LABORATORIUM PEMBELAJARAN ILMU KOMPUTER

FAKULTAS ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

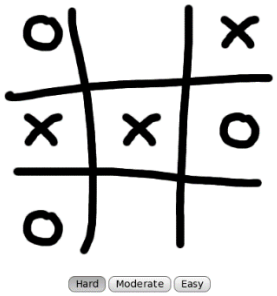
BAB	: UNINFORMED SEARCH
NAMA	: REZA AZZUBAIR WIJONARKO
NIM	: 155150200111182
TANGGAL	: 29/03/2017
JENIS	: LATIHAN
ASISTEN	: - ANNISA FITRIANI NUR - RISKI PUSPA DEWI D. P..

ACC

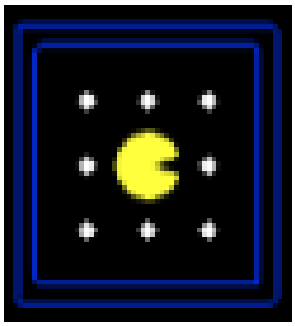
A. DEFINISI MASALAH

1. Formulasikan masalah berikut ini :

a. Tic Tac Toe



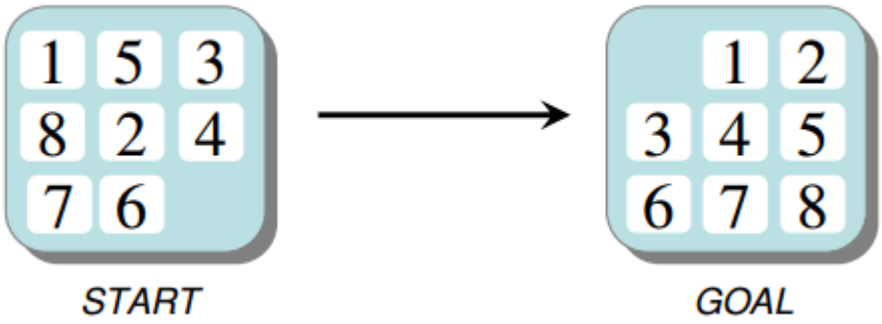
b. Simple Pac-Man



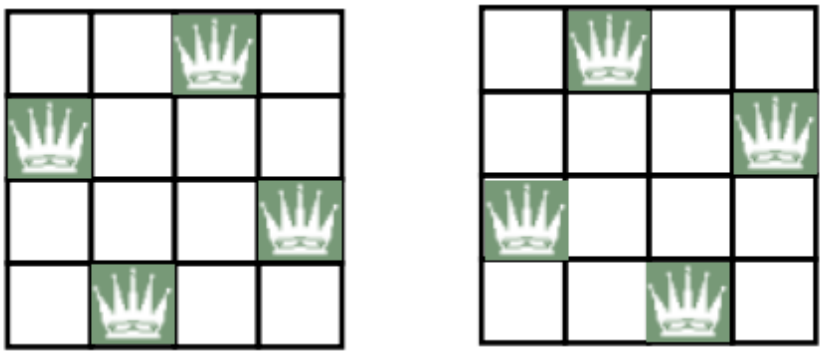
c. Permasalahan Ember Air

Dalam film Die Hard With A Vengeance, John McClane dan Zeus Carver menemukan sebuah koper berisi bom. Untuk menjinakkan bom tersebut diperlukan air tepat 4 liter. Untungnya di dekat koper terdapat kolam dengan air yang melimpah. Pada koper tersebut terdapat 2 ember, masing-masing berkapasitas 5 liter dan 3 liter; Pada dua ember tersebut tidak ada tanda ukuran sama sekali. Awalnya keduanya kosong. Mereka boleh mengisi air ke dalam ember tersebut dan juga boleh menumpahkan air dari ember ke kolam. Mereka juga boleh menuangkan air dari satu ember ke ember yang lain. Tujuannya adalah mendapatkan tepat 4 liter air.

d. 8-Puzzle Problem



e. 4-Queens problem

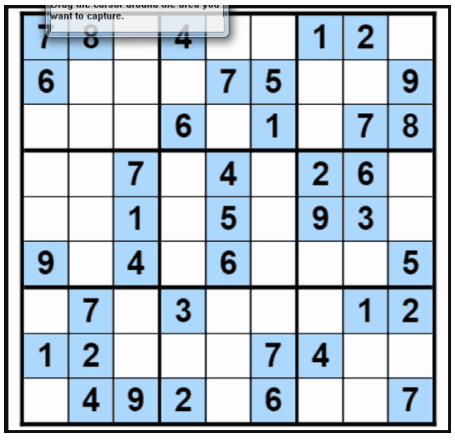


f. Permasalahan Misionaris dan Kanibal

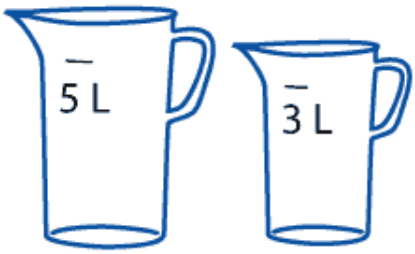
Di sebuah tepi sungai ada tiga misionaris dan tiga kanibal. Ada satu perahu yang tersedia yang dapat menampung hingga dua orang dan mereka ingin menggunakannya untuk menyeberangi sungai. Jika jumlah kanibal melebihi jumlah misionaris di kedua tepi sungai itu, para misionaris akan dimakan.

Bagaimana caranya menggunakan perahu tersebut untuk membawa semua misionaris dan kanibal di seberang sungai dengan aman?

g. Sudoku



2. Dalam film Die Hard With A Vengeance, John McClane dan Zeus Carver menemukan sebuah koper berisi bom. Untuk menjinakkan bom tersebut diperlukan air tepat 4 liter. Untungnya di dekat koper terdapat kolam dengan air yang melimpah. Pada koper tersebut terdapat 2 ember, masing-masing berkapasitas 5 liter dan 3 liter; Pada dua ember tersebut tidak ada tanda ukuran sama sekali. Awalnya keduanya kosong. Mereka boleh mengisi air ke dalam ember tersebut dan juga boleh menumpahkan air dari ember ke kolam. Mereka juga boleh menuangkan air dari satu ember ke ember yang lain. Tujuannya adalah mendapatkan tepat 4 liter air.



   a. Selesaikan permasalahan di atas secara manual menggunakan algoritama BFS
   b. Selesaikan permasalahan di atas secara manual menggunakan algoritama DFS

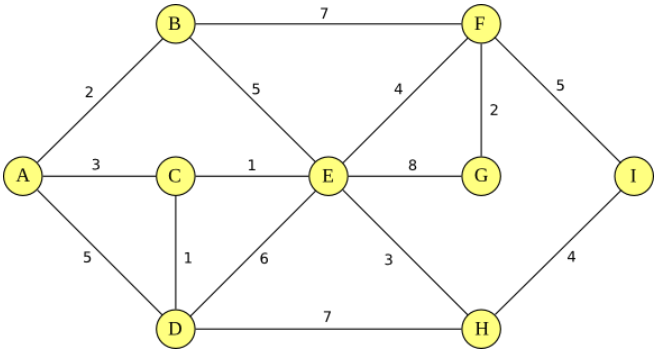
3. 8-Puzzle. Selesaikan permasalahan 8-Puzzle berikut ini secara manual dengan menuliskan detail langkah-langkah yang digunakan

1	2	3	1	2	3
4	8	5	4	5	6
7	6		7	8	

Start State **Goal State**

   Selesaikan permasalahan di atas secara manual menggunakan algoritama Iterative Deepening Search!

4. Carilah rute dari kota C ke kota I secara manual menggunakan Uniform Cost Search


B
7
F
2
5
4
5
2
A
3
C
1
E
8
G
I
5
1
6
3
4
D
7
H

B. JAWAB

1.

A. Tic Tac Toe

a) Initial state:
Dalam keadaan berikut:

O		x
x	x	O
O		

b) Goal state:
Salah satu simbol menang. Contoh:

O	x	x
x	x	O
O	x	O

c) Successors:
Setiap proses menambahkan X dan O.

d) Path cost :
Setiap langkah yang dilakukan bernilai 1.

B. Simple Pac – Man

a) Initial state:
Butiran-butiran putih sebanyak 8 yang masih utuh dan tersebar mengelilingi Pac-Man.

b) Goal state:
Pac-Man memakan semua butiran itu.

c) Successors:
Makan, maju, hadap kanan, hadap kiri, hadap atas, hadap bawah.

d) Path cost:
Setiap langkah maju yang dilakukan bernilai 1.

C. Pemasalahan Ember Air

a) Initial state:
Air, ember 5 liter kosong, dan ember 3 liter kosong.

b) Goal state:
Ember 5 liter berisi air sebanyak 4 liter.

c) Successors:

Mengisi ember (A atau B) hingga penuh, membuang semua isi ember (A atau B), menuang isi ember (A ke B) hingga salah satu kosong atau salah satu penuh.

d) Path cost:
Setiap aksi yang dilakukan bernilai 1.

D. 8 puzzle problem

a) Initial state:

b) Succesor
Dengan memindahkan bagian yang kosong ke sebelah kiri, kanan, atas atau bawah.

c) Goal

d) Path
Setiap perpindahan akan melibatkan 1 angka. Banyaknya cost sama dengan jumlah perpindahan 1 angka sampai menuju goal.

E. 4 – Queens Problem

a) Initial state:
Dalam kotak 4 x 4 tidak terdapat queen sama sekali.

b) Successor function:
Menggeser angka ke arah atas, bawah, kiri, kanan atau diagonal.

c) Goal state :
Semua queen terpasang pada kotak tanpa termakan queen lainnya.

d) Path cost:
Setiap jalur yang dapat dimakan salah satu queen bernilai 0.

F. Permasalahan Misionaris dan kanibal

a) Initial state:
Diseberang pulau terdapat 3 Misionaris dan 3 Kanibal.

b) Successor function:

Pilih Salah satu jenis Misionaris atau kanibal untuk menjadi penjembatan ke pulau seberang.

c) Goal state :
Untuk menyebrangkan 3 misionaris dan 3 kanibal tanpa seorang pun misionaris dimakan oleh kanibal.

d) Path cost :
Setiap langkah yang dilakukan bernilai 1.

G. Sudoku

a) Initial state:
Sudoku 9 x 9 dengan 9 persegi 3 x 4 di dalamnya dan angka-angka awal:

	1	2	3	4	5	6	7	8	9
A	7	8		4			1	2	
B	6				7	5			9
C				6		1		7	8
D			7		4		2	6	
E			1		5		9	3	
F	9		4		6				5
G		7		3				1	2
H	1	2				7	4		
I		4	9	2		6			7

b) Successors:
Angka berapapun yang mengisi suatu kotak dengan syarat: 1) angka itu tidak sama dengan angka-angka lain dalam satu baris horizontal; 2) angka itu tidak sama dengan angka-angka lain dalam satu baris vertikal; 3) angka itu tidak berulang dalam satu kotak 3 x 3 yang sama.

c) Goal state:
Ketika semua kotak terisi dan memenuhi aturan-aturan yang telah dijelaskan pada bagian b) Successors:

	1	2	3	4	5	6	7	8	9
A	7	8	5	4	3	9	1	2	6
B	6	1	2	8	7	5	3	4	9
C	4	9	3	6	2	1	5	7	8
D	8	5	7	9	4	3	2	6	1
E	2	6	1	7	5	8	9	3	4
F	9	3	4	1	6	2	7	8	5
G	5	7	8	3	9	4	6	1	2
H	1	2	6	5	8	7	4	9	3
I	3	4	9	2	1	6	8	5	7

d) Path cost:
Tiap mengisi sebuah kotak dengan sebuah angka, cost-nya adalah 1.

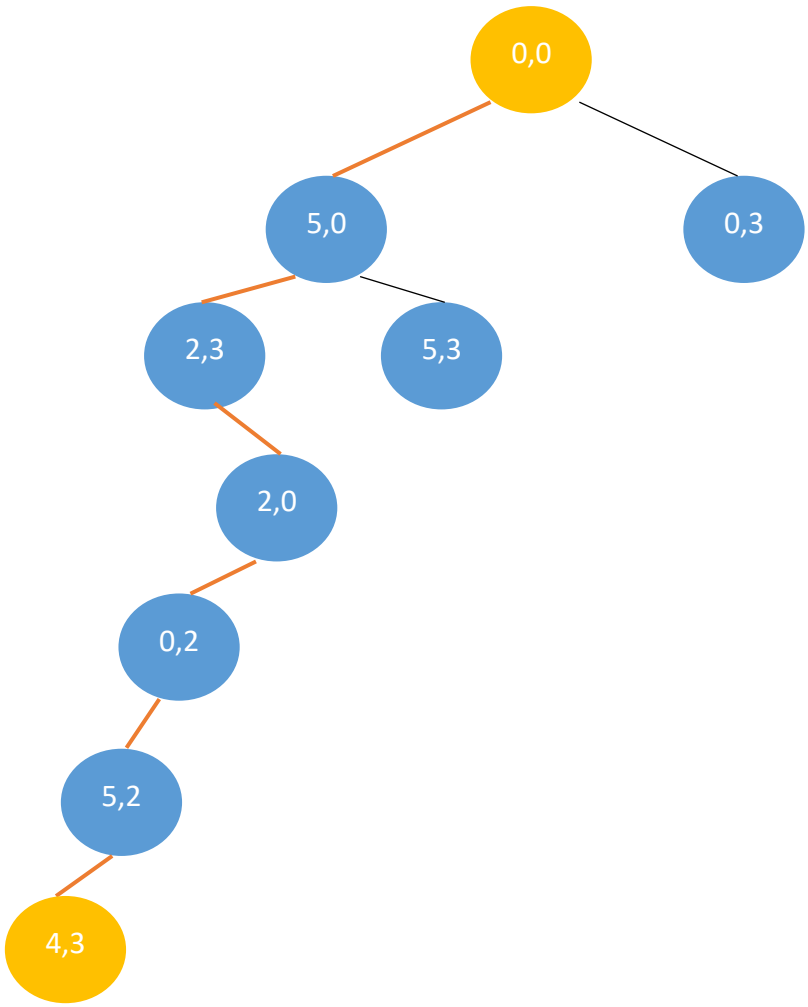
2. Penyelesaian masalah menggunakan algoritma:
    a) BFS
       Child kiri – operasi apapun terhadap gelas A (5L) dan gelas A ke gelas B (3L).
       Child kanan – operasi apapun terhadap gelas B dan gelas B ke gelas A.

       Solution set: mengisi gelas A, menuang isi gelas A ke gelas B, membuang isi gelas B, menuang isi gelas A ke B, mengisi gelas A, menuang isi gelas A ke gelas B.

b) DFS

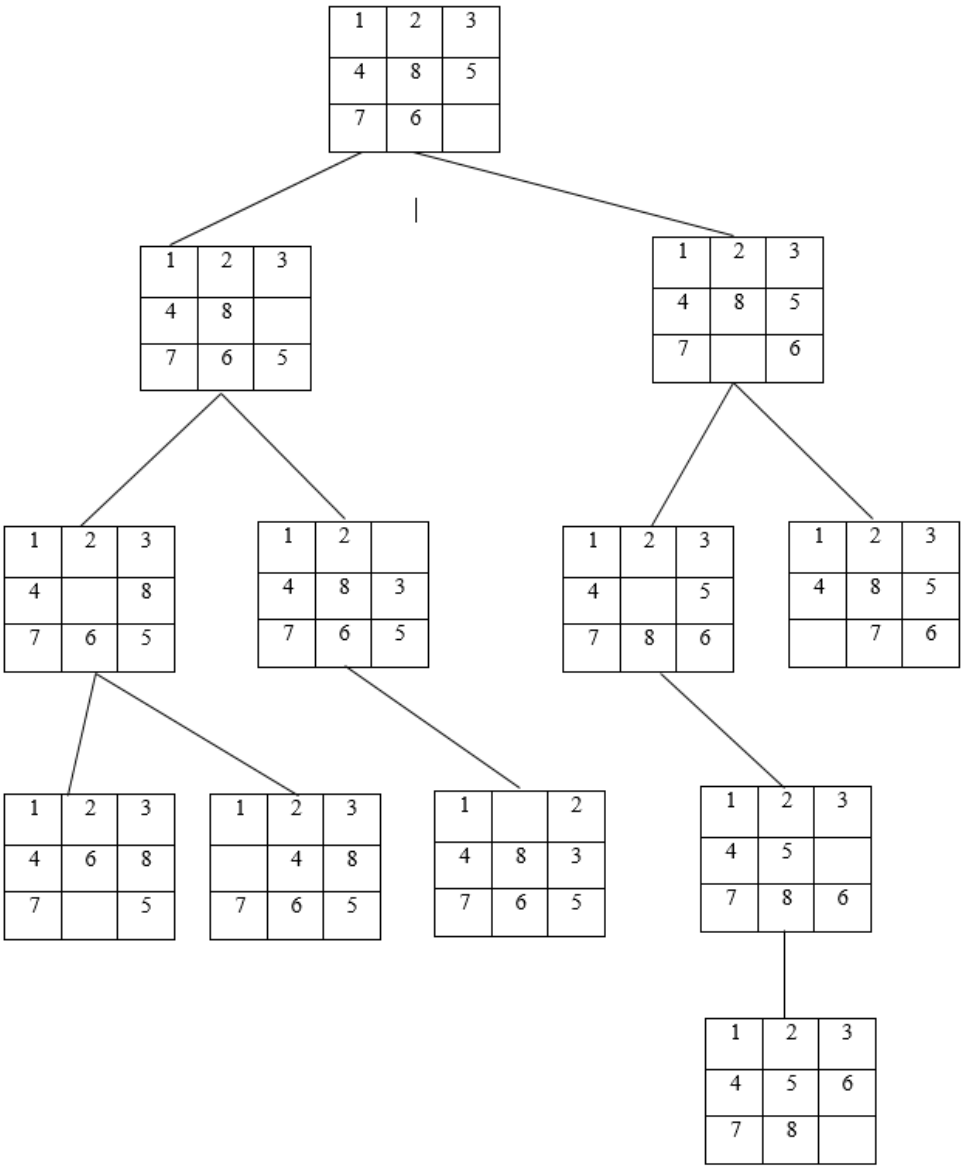
Child kiri – operasi apapun terhadap gelas A dan gelas A ke gelas B.

Child kanan – operasi apapun terhadap gelas B dan gelas B ke gelas A.

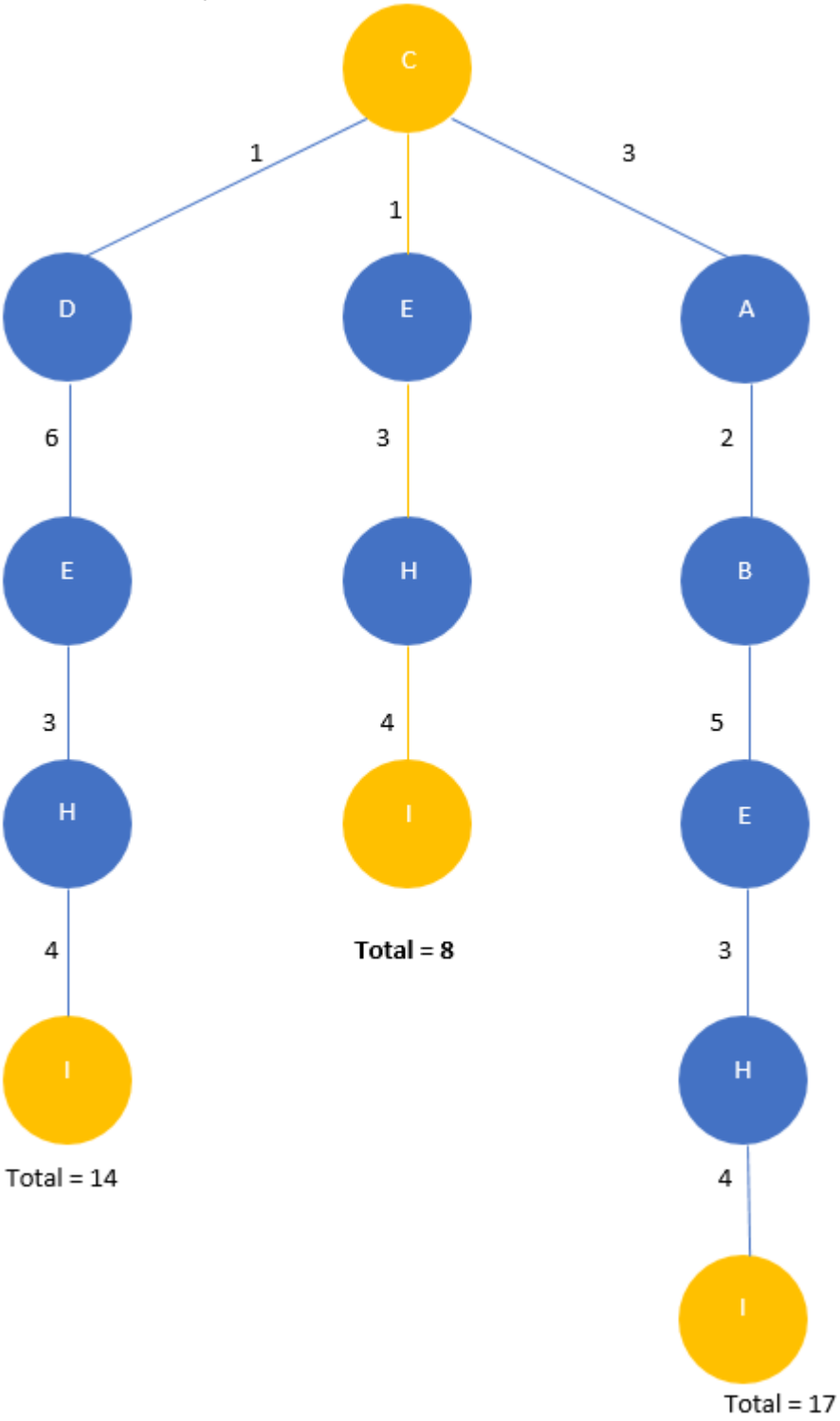


Solution set: mengisi gelas A, menuang isi gelas A ke gelas B, membuang isi gelas B, menuang isi gelas A ke B, mengisi gelas A, menuang isi gelas A ke gelas B.

3. Masalah 8-puzzle:



4. Uninformed cost search:



Pengerjaan di atas mengikuti algoritma berikut:

Masukkan root ke dalam antrian

While antrian not empty

Pilih elemen dengan prioritas maksimum dari antrian (dalam hal ini cost terkecil)

If prioritas sama, jalur dipilih secara alphabetis terkecil

If jalur sampai ke goal state, print jalur and exit

Else

Masukkan semua children dari elemen yang dipilih, dengan cost ssssssskumulatif sebagai prioritas (cost terkecil memiliki prioritas terbesar)

Dari kota C ke kota I

Keterangan => (-) artinya melalui

[C – D = 1] , [C – E = 1] , [C – A = 3] bandingkan tiap cost-nya

[C – D = 1]
[C – D – A = 6] , [C – D – E = 7] bandingkan dengan [C – E = 1] dan [C – A = 3]

[C – E = 1]
[C – E – F = 5] , [C – E – H =4] bandingkan dengan [C – A = 3] , [C – D – A = 6] dan [C – D – E = 7]

[C – E – H = 4]
[C – E – H – I = 8] bandingkan dengan [C – E – H – I = 11] (tidak bisa dilewati karena node D sudah dieksplorasi)

[C – E – H – I = 8]

C. KESIMPULAN

1. Jelaskan apa maksud dan komponen dari problem solving agent?
   Problem solving agent adalah jenis goal-based agent yang dapat merumuskan tujuan (goal formulation) berdasarkan performance measure-nya, merumuskan masalah (problem formulation) berdasarkan tujuannya, dan mengeksekusi aksi-aksi untuk mencapai tujuannya menggunakan algoritma tertentu. Problem solving agent harus beroperasi dalam environment yang observable, known, dan deterministic agar penilaian-penilaiannya baik dan benar.
   Bila diteliti lebih jauh, sifat problem formulation agen ini terdiri atas 5 komponen:
   1) Initial state, yakni keadaan awal si agen dalam environment tempat ia hendak memecahkan permasalahan
   2) Aksi-aksi yang mungkin agar agen dapat mencapai tujuannya
   3) Penjelasan tentang akibat dari suatu aksi yang bisa ia lakukan (transition state)
   4) Goal test, yakni penentu apakah si agen sudah mencapai goal state atau belum
   5) Path cost, yakni nilai dalam angka dari tiap aksi (path) yang ia lakukan
2. Jelaskan apa maksud dari Uninformed Search?
   Uninformed search (blind search) adalah algoritma pencarian solusi yang pelakunya (dalam hal ini agen) tidak memiliki pengetahuan (informasi) apa-apa tentang environment tempat ia bekerja. Jadi, agen hanya mengetahui sebuah permasalahan dan definisi permasalahan itu dan memilih action berikutnya tanpa mempertimbangkan pilihan-pilihannya terdahulu. Contohnya adalah algoritma breadth first search (BFS) yang mencari solusi tanpa mempertimbangkan childs yang telah dilaluinya.
3. Jelaskan apa perbedaan Uninformed Search dan Informed Search?
   Bila uninformed search merupakan algoritma pencarian solusi yang tidak mempertimbangkan pilihan-pilihannya yang terdahulu, uninformed search (heuristic search) merupakan algoritma pencarian solusi yang mempertimbangkan action yang akan diambil agen berdasarkan pilihan-pilihannya terdahulu. Contohnya adalah algoritma A* (A star) yang memilih action atau state berikutnya berdasarkan cost tertentu dan “informasi heuristik” dari fungsi heuristik (heuristic function).

LABORATORIUM PEMBELAJARAN ILMU KOMPUTER

FAKULTAS ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

BAB	: UNINFORMED SEARCH
NAMA	: REZA AZZUBAIR WIJONARKO
NIM	: 155150200111182
TANGGAL	: 29/03/2017
JENIS	: TUGAS
ASISTEN	: - ANNISA FITRIANI NUR - RISKI PUSPA DEWI D. P..

ACC

A. DEFINISI MASALAH

Anda memiliki 2 ember, masing-masing berkapasitas n liter dan m liter. Pada dua ember tersebut tidak ada tanda ukuran sama sekali. Awalnya keduanya kosong. Anda boleh mengisi air ke dalam ember tersebut dan juga boleh menumpahkan air dari ember ke kolam. Anda juga boleh menuangkan air dari satu ember ke ember yang lain. Tujuannya adalah mendapatkan tepat k liter air di mana.

Buatlah program dengan menggunakan algoritma Iterative-Deepening Search untuk menyelesaikan masalah tersebut.

Input:

a. Nilai kapasitas ember pertama (n)
b. Nilai kapasitas ember kedua (m)
c. Nilai kapasitas air yang diinginkan (k)

Output:

a. Semua lagkah-langkah yang sudah dicoba untuk pencarian solusi per masing-maing limit (Semua node yang sudah diexpand)
b. Urutan langkah yang merupakan solusi
c. Berapa jumlah operasi yang harus dilakukan untuk sampai pada solusi

Kasus khusus yang wajib diujikan:

a. m = 3; n=5; k=4;
b. m = 4; n=3; k=2;
c. m = 5; n=7; k=1;
d. m = 9; n=7; k=2;
e. m = 11; n=7; k=7;
f. m = 4; n=7; k=6;
g. m = 5; n=8; k=3;
h. m = 5; n=8; k=4;
i. m = 6; n=11; k=8;
j. m = 7; n=11; k=5;
k. m = 9; n=11; k=8;
l. m = 12; n=11; k=6;

B. JAWAB

State.java

```
1   package praktikumkb2;
2   public class State {
3
4       int m, n, cara, level;
5       State pre;
6
7       public State(int m, int n, int cara, int level) {
8           this.m = m;
9           this.n = n;
10          this.cara = cara;
11          this.level = level;
12      }
13
14      public State(int m, int n, int cara,int level, State
15  pre) {
16          this.m = m;
17          this.n = n;
18          this.cara = cara;
19          this.level=level;
20          this.pre = pre;
21      }
22
23      @Override
24      public String toString() {
25          switch(cara){
26              case 0:
27                  return String.format("Level "+level+" :
28  %s", "Initial State, kedua wadah kosong");
29              case 1:
30                  return String.format("Level "+level+" :
31  %s", "Isi penuh wadah m "+m+" liter, m = "+m+" n = "+n);
32              case 2:
33                  return String.format("Level "+level+" :
34  %s", "Isi penuh wadah n "+n+" liter, m = "+m+" n = "+n);
35              case 3:
36                  return String.format("Level "+level+" :
37  %s", "Buang semua air di wadah m, m = "+m+" n = "+n);
38              case 4:
39                  return String.format("Level "+level+" :
40  %s", "Buang semua air di wadah n, m = "+m+" n = "+n);
41              case 5:
42                  return String.format("Level "+level+" :
43  %s", "Tuang air liter dari wadah m ke wadah n, m = "+m+" n
44  = "+n);
45              case 6:
46                  return String.format("Level "+level+" :
47  %s", "Tuang air dari wadah n ke wadah m, m = "+m+" n =
48  "+n);
49              default:
50                  return String.format("Level "+level+" :
51  %s", "END");
52          }
53      }
```

```
54
55      @Override
56      public boolean equals(Object obj) {
57          if (this == obj) {
58              return true;
59          }
60          if (obj == null) {
61              return false;
62          }
63          if (getClass() != obj.getClass()) {
64              return false;
65          }
66          State other = (State) obj;
67          if(m == other.m && n == other.n){
68              return true;
69          }
70          if(m != other.m){
71             return false;
72          }
73          if(n != other.n) {
74              return false;
75          }
76          return true;
77      }
78  }
```

waterIDS.java

```
1   package praktikumkb2;
2   import java.util.LinkedList;
3   public class waterIDS {
4       int wadah1,wadah2;
5       boolean ketemu=false;
6       LinkedList<State> route=new LinkedList();
7       int totStep=0;
8       int step=0;
9
10      public void start(int m,int n, int target){
11          State Initial=new State(0,0,0,0);
12          wadah1=m;
13          wadah2=n;
14          int depth=1;
15          while(ketemu==false){
16              step=0;
17              System.out.println("-------------------------
18  );
19              System.out.println("Pencarian Untuk kedalaman
20  = "+depth);
21              dfs(Initial, target, depth);
22              System.out.println("-------------------------
23  );
24              System.out.println("Jumlah langkah yang
25  dilakukan dalam IDS kedalaman "+depth+" adalah "+step);
26              totStep+=step;
27              depth++;
28              System.out.println("-------------------------
29  );
```

```
30          }
31      }
32
33      private void dfs(State initial, int target, int
34  depth){
35          LinkedList<State> stack=new LinkedList();
36          LinkedList<State> Duplicate=new LinkedList();
37
38          State Finish=null;
39
40          Duplicate.push(initial);
41          stack.push(initial);
42
43          while(!stack.isEmpty()){
44              State curr=stack.pop();
45              step++;
46              System.out.println(curr);
47              if(curr.m==target || curr.n==target){
48                  Finish=curr;
49                  makePath(Finish);
50                  ketemu=true;
51                  break;
52              }
53
54              if(curr.m==0){ //wadah m diisi
55                  State next=new State(curr.m+wadah1,
56  curr.n, 1, curr.level+1, curr);
57                  checkDup(stack,Duplicate,next,depth);
58              }
59
60              if(curr.n==0){ //wadah n diisi
61                  State next=new State(curr.m,
62  curr.n+wadah2, 2, curr.level+1, curr);
63                  checkDup(stack,Duplicate,next,depth);
64              }
65
66              if(curr.m > 0){ //wadah m dikosongi
67                  State next=new State(0, curr.n, 3,
68  curr.level+1, curr);
69                  checkDup(stack,Duplicate,next,depth);
70              }
71
72              if(curr.n > 0){ //wadah n dikosongi
73                  State next=new State(curr.m, 0, 4,
74  curr.level+1, curr);
75                  checkDup(stack,Duplicate,next,depth);
76              }
77
78              if(curr.m > 0 && curr.n < wadah2){ //wadah m
79  tuang ke wadah n sampai penuh/sampai m habis
80                  int tuang=Math.min(curr.m,(wadah2-
81  curr.n));
82                  int sisa= (curr.m - tuang);
83                  if(sisa < 0){
84                      sisa=0;
85                  }
86                  State next=new State(sisa, curr.n+tuang,
87  5, curr.level+1, curr);
88                  checkDup(stack,Duplicate,next,depth);
```

```
89              }
90
91              if(curr.m < wadah1 && curr.n > 0) { //wadah n
92  tuang ke wadah m sampai penuh/sampai n habis
93                  int tuang=Math.min(curr.n,(wadah1 -
94  curr.m));
95                  int sisa=(curr.n - tuang);
96                  if(sisa < 0){
97                      sisa=0;
98                  }
99                  State next=new State(curr.m+tuang, sisa,
100 6, curr.level+1, curr);
101                 checkDup(stack,Duplicate,next,depth);
102             }
103         }
104
105         if(Finish != null){
106             System.out.println("Wadah m Wadah n");
107             System.out.println(Finish.m +"
108 "+Finish.n);
109         } else {
110             System.out.println("Error: Not Possible");
111         }
112     }
113
114     private void checkDup(LinkedList<State> asli,
115 LinkedList<State> Dup, State cek, int depth){
116         if(!Dup.contains(cek) && cek.level < depth){
117             asli.push(cek);
118             Dup.push(cek);
119         }
120     }
121
122     private void makePath(State goal){
123         route.push(goal);
124         State path=goal;
125         while(path.pre!=null){
126             route.push(path.pre);
127             path=path.pre;
128         }
129     }
130
131     public void getPath(){
132         System.out.println("-----------------------------
133 );
134         System.out.println("Urutan Solusi Langkah");
135         System.out.println("-----------------------------
136 );
137         while(!route.isEmpty()){
138             System.out.println(route.pop());
139         }
140         System.out.println("-----------------------------
141 );
142     }

143     public int getStep() {
144         return totStep;
145     }
146 }
```

waterIDS.java

```
1   package praktikumkb2;
2   import java.util.Scanner;
3   public class Main {
4
5       public static void main(String[] args) {
6           waterIDS test = new waterIDS();
7           Scanner in=new Scanner(System.in);
8           System.out.println("Simulasi Wadah Air IDS
9   Search");
10          System.out.println("----------------------------
11  );
12          System.out.print("Input ukuran wadah m\t: ");
13          int m=in.nextInt();
14          System.out.print("Input ukuran wadah n\t: ");
15          int n=in.nextInt();
16          System.out.print("Input target air\t: ");
17          int k=in.nextInt();
18          test.start(m, n, k);
19          System.out.println("----------------------------
20  );
21          test.getPath();
22          System.out.println("Jumlah langkah total yang
23  dibutuhkan IDS : " + test.getStep() + " langkah");
24      }
25  }
```

Penjelasan

State.java

- Baris 4-5: mendeklarasi instance variables yang ada pada class state adalah m, n, cara, level dan pre. Variable pre digunakan untuk menyimpan state sebelum state sekarang.
- Baris 7-12: constructor dengan parameter m, n, cara, dan level untuk menginisialisasi instance variables pada baris 4.
- Baris 14-21: constructor dengan parameter m, n, cara, level, dan pre untuk menginisialisasi instance variables pada baris 4-5.
- Baris 24-53: instance method toString digunakan untuk mencetak langkah dengan mengembalikan String yang sesuai nilai dari variable cara.
- Baris 56-77: instance method equals digunakan untuk membandingkan state sekarang dengan state yang ada di parameter apakah sama atau tidak.

waterIDS.java

- Baris 4-8: mendeklarasi dan/atau menginisialisasi instance variables wadah1, wadah2, dan ketemu untuk mengecek status apakah target sudah ketemu; step dan totStep untuk menghitung langkah tiap level dan total langkah. Selain itu terdapat variable linked list route untuk menyimpan langkah solusi.
- Baris 10-31: instance method void start digunakan untuk memulai proses pencarian state sesuai parameter-parameter target, wadah m, wadah n. Dalam method ini terdapat variable depth yang digunakan untuk

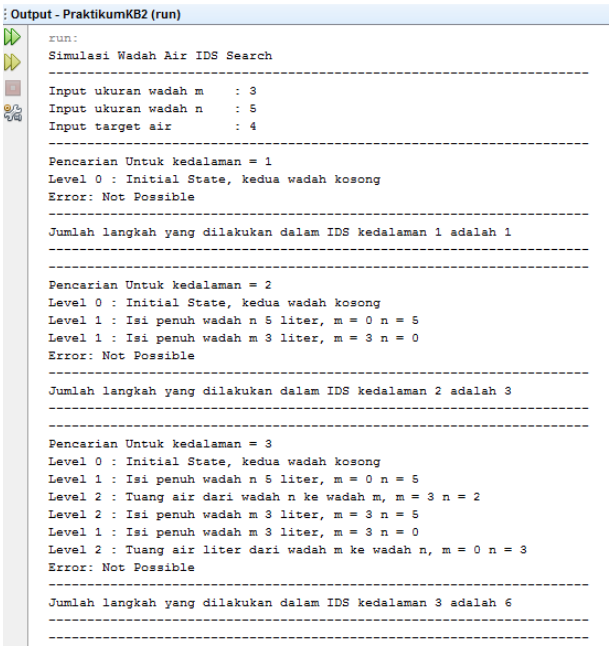
menentukan seberapa dalam pencarian dilakukan. Pencarian sendiri dilakukan selama variable ketemu false dan dengan method dfs.

- Baris 33-112: method void dfs merupakan pencarian inti dari program. Parameternya merupakan initial, target dan depth. Method ini akan melakukan push initial state ke linkedlist stack dan Duplicate. Selama stack tidak kosong method akan melakukan pop dari stack dan menyimpan state yang dipop divariable curr. Lalu dicek apakah m atau n dari curr memenuhi target, jika iya ganti nilai var ketemu jadi true dan buat rute langkah dengan method makePath lalu hentikan iterasi.
  Dalam method dfs juga dilakukan push successor yang mungkin dari initial state. Dalam penyimpanan ditambah data cara yang menunjukkan langkah apa yang dilakukan dan data predecessor dari successor untuk mempermudah tracing back nya. Saat melakukan push, method ini dibantu method checkDup yang merupakan syarat kapan method akan melakukan push dari successor.
- Baris 114-120: instance method checkDup memiliki parameter linkedlist<State> asli dan Dup, state yang ingin dipush dan depth. Method ini akan melakukan push ke linkedlist<State> asli dan Dup jika state yang akan dipush tidak ada di Dup dan levelnnya kurang dari depth.
- Baris 122-129: instance method makePath berparameter goal (state terakhir) dari pencarian. Lalu method akan melakukan push state tersebut dan predecessor nya secara berurutan sampai ke state yang tidak memiliki predecessor (initial state) ke linkedlist route.
- Baris 131-142: instance method getPath digunakan untuk mencetak solusi dengan mencetak nilai pop dari linkedlist route.
- Baris 143-145: instance method getStep digunakan untuk mengembalikan nilai totStep.

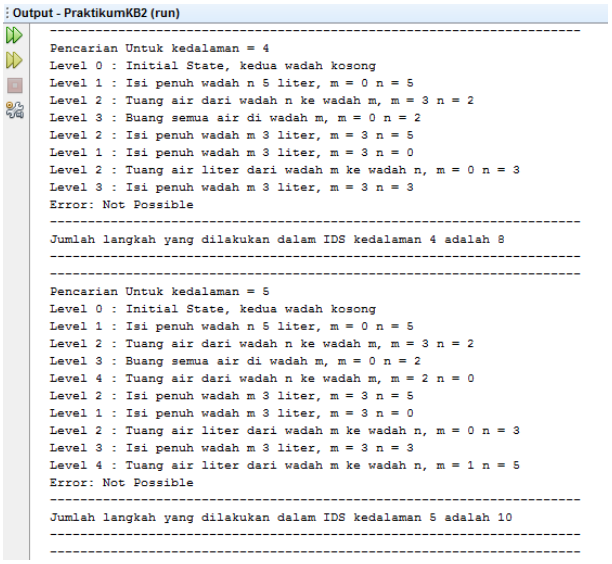
Main method (baris 123-138)

- Baris 3-25: method main diawali dengan menginputkan nilai wadah m, wadah n dan target untuk diproses lalu mencetak path solusi yang benar dan langkah total yang dilakukan pencarian IDS.

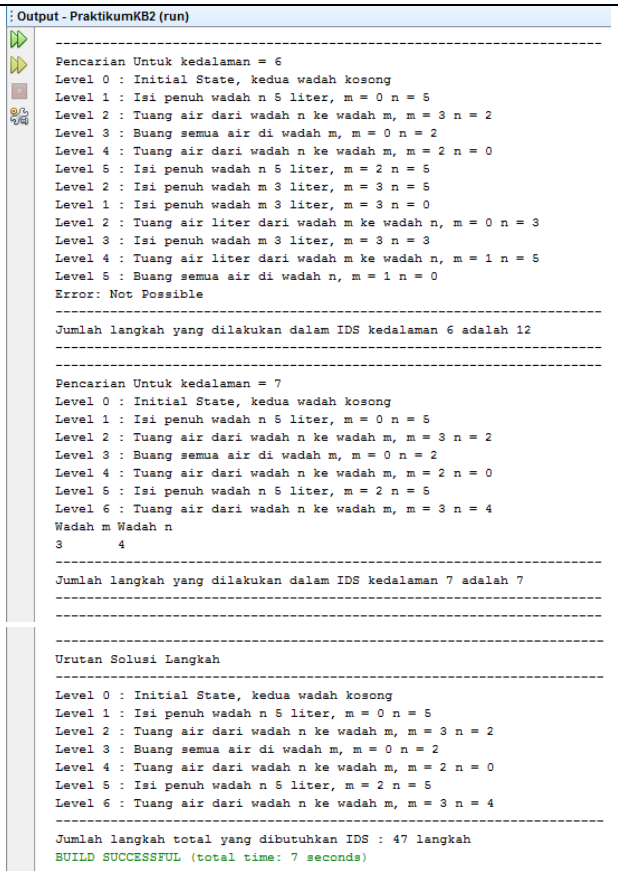
Screenshot:



```
Output - PraktikumKB2 (run)
run:
Simulasi Wadah Air IDS Search
--------------------------------------------------------------------------------
Input ukuran wadah m    : 3
Input ukuran wadah n    : 5
Input target air        : 4
--------------------------------------------------------------------------------
Pencarian Untuk kedalaman = 1
Level 0 : Initial State, kedua wadah kosong
Error: Not Possible
--------------------------------------------------------------------------------
Jumlah langkah yang dilakukan dalam IDS kedalaman 1 adalah 1
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Pencarian Untuk kedalaman = 2
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 5 liter, m = 0 n = 5
Level 1 : Isi penuh wadah m 3 liter, m = 3 n = 0
Error: Not Possible
--------------------------------------------------------------------------------
Jumlah langkah yang dilakukan dalam IDS kedalaman 2 adalah 3
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Pencarian Untuk kedalaman = 3
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 5 liter, m = 0 n = 5
Level 2 : Tuang air dari wadah n ke wadah m, m = 3 n = 2
Level 2 : Isi penuh wadah m 3 liter, m = 3 n = 5
Level 1 : Isi penuh wadah m 3 liter, m = 3 n = 0
Level 2 : Tuang air liter dari wadah m ke wadah n, m = 0 n = 3
Error: Not Possible
--------------------------------------------------------------------------------
Jumlah langkah yang dilakukan dalam IDS kedalaman 3 adalah 6
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
```



```
Output - PraktikumKB2 (run)
--------------------------------------------------------------------------------
Pencarian Untuk kedalaman = 4
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 5 liter, m = 0 n = 5
Level 2 : Tuang air dari wadah n ke wadah m, m = 3 n = 2
Level 3 : Buang semua air di wadah m, m = 0 n = 2
Level 2 : Isi penuh wadah m 3 liter, m = 3 n = 5
Level 1 : Isi penuh wadah m 3 liter, m = 3 n = 0
Level 2 : Tuang air liter dari wadah m ke wadah n, m = 0 n = 3
Level 3 : Isi penuh wadah m 3 liter, m = 3 n = 3
Error: Not Possible
--------------------------------------------------------------------------------
Jumlah langkah yang dilakukan dalam IDS kedalaman 4 adalah 8
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Pencarian Untuk kedalaman = 5
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 5 liter, m = 0 n = 5
Level 2 : Tuang air dari wadah n ke wadah m, m = 3 n = 2
Level 3 : Buang semua air di wadah m, m = 0 n = 2
Level 4 : Tuang air dari wadah n ke wadah m, m = 2 n = 0
Level 2 : Isi penuh wadah m 3 liter, m = 3 n = 5
Level 1 : Isi penuh wadah m 3 liter, m = 3 n = 0
Level 2 : Tuang air liter dari wadah m ke wadah n, m = 0 n = 3
Level 3 : Isi penuh wadah m 3 liter, m = 3 n = 3
Level 4 : Tuang air liter dari wadah m ke wadah n, m = 1 n = 5
Error: Not Possible
--------------------------------------------------------------------------------
Jumlah langkah yang dilakukan dalam IDS kedalaman 5 adalah 10
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
```


Output - PraktikumKB2 (run)

Pencarian Untuk kedalaman = 6
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 5 liter, m = 0 n = 5
Level 2 : Tuang air dari wadah n ke wadah m, m = 3 n = 2
Level 3 : Buang semua air di wadah m, m = 0 n = 2
Level 4 : Tuang air dari wadah n ke wadah m, m = 2 n = 0
Level 5 : Isi penuh wadah n 5 liter, m = 2 n = 5
Level 2 : Isi penuh wadah m 3 liter, m = 3 n = 5
Level 1 : Isi penuh wadah m 3 liter, m = 3 n = 0
Level 2 : Tuang air liter dari wadah m ke wadah n, m = 0 n = 3
Level 3 : Isi penuh wadah m 3 liter, m = 3 n = 3
Level 4 : Tuang air liter dari wadah m ke wadah n, m = 1 n = 5
Level 5 : Buang semua air di wadah n, m = 1 n = 0
Error: Not Possible

Jumlah langkah yang dilakukan dalam IDS kedalaman 6 adalah 12

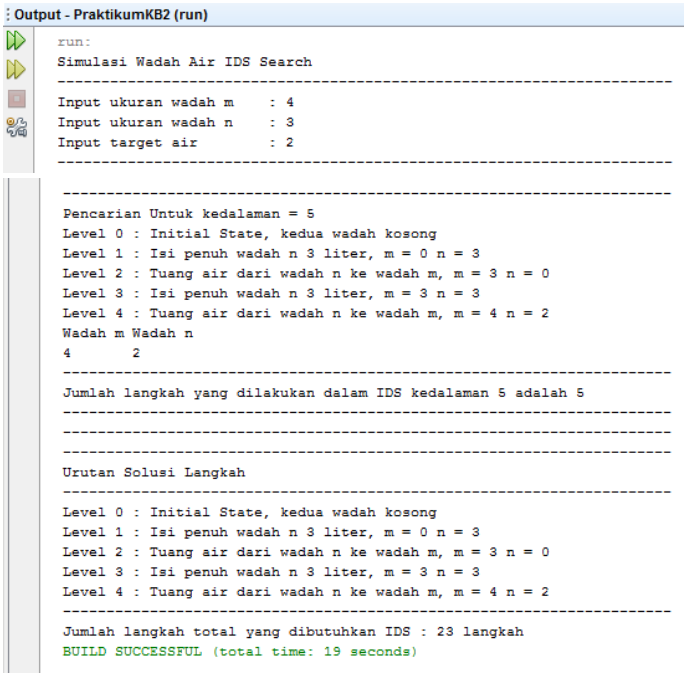
Pencarian Untuk kedalaman = 7
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 5 liter, m = 0 n = 5
Level 2 : Tuang air dari wadah n ke wadah m, m = 3 n = 2
Level 3 : Buang semua air di wadah m, m = 0 n = 2
Level 4 : Tuang air dari wadah n ke wadah m, m = 2 n = 0
Level 5 : Isi penuh wadah n 5 liter, m = 2 n = 5
Level 6 : Tuang air dari wadah n ke wadah m, m = 3 n = 4
Wadah m Wadah n
3 4

Jumlah langkah yang dilakukan dalam IDS kedalaman 7 adalah 7

Urutan Solusi Langkah

Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 5 liter, m = 0 n = 5
Level 2 : Tuang air dari wadah n ke wadah m, m = 3 n = 2
Level 3 : Buang semua air di wadah m, m = 0 n = 2
Level 4 : Tuang air dari wadah n ke wadah m, m = 2 n = 0
Level 5 : Isi penuh wadah n 5 liter, m = 2 n = 5
Level 6 : Tuang air dari wadah n ke wadah m, m = 3 n = 4

Jumlah langkah total yang dibutuhkan IDS : 47 langkah
BUILD SUCCESSFUL (total time: 7 seconds)


Output - PraktikumKB2 (run)
run:
Simulasi Wadah Air IDS Search

Input ukuran wadah m : 4
Input ukuran wadah n : 3
Input target air : 2

Pencarian Untuk kedalaman = 5
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 3 liter, m = 0 n = 3
Level 2 : Tuang air dari wadah n ke wadah m, m = 3 n = 0
Level 3 : Isi penuh wadah n 3 liter, m = 3 n = 3
Level 4 : Tuang air dari wadah n ke wadah m, m = 4 n = 2
Wadah m Wadah n
4 2

Jumlah langkah yang dilakukan dalam IDS kedalaman 5 adalah 5

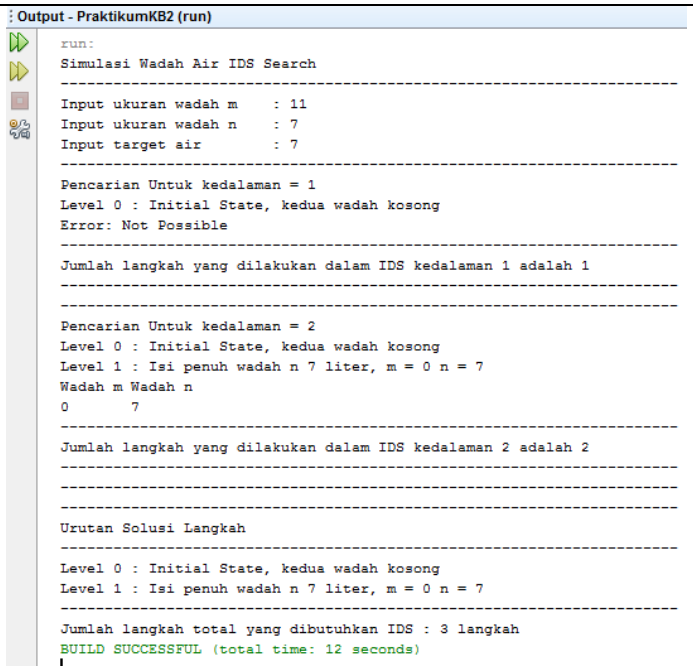
Urutan Solusi Langkah

Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 3 liter, m = 0 n = 3
Level 2 : Tuang air dari wadah n ke wadah m, m = 3 n = 0
Level 3 : Isi penuh wadah n 3 liter, m = 3 n = 3
Level 4 : Tuang air dari wadah n ke wadah m, m = 4 n = 2

Jumlah langkah total yang dibutuhkan IDS : 23 langkah
BUILD SUCCESSFUL (total time: 19 seconds)

```
Output - PraktikumKB2 (run)
run:
Simulasi Wadah Air IDS Search
--------------------------------------------------------------------------
Input ukuran wadah m    : 9
Input ukuran wadah n    : 7
Input target air        : 2
--------------------------------------------------------------------------
--------------------------------------------------------------------------
Pencarian Untuk kedalaman = 3
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 7 liter, m = 0 n = 7
Level 2 : Tuang air dari wadah n ke wadah m, m = 7 n = 0
Level 2 : Isi penuh wadah m 9 liter, m = 9 n = 7
Level 1 : Isi penuh wadah m 9 liter, m = 9 n = 0
Level 2 : Tuang air liter dari wadah m ke wadah n, m = 2 n = 7
Wadah m Wadah n
2       7
--------------------------------------------------------------------------
Jumlah langkah yang dilakukan dalam IDS kedalaman 3 adalah 6
--------------------------------------------------------------------------
--------------------------------------------------------------------------
--------------------------------------------------------------------------
Urutan Solusi Langkah
--------------------------------------------------------------------------
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah m 9 liter, m = 9 n = 0
Level 2 : Tuang air liter dari wadah m ke wadah n, m = 2 n = 7
--------------------------------------------------------------------------
Jumlah langkah total yang dibutuhkan IDS : 10 langkah
BUILD SUCCESSFUL (total time: 5 seconds)
```

```
Output - PraktikumKB2 (run)
run:
Simulasi Wadah Air IDS Search
--------------------------------------------------------------------------
Input ukuran wadah m    : 5
Input ukuran wadah n    : 8
Input target air        : 3
--------------------------------------------------------------------------
--------------------------------------------------------------------------
Pencarian Untuk kedalaman = 3
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 8 liter, m = 0 n = 8
Level 2 : Tuang air dari wadah n ke wadah m, m = 5 n = 3
Wadah m Wadah n
5       3
--------------------------------------------------------------------------
Jumlah langkah yang dilakukan dalam IDS kedalaman 3 adalah 3
--------------------------------------------------------------------------
--------------------------------------------------------------------------
--------------------------------------------------------------------------
Urutan Solusi Langkah
--------------------------------------------------------------------------
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 8 liter, m = 0 n = 8
Level 2 : Tuang air dari wadah n ke wadah m, m = 5 n = 3
--------------------------------------------------------------------------
Jumlah langkah total yang dibutuhkan IDS : 7 langkah
BUILD SUCCESSFUL (total time: 4 seconds)
```


Output - PraktikumKB2 (run)
run:
Simulasi Wadah Air IDS Search
Input ukuran wadah m : 11
Input ukuran wadah n : 7
Input target air : 7
Pencarian Untuk kedalaman = 1
Level 0 : Initial State, kedua wadah kosong
Error: Not Possible
Jumlah langkah yang dilakukan dalam IDS kedalaman 1 adalah 1
Pencarian Untuk kedalaman = 2
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 7 liter, m = 0 n = 7
Wadah m Wadah n
0 7
Jumlah langkah yang dilakukan dalam IDS kedalaman 2 adalah 2
Urutan Solusi Langkah
Level 0 : Initial State, kedua wadah kosong
Level 1 : Isi penuh wadah n 7 liter, m = 0 n = 7
Jumlah langkah total yang dibutuhkan IDS : 3 langkah
BUILD SUCCESSFUL (total time: 12 seconds)