Stove Diagnostics

```
; The program is designed to diagnose a problem with a gas or electric
; stove and recommend the best way to fix the appliance. The problems
; and ways to fix the problem are as follows.
;
; First the program determines if the stove is electric or gas and if
; the problem is with the burners or the oven then
;
;               PROBLEM
;
; no power to the stove
;       Check the fuse box and reset any blown fuses. If this does not work
;       call a professional
;
; ELECTRIC STOVE
; burner not working
;       Trade places with a good element. If the element still does not work
;       replace it. If the element now works then remove the element and
;       check the terminals for a blue tinge or a black coating or if the
;       ends are bent. All can mean that the terminal block unit is bad.
;       Check the unit for burn marks and if found replace. Else test the
;       burner switch and replace if bad. Else you have a bad wire and
;       you must trace the wire from the power to the switch to the
;       burner to find the problem.
;
; oven not working
;       Test the oven element to see if the element is bad and replace if
;       neccessary. Else check the switch and replace if bad. Else check
;       for a bad wire.
;
; GAS STOVE
; burner not working
;       Check the burner igniters and replace if bad. Else check the
;       control switch and replace if bad. Else check the electrode
;       unit and replace if bad. Else call a professional.
;
; oven not working
;       Check the igniter for a glow. If none replace the igniter. Else
;       check and clean the orffice and replace. Check for any leaks
;       during reinstallation. Else call a professional.
```

Begining

```
CLIPS> (reset)
CLIPS> (agenda)
9980   init: f-0
For a total of 1 activation.
CLIPS>

;*******************************
;rule1:
;       initializes the program
;*******************************

(defrule init
 (declare (salience 9980))
 ?x <- (initial-fact) =>
 (retract ?x)
 (assert (need professional n))
 (assert (start)))
```

Opening Message

```
CLIPS> (agenda)
-9000  begin: f-2
For a total of 1 activation.
CLIPS> (run 1)
The program is designed to help in repairing an electric or
gas stove.  The program will take you step by step through
the procedures to repair the appliance.
do you have a
  1) electric stove
  2) gas stove
  3) or exit the program
Choose 1 - 3 -> 3
CLIPS>
;*****************************
(defrule begin
 (declare (salience -9000))
 ?x <- (start)
=>
 (printout t crlf crlf crlf
        "The program is designed to help in repairing an electric or" crlf
        "gas stove.  The program will take you step by step through" crlf
        "the procedures to repair the appliance." crlf crlf
        "do you have a" crlf
        "  1) electric stove" crlf
        "  2) gas stove" crlf
        "  3) or exit the program" crlf
        "Choose 1 - 3 -> ")
 (retract ?x)
 (assert (oven type =(read))))

;*****************************
(defrule ck-select
 (declare (salience 9998))
 ?x <- (oven type ?ch&:(not (numberp ?ch)))
 (or (test (<= ?ch 0))
     (test (> ?ch 3)))
=>
 (printout t crlf crlf
        "*** your selection must be either 1,2, or 3 ***" crlf)
 (retract ?x)
 (assert (start)))
```

Quit

```
CLIPS> (agenda)
9700   quit: f-3
-500   burner-prob: f-3
-1000  oven-prob: f-3
For a total of 3 activations.
CLIPS>
```

```
;*****************************
;rule56:
(defrule quit
 (declare (salience 9700))
 (oven type 3)
 =>
 (printout t crlf crlf crlf crlf
        "Exiting the program " crlf crlf)
 (assert (stop))
 (assert (quit)))

;*****************************
;rule57:
(defrule finished
 (declare (salience 9990))
 ?w <- (quit)
 =>
 (retract ?w)
 (halt))

;*****************************
; rule2:
(defrule end1
 (declare (salience 9200))
 ?w <- (stop)
 =>
 (retract ?w)
 (reset)
 (halt))
```

```
CLIPS> (reset)
CLIPS> (run 1)
CLIPS> (agenda)
-9000  begin: f-2
For a total of 1 activation.
CLIPS> (run 1)
The program is designed to help in repairing an electric or
gas stove.  The program will take you step by step through
the procedures to repair the appliance.
do you have a
  1) electric stove
  2) gas stove
  3) or exit the program
Choose 1 - 3 -> 3
CLIPS> (agenda)
9700   quit: f-3
-500   burner-prob: f-3
-1000  oven-prob: f-3
For a total of 3 activations.
CLIPS> (run 1)

Exiting the program

CLIPS> (agenda)
9990   finished: f-5
9200   end1: f-4
-500   burner-prob: f-3
-1000  oven-prob: f-3
For a total of 4 activations.
CLIPS> (run 1)
CLIPS> (agenda)
9200   end1: f-4
-500   burner-prob: f-3
-1000  oven-prob: f-3
For a total of 3 activations.
CLIPS> (run 1)
CLIPS> (agenda)
9980   init: f-0
For a total of 1 activation.
CLIPS>
```

There is a Problem

```
;*****************************
;rule5
(defrule burner-prob
 (declare (salience -500))
 (oven type ?) =>
 (printout t crlf crlf "Is there a problem with the burners? y or n ")
 (assert (burner problem =(read))))

;*****************************
;rule6
(defrule oven-prob
 (declare (salience -1000))
 (oven type ?) =>
 (printout t crlf crlf "Is there a problem with the oven? y or n ")
 (assert (oven problem =(read))))

;*****************************
;rule7
(defrule ck-power
 (declare (salience 1500))
 (oven type 1) =>
 (printout t crlf crlf "Does the stove have any electrical power? y or n ")
 (assert (power =(read))))
```

Burner Problem

```
The program is designed to help in repairing an electric or
gas stove.  The program will take you step by step through
the procedures to repair the appliance.
do you have a
  1) electric stove
  2) gas stove
  3) or exit the program
Choose 1 - 3 -> 1
CLIPS> (agenda)
1500   ck-power: f-3
-500   burner-prob: f-3
-1000  oven-prob: f-3
For a total of 3 activations.
CLIPS> (run 1)
Does the stove have any electrical power? y or n y
CLIPS> (agenda)
-500   burner-prob: f-3
-1000  oven-prob: f-3
For a total of 2 activations.
CLIPS> (run 1)
Is there a problem with the burners? y or n y
CLIPS> (agenda)
0      element-type: f-3,f-5
0      any-elect-working: f-3,f-5
-10    ck-element-for-wear: f-3,f-5
-1000  oven-prob: f-3
For a total of 4 activations.
CLIPS> (facts)
f-1     (need professional n) CF 1.00
f-3     (oven type 1) CF 1.00
f-4     (power y) CF 1.00
f-5     (burner problem y) CF 1.00
For a total of 4 facts.
CLIPS>
```

Problem With Burners

```
;*****************************
;rule11
(defrule element-type
 (oven type 1)
 (burner problem y) =>
 (printout t crlf crlf "Is the burner element a " crlf crlf
                       " 1) push in/pull out type" crlf
                       " 2) screw in type" crlf crlf
                       "Choice -> ")
 (assert (burner type =(read))))
;*****************************
;rule12
(defrule any-elect-working
 (oven type 1)
 (burner problem y) =>
 (printout t crlf crlf
        "Is at least one burner of each size still working? y or n ")
 (assert (some work =(read))))

;*****************************
;rule24:
(defrule ck-element-for-wear
 (declare (salience -10))
 (oven type 1)
 (burner problem y) =>
 (printout t crlf crlf
        "Electric range elements use nichrome wires made of a nickle" crlf
        "and chromium alloy covered by a metal sheathing with a black" crlf
        "coating. Does the outside coating of the element appear to " crlf
        "have worn away at any point? y or n ")
 (assert (element worn =(read))))
```

Push In/Pull Out Burner

```
CLIPS> (run 1)


Is the burner element a

 1) push in/pull out type
 2) screw in type

Choice -> 1
CLIPS> (agenda)
0      any-elect-working: f-3,f-5
-10    ck-element2: f-5,f-6
-10    blue-tinge: f-5,f-6
-10    ck-element-for-wear: f-3,f-5
-20    ck-element1: f-5,f-6
-1000  oven-prob: f-3
For a total of 6 activations.
CLIPS> (facts)
f-1     (need professional n) CF 1.00
f-3     (oven type 1) CF 1.00
f-4     (power y) CF 1.00
f-5     (burner problem y) CF 1.00
f-6     (burner type 1) CF 1.00
For a total of 5 facts.
CLIPS>
```

```
;*****************************
;rule12
(defrule any-elect-working
 (oven type 1)
 (burner problem y) =>
 (printout t crlf crlf
        "Is at least one burner of each size still working? y or n ")
 (assert (some work =(read))))

CLIPS> (run 1)
Is at least one burner of each size still working? y or n y
CLIPS> (agenda)
0      element-trade: f-7
-10    ck-element2: f-5,f-6
-10    blue-tinge: f-5,f-6
-10    ck-element-for-wear: f-3,f-5
-20    ck-element1: f-5,f-6
-1000  oven-prob: f-3
For a total of 6 activations.
CLIPS> (facts)
f-1     (need professional n) CF 1.00
f-3     (oven type 1) CF 1.00
f-4     (power y) CF 1.00
f-5     (burner problem y) CF 1.00
f-6     (burner type 1) CF 1.00
f-7     (some work y) CF 1.00
For a total of 6 facts.
CLIPS>
;*****************************
;rule 13
(defrule element-trade
 ?x <- (some work y) =>
 (printout t crlf crlf
        "Remove the element that is not working properly and exchange" crlf
        "it with another burner element of the same size." crlf crlf
        "***  be sure to cut the power before working ***" crlf crlf
        "Turn both element controls to high. Is the element that " crlf
        "was not working properly still not working? y or n ")
 (retract ?x)
 (assert (bad element =(read))))
```

```
CLIPS> (run 1)


Remove the element that is not working properly and exchange
it with another burner element of the same size.

***  be sure to cut the power before working ***

Turn both element controls to high. Is the element that
was not working properly still not working? y or n y
CLIPS> (agenda)
0      bad-element: f-8
-10    ck-element2: f-5,f-6
-10    blue-tinge: f-5,f-6
-10    ck-element-for-wear: f-3,f-5
-20    ck-element1: f-5,f-6
-1000  oven-prob: f-3
For a total of 6 activations.
CLIPS> (facts)
f-1     (need professional n) CF 1.00
f-3     (oven type 1) CF 1.00
f-4     (power y) CF 1.00
f-5     (burner problem y) CF 1.00
f-6     (burner type 1) CF 1.00
f-8     (bad element y) CF 1.00
For a total of 6 facts.
CLIPS>

;**************************
;rule14
(defrule bad-element
 ?x <- (bad element y)
 =>
 (printout t crlf crlf
        "The burner element is bad.  You must replace the bad " crlf
        "element.  Afterward, check the burner and if there" crlf
        "is still a problem rerun this program." crlf)
 (retract ?x)
 (assert (stop)))
```

Element is Bad

```
CLIPS> (run 1)


The burner element is bad.  You must replace the bad
element.  Afterward, check the burner and if there
is still a problem rerun this program.
CLIPS> (agenda)
9200   end1: f-9
-10    ck-element2: f-5,f-6
-10    blue-tinge: f-5,f-6
-10    ck-element-for-wear: f-3,f-5
-20    ck-element1: f-5,f-6
-1000  oven-prob: f-3
For a total of 6 activations.
CLIPS> (facts)
f-1     (need professional n) CF 1.00
f-3     (oven type 1) CF 1.00
f-4     (power y) CF 1.00
f-5     (burner problem y) CF 1.00
f-6     (burner type 1) CF 1.00
f-9     (stop) CF 1.00
For a total of 6 facts.
CLIPS> (run 1)
CLIPS> (agenda)
9980   init: f-0
For a total of 1 activation.
CLIPS>
```

```
;***************************
; rule2:
(defrule end1
 (declare (salience 9200))
 ?w <- (stop)
 =>
 (retract ?w)
 (reset)
 (halt))
```

Decision Tree

Structure of the program

```
init
-> begin
    1 -> 1500   ck-power: f-3
      y -> -500   burner-prob: f-3
             y -> 0     element-type: f-3,f-5
                    1 -> 0      any-elect-working: f-3,f-5
                           y -> 0      element-trade: f-7
                                  y -> 0      bad-element: f-8
                                  |        -> 9200   end1: f-9
                                  |              -> 9980   init: f-0
                                  n -> -10    ck-element2: f-5,f-6
                                       y -> 0      black-coating: f-9
                                       |     -> 0      ck-terminal-block: f-10
                                       |         y -> 0 burned-contacts: f-11
                                       |              ->  9200   end1: f-12
                                       n -> -10    blue-tinge: f-5,f-6
                                          y -> 0      blue-tinge-exists: f-10
                                             -> 0      ck-terminal-block: f-11
                                                y -> 0      burned-contacts: f-12
                                                   -> 9200   end1: f-13
```

Tree Search Space

Viewed as a Search Problem:

The Search Space is a Tree

A Particular node is a rule on the agenda
The agenda is ordered from the best to the worse rules

Upon execution of the rule, the node gets branches with new rules
Or if it fails, then it follows another branch.

Production System

Procedure **PRODUCTION**

$DATA \longleftarrow$ initial database

until $DATA$ satisfies the termination condition, **do**

begin

select some rules, A, in the set of rules that can be applied to $DATA$

$DATA \longleftarrow$ result of applying R to $DATA$

end

This is the same engine for a variety of problem solvers.

Here, the selection of rules is done by the pattern matching of the LHS

The ordering of the rules is done by the particular strategy based on the salience