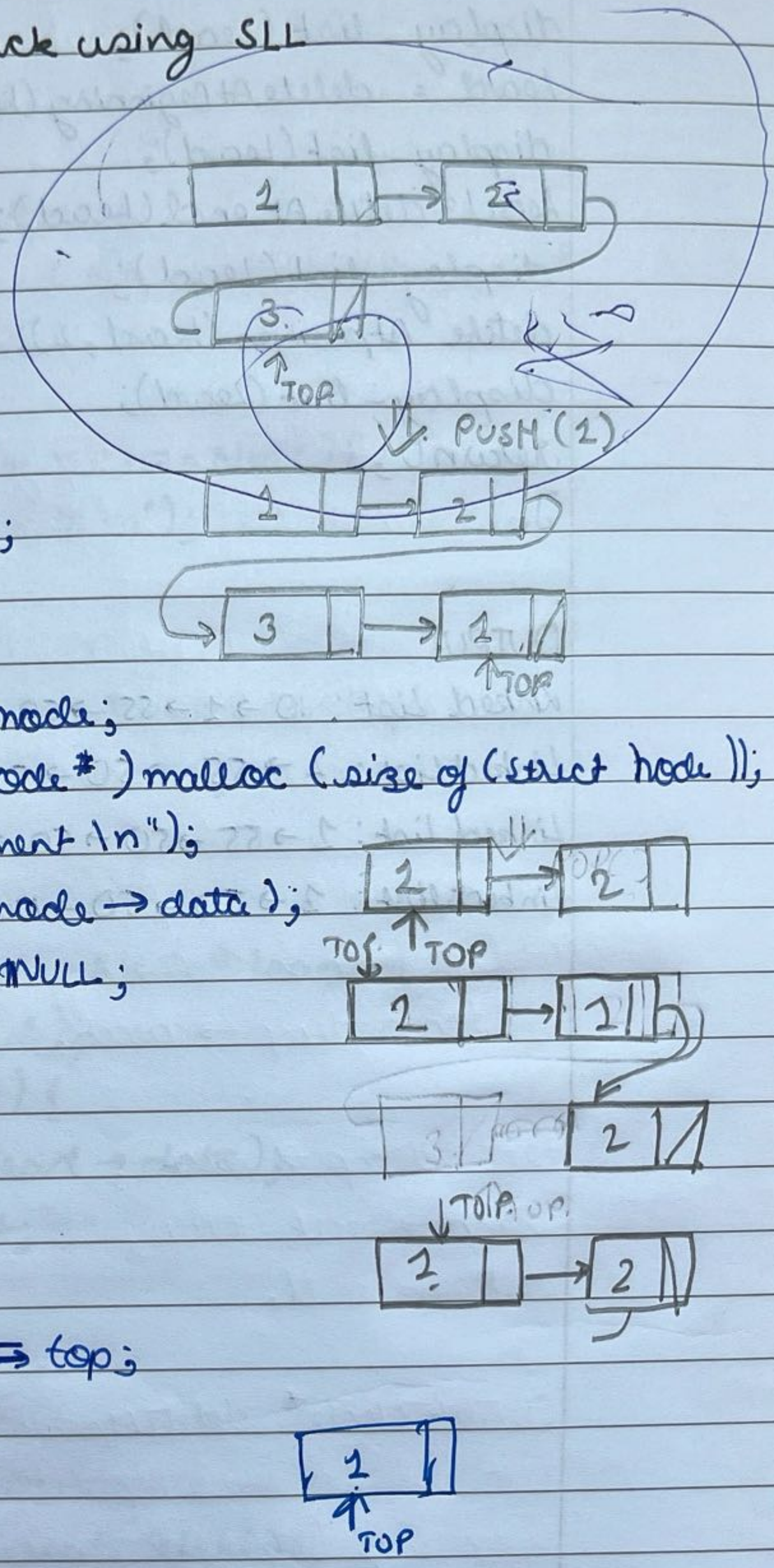
LAB PROGRAM

__/__/____

2a) Implementation of stack using SLL

```
#include <stdio.h>
# include <conio.h>
# include <stdlib.h>
struct node {
    int data;
    struct node * next;
};
Struct node *top = NULL;
void Push()
{
    struct node *new_node;
    new_node = (struct node*) malloc (sizeof (struct node));
    Printf ("Enter the element \n");
    Scanf ("%d", & new_node → data);
    new_node → next = NULL;
    if (top == NULL)
      {
        top = new_node;
      }
    else {
        new_node → next ⇒ top;
        top = new_node;
      }
  }
void pop ()
{ if (top == null)
      Printf ("Stack is empty");
  else
    {
```



```
        printf ("Deleted item o/o d ", top→data);
         top = top → next;
         }
}
void display()
 {
   struct node * temp;
   if (top = = NULL)
      Printf("Stack is empty \n");
   else
    {
      temp = top;
      while (temp != NULL) {
        Printf("%d \n", temp → data);
        temp = temp → next;
      }
   }
}
Voicd Main()
{ int choice;
  while (1) {
       Printf (" 1. push \n" );
       Print ("2. pop \n");
       Printf ("3. display \n");
       Printf ("4, Exist \n");
       Printf (" Enter your choice \n ");
       Scanf (" %d", & choice );
       Switch(choice)
       {
```

30/11/24.

```
        case 1: Push(); break;
        case 2: Pop(); break;
        case 3: display(); break;
        case 4: exist(0);
    }
  }
}
```

__/__/____

2b. Implementation of queue using SLL

```
#include <stdio.h>
#include <conio.h>
#include<stdlib.h>
Struct node
{ int data;
  struct node * next;
};
Struct node * front = NULL, * rear = NULL;
void insert() {
 Struct node * new_node;
 new_node = (Struct node*) malloc (sizeof (Struct node));
 Printf("Enter the element \n");
 Scanf("%d", &new_node → data);
 new_node → next = NULL;
 if (rear == NULL)
    { rear = new_node;
      front = new_node;
     }
    else {
      rear → next = new_node;
      rear = new_node;
     }
  }
void del() {
    if (front == NULL)
        Printf("Queue is empty");
    else {
     Printf("Deleted element is %d", front → data;
     front = front → next; } }
```

__/__/____

```
void display () {
    struct node * temp;
    if (front == NULL)
      Printf(" Queue is empty ");
      temp = front;
      while (temp != null)
        {
          Printf("%d \n", temp → data );
          temp = temp → next;
        }
    }
void main(){
    int choice;
    while (1) {
      Printf("2. insert \n");
      Printf ("2. Delet \n");
      Printf("3. Display\n");
      Printf ("4. Exit \n");
      Printf ("Enter choice \n");
      scanf ("%d", & choic);
Switch (choice) {
      case 1: insert(); break;
      case 2 : del (); break;
      case 3; display(); break;
      case 4: exit(0);
      }
}
```

__/__/____

1 Sorting, concatinati, reverse SLL.

```
#include <stdio.h>
#include <stdlib.h>
struct node {
    int data;
    struct node * next;
};
void insertend (struct node ** head, int value)
{
    struct node * newNode = (struct node *)malloc (
        sizeof (struct node ));
    NewNode → data = value;
    newNode → next = NULL;
    if (* head == NULL ) {
        * head = newNode;
        return;
    }
    struct node* current → next;
    }
    current → next = newNode;
}
Printlist ( struct node * head ) {
    while ( head != NULL) {
        Printf ("%d → ", head → data);
        head = head → next;
    }
    PrintF ("NULL\n");
}
void insertionsort ( struct node** head ) {
```

```
if (*head == NULL || (*head) -> next == NULL)
    {
        return;
    }
struct node* sorted = NULL;
struct node* current = *head;
while (current != NULL) {
    struct node* next = current -> next;
    if (sorted == NULL || sorted -> data > curr -> data)
    {
        current -> next = sorted;
        sorted = current;
    } else {
        struct node* temp = sorted;
        while (temp -> next != NULL && temp -> next
                -> data < current -> data)
        {
            temp = temp -> data;
        }
        current -> next = temp -> next;
        temp -> next = current;
    }
    current = next;
void concatinate lists (struct node** list1, struct node**
                                                list2)
{ if (*list1 == NULL)
        *list1 = *list2;
    } else {
        struct node* curr = *list1;
```

```
while (current → next != null)
    {
        current = current → next;
    }
    current → next = *list2;
}}
int main()
{
    struct node* list1 = null;
    struct node* list2 = null;
    insertend (&list2, 2);
    insertend (&list2, 4);
    insertend (&list2, 8);
    insertend (&list2, 10);
    insertend (&list2, 7);

    Printlist (list2);
    reverselist (&list2);
    Printlist (list2);

    concatinatelists (&list1, &list2);
    Printlist (list2);
    return 0;
}.
```