick count: 1464 Node diperiksa: 1740 Banyak pencarian: 11485 Branching factor: 1.2027359008789062 Langkah: 41 / 41 Step: <math>\emptyset J-2 I+1 F+2 G+2 H+2 E-2 P+1</math> <math>B+3 P-1 E+2 H-3 E-2 P+1 D-1 C-3 F+1</math> <math>P+2 E+1 G-3 E-1 P-2 F-3 C-2 B+1 D+1</math> <math>I-4 F+2 P-1 E+2 H-3 E-2 P+1 H-3 B-3</math> <math>I-3 D-3 C-3 F+1 P+2 J-3 P+3</math></p>	<p>Waktu eksekusi: 9.36560000001356ms Tick count: 1765 Node diperiksa: 2082 Banyak pencarian: 13668 Branching factor: 1.2219467163085938 Langkah: 39 / 39 Step: <math>\emptyset J-2 I+1 F+2 G+2 H+2 E-2 P+1</math> <math>B+3 P-1 E+2 H-3 E-2 D-1 C-3 F+1 P+3</math> <math>E+1 G-3 E-1 P-3 F-3 C-3 B+1 D+1 I-4</math> <math>F+2 E+2 H+3 E-2 P+1 H B-3 I-1 D-1</math> <math>C-1 I-3 C-2 F+1 P+3</math></p>																	
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive																	
<p>Waktu eksekusi: 25.617899999953806ms Tick count: 2399 Node diperiksa: 2600 Banyak pencarian: 18649 Branching factor: 1.3386993408203125 Langkah: 29 / 29 Step: <math>\emptyset J-2 I+1 F+2 G+2 H+2 E-2 P+1</math> <math>B+4 P-1 E+2 G-3 H-3 E-1 F-2 I-4 F+2</math> <math>E+1 H+3 E-2 P+1 H B-3 I-1 D-1 C-3</math> <math>F+1 P+2 J-3 P+3</math></p>	<p>Waktu eksekusi: 38.82230000011623ms Tick count: 2390 Node diperiksa: 2559 Banyak pencarian: 18593 Branching factor: 1.354793823242188 Langkah: 28 / 28 Step: <math>\emptyset J-2 I+1 F+2 G+2 H+2 E-2 P+1</math> <math>B+4 P-1 E+2 G-3 H-3 E-2 F-2 I-4 F+2</math> <math>E+2 H+3 E-2 P+1 H B-3 I-1 D-1 C-3</math> <math>J-3 F+1 P+3</math></p>	<p>Waktu eksekusi: 30.572200000286102ms Tick count: 2329 Node diperiksa: 2438 Banyak pencarian: 18127 Branching factor: 1.352813720703125 Langkah: 28 / 28 Step: <math>\emptyset J-2 I+1 F+2 H+2 G+2 E-2 P+1</math> <math>B+4 P-1 E+2 G-3 H-3 E-2 F-2 I-4 F+2</math> <math>E+2 H+3 E-2 P+1 H B-3 I-1 D-1 C-3</math> <math>J-3 F+1 P+3</math></p>																	

#### 4.28. Jam-28

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-28</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>30</td><td>11</td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> B+1 F+3 L-1 M-1            J+2 D+2 H+1 E+1 I-3            B+1 G-1 M-1 C+1 L+2            G-3 L-2 B-1 I+3 H-1            E-1 D-2 L+3 F-3 B-1            P+3 L-2 C-1 J-1 M+2            P+1         </td><td>         FFFB ..          .. LBGG          PPL ... K          HELIIM          HECCCM          DDJJ.M       </td></tr> </tbody> </table>	Nama	:	Jam-28	6 6	Banyak Step Optimal	:	30	11	Step Optimal	:	$\emptyset$ B+1 F+3 L-1 M-1 J+2 D+2 H+1 E+1 I-3 B+1 G-1 M-1 C+1 L+2 G-3 L-2 B-1 I+3 H-1 E-1 D-2 L+3 F-3 B-1 P+3 L-2 C-1 J-1 M+2 P+1	FFFB .. .. LBGG PPL ... K HELIIM HECCCM DDJJ.M			
Nama	:	Jam-28	6 6												
Banyak Step Optimal	:	30	11												
Step Optimal	:	$\emptyset$ B+1 F+3 L-1 M-1 J+2 D+2 H+1 E+1 I-3 B+1 G-1 M-1 C+1 L+2 G-3 L-2 B-1 I+3 H-1 E-1 D-2 L+3 F-3 B-1 P+3 L-2 C-1 J-1 M+2 P+1	FFFB .. .. LBGG PPL ... K HELIIM HECCCM DDJJ.M												
UCS	GBFS Car Distance	GBFS Car Blocked													
<p>Waktu eksekusi: 7.0670000007376075ms            Tick count: 2065            Node diperiksa: 2250            Banyak pencarian: 16737            Branching factor: 1.318992614746460938            Langkah: 30 / 30            Step: <math>\emptyset</math> B+1 F+3 L-1 M-1 J+2 D+2 H+1            E+1 I-3 B+1 G-1 M-1 C+1 L+2 G-3 L-2            B-1 I+3 H-1 E-1 D-2 L+3 F-3 B-1 P+3            L-2 C-1 J-1 M+2 P+1</p>	<p>Waktu eksekusi: 2.260999999940395ms            Tick count: 1857            Node diperiksa: 1282            Banyak pencarian: 6674            Branching factor: 1.2629623413085938            Langkah: 31 / 31            Step: <math>\emptyset</math> B+1 F+3 L-1 M-1 J+2 D+2 H+1            E+1 I-3 B+1 G-1 M-1 C+1 L+2 G-3 L-2            B-1 I+3 H-1 E-1 D-2 L+3 P+1 F-3 B-1            P+2 L-2 C-1 J-1 M+2 P+1</p>	<p>Waktu eksekusi: 2.3355999998748302ms            Tick count: 481            Node diperiksa: 805            Banyak pencarian: 3397            Branching factor: 1.2138671875            Langkah: 33 / 33            Step: <math>\emptyset</math> J+1 D+1 H+1 M-1 J-1 D+1 E+1            B+1 F+3 L-1 D-1 G-1 M-1 C+1 L+2            G-3 L-2 B-1 I+3 H-1 E-1 D-2 L+3 F-3 B-1            P+2 L-2 C-1 J-1 M+2 P+1</p>													
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive													
<p>Waktu eksekusi: 9.472500000149012ms            Tick count: 1447            Node diperiksa: 1853            Banyak pencarian: 11679            Branching factor: 1.3013916015625            Langkah: 30 / 30            Step: <math>\emptyset</math> B+1 F+3 L-1 M-1 J+2 D+2 H+1            E+1 I-3 B+1 G-1 M-1 C+1 L+2 G-3 L-2            B-1 I+3 H-1 E-1 D-2 L+3 F-3 B-1 P+3            L-2 C-1 J-1 M+2 P+1</p>	<p>Waktu eksekusi: 23.91800000052699ms            Tick count: 1448            Node diperiksa: 1838            Banyak pencarian: 11607            Branching factor: 1.30108642578125            Langkah: 30 / 30            Step: <math>\emptyset</math> M-1 J+2 D+2 H+1 E+1 B+1 F+3            L-1 I-3 B+1 G-1 M-1 C+1 L+2 G-3 L-2            B-1 I+3 H-1 E-1 D-2 L+3 F-3 B-1 P+3            L-2 C-1 J-1 M+2 P+1</p>	<p>Waktu eksekusi: 29.710200000554323ms            Tick count: 1329            Node diperiksa: 1650            Banyak pencarian: 10568            Branching factor: 1.2966079771914062            Langkah: 30 / 30            Step: <math>\emptyset</math> B+1 F+3 L-1 M-1 J+2 D+2 H+1            E+1 I-3 B+1 G-1 M-1 C+1 L+2 G-3 B-1            L-2 I+3 H-1 E-1 D-2 L+3 F-3 J-1 B-1            P+3 L-2 C-1 M+2 P+1</p>													

#### 4.29. Jam-29

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-29</td><td>6</td><td>6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>31</td><td>11</td><td></td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> F+1 G-1 I+1 J-3  I-1 D+2 M-1 C+3 H+1  E-1 B+3 P+1 E+1 H-4  P-1 E-1 B-3 D-2 C-3  M+1 I+1 J+3 I-2 L+3  F+2 B-1 P+3 B+1 F-1  G-1 P+1 </td><td> FFF.L.  ..B.L.  PPB.LGK  HEEIIG  HDDJ.M  CCCJ.M </td><td></td></tr> </tbody> </table>	Nama	:	Jam-29	6	6	Banyak Step Optimal	:	31	11		Step Optimal	:	$\emptyset$ F+1 G-1 I+1 J-3 I-1 D+2 M-1 C+3 H+1 E-1 B+3 P+1 E+1 H-4 P-1 E-1 B-3 D-2 C-3 M+1 I+1 J+3 I-2 L+3 F+2 B-1 P+3 B+1 F-1 G-1 P+1	FFF.L. ..B.L. PPB.LGK HEEIIG HDDJ.M CCCJ.M		
Nama	:	Jam-29	6	6												
Banyak Step Optimal	:	31	11													
Step Optimal	:	$\emptyset$ F+1 G-1 I+1 J-3 I-1 D+2 M-1 C+3 H+1 E-1 B+3 P+1 E+1 H-4 P-1 E-1 B-3 D-2 C-3 M+1 I+1 J+3 I-2 L+3 F+2 B-1 P+3 B+1 F-1 G-1 P+1	FFF.L. ..B.L. PPB.LGK HEEIIG HDDJ.M CCCJ.M													
UCS	GBFS Car Distance	GBFS Car Blocked														
 <p> Waktu eksekusi: 50.775400000062436ms  Tick count: 4324  Node diperiksa: 4354  Banyak pencarian: 38390  Branching factor: 1.3453216552734375  Langkah: 31 / 31  Step: <math>\emptyset</math> F+1 G-1 I+1 J-3 I-1 D+2 M-1 C+3  H+1 E+1 B+3 P+1 E+1 H-4 P+1 E+1 B+3  D-2 C-3 M+1 H+1 J+3 I-2 L+3 F+2 B+1  P+3 B+1 F+1 G-1 P+1 </p>	 <p> Waktu eksekusi: 19.5921999999839067ms  Tick count: 4200  Node diperiksa: 4310  Banyak pencarian: 37510  Branching factor: 1.2423782348632812  Langkah: 41 / 41  Step: <math>\emptyset</math> G+1 I+1 J-2 D+2 C+1 H+1 E+1  B+2 P+1 J-2 P+1 H+1 M+1 C+2 B+1 E+1  H-3 E+1 B+1 C-3 M+1 H+1 P+1 J+1 F+1  H+1 P+1 B+2 D-2 J+3 L+1 F+2 B+1  P+2 L+2 P+1 B+1 F-1 G-1 P+1 </p>	 <p> Waktu eksekusi: 14.1372999999594688ms  Tick count: 1969  Node diperiksa: 3149  Banyak pencarian: 16324  Branching factor: 1.1911392211914062  Langkah: 0 / 45  Step: <math>\emptyset</math> G-2 H+1 J-4 D+2 C+1 H+1 E+1  B+2 P+1 H-1 M+1 C+2 B+1 E+1 H-3 E+1  J+1 P+1 H-1 P+1 B-2 D-3 B+1 P+1 H+1  F+1 J+1 C-3 M+1 H+1 J+4 F+1 G-1 P+1  B-2 I-2 L+3 G+1 F+2 B+1 P+3 B+1 F+1  G-1 P+1 </p>														
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive														
 <p> Waktu eksekusi: 34.28129999991506ms  Tick count: 4292  Node diperiksa: 4346  Banyak pencarian: 38182  Branching factor: 1.345062255859375  Langkah: 31 / 31  Step: <math>\emptyset</math> G+1 I+1 J-3 I-1 D+2 M-1 C+3  H+1 E+1 B+3 P+1 E+1 H-4 E+1 P+1  D-2 C-3 M+1 H+1 J+3 I-2 L+3 F+2  B+1 P+3 B+1 F+1 G-1 P+1 </p>	 <p> Waktu eksekusi: 27.94319999963045ms  Tick count: 4272  Node diperiksa: 4348  Banyak pencarian: 37962  Branching factor: 1.2957000732421875  Langkah: 35 / 35  Step: <math>\emptyset</math> G-2 H+1 J-4 H+1 D+2 M+1 C+3  H+1 E+1 B+3 P+1 E+1 H-3 E+1 B+1 C+3  M+1 H+1 J+1 F+1 H+1 P+1 B+2 D+2 J+3  I-2 L+3 G+1 F+2 B+1 P+3 B+1 F+1 G-1  P+1 </p>	 <p> Waktu eksekusi: 44.06859999988228ms  Tick count: 4312  Node diperiksa: 4345  Banyak pencarian: 38309  Branching factor: 1.318878173828125  Langkah: 33 / 33  Step: <math>\emptyset</math> G-2 H+1 J-4 H+1 M+1 C+3 H+1 E+1  D+2 B+3 P+1 J+1 F+1 E+1 H-4 E+1 P+1  B-3 D-2 C-3 M+1 H+1 J+3 I-2 L+3 G+1  F+2 B-1 P+3 B+1 F-1 G-1 P+1 </p>														

### 4.30. Jam-30

<table border="1"> <tbody> <tr> <td>Nama</td> <td>:</td> <td>Jam-30</td> <td>6</td> <td>6</td> </tr> <tr> <td>Banyak Step Optimal</td> <td>:</td> <td>32</td> <td>9</td> <td></td> </tr> <tr> <td>Step Optimal</td> <td>:</td> <td><math>\emptyset</math> E+1 H+1 C-1 D+2 I+1 F+3 P-1 B+1 J-1 C-2 E+1 G+3 E-1 C+1 J+1 B-1 P+3 F-3 H-1 I-1 B+4 P-2 H+1 F+1 J-3 C-1 E+1 G-4 E-1 D-1 C+3 P+3</td> <td>F.BJJ F.BG.. FPPG..K HHEE.C .....C IIDD.C</td> <td></td> </tr> </tbody> </table>  <p>Waktu eksekusi: 5.812099999974817ms Tick count: 1163 Node diperiksa: 1171 Banyak pencarian: 8622 Branching factor: 1.2639694213867188 Langkah: 32 / 32 Step: <math>\emptyset</math> E+1 H+1 C-1 D+2 B+1 F+3 P-1 I+1 C-2 E+1 G+3 E-1 C+1 J-1 B+1 P+3 F-3 H-1 I-1 B+4 P-2 H+1 F+1 J-3 C-1 E+1 G-4 E-1 D-1 C+3 P+3</p>	Nama	:	Jam-30	6	6	Banyak Step Optimal	:	32	9		Step Optimal	:	$\emptyset$ E+1 H+1 C-1 D+2 I+1 F+3 P-1 B+1 J-1 C-2 E+1 G+3 E-1 C+1 J+1 B-1 P+3 F-3 H-1 I-1 B+4 P-2 H+1 F+1 J-3 C-1 E+1 G-4 E-1 D-1 C+3 P+3	F.BJJ F.BG.. FPPG..K HHEE.C .....C IIDD.C		 <p>Waktu eksekusi: 2.97560000004708767ms Tick count: 925 Node diperiksa: 1025 Banyak pencarian: 6855 Branching factor: 1.2439727783203125 Langkah: 33 / 33 Step: <math>\emptyset</math> E+1 H+1 C-1 D+2 B+1 F+3 P-1 I+1 C-2 E+1 G+3 E-1 C+1 J-1 B+1 P+3 F-3 H-1 I-1 B+4 H+1 F+1 J-3 C-1 E+1 G-4 P-2 E-1 D-1 C+3 P+3</p>	 <p>Waktu eksekusi: 19.400399999693036ms Tick count: 822 Node diperiksa: 969 Banyak pencarian: 6139 Branching factor: 1.239227294921875 Langkah: 33 / 33 Step: <math>\emptyset</math> E+1 H+1 C-1 D+2 B+1 F+3 P-1 I+1 C-2 E+1 G+3 E-1 C+1 J-1 B+1 P+3 F-3 H-1 I-1 B+4 H+1 F+1 J-3 C-1 E+1 G-4 P-2 E-1 D-1 C+3 P+3</p>
Nama	:	Jam-30	6	6													
Banyak Step Optimal	:	32	9														
Step Optimal	:	$\emptyset$ E+1 H+1 C-1 D+2 I+1 F+3 P-1 B+1 J-1 C-2 E+1 G+3 E-1 C+1 J+1 B-1 P+3 F-3 H-1 I-1 B+4 P-2 H+1 F+1 J-3 C-1 E+1 G-4 E-1 D-1 C+3 P+3	F.BJJ F.BG.. FPPG..K HHEE.C .....C IIDD.C														
 <p>Waktu eksekusi: 12.053899999707937ms Tick count: 1053 Node diperiksa: 1146 Banyak pencarian: 7788 Branching factor: 1.2494659423828125 Langkah: 33 / 33 Step: <math>\emptyset</math> E+1 H+1 C-1 D+2 B+1 F+3 P-1 I+1 C-2 E+1 G+3 E-1 C+1 J-1 B+1 P+3 F-3 H-1 I-1 B+4 H+1 F+1 J-3 C-1 E+1 P-2 G-4 P-2 E-1 D-1 C+3 P+3</p>	 <p>Waktu eksekusi: 6.702499999664724ms Tick count: 1084 Node diperiksa: 1146 Banyak pencarian: 8011 Branching factor: 1.250679016132812 Langkah: 33 / 33 Step: <math>\emptyset</math> E+1 H+1 D+1 B+1 F+3 P-1 B+1 I-3 G-1 C-3 E-1 D-1 G-4 E-1 C-1 J-3 B-1 P+3 F-3 H-1 I-1 B+4 P-2 H+1 F+1 J-3 C-1 E-1 G-4 E-1 D-1 C+3 P+3</p>	 <p>Waktu eksekusi: 17.30640000011772ms Tick count: 1110 Node diperiksa: 1165 Banyak pencarian: 8239 Branching factor: 1.2619400024414062 Langkah: 32 / 32 Step: <math>\emptyset</math> E+1 H+1 D+1 B+1 F+3 P-1 B+1 I-3 G-1 C-3 E-1 D-1 G-4 E-1 C-1 J-3 B-1 P+3 F-3 H-1 I-1 B+4 P-2 H+1 F+1 C-1 E-1 G-4 E-1 D-1 C+3 P+3</p>															

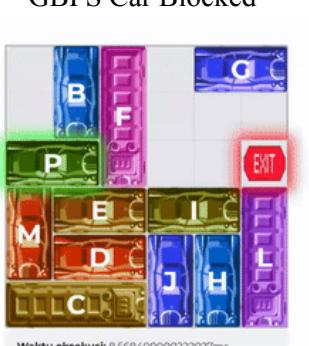
### 4.31. Jam-31

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-31</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>37</td><td>10</td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> B+1 E-2 P-1 C-2            D+3 C+2 P+1 E+4 B-1            P-1 C-3 I-3 C+3 G+1            H-1 L-1 D+1 G+1 P+3            C-3 I+1 E-3 I-1 C+3            B+1 E-1 P-3 C-2 G-1            D-4 C+2 G+1 P+3 C-1            J-1 L+2 P+1         </td><td>           BB.FFF            ... GHH            EPPG.LK            E.CIIL            DDC..L            ..CJJJ         </td></tr> </tbody> </table>	Nama	:	Jam-31	6 6	Banyak Step Optimal	:	37	10	Step Optimal	:	$\emptyset$ B+1 E-2 P-1 C-2 D+3 C+2 P+1 E+4 B-1 P-1 C-3 I-3 C+3 G+1 H-1 L-1 D+1 G+1 P+3 C-3 I+1 E-3 I-1 C+3 B+1 E-1 P-3 C-2 G-1 D-4 C+2 G+1 P+3 C-1 J-1 L+2 P+1	BB.FFF ... GHH EPPG.LK E.CIIL DDC..L ..CJJJ			
Nama	:	Jam-31	6 6												
Banyak Step Optimal	:	37	10												
Step Optimal	:	$\emptyset$ B+1 E-2 P-1 C-2 D+3 C+2 P+1 E+4 B-1 P-1 C-3 I-3 C+3 G+1 H-1 L-1 D+1 G+1 P+3 C-3 I+1 E-3 I-1 C+3 B+1 E-1 P-3 C-2 G-1 D-4 C+2 G+1 P+3 C-1 J-1 L+2 P+1	BB.FFF ... GHH EPPG.LK E.CIIL DDC..L ..CJJJ												
<p style="text-align: center;">UCS</p>  <div> <p>Waktu eksekusi: 25.162299999967217ms            Tick count: 3977            Node diperiksa: 4132            Banyak pencarian: 33209            Branching factor: 1.2707138061523438            Langkah: 37 / 37</p> <p>Step: <math>\emptyset</math> B+1 E2 P1 C2 D3 C2 P1            E4 B1 P3 C3 I3 C3 G1 H1 L1            D1 G1 P2 C3 H1 E3 I1 C3            E1 P3 C2 G1 D4 C2 G1 P3 C1            J1 L2 P1</p> </div>	<p style="text-align: center;">GBFS Car Distance</p>  <div> <p>Waktu eksekusi: 45.374099999666214ms            Tick count: 3063            Node diperiksa: 3722            Banyak pencarian: 25613            Branching factor: 1.2366867065429688            Langkah: 40 / 40</p> <p>Step: <math>\emptyset</math> B+1 E2 P1 C2 D3 C2 P1            E4 B1 P3 C3 I3 C3 G1 H1 L1            D1 G1 P2 C3 H1 E3 I1 C3            B+1 E1 P2 G1 P3 C2 D4 C2 P1            G1 P2 C1 I1 L2 P1</p> </div>	<p style="text-align: center;">GBFS Car Blocked</p>  <div> <p>Waktu eksekusi: 24.49430000036955ms            Tick count: 3441            Node diperiksa: 3664            Banyak pencarian: 28698            Branching factor: 1.2564697265625            Langkah: 38 / 38</p> <p>Step: <math>\emptyset</math> B+1 E2 P1 C2 D3 C2 P1            E4 B1 P3 C3 I3 C3 G1 H1 L1            D1 G1 P2 C3 H1 E3 I1 C3            B+1 E1 P2 G1 P3 C2 D4 C2 P1            C1 J1 L2 P1</p> </div>													
<p style="text-align: center;">A* Car Distance</p>  <div> <p>Waktu eksekusi: 71.89389999955893ms            Tick count: 3785            Node diperiksa: 4065            Banyak pencarian: 31670            Branching factor: 1.260101318359375            Langkah: 38 / 38</p> <p>Step: <math>\emptyset</math> B+1 E2 P1 C2 D3 C2 P1            E4 B1 P3 C3 I3 C3 G1 H1 L1            L1 D1 G1 P2 C3 H1 E3 I1 C3            B+1 E1 P3 C2 G1 D4 C2 G1 P3 C1            C1 J1 L2 P1</p> </div>	<p style="text-align: center;">A* Car Blocked</p>  <div> <p>Waktu eksekusi: 26.6214000005275ms            Tick count: 3811            Node diperiksa: 3981            Banyak pencarian: 31810            Branching factor: 1.2690658569335938            Langkah: 37 / 37</p> <p>Step: <math>\emptyset</math> B+1 E2 P1 C2 D3 C2 P1            E4 B1 P3 C3 I3 C3 G1 H1 L1            D1 G1 P2 C3 H1 E3 I1 C3            B+1 E1 P2 G1 P3 C2 D4 C2 G1 P3 C1            J1 L2 P1</p> </div>	<p style="text-align: center;">A* Car Blocked Recursive</p>  <div> <p>Waktu eksekusi: 73.42949999962002ms            Tick count: 3750            Node diperiksa: 3922            Banyak pencarian: 31295            Branching factor: 1.2684478759765625            Langkah: 37 / 37</p> <p>Step: <math>\emptyset</math> B+1 E2 P1 C2 D3 C2 P1            E4 B1 P3 C3 I3 C3 G1 H1 L1            D1 G1 P3 C3 H1 E3 I1 C3            E1 P3 C2 G1 D4 C2 G1 P3 C1            J1 L2 P1</p> </div>													

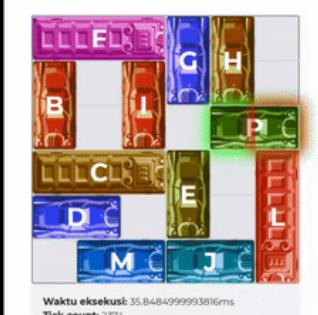
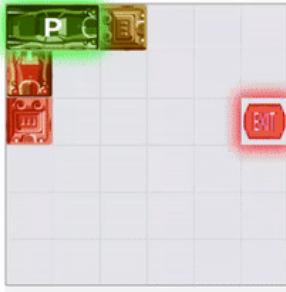
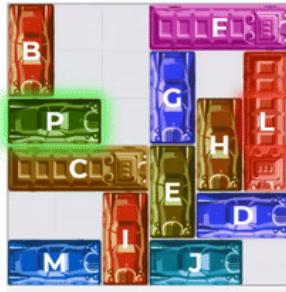
#### 4.32. Jam-32

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-32</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>37</td><td>10</td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> G+1 H-1 L-3 D+1          J-1 C+4 E+1 I-1 F+3          J+1 D-1 L+1 H+1 G-1          P+3 F-3 I+1 E-3 I-1          F+3 B+1 E-1 P-3 G+1          H-1 L-1 F-1 D+1 J-1          C-4 F+1 J+1 D-1 L+3          H+1 G-1 P+4       </td><td>         BBFGHH          .. FG..          PPF ... K          EIIDDL          E .. J.L          CC.J.L       </td></tr> </tbody> </table>	Nama	:	Jam-32	6 6	Banyak Step Optimal	:	37	10	Step Optimal	:	$\emptyset$ G+1 H-1 L-3 D+1 J-1 C+4 E+1 I-1 F+3 J+1 D-1 L+1 H+1 G-1 P+3 F-3 I+1 E-3 I-1 F+3 B+1 E-1 P-3 G+1 H-1 L-1 F-1 D+1 J-1 C-4 F+1 J+1 D-1 L+3 H+1 G-1 P+4	BBFGHH .. FG.. PPF ... K EIIDDL E .. J.L CC.J.L	<p><b>UCS</b></p> <pre> Waktu eksekusi: 4.554000000469587ms Tick count: 576 Node diperiksa: 601 Banyak pencarian: 3278 Branching factor: 1.1634564208984375 Langkah: 37 / 37 Step: <math>\emptyset</math> G+1 H-1 L-3 D+1 J+1 C+4 E+1 I-1 F+3 P+4 D+1 L+1 H+1 G-1 P+3 F-3 H-1 E-3 I-1 F-3 B+1 E-1 P-3 G+1 H-1 L-1 F-1 D+1 J-1 C-4 F+1 J+1 D-1 L+3 H+1 G-1 P+4       </pre>	<p><b>GBFS Car Distance</b></p> <pre> Waktu eksekusi: 1.9982000002637506ms Tick count: 505 Node diperiksa: 566 Banyak pencarian: 2843 Branching factor: 1.1470947265625 Langkah: 43 / 43 Step: <math>\emptyset</math> G+1 H-1 L-3 D+1 J+1 C+4 E+1 I-1 F+3 P+4 D+1 L+1 H+1 G-1 P+3 F-3 H-1 E-3 I-1 F-3 B+1 E-1 P-3 G+1 H-1 L-1 F-1 D+1 J-1 C-4 F+1 J+1 D-1 L+3 H+1 G-1 P+4       </pre>	<p><b>GBFS Car Blocked</b></p> <pre> Waktu eksekusi: 2.1698000002652407ms Tick count: 467 Node diperiksa: 508 Banyak pencarian: 2628 Branching factor: 1.1637802124023438 Langkah: 39 / 39 Step: <math>\emptyset</math> C+1 E+1 I-1 G+1 H-1 L-3 D+1 J+1 C+2 F+3 C+1 H+1 D+1 L+1 H+1 G-1 P+3 F-3 H-1 E-3 I-1 F-3 B+1 E-1 P-2 G+1 H-1 L-1 D+1 J-1 P-3 F-1 C-4 F+1 P+2 J-1 D-1 L-1 H+1 G-1 P+2 L+2 H+1 L+3 H+1 G-1 P+4       </pre>
Nama	:	Jam-32	6 6												
Banyak Step Optimal	:	37	10												
Step Optimal	:	$\emptyset$ G+1 H-1 L-3 D+1 J-1 C+4 E+1 I-1 F+3 J+1 D-1 L+1 H+1 G-1 P+3 F-3 I+1 E-3 I-1 F+3 B+1 E-1 P-3 G+1 H-1 L-1 F-1 D+1 J-1 C-4 F+1 J+1 D-1 L+3 H+1 G-1 P+4	BBFGHH .. FG.. PPF ... K EIIDDL E .. J.L CC.J.L												
<p><b>A* Car Distance</b></p> <pre> Waktu eksekusi: 4.075999999046333ms Tick count: 567 Node diperiksa: 598 Banyak pencarian: 3216 Branching factor: 1.1654891967773438 Langkah: 40 / 40 Step: <math>\emptyset</math> G+1 H-1 L-3 D+1 J+1 C+4 E+1 I-1 F+3 P+4 D+1 L+1 H+1 G-1 P+2 F-3 H-1 E-3 I-1 F-3 B+1 E-1 P-2 G+1 H-1 L-1 D+1 J-1 P-3 F-1 C-4 F+1 P+2 J-1 D-1 L-3 H+1 G-1 P+4       </pre>	<p><b>A* Car Blocked</b></p> <pre> Waktu eksekusi: 4.310800000093877ms Tick count: 516 Node diperiksa: 558 Banyak pencarian: 2916 Branching factor: 1.1730880737304688 Langkah: 38 / 38 Step: <math>\emptyset</math> G+1 E+1 G-1 H-1 L-3 D+1 J+1 C+3 F+3 H-1 D+1 L+1 H+1 G-1 P+3 F-3 H-1 E-3 I-1 F-3 B+1 E-1 P-3 G+1 H-1 L-1 D+1 J-1 P-3 F-1 C-4 F+1 P+2 D-1 L-3 H+1 G-1 P+4       </pre>	<p><b>A* Car Blocked Recursive</b></p> <pre> Waktu eksekusi: 9.010600000619888ms Tick count: 516 Node diperiksa: 553 Banyak pencarian: 2924 Branching factor: 1.1791839599609375 Langkah: 37 / 37 Step: <math>\emptyset</math> G+1 H-1 L-3 D+1 J+1 C+4 E+1 I-1 F+3 P+4 D+1 L+1 H+1 G-1 P+3 F-3 H-1 E-3 I-1 F-3 B+1 E-1 P-3 G+1 H-1 L-1 D+1 J-1 F-1 C-4 F+1 P+2 D-1 L-3 H+1 G-1 P+4       </pre>													

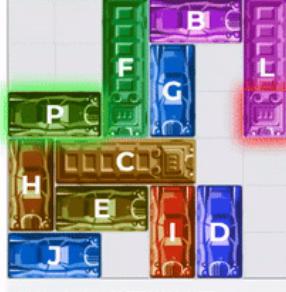
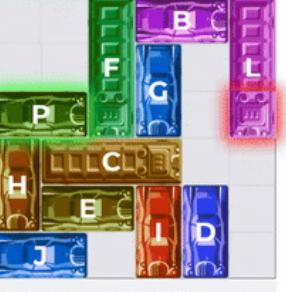
### 4.33. Jam-33

Nama : Jam-33	6 6
Banyak Step Optimal : 40	11 .BF.GG
Step Optimal : φ G-1 L-3 I+1 J-3 C+1 M+1 E-1 I-2 H-3 I+2 D+3 J+2 C+2 F+3 G-1 H-1 P+3 B+1 G-2 F-3 E+1 M-3 E-1 F+3 G+2 M-1 B-1 P-3 F-2 J-1 D-4 F+2 J+1 P+3 F-1 C-3 J+1 I-1 L+3 P+1	.BF ... PPF ... K MEEIIL MDDJHL CCCJHL
<b>UCS</b>  Waktu eksekusi: 23.394299999810755ms Tick count: 4068 Node diperiksa: 4183 Banyak pencarian: 37616 Branching factor: 1.25 Langkah: 40 / 40 Step: φ G-1 L-3 I+1 J-3 C+1 M+1 E-1 I-2 H-3 D+2 D+3 I+2 C+2 F+3 G-1 H-1 P+3 B+1 G-2 F-3 E+1 M-3 E-1 F+3 G+2 M-1 B-1 P-3 F-2 J-1 D-4 F+2 J+1 P+3 F-1 C-3 J+1 I-1 L+3 P+1	<b>GBFS Car Distance</b>  Waktu eksekusi: 10.663800000213087ms Tick count: 2290 Node diperiksa: 2850 Banyak pencarian: 19968 Branching factor: 1.1910957900390625 Langkah: 46 / 46 Step: φ G-1 L-3 I+1 J-3 C+1 M+1 E-1 I-2 H-3 D+2 D+3 I+2 C+2 F+3 P+1 H-1 D+1 I+2 P+1 G-1 H-1 P+2 B+1 G-2 F-3 E+1 M-3 E-1 F+3 G-2 M-1 B-1 P-3 F-2 J-1 D-4 F+2 J+1 P+3 F-1 C-3 J+1 I-1 L+3 P+1
<b>GBFS Car Blocked</b>  Waktu eksekusi: 8.568400000222027ms Tick count: 1121 Node diperiksa: 1763 Banyak pencarian: 8240 Branching factor: 1.1467919921875 Langkah: 0 / 51 Step: φ G-1 L-3 I+1 J-3 C+1 J-1 C+1 M+1 E-1 I-1 H-3 H-1 F-1 I-1 G-2 H-1 D-2 L-3 C-2 F-2 L-2 H-1 D-1 I+2 P+3 B+1 G-2 F-3 E+1 M-3 E-1 F+3 G+2 M-1 B-1 P-3 F-2 J-1 D-3 D+1 D-1 E+2 P+3 F-1 C-3 J+1 I-1 L+3 P+1	
<b>A* Car Distance</b>  Waktu eksekusi: 40.22440000064671ms Tick count: 2588 Node diperiksa: 3010 Banyak pencarian: 22723 Branching factor: 1.2325515747070312 Langkah: 40 / 40 Step: φ G-1 L-3 I-3 C+1 M+1 E-1 I-2 H-3 D+2 D+3 I+2 C+2 F+3 P+2 B+1 G-3 H-1 P+1 F-3 E+1 M-3 E-1 F+3 G+2 M-1 B-1 P-3 F-2 J-1 D-4 F+2 J+1 P+3 F-1 C-3 J+1 I-1 L+3 P+1	<b>A* Car Blocked</b>  Waktu eksekusi: 21.926800000481308ms Tick count: 2618 Node diperiksa: 3010 Banyak pencarian: 22859 Branching factor: 1.232551575768359375 Langkah: 40 / 40 Step: φ G-1 L-3 I-3 C+1 M+1 E-1 I-2 H-3 D+2 D+3 I+2 C+2 F+3 G-1 H-1 P+3 B+1 G-2 F-3 E+1 M-3 E-1 F+3 G+2 M-1 B-1 P-3 F-2 J-1 D-4 F+2 J+1 P+3 F-1 C-3 J+1 I-1 L+3 P+1
<b>A* Car Blocked Recursive</b>  Waktu eksekusi: 78.84900000039488ms Tick count: 2866 Node diperiksa: 3277 Banyak pencarian: 26221 Branching factor: 1.237454223632812 Langkah: 40 / 40 Step: φ G-1 L-3 I-3 C+1 M+1 E-1 I-2 H-3 D+2 D+3 I+2 C+2 F+3 G-1 H-1 P+3 B+1 G-2 F-3 E+1 M-3 E-1 F+3 G+2 M-1 B-1 P-3 F-2 J-1 D-4 F+2 J+1 P+3 F-1 C-3 J+1 I-1 L+3 P+1	

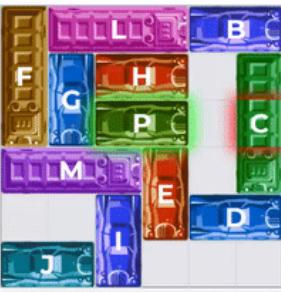
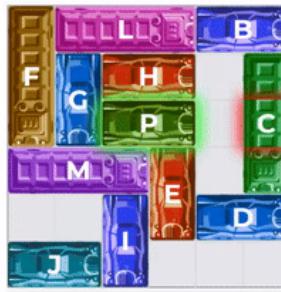
#### 4.34. Jam-34

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-34</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>43</td><td>11</td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> F-1 L-1 P+1 H-1 J+1            E+1 C+3 B+3 F-2 G-1 P-1            I-3 M+1 B+1 C-3 L+1 E-2            D-3 E+2 G+1 F+3 I-1 P+1            C+1 B-4 P-1 I+1 F-1 L-1            C+2 D-1 M-1 I+3 P+1 B+1            F-2 G-1 H-1 C-3 E-1 J-1            L+3 P+3         </td><td>           B .. FFF            B .. G..L            PP.GHLK            CCCEHL            .. IEEDD            MMIJJ.         </td></tr> </tbody> </table>	Nama	:	Jam-34	6 6	Banyak Step Optimal	:	43	11	Step Optimal	:	$\emptyset$ F-1 L-1 P+1 H-1 J+1 E+1 C+3 B+3 F-2 G-1 P-1 I-3 M+1 B+1 C-3 L+1 E-2 D-3 E+2 G+1 F+3 I-1 P+1 C+1 B-4 P-1 I+1 F-1 L-1 C+2 D-1 M-1 I+3 P+1 B+1 F-2 G-1 H-1 C-3 E-1 J-1 L+3 P+3	B .. FFF B .. G..L PP.GHLK CCCEHL .. IEEDD MMIJJ.	
Nama	:	Jam-34	6 6										
Banyak Step Optimal	:	43	11										
Step Optimal	:	$\emptyset$ F-1 L-1 P+1 H-1 J+1 E+1 C+3 B+3 F-2 G-1 P-1 I-3 M+1 B+1 C-3 L+1 E-2 D-3 E+2 G+1 F+3 I-1 P+1 C+1 B-4 P-1 I+1 F-1 L-1 C+2 D-1 M-1 I+3 P+1 B+1 F-2 G-1 H-1 C-3 E-1 J-1 L+3 P+3	B .. FFF B .. G..L PP.GHLK CCCEHL .. IEEDD MMIJJ.										
UCS	GBFS Car Distance	GBFS Car Blocked											
 <p> <b>Waktu eksekusi:</b> 54.44759999960661ms  <b>Tick count:</b> 4383  <b>Node diperiksa:</b> 4418  <b>Banyak pencarian:</b> 37481  <b>Branching factor:</b> 1228515625  <b>Langkah:</b> 0 / 43  <b>Step:</b>            0 F-1 L-1 P+1 H-1 J+1 C+3            B+3 F-2 G-1 P-1 I-3 M+1 B+3 L+1            E-2 D-3 E2 G1 F3 P1 C1 B-4            P1 H-1 F-1 L-1 C-2 D-1 M-1 I-3 P+1            B1 F2 G1 H1 C3 E1 J1 L3 P+3         </p>	 <p> <b>Waktu eksekusi:</b> 16.195000000298023ms  <b>Tick count:</b> 3555  <b>Node diperiksa:</b> 3655  <b>Banyak pencarian:</b> 29867  <b>Branching factor:</b> 11839447021484375  <b>Langkah:</b> 0 / 50  <b>Step:</b>            0 P+1 B+1 F-3 G-1 P+1 H-2 P+1            L+1 J+1 E+1 C+3 I+1 M+1 B+3 P+1 I-2 C-3            E-2 D-3 E1 I-1 L+3 H+3 G+1 F+3 H-1            P+1 L-1 I-1 E+1 C1 B3 C1 E1 I-1            L+1 B-1 P3 D-1 H-1 F2 H-3 L-3 I-1            E+1 C-3 H-2 P+1 B+1 F-1 G1 P+2 H-1            C3 E3 I3 L3 P+1         </p>	 <p> <b>Waktu eksekusi:</b> 35.8484999993816ms  <b>Tick count:</b> 2374  <b>Node diperiksa:</b> 3402  <b>Banyak pencarian:</b> 19752  <b>Branching factor:</b> 11474380493164062  <b>Langkah:</b> 57 / 57  <b>Step:</b>            0 P-2 H-2 P3 B2 P1 F-1 G-1 L-1            I-1 E+1 C+3 I-1 M+1 B+3 P+1 I-2 C-3            E-2 D-3 E1 I-1 L+3 H+3 G+1 F+3 H-1            P+1 L-1 I-1 E+1 C1 B3 C1 E1 I-1            L+1 B-1 P3 D-1 H-1 F2 H-3 L-3 I-1            E+1 C-3 H-2 P+1 B+1 F-1 G1 P+2 H-1            C3 E3 I3 L3 P+1         </p>											
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive											
 <p> <b>Waktu eksekusi:</b> 75.377300000074953ms  <b>Tick count:</b> 4184  <b>Node diperiksa:</b> 4341  <b>Banyak pencarian:</b> 35667  <b>Branching factor:</b> 1208892822265625  <b>Langkah:</b> 0 / 46  <b>Step:</b>            0 P+1 B+1 F-3 G-1 H-2 P+2 L-1            I+1 E+1 C+3 I-1 M+1 B+3 P-3 H+1 I-2            C-3 L+1 E2 D3 E2 G1 F3 I-1 P+1            C1 B-4 D1 P1 H-1 F-1 L-1 C2 H-2            P1 B1 F2 G1 H1 P+2 I2 C3 E1            J-1 L+3 P+1         </p>	 <p> <b>Waktu eksekusi:</b> 34.986399999819696ms  <b>Tick count:</b> 4144  <b>Node diperiksa:</b> 4369  <b>Banyak pencarian:</b> 35415  <b>Branching factor:</b> 1203826904296875  <b>Langkah:</b> 0 / 44  <b>Step:</b>            0 P+1 B+1 F-3 G-1 H-2 L-1 J+1            E+1 C+3 I-1 M+1 B+3 P-3 I-2 C-3 L+1            E-2 D-3 E2 H+3 C1 G1 F+3 I-1 P+1            B-4 P1 D1 M1 H1 F2 L1 H3 C+2            I+3 P1 B1 F1 G1 C3 E1 I-1 L+3            P+3         </p>	 <p> <b>Waktu eksekusi:</b> 97.31680000014603ms  <b>Tick count:</b> 3992  <b>Node diperiksa:</b> 4202  <b>Banyak pencarian:</b> 34152  <b>Branching factor:</b> 12192611694335938  <b>Langkah:</b> 0 / 44  <b>Step:</b>            0 P+1 B+1 F-3 G-1 H-2 L-1 J+1            E+1 C+3 I-1 M+1 B+3 P-3 I-2 C-3 E-2            D-3 E2 H+3 L1 C1 G1 F+3 I-1 P+1            B-4 P1 D1 M1 H1 F2 L1 H3 C+2            I+3 P1 B1 F1 G1 C3 E1 I-1 L+3            P+3         </p>											

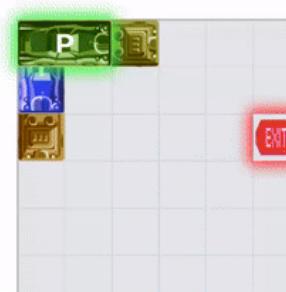
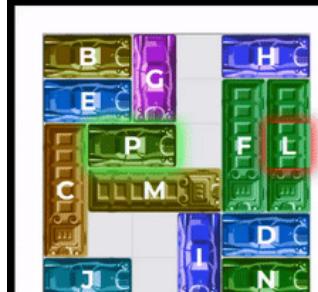
### 4.35. Jam-35

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-35</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>43</td><td>10</td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> L+1 B+1 G-1 D-3 J+1            H+1 C-1 I-2 E+3 I+2 G+1            B-1 L-1 C+3 H-1 J-1 F+3            P+1 H-3 P-1 F-3 C-3 L+1            B+1 G-1 I-2 E-4 D+3 I+2            G+1 B-1 L-1 C+3 F+3 B-2            G-1 P+3 H+1 B-1 F-3 C-1            L+3 P+1         </td><td>           .. FBBL            .. FG.L            PPFG.LK            HCCC..            HEEID.            JJ.ID.         </td></tr> </tbody> </table>	Nama	:	Jam-35	6 6	Banyak Step Optimal	:	43	10	Step Optimal	:	$\emptyset$ L+1 B+1 G-1 D-3 J+1 H+1 C-1 I-2 E+3 I+2 G+1 B-1 L-1 C+3 H-1 J-1 F+3 P+1 H-3 P-1 F-3 C-3 L+1 B+1 G-1 I-2 E-4 D+3 I+2 G+1 B-1 L-1 C+3 F+3 B-2 G-1 P+3 H+1 B-1 F-3 C-1 L+3 P+1	.. FBBL .. FG.L PPFG.LK HCCC.. HEEID. JJ.ID.				
Nama	:	Jam-35	6 6													
Banyak Step Optimal	:	43	10													
Step Optimal	:	$\emptyset$ L+1 B+1 G-1 D-3 J+1 H+1 C-1 I-2 E+3 I+2 G+1 B-1 L-1 C+3 H-1 J-1 F+3 P+1 H-3 P-1 F-3 C-3 L+1 B+1 G-1 I-2 E-4 D+3 I+2 G+1 B-1 L-1 C+3 F+3 B-2 G-1 P+3 H+1 B-1 F-3 C-1 L+3 P+1	.. FBBL .. FG.L PPFG.LK HCCC.. HEEID. JJ.ID.													
<p style="text-align: center;">UCS</p>  <p> <b>Waktu eksekusi:</b> 38.97429999988526ms  <b>Tick count:</b> 3936  <b>Node diperiksa:</b> 4075  <b>Banyak pencarian:</b> 33986  <b>Branching factor:</b> 1.2540283203125  <b>Langkah:</b> 43 / 43       </p> <p> <b>Step:</b>  <math>\emptyset</math> L+1 B+1 G-1 D-3 J+1 H+1 C-1 I-2            E+3 H-2 G-1 B-1 C-3 H-1 J-2 F+3            P+1 H-3 P-1 F-3 C-3 L+1 B+1 G-1 I-2 E-4 H-2 G-1 B-1 C-3 F+3 B-2            G-1 P+3 H-1 B-1 F-3 C-1 L+3 P+1       </p>	<p style="text-align: center;">GBFS Car Distance</p>  <p> <b>Waktu eksekusi:</b> 69.40029999986291ms  <b>Tick count:</b> 3569  <b>Node diperiksa:</b> 3708  <b>Banyak pencarian:</b> 31558  <b>Branching factor:</b> 1.19032958984375  <b>Langkah:</b> 49 / 49       </p> <p> <b>Step:</b>  <math>\emptyset</math> C-2 F+1 B-3 F-1 G-1 J+1 H+1            C-3 I-2 D-3 E-3 J-3 H-2 C-3 F-3 P+2            B+1 H-4 P-1 G-1 B-2 P-1 F-3 C-3 L+1            B+1 G-1 I-2 E-4 H-2 G-1 B-1 L-1 C-3            F-3 P+1 B-2 G-1 P+1 D-1 P+1 H+1 B-1            F-3 C-3 H-1 L-3 P+1       </p>	<p style="text-align: center;">GBFS Car Blocked</p>  <p> <b>Waktu eksekusi:</b> 45.72369999997318ms  <b>Tick count:</b> 3134  <b>Node diperiksa:</b> 3563  <b>Banyak pencarian:</b> 27219  <b>Branching factor:</b> 1.14643375  <b>Langkah:</b> 0 / 59       </p> <p> <b>Step:</b>  <math>\emptyset</math> C-2 F+1 B-2 G-1 B-1 F-1 C-1            L-3 I-3 H-4 C-2 D-4 L-2 E-3 H-1            L-1 I-3 H-3 C-3 F-3 B-1 P+1 H-4 P-1            G-1 D-1 B-2 F-3 C-1 L-1 B-1 G-1 C-2            L-2 E-2 D-2 E-1 H-1 J-1 L-2 E-1 J-3 H-1            D-1 G-1 B-1 L-3 C-3 F-3 B-2 G-1 P-3            H-1 B-1 F-3 G-1 L-3 P+1       </p>														
<p style="text-align: center;">A* Car Distance</p>  <p> <b>Waktu eksekusi:</b> 47.34019999951124ms  <b>Tick count:</b> 3912  <b>Node diperiksa:</b> 4002  <b>Banyak pencarian:</b> 33773  <b>Branching factor:</b> 1.289178466796875  <b>Langkah:</b> 44 / 44       </p> <p> <b>Step:</b>  <math>\emptyset</math> L+1 B+1 G-1 D-3 J+1 H+1 C-1 I-2            E+3 H-2 G-1 B-1 L-1 C-3 F+3 P+1            H-4 P-1 F-3 C-3 L-1 B+1 G-1 I-2 E-4            I-2 G-1 B-1 L-1 C-5 F+3 B-2 G-1 D-1            P+3 H-1 B-1 F-3 C-3 I-1 J-1 L+3 P+1       </p>	<p style="text-align: center;">A* Car Blocked</p>  <p> <b>Waktu eksekusi:</b> 30.517300000414252ms  <b>Tick count:</b> 3812  <b>Node diperiksa:</b> 3906  <b>Banyak pencarian:</b> 33104  <b>Branching factor:</b> 1.2067031860351562  <b>Langkah:</b> 0 / 46       </p> <p> <b>Step:</b>  <math>\emptyset</math> C-2 F+1 B-3 G-1 F-1 J+1 H+1            C-3 D-4 I-2 E-3 H-2 C-3 H-1 J-3 F-3            B+1 P+1 G-1 I-2 E-4 D-3 H-2 G-1 B-1            L-1 B-3 F-3 B-2 G-1 P+3 H-1 B-1 F-3            C-3 L-3 P+1       </p>	<p style="text-align: center;">A* Car Blocked Recursive</p>  <p> <b>Waktu eksekusi:</b> 45.146099999547005ms  <b>Tick count:</b> 3756  <b>Node diperiksa:</b> 3866  <b>Banyak pencarian:</b> 32791  <b>Branching factor:</b> 1.212066650390625  <b>Langkah:</b> 0 / 45       </p> <p> <b>Step:</b>  <math>\emptyset</math> C-2 F+1 B-3 G-1 F-1 J+1 H+1            C-3 D-4 I-2 E-3 H-2 C-3 H-1 J-3 F-3            P+1 G-1 D-1 B-3 H-3 P-1 F-3 C-3 L-1            B+1 G-1 I-2 E-4 H-2 D-3 G-1 B-1 L-1            C-3 F-3 B-2 G-1 P+3 H-1 B-1 F-3 C-3            L-3 P+1       </p>														

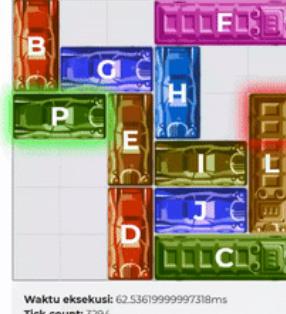
#### 4.36. Jam-36

<p>Nama : Jam-36          Banyak Step Optimal : 44          Step Optimal :</p> <pre>         φ H+1 P+1 E+1 M+1 F+1         L-1 B-1 C-1 M+2 I-3 M-1         C+1 B+1 L+1 F-1 M-2 E-1         J+4 E+1 M+1 F+1 L-1 B-1         C-1 M+2 G+3 I+3 H-2 P-2         M-1 C+1 B+1 L+1 F-1 M-2         E-3 M+1 F+1 L-1 E-1 D-1         J-1 C+2 P+3       </pre>	<p>6 6          11  <b>FLLLBB</b>  <b>FGHH.C</b>  <b>FGPP.CK</b>  <b>MMME.C</b>  <b>..IEDD</b>  <b>JJI ...</b></p>	
<p><b>UCS</b></p>  <p>Waktu eksekusi: 16.8448999999909103ms          Tick count: 2554          Node diperiksa: 2825          Banyak pencarian: 2222          Branching factor: 120599365234375          Langkah: 0 / 44</p> <p>Step:</p> <pre>         φ H+1 P+1 E+1 M+1 F+1 L-1 B-1         C-1 M+2 I-3 M-1 C+1 B+1 L-1 F-1 M-2         E-1 J+4 E+1 M+1 F+1 L-1 B-1 C-1 M+2         G+3 I+3 H-2 P-2 M-1 C+1 B+1 L-1 F-1         M-2 E-3 M+1 F+1 L-1 E-1 D-1 J-1 C+2         P+3       </pre>	<p><b>GBFS Car Distance</b></p>  <p>Waktu eksekusi: 12.956000000238479ms          Tick count: 1788          Node diperiksa: 2162          Banyak pencarian: 15823          Branching factor: 11902084350585938          Langkah: 0 / 45</p> <p>Step:</p> <pre>         φ P+1 H+1 E+1 M+1 F+1 L-1 B-1         C-1 M+2 I-3 M-1 C+1 B+1 L-1 F-1 M-2         E-1 J+4 E+1 M+1 F+1 L-1 B-1 C-1 M+2         G+3 I+3 H-2 M-1 C+1 B+1 L-1 F-1 M-2         P-2 E-3 M+1 F+1 L-1 E-1 P+2 D-1 J-1         C+2 P+3       </pre>	<p><b>GBFS Car Blocked</b></p>  <p>Waktu eksekusi: 31.970199999399483ms          Tick count: 1745          Node diperiksa: 1849          Banyak pencarian: 15372          Branching factor: 11842727661132812          Langkah: 0 / 46</p> <p>Step:</p> <pre>         φ H+1 P+1 E+1 M+1 F+1 L-1 B-1         C-1 M+2 I-3 M-1 C+1 B+1 L-1 F-1 M-2         E-1 J+4 E+1 M+1 F+1 L-1 B-1 C-1 M+2         G+3 I+3 P-2 M-1 C+1 B+1 L-1 F-1 M-2         E-1 J-1 E-1 D-1 C-2 H-1 E-1 M+1 F+1         L-1 E-1 P+3       </pre>
<p><b>A* Car Distance</b></p>  <p>Waktu eksekusi: 57.07429999951273ms          Tick count: 2254          Node diperiksa: 2581          Banyak pencarian: 19728          Branching factor: 11968154907226562          Langkah: 0 / 45</p> <p>Step:</p> <pre>         φ P+1 H+1 E+1 M+1 F+1 L-1 B-1         C-1 M+2 I-3 M-1 C+1 B+1 L-1 F-1 M-2         E-1 J+4 E+1 M+1 F+1 L-1 B-1 C-1 M+2         G+3 I+3 H-2 M-1 C+1 B+1 L-1 F-1 M-2         P-2 E-3 M+1 F+1 L-1 E-1 P+2 D-1 J-1         C+2 P+1       </pre>	<p><b>A* Car Blocked</b></p>  <p>Waktu eksekusi: 10.662399999797344ms          Tick count: 2169          Node diperiksa: 2506          Banyak pencarian: 18874          Branching factor: 12009811401367188          Langkah: 0 / 44</p> <p>Step:</p> <pre>         φ H+1 P+1 E+1 M+1 F+1 L-1 B-1         C-1 M+2 I-3 M-1 C+1 B+1 L-1 F-1 M-2         E-1 J+4 E+1 M+1 F+1 L-1 B-1 C-1 M+2         G+3 I+3 H-2 P-2 M-1 C+1 B+1 L-1 F-1         M-2 E-3 D-1 J-1 C-2 M+1 F+1 L-1 E-1         P+3       </pre>	<p><b>A* Car Blocked Recursive</b></p>  <p>Waktu eksekusi: 51.53610000014305ms          Tick count: 2003          Node diperiksa: 2337          Banyak pencarian: 17428          Branching factor: 11985244750976562          Langkah: 0 / 44</p> <p>Step:</p> <pre>         φ H+1 P+1 E+1 M+1 F+1 L-1 B-1         C-1 M+2 I-3 M-1 C+1 B+1 L-1 F-1 M-2         E-1 J+4 E+1 M+2 F+1 L-1 B-1 G+5 C-1         M+1 I+3 M-1 C+1 B+1 L-1 F-1 H-2 P-2         M-2 E-3 D-1 J-1 C-2 M+1 F+1 L-1 E-1         P+3       </pre>

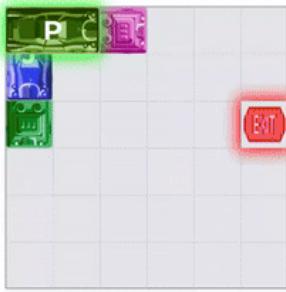
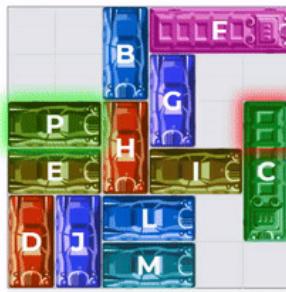
### 4.37. Jam-37

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-37</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>47</td><td>12</td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> J+1 C+1 P-1 G+1 H-2            F-1 L-1 M+2 G+2 E+1 P+1            C-1 J-1 G+1 M-2 F+1 H+1            B+1 C-2 M-1 I-3 M+1 C+1            B-1 H-1 F-1 M+2 G-1 J+1            C+2 E-1 P-1 G-2 M-2 F+1            H+1 G-1 P+1 C-1 D-2 J-1            N-2 L+3 H+1 I-1 F+2 P+3         </td><td>           BBG.HH            EEG.FL            CPP.FLK            CMMMFLL            C..IDD            JJ.INN         </td></tr> </tbody> </table>	Nama	:	Jam-37	6 6	Banyak Step Optimal	:	47	12	Step Optimal	:	$\emptyset$ J+1 C+1 P-1 G+1 H-2 F-1 L-1 M+2 G+2 E+1 P+1 C-1 J-1 G+1 M-2 F+1 H+1 B+1 C-2 M-1 I-3 M+1 C+1 B-1 H-1 F-1 M+2 G-1 J+1 C+2 E-1 P-1 G-2 M-2 F+1 H+1 G-1 P+1 C-1 D-2 J-1 N-2 L+3 H+1 I-1 F+2 P+3	BBG.HH EEG.FL CPP.FLK CMMMFLL C..IDD JJ.INN				
Nama	:	Jam-37	6 6													
Banyak Step Optimal	:	47	12													
Step Optimal	:	$\emptyset$ J+1 C+1 P-1 G+1 H-2 F-1 L-1 M+2 G+2 E+1 P+1 C-1 J-1 G+1 M-2 F+1 H+1 B+1 C-2 M-1 I-3 M+1 C+1 B-1 H-1 F-1 M+2 G-1 J+1 C+2 E-1 P-1 G-2 M-2 F+1 H+1 G-1 P+1 C-1 D-2 J-1 N-2 L+3 H+1 I-1 F+2 P+3	BBG.HH EEG.FL CPP.FLK CMMMFLL C..IDD JJ.INN													
<p style="text-align: center;">UCS</p>  <p>         Waktu eksekusi: 14.36560000001356ms          Tick count: 1942          Node diperiksa: 1950          Banyak pencarian: 15257          Branching factor: 1.79244995171875          Langkah: 0 / 47          Step: 0 J+1 C+1 P-1 G+1 H-2 F-1 L-1            M+2 G+2 E+1 C-1 J-1 G-1 M-2 F+1            H-1 B+1 C-2 M-1 I-3 M+1 C-1 H-1            F-1 M+2 G-1 J-1 C+2 E-1 P-1 G-2 M-2            F+1 H-1 G-1 P+1 C-1 D-2 I-1 N-2 L+3            I-1 F+2 P+3       </p>	<p style="text-align: center;">GBFS Car Distance</p>  <p>         Waktu eksekusi: 5.7040000000177324ms          Tick count: 1072          Node diperiksa: 1387          Banyak pencarian: 7507          Branching factor: 1.122657758789062          Langkah: 0 / 58          Step: 0 P+1 H-1 L-1 C+1 P-1 G-1 H-1 F-1            M+2 G+2 E+1 C-1 J-1 G-1 M-2 F+1            L-1 H+2 B+1 C-2 M-1 I-4 D-1 N-1 L+2            M+1 C-2 B-1 E-1 H-2 F-1 L-3 M+2            G-1 C+1 P-1 G-2 M-1 L-5 M-1 F+1            H+1 G+1 P+1 H-1 P+1 G-2 M-2 D-2            J-1 N-2 F+2 P+2 L-2 P+1       </p>	<p style="text-align: center;">GBFS Car Blocked</p>  <p>         Waktu eksekusi: 3.1371999997645617ms          Tick count: 854          Node diperiksa: 1032          Banyak pencarian: 5712          Branching factor: 1.1244354248046875          Langkah: 0 / 55          Step: 0 H-1 L-1 C+1 P-1 G-1 M-2 F+1            M+2 G+2 E+1 P+1 C-1 J-1 G-1 M-2 F+1            L-1 H+2 B+1 C-2 M-1 I-4 D-1 N-1 L+2            M+1 C-2 B-1 E-1 H-2 F-1 L-3 M+2            G-1 C+1 P-1 G-2 M-1 L-5 M-1 F+1            H+1 G+1 H-1 P+1 G-2 D-1 J-1 N-1            F+2 P+3       </p>														
<p style="text-align: center;">A* Car Distance</p>  <p>         Waktu eksekusi: 23.989200000065118ms          Tick count: 1937          Node diperiksa: 1953          Banyak pencarian: 15210          Branching factor: 1.745758056640625          Langkah: 0 / 48          Step: 0 J+1 C+1 P-1 G+1 H-2 F-1 L-1            M+2 G+2 P+2 E+1 C-1 J-1 G-1 M-2 F+1            H-1 B+1 C-2 M-1 P-1 I-3 M+1 C-1 H-1            H-1 F-1 M+2 G-1 J-1 C+2 E-1 P-1 G-2            M-2 F+1 L-1 H-2 G-1 I-1 P+2 C-1 D-2            J-1 N-2 F+2 L-2 P+3       </p>	<p style="text-align: center;">A* Car Blocked</p>  <p>         Waktu eksekusi: 21.52850000000149ms          Tick count: 1864          Node diperiksa: 1936          Banyak pencarian: 14583          Branching factor: 1.779632568359375          Langkah: 47 / 47          Step: 0 H-1 C+1 P-1 G-1 H-2 F-1 L-1            M+2 G+2 E+1 P+1 C-1 J-1 G-1 M-2 F+1 H+1            H-1 B+1 C-2 M-1 I-3 M+1 C-1 H-1            F-1 M+2 G-1 J-1 C+2 E-1 P-1 N-3 G-2            M-2 D-2 L-3 F+1 H-2 G-1 I-1 P+1 C-1            J-1 N-3 F+2 P+3       </p>	<p style="text-align: center;">A* Car Blocked Recursive</p>  <p>         Waktu eksekusi: 23.358899999409914ms          Tick count: 1893          Node diperiksa: 1941          Banyak pencarian: 14873          Branching factor: 1.1785202026367188          Langkah: 0 / 47          Step: 0 J+1 C+1 P-1 G+1 H-2 F-1 L-1            M+2 G+2 P+1 C-1 J-1 G-1 M-2 F+1 H+1            B+1 E+1 C-2 M-1 I-3 M+1 C-1 H-1            E-1 F-1 M+2 G-1 J-1 C+1 P-1 G-2 M-2            F+1 D-2 N-1 L-3 H-2 I-1 G-1 P+1 C-1            J-1 N-1 F+2 P+3       </p>														

### 4.38. Jam-38

<table border="1"> <tbody> <tr> <td>Nama</td><td>:</td><td>Jam-38</td><td>6 6</td></tr> <tr> <td>Banyak Step Optimal</td><td>:</td><td>48</td><td>10</td></tr> <tr> <td>Step Optimal</td><td>:</td><td> <math>\emptyset</math> F-1 L-2 I+1 H+1 G+2 E-1  D-1 J+1 C-3 H+2 I-1 L+1 F+1  E-1 P+1 B+3 P-1 E+1 F-1 L-1  I+1 H-1 C+3 B+1 D+1 E+1 G-3  E-1 H-2 I-4 L+1 F+1 E+1 H+1  G+3 E-2 P+1 I+1 B-4 P-1 I-1  D-2 J-4 D+1 H+1 C-1 L+2 P+4 </td><td> B .. FFF  BGGH ..  PPEH.LK  .. EIIL  .. DJJL  .. DCCC </td></tr> </tbody> </table>	Nama	:	Jam-38	6 6	Banyak Step Optimal	:	48	10	Step Optimal	:	$\emptyset$ F-1 L-2 I+1 H+1 G+2 E-1 D-1 J+1 C-3 H+2 I-1 L+1 F+1 E-1 P+1 B+3 P-1 E+1 F-1 L-1 I+1 H-1 C+3 B+1 D+1 E+1 G-3 E-1 H-2 I-4 L+1 F+1 E+1 H+1 G+3 E-2 P+1 I+1 B-4 P-1 I-1 D-2 J-4 D+1 H+1 C-1 L+2 P+4	B .. FFF BGGH .. PPEH.LK .. EIIL .. DJJL .. DCCC	
Nama	:	Jam-38	6 6										
Banyak Step Optimal	:	48	10										
Step Optimal	:	$\emptyset$ F-1 L-2 I+1 H+1 G+2 E-1 D-1 J+1 C-3 H+2 I-1 L+1 F+1 E-1 P+1 B+3 P-1 E+1 F-1 L-1 I+1 H-1 C+3 B+1 D+1 E+1 G-3 E-1 H-2 I-4 L+1 F+1 E+1 H+1 G+3 E-2 P+1 I+1 B-4 P-1 I-1 D-2 J-4 D+1 H+1 C-1 L+2 P+4	B .. FFF BGGH .. PPEH.LK .. EIIL .. DJJL .. DCCC										
UCS	GBFS Car Distance	GBFS Car Blocked											
 <p> Waktu eksekusi: 15.153800000436604ms  Tick count: 3711  Node diperiksa: 3994  Banyak pencarian: 28991  Branching factor: 1.925125122070312  Langkah: 0 / 48  Step:  <ul style="list-style-type: none"> <li><math>\emptyset</math> F-1 L-2 I+1 H+1 G+2 E-1 D-1 J+1</li> <li>C-3 H+2 I-1 L+1 F+1 E-1 P+1 B+3 D-1</li> <li>E-1 F+1 L-1 H-1 C-3 B-1 D-1 E-1</li> <li>G-3 E-1 H-2 I-4 L+1 F+1 E-1 H+1 C-3</li> <li>E-2 P+1 H-1 B-4 P-1 I-1 D-2 J-4 D+1</li> <li>H+1 C-1 L-2 P+4</li> </ul> </p>	 <p> Waktu eksekusi: 15.356699999421835ms  Tick count: 3264  Node diperiksa: 3735  Banyak pencarian: 25493  Branching factor: 1.2171936033515625  Langkah: 69 / 69  Step:  <ul style="list-style-type: none"> <li><math>\emptyset</math> F-1 L-2 I+1 H+1 G+2 E-1</li> <li>C-3 H+2 I-1 L+1 F+1 E-1 P+1 B+3 D-1</li> <li>E-1 F+1 L-1 H-1 C-3 D-1 E-1 G-3</li> <li>D-1 C-3 H+1 I-1 L+1 F+1 E-1 P-3 E+1</li> <li>F-1 L-1 H-1 C-3 D-1 E-1 P-3 E+1</li> <li>E-1 D-1 H-1 C-3 D-1 E-1 P-3 E+1</li> <li>C-3 D-1 I-1 H-1 C-3 D-1 E-1 P-3 E+1</li> <li>E-2 H-1 B-4 H-1 P-2 H-1 P-1 D-2 J-4</li> <li>D-1 P-2 H-1 P-2 C-1 L-2 P-4</li> </ul> </p>	 <p> Waktu eksekusi: 22.138899999670684ms  Tick count: 1575  Node diperiksa: 2529  Banyak pencarian: 1101  Branching factor: 1.063461303710938  Langkah: 0 / 69  Step:  <ul style="list-style-type: none"> <li><math>\emptyset</math> F-1 L-2 I+1 H+1 G+2 E-1</li> <li>D-1 C-3 H+1 I-1 F+1 E-1 P-3 B+3</li> <li>E-1 F+1 L-1 H-1 C-3 D-1 E-1 G-3</li> <li>E-1 D-1 C-3 H+1 I-1 L-1 F+1 E-1 P-3</li> <li>E-1 F-3 L-1 H-1 C-3 D-1 I-3 H+3</li> <li>L-1 F+3 E-1 P-3 B+3 I-1 E-2 G-3 E-2</li> <li>P-2 H-1 B-3 D-1 C-1 H-1 J-1 L-2</li> <li>B-1 D-1 C-3 J-2 H-1 J-1 D-1 P-4</li> </ul> </p>											
A* Car Distance	A* Car Blocked	A* Car Blocked Recursive											
 <p> Waktu eksekusi: 41.444099999995424ms  Tick count: 3381  Node diperiksa: 3788  Banyak pencarian: 26448  Branching factor: 1.122793579101562  Langkah: 0 / 52  Step:  <ul style="list-style-type: none"> <li><math>\emptyset</math> F-1 L-2 I+1 H+1 G+2 E-1 D-1 J+1</li> <li>C-3 H+2 I-1 L+1 F+1 E-1 P+1 B+3 D-1</li> <li>E-1 F+1 L-1 H-1 C-3 D-1 E-1 G-3</li> <li>E-1 H-2 I-3 L+1 F+1 E-1 H+2 P+3 B+1</li> <li>I-1 E-2 G-3 E-2 H-1 B-4 I-1 P-3 H-1</li> <li>D-2 J-4 D-1 H-1 P-5 C-1 L-2 P-4</li> </ul> </p>	 <p> Waktu eksekusi: 62.536199999997318ms  Tick count: 3294  Node diperiksa: 3577  Banyak pencarian: 25550  Branching factor: 1.1842880249023438  Langkah: 0 / 49  Step:  <ul style="list-style-type: none"> <li><math>\emptyset</math> F-1 L-2 I+1 H+1 G+2 E-1</li> <li>D-1 C-3 H-1 I-1 F-1 E-1 P-1 B-3</li> <li>P-1 E-1 F-3 L-1 H-1 C-3 B-1 D-1</li> <li>E-1 G-3 E-1 H-3 I-4 H-3 L-1 F-3 E-1</li> <li>G-3 E-2 P-1 H-1 B-4 P-1 I-1 D-1</li> <li>J-4 D-1 H-1 C-1 L-2 P-4</li> </ul> </p>	 <p> Waktu eksekusi: 40.326999999958277ms  Tick count: 3425  Node diperiksa: 3704  Banyak pencarian: 26587  Branching factor: 1.1653713989257812  Langkah: 0 / 49  Step:  <ul style="list-style-type: none"> <li><math>\emptyset</math> F-1 L-2 I+1 H+1 G+2 E-1 D-1</li> <li>C-3 I-1 H+2 I-1 L-1 F-1 E-1 P-1 B-3</li> <li>P-1 E-1 F-3 L-1 H-1 C-3 D-1 E-1</li> <li>G-3 H-3 I-1 E-1 B-1 I-4 L-2 H-2 F-3</li> <li>E-1 G-3 E-2 P-1 H-1 B-4 P-1 I-1 D-2</li> <li>C-1 L-1 J-3 D-2 H-1 P-4</li> </ul> </p>											

### 4.39. Jam-39

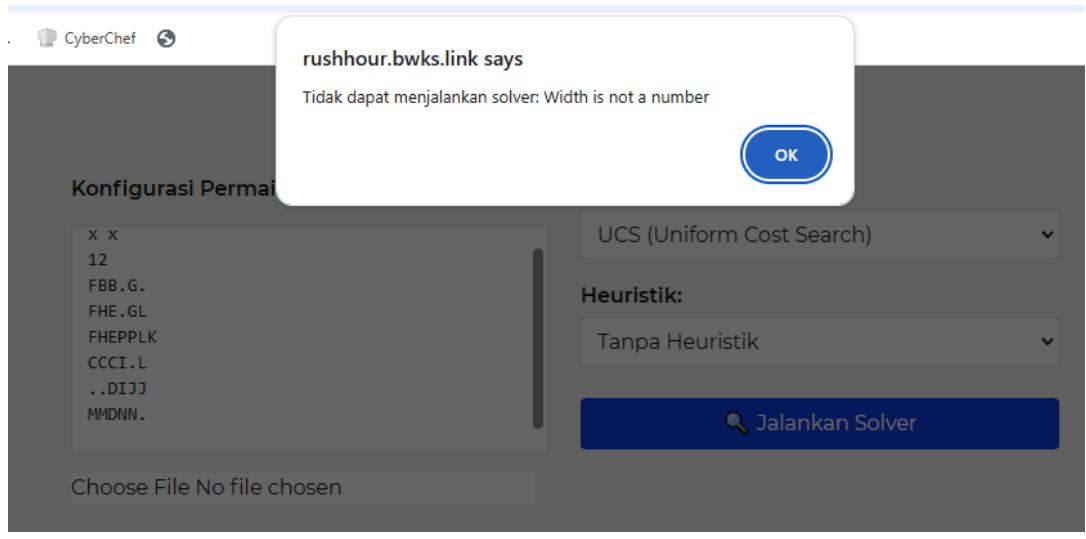
<p>Nama : Jam-39</p> <p>Banyak Step Optimal : 50</p> <p>Step Optimal :</p> <pre> <math>\emptyset</math> C-1 L+2 M+2 H+2 B+1 F-1 C-1 I+1 G+3 I-1 C+1 F+1 B-1 P+3 B+1 F-1 C-1 I+1 E+2 D-4 J-4 E-2 I-1 C+1 F+1 B-1 P-3 B+1 F-1 C-1 I+1 G-3 I-1 C+1 F+1 B-1 H-2 L-4 H+1 B+1 F-1 C-1 I+1 M-3 G+3 I-1 C+3 F+1 B-1 P+4 </pre>	<p>6 6 11 .. BFFF .. BG .. PPHG.CK EEHIIC DJLL.C DJMM ..</p>	
<p>UCS</p>  <p>Waktu eksekusi: 10.846199999563396ms Tick count: 3600 Node diperiksa: 3709 Banyak pencarian: 24859 Branching factor: 1.1790695190429688 Langkah: 0 / 50</p> <p>Step: <math>\emptyset</math> C-1 L+2 M+2 H+2 B+1 F-1 C-1 I+1 G+3 I-1 C+1 F+1 B-1 P+3 B+1 F-1 C-1 I+1 E+2 D-4 J-4 E-2 I-1 C+1 F+1 B-1 P-3 B+1 F-1 C-1 I+1 G-3 I-1 C+1 F+1 B-1 H-2 L-4 H+1 B+1 F-1 C-1 I+1 M-3 G+3 I-1 C+3 F+1 B-1 P+4</p>	<p>GBFS Car Distance</p>  <p>Waktu eksekusi: 15.050800000317395ms Tick count: 3984 Node diperiksa: 4165 Banyak pencarian: 27561 Branching factor: 1.13739013671875 Langkah: 0 / 63</p> <p>Step: <math>\emptyset</math> L+2 H+1 P+3 B+1 F-1 B+1 F-1 C-1 I+1 E+2 D-4 P+3 B+1 F-1 C-1 H+1 E+2 D-4 E-2 I-1 C+1 F+1 B-1 P-3 H+1 G-3 I-1 C+1 F+1 B-1 H-2 L-4 B+1 F-1 C-1 H+1 G-3 I-1 C+1 F+1 P+3 B+1 F-1 C-1 H+1 G-3 I-1 C+1 F+1</p>	<p>GBFS Car Blocked</p>  <p>Waktu eksekusi: 24.140599999576807ms Tick count: 3422 Node diperiksa: 3498 Banyak pencarian: 23760 Branching factor: 1.12539453125 Langkah: 65 / 65</p> <p>Step: <math>\emptyset</math> C-1 L+2 H+1 B+1 F-1 G-1 C-3 H+1 L+2 G+4 I-1 C+1 F+2 B-1 P+3 B+1 F-1 C-1 H+1 E+2 I-4 E-2 H-1 G-1 M-3 G+1 I-1 C+1 F+1 B-1 P-3 B+1 F-1 C-1 H-2 G-3 I-1 L-1 C+2 F+1 B-1 P+3 M-2 H-3 E-1 D-4 P-1 E-1 H-2 L-3 H+1 B+1 F-1 C-1 G-1 M-2 G+1 I-1 C-3 F+1 B-1 P+4</p>
<p>A* Car Distance</p>  <p>Waktu eksekusi: 61.39369999989867ms Tick count: 3732 Node diperiksa: 3854 Banyak pencarian: 25787 Branching factor: 1.1800460815429688 Langkah: 0 / 50</p> <p>Step: <math>\emptyset</math> C-1 L+2 M+2 H+2 B+1 F-1 C-1 I+1 G+3 I-1 C+1 F+1 B-1 P+3 B+1 F-1 C-1 H+1 E+2 D-4 J-4 E-2 I-1 C+1 F+1 B-1 P-3 B+1 F-1 C-1 I+1 G-3 I-1 C+1 F+1 B-1 H-2 L-4 H+1 B+1 F-1 C-1 I+1 M-3 G+3 I-1 C+3 F+1 B-1 P+4</p>	<p>A* Car Blocked</p>  <p>Waktu eksekusi: 47.36060000024736ms Tick count: 3542 Node diperiksa: 3605 Banyak pencarian: 24457 Branching factor: 1.1590652465820312 Langkah: 0 / 55</p> <p>Step: <math>\emptyset</math> C-1 L+2 H+1 P+3 B+1 F-1 H+1 L+2 G+4 I-1 C+1 F+3 B-1 P+3 B+1 F-1 C-1 H+1 E+2 D-4 I-4 E-2 I-1 C+1 F+1 B-1 P-3 H-1 B-1 F-1 C-1 H-1 G-3 I-1 L-1 C+3 F+1 B-1 H-1 L-3 H+1 B-1 F-1 C-1 G-1 M-2 G+1 I-1 B-1 P+4</p>	<p>A* Car Blocked Recursive</p>  <p>Waktu eksekusi: 40.68350000027567ms Tick count: 3525 Node diperiksa: 3564 Banyak pencarian: 24345 Branching factor: 1.16253662109375 Langkah: 0 / 54</p> <p>Step: <math>\emptyset</math> C-1 L+2 M+2 H+2 E-1 B+1 F-1 C-1 I+1 G+3 I-1 C+1 F+1 B-1 P+3 B+1 F-1 C-1 D-4 H-1 E-1 I-1 C+1 F+1 B-1 P-3 B+1 F-1 C-1 H-1 G-3 I-1 M-1 I-1 C-3 F+1 B-1 E-1 H-2 L-3 H+1 M-2 B-1 F-1 C-2 H-1 G-2 I-1 C-3 F+1 B-1 P+4</p>

#### 4.40. Jam-40

<p>Nama : Jam-40</p> <p>Banyak Step Optimal : 51</p> <p>Step Optimal :</p> <pre> <math>\emptyset</math> L-1 N+1 I+1 C+3 H+2 D-1 M+1 F+3 B-1 E-1 P-3 E+1 G+1 B+3 E-1 P+2 F-3 H-3 P-1 M-1 D+1 C-3 L+1 B+1 I-4 P+1 C+2 H+3 P-1 I+1 B-1 L-1 C+1 D-1 J-1 M+1 F+3 P-1 E+1 B-3 E-1 I-1 G-1 P+1 F-1 M-1 D+1 C-1 N-1 L+3 P+3 </pre>	<p>6 6</p> <p>12</p> <p>FBB.G.</p> <p>FHE.GL</p> <p>FHEPPLK</p> <p>CCCI.L</p> <p>..DIJJ</p> <p>MMDNN.</p>	
<p>UCS</p> <p>Waktu eksekusi: 26.086000000012666ms Tick count: 3024 Node diperiksa: 3202 Banyak pencarian: 24451 Branching factor: 1.743698120117188 Langkah: 51 / 51</p> <p>Step:</p> <pre> <math>\emptyset</math> L-1 N+1 I+1 C+3 H+2 D-1 M+1 F-3 B-1 E-1 P-1 G-1 H-1 C-1 D-1 F-3 H-3 P-1 M-1 D-1 C-3 L-1 B-1 I-4 P-1 C-2 H-3 P-1 H-1 B-1 L-1 C-1 D-1 J-1 M-1 F-3 P-1 E-1 B-3 E-1 G-1 F-1 F-3 M-3 D-1 C-1 N-1 L-3 P+3 </pre>	<p>GBFS Car Distance</p> <p>Waktu eksekusi: 13.3161999999302626ms Tick count: 2296 Node diperiksa: 2772 Banyak pencarian: 18484 Branching factor: 1.55853271484375 Langkah: 0 / 54</p> <p>Step:</p> <pre> <math>\emptyset</math> L-1 N+1 I+1 C+3 H+2 D-1 M+1 F-3 B-1 E-1 P-1 G-1 H-1 C-1 D-1 F-3 H-3 P-1 M-1 D-1 C-3 L-1 B-1 D-1 P-2 F-3 H-3 M-1 D-1 C-3 L-1 B-1 D-1 L-2 C-2 H-3 P-1 H-1 B-1 L-1 C-1 D-1 M-1 F-3 P-1 E-1 B-3 D-1 P-1 P-1 G-1 F-1 J-1 M-1 D-1 C-1 N-1 L-3 P-1 </pre>	<p>GBFS Car Blocked</p> <p>Waktu eksekusi: 14.5959999999903142ms Tick count: 1577 Node diperiksa: 1981 Banyak pencarian: 11900 Branching factor: 1.4515686603515625 Langkah: 0 / 54</p> <p>Step:</p> <pre> <math>\emptyset</math> L-1 N+1 I+1 C+3 H+2 D-1 M+1 F-3 B-1 E-1 P-1 G-1 H-1 C-1 D-1 F-3 H-3 P-1 M-1 D-1 C-3 L-1 B-1 I-4 J-1 N-1 L-2 P-1 C-2 H-3 P-1 E-1 B-1 L-3 C-1 D-1 M-1 F-3 P-1 E-1 B-1 G-1 B-1 L-1 B-1 P-1 F-1 J-1 M-1 D-1 C-1 L-3 P-1 </pre>
<p>A* Car Distance</p> <p>Waktu eksekusi: 56.7554999999702ms Tick count: 2912 Node diperiksa: 3771 Banyak pencarian: 23622 Branching factor: 1.734695434570312 Langkah: 0 / 51</p> <p>Step:</p> <pre> <math>\emptyset</math> L-1 N+1 I+1 C+3 H+2 D-1 M+1 F-3 B-1 E-1 P-1 G-1 H-1 C-1 D-1 F-3 H-3 M-1 D-1 C-3 L-1 B-1 P-1 I-4 P-1 C-2 H-3 P-1 H-1 B-1 L-1 C-1 D-1 M-1 F-3 P-1 E-1 B-3 E-1 I-1 G-1 P+3 F-1 J-1 M-1 D-1 C-1 N-1 L-3 P+3 </pre>	<p>A* Car Blocked</p> <p>Waktu eksekusi: 46.1489999999277294ms Tick count: 2704 Node diperiksa: 3032 Banyak pencarian: 21892 Branching factor: 1.674880981445312 Langkah: 0 / 52</p> <p>Step:</p> <pre> <math>\emptyset</math> L-1 N+1 I+1 C+3 H+2 D-1 M+1 F-3 B-1 E-1 P-1 G-1 H-1 C-1 D-1 F-3 H-3 P-1 M-1 D-1 C-3 L-1 B-1 I-4 J-1 N-1 L-2 P-1 C-2 H-3 P-1 H-1 B-1 L-3 C-1 D-1 M-1 F-3 P-1 E-1 B-1 G-1 H-1 G-1 P-1 F-1 M-1 D-1 C-1 L-3 P-3 </pre>	<p>A* Car Blocked Recursive</p> <p>Waktu eksekusi: 27.233400000025928ms Tick count: 2240 Node diperiksa: 2635 Banyak pencarian: 17493 Branching factor: 1.6717965698242188 Langkah: 0 / 52</p> <p>Step:</p> <pre> <math>\emptyset</math> L-1 N+1 I+1 C+3 H+2 D-1 M+1 F-3 B-1 E-1 P-1 G-1 H-1 C-1 D-1 F-3 H-3 P-1 M-1 D-1 C-3 L-1 B-1 I-4 N-1 J-1 L-2 P-1 C-2 H-3 P-1 H-1 B-1 L-3 C-1 D-1 M-1 F-3 P-1 E-1 B-1 G-1 L-3 G-1 P-1 F-1 M-1 D-1 C-1 L-3 P-3 </pre>

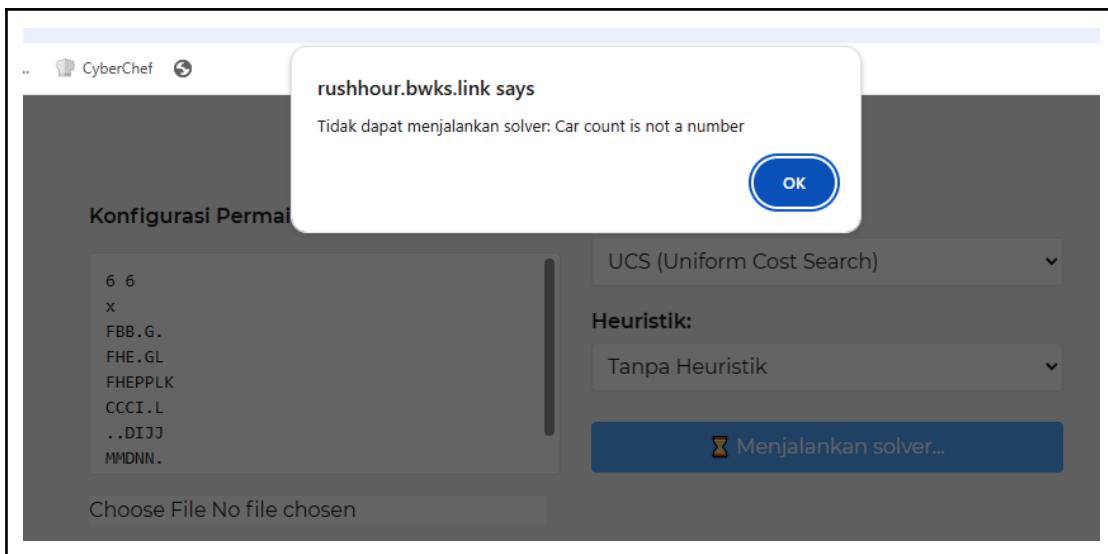
#### 4.41. Invalid Input

Alasan	:	Tinggi/Lebar bukan angka	x x 12 FBB.G. FHE.GL FHEPPLK CCCI.L .DIJJ MMDNN.
--------	---	--------------------------	---

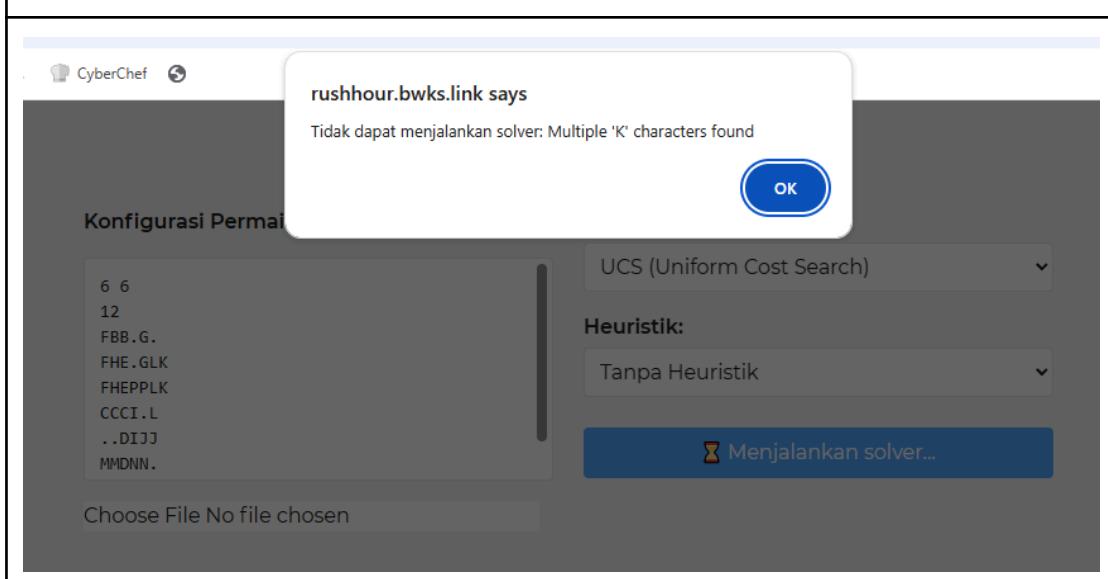


The screenshot shows the CyberChef interface. A modal dialog box is open, displaying the error message "rushhour.bwks.link says Tidak dapat menjalankan solver: Width is not a number". Below the modal, the main interface shows a configuration panel for a Rush Hour puzzle. The configuration panel includes fields for "Konfigurasi Permainan" (containing piece definitions like "x x", "12", "FBB.G.", etc.), search algorithms ("UCS (Uniform Cost Search)"), heuristics ("Tanpa Heuristik"), and a "Jalankan Solver" button. A file input field at the bottom is set to "Choose File No file chosen".

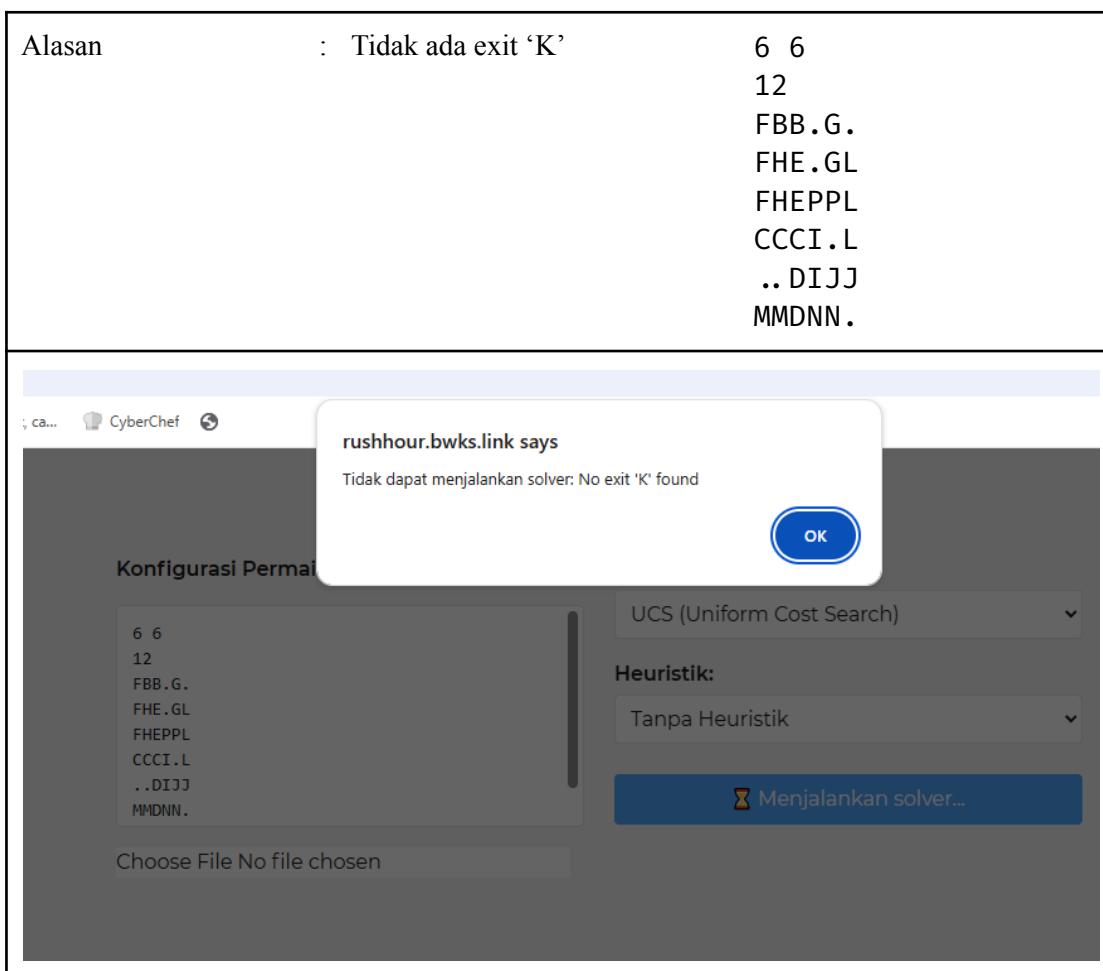
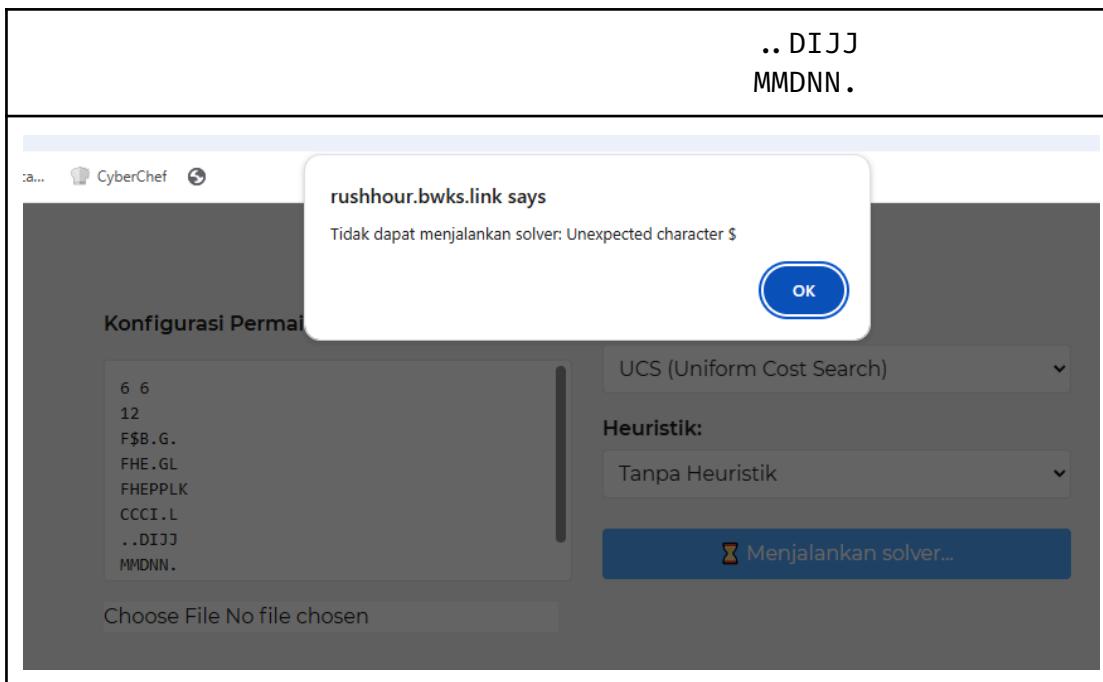
Alasan	:	Jumlah <i>piece</i> bukan angka	6 6 X FBB.G. FHE.GL FHEPPLK CCCI.L .DIJJ MMDNN.
--------	---	---------------------------------	--



Alasan	: Terdapat exit 'K' lebih dari 1	6_6 12 FBB.G. FHE.GLK FHEPPLK CCCI.L ..DIJJ MMDNN.
--------	----------------------------------	---



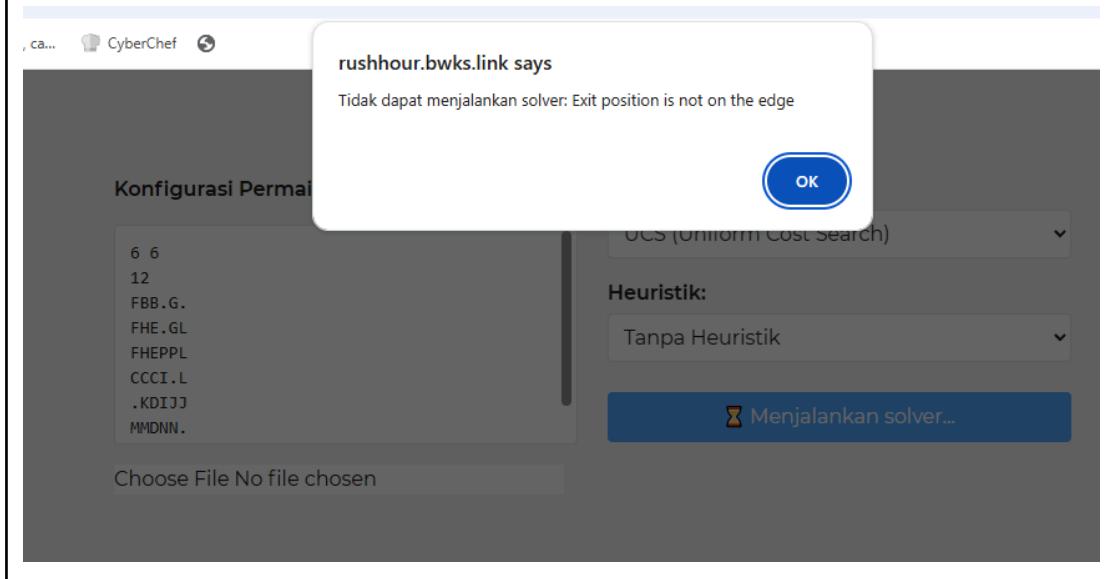
Alasan	: Karakter tidak dikenal/invalid	6_6 12 F\$B.G. FHE.GL FHEPPLK CCCI.L
--------	----------------------------------	---



Alasan	: Exit tidak berada di pinggir	6 6
--------	--------------------------------	-----

papan

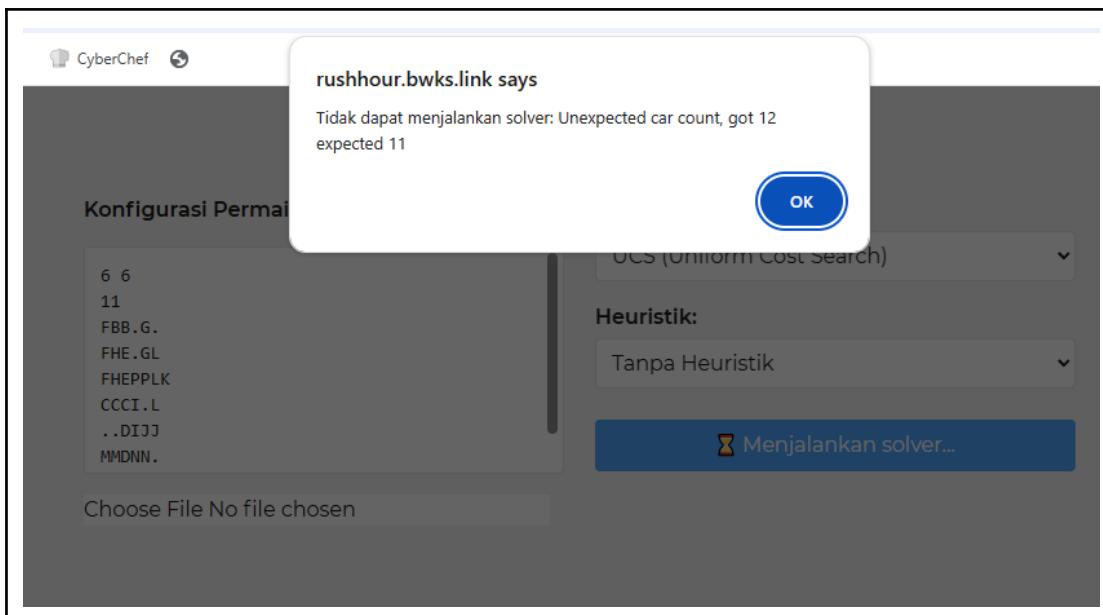
12  
FBB.G.  
FHE.GL  
FHEPPL  
CCCI.L  
.KDIJJ  
MMDNN.



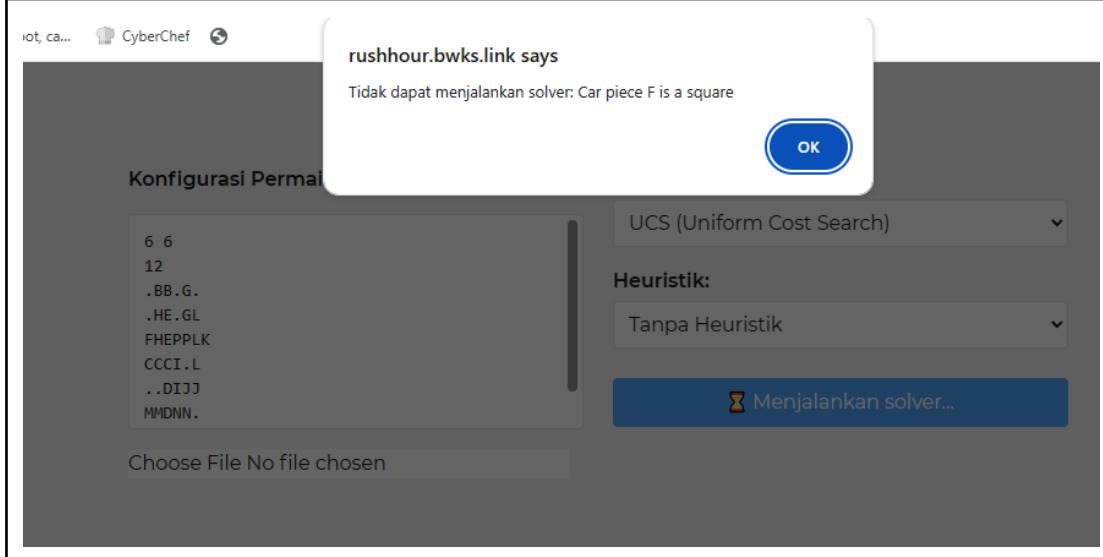
Alasan

: Jumlah piece tidak sesuai

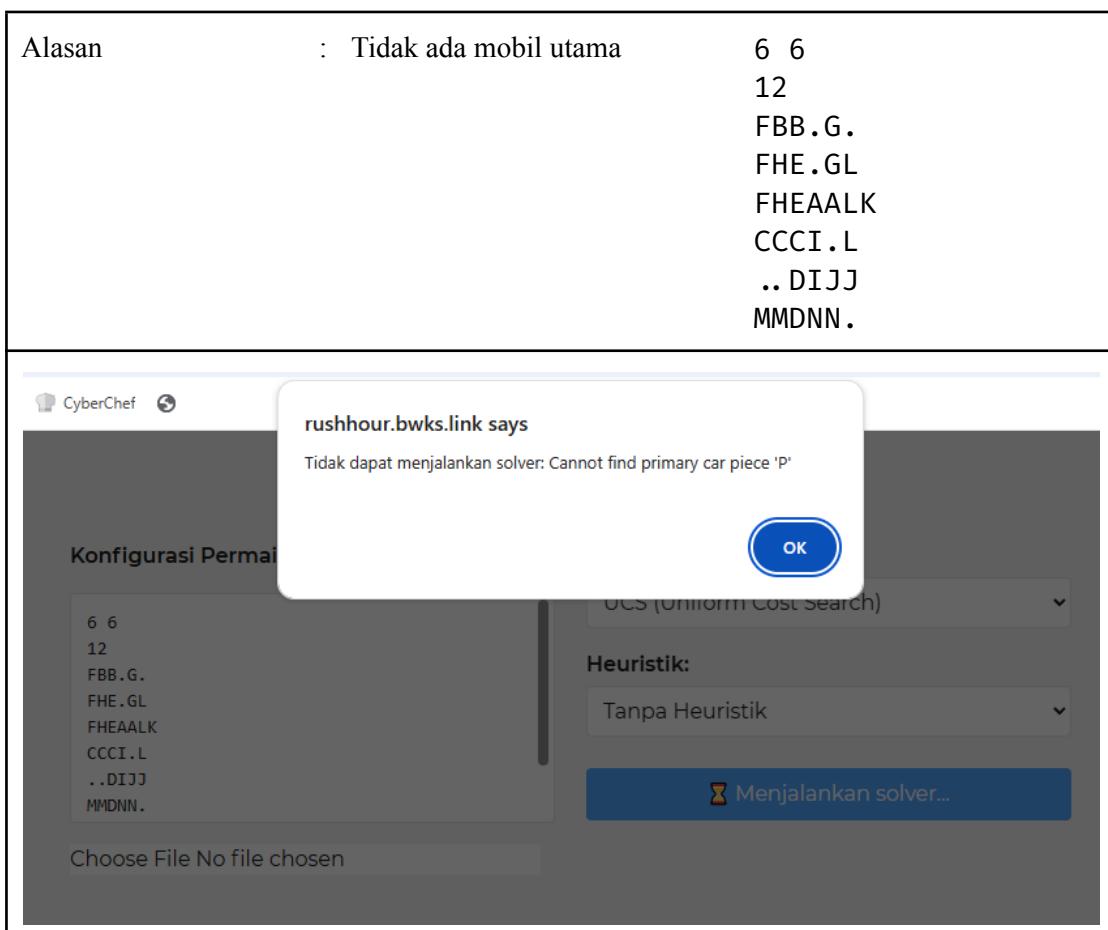
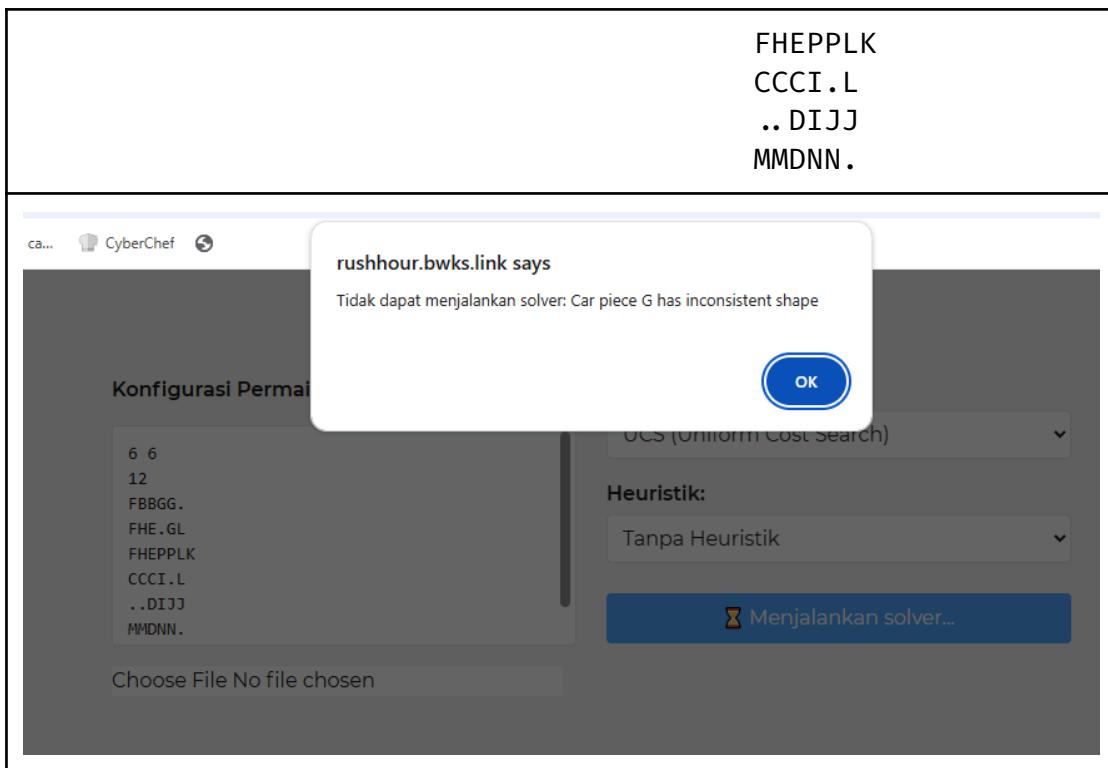
6 6  
11  
FBB.G.  
FHE.GL  
FHEPPLK  
CCCI.L  
.DIJJ  
MMDNN.



Alasan	: Piece dengan width 1 dan height 1	6 6 12 .BB.G. .HE.GL FHEPPLK CCCI.L .DIJJ MMDNN.
--------	-------------------------------------	---



Alasan	: Ukuran piece tidak konsisten	6 6 12 FBBGG. FHE.GL
--------	--------------------------------	-------------------------------



## BAB V ANALISIS

### 5.1. Analisis

Algoritma *Uniform Cost Search* (UCS) menjamin solusi yang optimal. Namun, kelemahannya adalah eksplorasi state yang sangat luas, sehingga secara teori membutuhkan waktu komputasi yang besar. Meskipun telah dilakukan optimasi melalui `QueueSolverUniform`—yang menghindari revisit state dengan `hashCode` yang sama selama heuristik bersifat konsisten—performa waktu yang dihasilkan dapat menyerupai A\* secara teknis, meskipun pendekatannya tetap murni UCS. Dalam UCS, fungsi evaluasi adalah  $g(n) = \text{step\_cost}(n)$  dengan  $h(n) = 0$ , yang berarti tidak ada heuristik yang digunakan.

*Greedy Best First Search* (GBFS) memiliki keunggulan dalam kecepatan eksekusi karena selalu memilih state dengan nilai heuristik terkecil tanpa memperhitungkan biaya langkah sebelumnya. Pendekatan ini sering kali menghasilkan waktu penyelesaian tercepat dalam *wall clock*, namun tidak menjamin solusi yang optimal. GBFS bersifat rakus terhadap keuntungan lokal dan dapat mengabaikan jalur yang lebih baik. Dalam formulasi GBFS, digunakan  $g(n) = 0$  dan  $h(n) = \text{heuristik}(n)$ .

A\* merupakan paduan antara UCS dan GBFS. A\* mempertimbangkan baik biaya langkah (`step_cost`) maupun nilai heuristik untuk mengarahkan pencarian ke solusi lebih efisien. A\* akan mengeksplorasi semua kemungkinan solusi seperti UCS, tetapi lebih terfokus karena bantuan heuristik, yang membuatnya cenderung lebih efisien. Rumus evaluasi A\* adalah  $g(n) = \text{step\_cost}(n)$  dan  $h(n) = \text{heuristik}(n)$ . Secara teoretis, A\* lebih efisien daripada UCS selama heuristik yang digunakan konsisten dan *admissible*.

Iterative Deepening A\* (IDA\*) adalah versi A\* yang diimplementasikan menggunakan pendekatan *recursive stack*. Keuntungan teoretis IDA\* adalah efisiensi memori karena tidak menyimpan seluruh antrian state seperti A\*. Namun, dalam implementasi berbasis JavaScript dan V8 engine, penghematan memori ini tidak tampak signifikan dalam hasil pengujian. IDA\* menggunakan fungsi evaluasi yang sama seperti A\*, yaitu  $g(n) = \text{step\_cost}(n)$  dan  $h(n) = \text{heuristik}(n)$ .

Dalam hal heuristik, tiga pendekatan digunakan. *Car Distance* merupakan heuristik paling dasar namun admissible, karena hanya menghitung jarak langsung antara mobil utama ke tujuan tanpa mempertimbangkan hambatan. *Car Blocking* memperbaikinya dengan menghitung jumlah mobil yang menghalangi jalan, tetapi admissible dan lebih efektif dari sebelumnya. *Car Blocking Recursive* mengembangkan konsep *Car Blocking* dengan menganalisis hambatan secara rekursif, memberikan estimasi yang lebih akurat tetapi memerlukan waktu komputasi yang lebih tinggi untuk satu iterasi solver.

Perlu dicatat bahwa UCS dan BFS akan menghasilkan urutan ekspansi dan solusi yang identik dalam konteks Rush Hour karena semua gerakan memiliki biaya yang sama. A\* secara teoritis lebih efisien dibandingkan UCS karena memanfaatkan arah dari heuristik. Sebaliknya, GBFS tidak menjamin solusi optimal karena hanya berfokus pada estimasi heuristik dan dapat mengabaikan jalur yang lebih baik secara global.

## 5.2. Kesimpulan

Program penyelesaian *puzzle* Rush Hour berhasil dikembangkan sesuai dengan spesifikasi yang ditetapkan, menggunakan bahasa pemrograman JavaScript. Seluruh fitur utama seperti implementasi algoritma pencarian (UCS, GBFS, A\*, IDA\*), berbagai heuristik (*car distance*, *car blocking*, dan *car blocking recursive*), serta optimasi struktural dan fungsional telah direalisasikan dengan baik. Selain itu, program juga dilengkapi GUI pengguna berbasis web yang interaktif, mendukung input dari teks maupun file, serta animasi visual solusi. Dengan penerapan berbagai teknik optimasi dan fitur tambahan seperti output video dan deployment online, program ini menunjukkan performa dan fungsionalitas yang solid dan dapat diandalkan.

## 5.3. Saran

Tugas Kecil ini telah berjalan dengan cukup baik, dan saya berterima kasih atas bantuan asisten yang telah memberikan kemudahan serta arahan dalam proses pengeraannya. Sebagai saran, ke depannya semoga bisa lebih dalam dan dapat lebih memfokuskan perhatian pada aspek optimasi program, khususnya dalam menganalisis jalur (path) yang diambil oleh setiap algoritma dan bagaimana strategi teknis maupun algoritmik dapat digunakan untuk mempercepat proses pencarian solusi. Meskipun demikian, saya cukup bangga karena tugas kecil ini dapat diselesaikan dengan baik dan mencerminkan pemahaman yang mendalam terhadap materi yang diberikan.

## LAMPIRAN

### Tautan

Link Github : [Tucil3\\_13523045](https://github.com/Tucil3_13523045)

Link Deployment : <https://rushhour.bwks.link/>

Link Docs : [Tucil3\\_13523045](#)

### Tabel Laporan

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan	✓	
2. Program berhasil dijalankan	✓	
3. Solusi yang diberikan program benar dan mematuhi aturan permainan	✓	
4. Program dapat membaca masukan berkas .txt dan menyimpan solusi berupa print board tahap per tahap dalam berkas .txt	✓	
5. [Bonus] Implementasi algoritma pathfinding alternatif	✓	
6. [Bonus] Implementasi 2 atau lebih heuristik alternatif	✓	
7. [Bonus] Program memiliki GUI	✓	
8. Program dan laporan dibuat (kelompok) sendiri	✓	