

Double Linked List – insert_at_last

Team Emertxe



Double Linked List – Insert at Last



Analysis – insert_at_last



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.



Analysis

insert_at_last



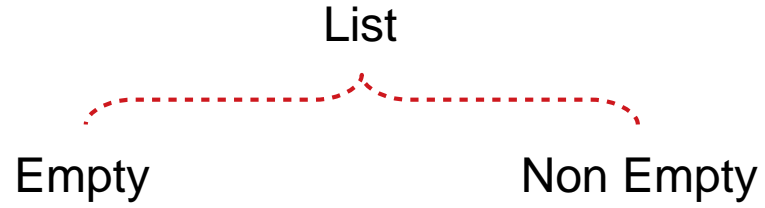
Cases :


Analysis
insert_at_last



List

Cases :






List Empty

Analysis



insert_at_last






List Empty

Head





List Empty

Head

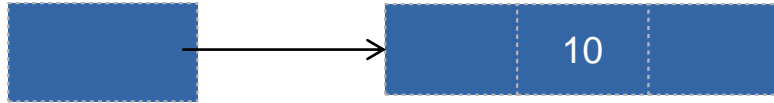
NULL

10



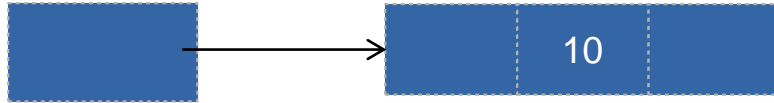
List Empty

Head



List Empty

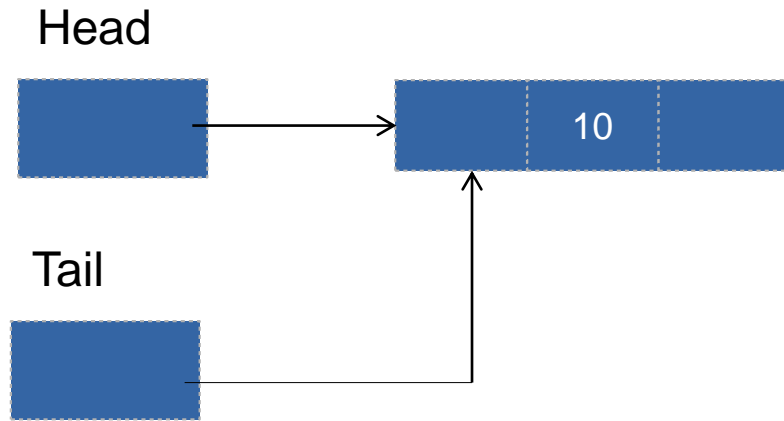
Head



Tail

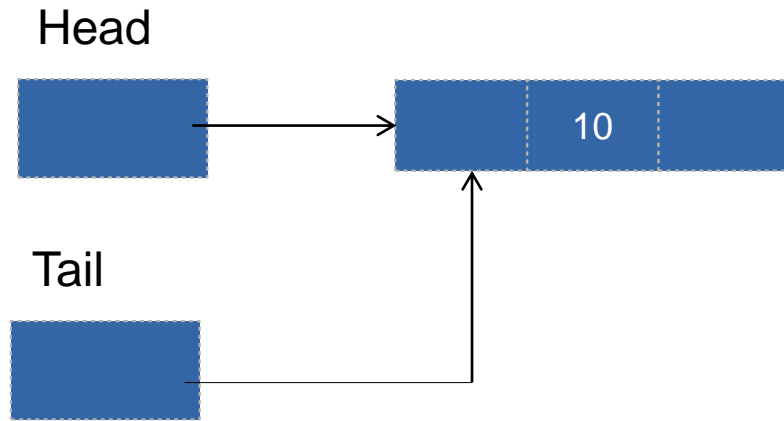


List Empty



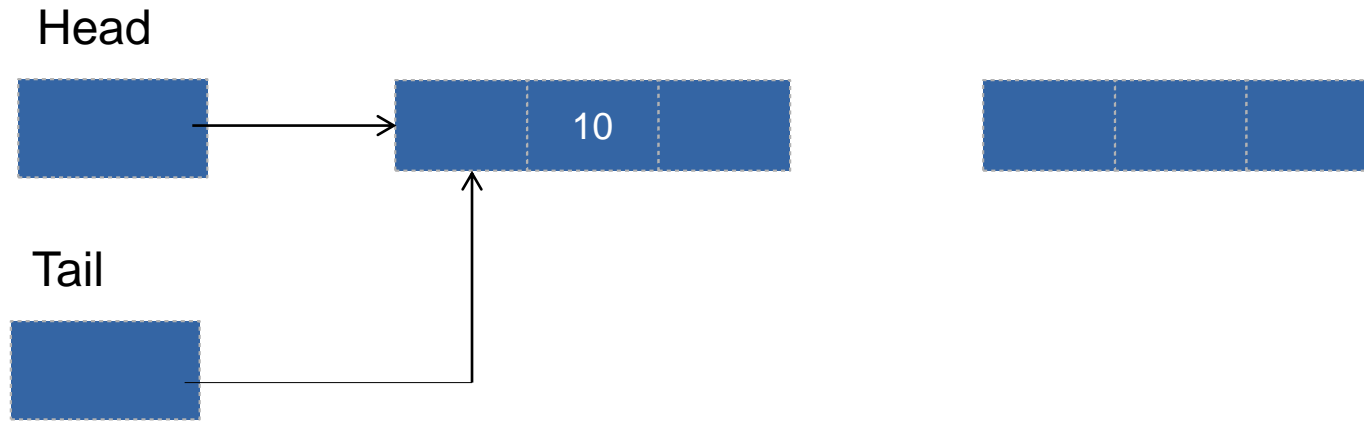
Non List Empty

Analysis
insert_at_last



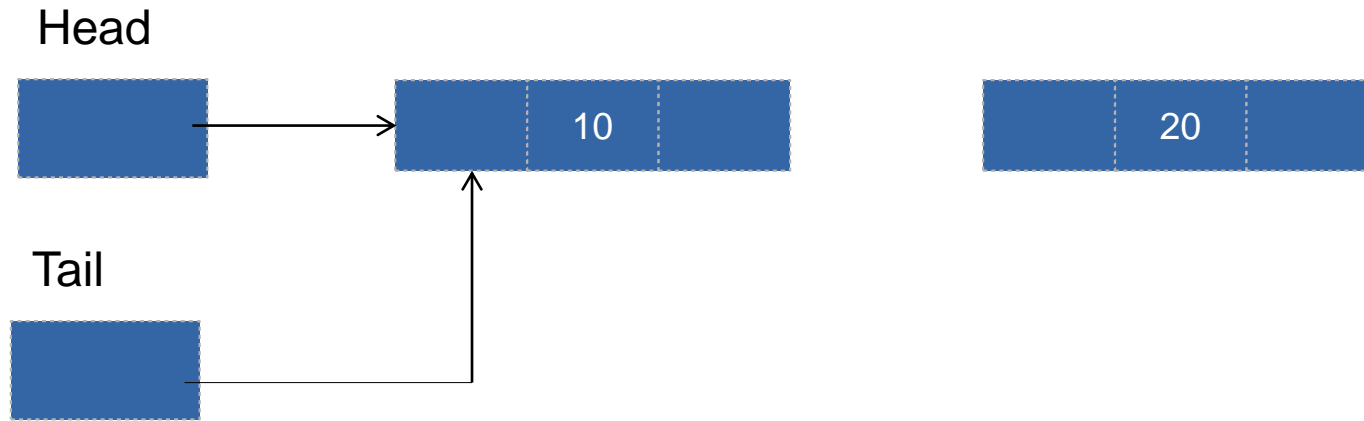
Non List Empty

Analysis
insert_at_last



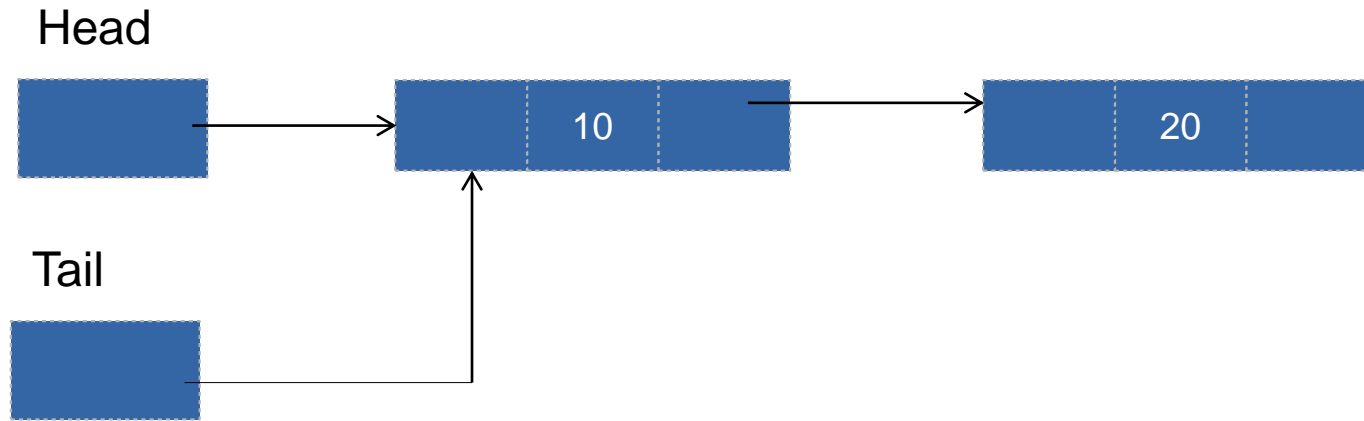
Non List Empty

Analysis
insert_at_last



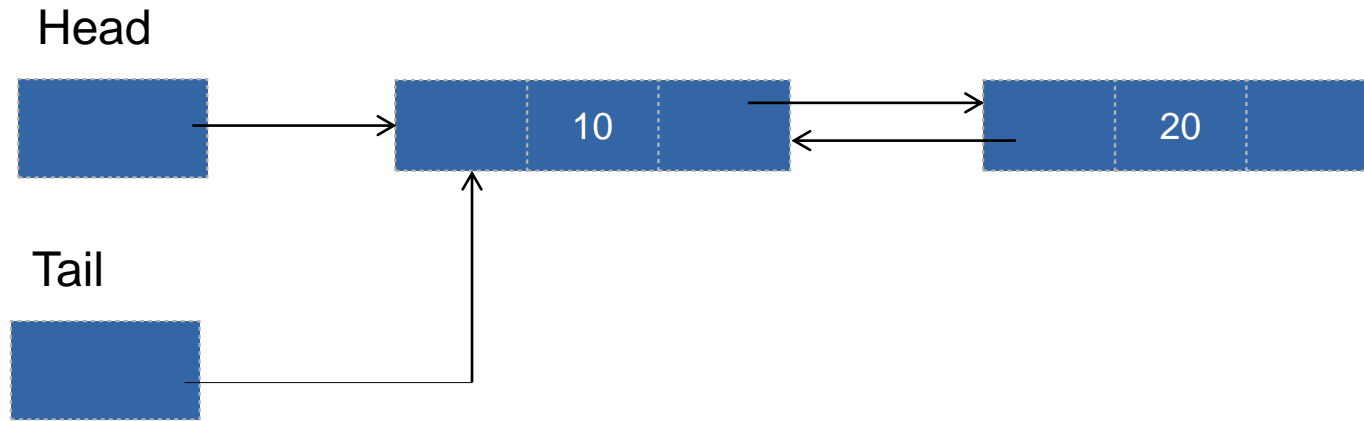
Non List Empty

Analysis
insert_at_last



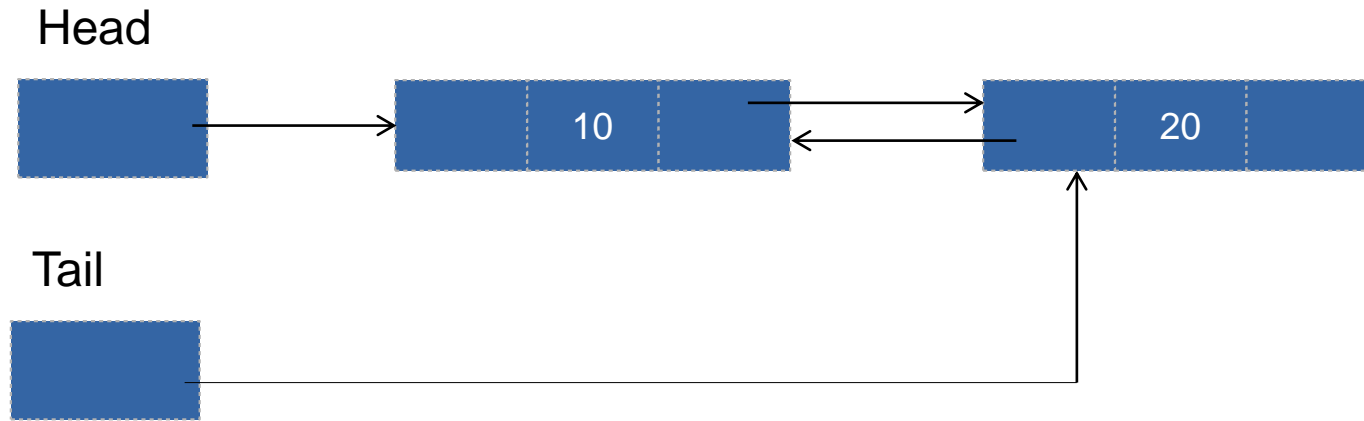
Non List Empty

Analysis
insert_at_last



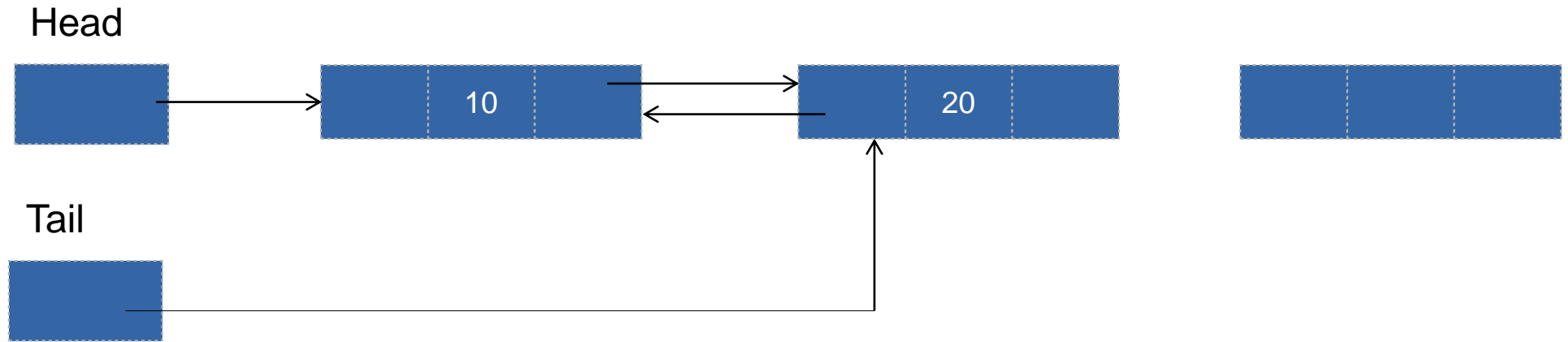
Non List Empty

Analysis
insert_at_last



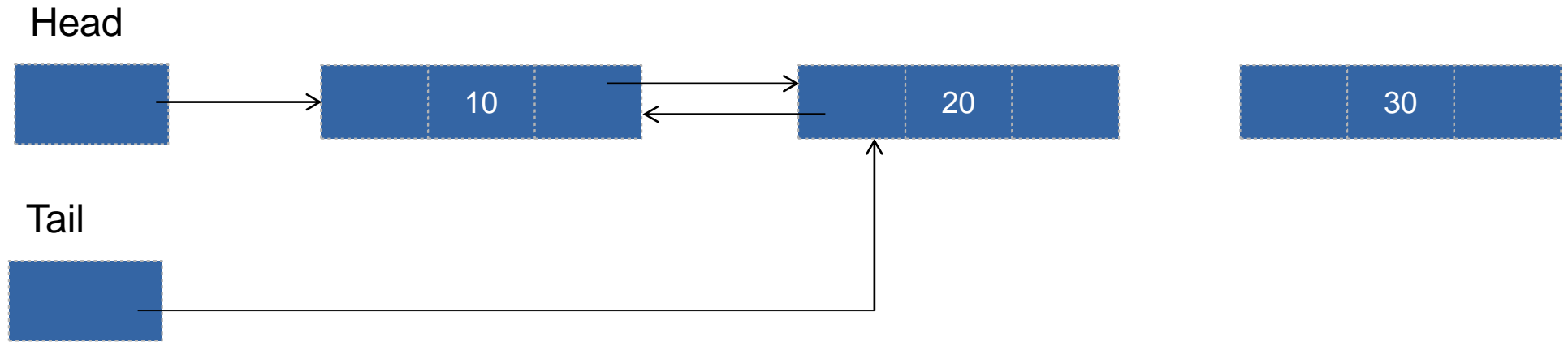
Non List Empty

Analysis
insert_at_last



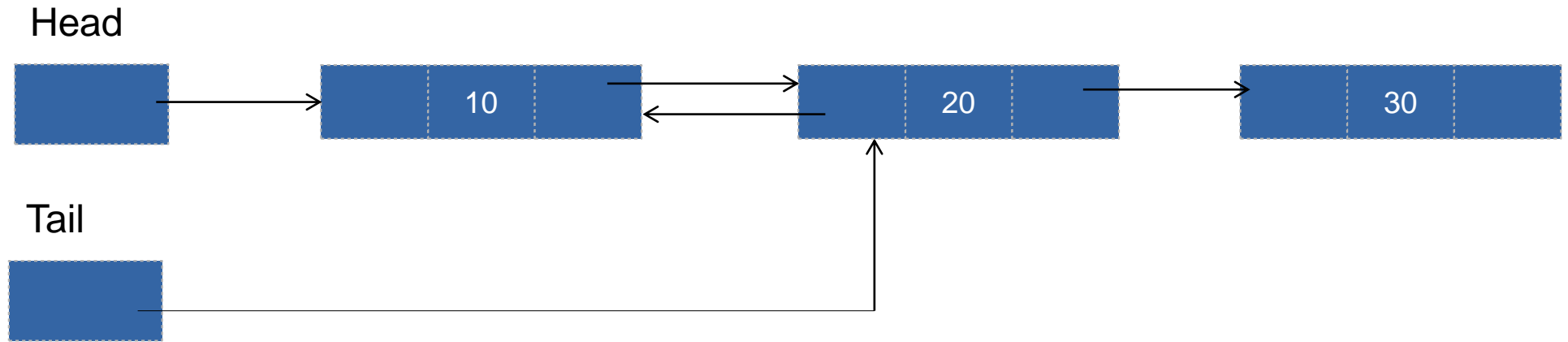
Non List Empty

Analysis
insert_at_last



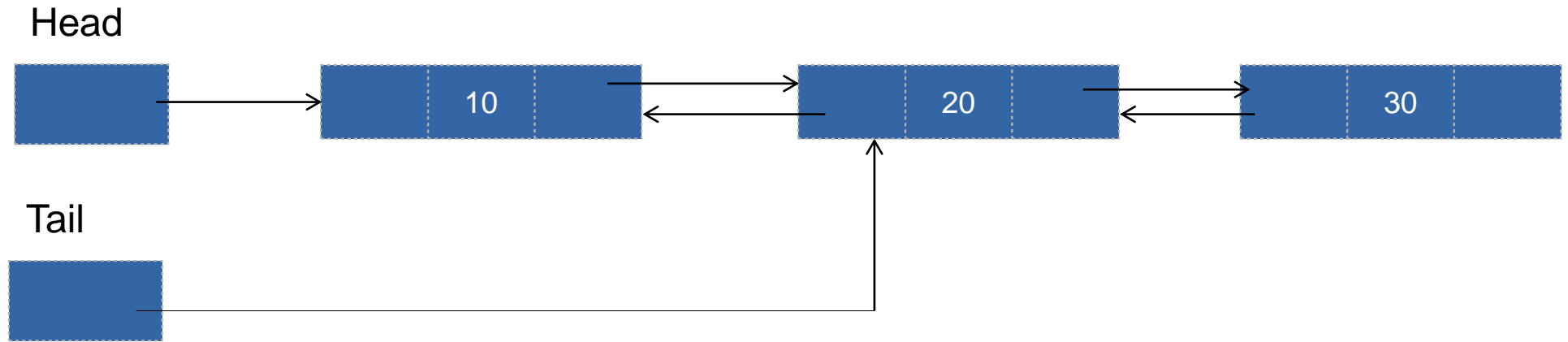
Non List Empty

Analysis
insert_at_last



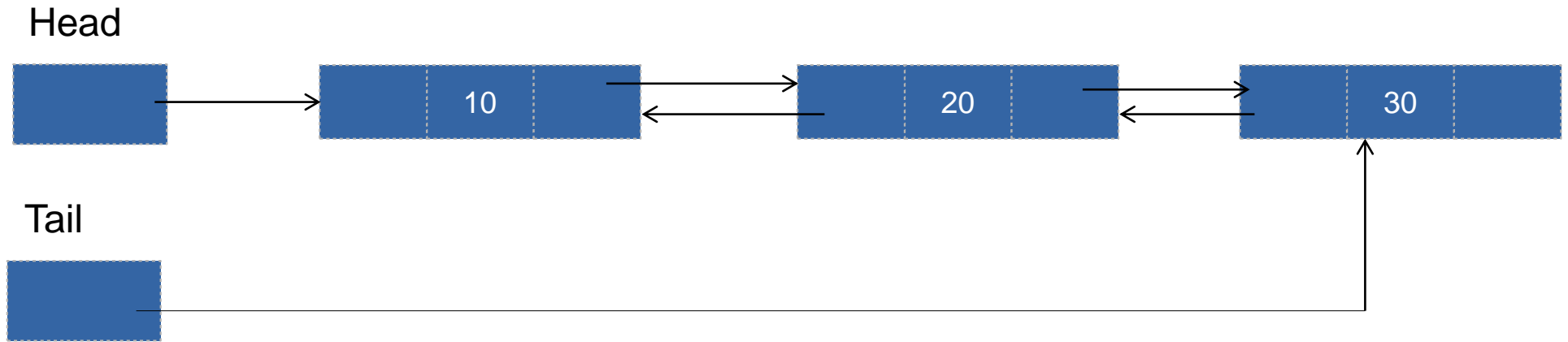
Non List Empty

Analysis
insert_at_last



Non List Empty

Analysis
insert_at_last



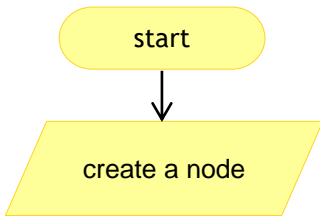
Flowchart

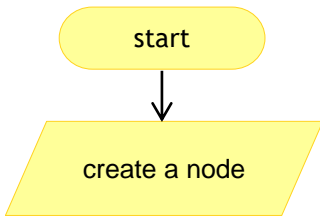
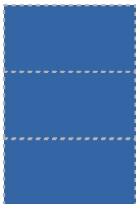


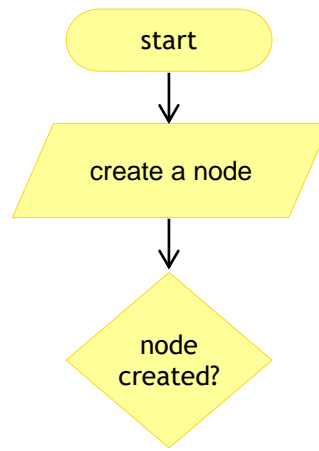


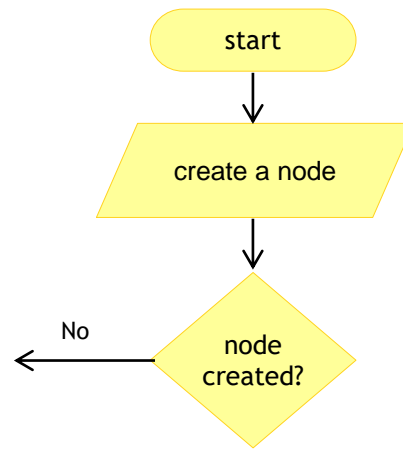
start

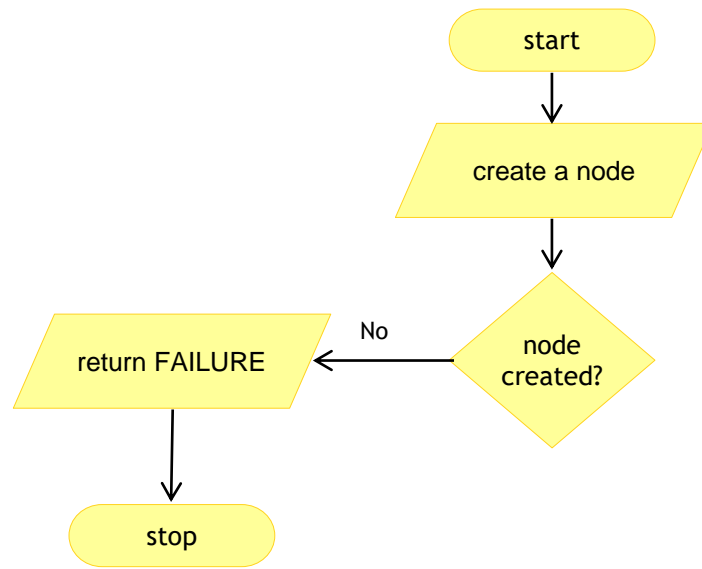


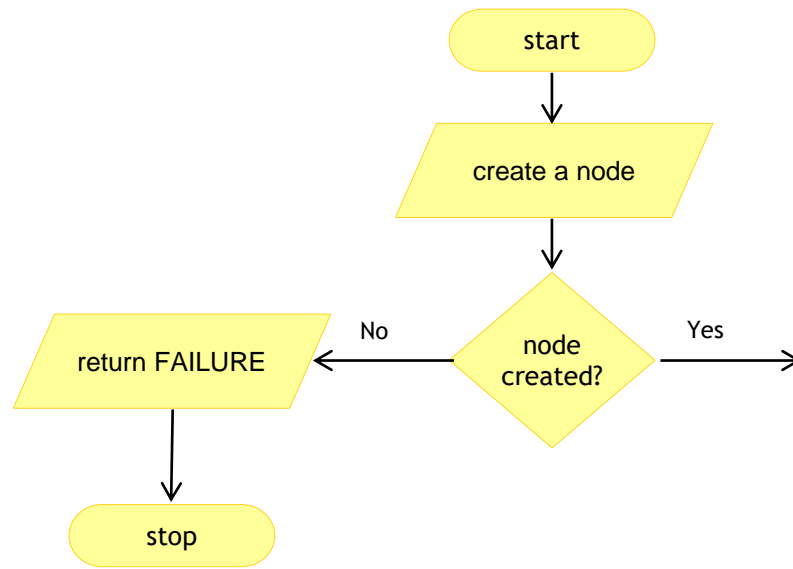


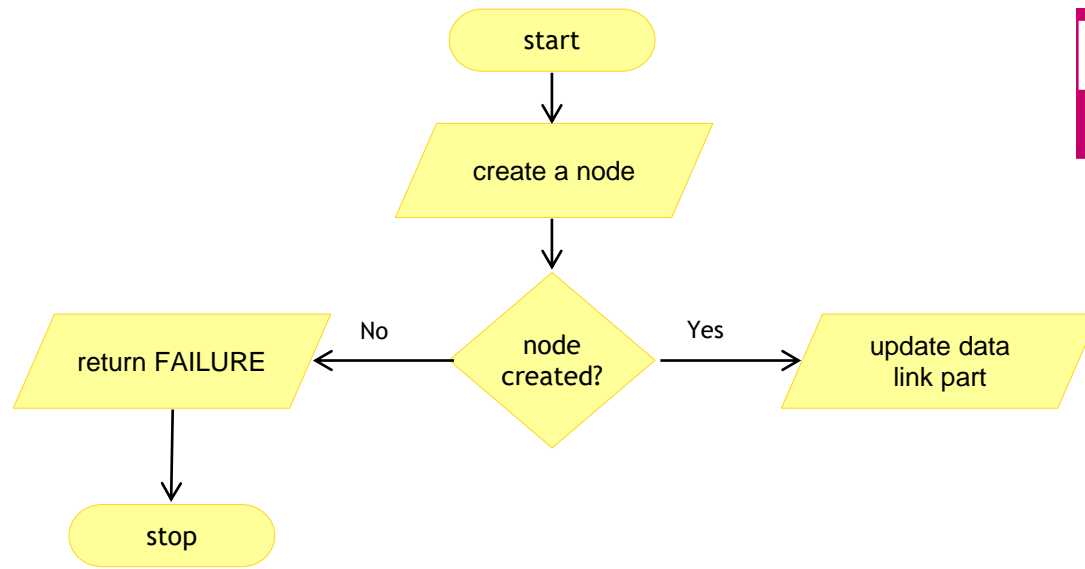




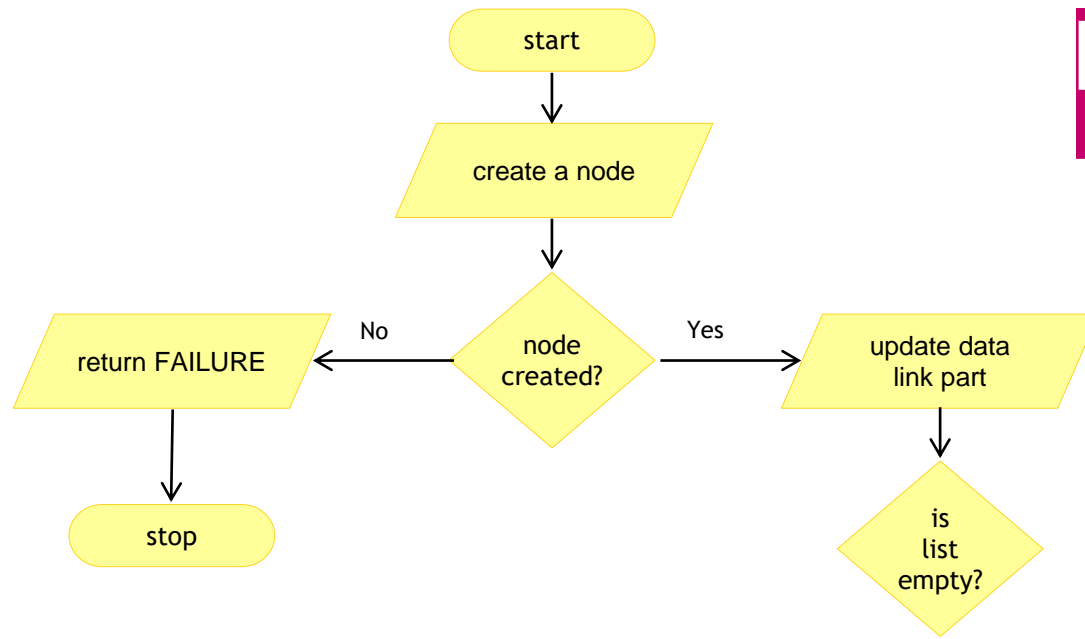




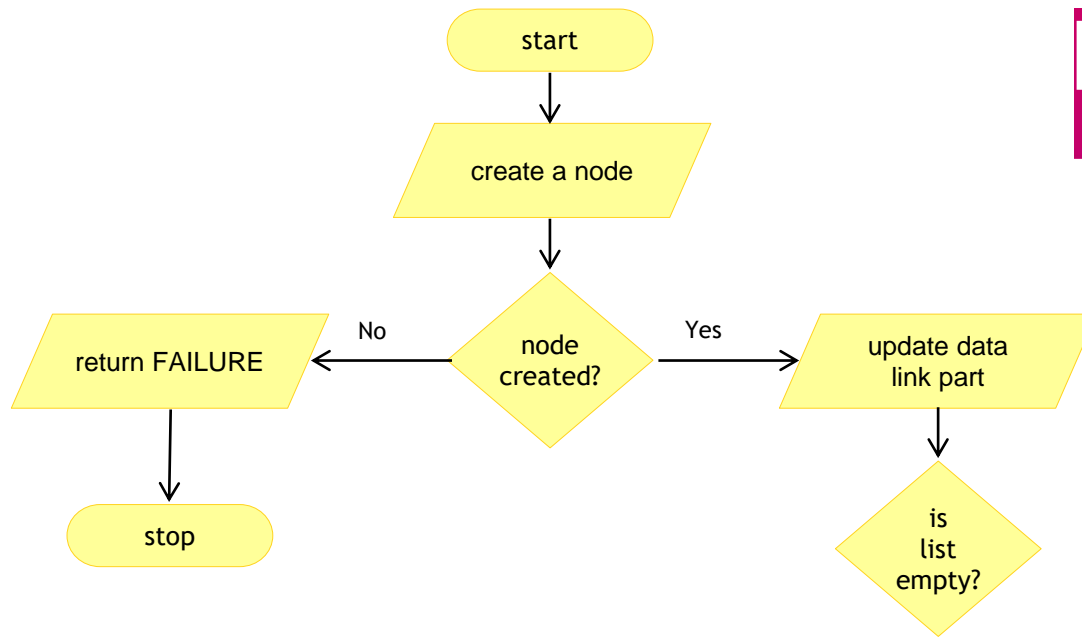




10



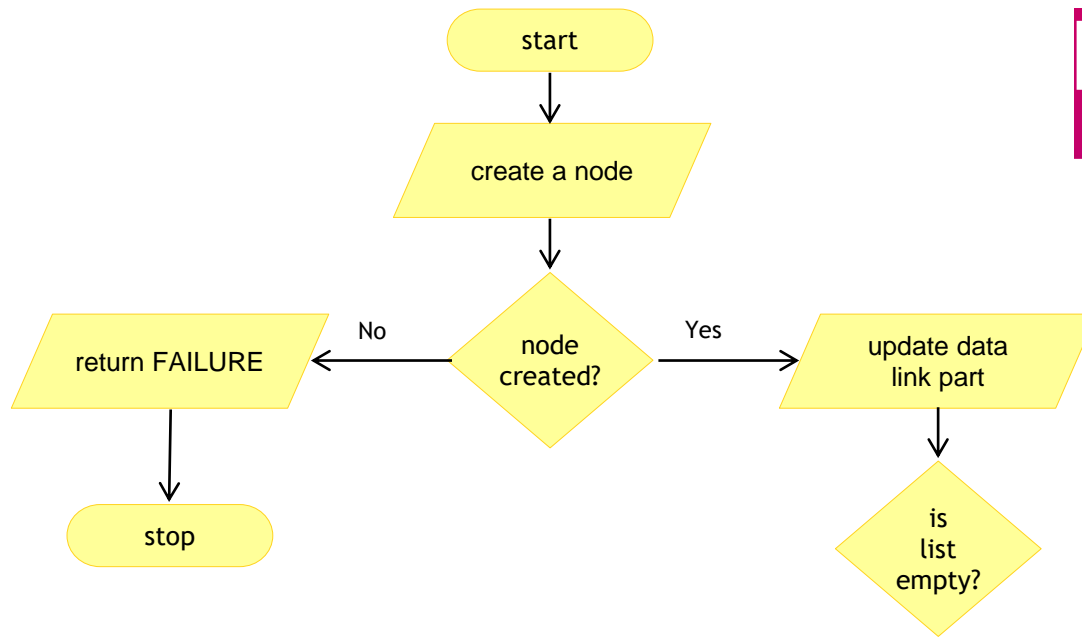
Head

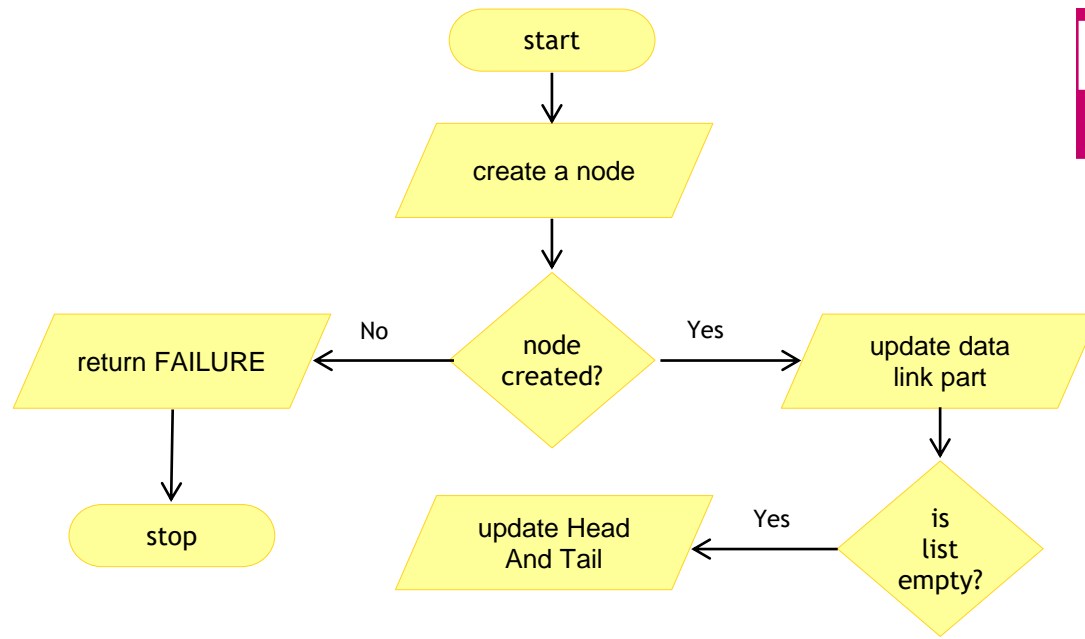


Head

NULL

10

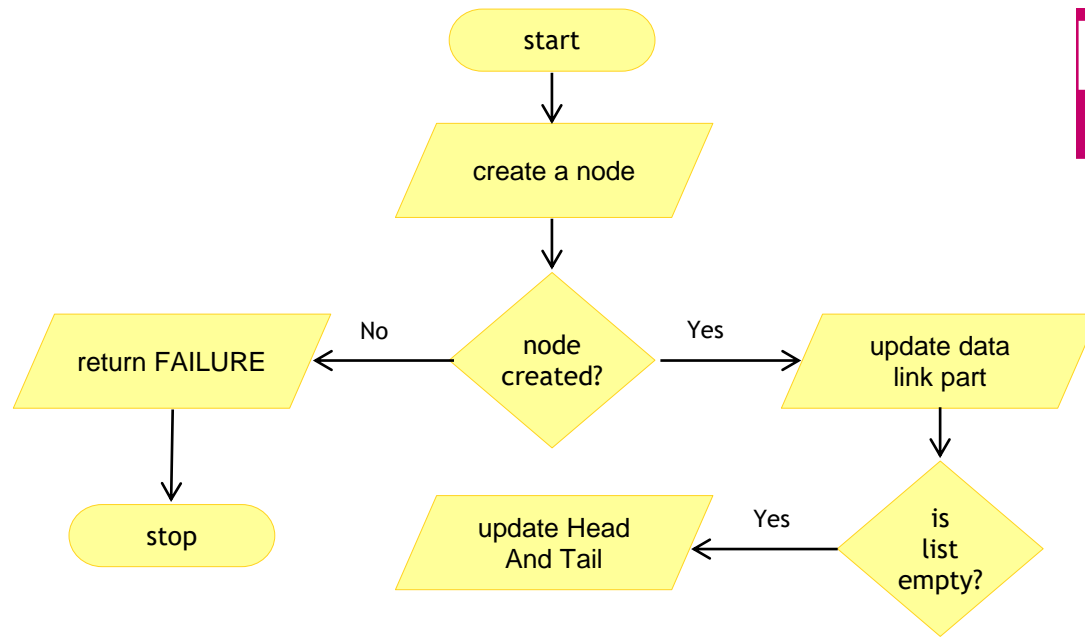


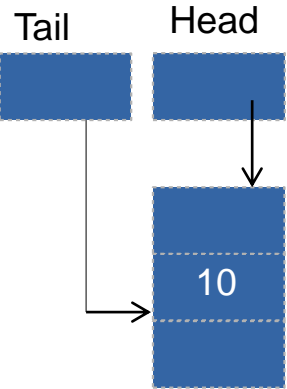
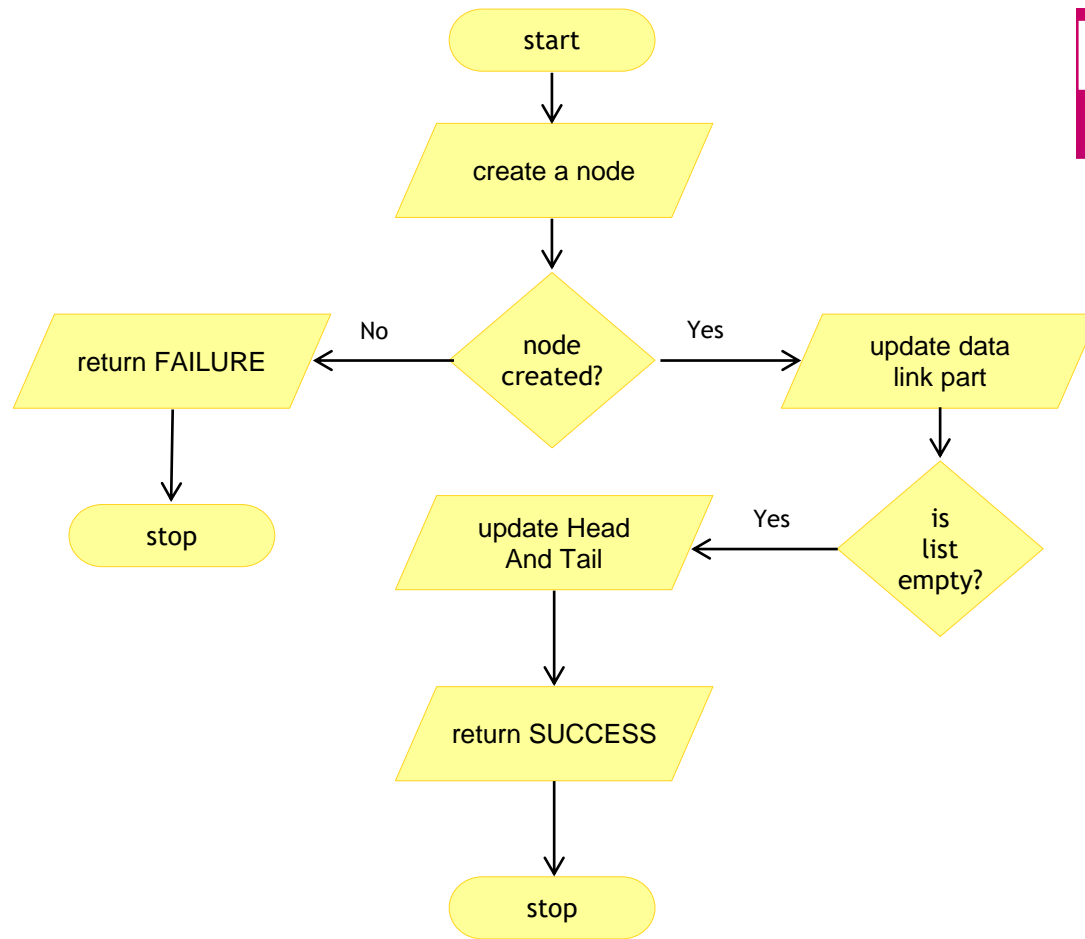


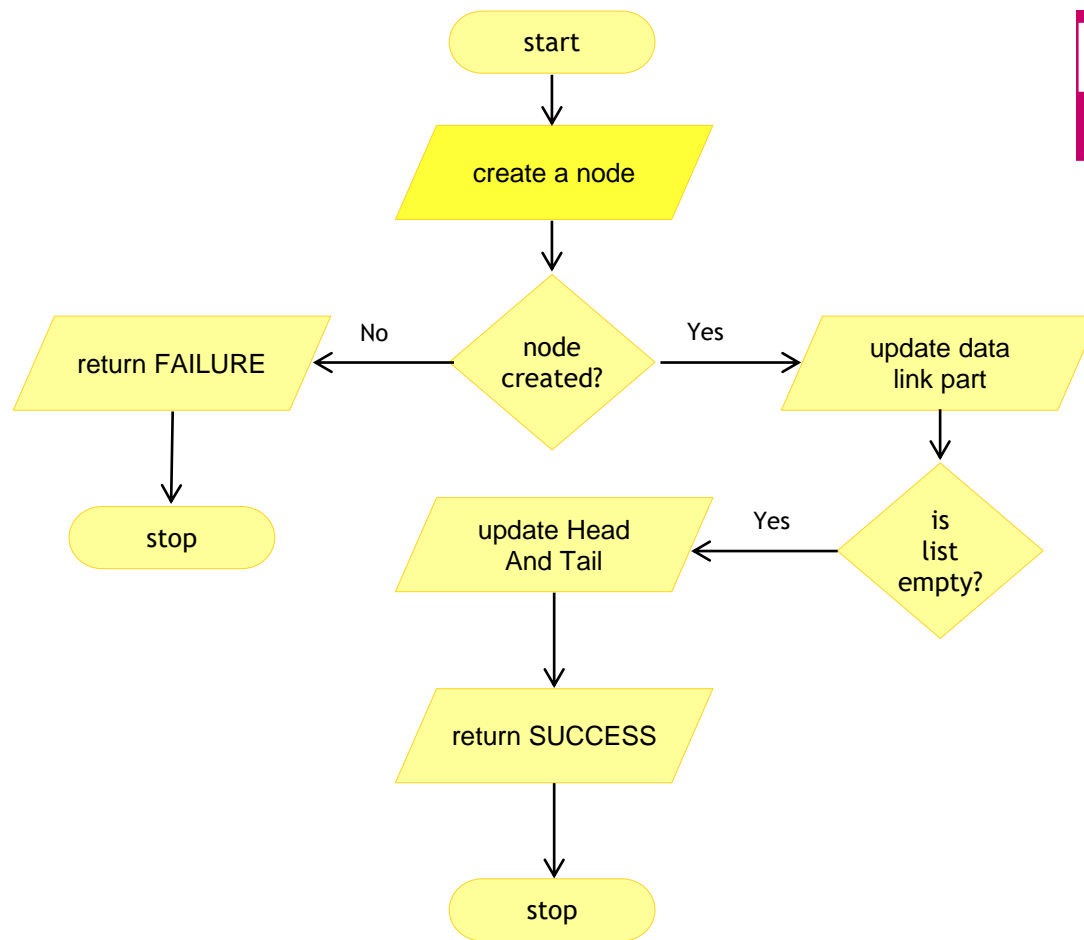
Head

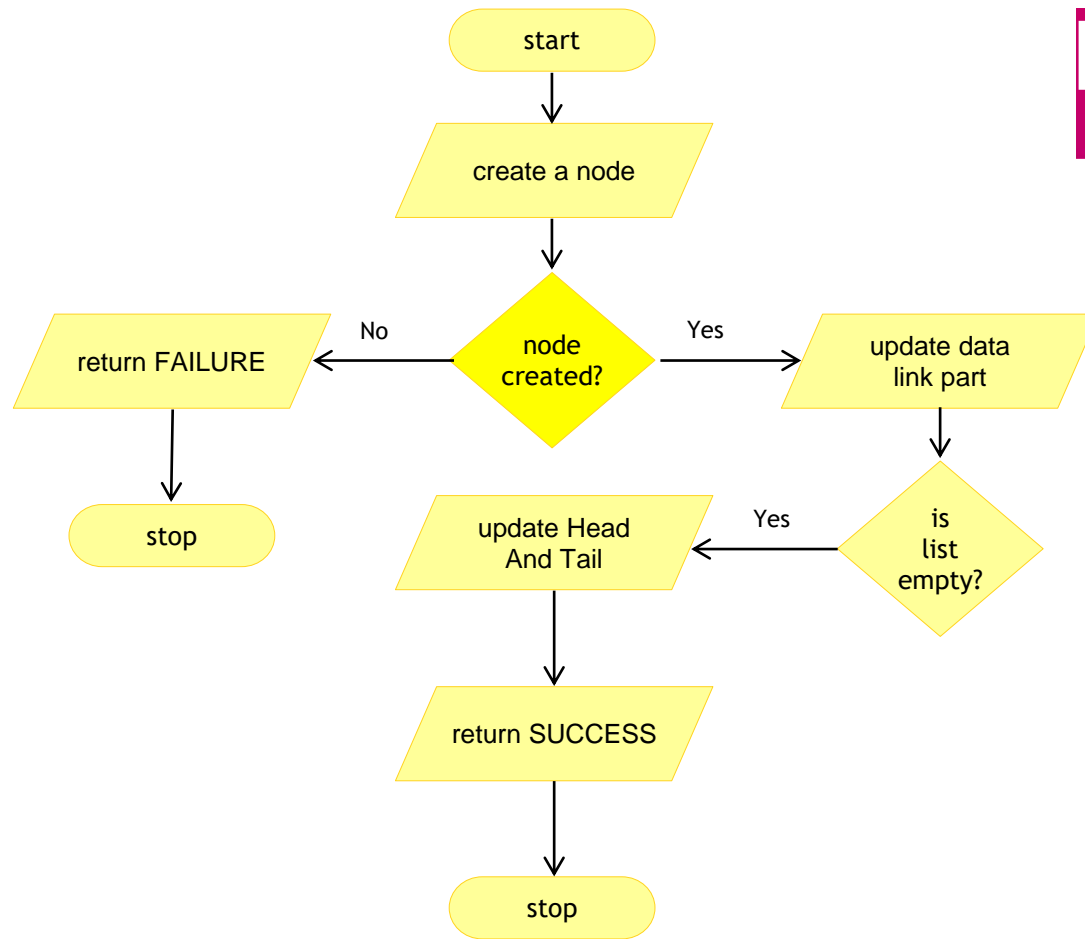
NULL

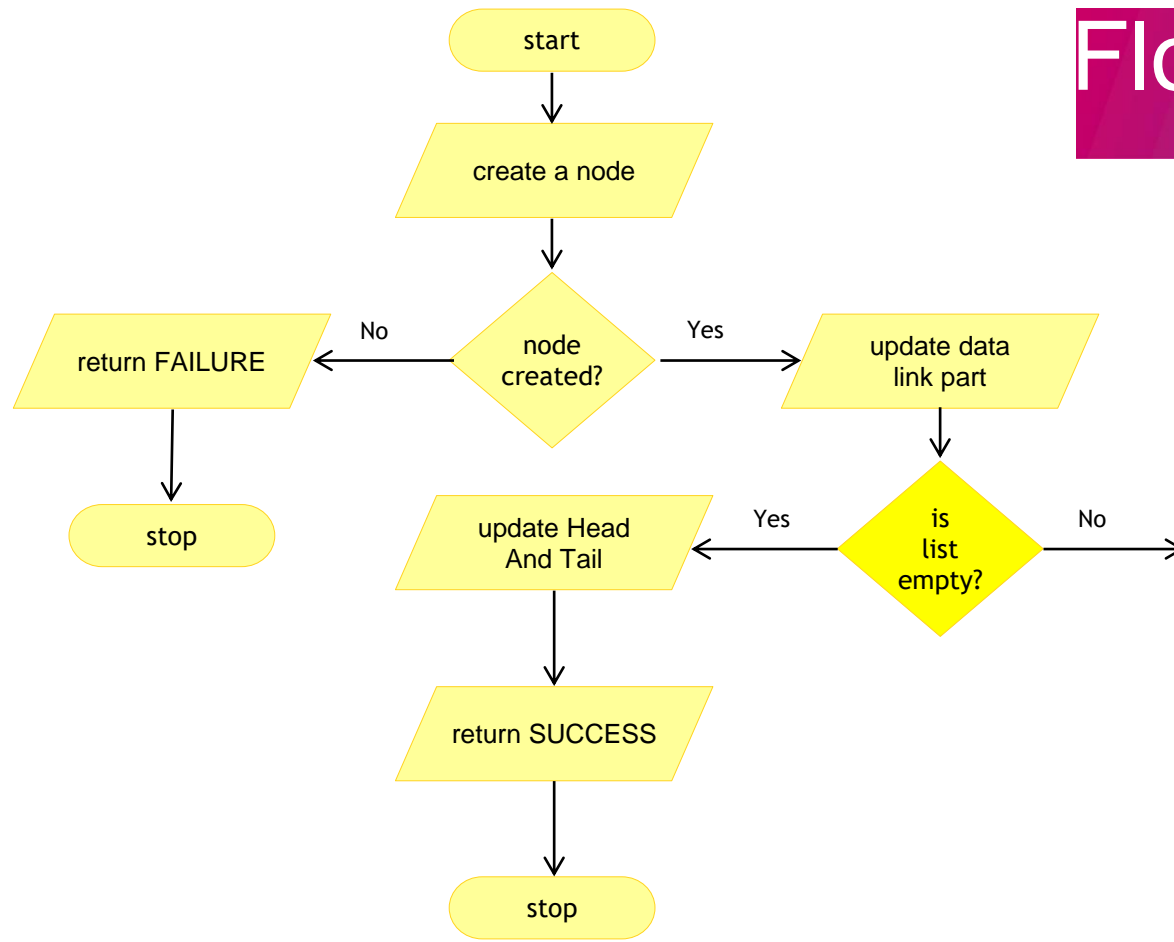
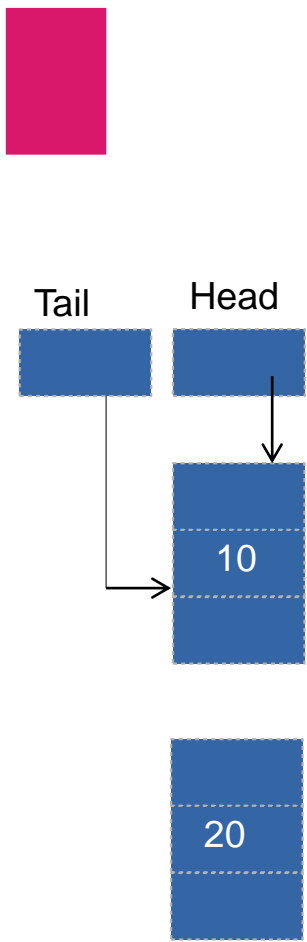
10

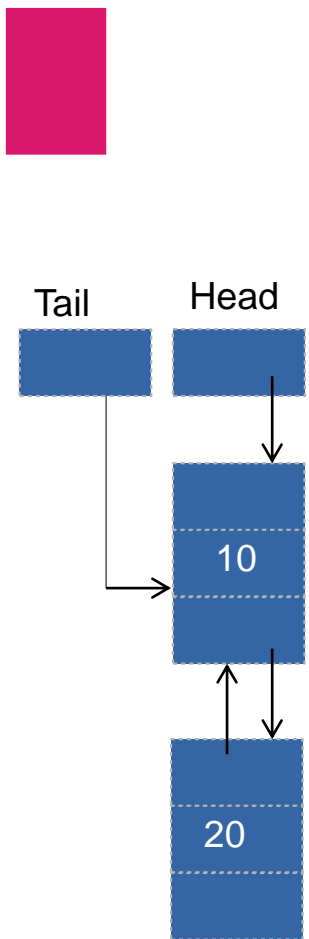
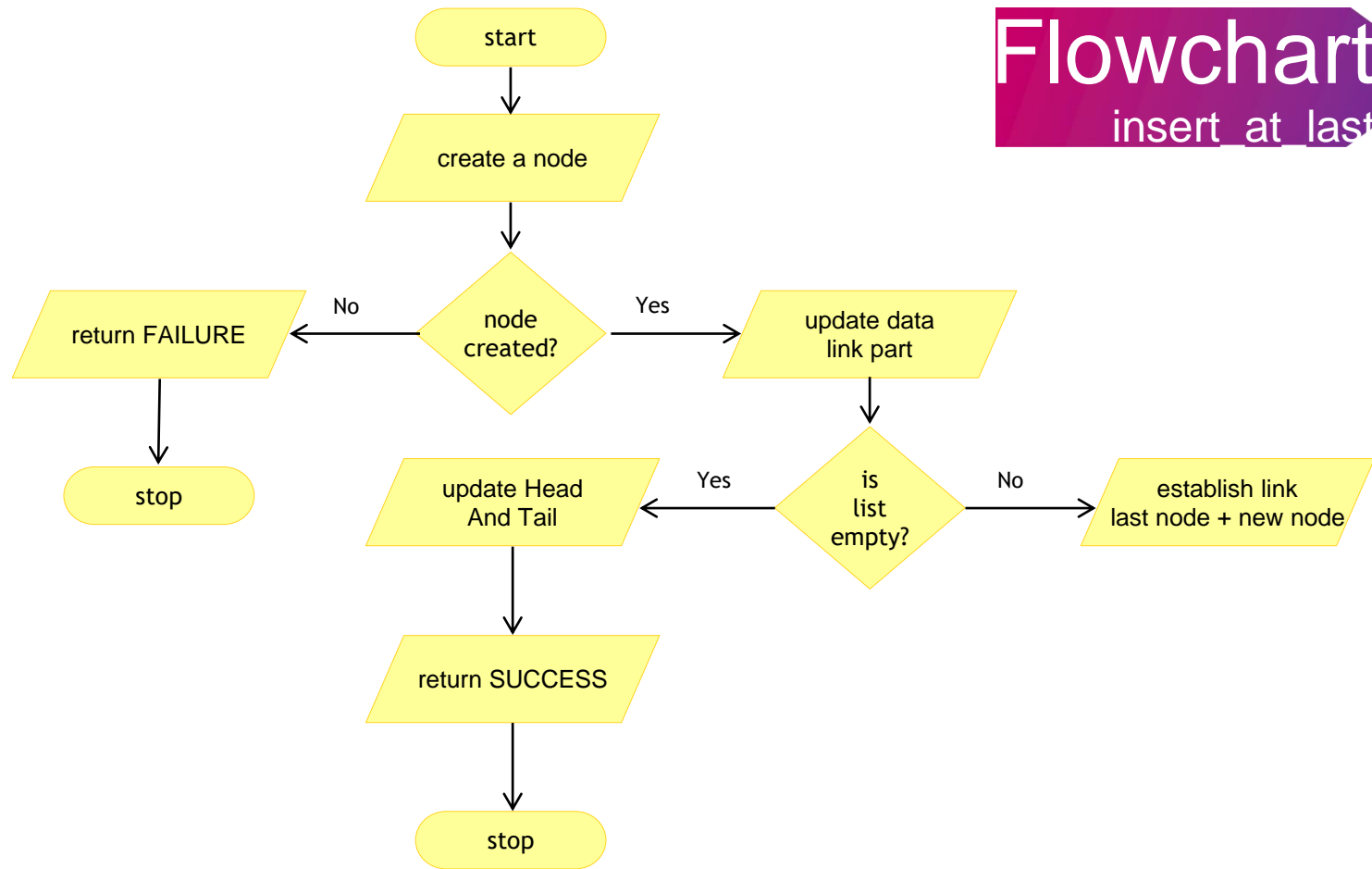


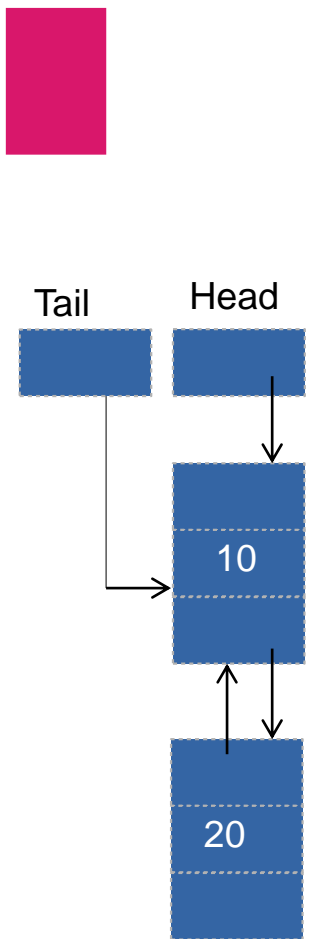
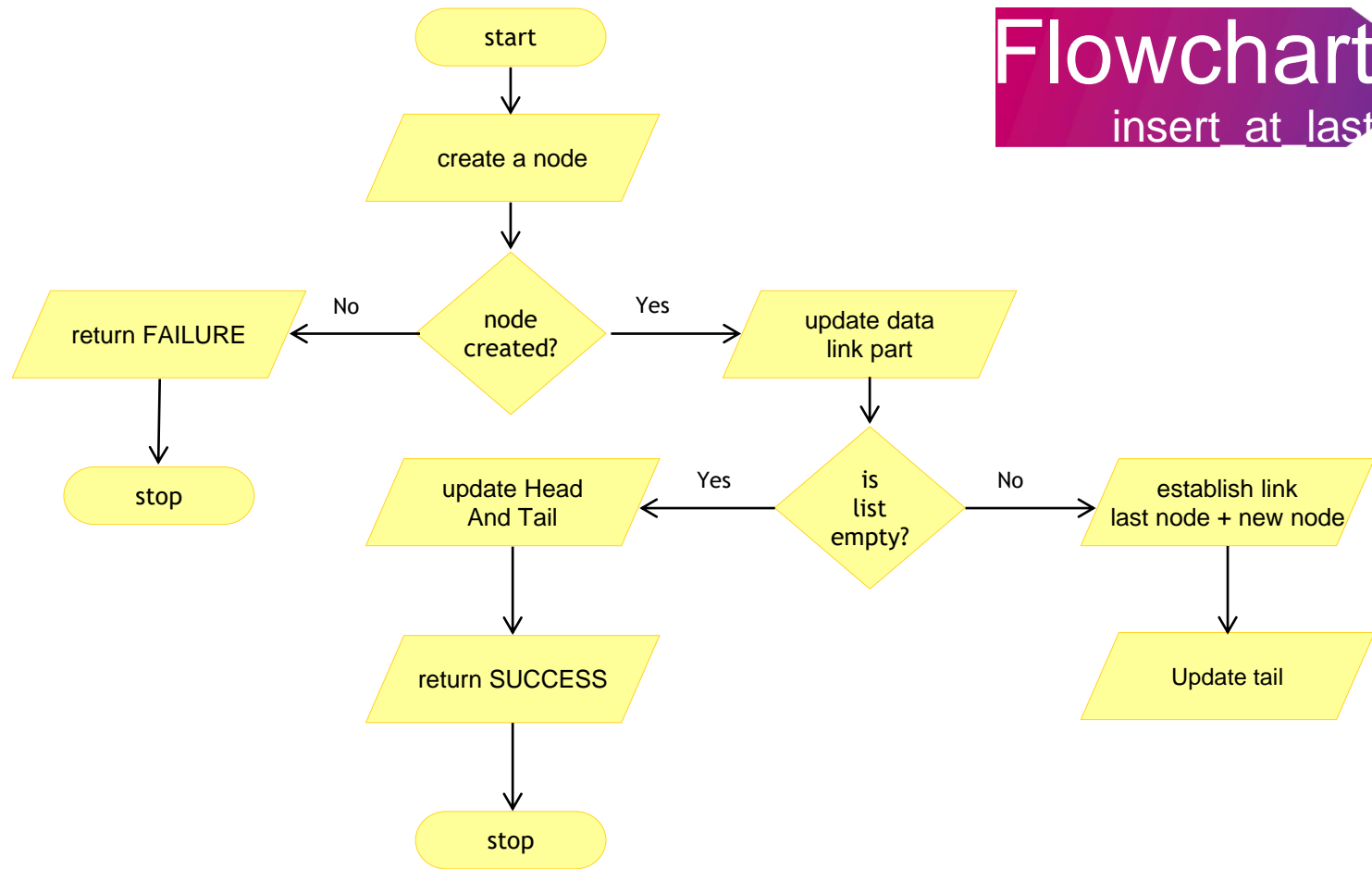


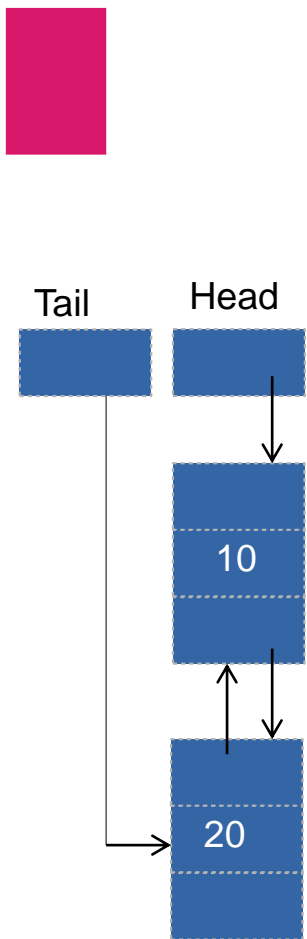
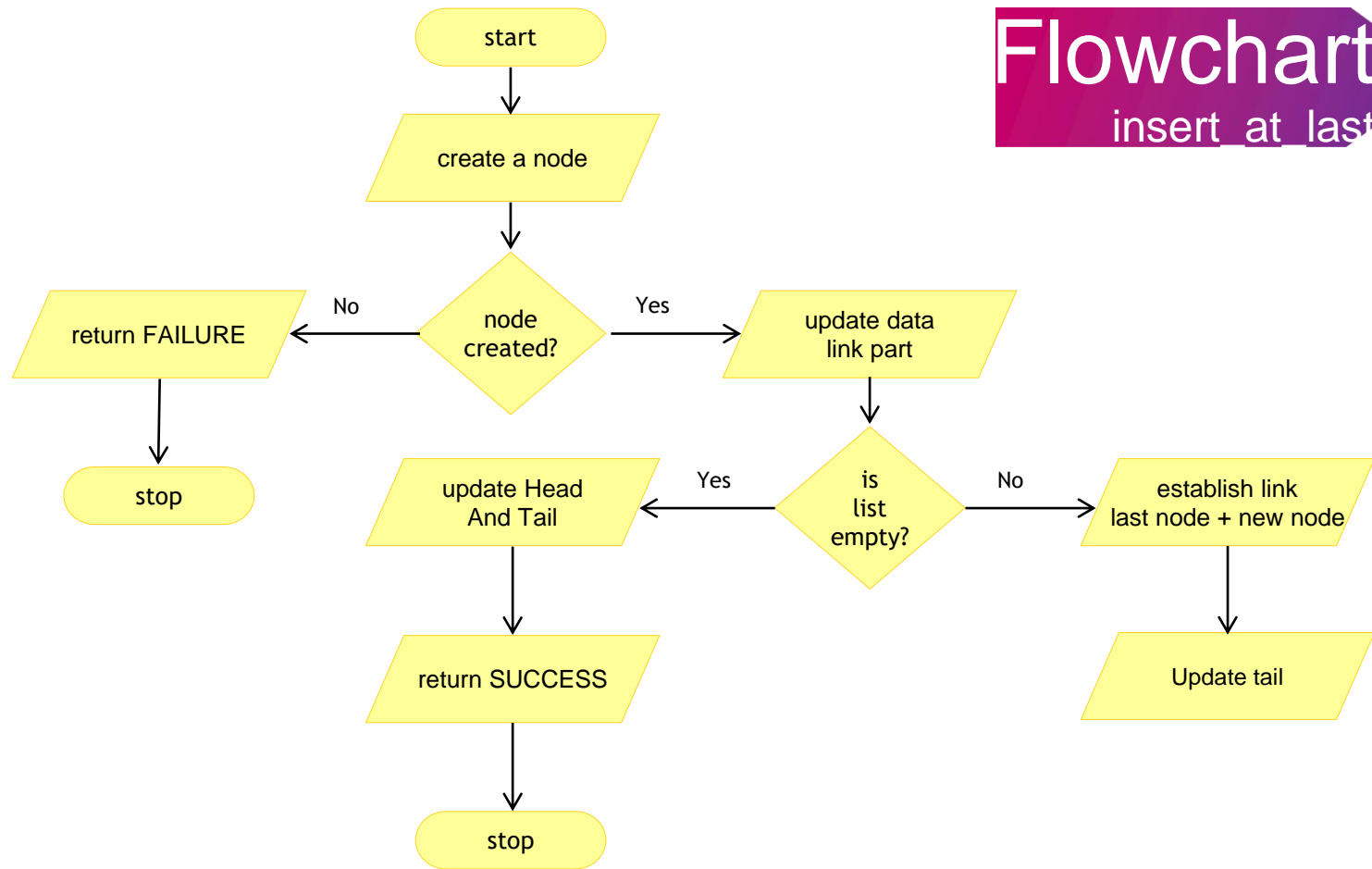


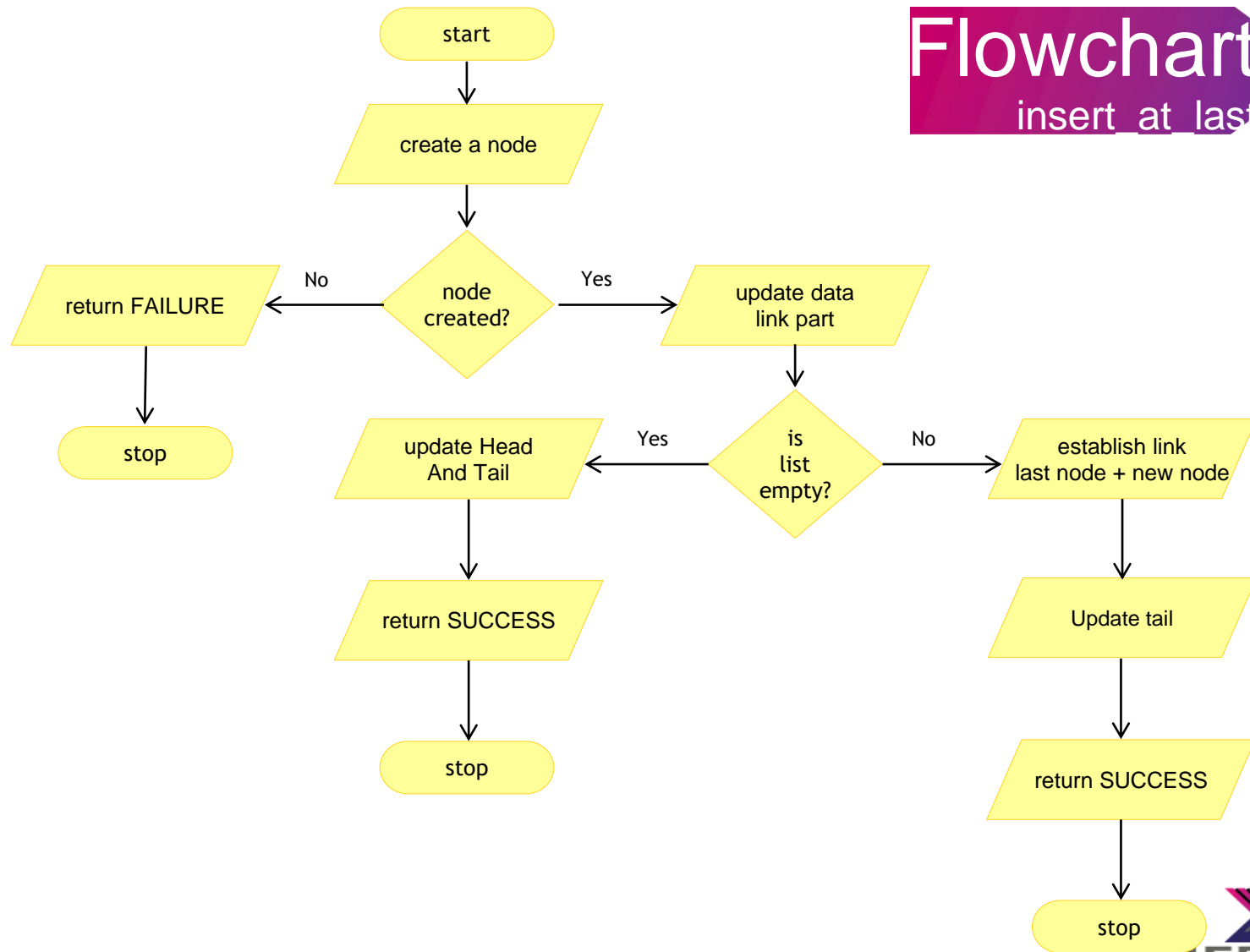
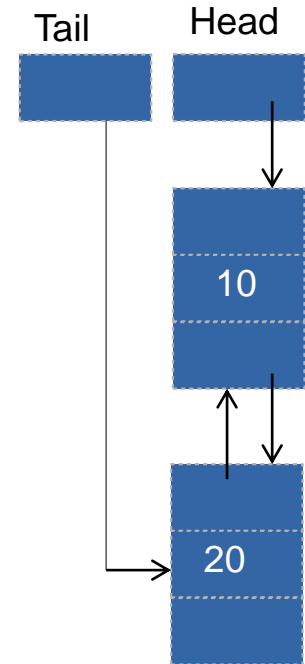












Algorithm

Algorithm : insert_at_last(Head,Tail,data)

1.Input Specification :-

Algorithm : insert_at_last(Head,Tail,data)

1.Input Specification :-

Head : Pointer containing first node address
Tail : Pointer containing last node address
data : Item to be added

Algorithm : insert_at_last(Head,Tail,data)

1.Input Specification :-

Head : Pointer containing first node address
Tail : Pointer containing last node address
data : Item to be added

2.Output Specification :-


Algorithm : insert_at_last(Head,Tail,data)

1.Input Specification :-

Head : Pointer containing first node address
Tail : Pointer containing last node address
data : Item to be added

2.Output Specification :-

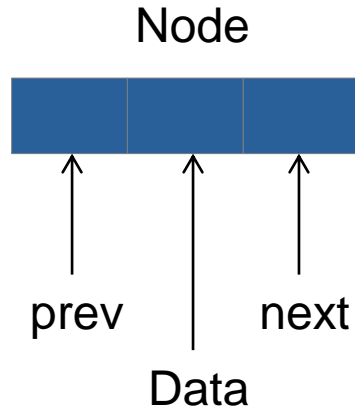
Status : SUCCESS / FAILURE



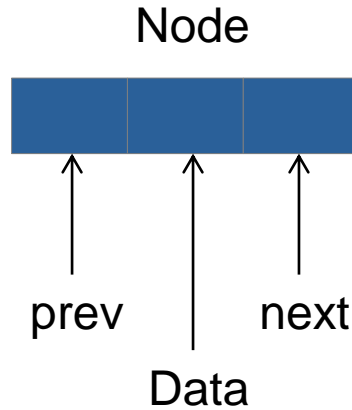
`insert_at_last(Head,Tail,data)`

Algorithm 
insert_at_last

`insert_at_last(Head,Tail,data)`



insert_at_last(Head,Tail,data)



```
typedef int data_t  
typedef struct node  
{  
    struct node *prev;  
    data_t data;  
    struct node *next;  
} Dlist;
```




insert_at_last(Head,Tail,data)

new = Memalloc(sizeof (Dlist))

Algorithm 
insert_at_last

insert_at_last(Head,Tail,data)

new = Memalloc(sizeof (Dlist))

insert_at_last(Head,Tail,data)

```
new = Memalloc(sizeof (Dlist) )  
if ( new = NULL)  
    return FAILURE
```

insert_at_last(Head,Tail,data)

```
new = Memalloc(sizeof (Dlist) )  
if ( new = NULL)  
    return FAILURE  
  
new → data = data  
new → next = NULL  
new → prev = NULL
```

insert_at_last(Head,Tail,data)

```
new = Memalloc(sizeof (Dlist) )
if ( new = NULL)
    return FAILURE
new → data = data
new → next = NULL
new → prev = NULL
```

NULL
10
NULL

Head

NULL

NULL

10

NULL

insert_at_last(Head,Tail,data)

```
new = Memalloc(sizeof (Dlist) )
```

```
if ( new = NULL)
```

```
    return FAILURE
```

```
new —>data = data
```

```
new —>next = NULL
```

```
new —>prev = NULL
```

```
if (Head = NULL)
```

Tail

Head

NULL

10

NULL

insert_at_last(Head,Tail,data)

```
new = Memalloc(sizeof (Dlist) )
```

```
if ( new = NULL)
```

```
    return FAILURE
```

```
new —>data = data
```

```
new —>next = NULL
```

```
new —>prev = NULL
```

```
if (Head = NULL)
```

```
    Head = Tail = new
```

Algorithm

insert_at_last



insert_at_last(Head,Tail,data)

```
new = Memalloc(sizeof (Dlist) )
if ( new = NULL)
    return FAILURE

new → data = data
new → next = NULL
new → prev = NULL

if (Head = NULL)
    Head = Tail = new
return SUCCESS
```




insert_at_last(Head,Tail,data)

```
new = Memalloc(sizeof (Dlist) )
```

```
if ( new = NULL)
```

```
    return FAILURE
```

```
new  —> data =    data
```

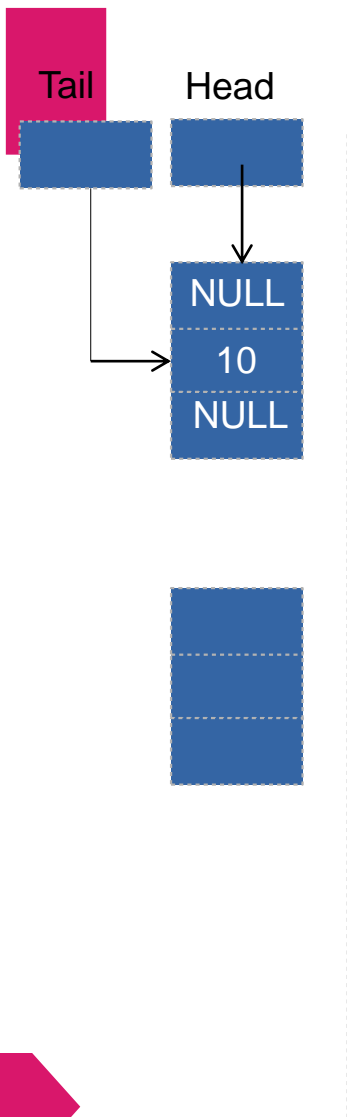
```
new  —> next =    NULL
```

```
new  —> prev =    NULL
```

```
if (Head = NULL)
```

```
    Head =    Tail =    new
```

```
    return SUCCESS
```



insert_at_last(Head,Tail,data)

```
new = Memalloc(sizeof (Dlist) )
```

```
if ( new = NULL)
```

```
    return FAILURE
```

```
new —> data = data
```

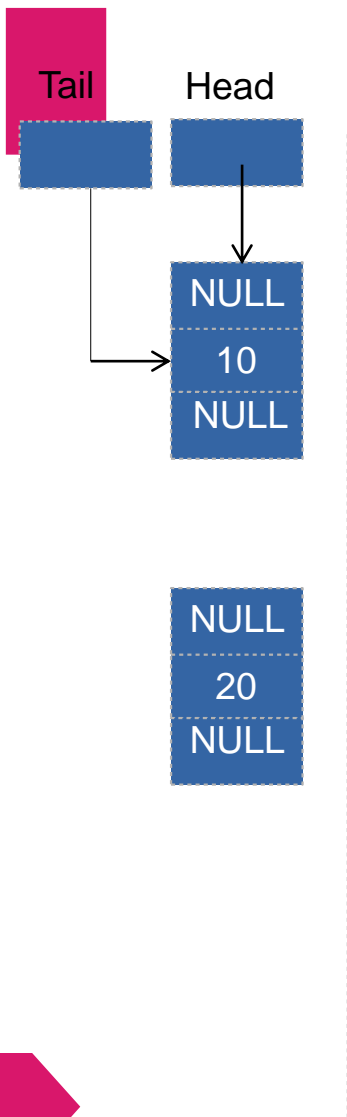
```
new —> next = NULL
```

```
new —> prev = NULL
```

```
if (Head = NULL)
```

```
    Head = Tail = new
```

```
    return SUCCESS
```



insert_at_last(Head,Tail,data)

```
new = Memalloc(sizeof (Dlist) )
```

```
if ( new = NULL)
```

```
    return FAILURE
```

```
new —> data = data
```

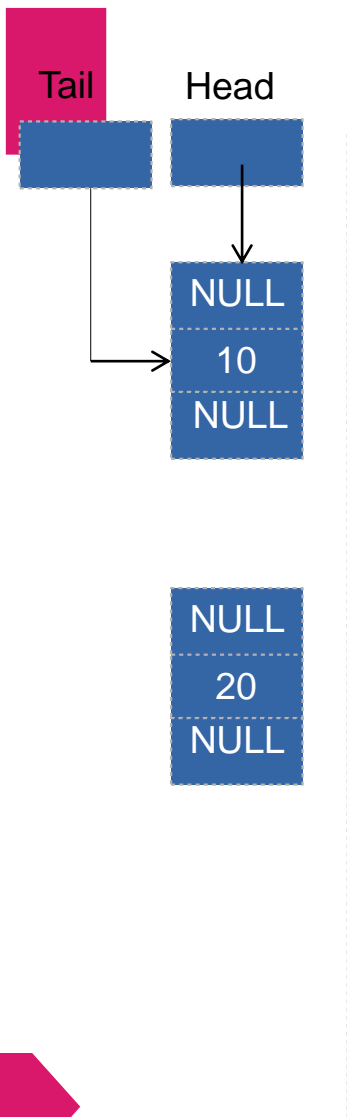
```
new —> next = NULL
```

```
new —> prev = NULL
```

```
if (Head = NULL)
```

```
    Head = Tail = new
```

```
    return SUCCESS
```



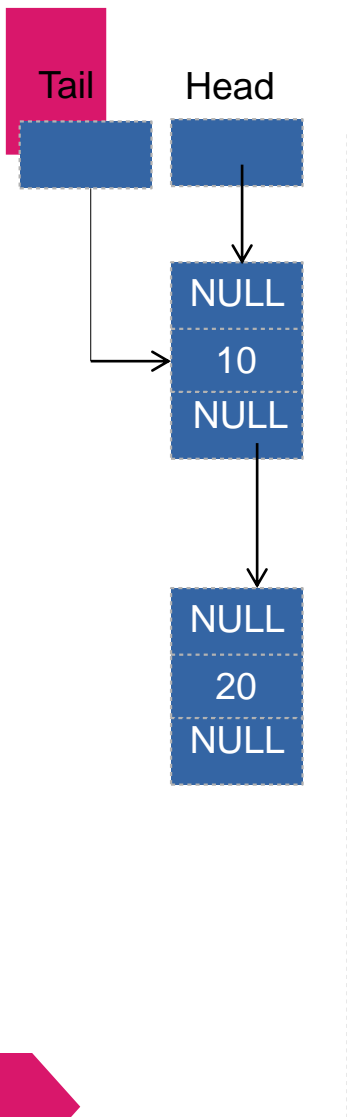
insert_at_last(Head,Tail,data)

```

new = Memalloc(sizeof (Dlist) )
if ( new = NULL)
    return FAILURE

new —> data = data
new —> next = NULL
new —> prev = NULL

if (Head = NULL)
    Head = Tail = new
return SUCCESS
    
```



insert_at_last(Head,Tail,data)

```
new = Memalloc(sizeof (Dlist) )
```

```
if ( new = NULL)
```

```
    return FAILURE
```

```
new —> data = data
```

```
new —> next = NULL
```

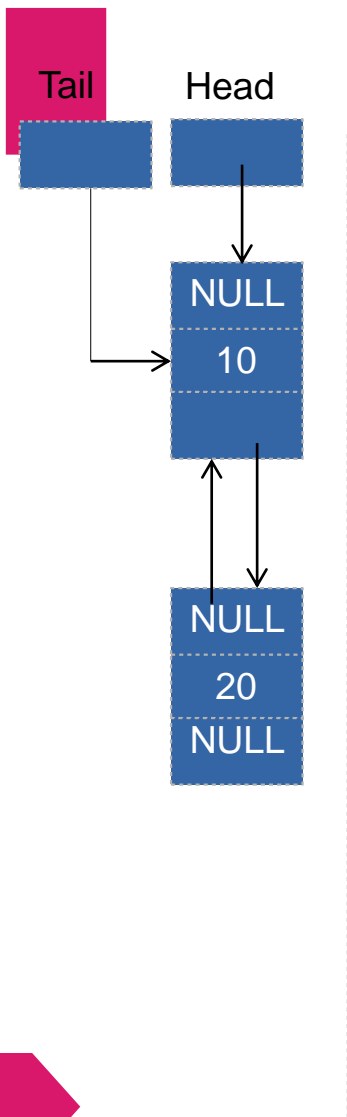
```
new —> prev = NULL
```

```
if (Head = NULL)
```

```
    Head = Tail = new
```

```
    return SUCCESS
```

```
Tail —> next = new
```



insert_at_last(Head,Tail,data)

```
new = Memalloc(sizeof (Dlist) )
```

```
if ( new = NULL)
```

```
    return FAILURE
```

```
new —> data = data
```

```
new —> next = NULL
```

```
new —> prev = NULL
```

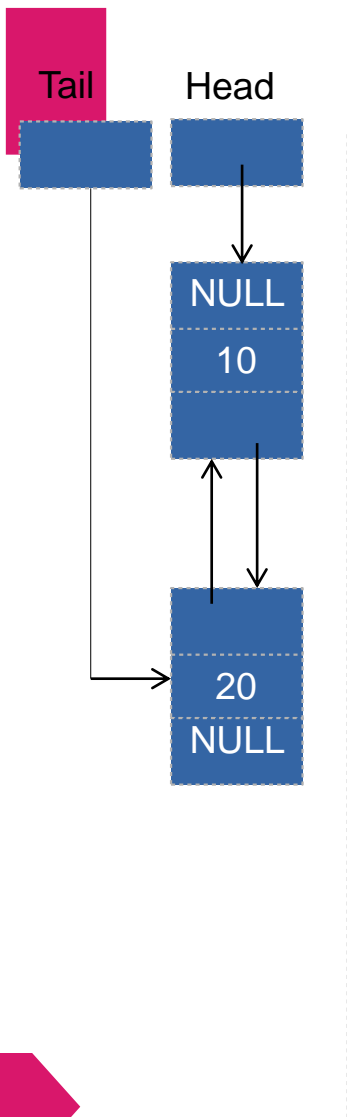
```
if (Head = NULL)
```

```
    Head = Tail = new
```

```
    return SUCCESS
```

```
Tail —> next = new
```

```
new —> prev = Tail
```



insert_at_last(Head,Tail,data)

```
new = Memalloc(sizeof (Dlist) )
```

```
if ( new = NULL)
```

```
    return FAILURE
```

```
new → data = data
```

```
new → next = NULL
```

```
new → prev = NULL
```

```
if (Head = NULL)
```

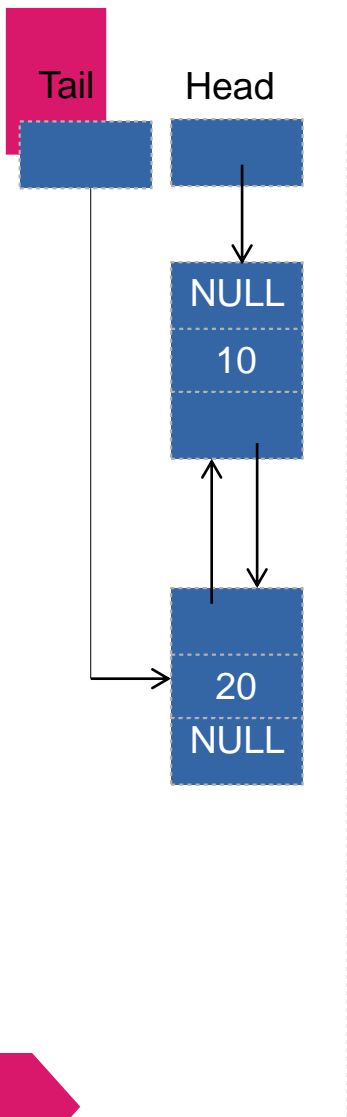
```
    Head = Tail = new
```

```
    return SUCCESS
```

```
Tail → next = new
```

```
new → prev = Tail
```

```
Tail = new
```



insert_at_last(Head,Tail,data)

```

new = Memalloc(sizeof (Dlist) )
if ( new = NULL)
    return FAILURE

new → data = data
new → next = NULL
new → prev = NULL

if (Head = NULL)
    Head = Tail = new
    return SUCCESS

Tail → next = new
new → prev = Tail
Tail = new
return SUCCESS
    
```


Code – Insert at Last

