

Homework 2

Michael Clausen

1. a) yes b) no c) no

2. a) $\exists x \exists y (P(x) \wedge P(y))$

b) $\forall x P(x)$

c) $\neg \exists x P(x)$

d) $\neg \forall x P(x)$

e) $\exists x (\neg P(x))$

f) $\forall x P(x) \rightarrow S(x)$

g) $\exists x \exists y (P(x) \wedge P(y) \wedge S(x) \wedge S(y))$

h) $\neg \exists x (P(x) \wedge S(x))$

i) $\exists x \exists y (P(x) \wedge P(y) \wedge \neg S(x) \wedge \neg S(y))$

3. a) Fail

b) $\text{Mary} \Rightarrow x, \text{John} \Rightarrow y$

c) $\text{Mary} \Rightarrow y, z \Leftarrow \text{John}$

d) $x \Leftarrow y, z \Leftarrow \text{John}$

e) fail

f) $\text{Parent}(\overset{y}{\underset{\uparrow}{x}}, \overset{y}{\underset{\uparrow}{x}})$
 $\text{Parent}(y, \text{John})$

$x \Leftarrow y, y \Leftarrow \text{John}$

g) $\text{Parent}(\overset{y}{\underset{\uparrow}{x}}, y)$
 $\text{Parent}(y, \text{Father}(y))$
 $x \Leftarrow y, \quad \text{Fail}$

h) $x \Leftarrow y, z \Leftarrow \text{Father}(y)$

4. a)

Rule:

```
(defrule parents
```

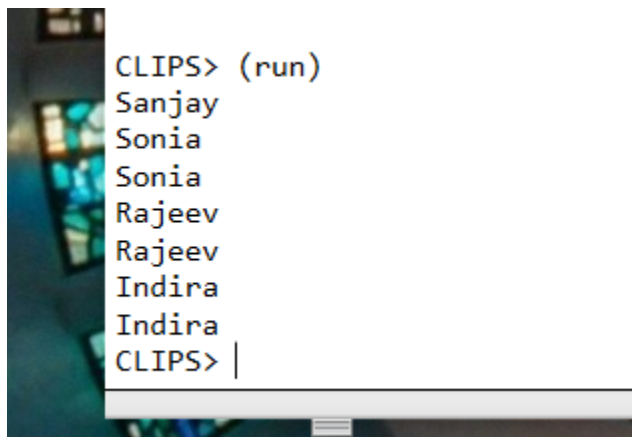
```
  (parent ?x ?y)
```

=>

```
  (printout t ?y crlf)
```

```
)
```

Output:



```
CLIPS> (run)
Sanjay
Sonia
Sonia
Rajeev
Rajeev
Indira
Indira
CLIPS> |
```

b) Rules:

```
(defrule cousins
```

```
  (sibling ?p1 ?p2)
```

```
  (parent ?cousin1 ?p1)
```

```
  (parent ?cousin2 ?p2)
```

=>

```
  (assert (cousin ?cousin1 ?cousin2))
```

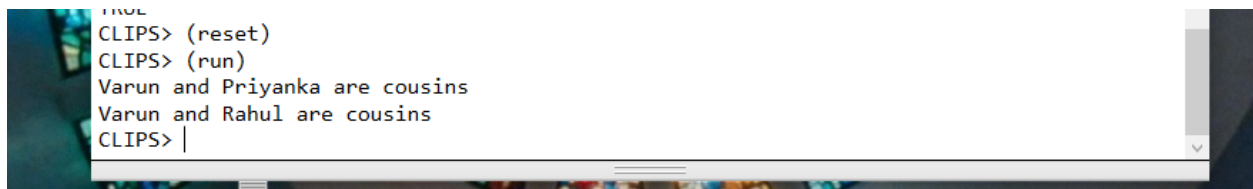
```
)
```

```

(defrule print
  (cousin ?cousin1 ?cousin2)
=>
  (printout t ?cousin1 " and " ?cousin2 " are cousins " crlf)
)

```

Output:



```

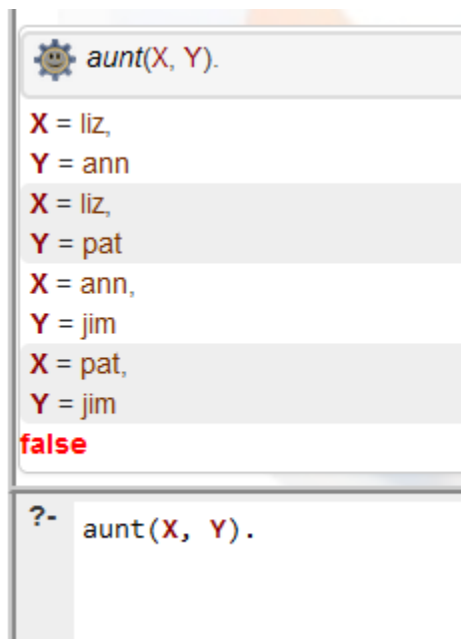
CLIPS> (reset)
CLIPS> (run)
Varun and Priyanka are cousins
Varun and Rahul are cousins
CLIPS>

```

5. a) Rule:

aunt(X, Y):- parent(Z, X), parent(Z, W), parent(W, Y), female(X).

Output:



```

aunt(X, Y).
X = liz,
Y = ann
X = liz,
Y = pat
X = ann,
Y = jim
X = pat,
Y = jim
false
?- aunt(X, Y).

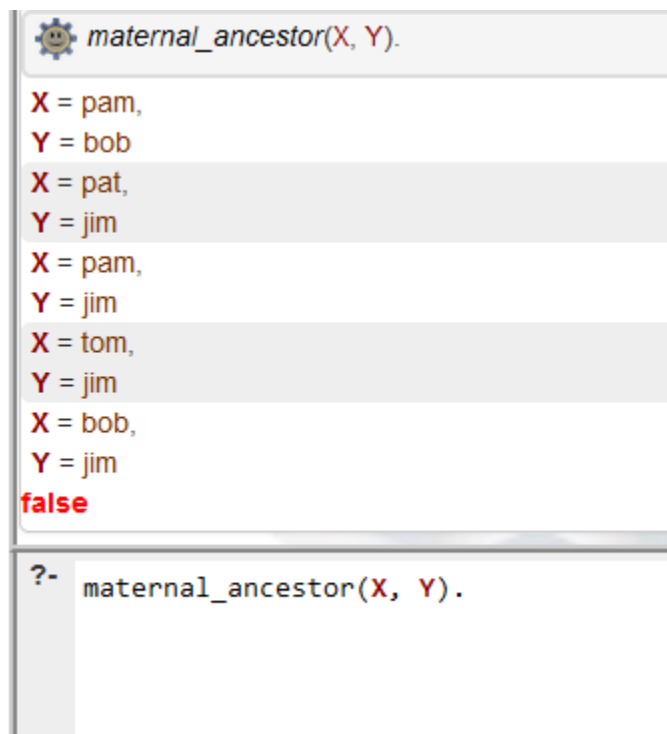
```

b) Rules:

maternal_ancestor(X, Z):- parent(X, Z), female(X).

maternal_ancestor(X, Z):- parent(X, Y), maternal_ancestor(Y, Z).

Output:



The image shows a Prolog interpreter window with a gear icon in the title bar. The window contains the following text:

```
maternal_ancestor(X, Y).  
X = pam,  
Y = bob  
X = pat,  
Y = jim  
X = pam,  
Y = jim  
X = tom,  
Y = jim  
X = bob,  
Y = jim  
false  
?- maternal_ancestor(X, Y).
```

6. a) T_{131} - Student has taken CPSC131

T_{338} - Student has taken Math338

T_{335} - Student has taken CPSC335

T_{375} - Student has taken CPSC375.

T_{481} - Student has taken CPSC481.

T_{483} - Student has taken CPSC483.

C_{375} - Student can take CPSC375.

C_{481} - Student can take CPSC481.

C_{483} - Student can take CPSC483.

C_{131} - Student can take CPSC131.

C_{338} - Student can take Math338.

C_{335} - Student can take CPSC335.

$$b) 1. (T_{131} \wedge T_{338}) \rightarrow C_{375}$$

$$2. (T_{335} \wedge T_{338}) \rightarrow C_{481}$$

$$3. (T_{375}) \rightarrow C_{483}$$

$$4. T_{375} \rightarrow (T_{131} \wedge T_{338})$$

$$T_{481} \rightarrow (T_{335} \wedge T_{338})$$

$$T_{483} \rightarrow T_{375}$$

$$5. T_{131} \rightarrow \neg C_{131}$$

$$T_{338} \rightarrow \neg C_{338}$$

$$T_{335} \rightarrow \neg C_{335}$$

$$T_{375} \rightarrow \neg C_{375}$$

$$T_{481} \rightarrow \neg C_{481}$$

$$T_{483} \rightarrow \neg C_{483}$$

c) i. Query: $T_{481} \rightarrow C_{375}$
Invalid

ii. $T_{481} \rightarrow C_{481}$
Invalid

iii. $T_{481} \rightarrow C_{483}$
Invalid

iv. $T_{481} \rightarrow T_{375}$
Invalid

d) i. Query: $(T_{131} \wedge T_{481}) \rightarrow C_{375}$
Valid

ii. $(T_{131} \wedge T_{481}) \rightarrow C_{481}$
Invalid

iii. $(T_{131} \wedge T_{481}) \rightarrow C_{483}$
Invalid

iv. $(T_{131} \wedge T_{481}) \rightarrow T_{375}$
Invalid

7. a) $\text{Taken}(x, y)$ - Student x has taken the course y .

$\text{Can-Take}(x, y)$ - Student x can take course y

$\text{Prereq}(x, y)$ - Course x is a prerequisite of course y

b) $\text{Prereq}(\text{CPSC131}, \text{CPSC375}) \wedge$
 $\text{Prereq}(\text{Math338}, \text{CPSC375})$

$\text{Prereq}(\text{CPSC335}, \text{CPSC481}) \wedge$
 $\text{Prereq}(\text{Math338}, \text{CPSC481})$

$\text{Prereq}(\text{CPSC375}, \text{CPSC483})$

$\forall x, y, z. (\text{Prereq}(x, y) \wedge \text{Taken}(y, z) \wedge$
 $\text{Prereq}(y, z)) \rightarrow \text{Can-Take}(x, z)$

$\forall x, y, z. (\text{Prereq}(x, y) \wedge \text{Prereq}(y, z) \wedge$
 $\text{Prereq}(x, z)) \rightarrow \text{Can-Take}(x, z)$

$\forall x, y, z. (\text{Prereq}(x, y) \wedge \text{Prereq}(y, z) \wedge$
 $\text{Prereq}(x, z)) \rightarrow \text{Can-Take}(x, z)$

$$\forall x \forall y (\neg \exists z (\text{Prereq}(z, y) \wedge \neg \text{Taken}(x, z)) \rightarrow \text{Can-Take}(x, y))$$

$$\forall x \forall y \text{ Taken}(x, y) \rightarrow (\neg \exists z (\text{Prereq}(z, y) \wedge \neg \text{Taken}(x, z)))$$

$$\forall x \forall y \text{ Taken}(x, y) \rightarrow \neg \text{Can-Take}(x, y)$$

c) i. Query: $\text{Taken}(x, \text{CPSC 481}) \rightarrow \text{Can-Take}(x, \text{CPSC 375})$

Invalid

ii. $\text{Taken}(x, \text{CPSC 481}) \rightarrow \text{Can-Take}(x, \text{CPSC 481})$

Invalid

iii. $\text{Taken}(x, \text{CPSC 481}) \rightarrow \text{Can-Take}(x, \text{CPSC 483})$

Invalid

iv. $\text{Taken}(x, \text{CPSC 481}) \rightarrow \text{Can-Take}(x, \text{CPSC 375})$

Invalid

d) Given: $\text{Taken}(x, \text{CPSC } 131) \wedge \text{Taken}(x, \text{CPSC } 481)$

i. Query: $\text{Can-Take}(x, \text{CPSC } 375)$

Valid

ii. $\text{Can-Take}(x, \text{CPSC } 481)$

Invalid

iii. $\text{Can-Take}(x, \text{CPSC } 483)$

Invalid

iv. $\text{Can-Take}(x, \text{CPSC } 375)$

Invalid

8. a)

```
(deffacts courses "some courses and their prerequisites"
  (course CPSC131)
  (course CPSC335)
  (course CPSC375)
  (course CPSC481)
  (course CPSC483)
  (course MATH338)
  (prereq CPSC375 CPSC131) ; CPSC375 has CPSC131 as a prerequisite
  (prereq CPSC375 MATH338)
  (prereq CPSC481 CPSC335)
  (prereq CPSC481 MATH338)
  (prereq CPSC483 CPSC375)
)
```

```
(defrule has_taken_course
  (course ?course)
  (prereq ?course ?prq)
  (taken ?course)
=>
  (assert (taken ?prq))
)
```

```
(defrule can_take_course
  (course ?course)
  (not (and (prereq ?course ?prq)
            (not (taken ?prq))))
  (not (taken ?course))
=>
  (assert (can_take ?course))
)
```

```
(defrule print_taken
  (taken ?course)
=>
  (printout t "Student HAS TAKEN " ?course crlf)
)
```

```
(defrule print_can_take
  (can_take ?course)
=>
  (printout t "Student CAN take " ?course crlf)
)
```

b) No additional changes would be needed. The program can be applied to any course with any number of prerequisites. This is because checks that there are not any prerequisites that the student has not met for a given course.

9. a)

```
course(cpsc131).
course(cpsc335).
course(cpsc375).
course(cpsc481).
course(cpsc483).
course(math338).
prereq(cpsc375, cpsc131). % cpsc375 has cpsc131 as a prerequisite
prereq(cpsc375, math338).
prereq(cpsc481, cpsc335).
prereq(cpsc481, math338).
prereq(cpsc483, cpsc375).
```

```
has_taken_course(Y) :- course(X), prereq(X, Y), taken(X).
has_taken_course(Y) :- taken(Y).
```

```
can_take_course(X) :- course(X), not(func(X)), not(taken(X)).
```

```
func(X) :- prereq(X, Y), not(taken(Y)).
```

b)

No additional changes would be needed. The program can be applied to any course with any number of prerequisites. This is because checks that there are not any prerequisites that the student has not met for a given course.