PP/018/III/R4

LABORATORIUM INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS AHMAD DAHLAN

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

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 pelaksanaanya 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 petnujuk 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.

	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.

Miftahurrahma Rosyda, S.Kom, M.Eng.

	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

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 ditambahnya 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 *Custom Package* 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 kuiah 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	Kepala Laboratorium Praktikum Informatika
Dewi Soyusiawaty,S.T., M.T	**Lisna Zahrotun, S.T., M.Cs.**
NIY. 60040497	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 diasisteni 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

DAFTAR GAMBAR

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	1. **1 kelas google class room** dibuat sesuai jadwal praktikum yang sudah tersedia (persesi). 2. Jika satu kelas praktikum 40 praktikan, maka dibagi menjadi 4 kelompok, **Kelompok praktikum dibagi menjadi 10 praktikan** per room dengan 1 asisten pendamping untuk setiap kelompok. 3. **Pret test** diberikan selama **15 menit** dengan menggunakan google form dan toleransi waktu 10 menit 4. **Materi praktikum** didistribusikan melalui Google Class Room dalam bentuk PPT dan video 5. **Penyampaian materi** akan dibuat video penyampaian materi dengan **durasi 15 menit** oleh asisten penanggung jawab 6. Pelaksanaan **Praktikum** diberikan dengan mengerjakan sesuai langkah-langkah praktikum dengan **durasi 30 menit** 7. **Post Test** diberikan dan diselaikan pengerjannya dengan durasi jam melalui google class room dengan **durasi 30 menit** 8. **Tugas dosen pengampu:** membuat soal pre test, membuat soal response, berkoordinasi dengan setiap asisten sesuai sesi dan kelompoknya 9. **Tugas asisten:** membuat video penjelasan materi, koreksi pre test, pelaksanaan praktikum dan post test
2	Implementasi Pencarian Buta	90 menit	**Sama dengan minggu 1**
3	Implementasi Pencarian Heuristik	90 menit	**Sama dengan minggu 1**
4	Representasi Pengetahuan	90 menit	**Sama dengan minggu 1**

5	Implementasi *Forward Chaining* dan *backward Chaining*	90 menit	**Sama dengan minggu 1**
6	Implementasi Representasi Pengetahuan Dengan Model Pohon	90 menit	**Sama dengan minggu 1**
7	*Hierarcy Planning*	90 menit	**Sama dengan minggu 1**
8	*Multi Agent*	90 menit	**Sama dengan minggu 1**
9	Implementasi representasi dengan semantic Net	90 menit	**Sama dengan minggu 1**
10	Implementasi Probabilstik dalam AI	90 menit	**Sama dengan minggu 1**
11	Pembuatan Package dalam Phyton	90 menit	**Sama dengan minggu 1**
12	Merancang Aplikasi Dalam Kecerdasan Buatan (Sistem Pakar, Jaringan Syaraf Tiruan)	90 menit	**Sama dengan minggu 1**

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 popular 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 pemograman 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 komperehensif. 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 Phyton (Jupiter 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! 2. Jelaskan fungsi mesin inferensi dalam suatu pelacakan solusi! 3. Jelaskan 2 perbedaan model pelacakan graph dengan model pelacakan pohon!	25 35 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 *package*nya. 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 substract
  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 varibel 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,)`

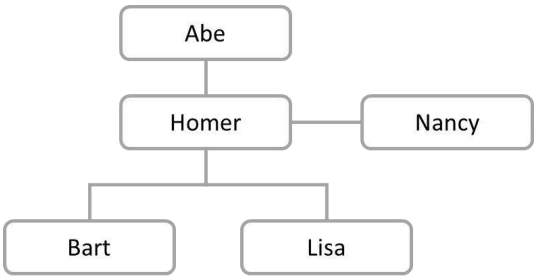
Fakta dan Relasi

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`	`fact(father("Abe","Homer"))`
California is a coastal	`coastal`	`fact(coastal("California"))`
California is adjacent to Arizona	`adjacent`	`fact(adjacent("California, "Arizona"))`

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	`fact(father( "Homer", "Bart"))`
Homer is the father of Lisa	`fact(father( "Homer", "Lisa"))`
Abe is the father of Homer	`fact(father( "Abe", "Homer"))`

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 baris6 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
```

```
def get_sibling(x, y):
    temp = var()
    return conde((parent(temp, x), parent(temp, y)))

if __name__=='__main__':
    parent = Relation()
    facts(parent, ("Homer", "Bart"),
                  ("Homer", "Lisa"),
                  ("Abe",  "Homer"))
    x = var()
    output = run(1, x, parent(x, "Bart"))
    print("\nNama ayah Bart : ", output[0])

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

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

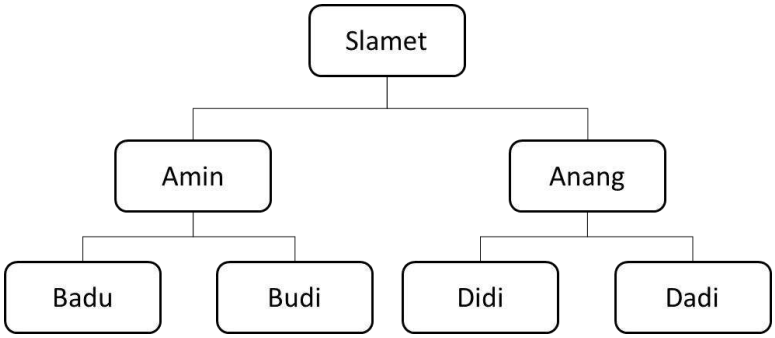
Aturan Penilaian (Total Skor: 100):

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
				Total Nilai	**100**

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	Coba ketik program di atas dengan bahasa Python. Caranya: a. Buka jendela editor sesuai pilihan b. Untuk menulis sintaks Program tulis pada jendela editor c. Ketik semua program di atas (listing 1.5) d. Run program dengan menggunakan console e. Perhatikan apa yang terjadi f. Cobalah beberapa query	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. ***Hint***: lihat kembali contoh pada Listing 1.4)	Hasil running program	60
				Total Nilai	**100**

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%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

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 pemgrograman

2.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mengetahui cara kerja pemrograman Phyton sebagai salah satu bahasa pemrograman yang popular 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

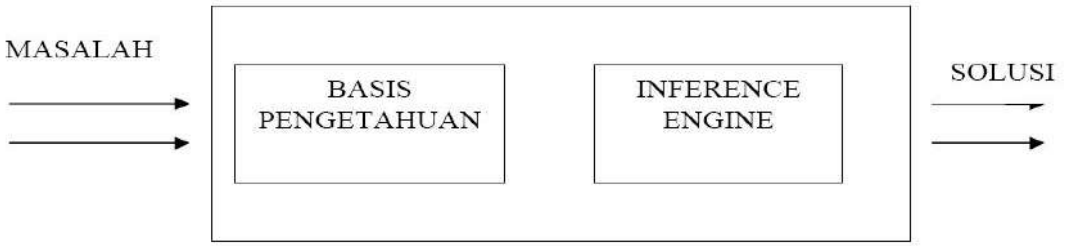
Indikator ketercapaian diukur dengan:

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 (*knowlage base*). Sistem harus memiliki *inference engine* agar sistem mampu mengambil kesimpulan berdasarkan fakta atau pengetahuan. *Output* yang diberikan berupa solusi masalah sebagai hasil dari interfrensi.

Secara umum untuk membangun sistem uang mampu menyelesaiakn masalah perlu mempertimbangkan 4 hal:

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

b. Penyelesaian Masalah dalam AI

Dalam pemyelesaian masalah dengan teknik AI menyangkut beberapa langkah yaitu:

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

Dalam menyelsaikan 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 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:

$R(K_1, K_2, ..., K_N)$

$R(K_{J1}, K_{J2}, ..., K_{JN})$

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 “Permaian Catur”, maka harus ditentukan:

1. Posisi awal pada papan catur

   Posisi awal setiap permaian catur selalu sama, yitu 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 hotizontal 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 uang 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 peripheralnya, 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! 2. Jelaskan kelebihan serta kekurangan pelacakan buta! 3. Jelaskan langkah-langkah pencarian solusi dengan pelacakan Buta untuk jenis Deep First Search lengkapi dengan contoh graphnya!	25 25 50
			Total	**100**

2.6. LANGKAH PRAKTIKUM

Dalam praktikum ini menggunakan bahasa pemrograman Phyton. Para praktikan diminta memjalankan 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 phyton untuk kasus pelacakan	Hasil pekerjaan praktikum langkah 6	25

2.	CPL-05	CPMK-03	Lakukan tugas praktikumlangkah 7.
			a. Bagimanakah hasilnya..?
			b. Berdasarkan program pada listing 2.2 tersebut berikanlah goal di bawah ini untuk kedua program tersebut. Bagaimana hasilnya? (hint: gunakan `membero`)
			`Query 1:`
			`suka("ellen",X),suka("tom",X)`
			`Query 2:`
			`suka("mary",X),suka("ann",X)`
			c. Cobalah untuk menambahkan fakta-fakta baru untuk pengembangan program pada listing 2.2. Bagimana hasilnya..?
				Hasil pekerjaan praktikum langkah 7	25

3.	CPL-05	CPMK-03	Lakukan langkah Praktikum 8.
			a. Jalankan program tersebut dengan menambahkan *goal* sebagai berikut:
			`ukuran(z, "besar").`
			Tulis ke dalam program dan tampilkan keluaran program
			`warna(z, "cokelat").`
			Tulis ke dalam program dan tampilkan keluaran program
			Tambahkan kode ke dalam program untuk menampilkan binatang yang
				Hasil pekerjaan praktikum langkah 8	15

			berukuran besar **dan** berwarna cokelat b. Tambahkan kode ke dalam program untuk menampilkan binatang yang: berwarna gelap berukuran besar **dan** berwarna gelap 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.		15 20
				Total Nilai	**100**

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
				Total Nilai	**100**

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%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

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 pemgrograman

3.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mengetahui cara kerja pemrograman Phyton sebagai salah satu bahasa pemrograman yang popular 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 permasalahn 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 - Fisth 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 Phyton
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! 2. Berilah contoh kasus penerapan pelacakan heuristic dalam kehidupan nyata! 3. Buatlah contoh kasus graph pelacakan dengan salah satu metode Heuristik!	25 30 45
			Total	**100**

3.6. LANGKAH PRAKTIKUM

Untuk dapat menjalankan praktikum untuk metode pelacakan *heuristic*, perlu dilakukan langkah persipan 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 selayaknya instalasi *package* dari sever Python.

```
pip install <nama_file_whl>
```

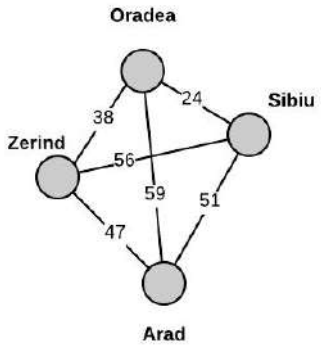
b. Melakukan strukturisasi Modul Pencarian dalam *Package*

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 struktir 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. Lihar 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

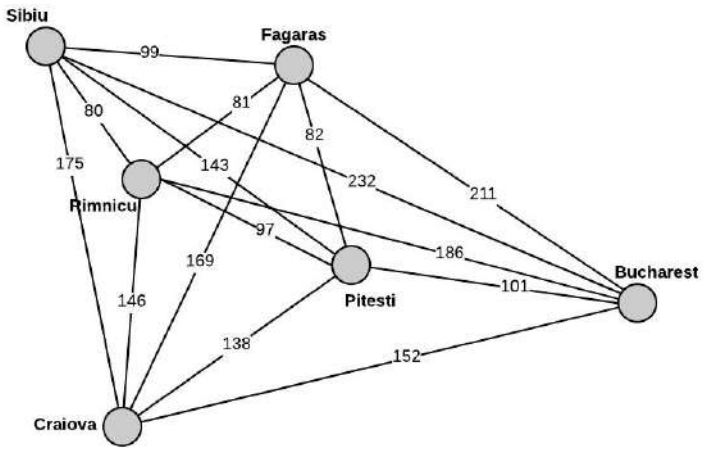
Jawablah pertanyaan berikut:

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: a. Unduh *custom package* melalui URL yang diberikan asisten dan instal *package* tersebut dan *import* modul yang ada pada *package* untuk menggunakannya dalam program. b. Tuliskan program yang ada pada contoh kasus gambar 3.1 dan tambahkan listing 3.4 untuk menjalankan program c. Program TSP dapat dijalankan dengan cara membuat objek baru `TSP_problem` dengan parameter `all_cities`. Selanjutnya panggil fungsi `hill_climbing` dengan parameter objek yang baru saja dibuat. d. Jalankan program tersebut dan tampilkan hasilnya	Hasil implementasi koding	50
				Total Nilai	**100**

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
				Total Nilai	**100**

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%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

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 menerapakan 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 Produkction 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 Phyton

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 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 £ 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*, …. 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 ~*p*, adalah proposisi tidak *p.*

Catatan:

1. *Beberapa literatur menggunakan notasi "Øp", " p ", atau "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

~*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 Phyton
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! 2. Jelaskan peranan representasi pengetahuan dalam suatu aplikasi AI! 3. Jelaskan penerapan aplikasi AI yang menggunakan representasi pengetahuan berbentuk frame, list dan Naskah! **(Setiap contoh bernilai 15)**	25 30 45
			Total	**100**

4.6. LANGKAH PRAKTIKUM

1. Peserta duduk di kursi dan kompuyer 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
				Total Nilai	**100**

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 dipersiapkankan!	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 kompuyer 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	Untuk membuat *custom package* `ai_pkg` versi 0.0.2 dilakukan dengan dengan cara: 1. Unduh semua file modul yang ada *custom package* `ai_pkg` dari URL yang diberikan asisten. 2. Tambahkan kode-kode yang dikerjakan pada praktikum materi 3, 4, 5, 6, dan 8 ke dalam modul 3. Ubahlah versi menjadi versi 0.0.2 4. Tambahkan file-file yang dibutuhkan sesuai petunjuk pada materi 9 dan buatlah file *custome package*-nya. 5. Kumpulkan file `.whl` dan `.tar.gz` yang ada pada folder `dist`	Hasil kerja langkah 6	80
2.	CPL-08	CPMK-05	Bagaimana hasilnya..? Tunjukkan hasil dari pembuatan package-pakckage tersebut!	Hasil praktikum	20
				Total Nilai	**100**

10.7. POST TEST

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-07	CPMK-04	Cobalah membuat file setup.py dengan menuliskan koding berikut ini:		30

```
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",
    ],
)
```

			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
				Total Nilai	**100**

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%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

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 menerapakan 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 menyelesaiakan 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, menyebakan 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 bhasa pemrograman non proseduran dan procedural. Bahasa pemrograman banyak dipakai diantaranya MYCin, matlab, prolog, dan phyton. Namun bahasa proseural 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 tersedi 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

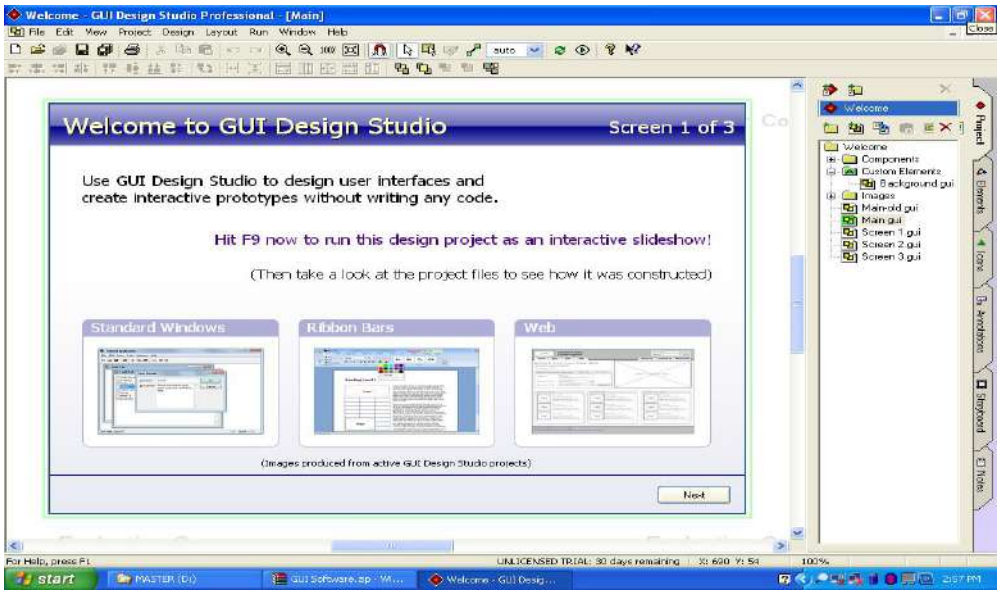
Jawablah pertanyaan berikut (**Total Skor: 100):**

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
			Total	**100**

11.6. LANGKAH PRAKTIKUM

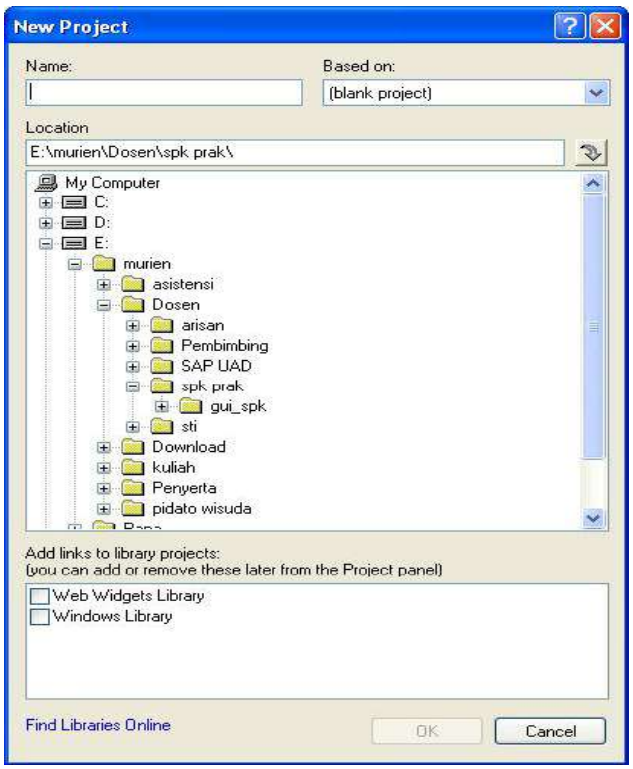
1. Peserta duduk di kursi dan kompuyer yang telah disediakan (ini dipakai untuk seterusnya hingga praktikum terakhir.
2. Praktikan wajib membaca materi praktikum sebelum dilaksanakan
3. Praktikan mengikuti *pre-test* yang diberikan oleh Asistem praktikum
4. Memastikan bahwa perangkat lunak yang digunakan 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.

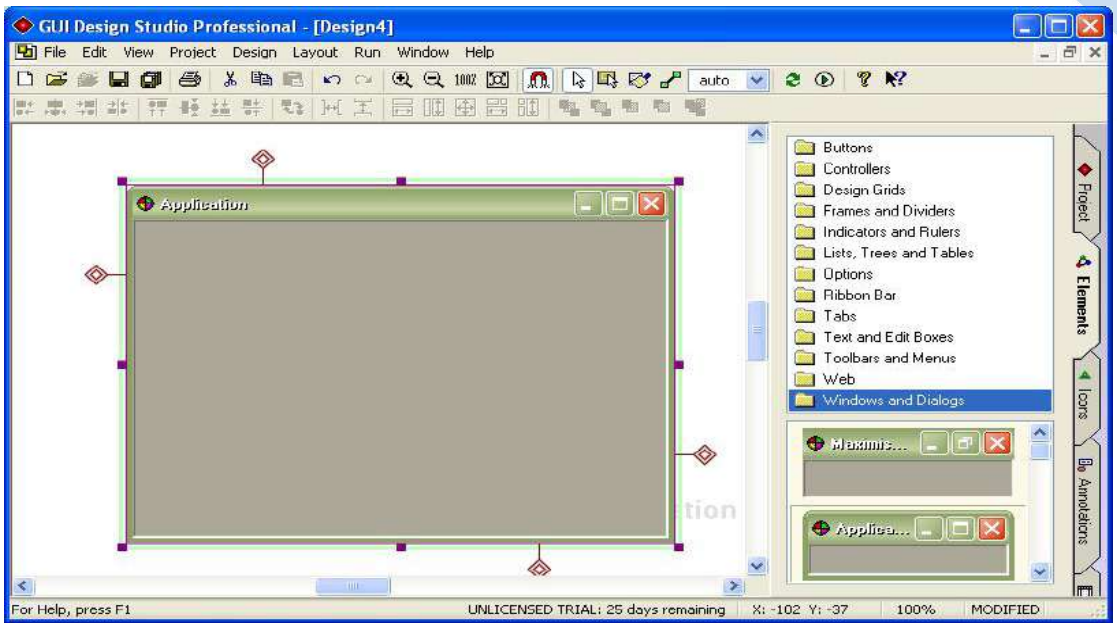


Gambar 11.2 Form untuk memilih tempat penyimpanan New Project

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 Crtl+N. Setelah keluar jendelan dialog baru silakan anda menambahkan komponen. *Drag* dan *drop* komponen yang sudah dipilih yang diinginkan sesuai dengan tema rancangan. Seperti Gambar 10.4.

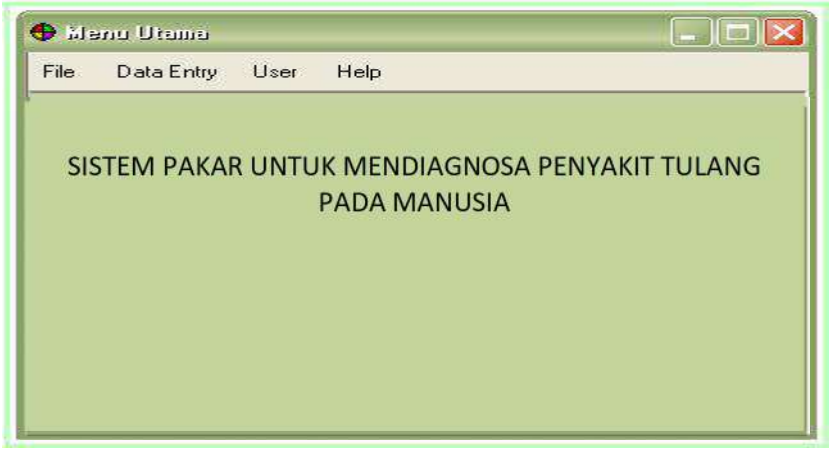


Gambar 11.4 Tampilan lembar kerja tempat membuat desain interface

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

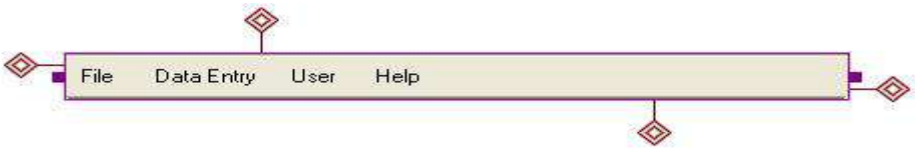
Desain Interface

a. Menu Utama



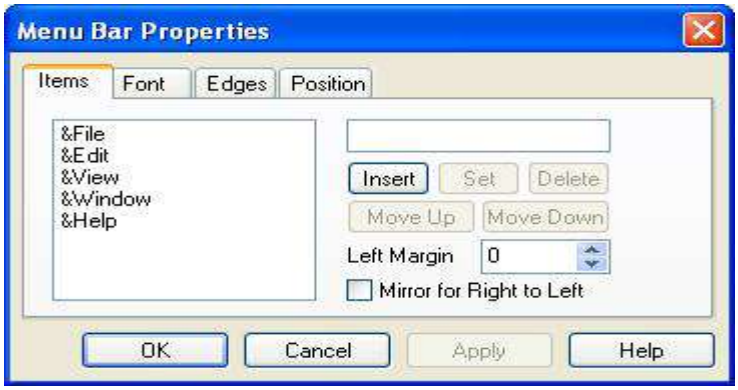
Gambar 11.5 Rancangan menu utama

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 programer. 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.



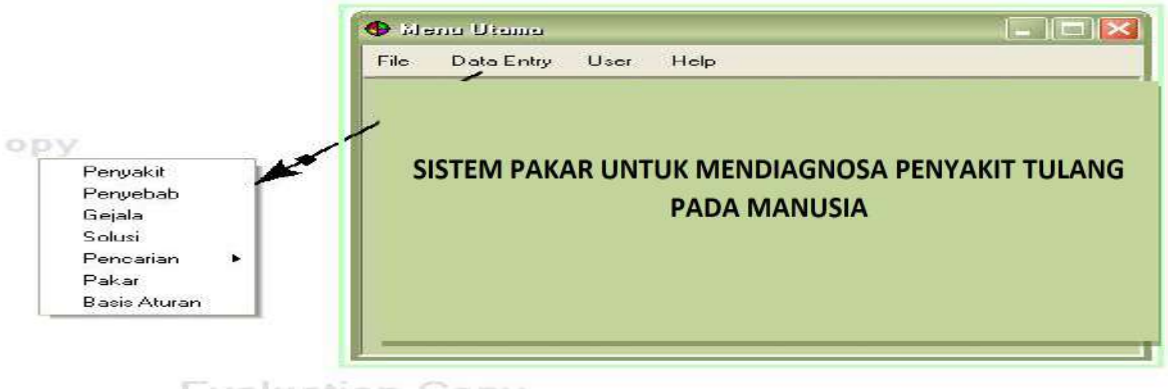
Gambar 11.7 Tampilan Menu Bar Properties

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 

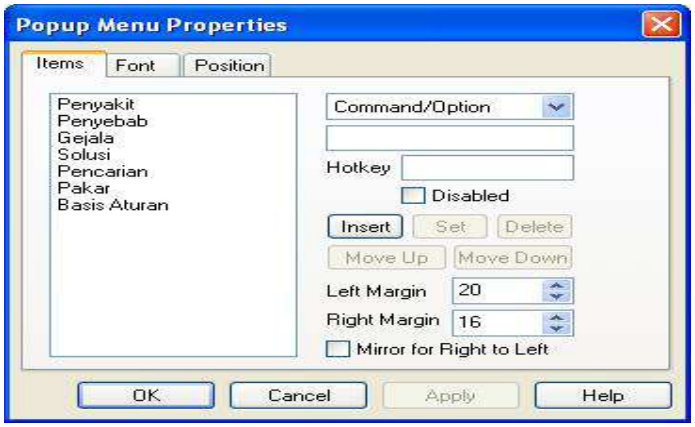


Gambar 11.9 Rancangan Menu Utama dengan Popup Menu

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.



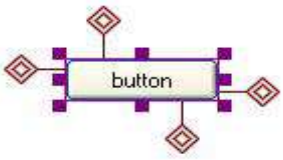
Gambar 11.10 Tampilan Popup Menu Properties

b. Menu Login

Gambar 11.11 Rancangan menu login

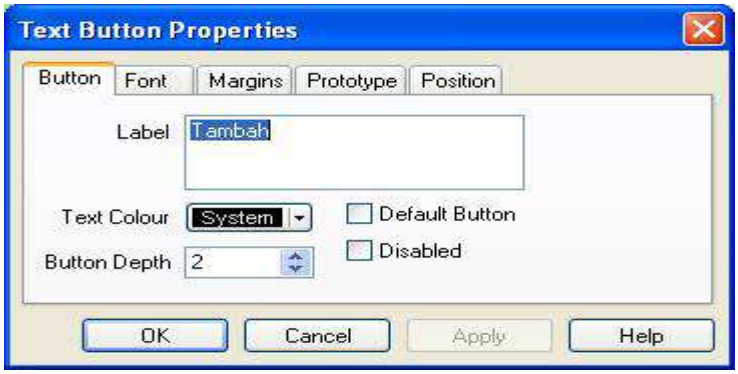
Untuk membuat tombol button :

- 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

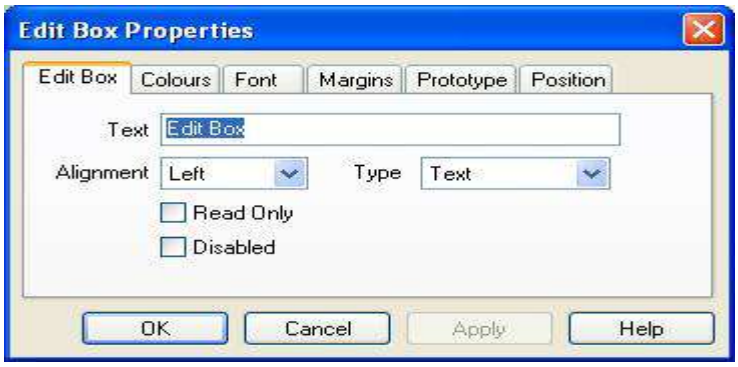
Gambar 11.14 Rancangan menu data penyakit

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 seeting pada Edit Box Proporties 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

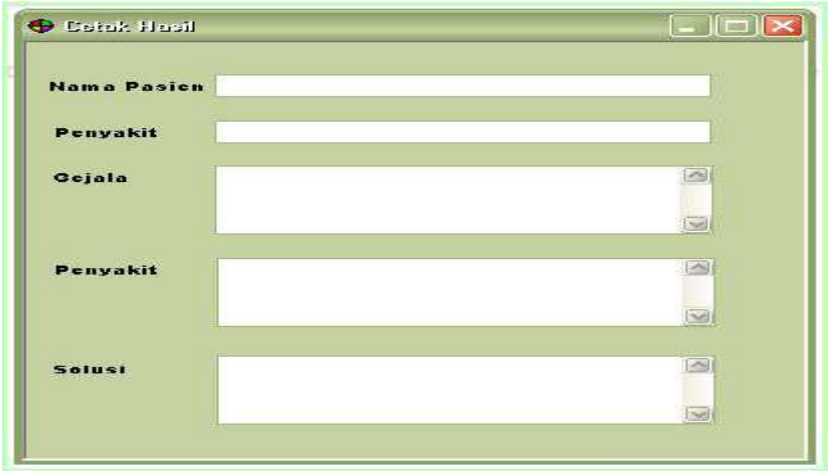
Menu ini lanjutan dari menu konsultasi. Jika pada menu konsultasi *user* memilih untuk rekam, maka gejala yang telah dipilih pada menu konsultasi akan ditampilkan pada form ini.

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
				Total Nilai	**100**

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
				Total Nilai	**100**

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%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

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


LABORATORIUM
TEKNIK INFORMATIKA
UNIVERSITAS AHMAD DAHLAN


UNIVERSITAS AHMAD DAHLAN