Data Structures

Double Linked List – insert_after

Team Emertxe

Double Linked List

Data Structure – Linked List

Analysis – insert_after

Analysis: Logic / Cases

Flowchart

Algorithm

Code

Insert after - Analysis

Analysis

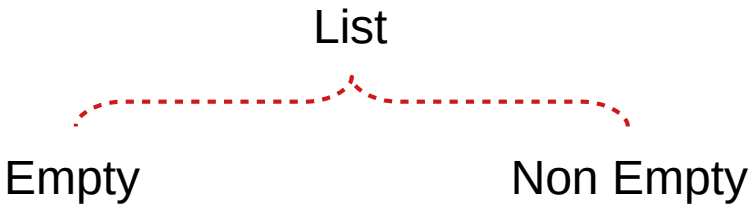
insert_after

Cases :

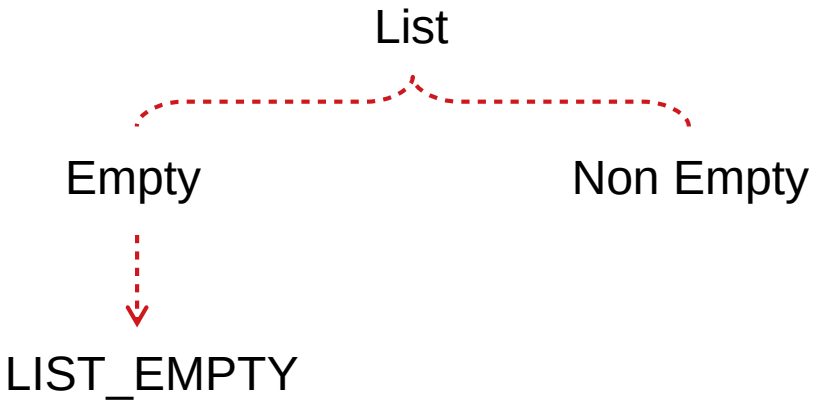
Analysis
insert_after

List

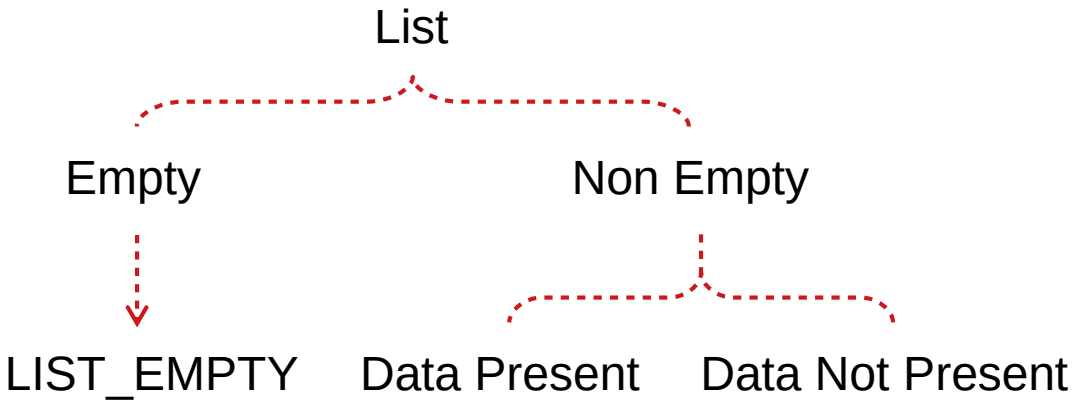
Cases :



Cases :

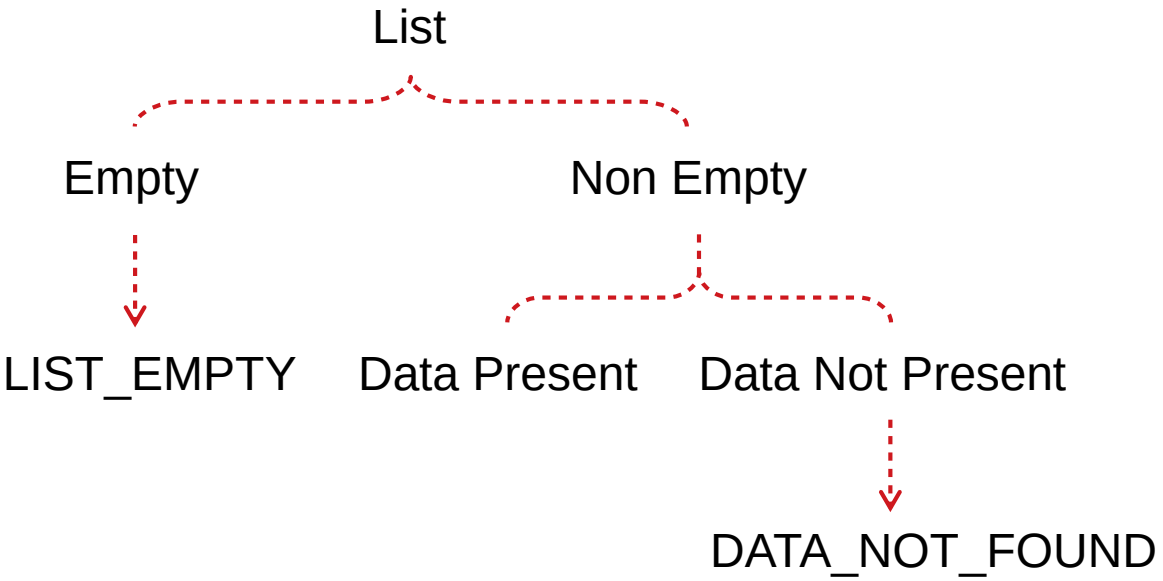


Cases :


List
Empty
Non Empty
LIST_EMPTY
Data Present
Data Not Present

Cases :

Analysis
insert_after



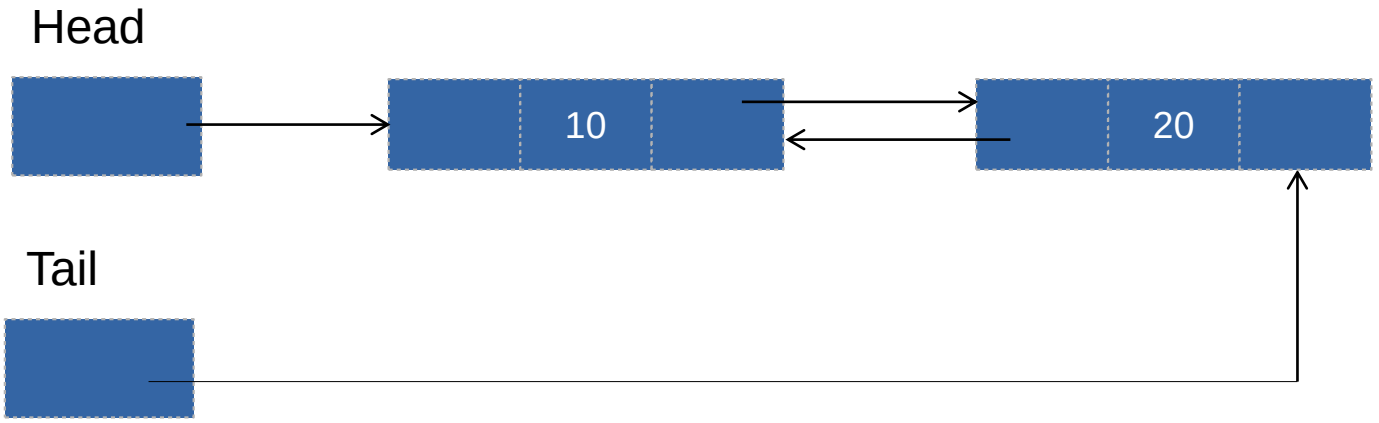
EMERTXE

Analysis

insert_after

Analysis
insert_after

Non Empty List

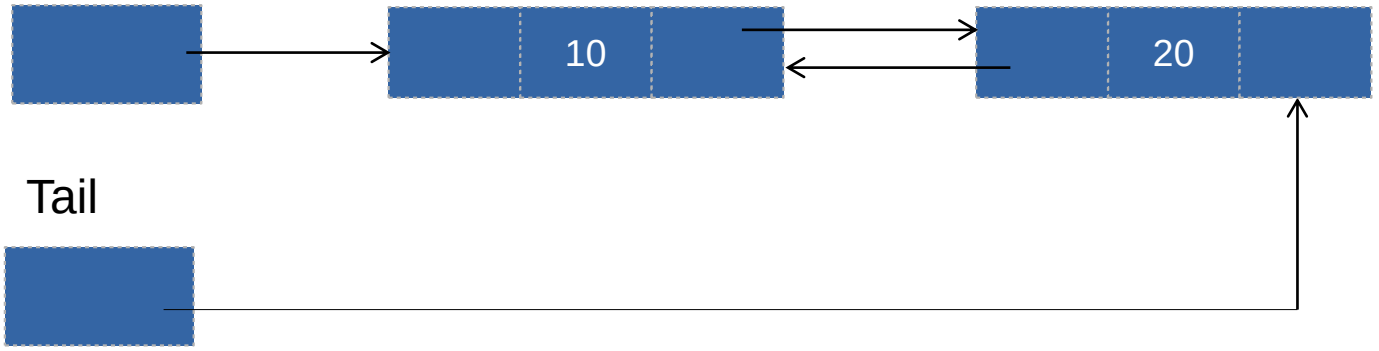


Analysis
insert_after

Non Empty List

g_data = 40 n_data = 25

Head



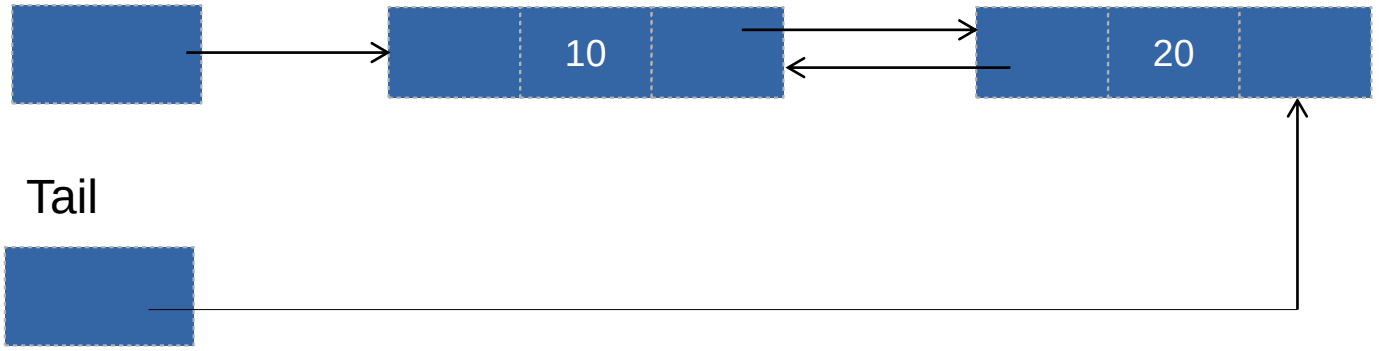
EMERTXE

Analysis
insert_after

Non Empty List

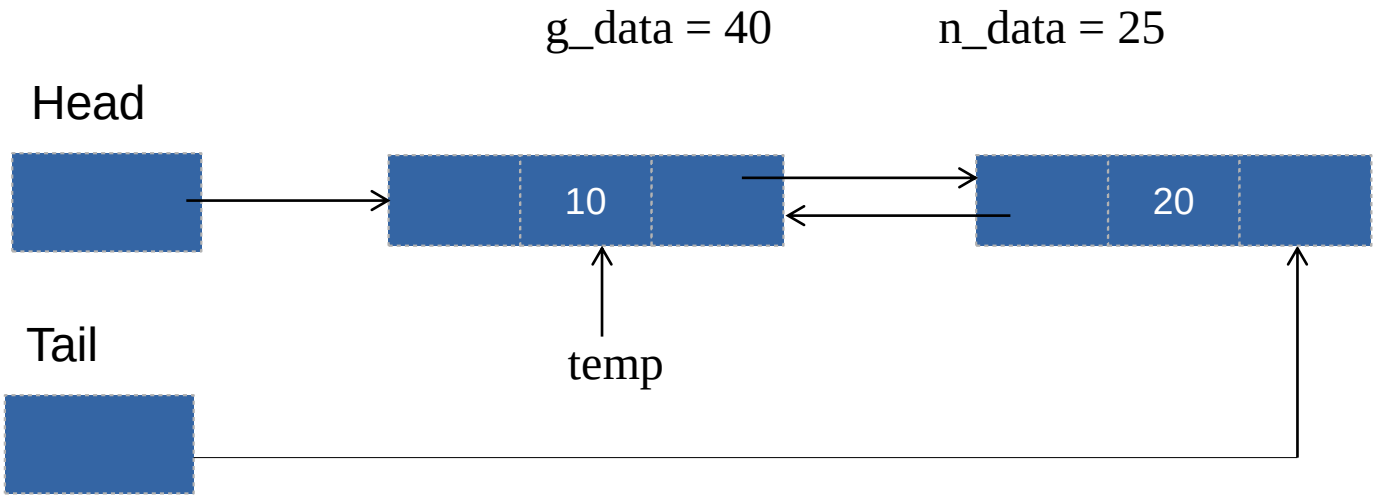
g_data = 40 n_data = 25

Head



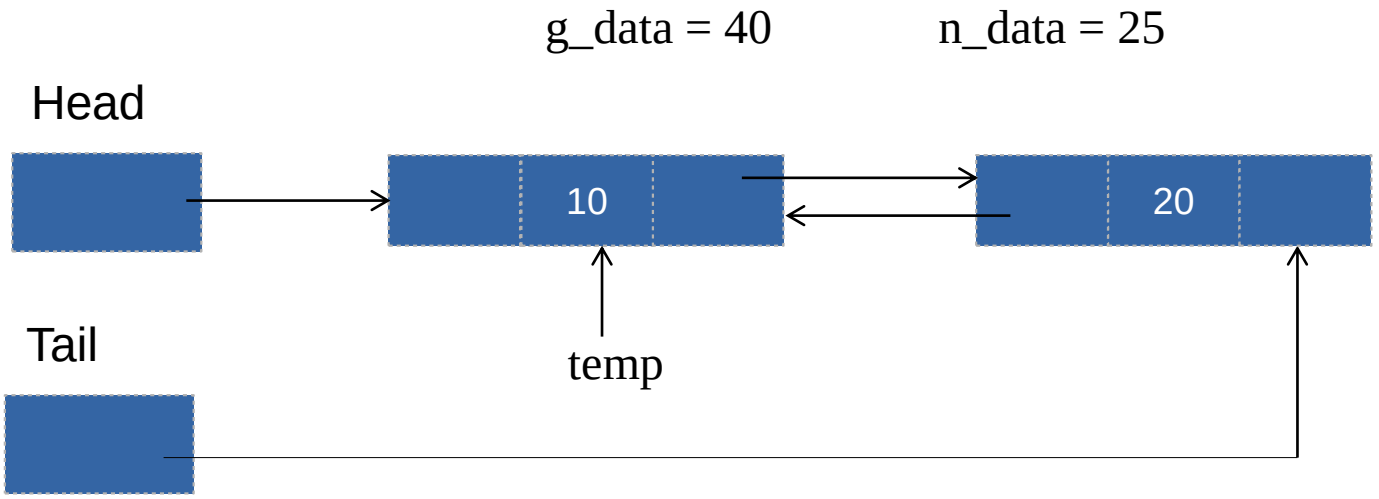
Analysis
insert_after

Non Empty List



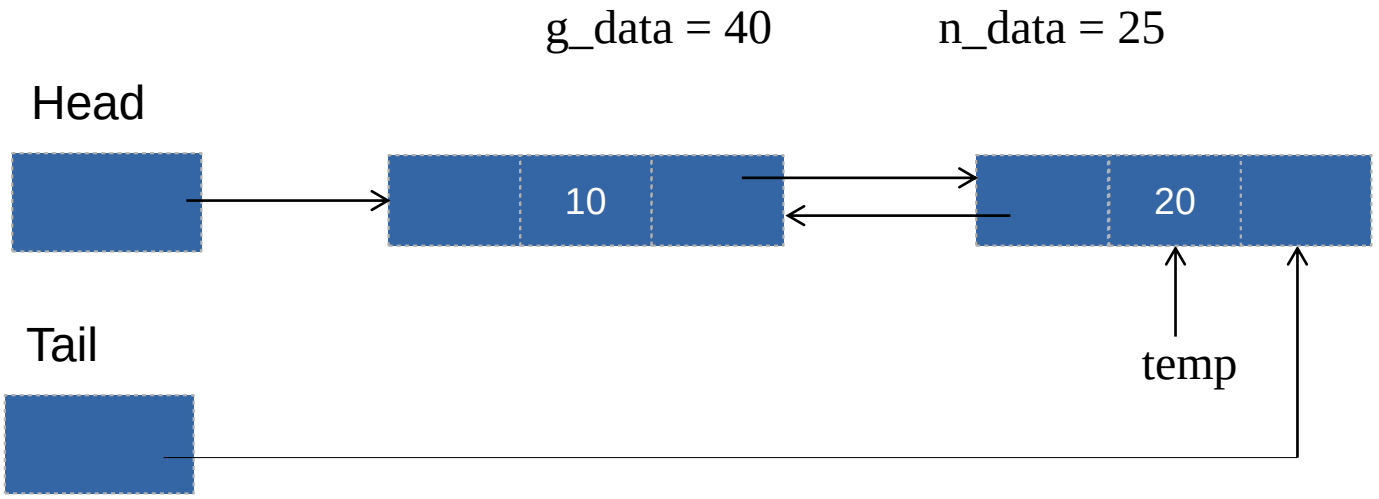
Analysis
insert_after

Non Empty List



Analysis
insert_after

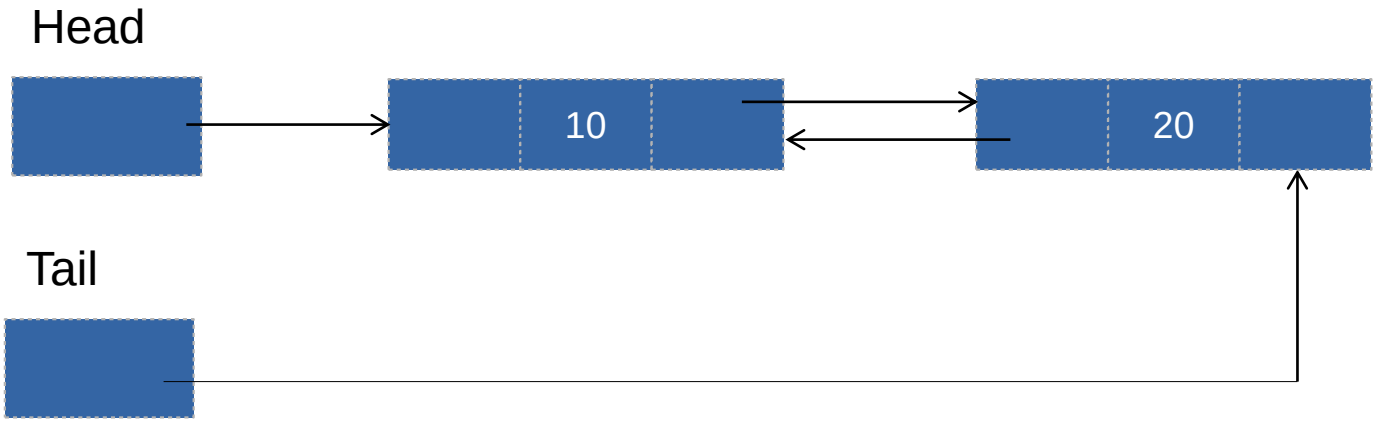
Non Empty List



Analysis
insert_after

Non Empty List

g_data = 40 n_data = 25



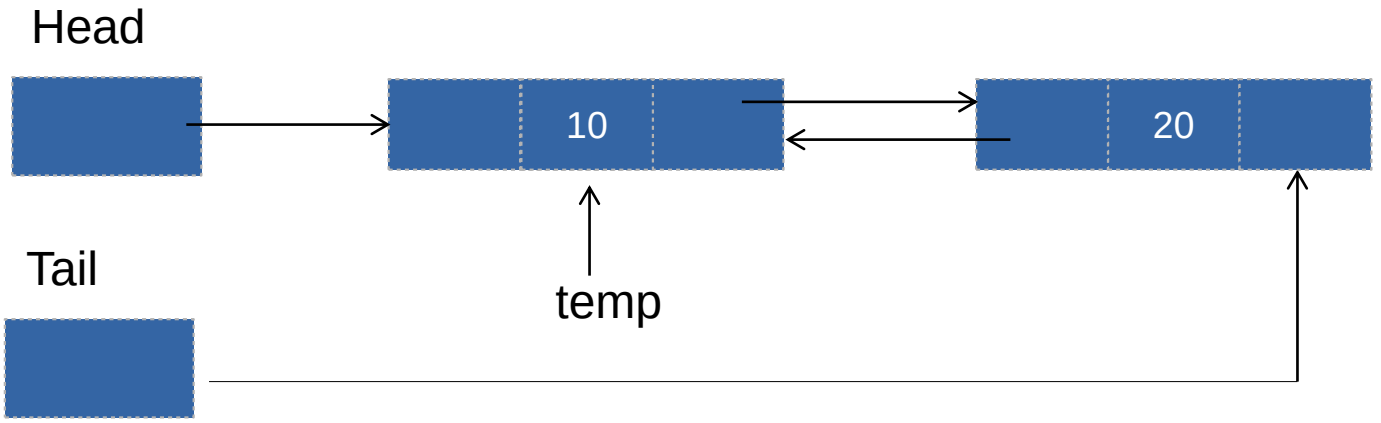
temp = NULL

EMERTXE

Analysis
insert_after

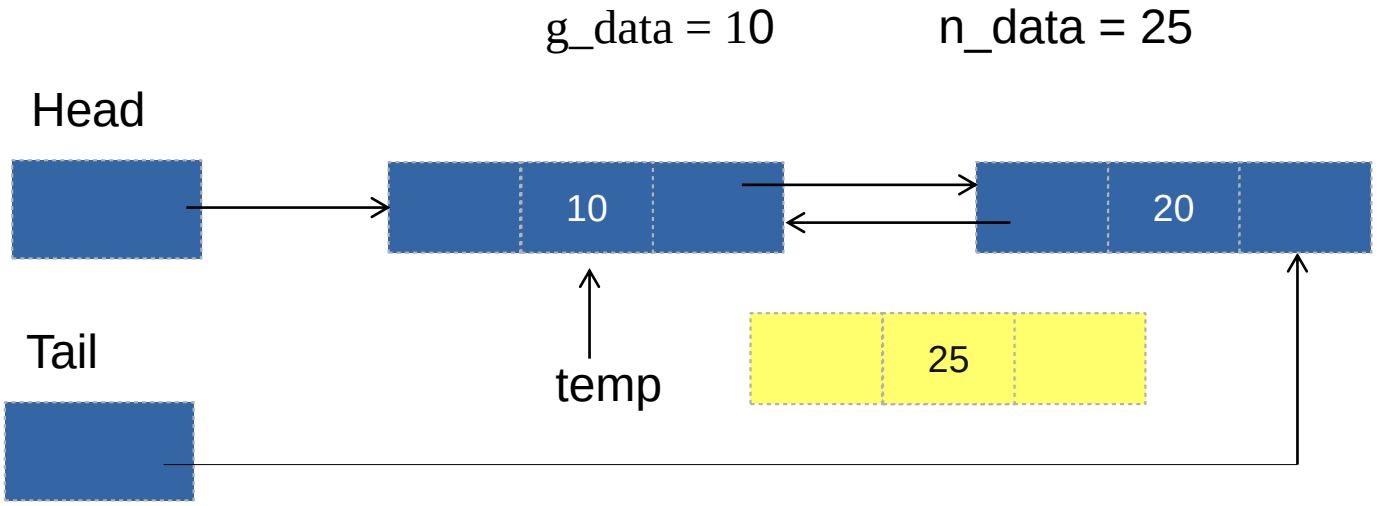
Non Empty List

g_data = 10 n_data = 25



Analysis
insert_after

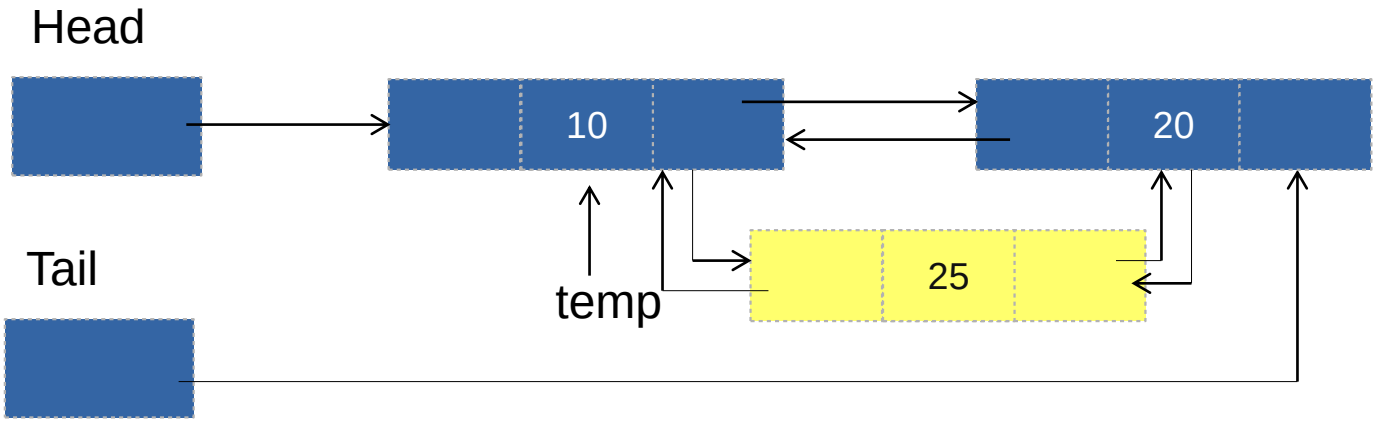
Non Empty List



Analysis
insert_after

Non Empty List

g_data = 10 n_data = 25



Insert after - Flowchart

Flowchart

insert_after

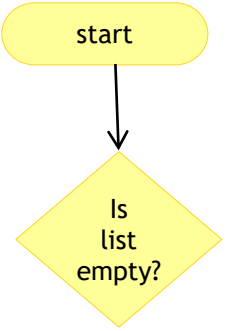
start

Flowchart

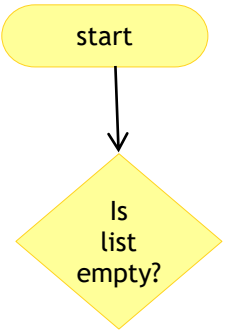
insert_after

Flowchart

insert_after



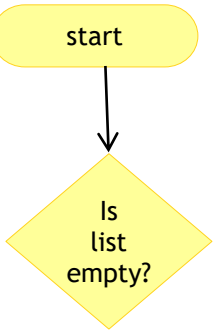
Flowchart
insert_after



Head

EMERTXE

Flowchart
insert_after

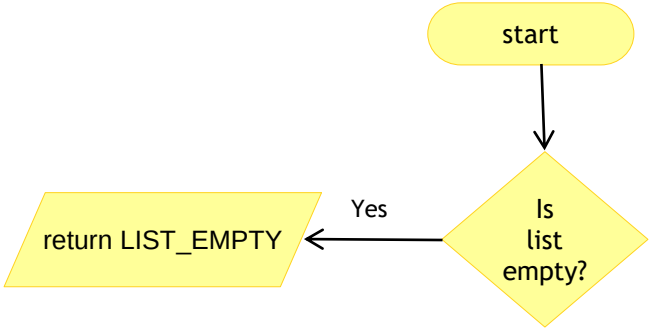


Head

NULL

EMERTXE

Flowchart
insert_after



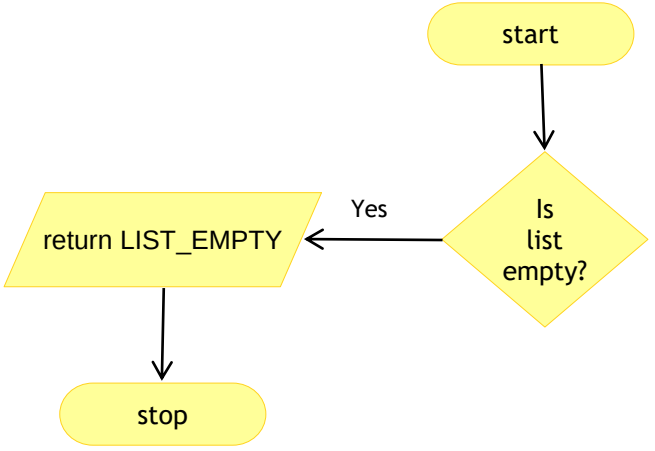
Head

NULL

Flowchart
insert_after

Head

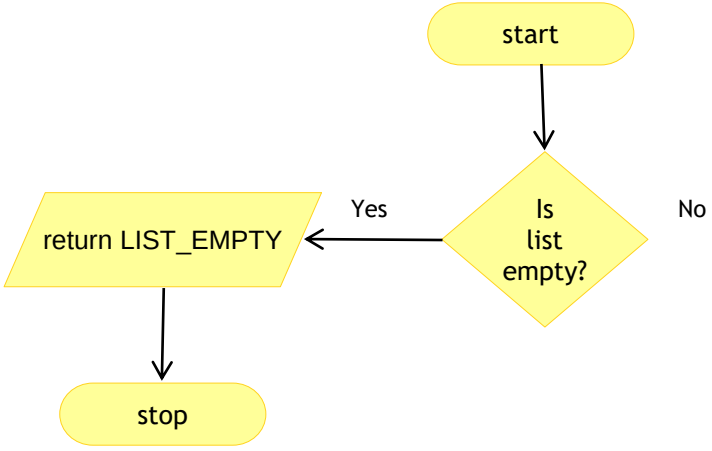
NULL



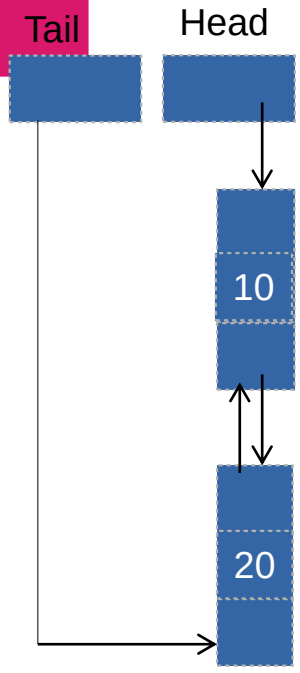
Flowchart
insert_after

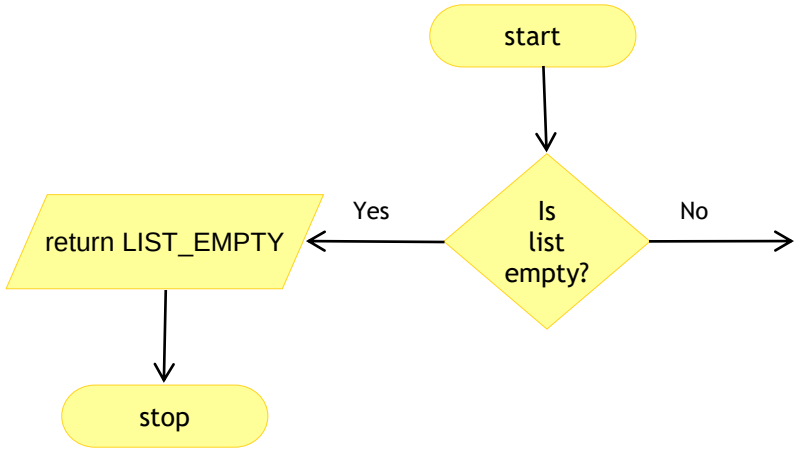
Head

NULL



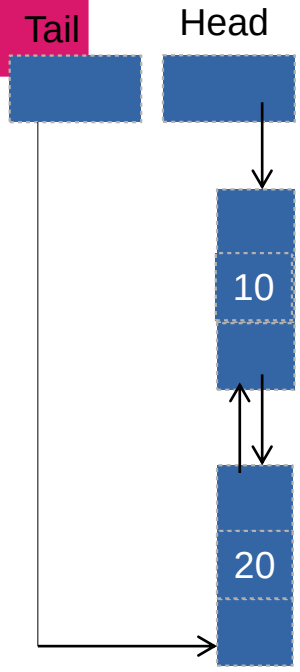
Flowchart
insert_after





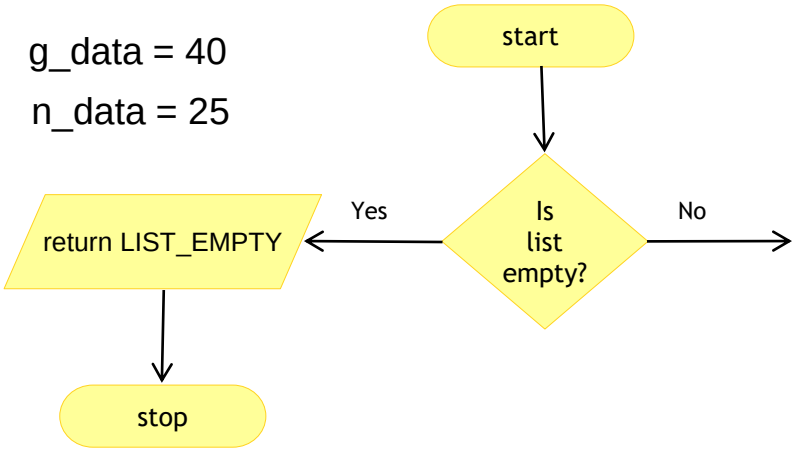
EMERTXE

Flowchart
insert_after

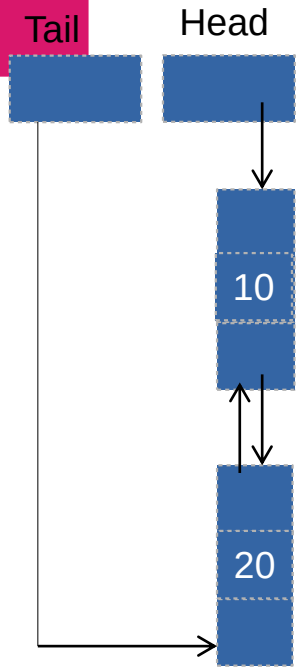


g_data = 40

n_data = 25

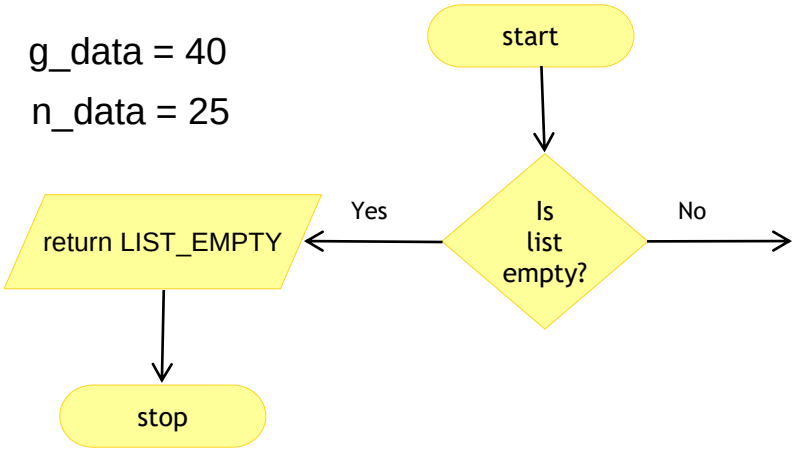


Flowchart
insert_after

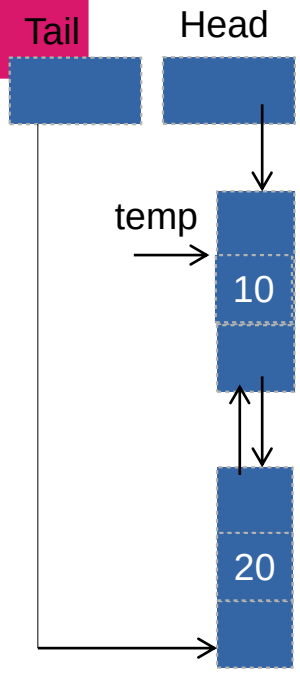


g_data = 40

n_data = 25

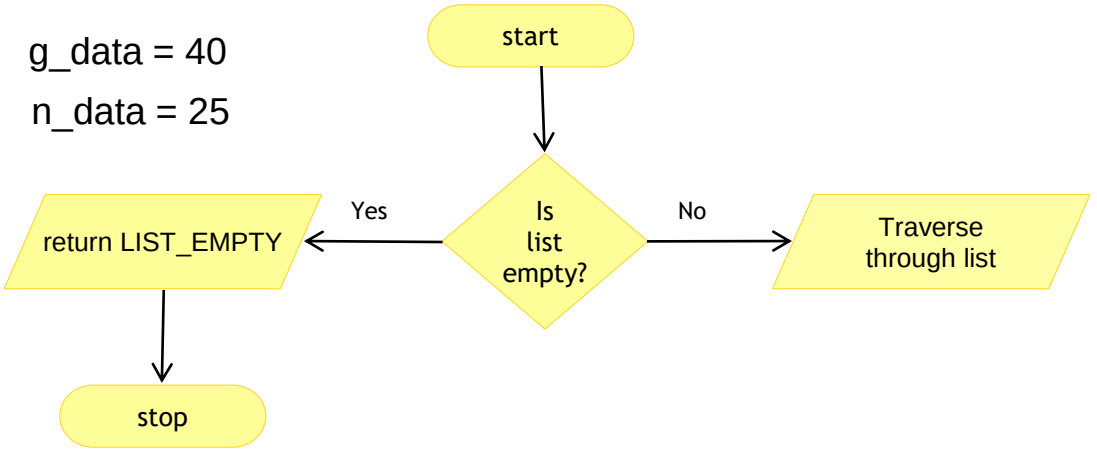


Flowchart
insert_after

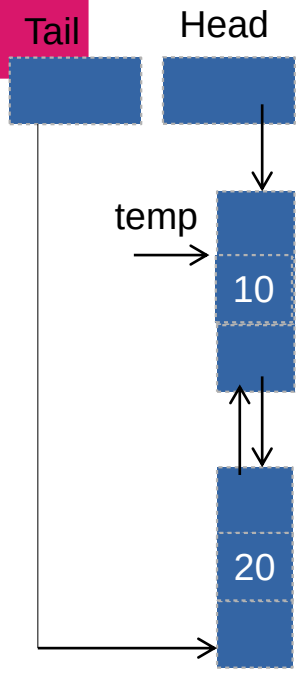


g_data = 40

n_data = 25

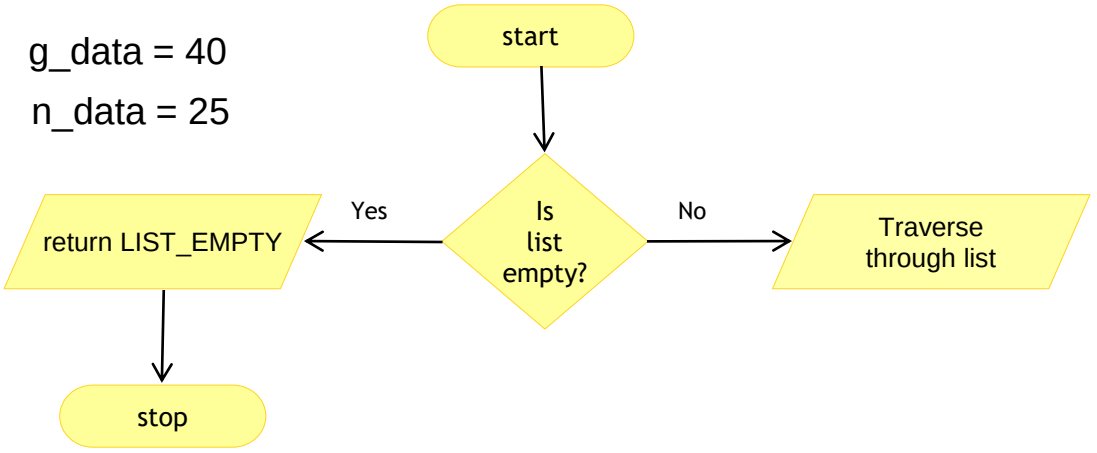


Flowchart
insert_after

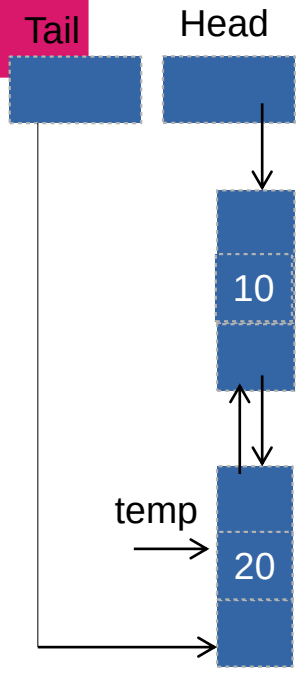


g_data = 40

n_data = 25

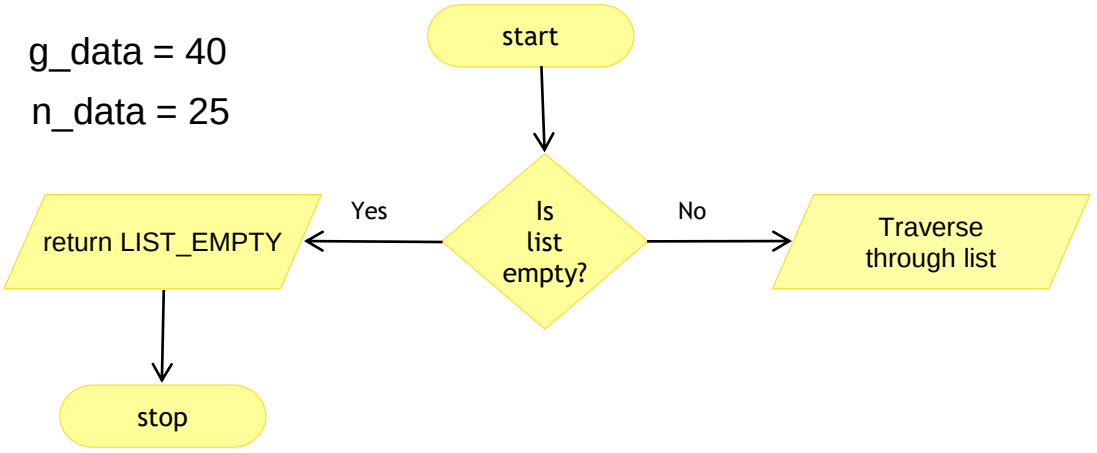


Flowchart
insert_after

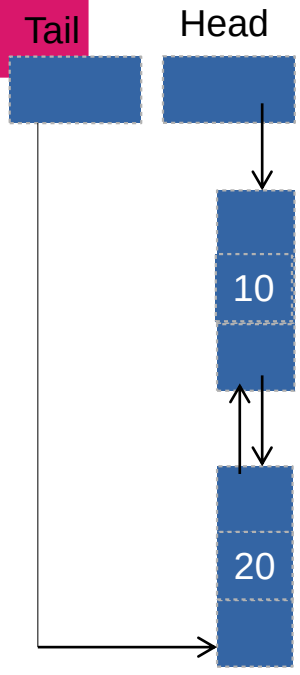


g_data = 40

n_data = 25



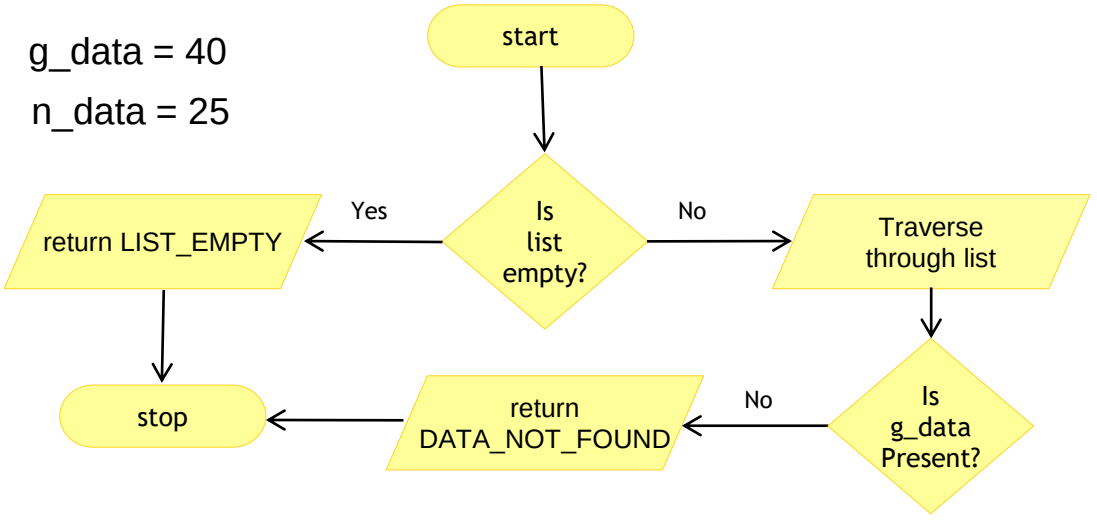
Flowchart
insert_after



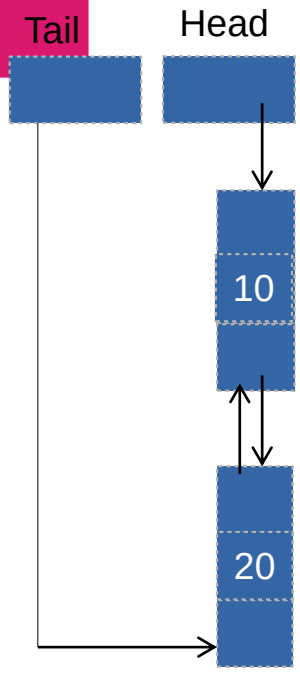
temp = NULL

g_data = 40

n_data = 25

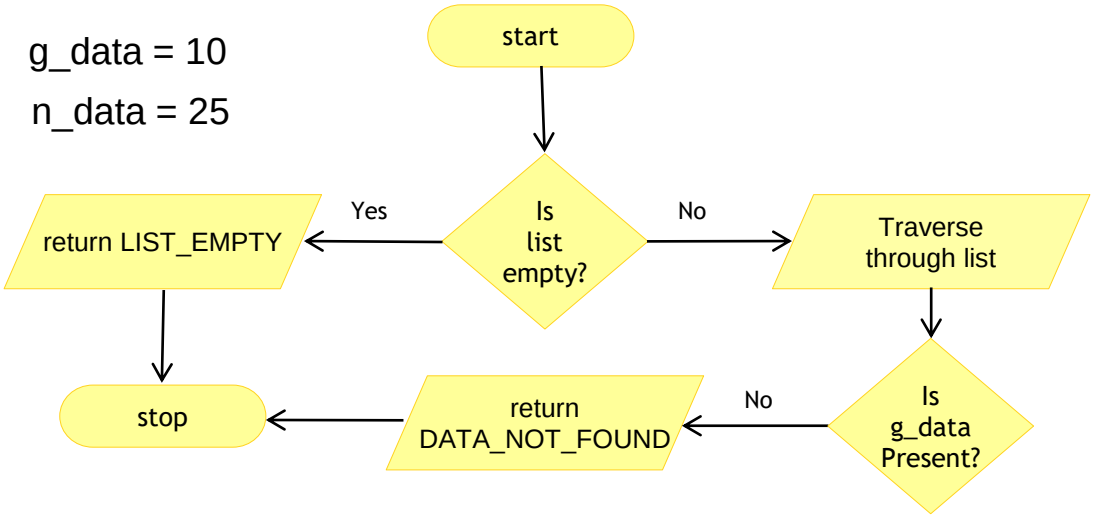


Flowchart
insert_after

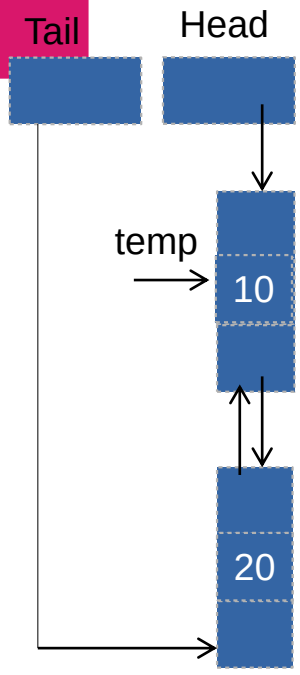


g_data = 10

n_data = 25

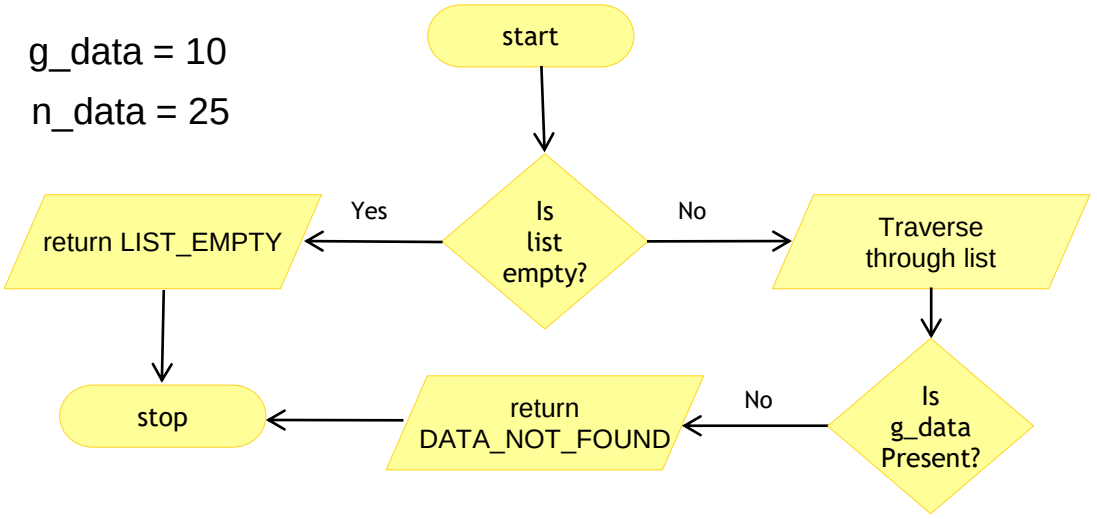


Flowchart
insert_after

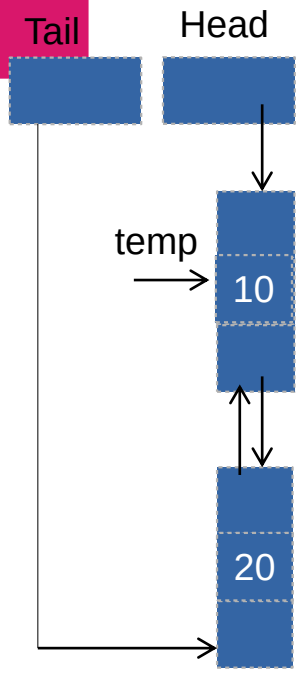


g_data = 10

n_data = 25

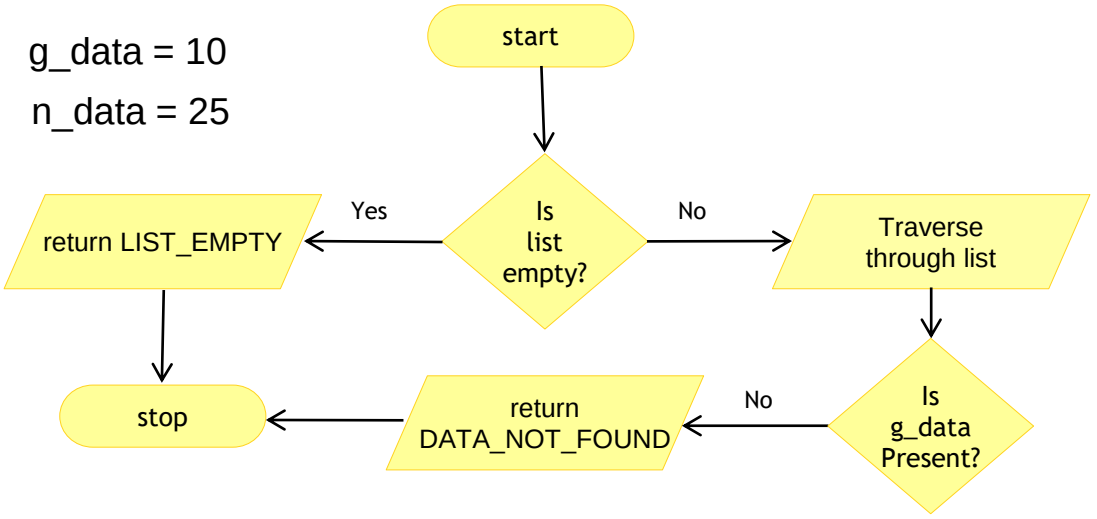


Flowchart
insert_after

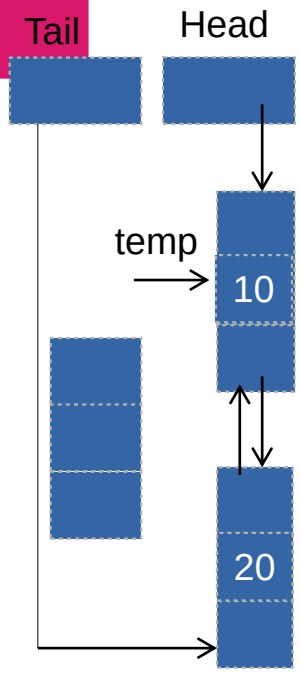


g_data = 10

n_data = 25

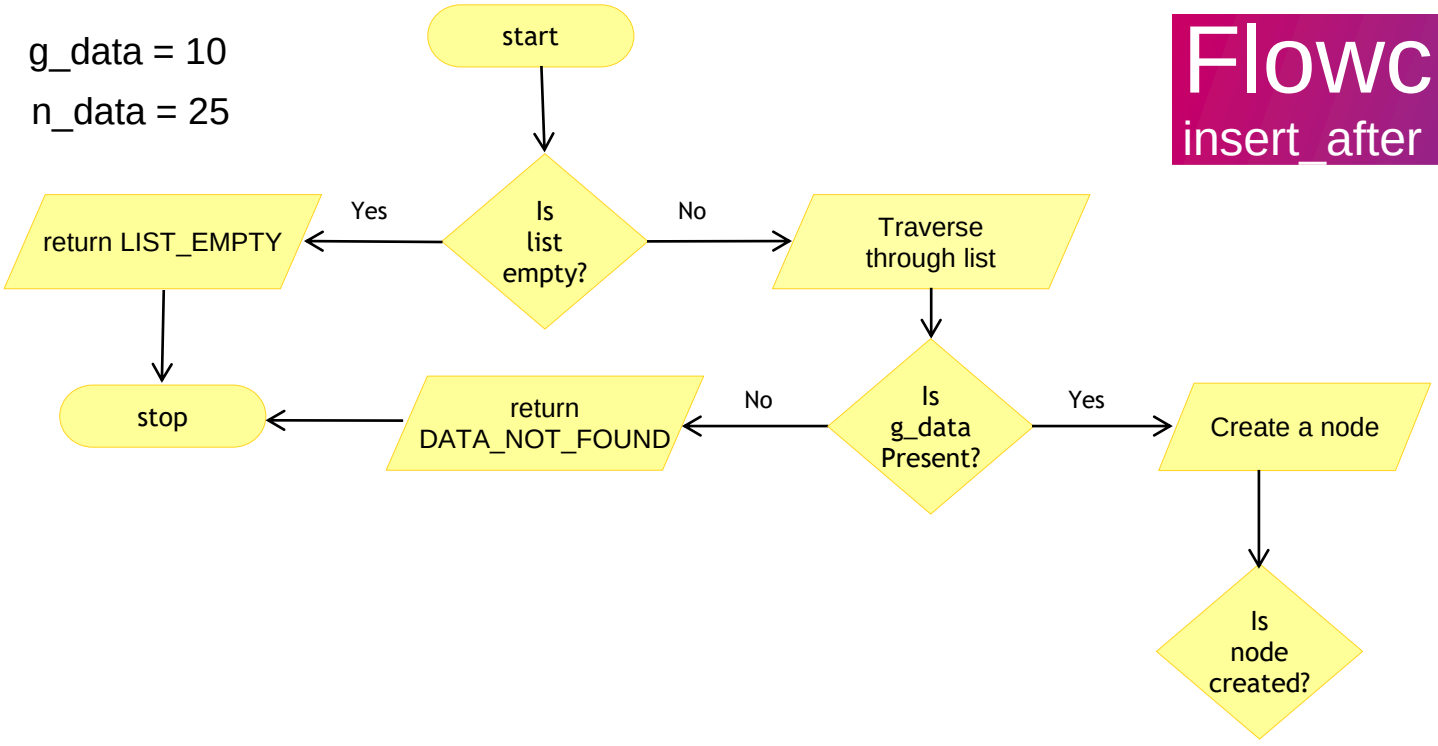


Flowchart
insert_after

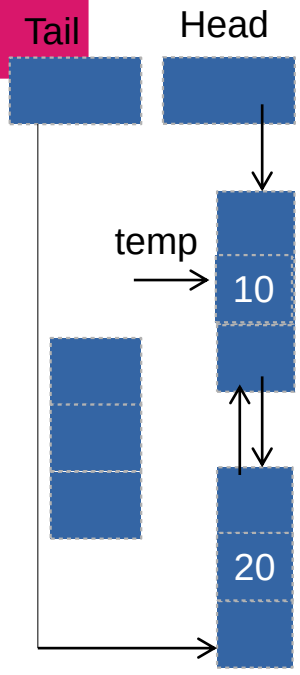


g_data = 10

n_data = 25

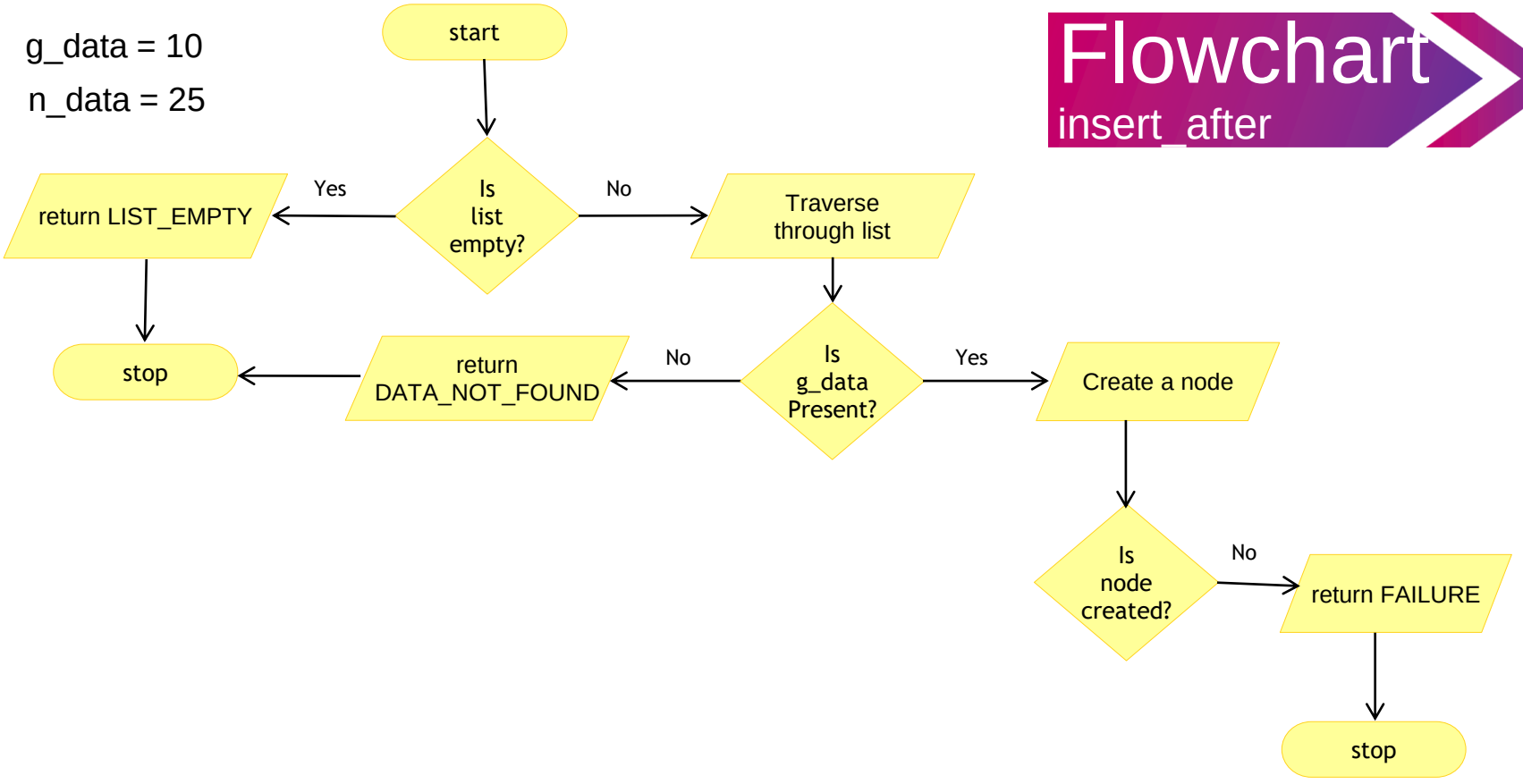


Flowchart
insert_after

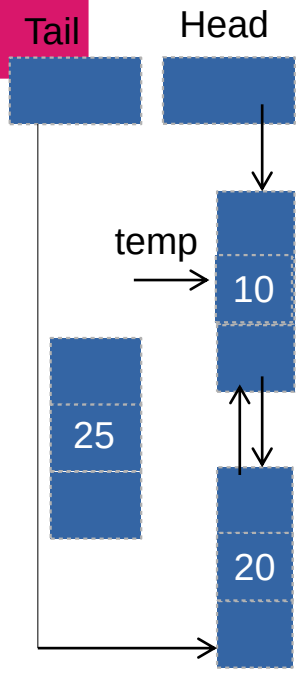


g_data = 10

n_data = 25

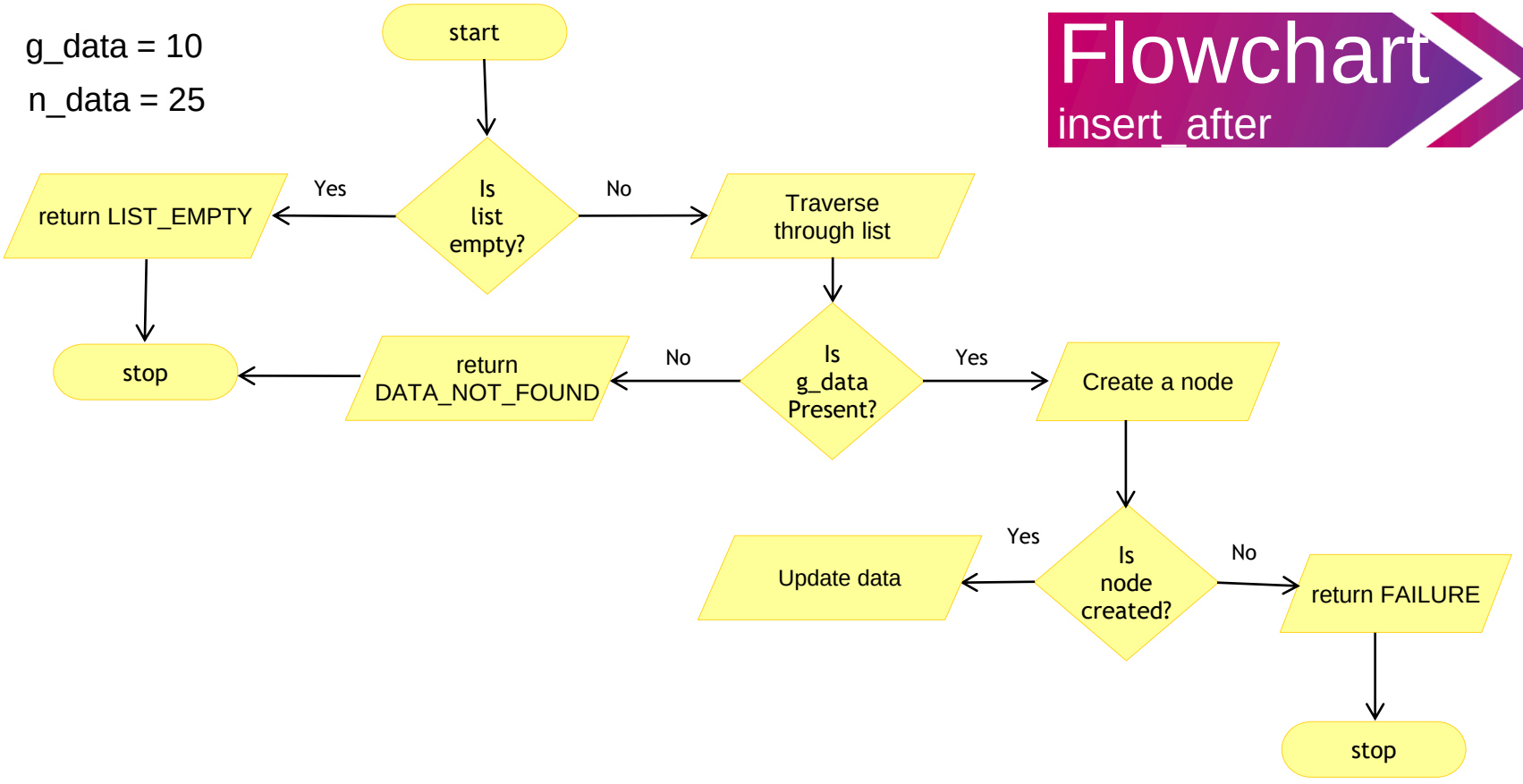


Flowchart
insert_after

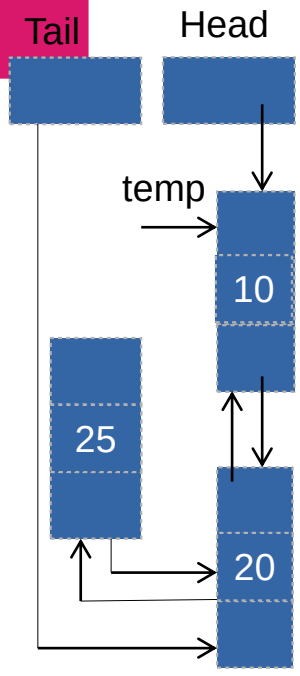


g_data = 10

n_data = 25

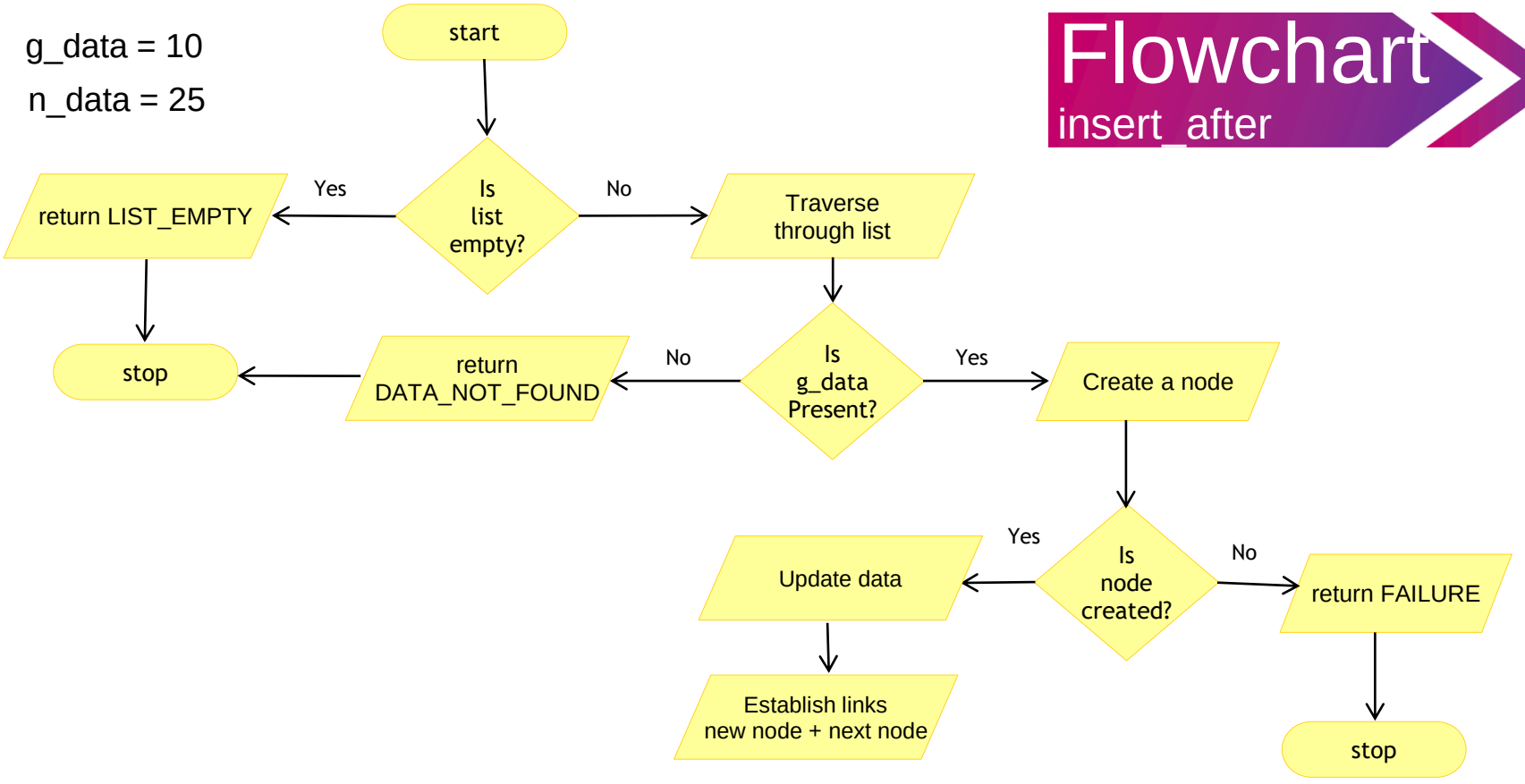


Flowchart
insert_after

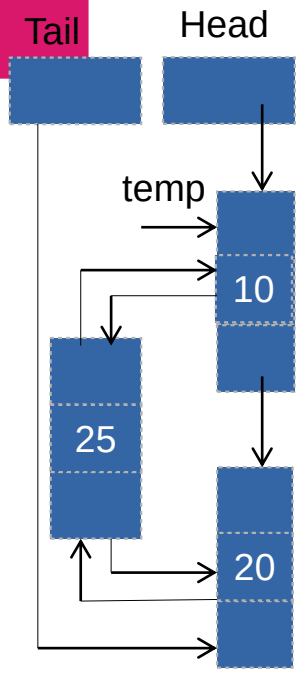


g_data = 10

n_data = 25

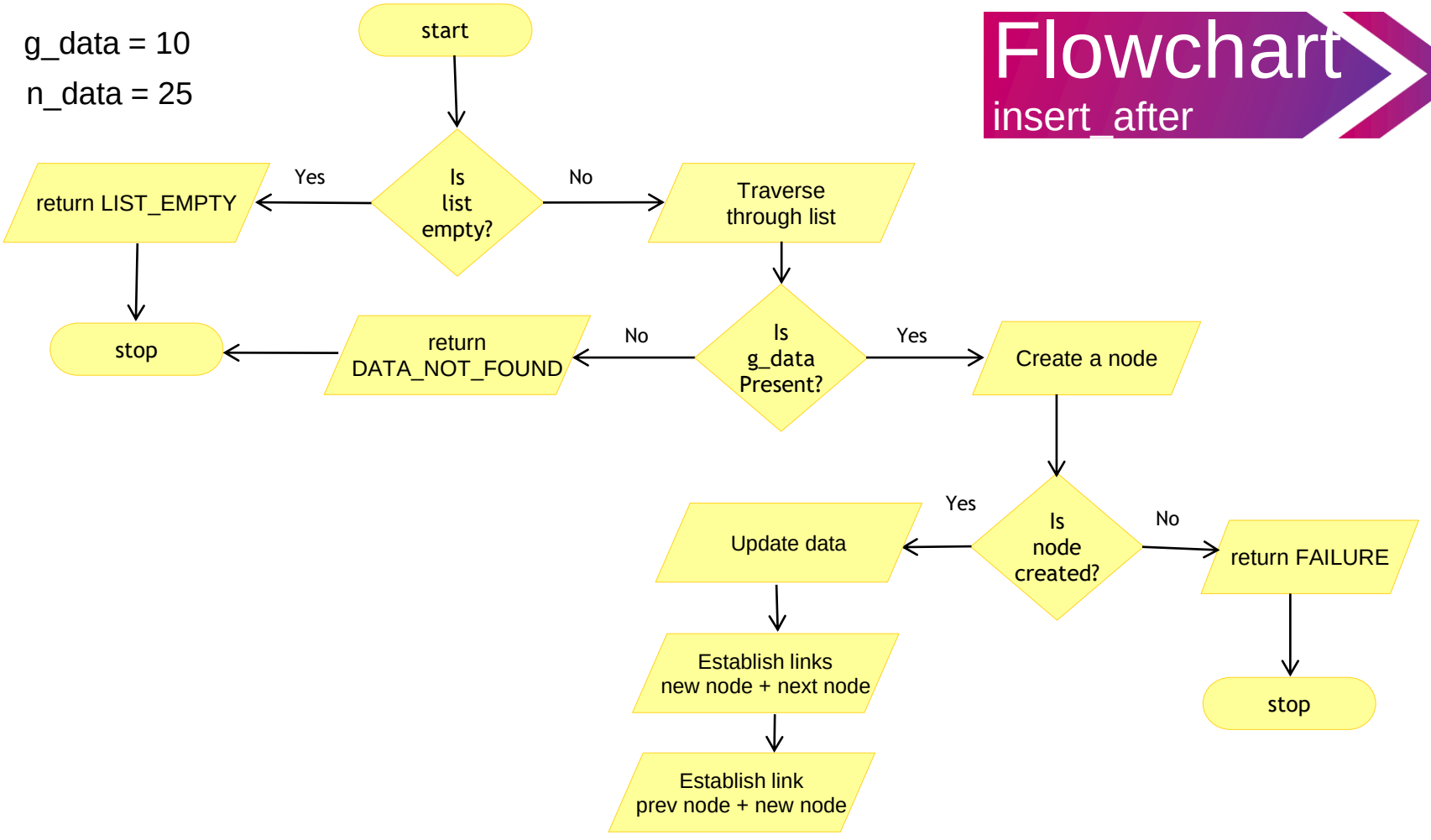


Flowchart
insert_after

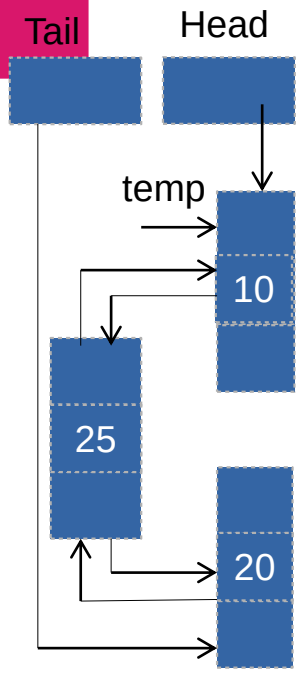


g_data = 10

n_data = 25

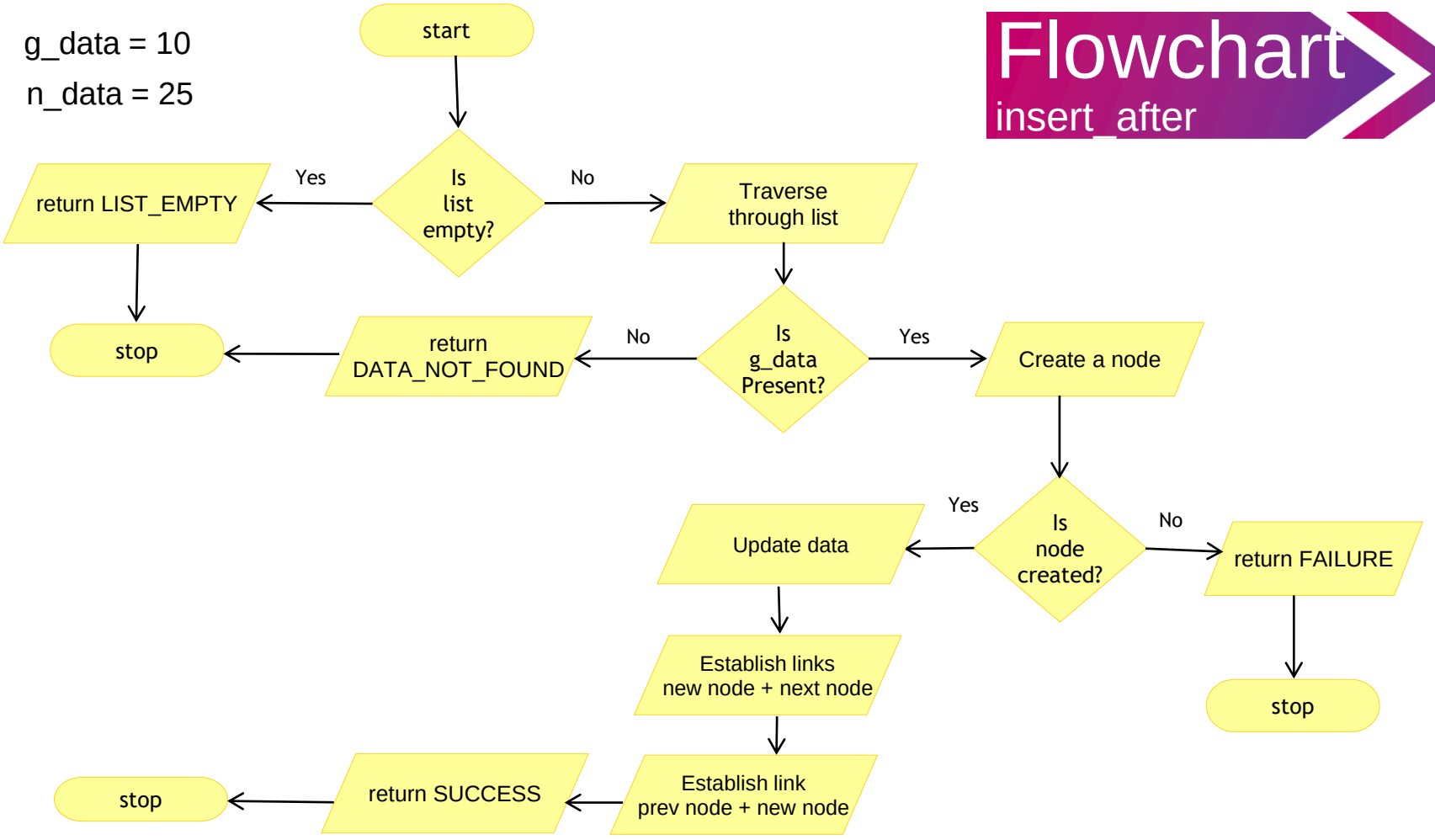


Flowchart
insert_after



g_data = 10

n_data = 25



EMERTXE

Insert after - Algorithm

Algorithm
insert_after

Algorithm
insert_after

Algorithm : Insert_after(Head,Tail,g_data,n_data)

1.Input Specification :-

Algorithm
insert_after

Algorithm : Insert_after(Head,Tail,g_data,n_data)

1.Input Specification :-

Head : Pointer containing first node address

Tail : Pointer containing last node address

g_data : Item after which we wish to insert the n_data

Algorithm
insert_after

Algorithm : Insert_after(Head,Tail,g_data,n_data)

1.Input Specification :-

Head : Pointer containing first node address

Tail : Pointer containing last node address

g_data : Item after which we wish to insert the n_data

n_data : Item to be added

Algorithm
insert_after

Algorithm : Insert_after(Head,Tail,g_data,n_data)

1.Input Specification :-

Head : Pointer containing first node address

Tail : Pointer containing last node address

g_data : Item after which we wish to insert the n_data

n_data : Item to be added

2.Output Specification :-

Algorithm
insert_after

Algorithm : Insert_after(Head,Tail,g_data,n_data)

1.Input Specification :-

Head : Pointer containing first node address
Tail : Pointer containing last node address
g_data : Item after which we wish to insert the n_data
n_data : Item to be added

2.Output Specification :-

Status : SUCCESS / FAILURE
LIST_EMPTY / DATA_NOT_FOUND

Insert_after(Head,Tail,g_data,n_data)

Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

Algorithm
insert_after

g_data = 40

n_data = 25

EMERTXE

Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)

g_data = 40

n_data = 25

Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)

g_data = 40

n_data = 25

Head

NULL

Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY

g_data = 40

n_data = 25

Head

NULL

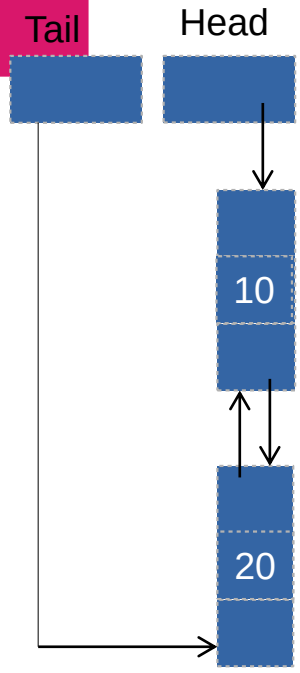
Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY

g_data = 40

n_data = 25



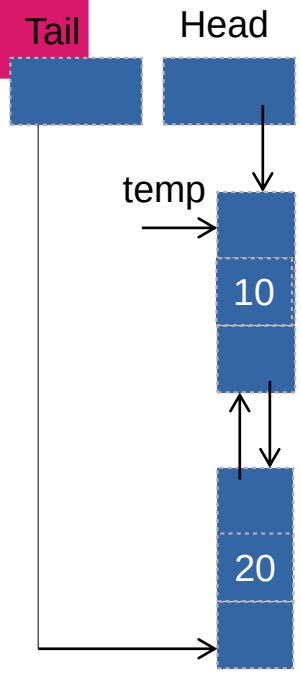
Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY

Algorithm
insert_after

g_data = 40

n_data = 25



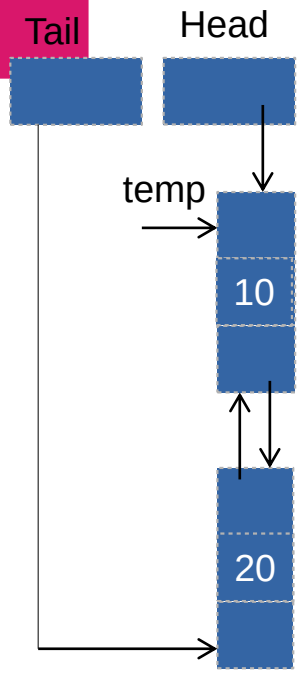
Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head

Algorithm
insert_after

g_data = 40

n_data = 25



Algorithm
insert_after

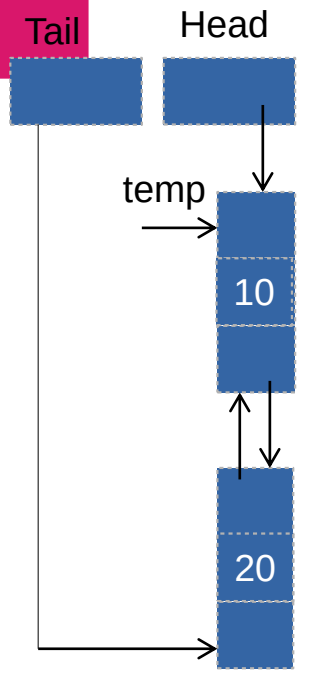
Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)

g_data = 40

n_data = 25

EMERTXE



Algorithm
insert_after

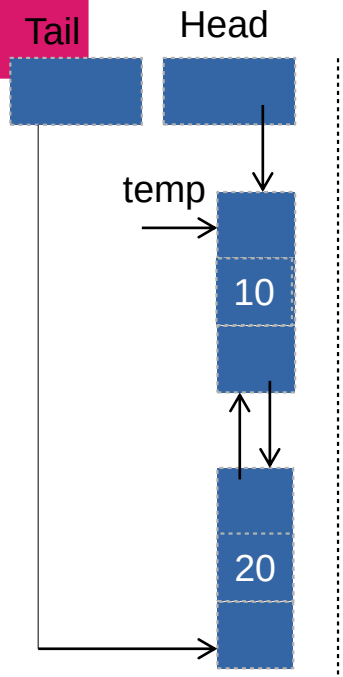
Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)

g_data = 40

n_data = 25

EMERTXE



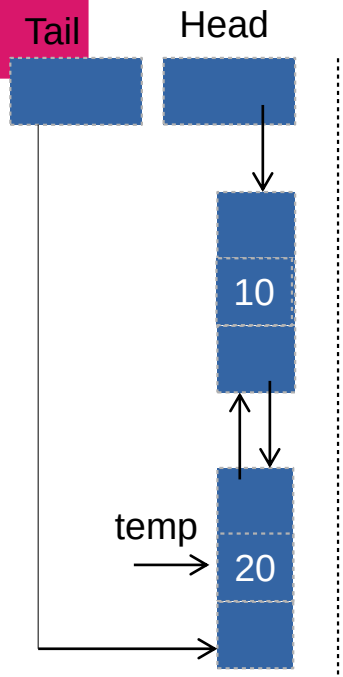
Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next

Algorithm
insert_after

g_data = 40

n_data = 25



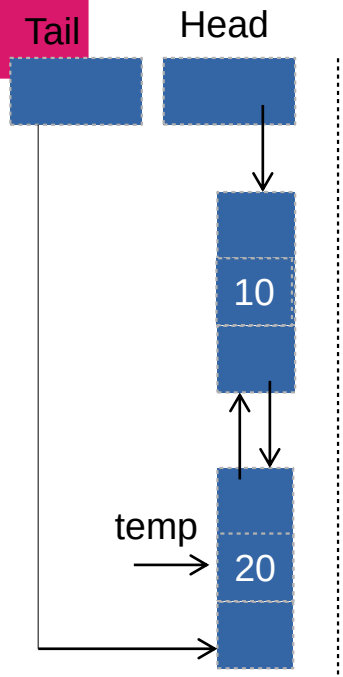
Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next

Algorithm
insert_after

g_data = 40

n_data = 25



Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

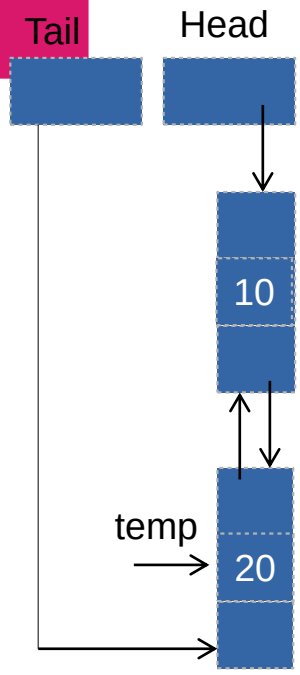
1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp⟶ data != g_data)
   temp = temp ⟶next

g_data = 40

n_data = 25

EMERTXE

Algorithm
insert_after



temp = NULL

Insert_after(Head,Tail,g_data,n_data)

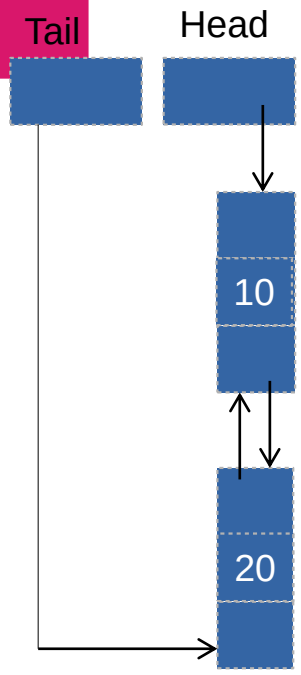
1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next

g_data = 40

n_data = 25

EMERTXE

Algorithm
insert_after



temp = NULL

Insert_after(Head,Tail,g_data,n_data)

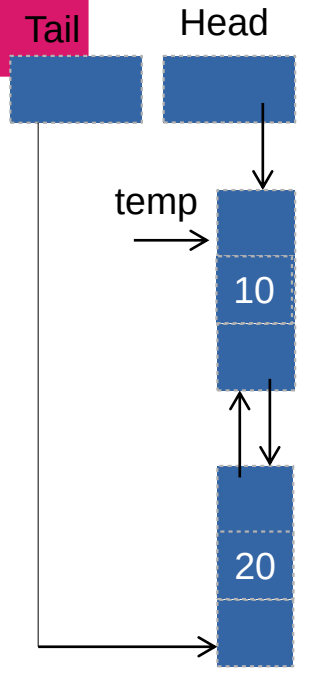
1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next

g_data = 40

n_data = 25

4. return DATA_NOT_FOUND

EMERTXE



Algorithm
insert_after

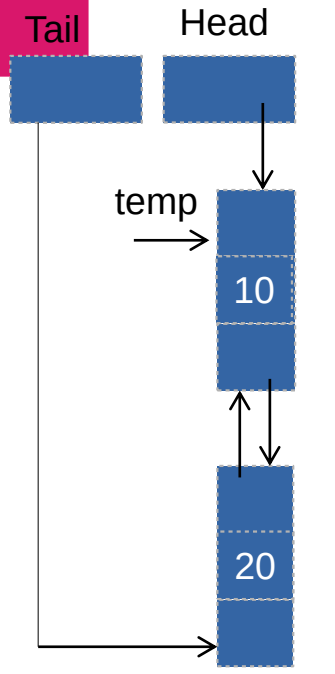
Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp → data != g_data)
   temp = temp → next

g_data = 10

n_data = 25

EMERTXE



Algorithm
insert_after

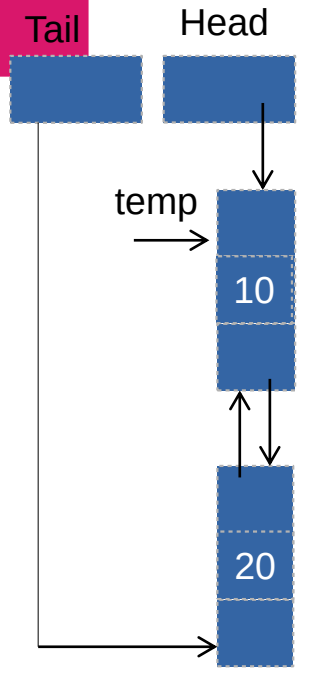
Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp → data != g_data)
   temp = temp → next

g_data = 10

n_data = 25

EMERTXE



Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

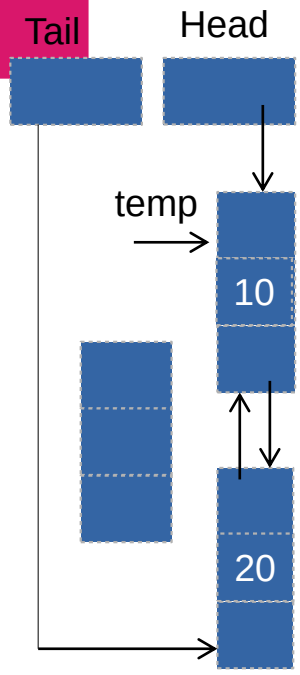
1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp → data != g_data)
   temp = temp → next

g_data = 10

n_data = 25

EMERTXE

Algorithm
insert_after



Insert_after(Head,Tail,g_data,n_data)

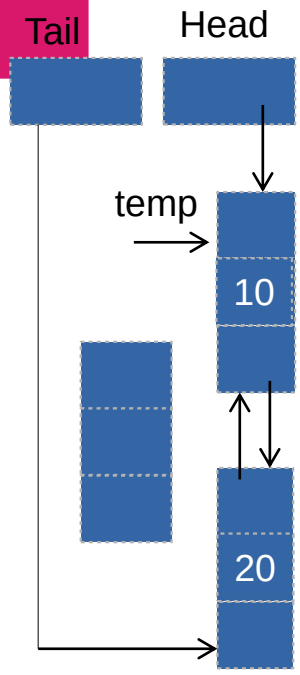
1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))

g_data = 10

n_data = 25

EMERTXE

Algorithm
insert_after



Insert_after(Head,Tail,g_data,n_data)

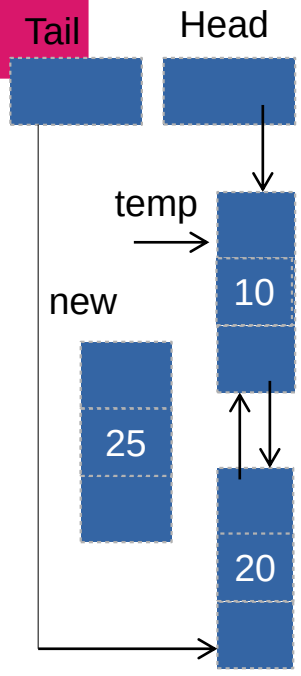
1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE

g_data = 10

n_data = 25

EMERTXE

Algorithm
insert_after



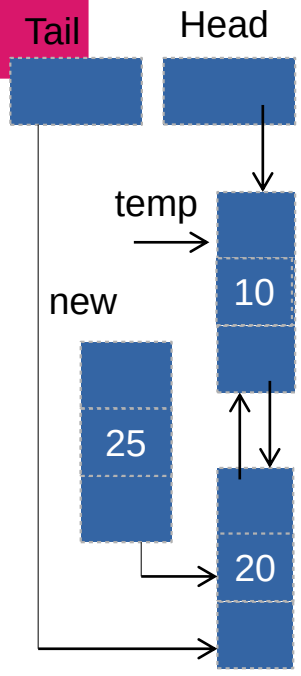
Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp → data != g_data)
   temp = temp → next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new → data = n_data

g_data = 10

n_data = 25

EMERTXE



Algorithm
insert_after

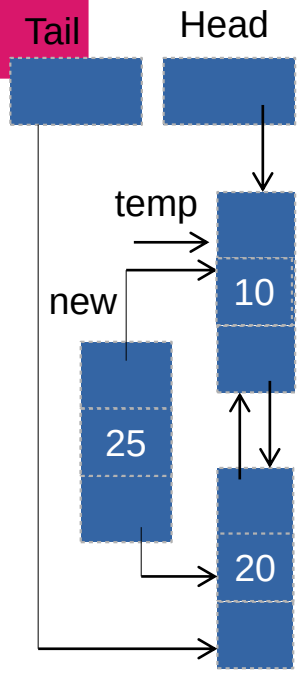
Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp → data != g_data)
   temp = temp → next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new → data = n_data
   new → next = temp → next

g_data = 10

n_data = 25

EMERTXE



Algorithm
insert_after

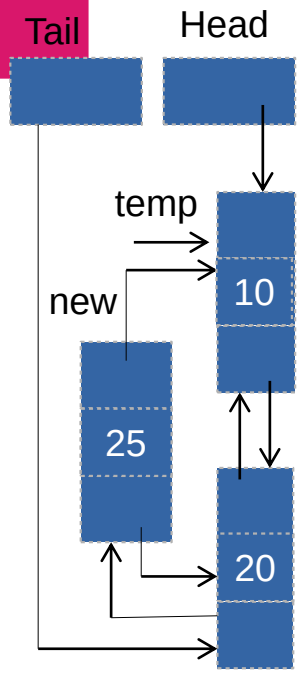
Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new ⟶ data = n_data
   new ⟶ next = temp ⟶ next
   new ⟶ prev = temp

g_data = 10

n_data = 25

EMERTXE



Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new ⟶ data = n_data
   new ⟶ next = temp ⟶ next
   new ⟶ prev = temp

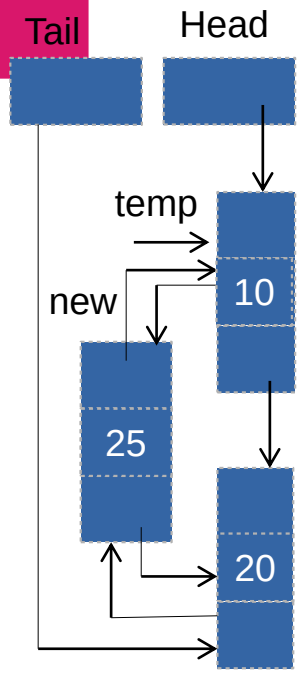
   temp ⟶ next ⟶ prev = new

g_data = 10

n_data = 25

EMERTXE

Algorithm
insert_after



Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new ⟶ data = n_data
   new ⟶ next = temp ⟶ ~~next~~
   new ⟶ prev = temp

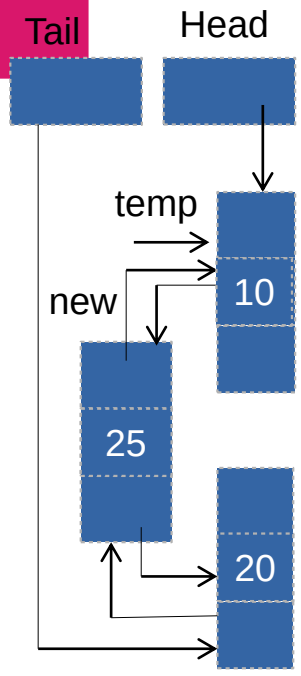
   temp ⟶ next ⟶ prev = new

   temp ⟶ next = new

g_data = 10

n_data = 25

EMERTXE



Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new ⟶ data = n_data
   new ⟶ next = temp ⟶ next
   new ⟶ prev = temp

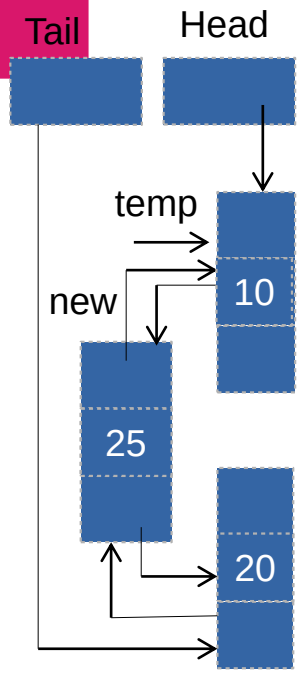
   temp ⟶ next ⟶ prev = new

   temp ⟶ next = new
   return SUCCESS

g_data = 10

n_data = 25

EMERTXE



Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new ⟶ data = n_data
   new ⟶ next = temp ⟶ next
   new ⟶ prev = temp

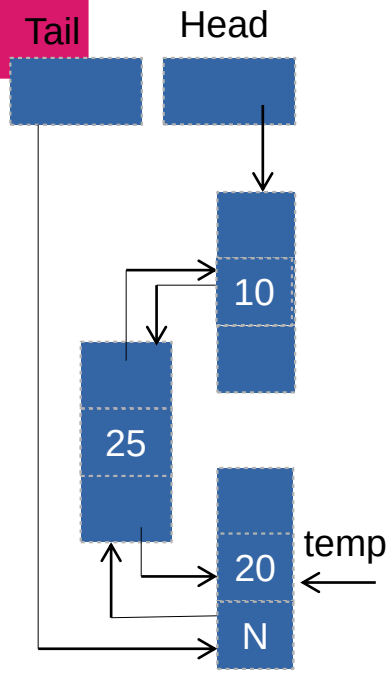
   temp ⟶ next ⟶ prev = new

   temp ⟶ next = new
   return SUCCESS

g_data = 20

n_data = 50

EMERTXE



Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new ⟶ data = n_data
   new ⟶ next = temp ⟶ next
   new ⟶ prev = temp

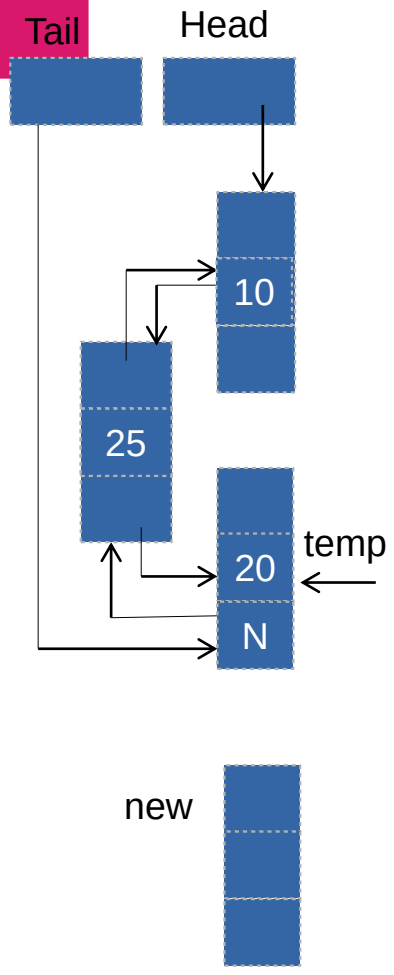
   temp ⟶ next ⟶ prev = new

   temp ⟶ next = new
   return SUCCESS

g_data = 20

n_data = 50

EMERTXE



Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new ⟶ data = n_data
   new ⟶ next = temp ⟶ next
   new ⟶ prev = temp

   temp ⟶ next ⟶ prev = new

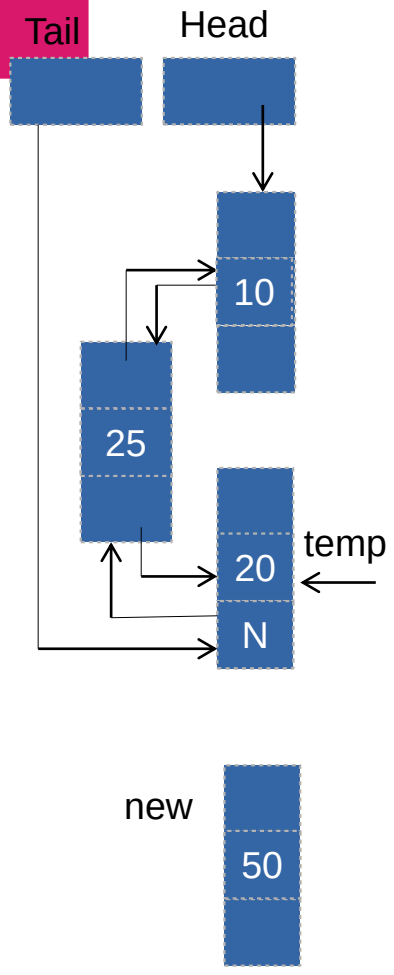
   temp ⟶ next = new
   return SUCCESS

g_data = 20

n_data = 50

EMERTXE

Algorithm
insert_after



Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new ⟶ data = n_data
   new ⟶ next = temp ⟶ next
   new ⟶ prev = temp

   temp ⟶ next ⟶ prev = new

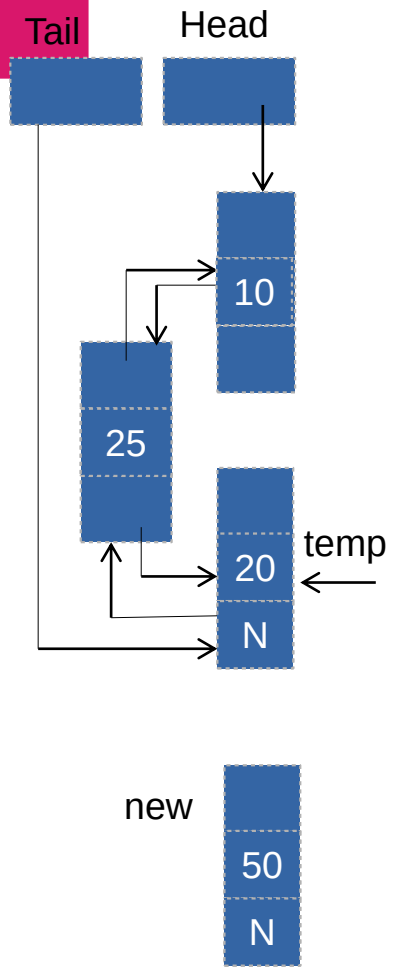
   temp ⟶ next = new
   return SUCCESS

g_data = 20

n_data = 50

EMERTXE

Algorithm
insert_after



Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new ⟶ data = n_data
   new ⟶ next = temp ⟶ next
   new ⟶ prev = temp

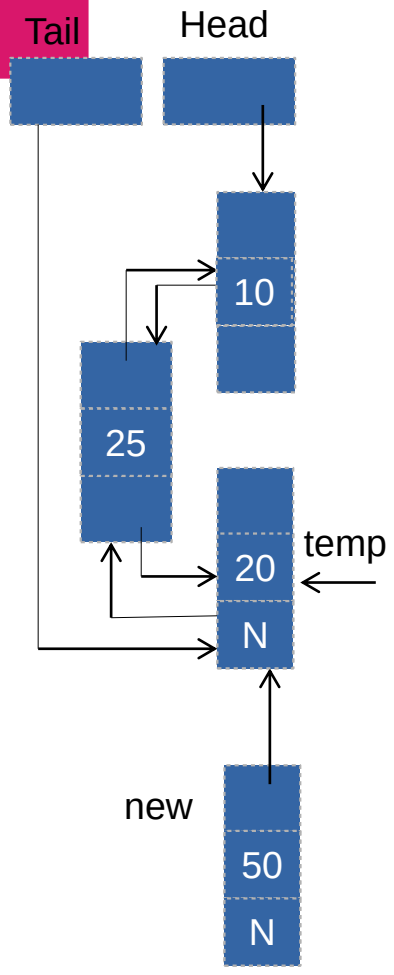
   temp ⟶ next ⟶ prev = new

   temp ⟶ next = new
   return SUCCESS

g_data = 20

n_data = 50

EMERTXE



Algorithm
insert_after

Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new ⟶ data = n_data
   new ⟶ next = temp ⟶ next
   new ⟶ prev = temp

   temp ⟶ next ⟶ prev = new

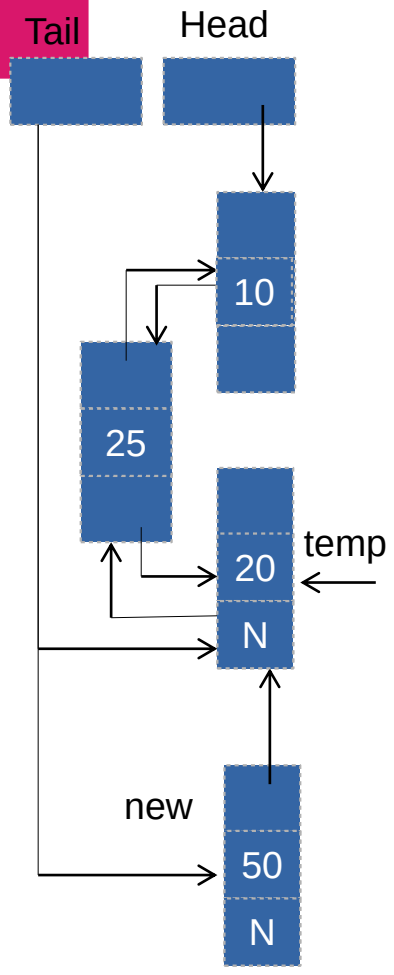
   temp ⟶ next = new
   return SUCCESS

g_data = 20

n_data = 50

EMERTXE

Algorithm
insert_after



Insert_after(Head,Tail,g_data,n_data)

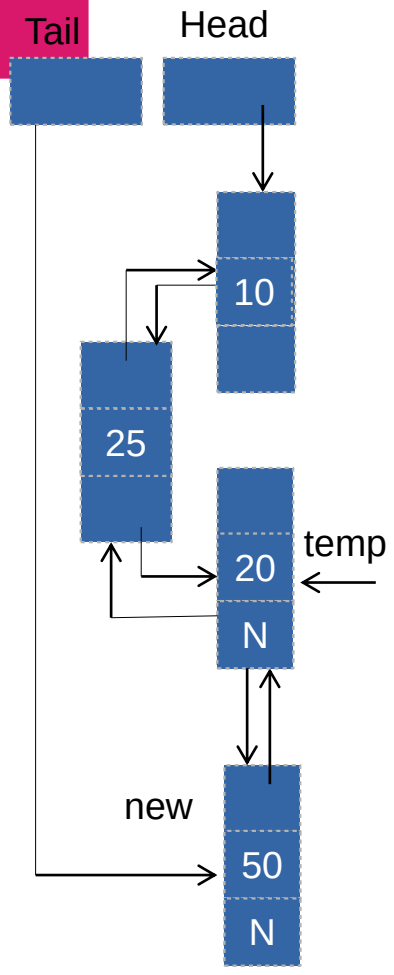
1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp ⟶ data != g_data)
   temp = temp ⟶ next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new ⟶ data = n_data
   new ⟶ next = temp ⟶ next
   new ⟶ prev = temp
   if (temp != Tail)
   temp ⟶ next ⟶ prev = new
   else
   Tail = new
   temp ⟶ next = new
   return SUCCESS

g_data = 20

n_data = 50

EMERTXE

Algorithm
insert_after



Insert_after(Head,Tail,g_data,n_data)

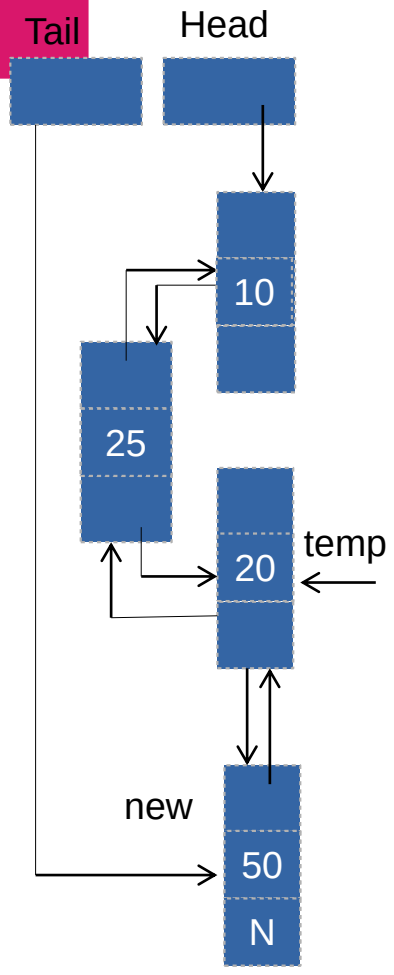
1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp → data != g_data)
   temp = temp → next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new → data = n_data
   new → next = temp → next
   new → prev = temp
   if (temp != Tail)
   temp → next → prev = new
   else
   Tail = new
   temp → next = new
   return SUCCESS

g_data = 20

n_data = 50

EMERTXE

Algorithm
insert_after



Insert_after(Head,Tail,g_data,n_data)

1. if (Head = NULL)
   return LIST_EMPTY
2. temp = Head
3. while (temp != NULL)
   3.1 if (temp → data != g_data)
   temp = temp → next
   3.2 else
   new = Memalloc(sizeof (Dlist))
   if (new = NULL)
   return FAILURE
   new → data = n_data
   new → next = temp → next
   new → prev = temp
   if (temp != Tail)
   temp → next → prev = new
   else
   Tail = new
   temp → next = new
   return SUCCESS
4. return DATA_NOT_FOUND

g_data = 20

n_data = 50

EMERTXE

Double Linked List – Insert after - Code