CLIPS

C LANGUAGE INTEGRATED PRODUCTION SYSTEM

Lecture# 4
Expert System lab Work
INTRODUCTION

Representation and Reasoning Knowledge

Rule-based

Procedural

CLIPS

OOP
Environment in CLIPS
3 Main Component

- Set Of Rules
- Fact list
- To Control Execution Process

Knowledge Base
Inference Engine
NOTATION

- Like LISP programming Language signed by double colon “()”.

- e.g :
  
  `(exit)`

- Case Sensitive
DATA TYPE

There are 7 data type called as CLIPS primitive data type:

1. Float
2. Integer
3. Symbol
4. String
5. external address
6. instance name
7. instance address.
START WITH CLIPS

• By Default interface of CLIPS is *command prompt* as interpreter
FACTS

- To solve the problem in expert system, we must have data will became a resource of knowledge.

- Data or Information in CLIPS called as set of facts.

- The facts consist of `relation-name` and followed by `slot`
EXAMPLE

Relation name
Value Ninon, 15, brown, black

(person
  (name "ninon")
  (age 15)
  (eye-color brown)
  (hair-color black))

Slot Name, age, eye-color, dan hair-color
DEFTEMPLATE CONSTRUCTION

Before the facts is made, CLIPS must know suitable slot for being defined to a relation name.

Mechanisme to create slot is deftemplate construction.

Deftemplate like format of the record in C or Pascal programming language.
**SYNTAX**

- Deftemplate syntax
  
  \[(\text{deftemplate} \ <\text{relation-name}>\ [\text{optional-comment}] \ <\text{slot-definition}>*\)]

- Syntax of description \(<\text{slot-definition}>\)
  
  \[(\text{slot} \ <\text{slot-name}> | (\text{multislot} \ <\text{slot-name}>))\]
example

- Deftemplate Person
  (deftemplate person
    (slot name)
    (slot age)
    (slot eye-color)
    (slot hair-color))

- Deftemplate-facts are the fact with format deftemplate
- ordered-facts are the fact without format deftemplate
ADD, MODIFICATION, DUPLICATE AND REMOVE FACTS

- **Adding Fact**
  Facts can be added to working memory with command `assert`

- **Syntax**
  `(assert <fact>)`
Example

CLIPS>
(deftemplate person
   (slot name)
   (slot age)
   (slot eye-color)
   (slot hair-color)) <ENTER>

CLIPS>
(assert (person (name "ninon")
   (age 15)
   (eye-color brown)
   (hair-color black))) <ENTER>

<fact-0>
EXAMPLE

To appear set of fact used command (facts).

CLIPS>(facts)

f-0 (person (name "ninon") (age 30) (eye-color brown) (hair-color black))

For a total of 1 fact
We can use command assert.

(assert (person (name "adeth")
(age 17)
(eye-color blue)
(hair-color brown))) <ENTER>

<Ketik perintah : (facts)>
ADITION, MODIFICATION, DUPLICATION AND DELETION FACTS

- Modification Facts
  Use command (modify) if facts have been in working memory

- General Format:
  (modify <fact-index> <slot-modifier>)}
Example

- Name Modification

```
(modify 1 (name "ade trisetyo"))
```

- Identifier of fact
- Value will be changed
- Slot will be modified
example

- Result of Modification

```
CLIPS> (modify 1 (name "ade trisetyo"))

<Fact-2>
CLIPS> (facts)
CLIPS> (facts)
f-0  (person (name "ninon") (age 15) (eye-color brown) (hair-color black))
f-2  (person (name "ade trisetyo") (age 17) (eye-color pink) (hair-color black))
```

For a total of 2 facts.
Duplication facts
Use Command (**duplicate**)

General Format:

```
(duplicate <fact-index> <slot-duplicate>)
```
DUPLICATION FACTS

Example

(duplicate 0 (age 12))
DELETION FACTS

- Deletion Facts
  Use Command *(retract)*

- General Format
  *(retract <fact-index>)*
Deletion Facts
Use Command (retract)

Example
(retract 3)
DEFFFACTS COMMAND

- Use for Initializing facts.
- Called as *initial knowledge*.
- Like *assert*. 
CON’T

- General Format

(deffacts <deffacts name> [<optional comment>] <facts>)
CON’T

- Use command reset to ensure that the facts have been in working memory.

- Command: 
  (reset)

- Sign (==>) refers that the facts have been in working memory.
DEFFACT VS ASSERT

- Assert

```clp
(deftemplate ilkom
  (slot matakuliah)
  (slot dosen)
  (slot mahasiswa))
(assert (ilkom (matakuliah sistem pakar)
              (dosen marimin)
              (mahasiswa tingkat3)))
(assert (ilkom (matakuliah data mining)
              (dosen imas sitanggang)
              (mahasiswa tingkat3)))
```
DEFFACT VS ASSERT

Assert load

```
CLIPS (V6.24 06/15/06)
CLIPS> (load "D:/ASPRAK/Praktikum_Sistem_Pakar/CLIPSWin/contoh/fact assert"
Defining deftemplate: ilkom

[CSTRCPSPR1] Expected the beginning of a construct.
FALSE
CLIPS> |
```
DEFFACT VS ASSERT

- Deffact

```clp
(deftemplate ilmukomputer
  (slot matakuliah)
  (slot dosen)
  (slot mahasiswa))

(deffacts akademik
  (ilmukomputer (matakuliah sistempakar)
    (dosen marimin)
    (mahasiswa tingkat3))

(ilmukomputer (matakuliah datamining)
  (dosen imas)
  (mahasiswa tingkat3))
```
DEFFACT VS ASSERT

- Deffact load

```
CLIPS (V6.24 06/15/06)
CLIPS> (load "D:/ASPRAK/Pra}tkum_Sistem_Pakar/CLIPSWin/contoh/fact deffact.CLP")
Defining deftemplate: ilmukomputer
Defining deffacts: akademik
```

TRUE

CLIPS>
WATCH COMMAND

- Use for Debugging Program.

- General Format
  (watch <watch-item>)
MECHANISMS OF KNOWLEDGE REPRESENTATION

RULE BASE SYSTEM

General Format of RULE:

(defrule <rule name> [<comment.]
  <pattern>* ; Left-Hand Side of the rule
=>
  <action>* ) ; Right-Hand Side of the rule
AGENDA

- Represent list of the rules have been used.

- Command (agenda)
RUN

- For Running our program.

- Commanddd
  (run)
Example

; Deftemplates
(deftemplate suhu
  (slot kondisi))
(deftemplate respon
  (slot lajureaksi))

;Make Rule
(defrule tinggi-suhu
  (suhu (kondisi tinggi))
  =>
  (assert (respon
    (lajureaksi naik))))
(defrule rendah-suhu
  (suhu (kondisi rendah))
  =>
  (assert (respon
    (lajureaksi turun))))
LOAD

CLIPS (V6.24 06/15/06)
CLIPS> (load "D:/ASPRAK/Praktikum_Sistem_Pakar/CLIPSWin/cont")
Defining deftemplate: suhu
Defining deftemplate: respon
Defining defrule: tinggi-suhu +)
Defining defrule: rendah-suhu +)
TRUE
CLIPS> |
Rule Base System

- Pattern matching
- Parameter driven
- Forward chaining
- Backward chaining
Pattern Matching in a Forward Chaining System

IF condition THEN consequent (or action).
IF the volume of a product-container is less than a specified amount, THEN increase the fill volume.
IF the sensor is dead, THEN inspect sensor.
Forward Chaining Example

Facts:
the volume of item29 is 21 the volume of item17 is 18 the temperature of item23 is -5

Rule:
IF the volume of ?item is < 20, THEN the fill amount of ?item should be increased