Programming Languages

HW5

Melih Yabaş
161044072

```
legs(X,2) :- mammal(X), arms(X,2).
legs(X,4) :- mammal(X), arms(X,0).
mammal(horse).
arms(horse,0).
?- legs(horse,4).
```

In prolog there are some predicates, facts and queries. In this program, given a common lisp list we will check an operation for the queires.

```
(
        ( (“legs” (“X” 2)) ( (“mammal” (“X”)) (“arms” (“X” 2)) ) )
        ( (“legs” (“X” 4)) ( (“mammal” (“X”)) (“arms” (“X” 0)) ) )
        ( (“mammal” (“horse”)) () )
        ( (“arms” (“horse” 0)) () )
        ( () (“legs” (“horse” 4)) )
)
```

This list generated by a parser. We need to keep it in an usual way.
If we read this file line by line, we can keep them in different lists. These lists are facts, predicates and queries. This separation operation could be done using specific characters in those. (For example a fact contains “())” characters)

```
(defun getSubstring (needle haystack &key (test 'char=))
  (search (string needle)
          (string haystack)
          :test test))
```

getSubstring function gives the index of a character in a string.
When we check whether a query is true or not, we should compare the strings.

This is a query: legs(horse,4).
This is a predicate: legs(X,4) :- mammal(X), arms(X,0).

We should put “horse” where there is an X in the predicate.
It takes a string and a word as parameters. Puts this word into appropriate position.

```
(defun changeXIntoString (str word)

    (setf newStr nil)

    (loop for i from 0 to (- (length str) 1)
        do()
        (if (and (>= (char-code (char str i)) 65) (<= (char-code (char str i )) 90))
            (progn

                (setf newStr (concatenate 'string (subseq str 0 i) word))
                (loop while (or (and (>= (char-code (char str i)) 65) (<= (char-code (char str i )) 90))
                    (and (>= (char-code (char str i)) 97) (<= (char-code (char str i )) 122)))
                    do(setf i (+ i 1))

                )
                (setf newStr (concatenate 'string newStr (subseq str i)))
            )
        )
    )

    (if (eq nil (getSubstring "))" newStr))
        (setf newStr (concatenate 'string newStr ")"))
    )
    newStr
)
```

```
(defun getQueryContent (query)

    (setf content (subseq query 1))

    (setf content (subseq content ( + (getSubstring "(" content) 1)(- (getSubstring ")" content) 1) ) )

    content
)
```

GetQueryContent gives the content of a query.

Check function first checks whether given query is a fact or not.
If it is not, it is a predicate, so we should check right side of a predicate.

```
   (setf bool 0)
   (loop for j from 0 to (- (length facts2) 1)
           do()
           (if (string= query (nth j facts2))
               (progn
                   (return-from check T)
               )
           )
   )

   (setf queryContent (getQueryContent query))
   (loop for i from 0 to (- (length predicates2) 1)
       do()
       (if (string= query (changeXIntoString (nth 0 (nth i predicates2)) content))
           (progn
               (loop for j from 0 to ( - (length (nth 1 (nth i predicates2))) 1)
                   do()
                   (loop for k from 0 to ( - (length facts2) 1)
                       do()
                       (if (string= (nth k facts2) (changeXIntoString (nth j (nth 1 (nth i predicates2))) queryContent))
                           (progn
                               (setq bool (+ 1 bool))
                           )
                       )
                   )
               )
           )
       )
   )
```

How to Run This Program?

In the Directory;

>clisp hw5.lisp

Output