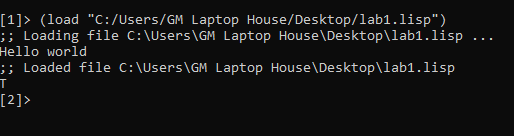
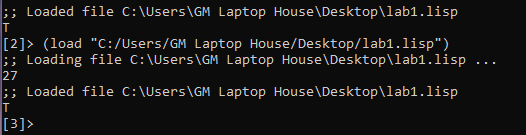
**LAB 1 Artificial Intelligence**

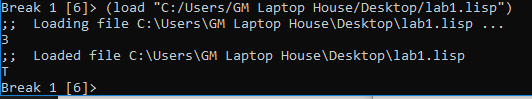
1. Write and practice **all** the exercise programs included in the manual.
2. (write-line "Hello World")



2 .(write(+ 7 9 11))



3.(write(/ (\* 2 (+ 5 4)) 6))



4. (setq x 10)

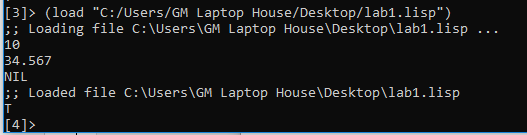
(setq y 34.567)

(setq ch nil)

(print x)

(print y)

(print ch)

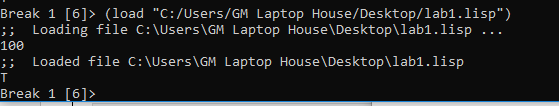
****

5. (defun sum(n1 n2 n3 n4)

( + n1 n2 n3 n4)

)

(write(sum 10 20 30 40))

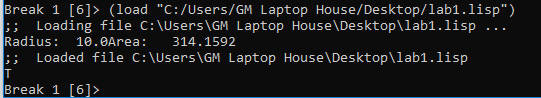
****

6. "Calculates area of a circle with given radius"

(format t "Radius: ~5f" rad)

(format t "Area: ~10f" (\* 3.141592 rad rad)))

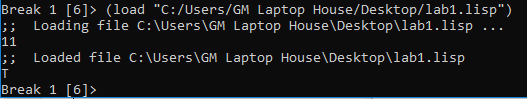
(area-circle 10)

****

7. (setf x 0)

(loop (setf x (+ x 1)) (when (> x

10) (return x))) (write x)

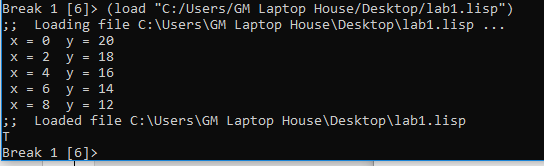


8. (do ((x 0 (+ 2 x))

(y 20 ( - y 2)))

((= x y)(- x y))

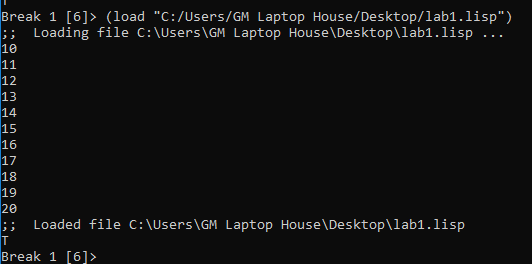
(format t "~% x = ~d y = ~d" x y))



9. (loop for a from 10 to 20

do (print a)

)

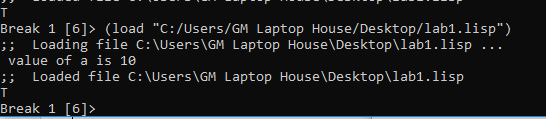
****

10. (setq a 10)

(if (> a 20)

(format t "~% a is less than 20"))

(format t "~% value of a is ~d " a)

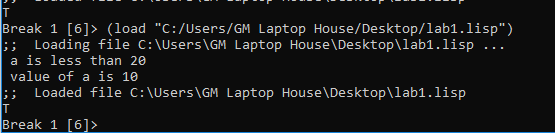
****

11. (setq a 10)

(if (> a 20)

then (format t "~% a is less than 20"))

(format t "~% value of a is ~d " a)

****

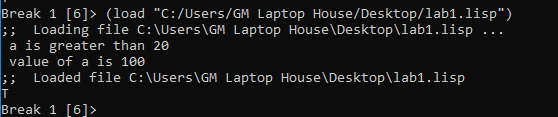
12. (setq a 100)

(if (> a 20)

(format t "~% a is greater than 20")

(format t "~% a is less than 20"))

(format t "~% value of a is ~d " a)

****

13.(setq day 4)

(case day

(1 (format t "~% Monday"))

(2 (format t "~% Tuesday"))

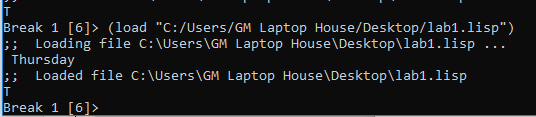
(3 (format t "~% Wednesday"))

(4 (format t "~% Thursday"))

(5 (format t "~% Friday"))

(6 (format t "~% Saturday"))

(7 (format t "~% Sunday")))

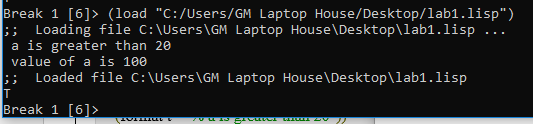
****

14 . (setq a 100)

(when (> a 20)

(format t "~% a is greater than 20"))

(format t "~% value of a is ~d " a)

****

1. Write at least three meaningful programs in CLISP using following CLISP macros respectively. Write individual programs.

* defun
* defstructure
* defstruct
* defmethod

**defun :**

(defun area-circle(rad)

"Calculates area of a circle with given radius"

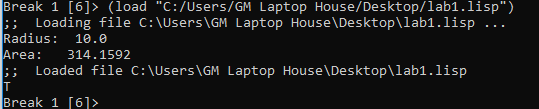
(terpri)

(format t "Radius: ~5f" rad)

(format t "~%Area: ~10f" (\* 3.141592 rad rad))

)

(area-circle 10)

****

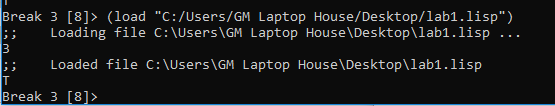
* **defstruct**

(defstruct point x y z) ; define structure

(defvar my-point) ; define var

(setf my-point (make-point :x 3 :y 4 :z 12)) ; set value of slots of var

(point-x my-point) ; acces to slots



* **defstructure**
* [Live Demo](http://tpcg.io/hlyF3w)

(defstruct book

title

author

subject

book-id

)

( setq book1 (make-book :title "C Programming"

:author "Nuha Ali"

:subject "C-Programming Tutorial"

:book-id "478")

)

( setq book2 (make-book :title "Telecom Billing"

:author "Zara Ali"

:subject "C-Programming Tutorial"

:book-id "501")

)

(write book1)

(terpri)

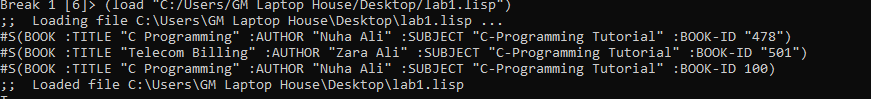
(write book2)

(setq book3( copy-book book1))

(setf (book-book-id book3) 100)

(terpri)

(write book3)

****

* **Defmethod**

(defclass box ()

((length :accessor box-length)

(breadth :accessor box-breadth)

(height :accessor box-height)

)

)

(setf item (make-instance 'box))

(setf (box-length item) 10)

(setf (box-breadth item) 10)

(setf (box-height item) 5)

(format t "Length of the Box is ~d~%" (box-length item))

(format t "Breadth of the Box is ~d~%" (box-breadth item))

(format t "Height of the Box is ~d~%" (box-height item))

