Data Structure Module

Team Emertxe

Single Linked List

- It is linear data structure that consist of sequence of node which are connected to each other to form a list. Node consist of 2 fields i.e. data field and link field.
- Data -> item to be added into the list.
- Link -> link to the next node.

Linked List – insert_at_last

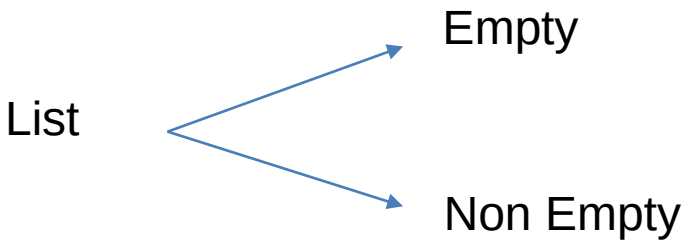
Insert: At Last

Steps

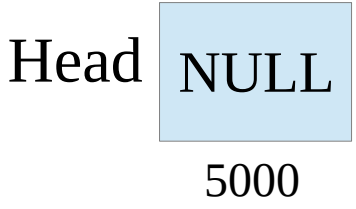
- Analysis : Logic /Cases
- Flowchart
- Algorithm
- Code

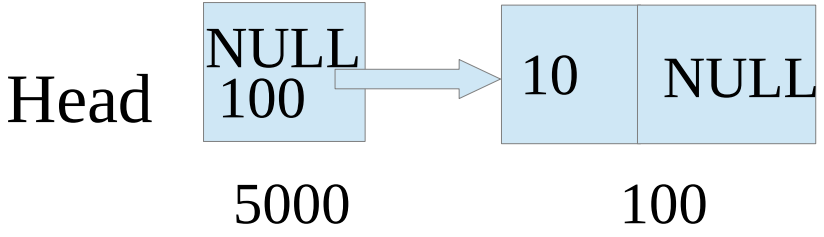
1.Analysis

- Cases :

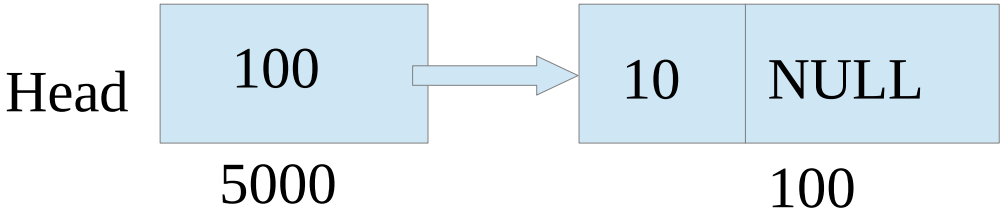


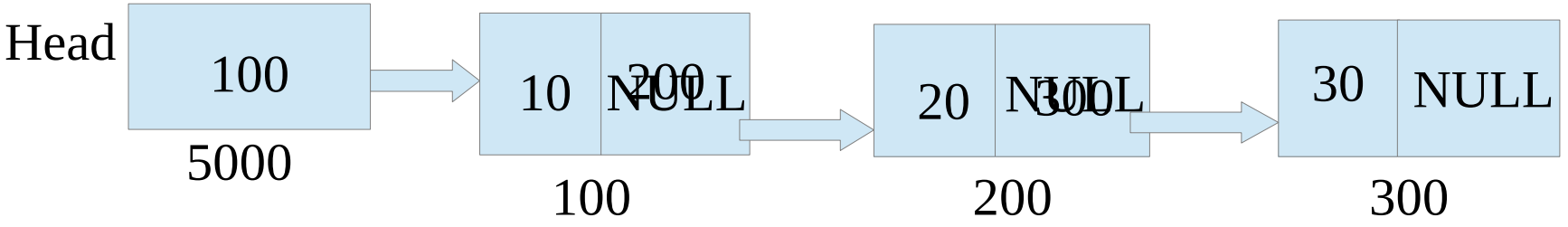
Case 1: List is Empty



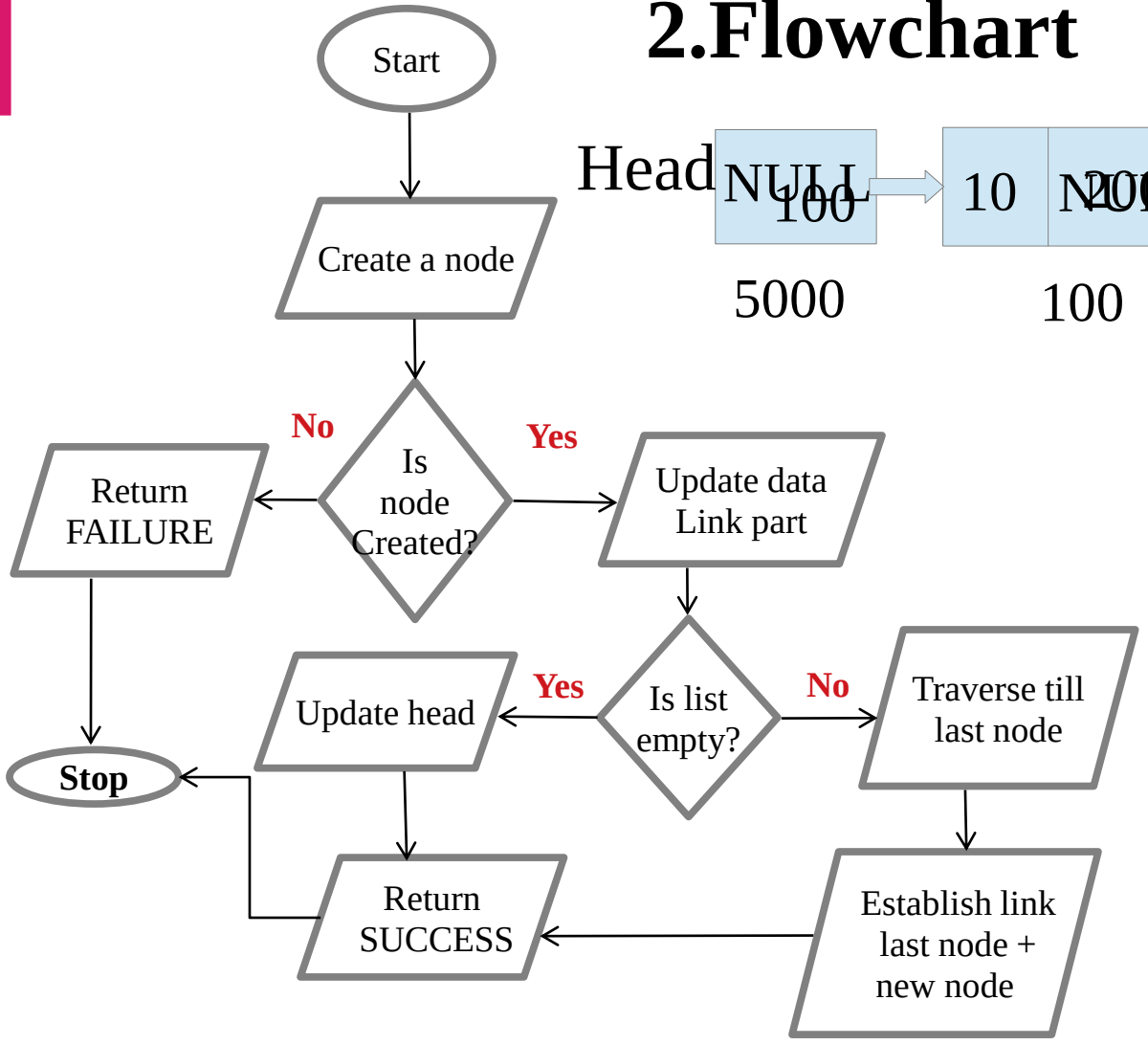


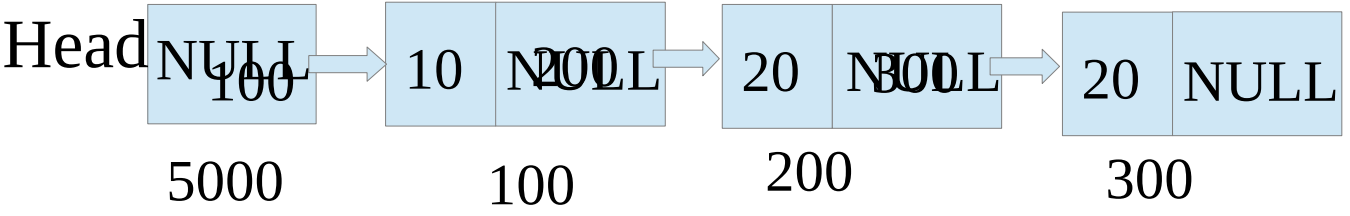
Case 2 : List is Non Empty



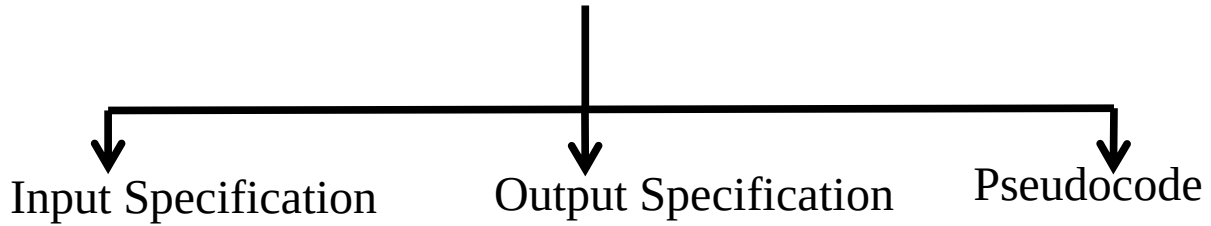


2.Flowchart


Start
Create a node
Is node Created?
No
Yes
Return FAILURE
Update data Link part
Is list empty?
Yes
No
Update head
Traverse till last node
Stop
Return SUCCESS
Establish link last node + new node


Head
100
NULL
10
200
20
300
20
NULL
5000
100
200
300

3.Algorithm

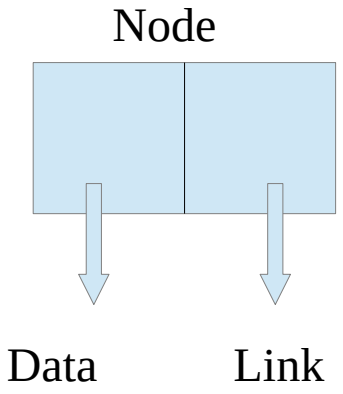


Algorithm : Insert_at_last(Head,n_data)

1.Input Specification :- Head : Pointer containing first node address
n_data : Item to be added

2.Output Specification :- Status : SUCCESS /FAILURE

How to create node?



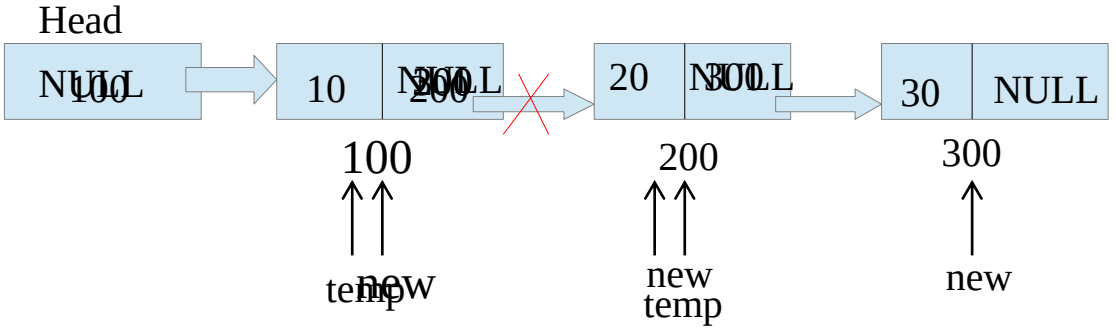

```
typedef int data_t
typedef struct node
{
    data_t data;
    struct node *link
}Slist_t;
```

```
int a = 10;
char  *q = &a;
int *ptr = &a;
```

```
Slist_t *Head = NULL;
Slist_t *new;
```

Pseudo code

1. new <- Memalloc(sizeof (Slist))
2. if (new = NULL)
   return FAILURE
3. new ⟶ data <- n_data
   new ⟶ link <- NULL
4. if (Head = NULL)
   Head <- new
   return SUCCESS
5. temp <- Head
6. while (temp ⟶ link != NULL)
   temp <- temp ⟶ link
7. temp ⟶ link <- new
   return SUCCESS



Code – insert_at_last(Head,n_data)