

	nsertion at middle
(2) Suscrtion at last	1
(3) If Avail = Null underflow	than write ("Stack was
return (Avail)	110000
© [Initialize a new node] New - Avail : Inble Stack)	© CInitialize a new nood New - avail (3) [Remove free node from available stack] Avail - avail - Link
(3) [Remare, free node from available	O FIs the list empty NULL
(3) [Remove, free node link Avail - Avail - link (4) [Initicalizes field of a new node] Info (New) - x Link (New) - Null	there (Neur)
(b) [Initializes field of Info (New) - 2 Null	new nude precede (FIRST)
	then LINK (NEW) Return (NEW) Return (NEW)
(3) [Is the list empty?] If first=NULL then Return (first)	(6) [Initialize Tirst Louise of new node)
(6 [Initialize search for the last node) save = first	(SAVE) INFO (LINK (SAVE)) > INFO (LINK (SAVE))
save - first	SAVE < LINK CSAVE) SAVE (SAVE) SAVE (SAVE)
(7) Search for and of list? Repeat while (Link (save) + NULL) Save - Link (save)	SAVE < LINK CSAVE) SAVE < LINK CSAVE) SAVE < LINK CSAVE) LINK CNEW) = LINK CSAVE) LINK (SAVE) < NEW LINK (SAVE) < NEW LINK (SAVE) < NEW LINK (SAVE) < NEW
save < Link (save)	(3) [Return first nude prointer] Return (FIRST)
(E) [Set link field of list nude of new?] Link Coave) - New	1 Delete the Open Element
(9) [Return tirst node]	
return (first)	11 Clarker 100
	return dearch for x)
	E) [Institution of First

(3) [Find x] Repeat through (3) while Repeat through (3) and link (3) are 1 to 1.0%
3 China & Straugh (5) and Link (5)
Repeat of X are
© [Update predecesseer marker] PRED = SAVE
m - 1 1 de spederesseer mais
6) Copacite process
@ EMOVE to next neede)
SAVE = Link (SAVE)
SAVE
6) [End of the 18st]
Tf SAVE # X not found') then write (Node not found') seturn
then write CHOICE
return
6) ra 11 kg
Delete X)
If X = First - Link (First) then First - Link (First) - Link (X)
then First < Link (PRED) < Link(X) else Link (PRED)
else ciroce
8 CFREF Deletal Node]
FREE (X)
FRECENT
→ Doubly Linked List.
1) Insertion at begin
2) Insertion at last
3) Inserticen at middle
a Deletion at begin
3 Deletion at last
_ 6 Deletion at middle
on the state of the state of
DInsertion from beginning
Θ
Avail = NULL
(2)
NEW - AVAIL

B) Avail - Next
Bre - New -> Bre - New
New - data < Value (x)
6 New → Next ← Start
3 Start → Pre < new
Sturt - Niew
© Exit
(2) Insertion at end.
Avail - NULL
New < Avall
Avail - Avail - Next
Dew → Next ← Null
. Sew > data ← value
© ptr ← start
While (ptr-) next! = NULL)
$ \begin{array}{c} \emptyset \\ \text{ptr} \leftarrow \text{ptr} \rightarrow \text{next} \\ (9) \end{array} $
ptr → next ← new
New -> pre - ptr

Middle
3 Insertion at Middle
1 Avail - NULL
\mathfrak{D} Ava°
3 Avail - Avail - Next
B) New -> data = value
3 - ctall
6 Report Step 4: ptr-data = Value
Ptr ← ptr → next
New → Next ← ptr → next.
(10) New → prev ← ptr
ptr-> next < new
(b) Deletion at begining
D ptr + start
start ← start → Next
Sturt -> prev < null
free (ptr)

	and the second s
Deletion at last	
Deletion	and the second s
ptn = start	- The second sec
(D) next	+ NUIL
Repeat step 3 pts - next	71300
A	
ptr = ptr - next	
D ptn → prev → next ← nu	I amount of the second
pla - prev - next	
free (ptr)	
@ Deletion at Middle	3
(2) date	a! = value
ptr = start Repeat step 3: ptr -> data ptr = ptr -> next	
(3)	
ptr = ptr	
temp < ptr - next	
(5) next - temp-s ne	xt
temp - next - prev - pt	tr
temp - next = p	
- Circular Linked List	
Deletion Insertion at last	d Bist
3 Inserticus in Ordered Links	
Deletion Insertion of lass (3) Insertion in Ordered Links (3) Delete Element	
1 L basining	and the second s
(3) Insertion at begining (1) [Create new empty mode]	
New < Node	mode and 15 link to the
(1) [Create new empty videos New & Node. (2) [Initialize fields of new into (New) < X	
(2) (Initialize flects) ← X	
1.50	0 11 0 0

then LINK (NEW) - NEW If First = NULL FIRSTE LASTE NEW else LINK (NEW) - FIRST LINK (LAST) ENEW FIRST - NEW Return 2) Insertion at last. @ [Create New Empty node]

New = Node

Of new node and its link to

B [Initialize fields of new node and its link to

Here 10 +7 the lest) If FIRST = NULL then LINK (NEW) < NEW FJRST = LAST = NEW else LINK (NEW) < FIRST LINK (LAST) NEW LAST - NEW Return 3 Insertion in Ordered Linked hist [Create New Empty Node] New - Node [Copy information content into new neede] ENFO (NEW) - X [Is linked list is ampty?] If FIRST = NULL then LINK (NEW) - NEW FIRST & LAST & NEW Return 1 [Does new node precedes all other nodes in list If INFO (NEW) & INFO (FIRST) then LINK (NEW) < FIRST LINK (LAST) ~ NEW FIRST -NEW Return

© [Initialize Temporary pointer] save ← First © [Search for Predecessor of new node] Repeat while save ≠ LAST and INFO (NEW) > INFO(LINK (SAVE)) Save ← Link (Save)
(6) [Search for Predecessor of new node] Repeat while save \$ LAST and INFO(NEW) > INFO(LINK(SAVE))
C C () ()
1000CL 2000
Save = link (Save)
Joseph D. Jacobs (A)
of Set link field of New node and its medicine
CINK (NEW) - CINK (Sate
LINK (Sour) < NEW
If some = last
then last < New
(8) [Finished]
Return
(a) Delete Element
1) [Is Empty List?]
7 L Linit = NVIC
then write ("Empty")
buturn
@ [Initialize Search for x]
Save - First
(3) [Find x] Repeat thrugh step 5 while Save + X and so
Repeat Might sig Save & Last
@ Cupolate Predecessor marker)
Pred = save
@ Move to Next Node
Save = Link (Sour)
@ [End of linked list]
$T = X \times X \times X$
then write ("Node not found")
return 2 Fold de 17
TH X = First
then First - Link (tirst)
link (Last) + First
else Link (Pred) = Link(x)

If x = Last

then LAST (-PRED

(8) [FREE Deleted Node)

free(X)