



# PETUNJUK PRAKTIKUM

EDISI KURIKULUM OBE  
KECERDASAN BUATAN

Penyusun:  
Sri Winiarti, S.T., M.Cs.  
Miftahurrahma Rosyda, S.Kom, M.Eng.

2022

## HAK CIPTA

### PETUNJUK PRAKTIKUM KECERDASAN BUATAN

**Copyright© 2022,**

Sri Winiarti, S.T., M.Cs.

Miftahurrahma Rosyda, S.Kom, M.Eng.

#### **Hak Cipta dilindungi Undang-Undang**

Dilarang mengutip, memperbanyak atau mengedarkan isi buku ini, baik sebagian maupun seluruhnya, dalam bentuk apapun, tanpa izin tertulis dari pemilik hak cipta dan penerbit.

#### **Diterbitkan oleh:**

**Program Studi Informatika**

Fakultas Teknologi Industri

Universitas Ahmad Dahlan

Jalan Ring Road Selatan, Tamanan, Banguntapan, Bantul Yogyakarta 55166

#### **Penulis**

: Sri Winiarti, S.T., M.Cs.

Miftahurrahma Rosyda, S.Kom, M.Eng.

#### **Editor**

: Laboratorium Informatika, Universitas Ahmad Dahlan

#### **Desain sampul**

: Laboratorium Informatika, Universitas Ahmad Dahlan

#### **Tata letak**

: Laboratorium Informatika, Universitas Ahmad Dahlan

#### **Ukuran/Halaman**

: 21 x 29,7 cm / 110 halaman

#### **Didistribusikan oleh:**



#### **Laboratorium Informatika**

Universitas Ahmad Dahlan

Jalan Ring Road Selatan, Tamanan, Banguntapan, Bantul Yogyakarta 55166

Indonesia

## KATA PENGANTAR

Kurikulum berbasis *Outcome Based Education* (OBE) telah berlaku di prodi Informatika Universitas Ahmad Dahlan telah ditetapkan dan diberlakukan pelaksanaannya mulai Semester Gasal T.A 2021/2022. Inti dari pelaksanaan kurikulum OBE adalah mengukur capaian dalam setiap proses pembelajaran dengan berpedoman pada Capaian Profil Lulusan (CPL) Prodi Informatika dan Capaian Pembelajaran Mata Kuliah (CPMK). Dalam Rencana Pembelajaran Semester (RPS) OBE yang telah disusun untuk setiap mata kuliah telah didesain metode pembelajaran, materi kuliah, referensi hingga metode asesmen yang disesuaikan dengan CPL dan CPMK. Mengacu pada pada metode asesmen yang terdapat dalam RPS setiap mata kuliah, maka perlu disediakan berbagai perangkat pembelajaran salah satunya Buku Petunjuk Praktikum.

Dampak kurikulum OBE di prodi Informatika UAD menyebabkan perubahan struktur mata kuliah dan materi pembelajaran. Perlunya penyesuaian dengan kurikulum OBE berlaku juga pada Mata kuliah Kecerdasan Buatan. Dengan adanya perbaikan RPS pada mata kuliah Kecerdasan Buatan maka perlu dilakukan penyesuaian dam materi pembelajaran mulai dari teori hingga praktek. Dengan demikian sudah dilakukan perbaikan pada pelaksanaan praktikum Kecerdasan Buatan dengan menyesuaikan dengan format Buku petunjuk praktikum Edisi OBE.

**Perbaikan yang dilakukan adalah** dengan menentukan **CPL dan CPMK** mata kuliah Kecerdasan Buatan untuk setiap pertemuan praktikum, perbaikan **pada asesmennya** yang meliputi: pre test, pelaksanaan praktikum, serta responsi. Perubahan materi dilakukan karena sudah sesuai dengan RPS OBE, namun ada penambahan **yang dilakukan pada studi kasus** yang diberikan pada setiap pertemuan.

Demikianlah informasi terkait perubahan yang dilakukan pada Buku Praktikum Kecerdasan Buatan Edisi ke 3 ini dilakukan. Harapannya proses pengukuran capaian pembelajaran dapat terukur dengan baik sesuai dengan target yang ditetapkan dalam desain RPS OBE pada Mata Kuliah Kecerdasan Buatan.

Yogyakarta, 04 Maret 2022

Penyusun




**Sri Winiarti, S.T., M.Cs**


NIY. 60020388

## DAFTAR PENYUSUN

### **Sri Winiarti, S.T., MCs.**

	<p>Penulis merupakan dosen pengampu mata Kuliah Statistika Informatika, Basis Data, Artificial Intelligence, dan Metodologi Penelitian. Pengalaman mengajar dimulai sejak tahun 2000 hingga saat ini di Prodi Informatika UAD. Bidang minat penelitian di Sistem cerdas seperti: sistem pakar, SPK Cerdas, Klasifikasi Image dan data mining. Pengalaman organisasi sebagai anggota profesi IAENG sejak 2017 hingga sekarang, pengurus APTIKOM Wilayah 5, sebagai asesor LSP pada skema Database Administrator, Asesor Audit IT SPBE RPAN 2021, pernah meraih kaprodi berprestasi tingkat Kopertis DIY. Publikasi terkait bidang yang diminati sudah banyak dipublikasikan baik nasional maupun internasional.</p>
---	--

### **Miftahurrahma Rosyda, S.Kom, M.Eng.**

	<p>Penulis merupakan Dosen di prodi Informatika UAD dengan bidang Keilmuan Sistem Cerdas. Mulai bergabung menjadi dosen di Informatika UAD sejak tahun 2019. Bidang minat riset mencakup bidang ilmu Artificial Intelligence. Mata Kuliah yang diampu kecerdasan Buatan, Basis Data dan Statistika Informatika</p>
--	--

## HALAMAN REVISI

Yang bertanda tangan di bawah ini:

Nama : Sri Winiarti, S.T., M.Cs

NIP/NIY : 60020388

Jabatan : Dosen Pengampu Mata Kuliah **Kecerdasan Buatan**

Dengan ini menyatakan pelaksanaan Revisi Petunjuk Praktikum **Kecerdasan Buatan** untuk Program Studi Informatika telah dilaksanakan dengan penjelasan sebagai berikut:

No	Keterangan Revisi	Tanggal Revisi	Nomor Modul
1	a. Perubahan pada materi yang disusun sesuai RPS, dan kasus-kasus yang diberikan. Software yang digunakan matlab, turbo prolog dan GUI	2014	1
2	1. Perubahan pada materi, yakni ditambahkan materi software agent dan hierarchy planning dengan menggunakan aplikasi matlab. Dengan demikian tools yang dipakai selain turbo prologh juga menggunakan matlab	2016	2
3	1. Perubahan software pada materi 1 – 6, dan 8 menggunakan Python 2. Penambahan materi 9 Pembuatan <i>Custom Package</i> Python 3. Materi 10 menggunakan software GUI	2019	3
4	Perubahan template modul baru dari Lab	2020	4
5	Penyesuaian kurikulum OBE dengan menambahkan pengukuran setiap CPL dan CPMK dari mata kuliah KCB	2022	5

Yogyakarta, 05 Maret 2022

Penyusun



Sri Winiarti, S.T., M.Cs

NIY. 60020388

## HALAMAN PERNYATAAN

Yang bertanda tangan di bawah ini:

Nama : Lisna Zahrotun, S.T., M.Cs.

NIK/NIY : 60150773

Jabatan : Kepala Laboratorium Informatika

Menerangkan dengan sesungguhnya bahwa Petunjuk Praktikum ini telah direview dan akan digunakan untuk pelaksanaan praktikum di Semester Gasal Tahun Akademik 2021/2022 di Laboratorium Praktikum Informatika, Program Studi Informatika, Fakultas Teknologi Industri, Universitas Ahmad Dahlan.

Yogyakarta, 5 Maret 2022

Mengetahui,  
Ketua Kelompok Keilmuan Sistem Cerdas



**Dewi Soyusiawaty, S.T., M.T**  
NIY. 60040497

Kepala Laboratorium Praktikum  
Informatika



**Lisna Zahrotun, S.T., M.Cs.**  
NIY. 60150773

## VISI DAN MISI PRODI INFORMATIKA

### VISI

Menjadi Program Studi Informatika yang diakui secara internasional dan unggul dalam bidang Informatika serta berbasis nilai-nilai Islam.

### MISI

1. Menjalankan pendidikan sesuai dengan kompetensi bidang Informatika yang diakui nasional dan internasional
2. Meningkatkan penelitian dosen dan mahasiswa dalam bidang Informatika yang kreatif, inovatif dan tepat guna.
3. Meningkatkan kuantitas dan kualitas publikasi ilmiah tingkat nasional dan internasional
4. Melaksanakan dan meningkatkan kegiatan pengabdian masyarakat oleh dosen dan mahasiswa dalam bidang Informatika.
5. Menyelenggarakan aktivitas yang mendukung pengembangan program studi dengan melibatkan dosen dan mahasiswa.
6. Menyelenggarakan kerja sama dengan lembaga tingkat nasional dan internasional.
7. Menciptakan kehidupan Islami di lingkungan program studi.

## TATA TERTIB LABORATORIUM INFORMATIKA

### DOSEN/KOORDINATOR PRAKTIKUM

1. Dosen harus hadir saat praktikum minimal 15 menit di awal kegiatan praktikum dan menandatangani presensi kehadiran praktikum.
2. Dosen membuat modul praktikum, soal seleksi asisten, pre-test, post-test, dan responsi dengan berkoordinasi dengan asisten dan pengampu mata praktikum.
3. Dosen berkoordinasi dengan koordinator asisten praktikum untuk evaluasi praktikum setiap minggu.
4. Dosen menandatangani surat kontrak asisten praktikum dan koordinator asisten praktikum.
5. Dosen yang tidak hadir pada slot praktikum tertentu tanpa pemberitahuan selama 2 minggu berturut-turut mendapat teguran dari Kepala Laboratorium, apabila masih berlanjut 2 minggu berikutnya maka Kepala Laboratorium berhak mengganti koordinator praktikum pada slot tersebut.

### PRAKTIKAN

1. Praktikan harus hadir 15 menit sebelum kegiatan praktikum dimulai, dan dispensasi terlambat 15 menit dengan alasan yang jelas (kecuali asisten menentukan lain dan patokan jam adalah jam yang ada di Laboratorium, terlambat lebih dari 15 menit tidak boleh masuk praktikum & dianggap Inhal).
2. Praktikan yang tidak mengikuti praktikum dengan alasan apapun, wajib mengikuti INHAL, maksimal 4 kali praktikum dan jika lebih dari 4 kali maka praktikum dianggap GAGAL.
3. Praktikan harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
  - a. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki-laki / Perempuan) dan Topi harus Dilepas.
  - b. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
  - c. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki-laki / Perempuan).
  - d. Laki-laki tidak boleh memakai gelang, anting-anting ataupun aksesoris Perempuan.
4. Praktikan tidak boleh makan dan minum selama kegiatan praktikum berlangsung, harus menjaga kebersihan, keamanan dan ketertiban selama mengikuti kegiatan praktikum atau selama berada di dalam laboratorium (tidak boleh membuang sampah sembarangan baik kertas, potongan kertas, bungkus permen baik di lantai karpet maupun di dalam ruang CPU).
5. Praktikan dilarang meninggalkan kegiatan praktikum tanpa seizin Asisten atau Laboran.
6. Praktikan harus meletakkan sepatu dan tas pada rak/loker yang telah disediakan.
7. Selama praktikum dilarang NGENET/NGE-GAME, kecuali mata praktikum yang membutuhkan atau menggunakan fasilitas Internet.
8. Praktikan dilarang melepas kabel jaringan atau kabel power praktikum tanpa sepengetahuan laboran
9. Praktikan harus memiliki FILE Petunjuk praktikum dan digunakan pada saat praktikum dan harus siap sebelum praktikum berlangsung.
10. Praktikan dilarang melakukan kecurangan seperti mencontek atau menyalin pekerjaan praktikan yang lain saat praktikum berlangsung atau post-test yang menjadi tugas praktikum.
11. Praktikan dilarang mengubah setting software/hardware komputer baik menambah atau mengurangi tanpa permintaan asisten atau laboran dan melakukan sesuatu yang dapat merugikan laboratorium atau praktikum lain.



12. Asisten, Koordinator Praktikum, Kepala laboratorium dan Laboran mempunyai hak untuk menegur, memperingatkan bahkan meminta praktikan keluar ruang praktikum apabila dirasa anda mengganggu praktikan lain atau tidak melaksanakan kegiatan praktikum sebagaimana mestinya dan atau tidak mematuhi aturan lab yang berlaku.
13. Pelanggaran terhadap salah satu atau lebih dari aturan diatas maka Nilai praktikum pada pertemuan tersebut dianggap 0 (NOL) dengan status INHAL.

### ASISTEN PRAKTIKUM

- a. Asisten harus hadir 15 Menit sebelum praktikum dimulai (konfirmasi ke koordinator bila mengalami keterlambatan atau berhalangan hadir).
- b. Asisten yang tidak bisa hadir WAJIB mencari pengganti, dan melaporkan kepada Koordinator Asisten.
- c. Asisten harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
  - a. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki-laki / Perempuan) dan Topi harus Dilepas.
  - b. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
  - c. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki-laki / Perempuan).
  - d. Laki-laki tidak boleh memakai gelang, anting-anting ataupun aksesoris Perempuan.
- d. Asisten harus menjaga kebersihan, keamanan dan ketertiban selama mengikuti kegiatan praktikum atau selama berada di laboratorium, menegur atau mengingatkan jika ada praktikan yang tidak dapat menjaga kebersihan, ketertiban atau kesopanan.
- e. Asisten harus dapat merapikan dan mengamankan presensi praktikum, Kartu Nilai serta tertib dalam memasukan/Input nilai secara Online/Offline.
- f. Asisten harus dapat bertindak secara profesional sebagai seorang asisten praktikum dan dapat menjadi teladan bagi praktikan.
- g. Asisten harus dapat memberikan penjelasan/pemahaman yang dibutuhkan oleh praktikan berkenaan dengan materi praktikum yang diasistensi sehingga praktikan dapat melaksanakan dan mengerjakan tugas praktikum dengan baik dan jelas.
- h. Asisten tidak diperkenankan mengobrol sendiri apalagi sampai membuat gaduh.
- i. Asisten dimohon mengkoordinasikan untuk meminta praktikan agar mematikan komputer untuk jadwal terakhir dan sudah dilakukan penilaian terhadap hasil kerja praktikan.
- j. Asisten wajib untuk mematikan LCD Projector dan komputer asisten/praktikan apabila tidak digunakan.
- k. Asisten tidak diperkenankan menggunakan akses internet selain untuk kegiatan praktikum, seperti Youtube/Game/Medsos/Streaming Film di komputer praktikan.

### LAIN-LAIN

1. Pada Saat Responsi Harus menggunakan Baju Kemeja untuk Laki-laki dan Perempuan untuk Praktikan dan Asisten.
2. Ketidakhadiran praktikum dengan alasan apapun dianggap INHAL.
3. Izin praktikum mengikuti aturan izin SIMERU/KULIAH.
4. Yang tidak berkepentingan dengan praktikum dilarang mengganggu praktikan atau membuat keributan/kegaduhan.
5. Penggunaan lab diluar jam praktikum maksimal sampai pukul 21.00 dengan menunjukkan surat ijin dari Kepala Laboratorium Prodi Teknik Informatika.

Yogyakarta, 5 Maret 2022

Kepala Laboratorium Praktikum  
Informatika



Lisna Zahrotun, S.T., M.Cs.  
NIY. 60150773

## DAFTAR ISI

HAK CIPTA .....	1
KATA PENGANTAR .....	2
DAFTAR PENYUSUN .....	3
HALAMAN REVISI .....	4
HALAMAN PERNYATAAN .....	5
VISI DAN MISI PRODI INFORMATIKA .....	6
TATA TERTIB LABORATORIUM INFORMATIKA .....	7
DAFTAR ISI .....	10
DAFTAR GAMBAR .....	11
DAFTAR TABEL .....	12
SKENARIO PRAKTIKUM SECARA DARING .....	13
PRAKTIKUM 1: KONSEP DASAR PELACAKAN .....	15
PRAKTIKUM 2: IMPLEMENTASI PELACAKAN BUTA .....	26
PRAKTIKUM 3: IMPLEMENTASI PELACAKAN HEURISTIK .....	34
PRAKTIKUM 4: IMPLEMENTASI REPRESENTASI PENGETAHUAN .....	43
PRAKTIKUM 5: IMPLEMENTASI PENALARAN/REASONING .....	51
PRAKTIKUM 6: IMPLEMENTASI <i>HIERARCHICAL PLANNING</i> .....	58
PRAKTIKUM 7: IMPLEMENTASI <i>MULTY AGENT</i> .....	64
PRAKTIKUM 8: IMPLEMENTASI JARINGAN SEMANTIK .....	70
PRAKTIKUM 9: IMPLEMENTASI FAKTOR KETIDAK PASTIAN .....	77
PRAKTIKUM 10: IMPLEMENTASI <i>CUSTOM PACKAGE</i> PYTHON .....	86
PRAKTIKUM 11: IMPLEMENTASI KECERDASAN BUATAN (SISTEM PAKAR, JARINGAN SYARAF TIRUAN) 94	
DAFTAR PUSTAKA .....	109

## DAFTAR GAMBAR

Gambar 1.1 Contoh Silsilah Keluarga .....	20
Gambar 1.2 Contoh Silsilah Keluarga .....	23
Gambar 2.1 Sistem Kecerdasan Buatan .....	27
Gambar 3.2 Peta Kota .....	37
Gambar 3.3 Peta Kota .....	40
Gambar 4.1 Daftar operator dan simbol pada program logika proposisi .....	48
Gambar 5.1 Peta Kota .....	52
Gambar 5.2 Pohon Pelacakan Menggunakan BFS .....	53
Gambar 5.3 Solusi Permasalahan BFS .....	53
Gambar 5.4 Pohon pelacakan untuk kasus ke-2 BFS .....	55
Gambar 6.1 Keluaran Program Berupa Detail dari Aksi <code>goto_airport</code> .....	60
Gambar 11.1 Tampilan Form Utama GUI Design Studio .....	96
Gambar 11.2 Form untuk memilih tempat penyimpanan New Project .....	97
Gambar 11.3 Komponen untuk membangun interface .....	97
Gambar 11.4 Tampilan lembar kerja tempat membuat desain interface .....	98
Gambar 11.5 Rancangan menu utama .....	98
Gambar 11.6 Image Task Bar .....	99
Gambar 11.7 Tampilan Menu Bar Properties .....	99
Gambar 11.8 Rancangan Judul Pada Menu Utama .....	99
Gambar 11.9 Rancangan Menu Utama dengan Popup Menu .....	100
Gambar 11.10 Tampilan Popup Menu Properties .....	100
Gambar 11.11 Rancangan menu login .....	100
Gambar 11.12 Image Button .....	101
Gambar 11.13 Tampilan Text Button Properties .....	101
Gambar 11.14 Rancangan menu data penyakit .....	101
Gambar 11.16 Rancangan menu data gejala .....	102
Gambar 11.17 Rancangan menu data penyebab .....	103
Gambar 11.18 Rancangan menu data solusi .....	103
Gambar 11.19 Rancangan menu data pasien .....	104
Gambar 11.20 Rancangan menu aturan .....	104
Gambar 11.21 Rancangan menu konsultasi .....	104
Gambar 11.22 Rancangan menu rekam .....	105
Gambar 11.23 Rancangan menu lanjut .....	105
Gambar 11.24 Rancangan menu cetak hasil .....	106
Gambar 10.25 Rancangan menu help .....	106

## DAFTAR TABEL

## SKENARIO PRAKTIKUM SECARA DARING

Nama Mata Praktikum : Kecerdasan Buatan  
 Jumlah Pertemuan : 12 Teori + 1 Responsi

**TABEL SKENARIO PRAKTIKUM DARING**

Pertemuan ke	Judul Materi	Waktu (Lama praktikum sampai pengumpulan posttest)	Skenario Praktikum dari pemberian pre-test, post-test dan pengumpulannya serta mencantumkan metode yang digunakan misal video, whatsapp group, Google meet atau lainnya
1	Konsep Pelacakan Dengan Python	90 menit	<ol style="list-style-type: none"> <li>1. <b>1 kelas google class room</b> dibuat sesuai jadwal praktikum yang sudah tersedia (persesi).</li> <li>2. Jika satu kelas praktikum 40 praktikan, maka dibagi menjadi 4 kelompok, <b>Kelompok praktikum dibagi menjadi 10 praktikan</b> per room dengan 1 asisten pendamping untuk setiap kelompok.</li> <li>3. <b>Pret test</b> diberikan selama <b>15 menit</b> dengan menggunakan google form dan toleransi waktu 10 menit</li> <li>4. <b>Materi praktikum</b> didistribusikan melalui Google Class Room dalam bentuk PPT dan video</li> <li>5. <b>Penyampaian materi</b> akan dibuat video penyampaian materi dengan <b>durasi 15 menit</b> oleh asisten penanggung jawab</li> <li>6. Pelaksanaan <b>Praktikum</b> diberikan dengan mengerjakan sesuai langkah-langkah praktikum dengan <b>durasi 30 menit</b></li> <li>7. <b>Post Test</b> diberikan dan diselaikan pengerjaannya dengan durasi jam melalui google class room dengan <b>durasi 30 menit</b></li> <li>8. <b>Tugas dosen pengampu:</b> membuat soal pre test, membuat soal response, berkoordinasi dengan setiap asisten sesuai sesi dan kelompoknya</li> <li>9. <b>Tugas asisten:</b> membuat video penjelasan materi, koreksi pre test, pelaksanaan praktikum dan post test</li> </ol>
2	Implementasi Pencarian Buta	90 menit	<b>Sama dengan minggu 1</b>
3	Implementasi Pencarian Heuristik	90 menit	<b>Sama dengan minggu 1</b>
4	Representasi Pengetahuan	90 menit	<b>Sama dengan minggu 1</b>

5	Implementasi <i>Forward Chaining</i> dan <i>backward Chaining</i>	90 menit	<b>Sama dengan minggu 1</b>
6	Implementasi Representasi Pengetahuan Dengan Model Pohon	90 menit	<b>Sama dengan minggu 1</b>
7	<i>Hierarchy Planning</i>	90 menit	<b>Sama dengan minggu 1</b>
8	<i>Multi Agent</i>	90 menit	<b>Sama dengan minggu 1</b>
9	Implementasi representasi dengan semantic Net	90 menit	<b>Sama dengan minggu 1</b>
10	Implementasi Probabilistik dalam AI	90 menit	<b>Sama dengan minggu 1</b>
11	Pembuatan Package dalam Python	90 menit	<b>Sama dengan minggu 1</b>
12	Merancang Aplikasi Dalam Kecerdasan Buatan (Sistem Pakar, Jaringan Syaraf Tiruan)	90 menit	<b>Sama dengan minggu 1</b>

## PRAKTIKUM 1: KONSEP DASAR PELACAKAN

**Pertemuan ke** : 1

**Total Alokasi Waktu** : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 30 menit
- Post-Test : 30 menit

**Total Bobot Penilaian** : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

**Pemenuhan CPL dan CPMK:**

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas
CPMK-01	Mampu merumuskan ruang masalah yang efisien untuk masalah yang diekspresikan dalam bentuk model graph dan Pohon
CPMK-02	Mampu menjelaskan konsep mesin Turing dan melakukan analisa untuk penalaran mesin dan penalaran manusia untuk menentukan karakteristik masalah dengan sistem cerdas

### 1.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mengetahui cara kerja pemrograman Phyton sebagai salah satu bahasa pemrograman yang populer saat ini dalam bidang kecerdasan buatan.
2. Mampu menerapkan teori strategi algoritma, struktur data, statistika inferensi untuk memecahkan permasalahan dalam proses pelacakan ke dalam Bahasa pemrograman

### 1.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-03	CPMK-01	Kemampuan mahasiswa dalam menjelaskan konsep dasar pelacakan
	CPMK-02	Kemampuan mahasiswa dalam implementasi konsep dasar pelacakan dengan membuat program sederhana dalam bahasa pemrograman Phyton



### 1.3. TEORI PENDUKUNG

#### A. Sejarah Python

1. Python merupakan bahasa pemrograman yang multiguna karena bisa digunakan untuk berbagai macam aplikasi seperti pengembangan website, komputasi matematika, antar-muka grafis, dan lain-lain.
2. Dikembangkan oleh Guido Van Rossum sejak tahun 1980-an dan pertama kali dirilis pada Februari 1991

#### B. Perbedaan Python dengan Bahasa AI Lainnya

Dalam bidang kecerdasan buatan Prolog dan Lisp merupakan bahasa yang populer terutama di kalangan akademisi. Prolog populer di Eropa, sedangkan di Amerika peneliti mengembangkan bahasa pemrograman yang mirip, yaitu LISP. Akhir-akhir ini Python muncul sebagai bahasa pemrograman yang sangat populer dalam bidang kecerdasan buatan.

Tidak seperti Prolog yang menggunakan konsep *logic programming*, Python dapat mendukung beberapa konsep pemrograman, seperti pemrograman berorientasi objek, pemrograman imperatif, pemrograman fungsional, pemrograman prosedural. Banyaknya dukungan jenis pemrograman ini dikarenakan sintaks bahasa pemrograman Python sangat simpel dan fleksibel dengan dukungan *package* yang sangat komprehensif. Komunitas *open-source* yang sangat besar dan aktif menghasilkan banyak sekali *package* yang dapat digunakan untuk memecahkan permasalahan-permasalahan dalam bidang kecerdasan buatan. Contohnya TensorFlow yang dikembangkan oleh tim *Google Brain*, Scikit-learn yang banyak digunakan untuk *machine learning*, Keras untuk pengembangan *neural network*, NLTK untuk pemrosesan bahasa alami, dan masih banyak lagi.

### 1.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Pemrograman Python (Jupyter note, Collabs, Anaconda)

### 1.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	1. Jelaskan konsep pelacakan dalam AI ada berapa jenis!	25
			2. Jelaskan fungsi mesin inferensi dalam suatu pelacakan solusi!	35
			3. Jelaskan 2 perbedaan model pelacakan graph dengan model pelacakan pohon!	40
Total				100

## 1.6. LANGKAH PRAKTIKUM

Sebelum melakukan praktikum mahasiswa memastikan bahwa computer sudah tersedia package-package yang dipakai dalam praktikum untuk bahasa pemrograman Phyton. Jika belum tersedia, maka perlu dilakukan instalasi package dalam Phyton. Karena dalam praktikum AI akan menggunakan Phyton, maka praktikan harus mengetahui cara instalasi phyton beserta package-package yang digunakan.

Salah satu keunggulan Python adalah dukungan package yang sangat beragam dan luas. Oleh karena itu, aplikasi-aplikasi yang dikembangkan menggunakan Python dapat menyesuaikan kompleksitasnya dengan memanfaatkan package yang tersedia di Python. Instalasi package dapat dilakukan secara bertahap tergantung dengan kebutuhan aplikasi.

### 1. Instalasi Package

Penambahan *package* ke dalam Python dapat dilakukan melalui perintah `pip install <nama package>`. Apabila `pip` belum terinstal, lakukan langkah berikut:

- a. Unduh `get-pip.py` melalui tautan berikut: <https://bootstrap.pypa.io/get-pip.py>.
- b. Jalankan perintah berikut untuk instalasi `pip`:

```
python get-pip.py
```

- c. Apabila `pip` telah terinstal, package Python dapat diinstal menggunakan perintah sebagai berikut:

```
pip install <nama_package>
```

*Package* yang telah didaftarkan pada PyPI (repository resmi untuk package python) dapat diinstal secara langsung dengan memanggil nama *packagenya*. Contohnya jika ingin menginstal *package* `numpy` untuk komputasi numerik di Python, maka perintah berikut ini dapat dijalankan:

```
pip install numpy
```

### 2. Melakukan Pemanggilan Module di dalam Package Python

*Module* merupakan file berekstensi `.py` yang memuat kode Python dan berisi beberapa fungsi maupun `Class`. *Module* dapat disatukan dalam satu *package* supaya kumpulan *module* tersebut menjadi terstruktur dan dapat didistribusikan secara mudah. Pemanggilan *module* dapat dilakukan menggunakan “dot/titik”, contohnya apabila ingin menggunakan *module* `core` dalam `numpy`, maka dapat dilakukan dengan `numpy.core`

Konten dari *module* dapat diakses dengan melakukan *import*. Ada tiga cara untuk melakukan *import module*:

- `import <nama_module>`  
contoh:  
`import numpy.core`  
`subtract(3,2)`
- `from <nama_module> import <nama_obyek>`  
contoh: `from numpy.core import subtract`  
`subtract(3,2)`
- `from <nama_module> import <nama_obyek> as <inisial>`  
contoh:  
`from numpy.core import subtract as sub`  
`sub(3, 2)`

### 3. Instalasi Package Kanren (LogPy)

Pada dasarnya *package* standar Python tidak mendukung *logic programming* seperti bahasa AI lain seperti Prolog. Oleh karena itu, *package* atau modul *logic programming* dibutuhkan untuk menerapkan konsep pelacakan dalam Python. Salah satu *package* yang digunakan adalah Kanren (dulu disebut LogPy). Dengan menggunakan Kanren, kita bisa mengekspresikan relasi, fakta dan melakukan *query* untuk menemukan nilai yang diinginkan. Kanren menyimpan data dalam bentuk fakta-fakta yang menjelaskan relasi di antara istilah-istilah.

Instalasi Kanren dapat dilakukan menggunakan `pip`:

```
pip install kanren
```

Setelah terinstal, *package* Kanren dapat digunakan dalam program dengan melakukan *import* pada program.

### 4. Menggunakan Modul Logika dalam Package Kanren

#### Variabel Logika

Variabel logika adalah variabel yang bisa memuat sembarang nilai, tetapi hanya satu nilai dalam satu waktu. Cara penggunaan variabel logika menggunakan Kanren adalah dengan mendeklarasikan variabel menggunakan `var`. Contohnya `x = var()` dimana `x` merupakan variabel logika.

## Constraint dan Goal

*Constraint* adalah ekspresi yang membatasi nilai dari variabel logika.

Contoh 1:

Setelah mendeklarasikan `x = var()`, variabel `x` dapat menampung sembarang nilai. Apabila ingin membatasi atau memberi *constraint* bahwa nilai variabel `x` adalah sama dengan 5, maka *constraint* variabel `x` dapat ditulis menjadi `eq(x, 5)`. *Goal* menyatakan hasil akhir yang ingin dicapai.

Contoh 2:

Misalnya variabel `y` dideklarasikan sebagai variabel menggunakan cara yang sama dengan contoh 1, `y=var()`. *Goal* yang ingin dicapai adalah persamaan `x, y) = (y, 3)`. Pada persamaan tersebut *goal* dapat ditulis menggunakan sintaks `eq`, yang menyatakan bahwa kedua espresi adalah sama (*equal*). *Goal* tersebut dapat ditulis sebagai berikut `eq((x, y), (y, 3))`.

## Ekspresi Logika (Logic Expression)

Ekspresi logika terdiri dari kumpulan variabel logika dan kumpulan *constraint* terhadap nilai dari variabel logika tersebut. Apabila `x` adalah variabel maka tiap ekspresi logika mengandung *constraint* yang membatasi nilai dari variabel `x`.

Berdasarkan contoh 1, ekspresi logika dapat dituangkan ke dalam program sebagai berikut:

```
1 from kanren.core import var, eq, run
2 x = var()
3 output = run(1, x, eq(5, x))
4 print(output)
```

Listing 1.1 Kode Program untuk contoh 1

Pertama, *import* sintaks dari *package* Kanren yang akan dipakai dalam program yaitu `var`, `eq`, dan `run` seperti pada baris pertama. Selanjutnya deklarasi variabel logika `x` seperti yang ditulis pada baris 2. Kemudian ekspresi logika yang dijalankan dengan menggunakan sintaks `run` pada baris 3, menunjukkan bahwa program tersebut meminta 1 nilai, yaitu variabel `x`, dengan *constraint* `x = 5`. Hasil program ditampilkan pada baris 4 sehingga keluaran dari program tersebut adalah sebagai berikut: **(5,)** Contoh 2 dapat dituliskan ke dalam program seperti berikut:

```
1 from kanren.core import var, eq, run
2 x = var()
3 y = var()
4 output = run(1, x, eq((x, y), (y, 3)))
5 print(output)
```

Listing 1.2 Kode program untuk contoh 2

Deklarasi variabel dan *import package* hampir sama dengan program pada contoh 1. Bedanya sekarang ada dua variabel yang dideklarasikan yaitu  $x$  dan  $y$ . Selanjutnya sintaks pada baris 4 menunjukkan bahwa program tersebut meminta 1 nilai, yaitu variabel  $x$ , dengan *goal* `eq((x, y), (y, 3))`. Dengan kata lain, cari nilai variabel  $x$  yang memenuhi *constraint*  $y == 3$  dan memenuhi syarat  $(x, y) = (y, 3)$  sehingga apabila dijalankan program tersebut akan menampilkan keluaran sebagai berikut: **(3,)**

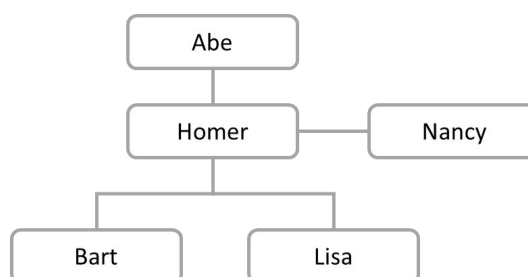
### Fakta dan Relasi

Deklarasi fakta dan relasi menggunakan Kanren mirip dengan deklarasi menggunakan bahasa Prolog. Contoh penggunaan fakta dan relasi menggunakan Kanren dapat dilihat pada tabel 1.

Tabel 1.1 Fakta dan Relasi

Fakta dalam kalimat	Relasi	Fakta
Abe is the father of Homer	father	<code>fact(father("Abe", "Homer"))</code>
California is a coastal	coastal	<code>fact(coastal("California"))</code>
California is adjacent to Arizona	adjacent	<code>fact(adjacent("California", "Arizona"))</code>

Contoh kasus silsilah keluarga seperti pada Gambar 1.1 dapat digunakan untuk memperjelas penggunaan fakta dan relasi dalam program Python menggunakan Kanren. Fakta dan relasi dapat dibuat menggunakan acuan seperti pada Tabel 1.1.



Gambar 1.1 Contoh Silsilah Keluarga

Relation dan facts diimport dari *package* Kanren pada baris 1 bersamaan dengan `var` dan `run` yang digunakan untuk menampung relasi, fakta, variabel logika, dan eksekusi ekspresi logika secara berurutan. Relasi yang dipakai pada contoh ini adalah `father`. Daftar fakta pada contoh silsilah keluarga menggunakan relasi `father` dapat dilihat pada Tabel 1.2.

Tabel 1.2 Fakta Silsilah Keluarga

Fakta dalam kalimat	Fakta
Homer is the father of Bart	<code>fact(father( "Homer", "Bart"))</code>
Homer is the father of Lisa	<code>fact(father( "Homer", "Lisa"))</code>
Abe is the father of Homer	<code>fact(father( "Abe", "Homer"))</code>

Deklarasi relasi `father` ditulis pada baris 3. Karena fakta-fakta pada Tabel 1.2 menggunakan relasi yang sama yaitu `father`, maka kumpulan fakta tersebut dapat dituliskan menjadi satu menggunakan `facts` seperti yang dituliskan pada baris 4.

```

1 from kanren.facts import Relation, facts
2 from kanren.core import var, run

3 father = Relation()
4 facts(father, ("Homer", "Bart"),
          ("Homer", "Lisa"),
          ("Abe", "Homer"))

5 x = var()
6 output = run(1, x, father(x, "Bart"))
7 print("\nNama ayah Bart : ", output[0])

```

*Listing 1.3 Kode Program untuk Silsilah Keluarga pada Gambar 1.1*

Baris 5 variabel logika dan baris 6 menunjukkan eksekusi ekspresi logika untuk mencari ayah dari Bart. Apabila dijalankan, program dapat menghasilkan keluaran seperti berikut:

```
Nama ayah Bart : Homer
```

Relasi baru dapat dideklarasikan dengan dua cara yaitu:

Dengan membuat deklarasi relasi baru menggunakan `Relation`

Menggunakan relasi yang sudah ada untuk membentuk relasi lain tanpa membuat `Relation` baru.

Implementasi cara pertama dapat dilihat pada baris 13-19 di dalam program di bawah ini. Relasi `sibling` dideklarasikan menggunakan `Relation` kemudian perlu didefinisikan fakta baru berdasarkan relasi tersebut. Cara ini menjadi tidak efektif karena terdapat redundansi data dan apabila jumlah saudara sangat banyak maka kita harus menambahkan fakta satu persatu sesuai jumlah fakta yang ada. Untuk kasus seperti ini kita dapat menggunakan relasi yang sudah ada untuk mendeteksi relasi baru. Dalam kasus ini relasi `Parent` dapat digunakan untuk mendeteksi relasi saudara tanpa membuat relasi baru. Implementasi penggunaan relasi `Parent` untuk mendeteksi nama saudara dapat dilihat pada fungsi `get_sibling` yang ada pada baris 2-4. Pemanggilan fungsi dan tampilan keluaran dapat dilihat pada baris 20-24.

```

1 from kanren.facts import Relation, facts
  from kanren.core import var, run, conde

```

```

2  def get_sibling(x, y):
3      temp = var()
4      return conde((parent(temp, x), parent(temp, y)))

5  if __name__ == '__main__':
6      parent = Relation()
7      facts(parent, ("Homer", "Bart"),
8                  ("Homer", "Lisa"),
9                  ("Abe", "Homer"))
10     x = var()
11     output = run(1, x, parent(x, "Bart"))
12     print("\nNama ayah Bart : ", output[0])

    # contoh definisi saudara (sibling) menggunakan relasi baru
13     sibling = Relation()
14     facts(sibling, ("Bart", "Lisa"),
15           ("Lisa", "Bart"))
16     brother = run(0, x, sibling(x, "Lisa"))
17     print("\nNama saudara laki-laki Lisa : ", brother[0])
18     sister = run(0, x, sibling(x, "Bart"))
19     print("\nNama saudara perempuan Bart : ", sister[0])

    '''
        contoh definisi saudara (sibling)
        menggunakan relasi yang sudah ada (parent)
    '''
20     siblings = run(0, x, get_sibling(x, "Bart"))
21     siblings = [x for x in siblings if x != "Bart"]
22     print("\nNama saudara Bart : ")
23     for item in siblings:
24         print(item)

```

Listing 1.4 Kode Program untuk Silsilah Keluarga Menggunakan Relasi Baru

Apabila program tersebut dijalankan akan mengeluarkan keluaran seperti berikut ini:

```

Nama ayah Bart : Homer

Nama saudara laki-laki Lisa : Bart

Nama saudara perempuan Bart : Lisa

Nama saudara Bart :
Lisa

```

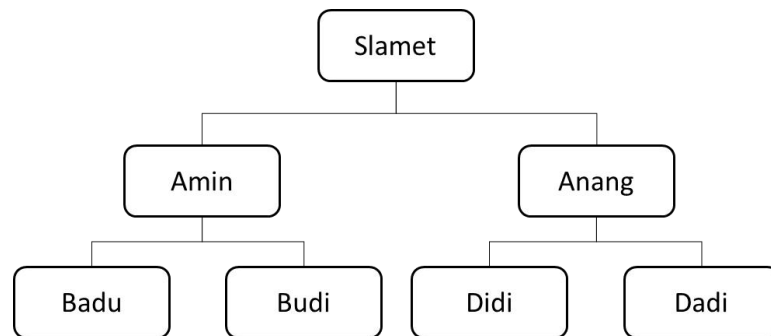
**Aturan Penilaian (Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-02	Lakukan langkah praktikum 1	Hasil pekerjaan praktikum langkah 1	25
2.	CPL-03	CPMK-02	Lakukan langkah Praktikum 2	Hasil pekerjaan praktikum langkah 2	25
3.	CPL-03	CPMK-02	Lakukan langkah Praktikum	Hasil pekerjaan praktikum langkah 3	25

4.	CPL-03	CPMK-02	Lakukan langkah Praktikum 4	Hasil pekerjaan praktikum langkah 4	25
<b>Total Nilai</b>					<b>100</b>

### 1.7. POST TEST

Perhatikan Gambar 1.2 merupakan silsilah suatu keluarga. Dengan menerapkan contoh program listing 1.2. carilah solusinya



Gambar 1.2 Contoh Silsilah Keluarga

Contoh program: Kasus Silsilah Keluarga pada gambar 1.2

```

from kanren.facts import Relation, facts
from kanren.core import var, run

parent = Relation()
facts(parent, ("Slamet", "Amin"),
        ("Slamet", "Anang"),
        ("Amin", "Badu"),
        ("Amin", "Budi"),
        ("Anang", "Didi"),
        ("Anang", "Dadi"))

x = var()
child = "Amin"
ayah = run(1, x, parent(x, child))
print("\nNama ayah " + child + ": ")
for item in ayah:
    print(item)
  
```

Listing 1.5 Contoh Program Silsilah Keluarga Menggunakan Package Kanren

Jawablah pertanyaan berikut:

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-02	<p>Coba ketik program di atas dengan bahasa Python. Caranya:</p> <ol style="list-style-type: none"> <li>Buka jendela editor sesuai pilihan</li> <li>Untuk menulis sintaks Program tulis pada jendela editor</li> <li>Ketik semua program di atas (listing 1.5)</li> <li>Run program dengan menggunakan console</li> <li>Perhatikan apa yang terjadi</li> <li>Cobalah beberapa query</li> </ol>	Hasil running program	40



			(minimal 3 query). Catatlah hasilnya		
2.	CPL-03	CPMK-02	Berdasarkan listing 1.5 cobalah kembangkan programnya, bila ditambahkan fakta kakek, anak atau paman dengan dua cara: a. Membuat relasi baru yaitu grandfather, children, dan uncle b. Tanpa membuat relasi baru (hanya gunakan relasi parent untuk membentuk relasi lain. <b>Hint:</b> lihat kembali contoh pada Listing 1.4)	Hasil running program	60
<b>Total Nilai</b>					<b>100</b>

### 1.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-02	30%		
3.	Post-Test	CPL-03	CPMK-02	50%		
<b>Total Nilai</b>						

**LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM**

<b>Nama :</b> <b>NIM :</b>	<b>Asisten:</b> <b>Paraf Asisten:</b>	<b>Tanggal:</b> <b>Nilai:</b>
-------------------------------	--	----------------------------------

--

## PRAKTIKUM 2: IMPLEMENTASI PELACAKAN BUTA

**Pertemuan ke** : 2

**Total Alokasi Waktu** : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 30 menit
- Post-Test : 30 menit

**Total Bobot Penilaian** : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

**Pemenuhan CPL dan CPMK:**

CPL - 03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas
CPMK-01	Mampu merumuskan ruang masalah yang efisien untuk masalah yang diekspresikan dalam bentuk model graph dan Pohon
CPL- 05	Mampu mengkaji / menganalisis implikasi pengembangan atau implementasi ilmu pengetahuan teknologi, menyusun deskripsi saintifik hasil kajian untuk pemecahan masalah dengan mempertimbangkan multidisiplin ilmu
CPMK-03	Mampu menerapkan konsep pelacakan secara heuristik maupun pelacakan buta yang sesuai untuk mencari solusi masalah kedalam suatu bahasa pemrograman

### 2.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mengetahui cara kerja pemrograman Phyton sebagai salah satu bahasa pemrograman yang populer saat ini dalam bidang kecerdasan buatan.
2. Mampu menerapkan teori strategi algoritma, struktur data, statistika inferensi untuk memecahkan permasalahan dalam proses pelacakan ke dalam Bahasa pemrograman

### 2.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

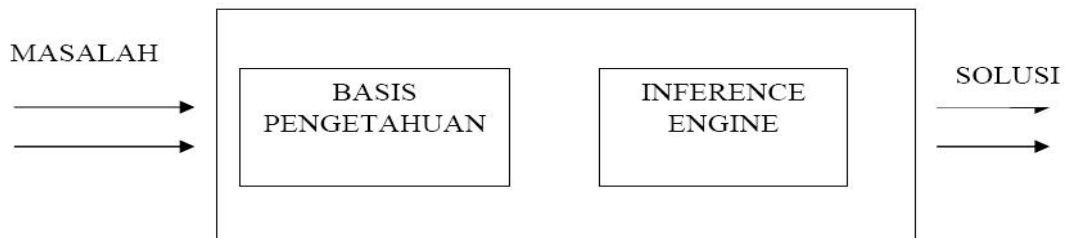
Indikator ketercapaian diukur dengan:

CPL-03	CPMK-01	Kemampuan mahasiswa dalam menjelaskan konsep dasar pelacakan Buta
CPL-05	CPMK-03	Kemampuan mahasiswa dalam menerapkan konsep pelacakan buta dalam program sederhana dengan menggunakan Pemrograman phyton

## 2.3. TEORI PENDUKUNG

### a. Representasi Masalah

Seperti telah diketahui pada sistem yang menggunakan kecerdasan buatan akan mencoba memberikan output berupa solusi suatu masalah berdasarkan kumpulan pengetahuan yang ada. Hal tersebut direpresentasikan pada Gambar 2.1.



Gambar 2.1 Sistem Kecerdasan Buatan

Berdasarkan Gambar 2.1, *input* yang diberikan pada sistem yang menggunakan kecerdasan buatan berupa masalah. Pada sistem harus dilengkapi dengan sekumpulan pengetahuan yang ada pada basis pengetahuan (*knowledge base*). Sistem harus memiliki *inference engine* agar sistem mampu mengambil kesimpulan berdasarkan fakta atau pengetahuan. *Output* yang diberikan berupa solusi masalah sebagai hasil dari inferensi.

Secara umum untuk membangun sistem yang mampu menyelesaikan masalah perlu mempertimbangkan 4 hal:

1. Mendefinisikan masalah dengan tepat. Pendefinisian ini mencakup spesifikasi yang tepat mengenai keadaan awal (*initial state*) dan solusi yang diharapkan.
2. Menganalisis masalah serta mencari beberapa titik penyelesaian masalah yang sesuai.
3. Merepresentasikan pengetahuan yang perlu untuk menyelesaikan masalah tersebut.
4. Memilih teknik penyelesaian masalah yang terbaik.

### b. Penyelesaian Masalah dalam AI

Dalam penyelesaian masalah dengan teknik AI menyangkut beberapa langkah yaitu:

1. Analisa Masalah
2. Representasi Masalah dan Pengetahuan
3. Inferensi
4. Penggunaan Bahasa AI

Dalam menyelesaikan masalah dalam AI, perlu melakukan analisa masalah sebagai langkah pertama. Langkah ini menganalisa masalah yang dihadapi dan mengungkapkan masalah tersebut dalam satu sistem simbol. Sistem tersebut dapat merupakan *diagram*, *skema*, *graf*, atau simbol- simbol yang lain. Sistem simbol ini harus diterjemahkan dalam bahasa

pemrograman AI. Sistem ini harus dapat mengungkapkan dengan tepat keadaan awal (*initial state*). Keadaan akhir atau sasaran yang dituju (*Goal State*). Misal contoh pedagang mengunjungi 10 kota. Keadaan awal adalah rute perjalanan yang ada dan dapat dilukiskan sebagai berikut:

$$R(K_1, K_2, \dots, K_N)$$

$$R(K_{J1}, K_{J2}, \dots, K_{JN})$$

Dengan jarak kota  $K_1$  ke kota  $K_{J1}$  adalah  $d_{ij}$ .

Dimana  $K_{J1}, K_{J2}$  adalah kota-kota dalam daftar pedagang tersebut.

Keadaan sasaran adalah salah satu rute perjalanan yang mempunyai jumlah  $d_{ij}$  minimum.

Secara umum pendefinisian masalah sebagai suatu ruang keadaan meliputi 3 hal yaitu:

1. Posisi Awal (*Initial State*)
2. Aturan (*Rule*)
3. Tujuan (*Goal*)

#### Contoh

Misal permasalahan yang dihadapi adalah “Permainan Catur”, maka harus ditentukan:

1. Posisi awal pada papan catur  
Posisi awal setiap permainan catur selalu sama, yaitu semua bidak diletakkan di atas papan catur dalam 2 sisi yaitu kubu putih dan kubu hitam.
2. Aturan- aturan untuk melakukan gerakan secara ilegal (*Rule*)  
Aturan- aturan (*rule*) berguna untuk menentukan gerakan suatu bidak, yaitu melangkah dari satu keadaan ke keadaan lain. Misal untuk mempermudah menunjukkan posisi bidak, setiap kotak ditunjukkan dalam huruf (a, b, c, d, e, f, g, h) pada arah horizontal dan angka (1, 2, 3, 4, 5, 6, 7, 8) pada arah vertikal. Suatu aturan untuk menggerakkan bidak dari posisi (e,2) ke (e,4) dapat ditunjukkan dengan aturan:  

```

IF bidak putih pada kotak (e,2) ,
    AND Kotak (e,3) Kosong,
    AND Kotak (e,4) Kosong,
    THEN Gerakkan bidak dari (e,2) ke (e,4)

```
3. Tujuan (*Goal*)  
Tujuan yang ingin dicapai adalah posisi pada papan catur yang menunjukkan kemenangan seseorang terhadap lawannya. Kemenangan ini ditandai dengan posisi RAJA yang sudah bergerak lagi.

## 2.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Perangkat Lunak: Bahasa dan Pemrograman Phyton
2. Perangkat Keras: Komputer dan periperalnya, modul Praktikum.

## 2.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	1. Jelaskan ada berapa jenis pelacakan Buta!	25
			2. Jelaskan kelebihan serta kekurangan pelacakan buta!	25
			3. Jelaskan langkah-langkah pencarian solusi dengan pelacakan Buta untuk jenis Deep First Search lengkapi dengan contoh graphnya!	50
			<b>Total</b>	<b>100</b>

## 2.6. LANGKAH PRAKTIKUM

Dalam praktikum ini menggunakan bahasa pemrograman Phyton. Para praktikan diminta menjalankan bahasa pemrograman Phyton yang sudah diinstal package sesuai praktikum minggu pertama. Mahasiswa dipastikan duduk sesuai dengan computer yang digunakan pada praktikum pertemuan sebelumnya.

1. Peserta duduk di kursi dan komputer yang telah disediakan (ini dipakai untuk seterusnya hingga praktikum terakhir.
2. Praktikan wajib membaca materi praktikum sebelum dilaksanakan
3. Praktikan mengikuti *pre-test* yang diberikan oleh Asistem praktikum
4. Memastikan bahwa perangkat lunak yang digunakan yaitu pemrograman Phyton telah tersedia.
5. Mengerjakan semua tugas praktikum sesuai dengan waktu yang disediakan.
6. **Tugas Praktik:** Ketiklah program Python berikut ini:

```
# Program aktivitas 1
from kanren.facts import Relation, facts, fact
from kanren.core import var, run
from kanren.goals import membero

suka = Relation()

facts(suka, ("ellen", "tenis"),
        ("john", "football"),
        ("john", "tenis"),
        ("mary", "renang"),
        ("tom", "tenis"),
        ("tom", "basket"),
        ("eric", "renang"),
        ("mary", "tenis"))
```

```

x = var()
tom_hobbies = run(0, x, suka("tom", x))
print("Tom: ", tom_hobbies)

for hobby in tom_hobbies:
    fact(suka, ("bill"), hobby)
bill_hobbies = run(0, x, suka("bill", x))
print("Bill: ", bill_hobbies)

mary_hobbies = run(0, x, suka("mary", x))
print("Mary: ", mary_hobbies)

for hobby in mary_hobbies:
    fact(suka, ("ann"), hobby)
ann_hobbies = run(0, x, suka("ann", x))
print("Ann: ", ann_hobbies)

```

*Listing 2.1 Program Hobi*

**7. Tugas selanjutnya: ketikkan listing 2.2 berikut:**

```

# Program aktivitas 2

from kanren.facts import Relation, facts, fact
from kanren.core import var, run
from kanren.goals import membero

suka = Relation()

facts(suka, ("ellen", "tenis"),
        ("john", "football"),
        ("mary", "renang"),
        ("tom", "tenis"),
        ("eric", "renang"))

x = var()
tom_hobbies = run(0, x, suka("tom", x))
print("Tom: ", tom_hobbies)

for hobby in tom_hobbies:
    fact(suka, ("bill"), hobby)
bill_hobbies = run(0, x, suka("bill", x))
print("Bill: ", bill_hobbies)

mary_hobbies = run(0, x, suka("mary", x))
print("Mary: ", mary_hobbies)

for hobby in mary_hobbies:
    fact(suka, ("ann"), hobby)
ann_hobbies = run(0, x, suka("ann", x))
print("Ann: ", ann_hobbies)

```

*Listing 2.2 Program Hobi 2*

**8. Ketikkan program pada listing 2.3 berikut ini dan perhatikan hasilnya!**

```

from kanren.facts import Relation, facts, fact
from kanren.core import var, run
from kanren.goals import membero
from kanren import vars

ukuran = Relation()
warna = Relation()
gelap = Relation()

```

```

facts(ukuran, ("beruang", "besar"),
      ("gajah", "besar"),
      ("kucing", "kecil"))

facts(warna, ("beruang", "cokelat"),
      ("kucing", "hitam"),
      ("gajah", "kelabu"))

fact(gelap, "hitam")
fact(gelap, "cokelat")

x = var()
kecil = run(0, z, ukuran(z, "kecil"))
print("hewan berukuran kecil: ", kecil)

```

*Listing 2.3 Contoh program Python*

**Aturan Penilaian (Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-05	CPMK-03	Buatlah sebuah contoh implementasi python untuk kasus pelacakan	Hasil pekerjaan praktikum langkah 6	25
2.	CPL-05	CPMK-03	<p>Lakukan tugas praktikum langkah 7.</p> <p>a. Bagimanakah hasilnya..?</p> <p>b. Berdasarkan program pada listing 2.2 tersebut berikanlah goal di bawah ini untuk kedua program tersebut. Bagaimana hasilnya? (hint: gunakan <code>membero</code>)</p> <p>Query 1:  <code>suka("ellen", X), suka("tom", X)</code></p> <p>Query 2:  <code>suka("mary", X), suka("ann", X)</code></p> <p>c. Cobalah untuk menambahkan fakta-fakta baru untuk pengembangan program pada listing 2.2. Bagaimana hasilnya..?</p>	Hasil pekerjaan praktikum langkah 7	25
3.	CPL-05	CPMK-03	<p>Lakukan langkah Praktikum 8.</p> <p>a. Jalankan program tersebut dengan menambahkan <i>goal</i> sebagai berikut:  <code>ukuran(z, "besar").</code>  Tulis ke dalam program dan tampilkan keluaran program  <code>warna(z, "cokelat").</code>  Tulis ke dalam program dan tampilkan keluaran program  Tambahkan kode ke dalam program untuk menampilkan binatang yang</p>	Hasil pekerjaan praktikum langkah 8	15



			berukuran <u>besar</u> <b>dan</b> berwarna <u>cokelat</u>		
			b. Tambahkan kode ke dalam program untuk menampilkan binatang yang berwarna <u>gelap</u>		15
			c. Setelah mencoba query tersebut, cobalah untuk menambahkan fakta dan relasi baru pada program tersebut. Tambahkan relasi jenis dan fakta jenis("beruang", "karnivora") dan jenis("kucing", "karnivora"). Tambahkan kode ke dalam program untuk menampilkan binatang berjenis karnivora.		20
<b>Total Nilai</b>					<b>100</b>

## 2.7. POST TEST

Berdasarkan listing program 2.3 cobalah lakukan modifikasi untuk membuat menu sederhana pemilihan menu makan di sebuah restoran. Variable dan fakta serta relation silahkan diatur.

Jawablah pertanyaan berikut:

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-05	CPMK-03	Tampilan kodingnya serta running. Apakah sukses..? jika belum perbaiki lakukan hingga program berhasil running.	Hasil running program	60
2.	CPL-05	CPMK-03	Buatlah beberapa query yang bisa menampilkan solusi	Hasil running program	40
<b>Total Nilai</b>					<b>100</b>

## 2.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-05	CPMK-03	30%		
3.	Post-Test	CPL-05	CPMK-03	50%		
<b>Total Nilai</b>						

**LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM**

<b>Nama :</b> <b>NIM :</b>	<b>Asisten:</b> <b>Paraf Asisten:</b>	<b>Tanggal:</b> <b>Nilai:</b>
-------------------------------	--	----------------------------------

--

## PRAKTIKUM 3: IMPLEMENTASI PELACAKAN HEURISTIK

**Pertemuan ke** : 3

**Total Alokasi Waktu** : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 30 menit
- Post-Test : 30 menit

**Total Bobot Penilaian** : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

**Pemenuhan CPL dan CPMK:**

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas
CPMK-01	Mampu merumuskan ruang masalah yang efisien untuk masalah yang diekspresikan dalam bentuk model graph dan Program sederhana
CPL- 05	Mampu mengkaji / menganalisis implikasi pengembangan atau implementasi ilmu pengetahuan teknologi, menyusun deskripsi saintifik hasil kajian untuk pemecahan masalah dengan mempertimbangkan multidisiplin ilmu
CPMK-03	Mampu menerapkan konsep pelacakan secara heuristik maupun pelacakan buta dan heuristik yang sesuai untuk mencari solusi masalah kedalam suatu bahasa pemrograman

### 3.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mengetahui cara kerja pemrograman Phyton sebagai salah satu bahasa pemrograman yang populer saat ini dalam bidang kecerdasan buatan.
2. Mampu menerapkan teori strategi algoritma, struktur data, statistika inferensi untuk memecahkan permasalahan dalam proses pelacakan ke dalam Bahasa pemrograman

### 3.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-03	CPMK-01	Kemampuan mahasiswa dalam menjelaskan konsep dasar pelacakan Heuristik
CPL-05	CPMK-03	Kemampuan mahasiswa dalam menerapkan konsep pelacakan Heuristik dalam program sederhana dengan menggunakan Pemrograman phyton

### 3.3. TEORI PENDUKUNG

Metode pencarian dan pelacakan adalah hal penting dalam kecerdasan buatan atau kita sebut dengan AI. Dalam kecerdasan buatan lebih di fokuskan dalam hal pencarian, karena AI harus dapat mencari solusi / jawaban atas suatu permasalahan dalam sekumpulan kemungkinan ruang keadaan.

Terdapat 4 kriteria yang dapat mengukur dalam hal pencarian dalam kecerdasan buatan:

1. *Completeness*
2. *Time complexity*
3. *Space Complexity*
4. *Optimality*

Di dalam kecerdasan buatan terdapat 2 metode pencarian dan pelacakan diantaranya adalah:

1) Pencarian buta (Blind Search): Dalam metode pencarian buta ini dibagi menjadi 2 yaitu:

- a. *Breadth - First Search*: Metode ini akan mulai mencari dari node yang paling kiri, kemudian berpindah ke-node se-level dengannya, dan berulang - ulang trus hingga menemukan solusi yang dimaksud.

Keuntungan metode *Breadth-First Search* adalah: pasti menemukan solusi yang dicari, tidak akan mengalami jalan buntu / tidak menemukan solusi.

- b. *Depth - First Search*: Metode ini dimulai dari semua node-node anaknya kemudian berpindah ke node-node se-level nya.

Kelemahan metode *Breadth-First Search* adalah: memerlukan memori yang cukup besar, karena metode ini mengecek keseluruhan node yang ada dan membutuhkan waktu yang lebih untuk mengecek semua node yang ada tersebut.

2) Pencarian terbimbing (Heuristic search):

Istilah *Heuristic* diambil dari bahasa Yunani yang berarti menemukan. *Heuristic* merupakan suatu strategi untuk melakukan proses pencarian (search) ruang problema secara selektif, yang memandu proses pencarian yang kita lakukan disepanjang jalur yang memiliki kemungkinan sukses paling besar. Dalam metode pencarian terbimbing atau kita sebut *Heuristic search* terbagi menjadi 4 macam yaitu:

- a. Pembangkit & Pengujian

Pembangkit & Pengujian: Pada prinsipnya metode ini merupakan penggabungan antara *depth-first search* dengan pelacakan mundur (backtracking), yaitu bergerak ke belakang menuju pada suatu keadaan awal. Terdapat kelemahan dalam metode ini:

1. Perlu membangkitkan semua kemungkinan sebelum dilakukan pengujian
2. Membutuhkan waktu yang cukup lama dalam pencariannya

b. *Hill Climbing*

Metode ini merupakan jenis pelacakan heuristik, karena dalam pencarian solusinya selalu mempertimbangkan node yang memiliki nilai terbaik atau dengan model bertahap.

1. Merupakan metode pelacakan yang mengkombinasikan pelacakan *Generate and Test* dengan *Backtracking*.
2. Untuk langkah awal pelacakan dengan memilih node yang memiliki nilai terbaik/terbesar (fungsi heuristik).

c. *Best First Search* (BFS)

d. *Simulated Annealing*

### 3.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Perangkat Lunak: Bahasa dan Pemrograman Python
2. Perangkat Keras: Komputer dan periperalnya, modul Praktikum.

### 3.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	1. Jelaskan perbedaan pelacakan Heuristik dan pelacakan Buta!	25
			2. Berilah contoh kasus penerapan pelacakan heuristic dalam kehidupan nyata!	30
			3. Buatlah contoh kasus graph pelacakan dengan salah satu metode Heuristik!	45
Total				100

### 3.6. LANGKAH PRAKTIKUM

Untuk dapat menjalankan praktikum untuk metode pelacakan *heuristic*, perlu dilakukan langkah persiapan oleh semua praktikan. Langkah-langkahnya sebagai berikut:

#### 1. Melakukan Kustomisasi Python *Package* untuk Pencarian

Untuk materi praktikum ini dan seterusnya akan digunakan *custom package* yang mendefinisikan struktur data dan fungsi-fungsi dasar yang ada pada materi modul praktikum. *Custom package* dikemas dalam bentuk Python *wheels* yang merupakan standar distribusi *package* Python. Langkahnya sebagai berikut:

a. *Instalasi*

Untuk melakukan instalasi cukup dengan mengunduh *package wheels* dengan ekstensi `.whl` dan mengeksekusi perintah selanjutnya instalasi *package* dari server Python.

```
pip install <nama_file_whl>
```

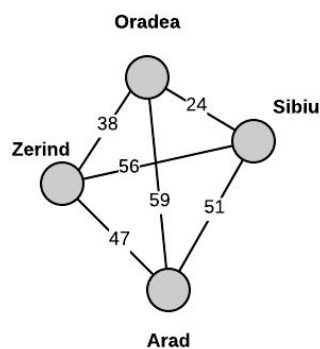
b. Melakukan strukturisasi Modul Pencarian dalam *Package*

*Custom package* yang akan digunakan untuk praktikum bernama `ai_pkg`. Di dalam modul ini terdapat beberapa modul, salah satu diantaranya adalah modul `search`. Dalam modul ini terdapat tiga class yang mendefinisikan struktur data yang digunakan untuk metode pencarian.

```
ai_pkg/
...
    search.py
        Graph <class>
        Node <class>
        Problem
<class>
    utils.py
...
```

## 2. Bacalah kasus pencarian dengan metode Hill Climbing

Berikut adalah contoh pengaplikasian pelacakan dengan metode *Hill Climbing* untuk menjawab permasalahan *Traveling Salesman Problem* (TSP). Permasalahannya adalah apabila diberikan beberapa kota dan diketahui jarak antara masing-masing kota, maka solusinya adalah rute terpendek yang bisa menjangkau semua kota dan kembali pada kota asal. Lihat Gambar 3.2.



Gambar 3.2 Peta Kota

Permasalahan TSP:

Ada 4 kota yaitu Zerind, Oradea, Sibiu, dan Arad dan jarak antar kota dapat dilihat pada gambar 3.1. Asumsi jarak antar kota adalah simetris sehingga jarak dari Zerind ke Oradea dan sebaliknya adalah sama yaitu 38.

Kasus TSP dapat dimodelkan sebagai graf berbobot dan tidak berarah (*undirected weighted graph*). Tiap kota direpresentasikan sebagai “simpul” (vertex atau *node*) dan tiap jalur antar kota direpresentasikan sebagai “sisi” (*edge*).

```
city_map = Graph(dict(
    Oradea=dict(Oradea=0, Sibiu=24, Arad=59, Zerind=38),
    Sibiu=dict(Oradea=24, Sibiu=0, Arad=51, Zerind=56),
    Arad=dict(Oradea=59, Sibiu=51, Arad=0, Zerind=47),
    Zerind=dict(Oradea=38, Sibiu=56, Arad=47, Zerind=0)),
    directed=False)
```

*Listing 3.1 Kode untuk Representasi Graf Kota pada Gambar 3.1*

Pertama, `Graph` dibentuk dengan parameter berupa `dict` dan `directed`. Nama kota dan jarak masing-masing kota dapat didefinisikan secara langsung menggunakan `dict`. Pada kasus TSP digunakan graf tidak berarah maka parameter `directed` diatur menjadi `False`. Dengan membuat objek `Graph` baru, maka simpul graf akan terbentuk otomatis menggunakan class `Node` yang ada pada module `search`. Representasi graf kota pada gambar 3.1 dapat dituliskan seperti pada Listing 3.1.

```
distances = {}
class TSP_problem(Problem):
    def generate_neighbour(self, state):
        neighbour_state = state[:]
        left = random.randint(0, len(neighbour_state) - 1)
        right = random.randint(0, len(neighbour_state) - 1)
        if left > right:
            left, right = right, left
        neighbour_state[left:right + 1] =
reversed(neighbour_state[left:right + 1])
        return neighbour_state

    def actions(self, state):
        return [self.generate_neighbour]

    def result(self, state, action):
        return action(state)

    def path_cost(self, state):
        cost = 0
        for i in range(len(state) - 1):
            current_city = state[i]
            next_city = state[i + 1]
            cost += distances[current_city][next_city]
        cost += distances[state[0]][state[-1]]
        return cost

    def value(self, state):
        return -1 * self.path_cost(state)
```

*Listing 3.2 Kode untuk Definisi Problem TSP*

Selanjutnya untuk buat permasalahan TSP dengan membuat *class* dengan nama `TSP_problem` yang mengimplementasikan *abstract class* `Problem` pada *package* yang digunakan. Terdapat empat fungsi dasar yaitu: `actions`, `result`, `path_cost`, dan `value`. Keempat fungsi dasar ini harus diimplementasikan karena digunakan dalam eksekusi TSP di dalam class `Node`.

Kode untuk metode Hill Climbing dapat dilihat pada Listing 3.3.

```
def hill_climbing(problem):
    def find_neighbors(state, number_of_neighbors=100):
        neighbors = []
        for i in range(number_of_neighbors):
            new_state = problem.generate_neighbour(state)
            neighbors.append(Node(new_state))
            state = new_state
        return neighbors

    current = Node(problem.initial)
    while True:
        neighbors = find_neighbors(current.state)
        if not neighbors:
            break
        neighbor = argmax_random_tie(neighbors, key=lambda node:
problem.value(node.state))
        if problem.value(neighbor.state) <= problem.value(current.state):
            break
        current.state = neighbor.state
    return current.state
```

*Listing 3.3 Kode untuk Metode Hill Climbing*

Potongan program untuk ditambahkan pada kasus Traveling Salesman Problem (TSP) pada listing 3.1.

```
if __name__ == '__main__':
    all_cities = []
    cities_graph = city_map.graph_dict
    for city_1 in cities_graph.keys():
        distances[city_1] = {}
        if(city_1 not in all_cities):
            all_cities.append(city_1)
        for city_2 in cities_graph.keys():
            if(cities_graph.get(city_1).get(city_2) is not None):
                distances[city_1][city_2] = cities_graph.get(city_1).get(city_2)
```

*Listing 3.4 Kode untuk Mendefinisikan Jarak Kota*



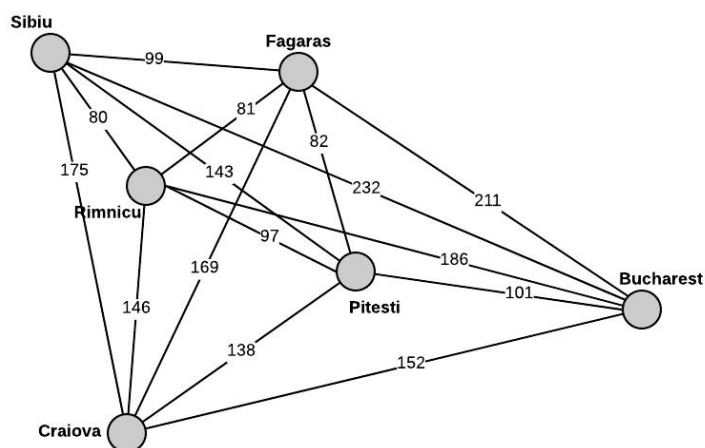
Jawablah pertanyaan berikut:

**Aturan Penilaian (Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-05	CPMK-03	Lakukanlah tugas 1 pada langkah 1	Hasil pekerjaan praktikum langkah 1	30
2.	CPL-05	CPMK-03	Ketik pkode program listing 3.1 untuk kasus Traveling Salesman Problem (TSP)	Hasil pekerjaan praktikum langkah 2	20
3.	CPL-05	CPMK-03	Jalankan program Traveling Salesman Problem dengan cara: <ol style="list-style-type: none"> <li>Unduh <i>custom package</i> melalui URL yang diberikan asisten dan instal <i>package</i> tersebut dan <i>import</i> modul yang ada pada <i>package</i> untuk menggunakannya dalam program.</li> <li>Tuliskan program yang ada pada contoh kasus gambar 3.1 dan tambahkan listing 3.4 untuk menjalankan program</li> <li>Program TSP dapat dijalankan dengan cara membuat objek baru <code>TSP_problem</code> dengan parameter <code>all_cities</code>. Selanjutnya panggil fungsi <code>hill_climbing</code> dengan parameter objek yang baru saja dibuat.</li> <li>Jalankan program tersebut dan tampilkan hasilnya</li> </ol>	Hasil implementasi koding	50
<b>Total Nilai</b>					<b>100</b>

#### a. POST TEST

Lakukanlah tugas selanjutnya berdasarkan gambar 3.3. dengan menjawab pertanyaan:



Gambar 3.3 Peta Kota

Jawablah pertanyaan berikut:

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-05	CPMK-03	Buatlah program TSP dengan menggunakan data kota seperti gambar berikut dan catat hasilnya	Hasil running program	60
2.	CPL-05	CPMK-03	Lakukan beberapa percobaan catatlah hasilnya	Hasil running program	40
<b>Total Nilai</b>					<b>100</b>

#### b. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-05	CPMK-03	30%		
3.	Post-Test	CPL-05	CPMK-03	50%		
<b>Total Nilai</b>						

**LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM**

<b>Nama :</b> <b>NIM :</b>	<b>Asisten:</b> <b>Paraf Asisten:</b>	<b>Tanggal:</b> <b>Nilai:</b>
-------------------------------	--	----------------------------------

--

## PRAKTIKUM 4: IMPLEMENTASI REPRESENTASI PENGETAHUAN

**Pertemuan ke** : 4

**Total Alokasi Waktu** : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 30 menit
- Post-Test : 30 menit

**Total Bobot Penilaian** : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

**Pemenuhan CPL dan CPMK:**

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas
CPMK-01	Mampu merumuskan ruang masalah yang efisien untuk masalah yang diekspresikan dalam bentuk model graph dan Program sederhana
CPL-07	Mampu memilih, membuat dan menerapkan teknik, sumber daya, penggunaan perangkat teknik modern dan implementasi teknologi informasi untuk memecahkan masalah
CPMK-04	Mampu menerapkan berbagai model representasikan pengetahuan untuk memecahkan masalah ke dalam suatu aplikasi AI

### 4.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mampu merepresentasikan pengetahuan untuk menyajikan pengetahuan dengan berbagai model dalam konsep kecerdasan buatan.
2. Mampu menjelaskan model representasi untuk menyajikan pengetahuan dalam bentuk Aturan Production If—Then, Frame, List, Pohon keputusan, grap, logika proposisi dan Tabel Keputusan.

## 4.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-03	CPMK-01	Kemampuan mahasiswa dalam menjelaskan konsep dasar Representasi pengetahuan untuk penyelesaian masalah AI dengan baik
CPL-07	CPMK-04	Kemampuan mahasiswa dalam menerapkan konsep representasi pengetahuan yang sesuai untuk menentukan solusi dalam bahasa pemrograman Python

## 4.3. TEORI PENDUKUNG

Pengetahuan (*Knowledge*) didefinisikan sebagai fakta atau kondisi sesuatu atau keadaan yang timbul karena suatu pengalaman. Pengetahuan merupakan Cabang ilmu filsafat, yaitu *Epistemology*, berkenaan dengan sifat, struktur dan keaslian dari *knowledge*. Dalam konsep AI, ada berbagai teknik menyajikan pengetahuan, yaitu:

- 1) Aturan Produksi
- 2) Jaringan Semantik
- 3) Graph
- 4) *Frame* dan *Scemata*
- 5) *Logika Proposisi*
- 6) *List*
- 7) *Tabel keputusan*

Pada praktikum pertemuan ke 4 ini, materi representasi pengetahuan yang dibahas adalah logika proposisi. Representasi ini menggunakan ekspresi-ekspresi dalam logika formal untuk merepresentasikan basis pengetahuan. Bahasa representasi harus dapat membuat seorang programmer mampu mengekspresikan pengetahuan untuk mendapatkan solusi suatu masalah.

Di dalam matematika, tidak semua kalimat berhubungan dengan logika. Hanya kalimat yang bernilai benar atau salah saja yang digunakan dalam penalaran. Kalimat tersebut dinamakan **proposisi** (*preposition*).

Proposisi adalah kalimat deklaratif yang bernilai benar (*true*) atau salah (*false*), tetapi tidak dapat sekaligus keduanya. Kebenaran atau kesalahan dari sebuah kalimat disebut nilai kebenarannya (*truth value*).

Tiga buah contoh berikut ini dapat mengilustrasikan kalimat mana yang merupakan proposisi dan mana yang bukan. Pernyataan-pernyataan berikut ini;

- (a) 6 adalah bilangan genap.
- (b) Soekarno adalah Presiden Indonesia yang pertama.

- (c)  $2 + 2 = 4$ .
- (d) Ibukota Provinsi Jawa Barat adalah Semarang.
- (e)  $12^3 \neq 19$ .
- (f) Kemarin hari hujan.
- (g) Suhu di permukaan laut adalah 21 derajat Celcius.
- (h) Pemuda itu tinggi.
- (i) Kehidupan hanya ada di planet Bumi.

Semuanya merupakan proposisi. Proposisi a, b, dan c bernilai benar, tetapi proposisi d salah karena ibukota Jawa Barat seharusnya adalah Bandung dan proposisi e bernilai salah karena seharusnya  $12^3 \neq 19$ . Proposisi f sampai i memang tidak dapat langsung ditetapkan kebenarannya, namun satu hal yang pasti, proposisi-proposisi tersebut tidak mungkin benar dan salah sekaligus. Kita bisa menetapkan nilai proposisi tersebut benar atau salah. Misalnya, proposisi f bisa kita andaikan benar (hari kemarin memang hujan) atau salah (hari kemarin tidak hujan). Demikian pula halnya untuk proposisi g dan h. Proposisi i bisa benar atau salah, karena sampai saat ini belum ada ilmuwan yang dapat memastikan kebenarannya.

Secara simbolik, proposisi biasanya dilambangkan dengan huruf kecil seperti  $p, q, r, \dots$ . Misalnya,  $p$  : 6 adalah bilangan genap.

Untuk mendefinisikan  $p$  sebagai proposisi “6 adalah bilangan genap”. Begitu juga untuk  $q$  : Soekarno adalah Presiden Indonesia yang pertama.

$r$  :  $2 + 2 = 4$ .

dan sebagainya. Kita dapat membentuk proposisi baru dengan cara mengkombinasikan satu atau lebih proposisi. Operator yang digunakan untuk mengkombinasikan proposisi disebut **operator logika**. Operator logika dasar yang digunakan adalah **dan** (*and*), **atau** (*or*), dan **tidak** (*not*). Dua operator pertama dinamakan operator **biner** karena operator tersebut mengoperasikan dua buah proposisi, sedangkan operator ketiga dinamakan operator **uner** karena ia hanya membutuhkan satu buah proposisi. Proposisi baru yang diperoleh dari pengkombinasian tersebut dinamakan **proposisi majemuk** (*compound proposition*). Proposisi yang bukan merupakan kombinasi proposisi lain disebut **proposisi atomik**. Dengan kata lain, proposisi majemuk disusun dari proposisi-proposisi atomik. Metode pengkombinasian proposisi dibahas oleh matematikawan Inggris yang bernama George Boole pada tahun 1854 di dalam bukunya yang terkenal, *The Laws of Thought*. Proposisi majemuk ada tiga macam, yaitu konjungsi, disjungsi, dan ingkaran. Ketiganya didefinisikan sebagai berikut:

Misalkan  $p$  dan  $q$  adalah proposisi. **Konjungsi** (*conjunction*)  $p$  dan  $q$ , dinyatakan dengan notasi  $p \wedge q$ , adalah proposisi  $p$  dan  $q$ . **Disjungsi** (*disjunction*)  $p$  dan  $q$ , dinyatakan dengan notasi  $p \vee q$ , adalah

proposisi  $p$  atau  $q$ . **Ingkaran** atau (*negation*) dari  $p$ , dinyatakan dengan notasi  $\sim p$ , adalah proposisi tidak  $p$ .

**Catatan:**

1. Beberapa literatur menggunakan notasi " $\neg p$ ", " $\bar{p}$ ", atau " $\text{not } p$ " untuk menyatakan ingkaran.
2. Kata "tidak" dapat dituliskan di tengah pernyataan. Jika kata "tidak" diberikan di awal pernyataan maka ia biasanya disambungkan dengan kata "benar" menjadi "tidak benar". Kata "tidak" dapat juga diganti dengan "bukan" bergantung pada rasa bahasa yang tepat untuk pernyataan tersebut.

Contoh :

Diketahui proposisi-proposisi berikut:

$p$  : Hari ini hujan

$q$  : Murid-murid diliburkan dari sekolah

maka

$p \wedge q$  : Hari ini hujan dan murid-murid diliburkan dari sekolah

$p \vee q$  : Hari ini hujan atau murid-murid diliburkan dari sekolah

$\sim p$  : Tidak benar hari ini hujan (atau dalam kalimat lain yang lebih lazim: Hari ini *tidak* hujan)

#### 4.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Perangkat Lunak: Bahasa dan Pemrograman Python
2. Perangkat Keras: Komputer dan periperalnya, modul Praktikum.

#### 4.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	1. Jelaskan yang Anda ketahui tentang Representasi Pengetahuan!	25
			2. Jelaskan peranan representasi pengetahuan dalam suatu aplikasi AI!	30
			3. Jelaskan penerapan aplikasi AI yang menggunakan representasi pengetahuan berbentuk frame, list dan Naskah! <b>(Setiap contoh bernilai 15)</b>	45
Total				100

#### 4.6. LANGKAH PRAKTIKUM

1. Peserta duduk di kursi dan kompyuer yang telah disediakan (ini dipakai untuk seterusnya hingga praktikum terakhir.
2. Praktikan wajib membaca materi praktikum sebelum dilaksanakan
3. Praktikan mengikuti *pre-test* yang diberikan oleh Asistem praktikum
4. Memastikan bahwa perangkat lunak yang digunakan yaitu pemrograman Phyton telah tersedia.
5. Mengerjakan semua tugas praktikum sesuai dengan waktu yang disediakan.
6. Ketiklah koding program logika proposisi pada listing 4.1.

```
from ai_pkg.utils import Expr

def is_prop_symbol(s):
    return isinstance(s, str) and s[:1].isalpha() and s[0].isupper()

def is_true(exp, model={}):
    if exp in (True, False):
        return exp
    op, args = exp.op, exp.args
    if is_prop_symbol(op):
        return model.get(exp)
    elif op == '~':
        p = is_true(args[0], model)
        if p is None:
            return None
        else:
            return not p
    elif op == '|':
        result = False
        for arg in args:
            p = is_true(arg, model)
            if p is True:
                return True
            if p is None:
                result = None
        return result
    elif op == '&':
        result = True
        for arg in args:
            p = is_true(arg, model)
            if p is False:
                return False
            if p is None:
                result = None
        return result
    p, q = args
    if op == '==>':
        return is_true(~p | q, model)
    elif op == '<==':
        return is_true(p | ~q, model)
    pt = is_true(p, model)
    if pt is None:
        return None
    qt = is_true(q, model)
    if qt is None:
```



```

    return None
    if op == '<=>':
        return pt == qt
    elif op == '^':
        return pt != qt
    else:
        raise ValueError("illegal operator" + str(exp))

if __name__ == '__main__':
    A, B = map(Expr, 'AB')
    model = {A: False, B: True}
    query = (A & B)
    print(query, ' : ', is_true(query, model))

```

*Listing 4.1 Kode Logika Proposisi*

Operator	Simbol
Not	~
And	&
Or	
implication	=>
if and only if	<=>

*Gambar 4.1 Daftar operator dan simbol pada program logika proposisi*

Jawablah pertanyaan berikut:

**Aturan Penilaian (Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-07	CPMK-04	Lakukanlah tugas 1 pada langkah 6 lakukan running program. Bagaimana hasilnya?	Hasil pekerjaan praktikum langkah 6	30
2.	CPL-07	CPMK-04	Tambahkan beberapa query di bawah ini ke dalam program pada listing 4.1 untuk membuktikan ekspresi logika proposisi. Catatlah hasilnya. Query 1 : not ( A and B) Query 2 : not ( A and B) or not (A or B) Query 3 : not ( A or B) and not ( A and B) Query 4: not (A or B) and not (B)	Hasil Running Query	40
3.	CPL-07	CPMK-04	Buatlah 3 query berdasarkan koding tersebut, catat hasilnya!	Hasil query	30
<b>Total Nilai</b>					<b>100</b>

#### 4.7. POST TEST

Berdasarkan Listing 4.1. Jawablah pertanyaan berikut:

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-07	CPMK-04	Dengan menggunakan program pada listing 4.1: <b>a.</b> Tambahkan Expr C dan beri nilai False kemudian berikan query berikut ke dalam program: Query 1 : (A or B) and (C or D) Query 2 : (A and B) and (C or D) <b>b.</b> Tambahkan Expr D dan beri nilai True kemudian jalankan kembali query-query yang ada pada poin 2.a dan bandingkan hasilnya.	Hasil running program	60
2.	CPL-07	CPMK-04	Lakukan beberapa percobaan beberapa query yang Anda buat catatlah hasilnya	Hasil running program	40
<b>Total Nilai</b>					<b>100</b>

#### 4.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-07	CPMK-04	30%		
3.	Post-Test	CPL-07	CPMK-04	50%		
<b>Total Nilai</b>						

**LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM**

<b>Nama :</b> <b>NIM :</b>	<b>Asisten:</b> <b>Paraf Asisten:</b>	<b>Tanggal:</b> <b>Nilai:</b>
-------------------------------	--	----------------------------------

## PRAKTIKUM 5: IMPLEMENTASI PENALARAN/REASONING

**Pertemuan ke** : 5

**Total Alokasi Waktu** : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 30 menit
- Post-Test : 30 menit

**Total Bobot Penilaian** : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

**Pemenuhan CPL dan CPMK:**

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas
CPMK-01	Mampu merumuskan ruang masalah yang efisien untuk masalah yang diekspresikan dalam bentuk model graph dan Program sederhana
CPL-07	Mampu memilih, membuat dan menerapkan teknik, sumber daya, penggunaan perangkat teknik modern dan implementasi teknologi informasi untuk memecahkan masalah
CPMK-04	Mampu menerapkan berbagai model representasikan pengetahuan untuk memecahkan masalah ke dalam suatu aplikasi AI

### 5.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mampu menjelaskan konsep penalaran untuk pelacakan solusi dengan menggunakan metode *forward chaining* maupun *backward chaining*.
2. Mampu menggunakan model penalaran dengan menyesuaikan model representasi pengetahuan yang ada.

## 5.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

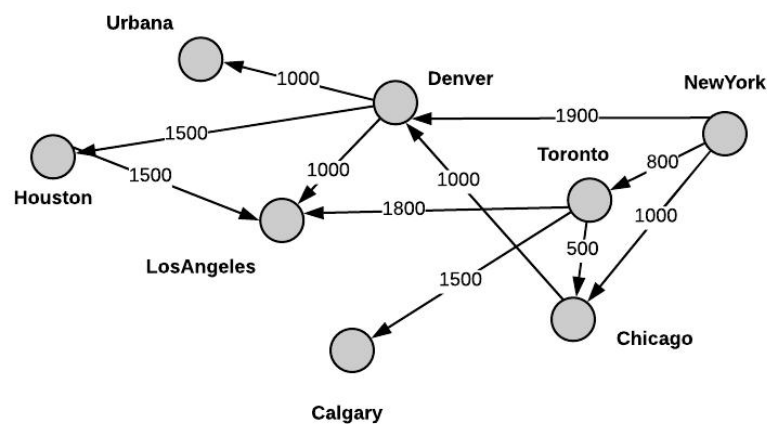
CPL-03	CPMK-01	Kemampuan mahasiswa dalam menjelaskan konsep dasar penalaran dengan metode forward chaining ataupun backward chaining untuk penyelesaian masalah AI dengan baik
CPL-07	CPMK-04	Kemampuan mahasiswa dalam menggunakan konsep penalaran untuk penyelesaian masalah dalam AI dengan menggunakan representasi pengetahuan yang sesuai dalam bahasa pemrograman Python

## 5.3. TEORI PENDUKUNG

Dalam praktikum ini akan menggunakan contoh model pohon untuk menyajikan pengetahuan. Model pohon merupakan bentuk penyajian representasi pengetahuan berbentuk grafis. Pada umumnya banyak diterapkan untuk kasus aplikasi diagnose seperti system pakar. Pada praktikum ini hanya akan dibahas beberapa saja sebagai sampel penerapan dari pelacakan dalam kecerdasan buatan.

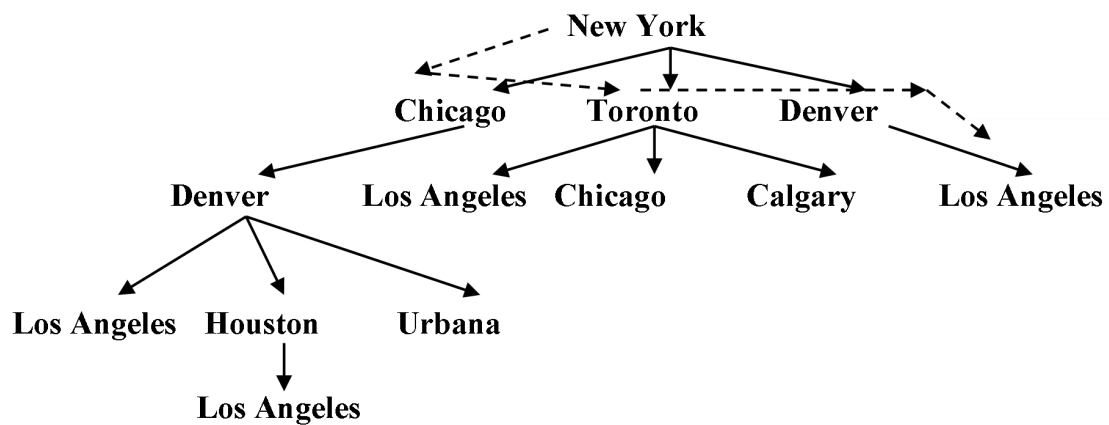
### Breadth First Search (BFS)

BFS adalah pencarian yang dimulai dari node akar terus ke level berikutnya dari kiri ke kanan, kemudian berpindah ke level berikutnya. Contoh kasus pada Gambar 5.1 menunjukkan kota dengan tipe graf terarah.



Gambar 5.1 Peta Kota

Apabila kota asal adalah New York dan kota tujuan adalah Los Angeles, maka alur pelacakan dengan metode BFS adalah seperti pada Gambar 5.2:



Gambar 5.2 Pohon Pelacakan Menggunakan BFS

Solusi dari kasus di atas dapat dilihat pada Gambar 5.3 berikut ini

```

TRACKING PATH: NewYork -> Chicago -> Toronto -> Denver -> LosAngeles
SOLUTION PATH: NewYork -> Denver -> LosAngeles
COST: 2900

```

Gambar 5.3 Solusi Permasalahan BFS

## 5.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Perangkat Lunak: Bahasa dan Pemrograman Python
2. Perangkat Keras: Komputer dan periperalnya, modul Praktikum.

## 5.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	1. Jelaskan 2 jenis metode dalam penalaran solusi dalam masalah AI!	25
			2. Dalam proses penalaran berkaitan dengan mesin inferensi sebagai solusi terbaik. Jelaskan kaitan mesin inferensi dengan proses penalaran!	25
			3. Diberikan aturan sebagai berikut: Kondisi awal: kota S Kondisi goal: kota T Jika rute yang terjadi adalah: If $S \rightarrow A \rightarrow B \rightarrow C \rightarrow D \rightarrow T$ and $S \rightarrow E \rightarrow F \rightarrow G \rightarrow T$ and $S \rightarrow H \rightarrow I \rightarrow J \rightarrow K \rightarrow L \rightarrow T$ and $S \rightarrow H \rightarrow M \rightarrow N \rightarrow K \rightarrow L \rightarrow T$ Gambarkan alur pelacakannya	45
			<b>Total</b>	<b>100</b>

## 5.6. LANGKAH PRAKTIKUM

1. Peserta duduk di kursi dan kompyuter yang telah disediakan (ini dipakai untuk seterusnya hingga praktikum terakhir.
2. Praktikan wajib membaca materi praktikum sebelum dilaksanakan
3. Praktikan mengikuti *pre-test* yang diberikan oleh Asistem praktikum
4. Memastikan bahwa perangkat lunak yang digunakan yaitu pemrograman Phyton telah tersedia.
5. Mengerjakan semua tugas praktikum sesuai dengan waktu yang disediakan.
6. Ketiklah kode program 5.1. berikut ini:

```
city_map = Graph(dict(
    NewYork=dict(Chicago=1000, Toronto=800, Denver=1900),
    Chicago=dict(Denver=1000),
    Denver=dict(LosAngeles=1000, Houston=1500, Urbana=1000),
    Houston=dict(LosAngeles=1500),
    Toronto=dict(LosAngeles=1800, Chicago=500, Calgary=1500)),
    directed=True)

class CityProblem(Problem):
    def __init__(self, initial, goal, graph):
        Problem.__init__(self, initial, goal)
        self.graph = graph

    def actions(self, A):
        return list(self.graph.get(A).keys())

    def result(self, state, action):
        return action

    def path_cost(self, cost, A, action, B):
        return cost + (self.graph.get(A, B) or infinity)

def breadth_first_search(problem):
    global track_path
    frontier = deque([Node(problem.initial)])
    explored = set()
    track_path = [problem.initial]
    while frontier:
        node = frontier.popleft()
        if problem.goal_test(node.state):
            return node
        explored.add(node.state)
        expanded = node.expand(problem)
        for child in expanded:
            track_path.append(child.state)
            if child.state not in explored and child not in frontier:
                if problem.goal_test(child.state):
                    return child
                frontier.append(child)
    return None

if __name__ == '__main__':
    track_path = []
    romania_problem = CityProblem(start, goal, city_map)
    node = breadth_first_search(romania_problem)
```

```

if node is not None:
    final_path = node.solution()
    final_path.insert(0, start)
    print('TRACKING PATH: ', ' ' -> '.join(track_path))
    print('SOLUTION PATH: ', ' ' -> '.join(final_path))

```

Listing 5.1 Kode Program Pelacakan Menggunakan BFS

Jawablah pertanyaan berikut:

**Aturan Penilaian (Total Skor: 100):**

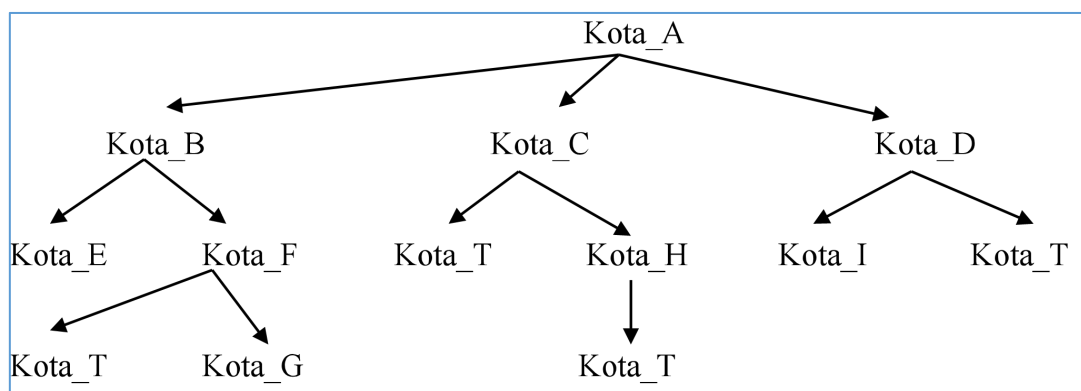
No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-07	CPMK-04	Lakukanlah tugas 1 pada langkah 6	Hasil pekerjaan praktikum langkah 6	30
2.	CPL-07	CPMK-04	Jalankan program dari listing 5.1. dengan cara:	Hasil Running Query	
			a. Unduh <i>custom package</i> melalui URL yang diberikan asisten dan instal <i>package</i> tersebut dan <i>import</i> modul yang ada pada <i>package</i> untuk menggunakannya dalam program		20
			b. Pada kode pada listing 5.1 lakukan penambahan kota asal "NewYork" dan kota tujuan "LosAngeles". Jalankan program dan tunjukkan hasilnya.		30
			c. Cobalah menggunakan kota tujuan lain dan tunjukkan hasilnya		20
Total Nilai					100

## 5.7. POST TEST

Berdasarkan Listing 5.1.lakukanlah modifikasi program dari listing 5.1. tersebut dengan mengacu pada pohon pelacakan berikut ini :

Kota asal : Kota\_A

Tujuan : Kota\_T



Gambar 5.4 Pohon pelacakan untuk kasus ke-2 BFS



No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-07	CPMK-04	Lakukan running program dari listing 5.1 yang telah dimodifikasi untuk kasus Gambar 5.4. Bagaimana hasilnya?	Hasil running program	70
2.	CPL-07	CPMK-03	Lakukan beberapa percobaan beberapa query yang Anda buat catatlah hasilnya	Hasil running program	30
<b>Total Nilai</b>					<b>100</b>

## 5.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-07	CPMK-04	30%		
3.	Post-Test	CPL-07	CPMK-04	50%		
<b>Total Nilai</b>						

**LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM**

<b>Nama :</b> <b>NIM :</b>	<b>Asisten:</b> <b>Paraf Asisten:</b>	<b>Tanggal:</b> <b>Nilai:</b>
-------------------------------	--	----------------------------------

--

## PRAKTIKUM 6: IMPLEMENTASI *HIERARCHICAL PLANNING*

**Pertemuan ke** : 6

**Total Alokasi Waktu** : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 30 menit
- Post-Test : 30 menit

**Total Bobot Penilaian** : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

**Pemenuhan CPL dan CPMK:**

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas
CPMK-01	Mampu merumuskan ruang masalah yang efisien untuk masalah yang diekspresikan dalam bentuk model graph dan Program sederhana
CPL-08	Mampu merancang dan mengimplementasikan algoritma/metode dalam mengidentifikasi dan memecahkan masalah yang melibatkan perangkat lunak dan pemikiran komputasi
CPMK-05	Mampu menganalisa masalah dengan menerapkan berbagai metode penalaran untuk menarik kesimpulan dalam mencapai solusi dan mengimplementasikannya dalam suatu aplikasi AI

### 5.4. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mampu menjelaskan konsep dasar *Hierchical Planning* dalam menyelesaikan masalah dalam AI.
2. Mampu menerapkan metode *hierarchy planning* ke dalam bahasa pemrograman Phyton

### 5.5. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-03	CPMK-01	Kemampuan mahasiswa dalam menjelaskan konsep dasar <i>Hierchical Planning</i> untuk penyelesaian masalah AI dengan baik
CPL-08	CPMK-05	Mampu menganalisa masalah dengan menerapkan <i>Hierchical Planning</i> untuk menarik kesimpulan dalam mencapai solusi dan mengimplementasikannya dalam suatu aplikasi AI

## 5.6. TEORI PENDUKUNG

*Hierarchical planning* digunakan untuk membangun rencana yang efisien karena dapat membagi masalah ke dalam level abstrak sebelum dilanjutkan kepada langkah yang lebih mendetail. Dengan menggunakan struktur hirarki, setiap level pada hirarki dapat dibagi menjadi langkah-langkah yang lebih kecil.

### *High-Level Action (HLA)*

Tiap HLA bisa mempunyai satu atau lebih *refinement* berupa rangkaian aksi yang lebih detail. Contohnya: aksi “Pergi ke bandara” bisa direpresentasikan dalam HLA `Go(Home, Airport)`. Untuk dapat pergi ke bandara dapat dilakukan dengan dua cara, yaitu dengan menyetir mobil sendiri atau naik taksi. Apabila opsi menyetir mobil dipilih maka aksi tersebut dapat dibagi lagi menjadi langkah yang lebih kecil yaitu menyetir dari rumah ke parkiran bandara, kemudian naik bis *shuttle* dari parkiran bandara ke terminal di bandara. Jadi HLA `Go(Home, Airport)` mempunyai dua *refinement* yaitu:

- `Drive(Home, AirportParking)` dan `Shuttle(AirportParking, Airport)`
- `Taxi(Home, Airport)`

### Planning Problem

Untuk menyelesaikan permasalahan *Hierarchical Planning* bisa digunakan modul planning di dalam *custom package*. Masukan dari algoritma *planning* adalah Problem dan hierarchy. hierarchy adalah kumpulan semua action (aksi) dan urutan aksi-aksi tersebut.

Seperti yang sudah dijelaskan pada contoh kasus pergi ke bandara, untuk pergi ke bandara dapat dilakukan dengan dua cara yaitu menyetir sendiri ke bandara atau menggunakan taksi. Kedua aksi tersebut mempunyai beberapa kondisi awal (*precondition*) dan akibat (*effect*). Apabila kita menggunakan taksi maka kondisi awal kita harus mempunyai uang tunai, sementara apabila kita ingin menyetir sendiri maka kita harus mempunyai mobil. hierarchy dari aksi-aksi yang dapat dilakukan dapat dituliskan sebagai berikut:

```
library = {
  'HLA': ['Go(Home,Airport)',
          'Go(Home,Airport)',
          'Drive(Home, AirportParking)',
          'Shuttle(AirportParking, Airport)',
          'Taxi(Home, Airport)'],
  'steps': [['Drive(Home,AirportParking)', 'Shuttle(AirportParking,
Airport)'],
            ['Taxi(Home, Airport)'],
            [],
            [],
            []],
  'precond': [['At(Home) & Have(Car)'],
              ['At(Home)'],
              ['At(Home) & Have(Car)'],
              ['At(AirportParking)'],
              ['At(Home)']]
```

```
'effect': [['At(Airport) & ~At(Home)'],
           ['At(Airport) & ~At(Home) & ~Have(Cash)'],
           ['At(AirportParking) & ~At(Home)'],
           ['At(Airport) & ~At(LongTermParking)'],
           ['At(Airport) & ~At(Home) & ~Have(Cash)']] }
```

Listing 6.1 Kode Hirarki untuk Kasus Pergi ke Airport

Aksi dengan level paling atas yang bisa dilakukan adalah pergi ke bandara (`goto_airport`). Aksi `goto_airport` dapat dituliskan seperti pada Listing 6.2.

```
goto_airport = HLA('Go(Home, Airport)', precondition='At(Home)', effect='At(Airport) & ~At(Home)')
```

Listing 6.2 Kode Aksi `goto_airport`

Setelah mendefinisikan hirarki dan aksi maka langkah selanjutnya adalah membuat Problem baru. Problem dapat dibentuk seperti pada Listing 6.3. Saat membuat Problem baru, parameter yang diperlukan adalah kondisi awal, kondisi akhir, dan aksi yang dilakukan. Pada kasus ini kondisi awalnya adalah berada di rumah 'At(Home) dan mempunyai uang Have(Cash). Kondisi akhir (*goal*) yang diharapkan adalah berada di bandara At(Airport) dan masih mempunyai uang Have(Cash).

```
PlanProblem('At(Home) & Have(Cash) & Have(Car) ', 'At(Airport) & Have(Cash)',
            [goto_airport])
```

Listing 6.3 Kode Problem `goto_airport`

Aksi yang dapat dilakukan adalah `goto_airport` sehingga keluaran dari permasalahan di atas adalah aksi-aksi berikut: Drive (Home, AirportParking) dan Shuttle (AirportParking, Airport). Artinya agar *goal* berada di bandara dan mempunyai uang ('At(Airport) & Have(Cash)') terpenuhi maka solusinya adalah menyetir mobil sendiri. Menyetir mobil dapat dibagi menjadi dua aksi detail yaitu menyetir dari rumah ke halaman parkir bandara kemudian naik bus *shuttle* dari halaman parkir bandara ke *gate* bandara. Detail masing-masing aksi dapat dilihat pada Gambar 6.1:

```
precondition: [At(Home), Have(Car)]
action: Drive (Home, AirportParking)
effect : [At(AirportParking), NotAt(Home)]

precondition: [At(AirportParking)]
action: Shuttle (AirportParking, Airport)
effect : [At(Airport), NotAt(LongTermParking)]
```

Gambar 6.1 Keluaran Program Berupa Detail dari Aksi `goto_airport`

#### 6.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Perangkat Lunak: Bahasa dan Pemrograman Python
2. Perangkat Keras: Komputer dan periperalnya, modul Praktikum.

## 6.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	1. Jelaskan apa yang dimaksud dengan dari Hierarchy planning!	25
			2. Jelaskan tahapan penyelesaian masalah dengan menggunakan metode <i>Hirachical planning</i> !	25
			3. Contoh penerapan <i>Hirarchical planning</i> dalam kasus pendaftaran praktikum di prodi Informatika UAD!	50
Total				100

## 6.6. LANGKAH PRAKTIKUM

Sebelum melaksanakan materi praktikum ini, mahasiswa diminta mengambil *package planning* dengan cara import. Caranya dengan mengetik koding seperti listing 6.6.

```
from ai_pkg.planning import (HLA, Problem as PlanProblem)

solution = PlanProblem.hierarchical_search(problem, library)
for i in range(0, len(solution)):
    print('precondition: ', solution[i].precond)
    print('action: ' , solution[i].name,  solution[i].args)
    print('effect : ',  solution[i].effect)
```

Listing 6.4 Kode Program Hierarchical Planning untuk Kasus Pergi ke Bandara

Jawablah pertanyaan berikut:

**Aturan Penilaian (Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-08	CPMK-05	Jalankan program untuk kasus pergi ke bandara dengan cara: Tuliskan kode pada Listing 6.4	Hasil running program	30
2.	CPL-08	CPMK-05	Tambahkan hierarch yang ada pada Listing 6.1, aksi goto_airport yang ada pada Listing 6.2, dan Plan Problem yang ada pada Listing 6.3 sebelum memanggil fungsi hierarchical_search	Hasil running program	40
3.	CPL-08	CPMK-05	Jalankan program dan bandingkan keluarannya dengan gambar 6.1	Hasil running program	30
<b>Total Nilai</b>					<b>100</b>

## 6.7. POST TEST

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-08	CPMK-05	Jalankan program dan bandingkan keluarannya dengan Gambar 6.1	Hasil running program	30
2.	CPL-08	CPMK-05	Rubahlah program di atas supaya hasil yang keluar adalah pergi menggunakan taksi seperti keluaran pada Gambar 6.2 berikut: <pre>precondition: [At(Home)] action: Taxi (Home, Airport) effect : [At(Airport), NotAt(Home), NotHave(Cash)]</pre>	Hasil running program	70
Total Nilai					100

## 6.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-08	CPMK-05	30%		
3.	Post-Test	CPL-08	CPMK-05	50%		
Total Nilai						

**LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM**

<b>Nama :</b> <b>NIM :</b>	<b>Asisten:</b> <b>Paraf Asisten:</b>	<b>Tanggal:</b> <b>Nilai:</b>
-------------------------------	--	----------------------------------

--



## PRAKTIKUM 7: IMPLEMENTASI *MULTY AGENT*

**Pertemuan ke** : 7

**Total Alokasi Waktu** : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 30 menit
- Post-Test : 30 menit

**Total Bobot Penilaian** : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

**Pemenuhan CPL dan CPMK:**

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas
CPMK-01	Mampu merumuskan ruang masalah yang efisien untuk masalah yang diekspresikan dalam bentuk model graph dan Program sederhana
CPL-08	Mampu merancang dan mengimplementasikan algoritma/metode dalam mengidentifikasi dan memecahkan masalah yang melibatkan perangkat lunak dan pemikiran komputasi
CPMK-05	Mampu menganalisa masalah dengan menerapkan berbagai metode penalaran untuk menarik kesimpulan dalam mencapai solusi dan mengimplementasikannya dalam suatu aplikasi AI

### 7.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mampu menjelaskan konsep dasar *Multy Agent* dalam menyelesaikan masalah AI.
2. Mampu penyelesaian masalah yang dapat diselesaikan dengan model *multi agent* ke dalam bahasa pemrograman Phyton.

### 7.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-03	CPMK-01	Kemampuan mahasiswa dalam menjelaskan konsep dasar problem <i>Multiy Agent</i> dalam bidang AI dengan baik
CPL-08	CPMK-05	Mampu menganalisa masalah multy agnet dan mengimpelemntasikannya dengan pemrograman Phyton

### 7.3. TEORI PENDUKUNG

#### A. Multiagent Planning Problem

Pada Materi 6 telah dijelaskan mengenai planning untuk agen tunggal (*single agent*). Pada kenyataannya, beberapa masalah membutuhkan beberapa agen yang bekerja secara bersama-sama. Masing-masing agen menghadapi permasalahan multi agen (*multiagent planning problem*) dimana tiap agen berusaha untuk mencapai tujuannya dengan bantuan maupun halangan dari agen lain.

Ketika hanya ada satu agen yang melakukan *planning*, maka hanya ada satu *goal* yang dicapai agen tersebut. Namun ketika ada dua atau lebih agen yang melakukan *planning*, maka masing-masing agen berbagi *goal* yang sama. Contohnya dalam permasalahan permainan tenis *double* dimana kedua pemain dalam satu tim mempunyai *goal* yang sama yaitu bisa mengembalikan bola yang dioper oleh tim lawan. Dalam *multiagent* ada permasalahan yaitu apabila agen-agen yang ada mempunyai *goal* yang berbeda. Pada kasus tenis *double*, *goal* pada tim yang berlawanan saling berkebalikan.

#### B. Planning dengan Beberapa Aksi yang Simultan

Tujuan dari bagian ini adalah menemukan rencana aksi yang sesuai dengan permasalahan *multiagent*. Rencana disebut benar atau sesuai yaitu apabila aksi tersebut dijalankan oleh agen-agen maka *goal* dapat tercapai.

Permasalahan tenis *double* adalah salah satu contoh kasus sederhana tentang multiagent. Pada kasus ini dua pemain tenis A dan B bermain dalam satu tim dan dapat berada pada empat posisi: kiri belakang (*LeftBaseLine*), kanan belakang (*RightBaseLine*), kiri depan (*LeftNet*), dan kanan depan (*RightNet*). Bola dapat dikembalikan apabila pemain berada pada tempat yang sama dengan posisi bola.

Kondisi-kondisi pada permasalahan ini adalah:

- *Approaching*(Ball, loc) : Bola Ball dioper menuju posisi loc
- *Returned*(Ball) : salah satu pemain berhasil memukul dan mengembalikan bola dari posisi yang benar yang menyebabkan bola kembali ke daerah tim lawan.
- *At*(player, loc) : pemain player berada pada posisi loc
- *~At*(player, loc) : pemain player tidak berada pada posisi loc

Apabila pada kondisi awal posisi pemain A berada di kiri belakang dan pemain B berada pada posisi kanan depan, bola dipukul oleh lawan menuju kanan belakang, maka kondisi awal dapat dituliskan sebagai berikut:

(A,LeftBaseLine) & At(B,RightNet) & Approaching(Ball,RightBaseLine) & Team(A,B) & Team(B,A)

Beberapa aksi yang dapat dilakukan untuk mencapai *goal* antara lain:

- `Hit(player, Ball, loc)` : aksi mengembalikan Bola `Ball` jika ada pemain `player` yang berada di posisi `loc`. Secara lengkap aksi ini dapat dituliskan sebagai berikut:

```
Action('Hit(player, Ball, loc)',
  precond='Approaching(Ball, loc) & At(player, loc)',
  effect='Returned(Ball)')
```

- `Go(player, to, loc)` : pemain `player` bergerak dari posisi `loc` ke posisi `to`. Secara lengkap aksi ini dapat dituliskan sebagai berikut:

```
Action('Go(player, to, loc)',
  precond='At(player, loc)',
  effect='At(player, to) & ~At(player, loc)')
```

*Goal* dari permainan ini adalah:

```
Returned(Ball) & At(a, RightNet) & At(a, LeftNet)
```

Pada *goal* di atas terdapat variable baru yaitu `a`. Hal ini dapat terjadi pada kasus *multiagent* karena kita tidak memperdulikan pemain mana yang berakhir pada posisi `RightNet` atau `LeftNet`. Selama masih ada pemain pada salah satu posisi tersebut maka *goal* tetap dapat tercapai.

## 7.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Perangkat Lunak: Bahasa dan Pemrograman Python
2. Perangkat Keras: Komputer dan periperalnya, modul Praktikum.

## 7.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	1. Jelaskan apa yang dimaksud dengan Multy Agent!	25
			2. Jelaskan tahapan penyelesaian masalah dengan menggunakan Multy Agent!	25
			3. Buatlah contoh penerapan Multi Agent!	25
			4. Jelaskan Komponen aplikasi Multi Agent!	25
Total				100

## 7.6. LANGKAH PRAKTIKUM

1. Peserta duduk di kursi dan kompyuer yang telah disediakan (ini dipakai untuk seterusnya hingga praktikum terakhir.
2. Praktikan wajib membaca materi praktikum sebelum dilaksanakan

3. Praktikan mengikuti *pre-test* yang diberikan oleh Asistem praktikum
4. Memastikan bahwa perangkat lunak yang digunakan yaitu pemrograman Phyton telah tersedia.
5. Mengerjakan semua tugas praktikum sesuai dengan waktu yang disediakan.
6. Ketiklah kode program listing 7.1. sebagai contoh kasus pada permainan tenis *double* dengan menggunakan bahasa pemrograman Phyton:

```

from ai_pkg.planning import PlanningProblem, Action, goal_test
from ai_pkg.utils import expr

def double_tennis_problem():
    initial = 'At(A, LeftBaseLine) & At(B, RightNet) & Approaching(Ball,
RightBaseLine) & Team(A, B) & Team(B, A)'
    goal = 'Returned(Ball) & At(a, RightNet) & At(a, LeftNet)'
    action = [Action('Hit(player, Ball, loc)',
                    precond='Approaching(Ball, loc) & At(player, loc)',
                    effect='Returned(Ball)'),
              Action('Go(player, to, loc)',
                    precond='At(player, loc)',
                    effect='At(player, to)')]

    return PlanningProblem(init=initial,
                           goals=goal,
                           actions=action)

if __name__ == '__main__':
    p = double_tennis_problem()
    print(goal_test(p.goals, p.init))

    solution = [expr("Go(A, RightBaseLine, LeftBaseLine)"),
                expr("Hit(A, Ball, RightBaseLine)"),
                expr("Go(A, LeftNet, RightBaseLine)")]

    for action in solution:
        p.act(action)

    print(goal_test(p.goals, p.init))

```

Listing 7.1 Kode Program Permainan Tenis Double

7. Jawablah pertanyaan pada tabel berikut:

**Aturan Penilaian (Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-08	CPMK-05	Lakukan langkah praktikum 6 dengan menuliskan semua kode program pada listing 8.1.	Hasil kerja langkah 6	40
2.	CPL-08	CPMK-05	Jalankan program Python <i>multiagent</i> permainan tenis <i>double</i> dengan cara: <ol style="list-style-type: none"> <li>a. Jalankan program menggunakan console</li> <li>b. Catat hasilnya</li> <li>c. Rubah atau tambahkan beberapa aksi pada <i>solution</i> untuk membuktikan bahwa solusi yang telah disebutkan pada listing 8.1 sudah mencapai <i>goal</i></li> </ol>	Hasil running program	60

<b>Total Nilai</b>	<b>100</b>
--------------------	------------

### 7.7. POST TEST

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-08	CPMK-05	<p>Pada kasus permainan tenis <i>double</i> terdapat solusi lain yaitu:</p> <p>Pemain A: pergi dari kiri belakang (<i>LeftBaseLine</i>) ke kiri depan (<i>LeftNet</i>) dan tidak melakukan apa-apa (<i>NoOp</i>)</p> <p>Pemain B: pergi dari kanan depan (<i>RightNet</i>) ke kanan belakang (<i>RightBaseLine</i>) dan mengembalikan bola (<i>Hit</i>)</p> <ol style="list-style-type: none"> <li>Tuliskan solusi di atas ke dalam program dengan cara:</li> <li>Tambahkan aksi <i>NoOp(player)</i> ke dalam <i>action</i> pada <i>PlanningProblem</i> dengan <i>precond</i> dan <i>effect</i> kosong (tidak ada aksi apapun sehingga kondisi sebelum dan sesudah aksi tidak berubah)</li> <li>Ganti solusi yang ada pada <i>solution</i> dengan solusi yang telah disebutkan di atas.</li> </ol>	Hasil running program	100
<b>Total Nilai</b>					<b>100</b>

### 7.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-08	CPMK-05	30%		
3.	Post-Test	CPL-08	CPMK-05	50%		
<b>Total Nilai</b>						

**LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM**

<b>Nama :</b> <b>NIM :</b>	<b>Asisten:</b> <b>Paraf Asisten:</b>	<b>Tanggal:</b> <b>Nilai:</b>
-------------------------------	--	----------------------------------

--

## PRAKTIKUM 8: IMPLEMENTASI JARINGAN SEMANTIK

**Pertemuan ke** : 8

**Total Alokasi Waktu** : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 30 menit
- Post-Test : 30 menit

**Total Bobot Penilaian** : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

**Pemenuhan CPL dan CPMK:**

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas
CPMK-01	Mampu merumuskan ruang masalah yang efisien untuk masalah yang diekspresikan dalam bentuk model graph dan Program sederhana
CPL-08	Mampu merancang dan mengimplementasikan algoritma/metode dalam mengidentifikasi dan memecahkan masalah yang melibatkan perangkat lunak dan pemikiran komputasi
CPMK-05	Mampu menganalisa masalah dengan menerapkan berbagai metode penalaran untuk menarik kesimpulan dalam mencapai solusi dan mengimplementasikannya dalam suatu aplikasi AI

### 8.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mampu menjelaskan konsep dasar Jaringan Semantik untuk menyajikan pengetahuan dalam menyelesaikan masalah AI.
2. Mampu membuat jaringan semantic dan menganalisanya untuk penyelesaian masalah dalam AI.

### 8.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

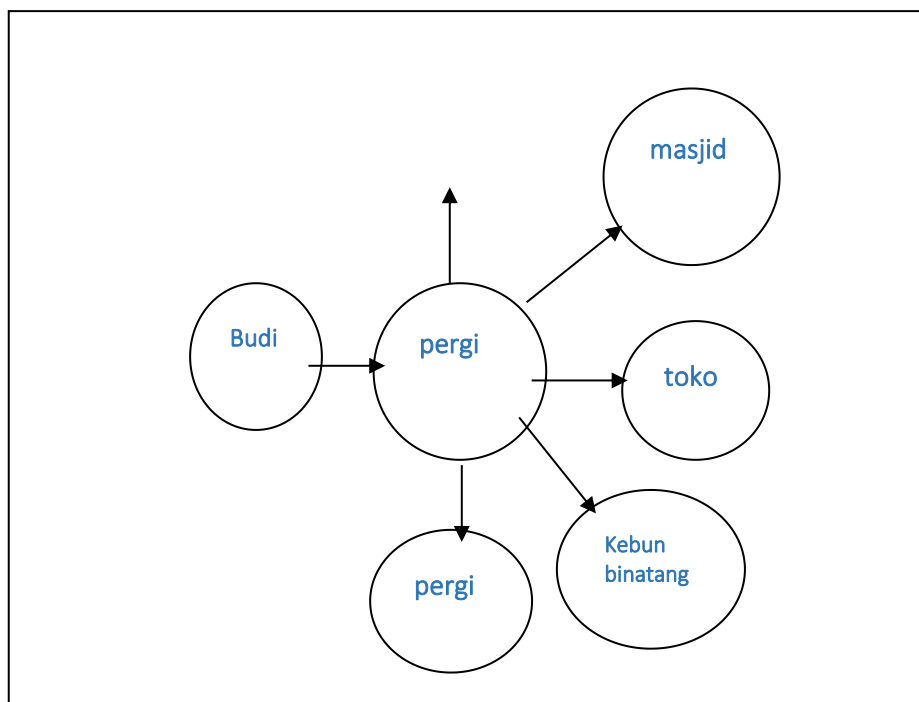
CPL-03	CPMK-01	Kemampuan mahasiswa dalam menjelaskan konsep dasar problem Jaringan Semantik dalam bidang AI dengan baik
CPL-08	CPMK-05	Mampu membuat dan menganalisa masalah Jaringan semantiks dalam

### 8.3. TEORI PENDUKUNG

Jaringan semantik merupakan gambaran pengetahuan grafis yang menunjukkan hubungan antar obyek. Jaringan semantik terdiri-dari lingkaran-lingkaran yang menunjukkan obyek dan informasi tentang obyek-obyek tersebut. Obyek ini bisa berupa benda atau peristiwa. Antara 2 obyek dihubungkan oleh arcd yang menunjukkan hubungan antar obyek. Gambar 8.1 merupakan contoh representasi pengetahuan dengan menggunakan jaringan semantik.

Salah satu kelebihan dari jaringan simantik adalah 'bisa mewariskan' sebagai contoh, pada Gambar 8.1 ada garis yang menghubungkan antara budi dengan laki-laki dan laki-laki ke makhluk hidup. Sehingga apabila ada pertanyaan: Apakah budi makhluk hidup? Maka kita bisa merunut garis dari maklik hidup kemudian ke laki-laki, dan akhirnya ke budi. Sehingga terbukti bahwa budi adalah makhluk hidup.

Sistem jaringan semantik ini selalu tergantung pada jenis masalah yang akan dipecahkan. Jika masalah itu bersifat umum, maka hanya memerlukan sedikit rincian. Jika ternyata masalah itu banyak melibatkan hal-hal lain, maka di didalam jarinagn awalnya diperlukan penjelasan yang lebih rinci lagi Gambar 8.2.. Menunjukkan rincian dari node awal budi, apabila budi hendak pergi ke berbagai tempat. Node budi dihubungkan dengan node baru,yaitu pergi.



Gambar 8.1. Model Jaringan Semantik



*Semantic Network* adalah representasi yang mengekspresikan solusi permasalahan dengan menggunakan network (graph berarah). Di dalamnya digunakan node (simpul) untuk merepresentasikan suatu kondisi, dan arc (link) untuk merepresentasikan relasi antar simpul. Semantic Network ini adalah representasi yang bersifat:

- Lexically*, di dalamnya terdapat node(simpul), link dan batasan-batasan khusus dari permasalahan.
- Structurally*, masing-masing link akan terkoneksi dari simpul yang paling depan (head node) sampai simpul yang paling belakang (tail node).
- Semantically*, semua simpul dan link merepresentasikan permasalahan tetap berada dalam batasannya.

Sebagai contoh adalah permasalahan petani, serigala, angsa dan padi. Seorang petani URAIAN MATERI ingin memindah dirinya sendiri, seekor serigala, seekor angsa gemuk, dan seikat padi yang berisi menyeberangi sungai. Sayangnya, perahunya sangat terbatas; dia hanya dapat membawa satu objek dalam satu penyeberangan. Dan lagi, dia tidak bisa meninggalkan serigala dan angsa dalam satu tempat, karena serigala akan memangsa angsa. Demikian pula dia tidak bisa meninggalkan angsa dengan padi dalam satu tempat. Mendeskripsikan permasalahan di atas dengan menggunakan bahasa natural bukanlah cara yang tepat. Dalam hal ini, kita dapat menggunakan Semantic Network untuk merepresentasikannya.

#### 8.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

- Perangkat Lunak: Bahasa dan Pemrograman Phyton
- Perangkat Keras: Komputer dan periperalnya, modul Praktikum.

#### 8.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1	CPL-03	CPMK-01	1. Jelaskan cara kerja Jaringan semantic!	25
			2. Jaringan semantic merupakan representasi pengetahuan berbentuk grafis. Jelaskan contoh penerapan jaringan semantic pada aplikasi-aplikasi bidang AI!	25
			3. Bila diberikan rule sebagai berikut: IF A and B Then C IF D and D Then E IF E then F IF C and G and H Then I	50

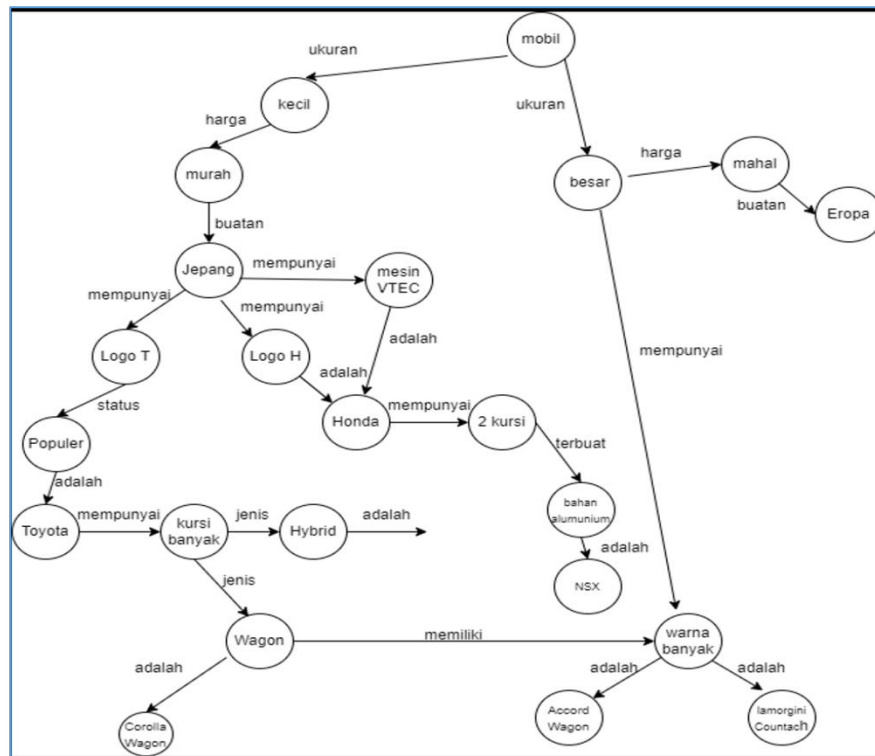
			IF J and K and L Then F IF J and M and N and O then P Buatlan semantic netnya!	
				<b>Total 100</b>

### 8.6. LANGKAH PRAKTIKUM

1. Peserta duduk di kursi dan kompyuer yang telah disediakan (ini dipakai untuk seterusnya hingga praktikum terakhir.
2. Praktikan wajib membaca materi praktikum sebelum dilaksanakan
3. Praktikan mengikuti *pre-test* yang diberikan oleh Asistem praktikum
4. Memastikan bahwa perangkat lunak yang digunakan yaitu Visio drawing atau Drawing IO telah tersedia.
5. Mengerjakan semua tugas praktikum sesuai dengan waktu yang disediakan.
6. Berdasarkan Rule berikut ini buatlah rancangan Jaringan Semantik nya!

R1	IF harganya murah THEN buatan Jepang
R2	IF ukurannya kecil THEN buatan Jepang
R3	IF ukurannya besar AND harganya mahal THEN buatan Eropa
R4	IF buatan Jepang AND punya logo bertanda "T" THEN mobil Toyota
R5	IF buatan Jepang AND mobil populer THEN mobil Toyota
R6	IF buatan Jepang AND punya logo bertanda "H" THEN mobil Honda
R7	IF buatan Jepang AND mempunyai mesin VTEC THEN mobil Honda
R8	IF mobil Toyota AND mempunyai kursi banyak AND jenisnya wagon THEN merk Carolla Wagon
R9	IF mobil Toyota AND mempunyai kursi banyak AND jenisnya hybrid THEN merk Prius
R10	IF mobil Honda AND mempunyai kursi banyak AND jenisnya wagon AND punya banyak pilihan warna THEN merk Accord Wagon
R11	IF mobil Honda AND hanya mempunyai 2 kursi AND bahan aluminium THEN merk NSX
R12	IF buatan Eropa AND mesinnya besar AND jenis sport AND punya banyak pilihan warna THEN merk Lamorghini Countach

Dari Rule tersebut, maka dihasilkan semantik Net pada Gambar 8.2.



Gambar 8.2. Hasil Jaringan semantic

7. Berdasarkan gambar 8.3 buatlah Aturannya!

#### Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-08	CPMK-05	Berdasarkan langkah 6 tersebut, jika diketahui fakta awal bahwa mobil harganya tidak mahal, mempunyai mesin VTEC, mempunyai banyak kursi dan pilihan warna, dan jenisnya wagon. Dengan menggunakan penalaran model <i>backward chaining</i> , buktikan bahwa merk mobil adalah Accord Wagon!	Hasil kerja langkah 6	20
2.	CPL-08	CPMK-05	Berdasarkan gambar semantic net pada Gambar 8.3. buatlah tabel Aturannya	Hasil langkah 7	40
3	CPL-08	CPMK-05	Buatlah Jaringan semantic dari Rule berikut: R1 : IF (Shape = long) AND (color = green or yellow) THEN fruit = banana R2 : IF (Shape = round or oblong) AND (Diameter > 4 inches) THEN Fruitclass = vine R3 : IF (Shape = round or oblong) and (Diameter < 4 inches) THEN Fruitclass = <b>tree</b> R4 : IF (Seedcount = 1 ) THEN Seedclass = stonefruit R5 : IF (Seedcount > 1 ) THEN Seedclass = <b>multiple</b> R6 : IF (Fruitclass = vine) AND (Color = green) THEN Fruit = watermelon R7 : IF (Fruitclass = vine) AND (Surface = rough)	Hasil desain jaringan semantik	40

			AND (Color = tan) THEN Fruit = honeydew R8 : IF (Fruitclass = vine) AND (Surface = smooth) AND (Color = yellow) THEN Fruit = Cantaloupe R9 : IF (Fruitclass = tree) AND (Color = orange) AND (Seedclass = stonefruit) THEN Fruit = apricot R10 : IF ( <b>Fruitclass = tree</b> ) AND ( <b>Color = orange</b> ) AND ( <b>Seedclass = multiple</b> ) THEN Fruit = orange R11 : IF (Fruitclass = tree) AND (Color = red OR yellow OR green) AND (Seedclass = multiple) THEN Fruit = apple		
<b>Total Nilai</b>					<b>100</b>

### 8.7. POST TEST

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-08	CPMK-05	Berdasarkan soal no 3 dari diagram Seamntik Net yang dibuat, tentukanlah:	Hasil analisa	25
			a. buah apakah ini jika diketahui fakta sebagai berikut : <ul style="list-style-type: none"> <li>• Diameter = 5 inch</li> <li>• Shape = round</li> <li>• SeedCount = 3</li> <li>• Color = yellow</li> <li>• Surface = smooth</li> </ul> b. buah apakah buah <b>Orange</b> atau bukan jika diketahui fakta <ul style="list-style-type: none"> <li>• Diameter = 3 inch</li> <li>• Shape = oblong</li> <li>• SeedCount = 2</li> <li>• Color = orange</li> </ul>	Hasil analisa	25
2	CPL-08	CPMK-05	Buatlah contoh kasus Anda sendiri yang terdiri dari aturan dan diagram semantic netnya.	Hasil Diagram semantic Net	50
<b>Total Nilai</b>					<b>100</b>

### 8.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-08	CPMK-05	30%		
3.	Post-Test	CPL-08	CPMK-05	50%		
<b>Total Nilai</b>						

**LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM**

<b>Nama :</b> <b>NIM :</b>	<b>Asisten:</b> <b>Paraf Asisten:</b>	<b>Tanggal:</b> <b>Nilai:</b>
-------------------------------	--	----------------------------------

--

## PRAKTIKUM 9: IMPLEMENTASI FAKTOR KETIDAK PASTIAN

**Pertemuan ke** : 9

**Total Alokasi Waktu** : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 30 menit
- Post-Test : 30 menit

**Total Bobot Penilaian** : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

**Pemenuhan CPL dan CPMK:**

CPL-05	Mampu mengkaji / menganalisis implikasi pengembangan atau implementasi ilmu pengetahuan teknologi, menyusun deskripsi saintifik hasil kajian untuk pemecahan masalah dengan mempertimbangkan multidisiplin ilmu
CPMK-03	Mampu menerapkan konsep pelacakan secara heuristik maupun pelacakan buta yang sesuai untuk mencari solusi masalah kedalam suatu bahasa pemrograman
CPL-08	Mampu merancang dan mengimplementasikan algoritma/metode dalam mengidentifikasi dan memecahkan masalah yang melibatkan perangkat lunak dan pemikiran komputasi
CPMK-05	Mampu menganalisa masalah dengan menerapkan berbagai metode penalaran untuk menarik kesimpulan dalam mencapai solusi dan mengimplementasikannya dalam suatu aplikasi AI

### 9.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mampu menjelaskan konsep dasar Statistik khususnya Teorema Bayes dalam mengatasi ketidak pastiaan saat proses pencarian solusi masalah AI.
2. Mampu melakukan analisa untuk mengatasi ketidak pastian untuk penyelesaian masalah dalam AI dengan menggunakan aplikasi WEKA.

### 9.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-05	CPMK-03	Kemampuan mahasiswa dalam menjelaskan konsep dasar statistic yang dipakai untuk mengatasi ketidak pastian dalam pencarian solusi
CPL-08	CPMK-05	Mampu menggunakan dan menganalisa aplikasi WEKA untuk pencarian

### 9.3. TEORI PENDUKUNG

Pada bagian terdahulu kita telah mempelajari teknik penalaran dengan model yang sangat lengkap dan konsisten. Namun, pada kenyataanya, banyak masalah di dunia ini yang tidak dapat dimodelkan secara lengkap dan konsisten. Misalnya (pembunuhan ABC):

Abby, Baby dan Caby adalah 3 orang tersangka terhadap suatu kejadian pencurian. Abby memiliki alibi, yaitu saat kejadian pencurian ia sedang bersama temannya pada sebuah Warung kopi yang dapat dipertanggungjawabkan di daerah XYZ. Baby juga mempunyai alibi, yaitu dengan pengakuan pamannya bahwa Baby menginap di rumah Pamannyanya. Caby juga mempunyai alibi yaitu; bahwa pada saat terjadi pencurian ia sedang pergi menonton ke Bioskop, namun tak seorangpun yang tahu. Dengan demikian dapat disimpulkan bahwa:

- (a) Abby bukan pencurinya;
- (b) Baby juga bukan pencurinya;
- (c) Abby atau Baby atau Caby adalah pencuri;

Dari sini terlihat bahwa Caby memiliki alibi yang paling lemah. Namun, jika pada suatu saat, ada saksi yang melihat Caby di Bioskop bahwa Caby sedang menonton pada saat itu, maka timbul suatu kepercayaan baru:

- (d) Caby bukan pencurinya.

Dari kejadian tersebut bahwa ada ketidak konsistenan fakta dari (a) sampai (d).

Suatu penalaran dimana adanya penambahan fakta baru mengakibatkan konsistenan disebut dengan "*penalaran Non Monotonis*".

Ciri-ciri dari penalaran non monotonis adalah:

1. Mengandung ketidakpastian;
2. Adanya perubahan padapengetahuan.

Sedangkan penalaran Monotonis memiliki ciri-ciri:

1. konsisten;
2. Pengetahuannya lengkap.

Untuk mengatasi ketidakpastian pada penalaran non monotonis, maka digunakan penalaran statistik.

### PROBABILITAS

Dalam pengembangan aplikasi di bidang AI, teori probabilistic paling banyak digunakan dalam proses pelacakan untuk pengambilan keputusan. Terutama untuk aplikasi yang bersifat diagnose, seperti: Sistem pakar. Definisi probabilitas menunjukkan kemungkinan sesuatu akan terjadi atau tidak.

$$p(x) = \frac{\text{jumlah kejadian berhasil}}{\text{jumlah semua kejadian}}$$

Misal dari 10 orang sarjana, 3 orang menguasai cisco, sehingga peluang untuk memilih sarjana yang menguasai cisco adalah :  $p(\text{cisco}) = 3/10 = 0.3$

### Metode Kepastian dengan Teorema Bayes

Salah satu konsep probabilistic yang paling sering dipakai adalah teorema Bayes. Teorema Bayes dipakai salah satunya untuk mengatasi ketidakpastian. Persamaan teorema bayes dapat dilihat sebagai berikut:

$$p(H_i | E) = \frac{p(E | H_i) * p(H_i)}{\sum_{k=1}^n p(E | H_k) * p(H_k)}$$

dengan:

$p(H_i | E)$  = probabilitas hipotesis  $H_i$  benar jika diberikan evidence (fakta)  $E$

$p(E | H_i)$  = probabilitas munculnya evidence (fakta)  $E$  jika diketahui hipotesis  $H_i$  benar

$p(H_i)$  = probabilitas hipotesis  $H_i$  (menurut hasil sebelumnya) tanpa memandang evidence (fakta) apapun

$n$  = jumlah hipotesis yang mungkin

Contoh:

Asih mengalami gejala ada bintik-bintik di wajahnya. Dokter menduga bahwa Asih terkena cacar dengan :

- probabilitas munculnya bintik-bintik di wajah, jika Asih terkena cacar  $\rightarrow p(\text{bintik2} | \text{cacar}) = 0,8$
- probabilitas Asih terkena cacar tanpa memandang gejala apapun  $\rightarrow p(\text{cacar}) = 0,4$
- probabilitas munculnya bintik-bintik di wajah, jika Asih terkena alergi  $\rightarrow p(\text{bintik2} | \text{alergi}) = 0,3$
- probabilitas Asih terkena alergi tanpa memandang gejala apapun  $\rightarrow p(\text{alergi}) = 0,7$
- probabilitas munculnya bintik-bintik di wajah, jika Asih jerawat  $\rightarrow p(\text{bintik2} | \text{jerawat}) = 0,9$
- probabilitas Asih jerawat tanpa memandang gejala apapun  $\rightarrow p(\text{jerawat}) = 0,5$

Maka :

$$p(H_i | E) = \frac{p(E | H_i) * p(H_i)}{\sum_{k=1}^n p(E | H_k) * p(H_k)}$$

- probabilitas Asih terkena cacar karena ada bintik-bintik di wajahnya :



$$p(\text{cacar} | \text{bintik}) = \frac{p(\text{bintik} | \text{cacar}) * p(\text{cacar})}{p(\text{bintik} | \text{cacar}) * p(\text{cacar}) + p(\text{bintik} | \text{alergi}) * p(\text{alergi}) + p(\text{bintik} | \text{jerawat}) * p(\text{jerawat})}$$

$$p(\text{cacar} | \text{bintik}) = \frac{(0.8) * (0.4)}{(0.8) * (0.4) + (0.3) * (0.7) + (0.9) * (0.5)} = \frac{0.32}{0.98} = 0.327$$

- probabilitas Asih terkena alergi karena ada bintik-bintik di wajahnya :

$$p(\text{alergi} | \text{bintik}) = \frac{p(\text{bintik} | \text{alergi}) * p(\text{alergi})}{p(\text{bintik} | \text{cacar}) * p(\text{cacar}) + p(\text{bintik} | \text{alergi}) * p(\text{alergi}) + p(\text{bintik} | \text{jerawat}) * p(\text{jerawat})}$$

$$p(\text{alergi} | \text{bintik}) = \frac{(0.3) * (0.7)}{(0.8) * (0.4) + (0.3) * (0.7) + (0.9) * (0.5)} = \frac{0.21}{0.98} = 0.214$$

- probabilitas Asih jerawat karena ada bintik-bintik di wajahnya :

$$p(\text{jerawat} | \text{bintik}) = \frac{p(\text{bintik} | \text{jerawat}) * p(\text{jerawat})}{p(\text{bintik} | \text{cacar}) * p(\text{cacar}) + p(\text{bintik} | \text{alergi}) * p(\text{alergi}) + p(\text{bintik} | \text{jerawat}) * p(\text{jerawat})}$$

$$p(\text{jerawat} | \text{bintik}) = \frac{(0.9) * (0.5)}{(0.8) * (0.4) + (0.3) * (0.7) + (0.9) * (0.5)} = \frac{0.45}{0.98} = 0.459$$

Jika setelah dilakukan pengujian terhadap hipotesis muncul satu atau lebih evidence (fakta) atau observasi baru maka :

$$p(H | E, e) = p(H | E) * \frac{p(e | E, H)}{p(e | E)}$$

dengan :

e = *evidence* lama

E = *evidence* atau observasi baru

$p(H | E, e)$  = probabilitas hipotesis H benar jika muncul *evidence* baru E dari *evidence* lama

$p(H | E)$  = probabilitas hipotesis H benar jika diberikan *evidence* E.

$p(e | E, H)$  = kaitan antara e dan E jika hipotesis benar

$P(e | E)$  = kaitan antara e dan E tanpa memandang hipotesis apapun.

### Algoritma Naive Bayes pada WEKA

Proses klasifikasi (*classify*) menggunakan metode *Naïve Bayesian Classifier* (dalam software Weka disebut sebagai metode *NaïveBayes*). WEKA dapat digunakan sebagai perangkat lunak pembelajaran mesin untuk menghasilkan dan model uji *Naive Bayes* secara otomatis. Kelas untuk *Naive Bayes classifier* menggunakan kelas estimator. Nilai estimator presisi *Numeric* dipilih berdasarkan analisis data training. *Classifier* ini menggunakan presisi *default* 0.1 untuk atribut numerik ketika *buildClassifier* disebut dengan kasus *training* nol. *Naive Bayes* tidak memilih fitur penting. Hasil dari training *Naive Bayes classifier* adalah *mean* dan *varians* untuk setiap fitur. Klasifikasi sampel baru

ke 'Ya' atau 'Tidak' didasarkan pada apakah nilai-nilai fitur dari sampel *test* terbaik dengan rata-rata dan varians dari fitur yang menjadi training baik untuk 'Ya' atau 'Tidak'.

#### 9.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Perangkat Lunak: Aplikasi WEKA
2. Perangkat Keras: Komputer dan periperalnya, modul Praktikum.

#### 9.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1	CPL-05	CPMK-03	1. Jelaskan bagaimana langkah-langkah penggunaan aplikasi WEKA untuk menganalisa kepastian	25
2	CPL-05	CPMK-03	2. Jelaskan 3 metode yang dipakai untuk mengatasi ketidakpastian dalam pencarian solusi	25
3	CPL-05	CPMK-03	3. Jelaskan keunggulan aplikasi WEKA	25
4	CPL-08	CPMK-05	4. Dalam Praktikum ini sebutkan aplikasi sejenis yang dapat digunakan untuk mengatasi ketidakpastian	25
<b>Total</b>				<b>100</b>

#### 9.6. LANGKAH PRAKTIKUM

1. Peserta duduk di kursi dan komputer yang telah disediakan (ini dipakai untuk seterusnya hingga praktikum terakhir).
2. Praktikan wajib membaca materi praktikum sebelum dilaksanakan
3. Praktikan mengikuti *pre-test* yang diberikan oleh Asistem praktikum
4. Memastikan bahwa perangkat lunak yang digunakan yaitu aplikasi WEKA dan Notepad telah tersedia.
5. Mengerjakan semua tugas praktikum sesuai dengan waktu yang disediakan.
6. Sebelum melaksanakan tugas praktikum, mahasiswa terlebih dahulu harus menginstall aplikasi WEKA. Aplikasi ini sebagai salah satu software yang dipakai untuk melakukan analisis terutama terori statistic.
7. Jika sudah tersedia, maka dapat mengerjakan **Tugas 1 praktikum** sebagai berikut:

##### Tugas 1

Diketahui sebuah data dari Dinas Kesehatan memiliki data jumlah pasien menginap pada Rumah sakit di wilayah Jogjakarta pada VIP, kelas I, Kelas II, Kelas III dan VVIP. Data tersebut ditunjukkan dalam Tabel 9.1.

*Tabel 9.1 Data jumlah pasien rawat inap RS wilayah Jogjakarta*

Halte	VVIP	VIP	KLS I	KLS II	KLS III	Kelas
Rumah sakit A	40	45	44	33	20	banyak
Rumah sakit B	21	33	43	50	42	sedang
Rumah sakit C	46	30	16	36	22	banyak
Rumah sakit D	53	26	26	39	21	banyak
Rumah sakit E	33	37	39	32	18	banyak
Rumah sakit F	21	27	35	42	39	sedikit
Rumah sakit G	37	47	38	32	18	banyak
Rumah sakit H	39	38	33	38	25	sedang

Dengan menggunakan weka hitunglah probabilitas menggunakan metode Kepastian Teorema Bayes.

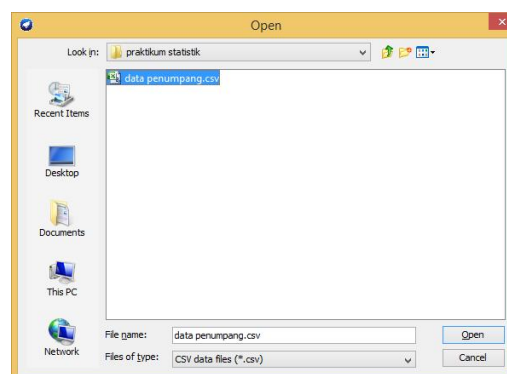
### Langkah Praktikum dengan menggunakan WEKA

1. Ketiklah data pada Tabel 9.1 pada excel dan simpanlah dalam bentuk .CSV
2. Bukalah program WEKA dimana tampilannya seperti pada Gambar 9.1.



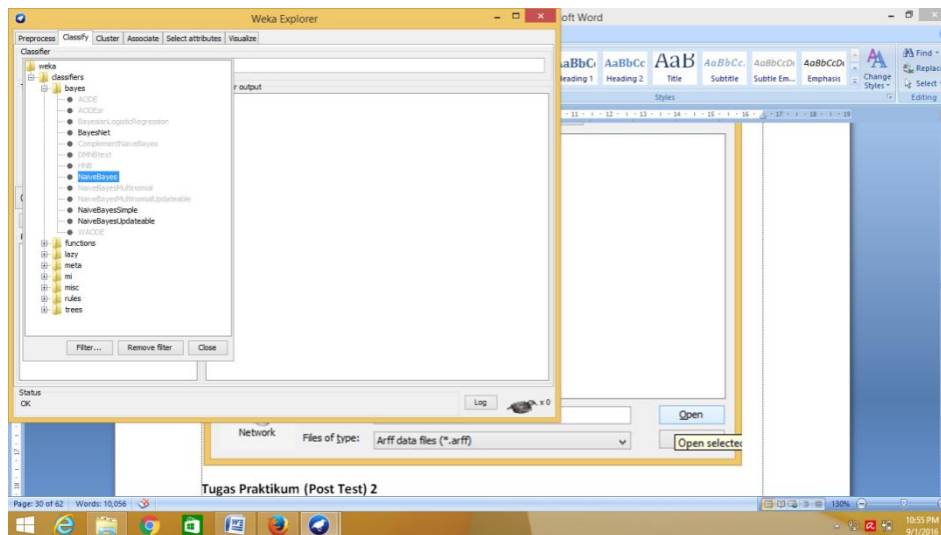
Gambar 9.1 Tampilan Awal WEKA

3. Bukalah file dan pilih dimana lokasi file diletakkan. Yang perlu menjadi catatan adalah file dari excel disimpan dalam CSV agar dapat dibaca oleh program WEKA.



Gambar 9.2 Open File .CSV

4. Pada Tabulasi atas pilihah classification -> Pilih bayes -> star



Gambar 9.3 Pmlihan metode BAYES

- Lakukan analisis dari hasil program WEKA tersebut, hasil proses penghitungan BAYES menggunakan WEKA.

#### Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-05	CPMK-03	Kerjakanlah langkah 7 Tugas 1)	Hasil kerja langkah 7	30
2.	CPL-08	CPMK-05	Tuliskan bagaimana output yang dihasilkan..?	Hasil langkah 7	20
3	CPL-08	CPMK-05	Buatlah hasil analisa berdasarkan data Tabel 9.1	Hasil analisa	50
<b>Total Nilai</b>					<b>100</b>

## 9.7. POST TEST

Tugas 2: diberikan data seperti Tabel 9.2. Lalu jawablah pertanyaannya:

Tabel 9.2 Data Lama Penulisan skripsi Mahasiswa

Mahasiswa	Masa penulisan Bab 1 dan pengambilan data (bln)	Masa penulisan bab 2 (bln)	Masa penulisan bab 3(bln)	Masa penulisan bab 4&5(bln)	Lama Masa penyelesaian skripsi
1	3.5	3.60	3.0	1	Cepat
2	4	3.25	2.5	2.5	Cepat
3	4	3.10	3.5	2.7	Lama
4	5	2.98	4.2	1.5	Sedang
5	4.5	3.00	2.1	2.0	Sedang
6	6	3.20	3.2	3.0	Lama
7	4	2.75	2.1	1	Lama
8	4.5	2.90	3.0	0.7	Lama
9	6	3.35	3.1	1.2	Sedang

10	4	3.10	4.3	2	Cepat
----	---	------	-----	---	-------

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-05	CPMK-03	Kerjakanlah Tugas 2 seperti langkah 7 dengan mengubah data ke dalam bentuk file excel (.csv)	Hasil langkah 7	30
2	CPL-08	CPMK-05	Lakukan pengamatan terhadap hasil outputnya, buatlah analisismu. Apa yang dapat Anda simpulkan..?	Hasil Analisa	70
<b>Total Nilai</b>					<b>100</b>

### 9.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-05	CPMK-03	20%		
2.	Praktik	CPL-08	CPMK-05	30%		
3.	Post-Test	CPL-08	CPMK-05	50%		
<b>Total Nilai</b>						

**LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM**

<b>Nama :</b> <b>NIM :</b>	<b>Asisten:</b> <b>Paraf Asisten:</b>	<b>Tanggal:</b> <b>Nilai:</b>
-------------------------------	--	----------------------------------

--

## PRAKTIKUM 10: IMPLEMENTASI *CUSTOM PACKAGE* PYTHON

**Pertemuan ke** : 10

**Total Alokasi Waktu** : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 30 menit
- Post-Test : 30 menit

**Total Bobot Penilaian** : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

**Pemenuhan CPL dan CPMK:**

CPL-07	Mampu memilih, membuat dan menerapkan teknik, sumber daya, penggunaan perangkat teknik modern dan implementasi teknologi informasi untuk memecahkan masalah
CPMK-04	Mampu menerapkan berbagai model representasikan pengetahuan untuk memecahkan masalah ke dalam suatu aplikasi AI
CPL-08	Mampu merancang dan mengimplementasikan algoritma/metode dalam mengidentifikasi dan memecahkan masalah yang melibatkan perangkat lunak dan pemikiran komputasi
CPMK-05	Mampu menganalisa masalah dengan menerapkan berbagai metode penalaran untuk menarik kesimpulan dalam mencapai solusi dan mengimplementasikannya dalam suatu aplikasi AI

### 10.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mampu menjelaskan teknologi informasi yang dipakai untuk proses pencarian solusi masalah AI.
2. Mampu membuat package-package dari tugas-tugas praktikum yang telah dikerjakan ke dalam bahasa pemrograman Python.

## 10.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-07	CPMK-04	Kemampuan mahasiswa dalam memilih teknologi informasi yang sesuai untuk package –package dalam pemrograman Phyton berdasarkan tugas praktikum yang dikerjakan
CPL-08	CPMK-05	Kemampuan mahasiswa dalam membuat package –package dalam pemrograman Phyton berdasarkan tugas praktikum yang dikerjakan

## 10.3. TEORI PENDUKUNG

### Struktur Package dalam Python

Dalam Phyton sudah ada package-package yang disediakan untuk berbagai penerapan bidang AI. Dengan demikian akan mempermudah pengguna dalam membuat program menggunakan package yang ada. Dapat dilakukan dengan melakukan customisasi.

Untuk membuat *custom package* kita perlu membuat *project* Python dengan susunan awal seperti berikut:

```
/ai_pkg
  /ai_pkg
    __init__.py
```

`ai_pkg` adalah nama dari *package* yang akan dibuat. Oleh karena itu pendeklarasian nama package perlu dilakukan di file `__init__.py`. Tambahkan kode berikut di file `__init__.py`.

```
name = "ai_pkg"
```

### Membuat File-file dalam *Package*

Selanjutnya kita membuat file-file lain yang akan disatukan ke dalam *project* ini dan didistribusikan sebagai *custom package*. Tambahkan file `setup.py`, `LICENSE`, dan `README.md` pada direktori *project*.

```
/ai_pkg
  /ai_pkg
    __init__.py
    setup.py
    LICENSE
    README.md
```



### Membuat setup.py

setup.py adalah *script* untuk membentuk package menggunakan [setuptools](#). Di dalam file ini didefinisikan semua hal tentang *package*, misalnya nama, versi, deskripsi, penulis, dan informasi lainnya.

```
import setuptools

with open("README.md", "r") as fh:
    long_description = fh.read()

setuptools.setup(
    name="ai_pkg",
    version="0.0.1",
    author="Example Author",
    author_email="author@example.com",
    description="A small example package",
    long_description=long_description,
    long_description_content_type="text/markdown",
    url="https://github.com/pypa/sampleproject",
    packages=setuptools.find_packages(),
    classifiers=[
        "Programming Language :: Python :: 3",
        "License :: OSI Approved :: MIT License",
        "Operating System :: OS Independent",
    ],
)
```

### Membuat README.md

File ini memuat deskripsi *package*. Bisa diisi dengan informasi apapun dari kegunaan *package*, contoh penggunaan, dll.

```
# Example Package

This is a simple example package. You can use

[Github-flavored Markdown] (https://guides.github.com/features/mastering-markdown/)
to write your content.
```

## Membuat LICENSE

Pembuatan file ini penting apabila *package* yang kita buat diupload dan didistribusikan

```
Copyright (c) 2018 The Python Packaging Authority
```

```
Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
```

```
The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.
```

```
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

## Pembuatan Distribusi *Package* Menggunakan Python Wheel

Langkah selanjutnya adalah untuk membuat distribusi *package* berupa *archive* yang dapat diinstal menggunakan `pip`. Untuk membuat distribusi *package*, diperlukan instalasi `setuptools` dan `wheel`.

```
pip install --user --upgrade setuptools wheel
```

Setelah `setuptools` dan `wheel` terinstal, jalankan perintah berikut pada direktori yang mengandung file `setup.py`.

```
python setup.py sdist bdist_wheel
```

Perintah tersebut akan menghasilkan dua file yang ada apada folder `dist`, `.tar.gz` adalah arsip kode dan `.whl` adalah file distribusi. Perintah `pip` akan menginstal *package* menggunakan file distribusi (`.whl`) tetapi file arsip (`.tar.gz`) dapat digunakan apabila file distribusi tidak tersedia.

```
dist/

ai_pkg-0.0.1-py3-none-any.whl

ai_pkg-0.0.1.tar.gz
```

## 10.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Perangkat Lunak : aplikasi Phyton menu custom package
2. Perangkat Keras: Komputer dan Periperalnya, dan Buku petunjuk praktikum.

## 10.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1	CPL-07	CPMK-04	1. Jelaskan apa yang dimaksud dengan package-package dalam pemrograman Phyton.	25
2	CPL-07	CPMK-04	2. Jelaskan saat pembuatan package dalam Phyton hal-hal apa saja yang harus dipersiapkan!	25
3	CPL-08	CPMK-05	3. Jelaskan tujuan adanya package dalam pemrograman Phyton bagi pengguna	25
4	CPL-08	CPMK-05	4. Jelaskan struktur package dalam phyton	25
Total				100

## 10.6. LANGKAH PRAKTIKUM

1. Peserta duduk di kursi dan kompyuer yang telah disediakan (ini dipakai untuk seterusnya hingga praktikum terakhir.
2. Praktikan wajib membaca materi praktikum sebelum dilaksanakan
3. Praktikan mengikuti *pre-test* yang diberikan oleh Asistem praktikum
4. Memastikan bahwa perangkat lunak yang digunakan yaitu pemrograman Phyton telah tersedia.
5. Mengerjakan semua tugas praktikum sesuai dengan waktu yang disediakan.
6. Jawablah pertanyaan berikut:

**Aturan Penilaian (Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-07	CPMK-04	<p>Untuk membuat <i>custom package</i> <code>ai_pkg</code> versi 0.0.2 dilakukan dengan dengan cara:</p> <ol style="list-style-type: none"> <li>1. Unduh semua file modul yang ada <i>custom package</i> <code>ai_pkg</code> dari URL yang diberikan asisten.</li> <li>2. Tambahkan kode-kode yang dikerjakan pada praktikum materi 3, 4, 5, 6, dan 8 ke dalam modul</li> <li>3. Ubahlah versi menjadi versi 0.0.2</li> <li>4. Tambahkan file-file yang dibutuhkan sesuai petunjuk pada materi 9 dan buatlah file <i>custome package</i>-nya.</li> <li>5. Kumpulkan file <code>.whl</code> dan <code>.tar.gz</code> yang ada pada folder <code>dist</code></li> </ol>	Hasil kerja langkah 6	80
2.	CPL-08	CPMK-05	Bagaimana hasilnya..? Tunjukkan hasil dari pembuatan package-pakckage tersebut!	Hasil praktikum	20
<b>Total Nilai</b>					<b>100</b>

**10.7. POST TEST**

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-07	CPMK-04	<p>Cobalah membuat file <code>setup.py</code> dengan menuliskan koding berikut ini:</p> <pre>import setuptools  with open("README.md", "r") as fh:     long_description = fh.read()  setuptools.setup(     name="ai_pkg",     version="0.0.1",     author="Example Author",     author_email="author@example.com",     description="A small example package",     long_description=long_description,     long_description_content_type="text/markdown",     url="https://github.com/pypa/sampleproject",     packages=setuptools.find_packages(),     classifiers=[         "Programming Language :: Python :: 3",         "License :: OSI Approved :: MIT License",         "Operating System :: OS Independent",     ], )</pre>		30

			Eksekusilah program tersebut di atas. Tuliskan hasilnya		
2	CPL-08	CPMK-05	Buatlah package License dan red.Me seperti yang ada dalam materi tersebut di atas. Tuliskan hasilnya	Hasil Analisa	70
<b>Total Nilai</b>					<b>100</b>

### 10.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-07	CPMK-03	20%		
2.	Praktik	CPL-08	CPMK-05	30%		
3.	Post-Test	CPL-08	CPMK-05	50%		
<b>Total Nilai</b>						

**LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM**

<b>Nama :</b> <b>NIM :</b>	<b>Asisten:</b> <b>Paraf Asisten:</b>	<b>Tanggal:</b> <b>Nilai:</b>
-------------------------------	--	----------------------------------

--

## PRAKTIKUM 11: IMPLEMENTASI KECERDASAN BUATAN (SISTEM PAKAR, JARINGAN SYARAF TIRUAN)

**Pertemuan ke** : 11

**Total Alokasi Waktu** : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 30 menit
- Post-Test : 30 menit

**Total Bobot Penilaian** : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

**Pemenuhan CPL dan CPMK:**

CPL-07	Mampu memilih, membuat dan menerapkan teknik, sumber daya, penggunaan perangkat teknik modern dan implementasi teknologi informasi untuk memecahkan masalah
CPMK-04	Mampu menerapkan berbagai model representasikan pengetahuan untuk memecahkan masalah ke dalam suatu aplikasi AI
CPL-08	Mampu merancang dan mengimplementasikan algoritma/metode dalam mengidentifikasi dan memecahkan masalah yang melibatkan perangkat lunak dan pemikiran komputasi
CPMK-05	Mampu menganalisa masalah dengan menerapkan berbagai metode penalaran untuk menarik kesimpulan dalam mencapai solusi dan mengimplementasikannya dalam suatu aplikasi AI

### 11.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mampu menjelaskan berbagai aplikasi AI untuk penyelesaian masalah dalam dunia nyata.
2. Mampu merancang aplikasi AI untuk penyelesaian masalah dalam dunia nyata dengan menggunakan aplikasi GUI.

## 11.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-07	CPMK-04	Kemampuan mahasiswa dalam memilih teknologi informasi yang sesuai untuk menyelesaikan masalah dunia nyata dengan aplikasi AI
CPL-08	CPMK-05	Kemampuan mahasiswa dalam membuat rancangan aplikasi AI untuk penyelesaian masalah dalam dunia nyata

## 11.3. TEORI PENDUKUNG

Semakin pesatnya perkembangan AI yang dipakai dalam kehidupan sehari-hari, menyebabkan berbagai aplikasi penyelesaian masalah manusia ikut berkembang. Berbagai aplikasi-aplikasi dalam bidang AI, diantaranya:

1. Sistem Pakar
2. Pengenalan Pola
3. Bahasa Alami
4. Computer vision
5. Deep learning
6. Pengolahan Citra
7. Data Mining
8. Robotika
9. Dan lain sebagainya

Semua aplikasi ini di desain sesuai kebutuhan manusia dalam menyelesaikan masalah. Aplikasi AI dibangun dengan berbagai bahasa pemrograman non prosedural dan procedural. Bahasa pemrograman banyak dipakai diantaranya MYCin, matlab, prolog, dan python. Namun bahasa prosedural juga dapat dipakai untuk membangun aplikasi AI, namun membutuhkan keahlian programming yang kompleks.

Pada praktikum kali ini menggunakan aplikasi GUI untuk membuat desain prototype untuk aplikasi AI. Tools ini tersedia secara free, dan dapat digunakan oleh seorang desain antarmuka sebelum implementasi kedalam suatu bahasa pemrograman.

## 11.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Perangkat Lunak: Aplikasi GUI
2. Perangkat keras: computer dan periperalnya



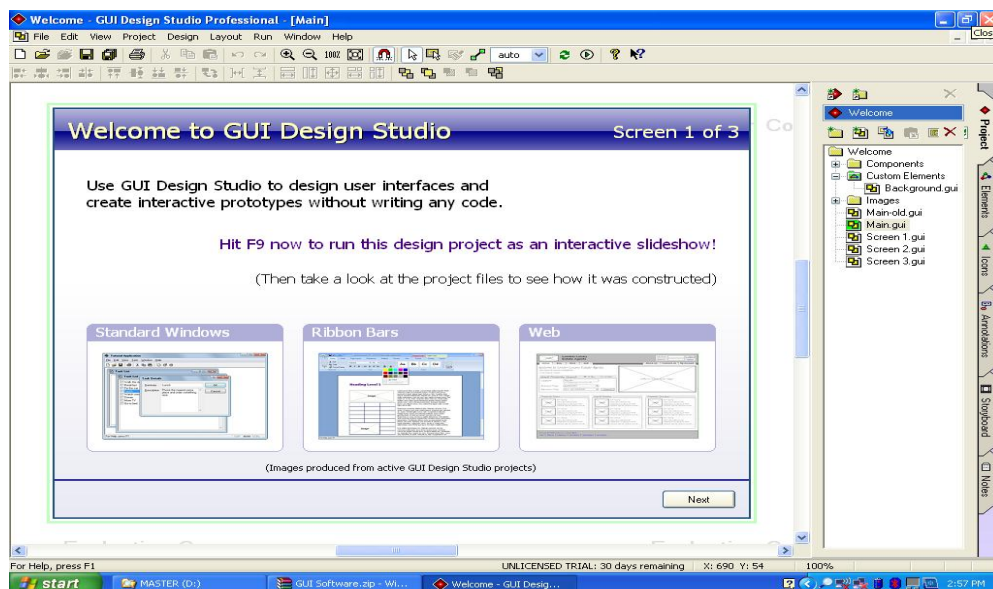
### 11.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1	CPL-07	CPMK-04	Jelaskan Jenis-jenis aplikasi AI yang Anda ketahui	30
2	CPL-07	CPMK-04	Jelaskan bagaimana karekteristik aplikasi AI untuk jenis Sistem Pakar..?	30
3	CPL-08	CPMK-05	Pada Praktikum ini kita menggunakan aplikasi GUI untuk membuat desain aplikasi. jelaskan tahapannya!	40
<b>Total</b>				<b>100</b>

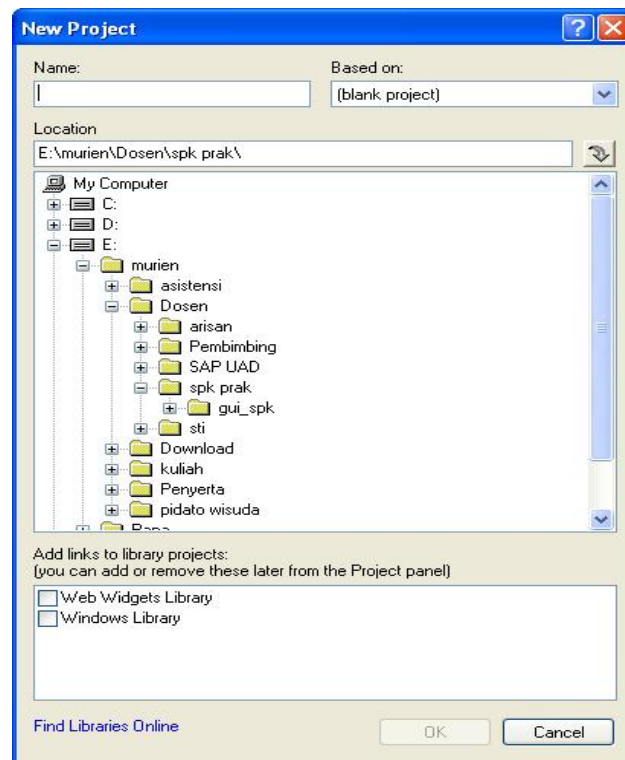
### 11.6. LANGKAH PRAKTIKUM

1. Peserta duduk di kursi dan kompyuer yang telah disediakan (ini dipakai untuk seterusnya hingga praktikum terakhir.
2. Praktikan wajib membaca materi praktikum sebelum dilaksanakan
3. Praktikan mengikuti *pre-test* yang diberikan oleh Asistem praktikum
4. Memastikan bahwa perangkat lunak yang digunakan yaitu pemrograman Phyton telah tersedia.
5. Mengerjakan semua tugas praktikum sesuai dengan waktu yang disediakan.
6. Ikutilah langkah penggunaan GUI dalam membuat aplikasi Sistem pakar. Langkah menggunakan *GUI Design Studio* sebagai berikut:
  - a. Buka *GUI Design Studio* maka tampilan form utamanya dapat dilihat pada Gambar 11.1.



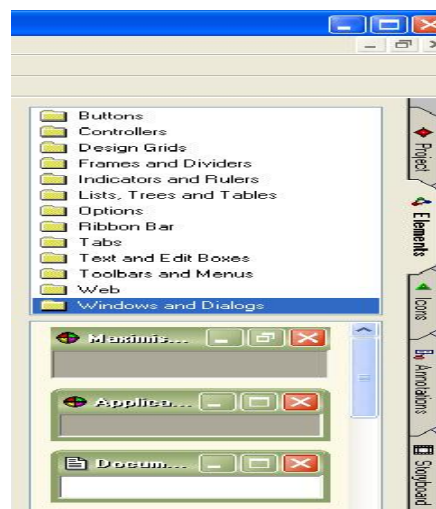
Gambar 11.1 Tampilan Form Utama GUI Design Studio

- b. Setelah form utama terbuka, pilih *New Project* dan berikan nama project anda serta jangan lupa memilih lokasi drive mana anda akan menyimpannya. Bila ingin menambahkan link untuk library diberikan centang. Seperti Gambar 11.2.



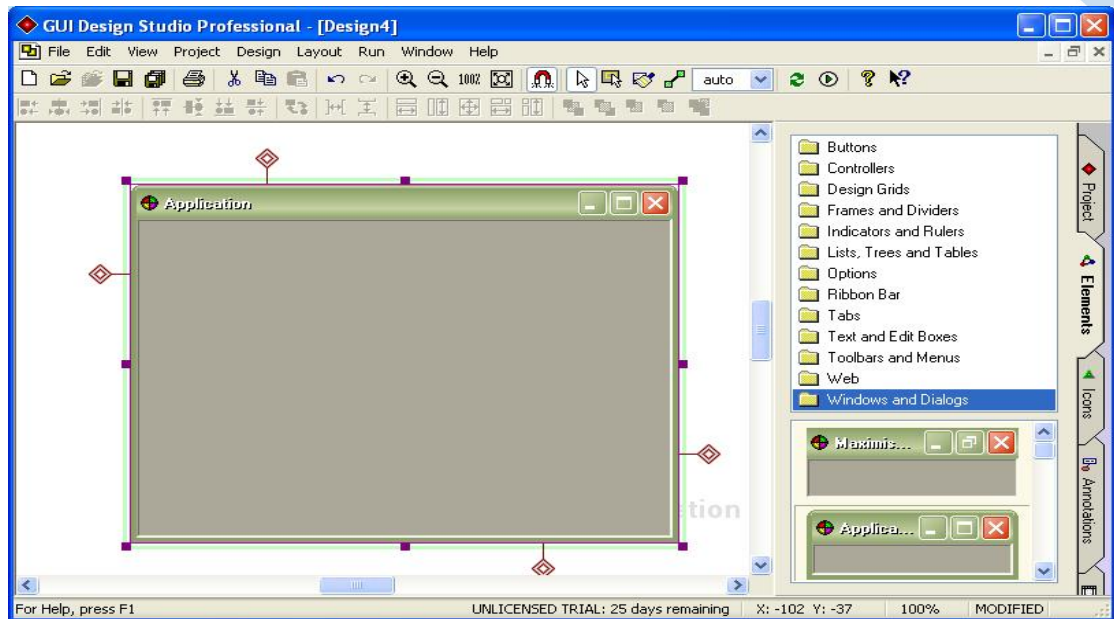
Gambar 11.2 Form untuk memilih tempat penyimpanan New Project

- c. Setelah tersimpan dengan project baru maka anda siap mendesain rancangan yang anda inginkan. Anda dapat menggunakan berbagai macam komponen ada disebelah kiri layar. Ada *Project*, *Element*, *Icon* dan lain-lain. Seperti Gambar 11.3.



Gambar 11.3 Komponen untuk membangun interface

- d. Untuk memulai desain rancangan, pertama silakan buat dulu *form* desain, dengan cara klik *New Design* pada project atau dengan Ctrl+N. Setelah keluar jendela dialog baru silakan anda menambahkan komponen. *Drag* dan *drop* komponen yang sudah dipilih yang diinginkan sesuai dengan tema rancangan. Seperti Gambar 10.4.

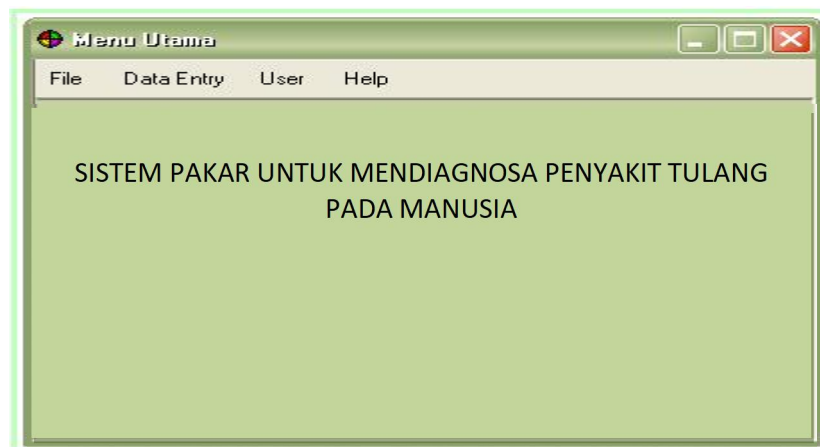


Gambar 11.4 Tampilan lembar kerja tempat membuat desain interface

- e. Untuk pengaturan setiap komponen atau elemen bisa dilakukan dengan cara *double click* komponen yang akan dimodifikasi. Dan untuk menggeser elemen bisa menggunakan tombol panah atas-bawah dan kiri-kanan pada *keyboard*, untuk mengecilkan dan membesarkan ukuran elemen bisa menggunakan kombinasi tombol Shift dan tombol 4 arah yang di sebutkan tadi.

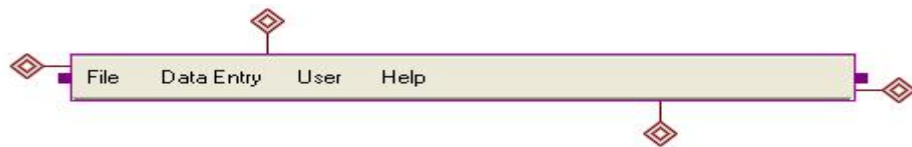
## Desain Interface

### a. Menu Utama



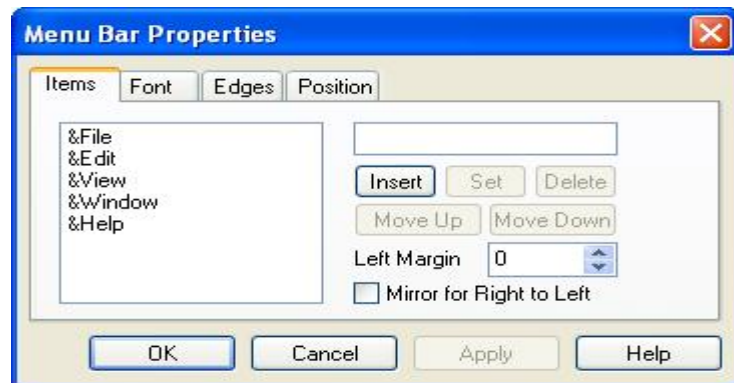
Gambar 11.5 Rancangan menu utama

Pada menu utama terdapat beberapa menu, yaitu menu file, data entry, user dan help. Menu file memiliki sub menu *login*, ganti *password* dan *exit*. Data entry memiliki sub menu penyakit, penyebab, solusi, gejala dan basis aturan. User memiliki sub menu data user dan diagnosa, serta help memiliki sub menu *about*, petunjuk penggunaan dan programmer. Untuk membuat task bar seperti di bawah ini :



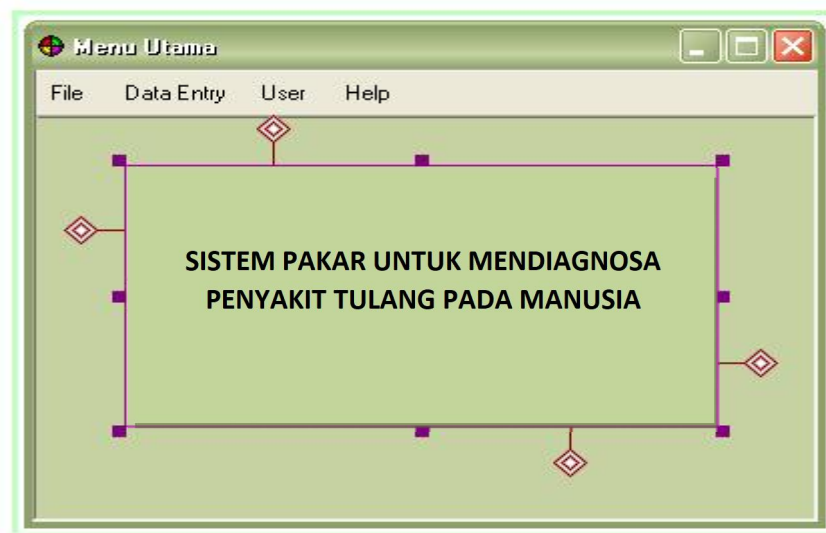
Gambar 11.6 Image Task Bar

- Pilih Elements → Toolbars dan Menus
- Tariklah image Menu Bar, bila ingin melakukan perubahan double klik pada image Menu Bar, kemudian lakukan setting pada Menu Bar properties seperti pada gambar dibawah ini.




Gambar 11.7 Tampilan Menu Bar Properties

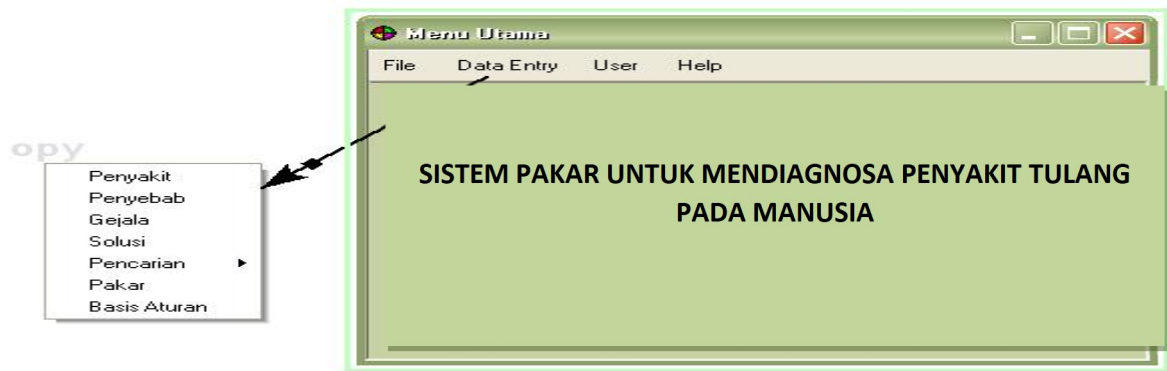
Untuk pemberian judul pada menu utama :



Gambar 11.8 Rancangan Judul Pada Menu Utama

- Pilih Elements → Text and Edit Boxes
- Kemudian lakukan setting pada Text Properties

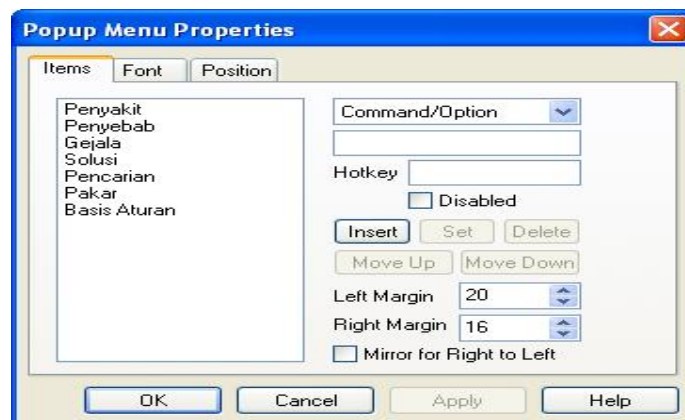
Untuk mengkoneksikan antara interface satu dengan interface lainnya dapat dilakukan link dari Menu Data Entry ke Popup Menu nya yaitu dengan memilih icon 



Gambar 11.9 Rancangan Menu Utama dengan Popup Menu

Untuk pembuatan Popup Menu dapat dilakukan dengan cara :

- Pilih Elements → Toolbars dan Menu
- Tariklah image Popup, bila ingin melakukan perubahan double klik pada image popup menu kemudian lakukan setting pada popup menu properties.



Gambar 11.10 Tampilan Popup Menu Properties

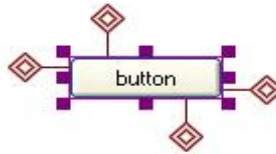
#### b. Menu Login



Gambar 11.11 Rancangan menu login

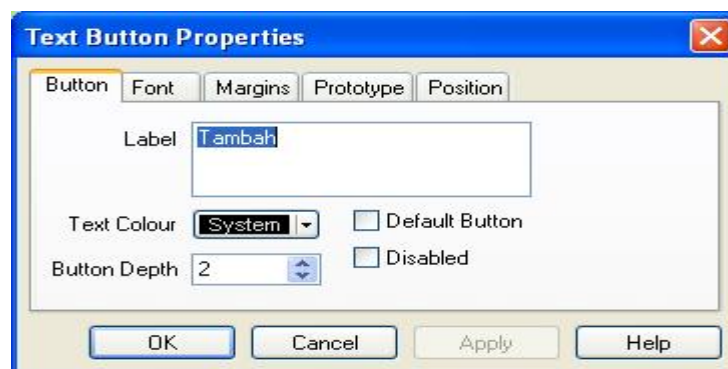
Untuk membuat tombol button :

- Pilih Elements → Buttons
- Kemudian pilih tombol button yang diinginkan, misalkan pada tombol button ini



Gambar 11.12 Image Button

- Bila ingin melakukan perubahan double klik pada tombol button kemudian lakukan penyetingan pada Text Button Properties seperti di bawah ini :



Gambar 11.13 Tampilan Text Button Properties

#### c. Menu Data Penyakit



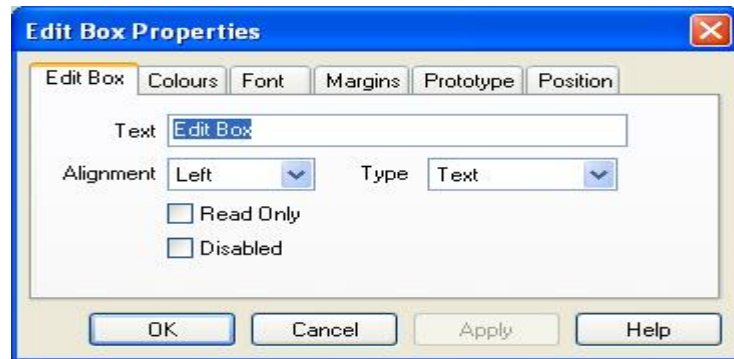
Gambar 11.14 Rancangan menu data penyakit

Menu ini dipakai untuk *input* data penyakit yang akan disimpan dalam sistem. Memungkinkan untuk menambahkan jenis penyakit baru, ataupun untuk menghapus salah satu jenis penyakit.

Untuk membuat edit text :

- Pilih Elements → Text and Edit Boxes
- Pilih image

Kemudian lakukan setting pada Edit Box Properties seperti gambar di bawah ini dengan cara double klik.



Gambar 11.15 Tampilan Edit Box Properties

d. Menu Data Gejala

Gambar 11.16 Rancangan menu data gejala

Menu ini dipakai untuk *input* data gejala dan probabilitas masing-masing penyakit. Memungkinkan untuk menambahkan gejala baru, ataupun untuk menghapus salah satu gejala.

e. Menu Data Penyebab

Gambar 11.17 Rancangan menu data penyebab

Menu ini dipakai untuk *input* data penyebab penyakit. Memungkinkan untuk menambahkan penyebab baru, ataupun untuk menghapus salah satu penyebab penyakit.

f. Menu Data Solusi

Gambar 11.18 Rancangan menu data solusi

Menu ini dipakai untuk *input* data solusi. Memungkinkan untuk menambahkan solusi baru, ataupun untuk menghapus salah satu solusi dari masing-masing penyakit.

g. Menu Data Pasien



Gambar 11.19 Rancangan menu data pasien

Menu ini dipakai untuk *input* data pasien. Memungkinkan untuk menambahkan pasien baru yang akan menggunakan sistem ini untuk berkonsultasi, ataupun untuk menghapus salah satu pasien.

#### h. Menu Aturan

Gambar 11.20 Rancangan menu aturan

Menu ini digunakan untuk menampilkan kaidah aturan yang dipakai untuk melakukan penelusuran terhadap penyakit dan solusi sesuai penyakit tersebut berdasarkan gejala.

#### i. Menu Diagnosa

Gambar11.21 Rancangan menu konsultasi

Menu ini digunakan untuk *user* yang ingin melakukan konsultasi. Nantinya *user* akan memilih gejala yang dirasakan.

i. Form Rekam

Gambar 11.22 Rancangan menu rekam

Menu ini lanjutan dari menu konsultasi. Jika pada menu konsultasi *user* memilih untuk rekam, maka gejala yang telah dipilih pada menu konsultasi akan ditampilkan pada form ini.

j. Rancangan Menu Lanjut

Gambar11.23 Rancangan menu lanjut

Menu ini lanjutan dari form rekam. Pada form ini, sistem akan menampilkan solusi penyakit yang sudah tersimpan dalam *database* yang gejalanya mirip dengan gejala baru yang dimasukkan oleh *user*.

## k. Cetak Hasil

Gambar 11.24 Rancangan menu cetak hasil

Menu ini lanjutan dari form lanjut. Jika pada form lanjut *user* memilih untuk cetak, maka hasil konsultasi tersebut dapat dicetak.

## l. Menu Help

Gambar 10.25 Rancangan menu help

Menu ini akan menampilkan pilihan *about* yang berisi tentang program, petunjuk penggunaan program dan identitas programmer.

**Aturan Penilaian (Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-08	CPMK-05	Lakukanlah langkah 6 untuk membuat GUI dari aplikasi sistem cerdas	Hasil kerja langkah 6	100
<b>Total Nilai</b>					<b>100</b>

### 11.7. POST TEST

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-08	CPMK-05	Dengan menggunakan aplikasi GUI, buatlah desain prototype dari salah satu aplikasi yang ada dalam kategori sebagai berikut (Pilih salah satu dari aplikasi yang dinyatakan): 1. Pengenalan Pola 2. Bahasa alami 3. Computer Vision 4. Data Mining		100
<b>Total Nilai</b>					<b>100</b>

### 11.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-07	CPMK-04	20%		
2.	Praktik	CPL-08	CPMK-05	30%		
3.	Post-Test	CPL-08	CPMK-05	50%		
<b>Total Nilai</b>						

**LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM**

<b>Nama :</b> <b>NIM :</b>	<b>Asisten:</b> <b>Paraf Asisten:</b>	<b>Tanggal:</b> <b>Nilai:</b>
-------------------------------	--	----------------------------------

--

## DAFTAR PUSTAKA

- [1] Hermawan, A. 2006. Jaringan Syaraf Tiruan. ANDI OFFSET, Yogyakarta
- [2] Kusumadewi S. 2003. Artificial Intelligence (Teknik dan Aplikasinya). Graha Ilmu, Yogyakarta.
- [3] Prateek, Joshi, Artificial Intelligence with Python, 2017.
- [4] Russel, Stuart, Artificial Intelligence a Modern Aproach, 2016
- [5] <https://github.com/logpy/logpy>
- [6] <https://packaging.python.org/tutorials/packaging-projects/>
- [7] <http://mbp.sourceforge.net/index.html>, tanggal akses 9 Agustus 2014, jam 13.10 WIB.
- [8] Winiarti,S, Buku Praktikum Kecerdasan Buatan UAD, 2016.
- [9] Winiarti.S, Buku Praktikum Kuliah Sistem Informasi, 2014.
- [10] Winiarti, Zahrotun S, Modul praktikum Statistika Informatika UAD 2016

