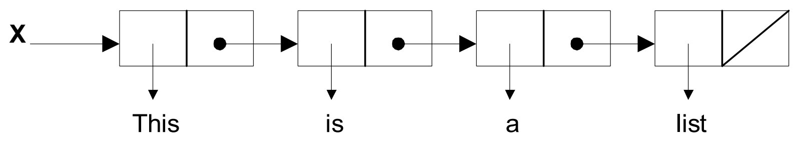
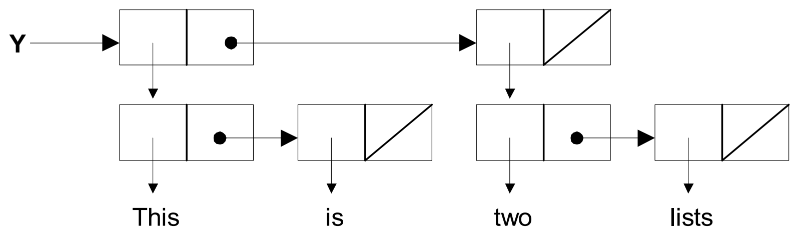
LISP
Data Structures

(setq X `(This is a list))

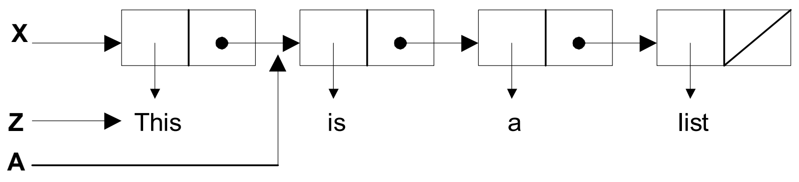


(setq Y `((This is)(two list)))



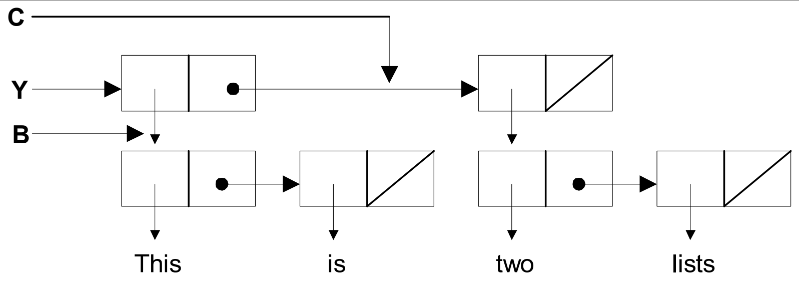
(setq Z (car X))

(setq A (cdr X))

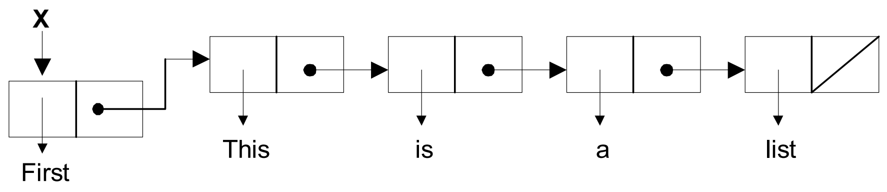


(setq B (car Y))

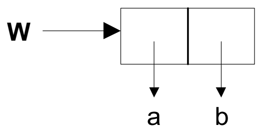
(setq C (cdr Y))



(setq X (cons 'First X))



(setq W (cons `a `b))



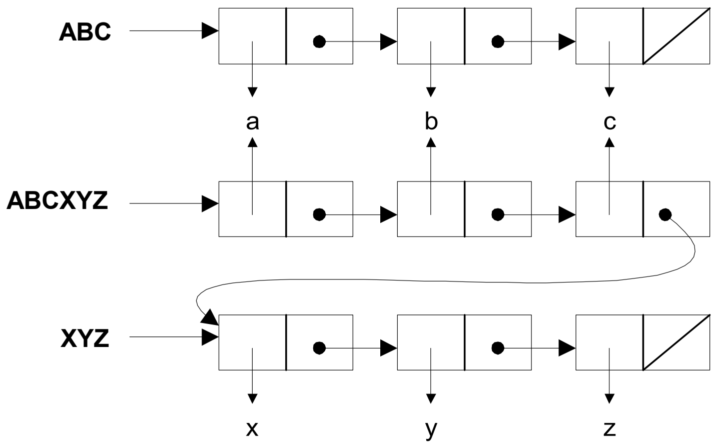
a.b

(setq ABC '(a b c))

(setq XYZ '(x y z))

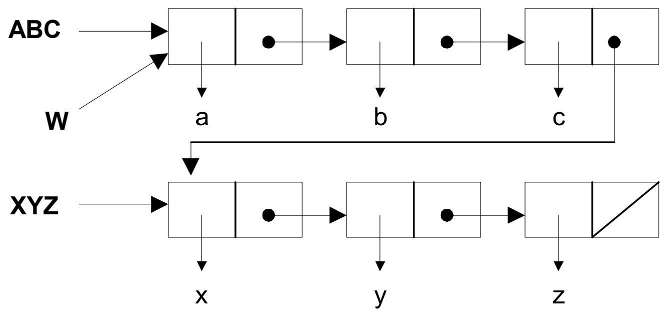
(setq ABCXYZ (APPEND ABC XYZ))

**APPEND does not change ABC or XYZ*



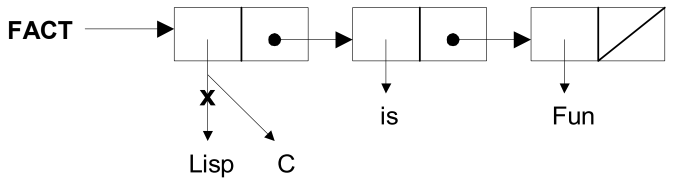
(setq W (NCONC ABC XYZ))

NCONC **will change the first list.*

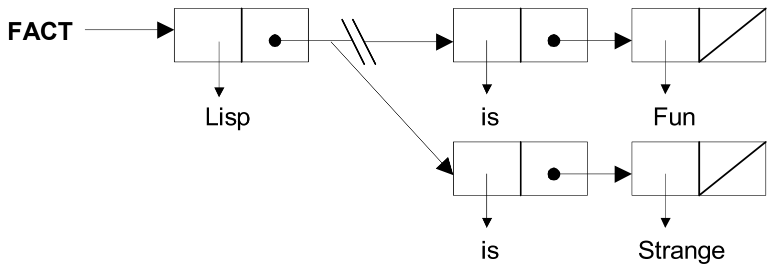


(setq FACT `(Lisp is Fun))

(RPLACA FACT 'C)



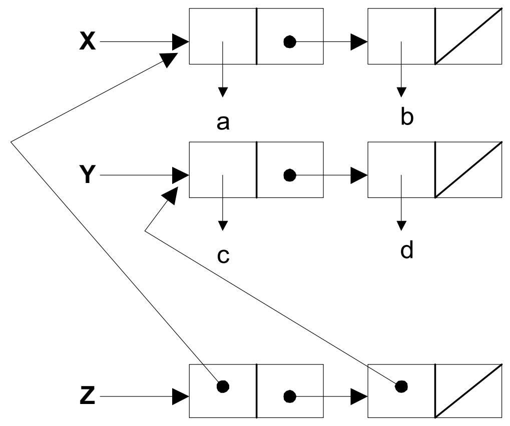
(RPLACD FACT `(is Strange))



(setq X `(a b))

(setq Y '(c d))

(setq Z (LIST X Y))



Examples:

(1)	(APPEND '(a b) '(c d))	(a b c d)
(2)	(LIST '(a b) '(c d))	((a b) (c d))
(3)	(CONS '(a b) '(c d))	((a b) c d)
(4)	(NCONC '(a b) '(c d))	(a b c d)

4 looks the same as 1 however:

(setq AB '(a b))	(a b)
(setq CD '(c d))	(c d)

AB	(a b)
CD	(c d)

(setq ABCD(NCONC AB CD))	(a b c d)

ABCD	(a b c d)
CD	(c d)
AB	(a b c d)