

Data Structures

# Double Linked List – delete\_at\_first

Team Emertxe



# Double Linked List



# Analysis `delete_at_first`



Analysis: Logic / Cases

Flowchart

Algorithm

Code

Analysis

A horizontal bar spanning the width of the slide, featuring a gradient from bright pink on the left to deep purple on the right. The bar ends in a double arrow pointing to the right, with the inner arrow in a lighter shade of purple and the outer arrow in a darker shade.

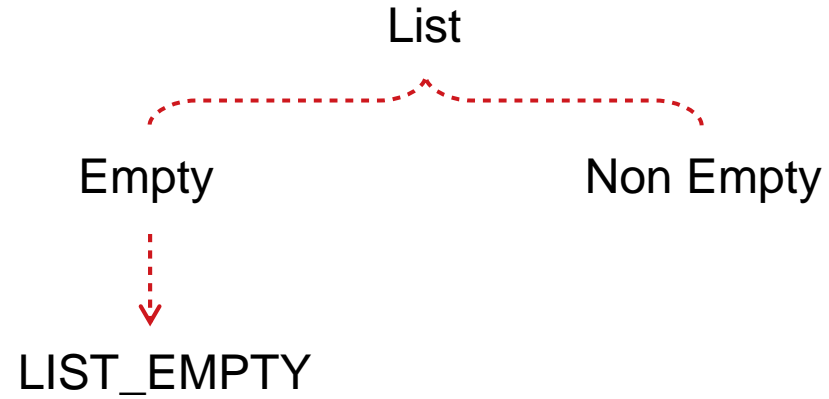
Cases :

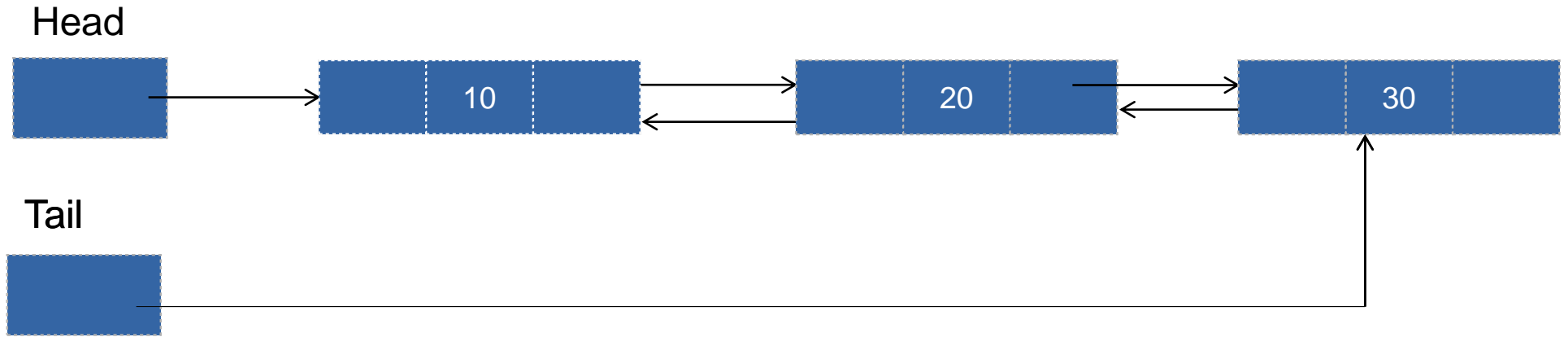
List

Cases :

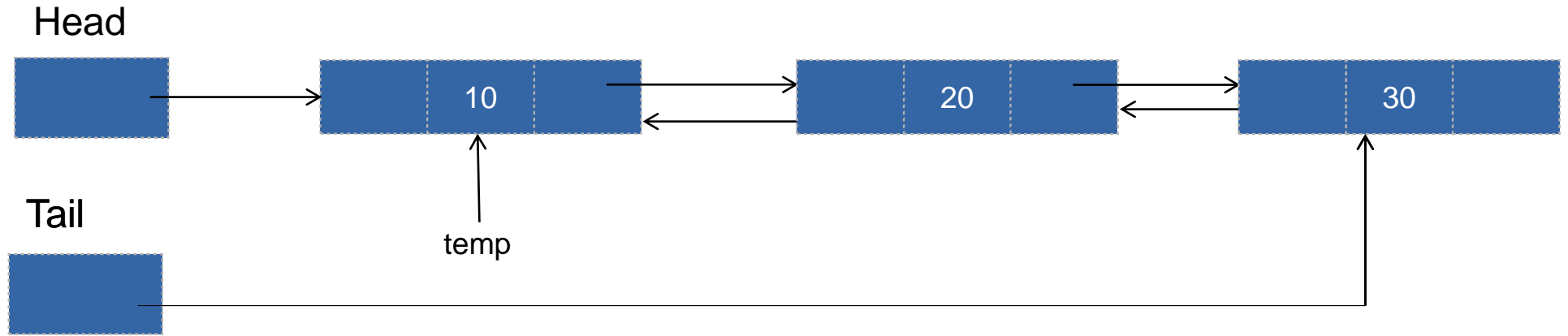


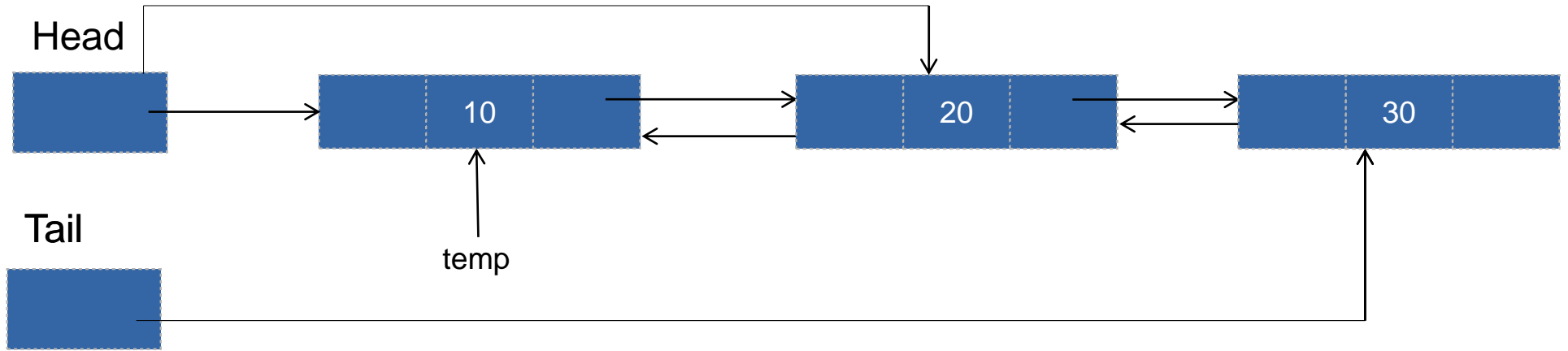
## Cases :

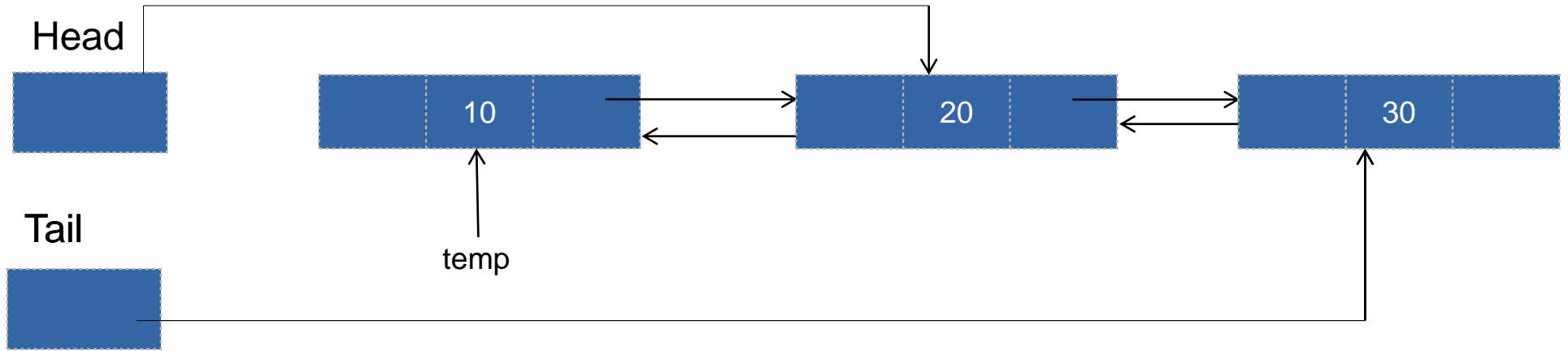












Flowchart

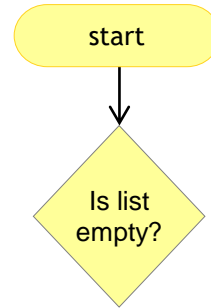


start

Head

# Flowchart

delete\_at\_first

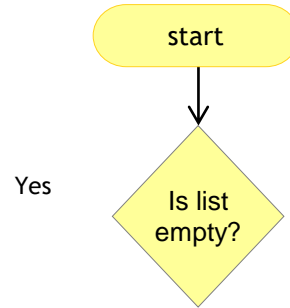


Head

NULL

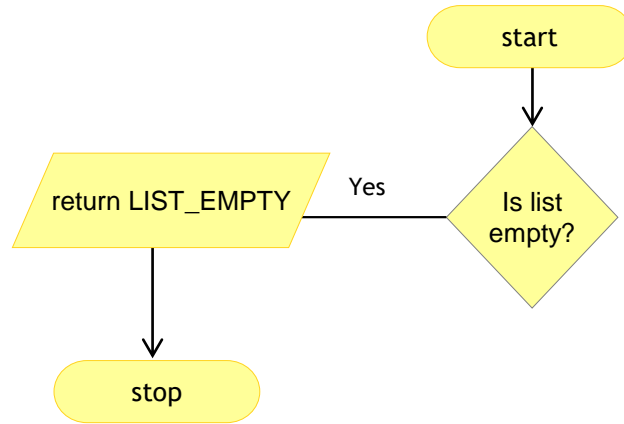
# Flowchart

delete\_at\_first



Head

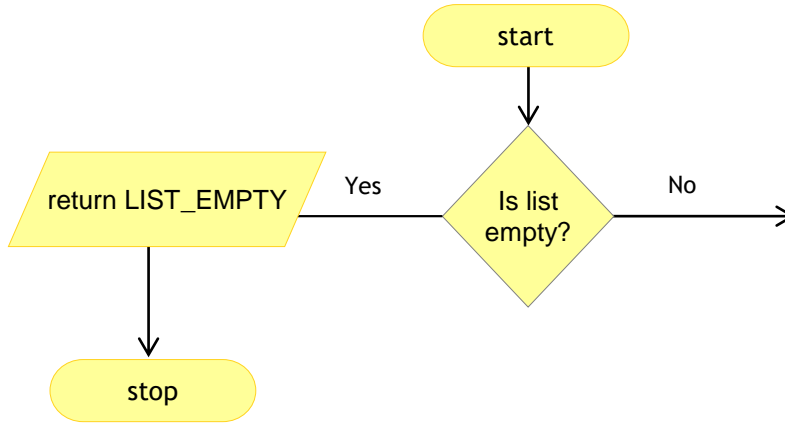
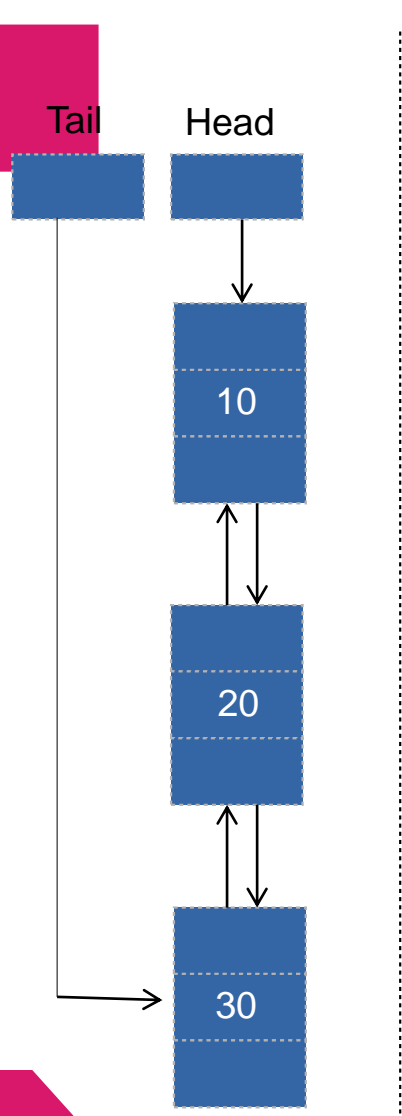
NULL



# Flowchart

delete\_at\_first



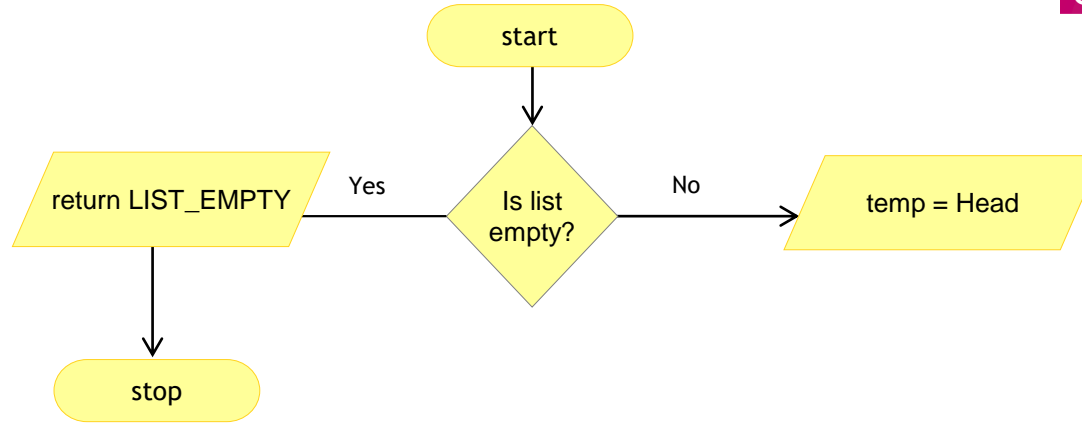
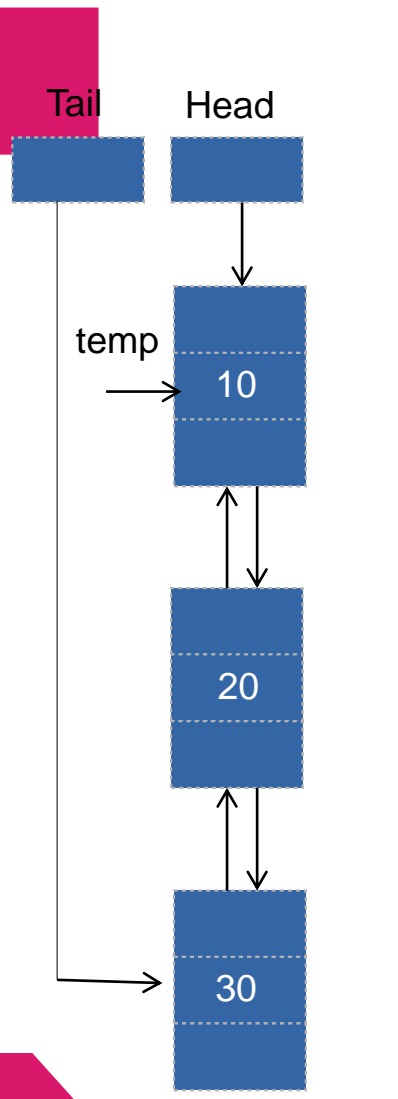


# Flowchart

delete\_at\_first

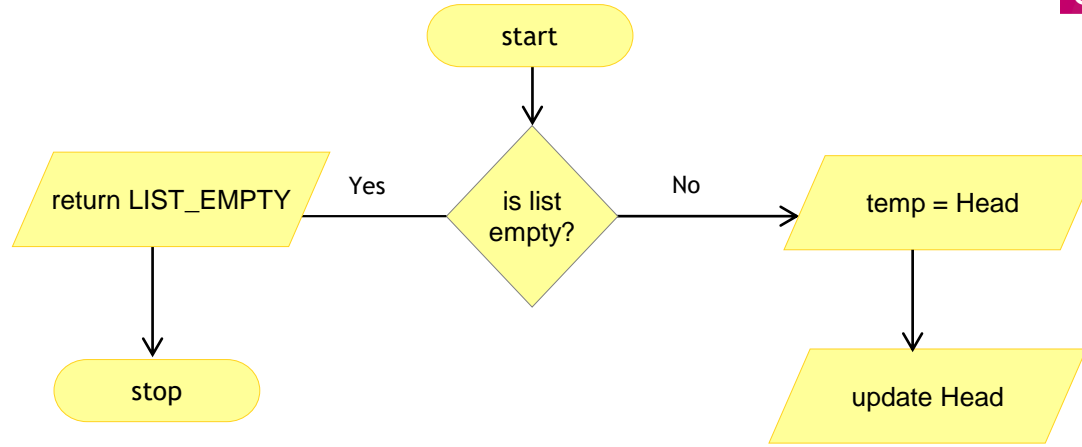
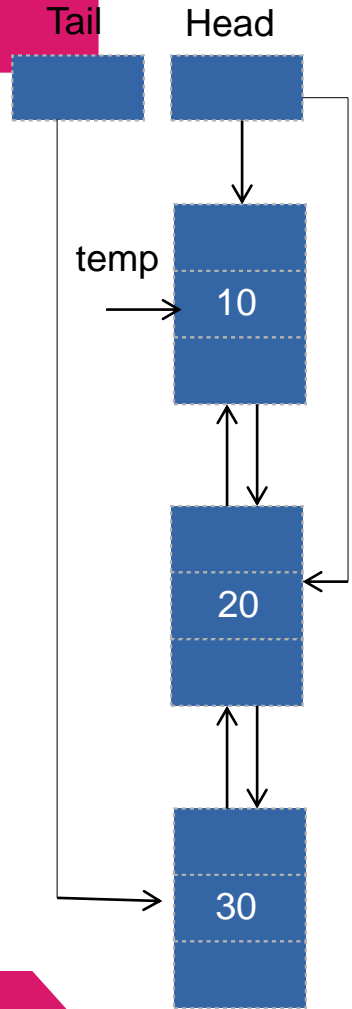
# Flowchart

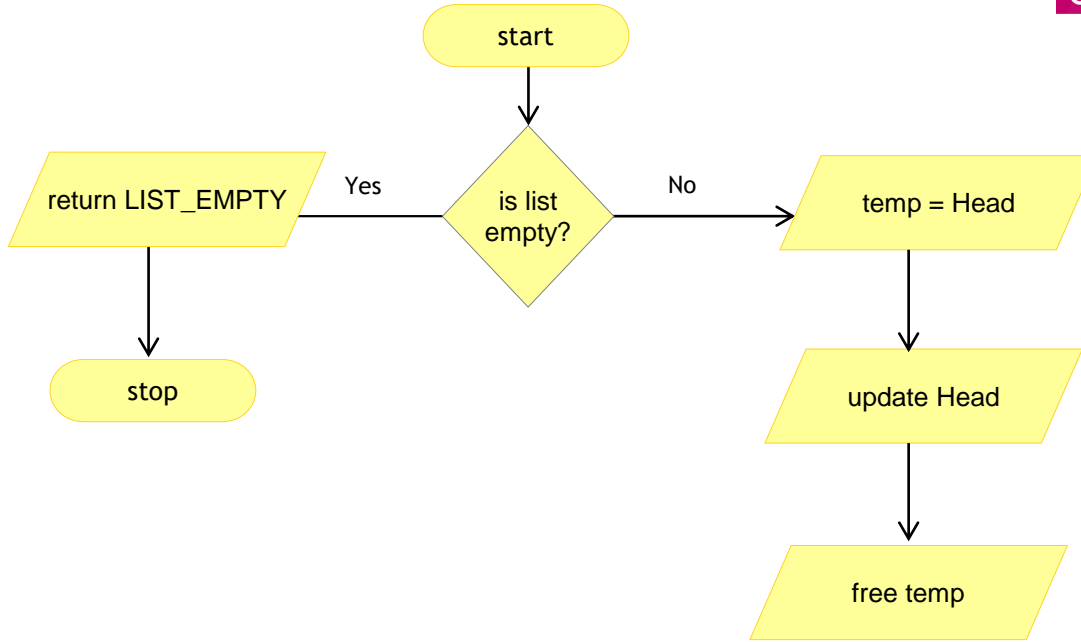
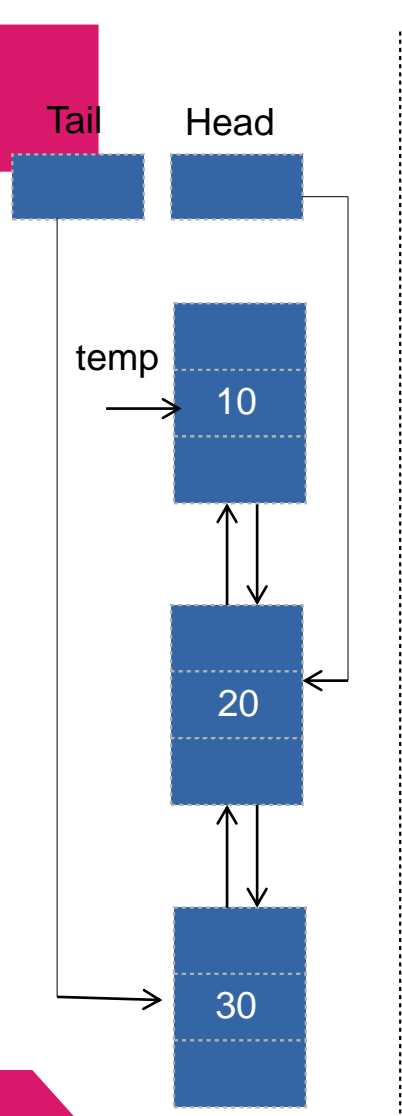
delete\_at\_first



# Flowchart

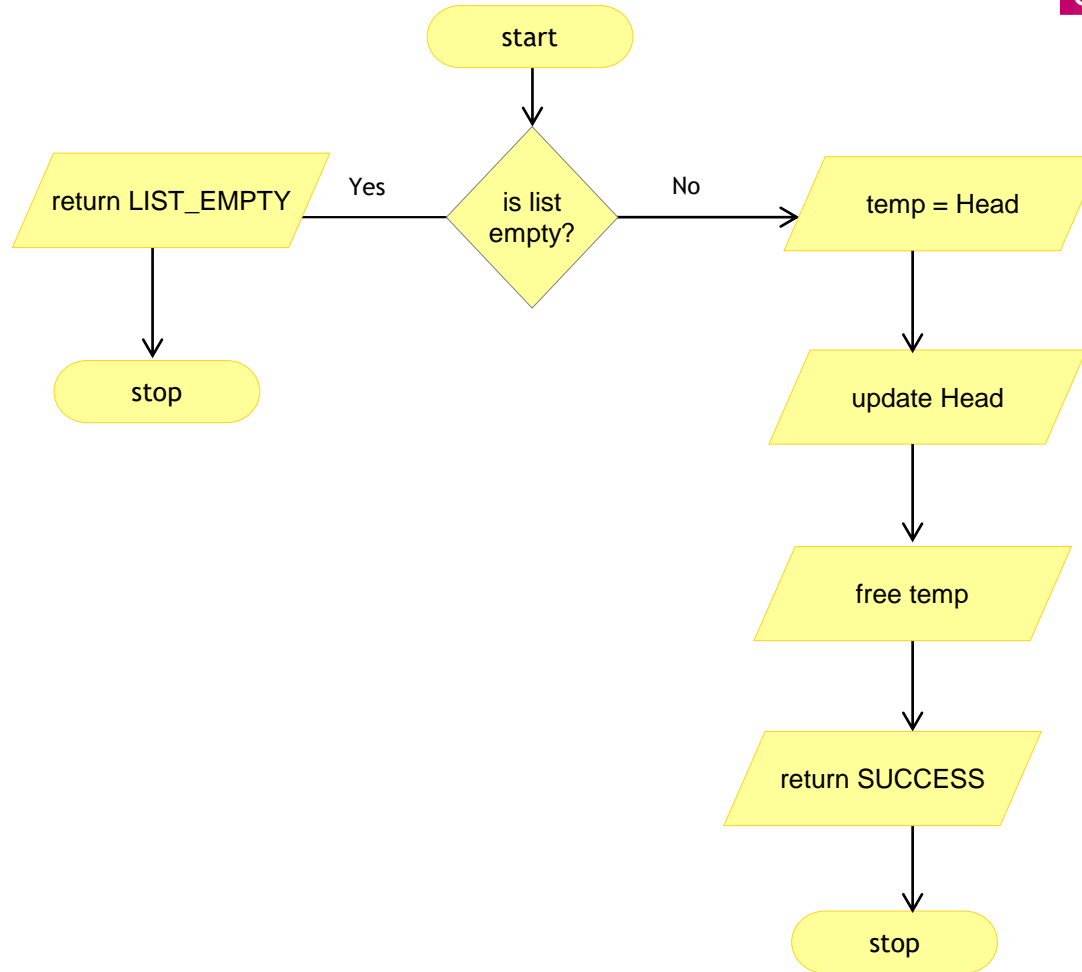
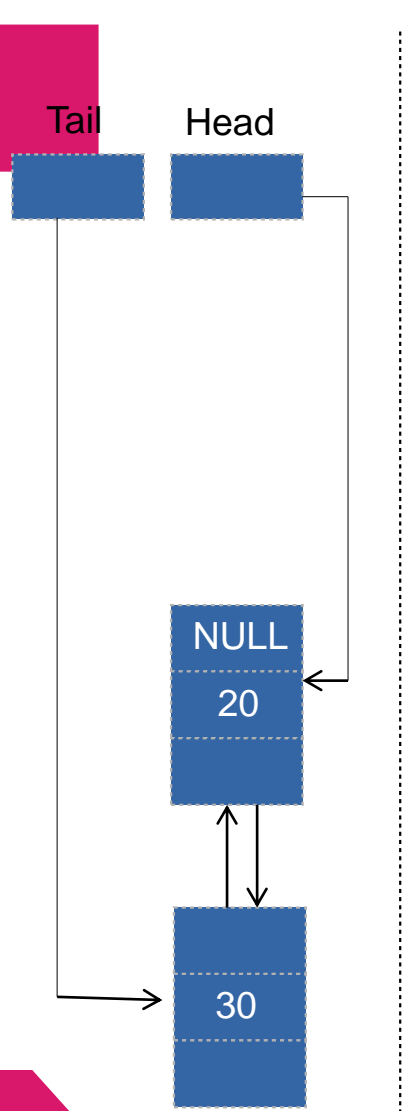
## delete\_at\_first





# Flowchart

delete\_at\_first



# Flowchart

delete\_at\_first

Algorithm

A decorative graphic at the bottom of the slide. It consists of a horizontal bar with a gradient from bright pink on the left to dark purple on the right. The bar ends in a double arrow pointing to the right, with the inner arrow being a lighter shade of pink and the outer arrow being a darker shade of purple.

## Algorithm : delete\_at\_first(Head)

if (Head = NULL)

Head

NULL

**Algorithm : delete\_at\_first(Head)**

if (Head = NULL)

return LIST\_EMPTY

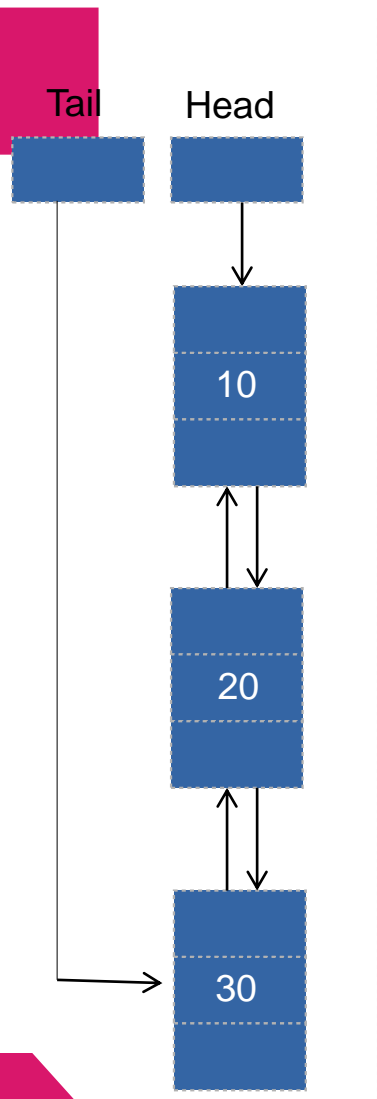
**Algorithm**  
delete\_at\_first



## Algorithm : delete\_at\_first(Head)

if (Head = NULL)

return LIST\_EMPTY



# Algorithm

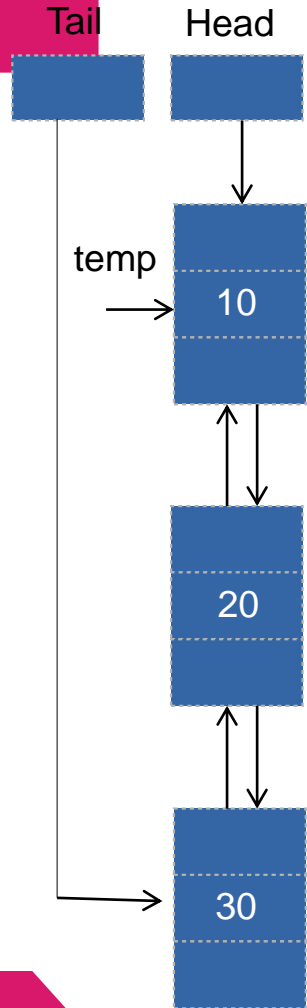
delete\_at\_first

## Algorithm : delete\_at\_first(Head)

if (Head = NULL)

return LIST\_EMPTY

temp = Head



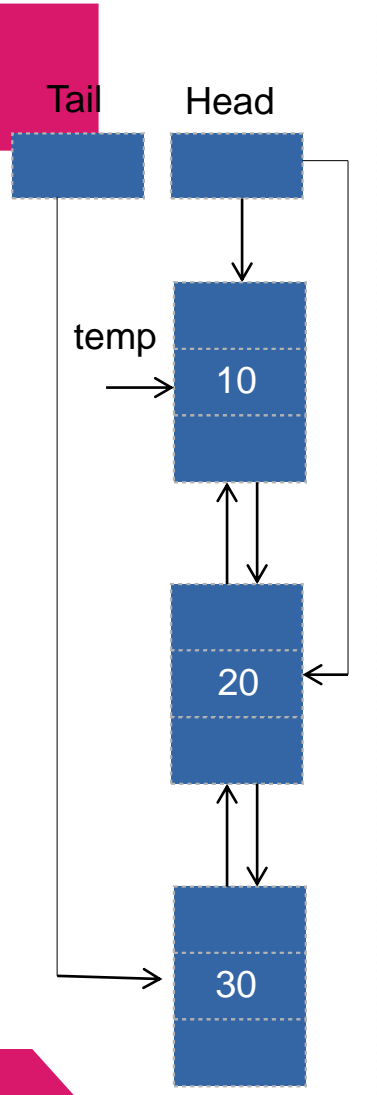
## Algorithm : delete\_at\_first(Head)

if (Head = NULL)

return LIST\_EMPTY

temp = Head

Head = temp → next



## Algorithm : delete\_at\_first(Head)

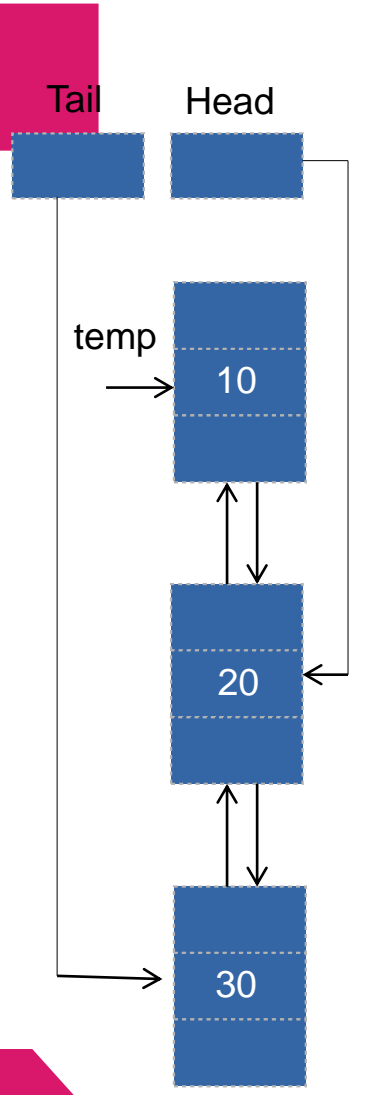
if (Head = NULL)

return LIST\_EMPTY

temp = Head

Head = temp → next

free temp



## Algorithm : delete\_at\_first(Head)

if (Head = NULL)

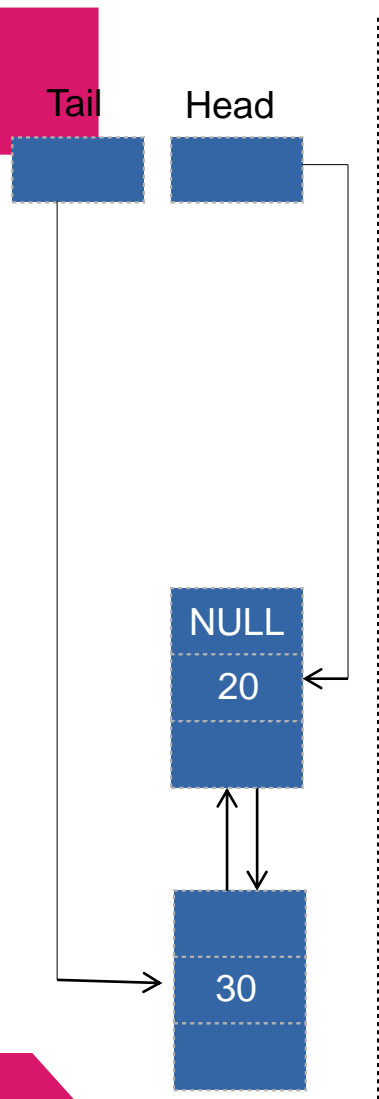
return LIST\_EMPTY

temp = Head

Head = temp → next

free temp

Head → prev = NULL



## Algorithm : delete\_at\_first(Head)

if (Head = NULL)

return LIST\_EMPTY

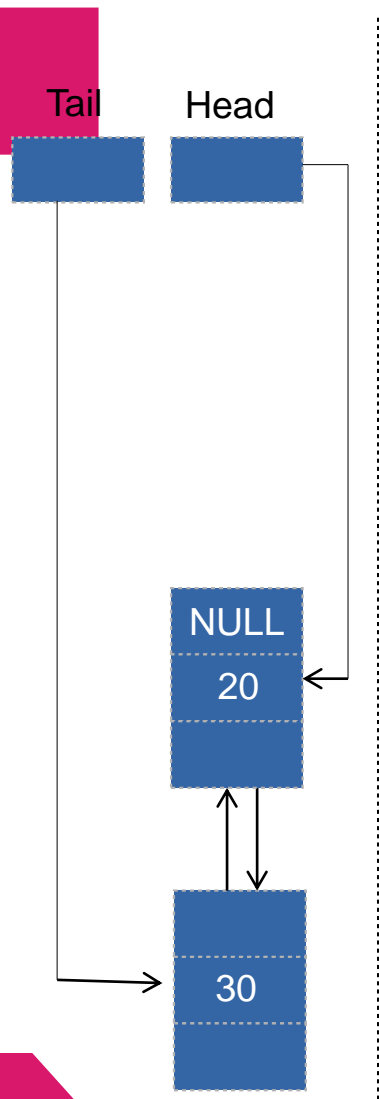
temp = Head

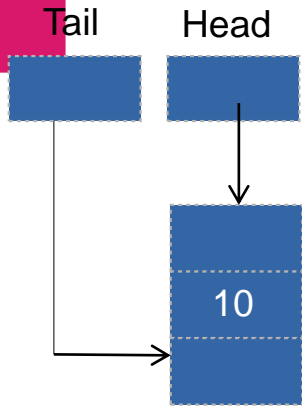
Head = temp → next

free temp

Head → prev = NULL

return SUCCESS





### Algorithm : delete\_at\_first(Head)

if (Head = NULL)

    return LIST\_EMPTY

temp = Head

Head = temp → next

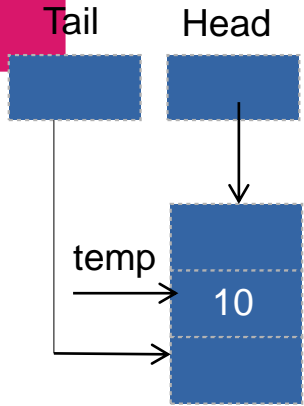
free temp

Head → prev = NULL

return SUCCESS

# Algorithm

delete\_at\_first



## Algorithm : delete\_at\_first(Head)

if (Head = NULL)

return LIST\_EMPTY

**temp = Head**

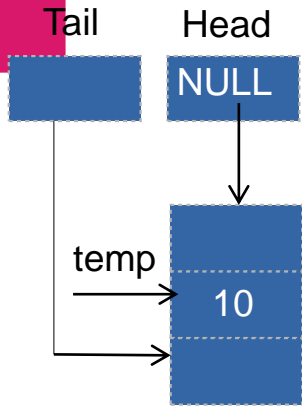
Head = temp → next

free temp

Head → prev = NULL

return SUCCESS





## Algorithm : delete\_at\_first(Head)

if (Head = NULL)

return LIST\_EMPTY

temp = Head

**Head = temp** → next

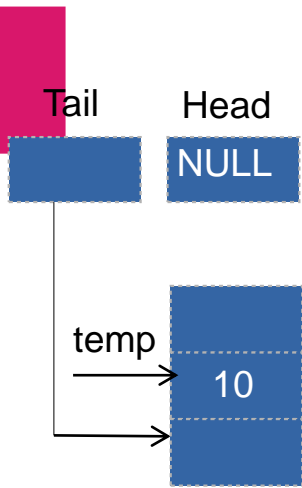
free temp

Head → prev = NULL

return SUCCESS

# Algorithm

delete\_at\_first



## Algorithm : delete\_at\_first(Head)

if (Head = NULL)

return LIST\_EMPTY

temp = Head

Head = temp → next

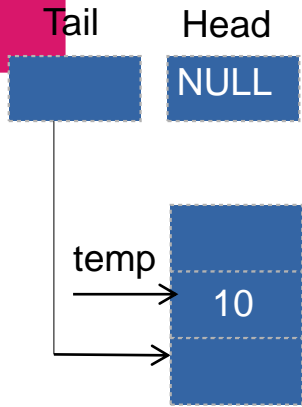
**free temp**

Head → prev = NULL

return SUCCESS

# Algorithm

delete\_at\_first



## Algorithm : delete\_at\_first(Head)

if ( Head = NULL)

return LIST\_EMPTY

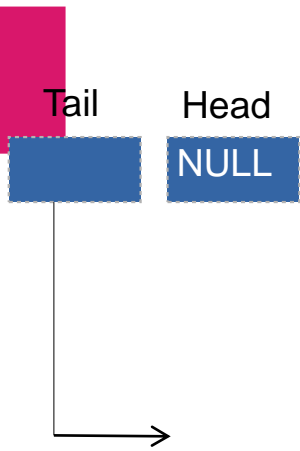
temp = Head

Head = temp → next

**free temp**

Head → prev = NULL

return SUCCESS



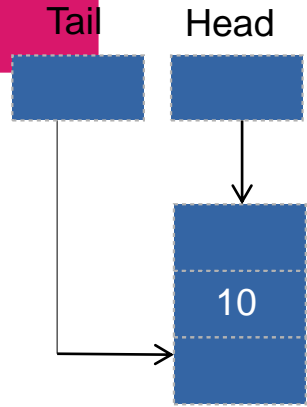
## Algorithm : delete\_at\_first(Head)

```
if ( Head = NULL)
    return LIST_EMPTY

temp = Head
Head = temp → next
free temp
Head → prev = NULL
return SUCCESS
```

## Algorithm

delete\_at\_first



## Algorithm : delete\_at\_first(Head,Tail)

if (Head = NULL)

    return LIST\_EMPTY

if (Head  $\longrightarrow$  next = NULL)

    free Head

    Head = Tail = NULL

    return SUCCESS

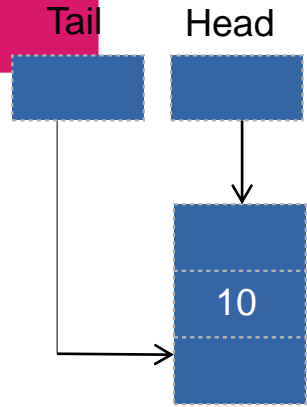
temp = Head

Head = temp  $\longrightarrow$  next

free temp

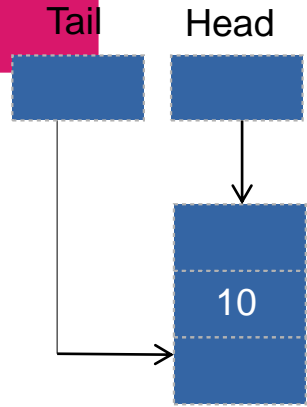
Head  $\longrightarrow$  prev = NULL

return SUCCESS



## Algorithm : delete\_at\_first(Head,Tail)

```
if ( Head = NULL)
    return LIST_EMPTY
if ( Head → next = NULL)
    free Head
    Head = Tail = NULL
    return SUCCESS
temp = Head
Head = temp → next
free temp
Head → prev = NULL
return SUCCESS
```



## Algorithm : delete\_at\_first(Head,Tail)

```
if ( Head = NULL)
    return LIST_EMPTYY
if ( Head → next = NULL)
    free Head
    Head = Tail = NULL
    return SUCCESS
temp = Head
Head = temp → next
free temp
Head → prev = NULL
return SUCCESS
```

Tail Head  
NULL NULL

### Algorithm : delete\_at\_first(Head,Tail)

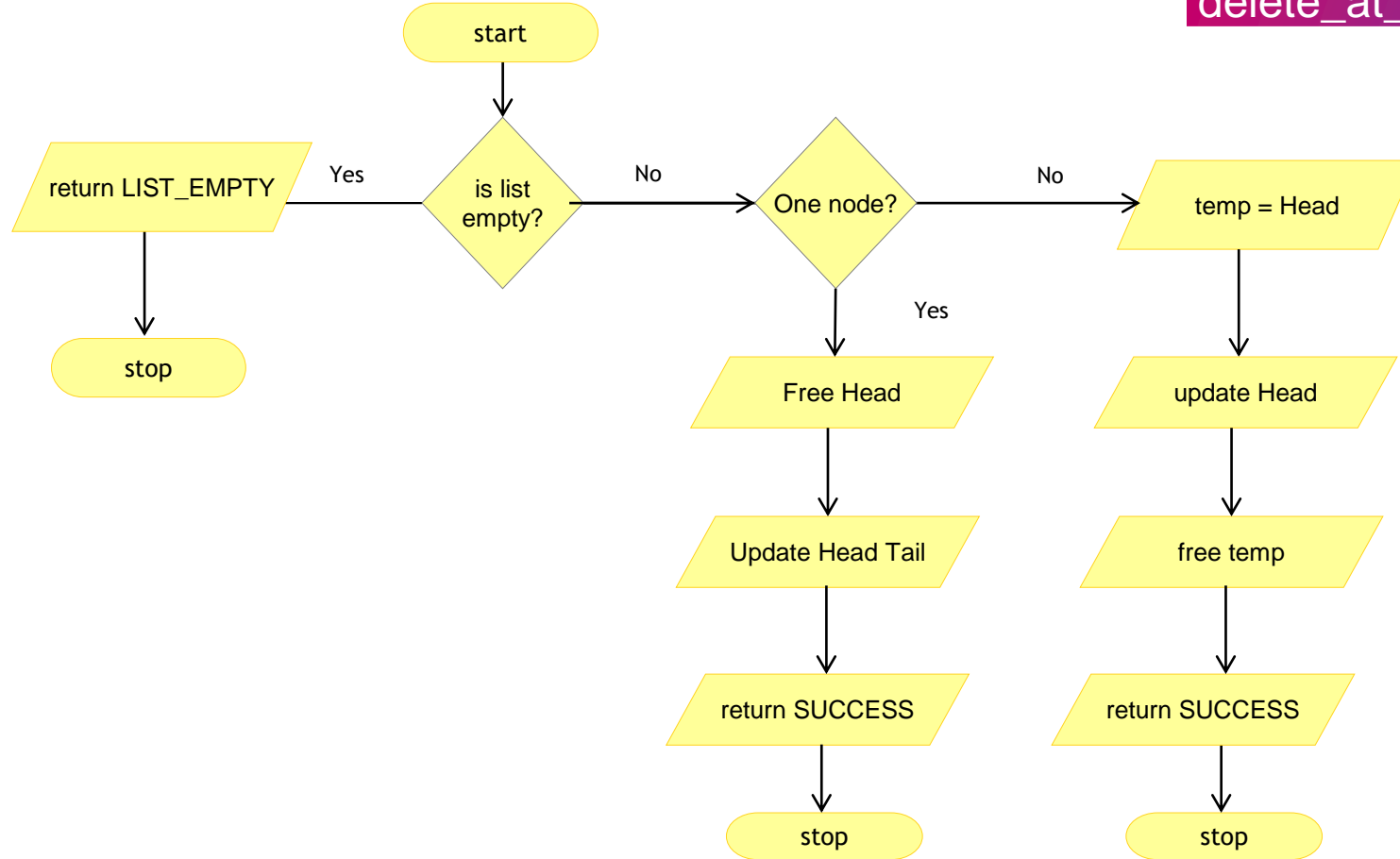
```
if ( Head = NULL)
    return LIST_EMPTY

if ( Head → next = NULL)
    free Head
    Head = Tail = NULL
    return SUCCESS

temp = Head
Head = temp → next
free temp
Head → prev = NULL
return SUCCESS
```

Algorithm  
delete\_at\_first







Double Linked List – delete\_at\_first -Code

