# CSC/CPE 481 Midterm Exam Winter 2003

Instructor: Franz J. Kurfess

**Task I – Multiple Choice Questions**
Mark the correct answers (only one per question).

a) Which of the following is *not* a main component of a typical expert system?

- ☐ knowledge base
- ☐ inference engine
- ☐ data base
- ☐ user interface 3

b) Which of the following is the best description of *knowledge acquisition*?

- ☐ the storage of knowledge in a format suitable for processing by computers
- ☐ the transfer of knowledge from humans to computers
- ☐ a computer-based mechanism for the generation of new conclusions from existing knowledge
- ☐ a description of the reasons why a particular solution was generated 3

c) Which of the following operations constitute the "inference engine cycle"?

- ☐ knowledge acquisition, knowledge representation, and reasoning
- ☐ forward and backward chaining
- ☐ conflict resolution, execution, and match
- ☐ the RETE algorithm 3

d) What does CLIPS stand for?

- ☐ C Language Implementation Production System
- ☐ it is a combination of C (its implementation language) and Lisp (its appearance)
- ☐ Common Lisp Implementation Production System
- ☐ it is based on the initials of its developers 3

e) Which of the following CLIPS commands creates an *instance* of a fact?

- ☐ `facts`
- ☐ `deftemplate`
- ☐ `assert`
- ☐ `retract` 3

Name:

f) Which of the following Clips commands lists the instances of facts currently known to Clips?

- ☐ `facts`
- ☐ `deftemplate`
- ☐ `assert`
- ☐ `retract` 3

g) What is the role of the *antecedent* in a Clips rule?

- ☐ it contains the `defrule` keyword, the name of the rule, and an optional comment string
- ☐ it specifies the patterns that are to be matched against the facts
- ☐ it separates the antecedent and the consequent of the rule
- ☐ it contains the actions to be performed when the rule fires 3

h) Which of the following statements is the best description of *a procedural knowledge*?

- ☐ knowledge that is available prior to perception through senses
- ☐ knowledge that is verifiable through sensory perception
- ☐ knowledge that indicates how to perform some activity
- ☐ knowledge that is difficult to express through language 3

i) What does it mean that a logical sentence is *valid*?

- ☐ the sentence is true under all possible interpretations in all possible worlds
- ☐ the sentence is true under all possible interpretations in some possible worlds
- ☐ the sentence is true if there exists a true interpretation in some possible world
- ☐ the sentence is syntactically correct 3

j) What is *conflict resolution* in rule-based systems?

- ☐ If there is no rule that possibly matches the currently active facts, conflicting variable bindings may be eliminated through conflict resolution.
- ☐ If there are several rules that possibly match the currently active facts, one of them must be selected.
- ☐ Inconsistent rules in the knowledge base are modified or eliminated through conflict resolution.
- ☐ Conflict resolution is a more efficient variant of the resolution proof method for logic. 3

Subtotal Task 1: 30

| Name: |
|---|

## Task II – Short Questions

1. Describe the main differences between *natural language* and *rules* for the representation of knowledge based on the criteria below. What are the respective advantages and problems? 15

| *Aspect* | *Natural Language* | *Rules* |
|---|---|---|
| *Expressiveness* | | |
| *Comprehensibility* | | |
| *Computational Complexity* | | |
| *Advantages* | | |
| *Problems* | | |

2. What is the role of *pattern matching* in CLIPS?. 7

3. What limitations and inconveniences have you encountered in CLIPS? Describe their nature for at least three, and what you did to overcome or work around them. 8

   (a) CLIPS limitation:
   - Nature:
   - Work-around:

   (b) CLIPS limitation:
   - Nature:
   - Work-around:

   (c) CLIPS limitation:
   - Nature:
   - Work-around:

   (d) CLIPS limitation:
   - Nature:
   - Work-around:

   (e) CLIPS limitation:
   - Nature:
   - Work-around:

(f) CLIPS limitation:

- Nature:

- Work-around:

(g) CLIPS limitation:

- Nature:

- Work-around:

(h) CLIPS limitation:

- Nature:

- Work-around:

(i) CLIPS limitation:

- Nature:

- Work-around:

(j) CLIPS limitation:

- Nature:

- Work-around:

(k) CLIPS limitation:

- Nature:

- Work-around:

(l) CLIPS limitation:

- Nature:

- Work-around:

(m) CLIPS limitation:

- Nature:

- Work-around:

Subtotal Task 2: 30

**Task III – Clips Program: Building Towers**

In this task, you need to trace the evaluation of a short Clips program. This program is a variation of the "blocks world" program discussed in class. The basic task is to arrange blocks into a stack according to their size. Complete the form on the next page, based on the program printed below by listing the asserted facts (indicated by `==>`), the retracted facts (indicated by `<==`), the rules on the agenda, and the actions performed in each step. The program should finish in step 8. The number of lines in each step does not necessarily indicate the number of entries for that step (some lines may be empty). 40

```
;; Stacking Blocks Example
;; from Peter Jackson, "Introduction to Expert Systems", 3d ed., p. 89-90
;; modified by Franz J. Kurfess, 02-04-2003

;; Templates: a block has color, size, location

(deftemplate block
  (field color (type SYMBOL))
  (field size (type INTEGER))
  (field place (type SYMBOL) (default heap))
)

(deftemplate on
  (field upper (type SYMBOL))
  (field lower (type SYMBOL))
  (field place (type SYMBOL) (default heap))
)

(deftemplate goal
  (field task (type SYMBOL))
)

;; Initialization

(deffacts initial-blocks
  (block (color red) (size 10))
  (block (color yellow) (size 20))
  (block (color blue) (size 30))
)

;; Rules

(defrule begin
  (initial-fact)
  =>
  (assert (goal (task find)))
)
```

```
;; pick up the largest block on the heap
(defrule pick-up
  ?my-goal <- (goal (task find))
  ?my-block <- (block (size ?S1) (place heap))
  (not (block (color ?C2) (size ?S2&:(> ?S2 ?S1)) (place heap)))
  =>
  (modify ?my-block (place hand))
  (modify ?my-goal (task build))
   (printout t ?my-block " is in location hand "  crlf)
   (printout t "Task: build." crlf)
)

;; first block is the foundation of the tower
(defrule place-first
  ?my-goal <- (goal (task build))
  ?my-block <- (block (place hand))
  (not (block  (place tower)))
  =>
  (modify ?my-block (place tower))
  (modify ?my-goal (task find))
   (printout t ?my-block " is in location tower "  crlf)
   (printout t "Task: find." crlf)
)

;; subsequent blocks go on top
(defrule put-down
  ?my-goal <- (goal (task build))
  ?my-block <- (block (color ?C0) (place hand))
  (block (color ?C1) (place tower))
  (not (on (upper ?C2) (lower ?C1)  (place tower)))
  =>
  (modify ?my-block (place tower))
  (assert (on (upper ?C0) (lower ?C1)  (place tower)))
  (modify ?my-goal (task find))
   (printout t "adding " ?my-block " to location tower "  crlf)
   (printout t "Task: find." crlf)
)

;; stop, all blocks on tower
(defrule stop
  ?my-goal <- (goal (task find))
  (not (block  (place heap)))
  =>
  (retract ?my-goal)
   (printout t "Finished!" crlf)
)
```

Subtotal Task 3: 40

**Total Points: 100**