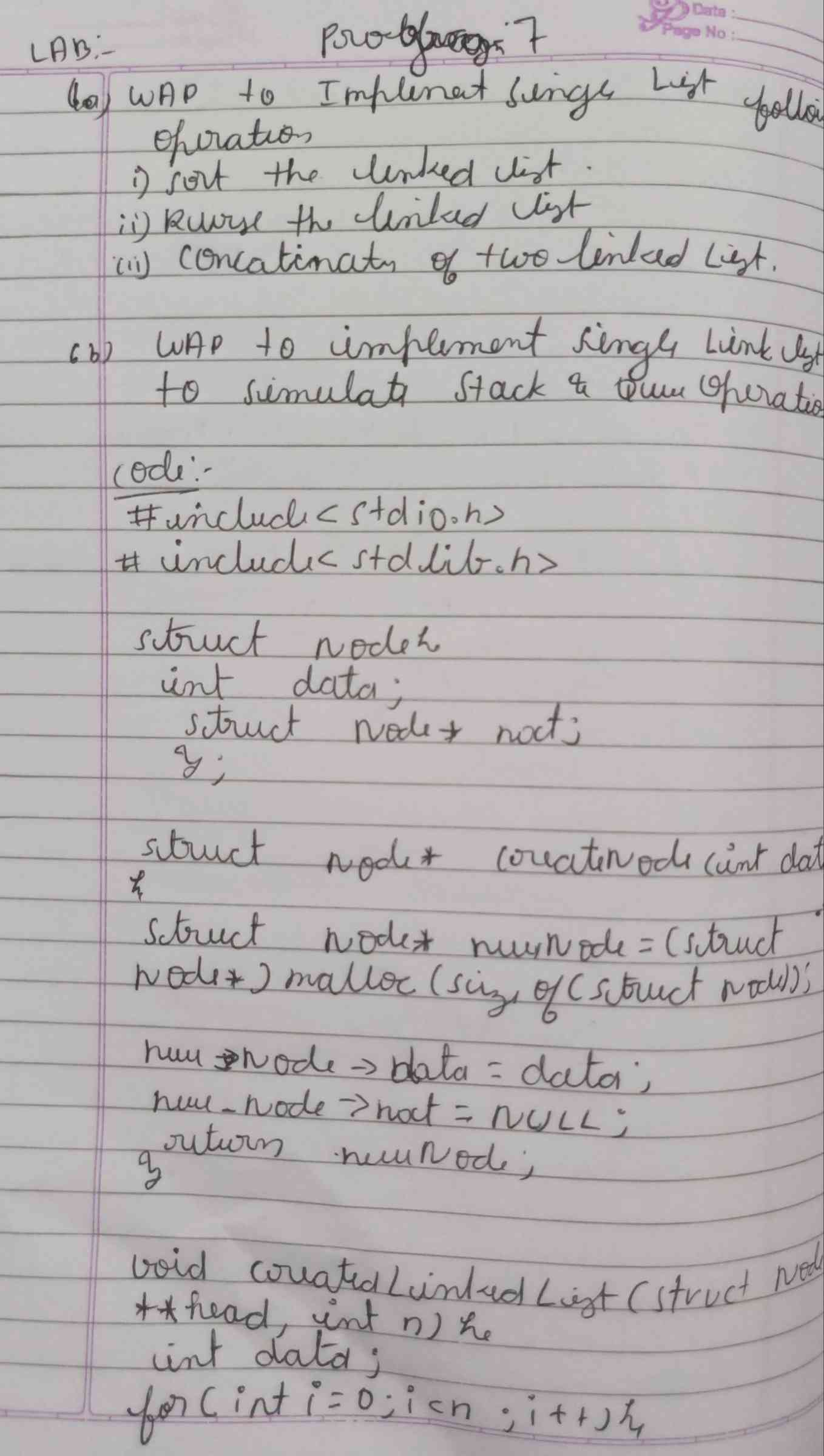
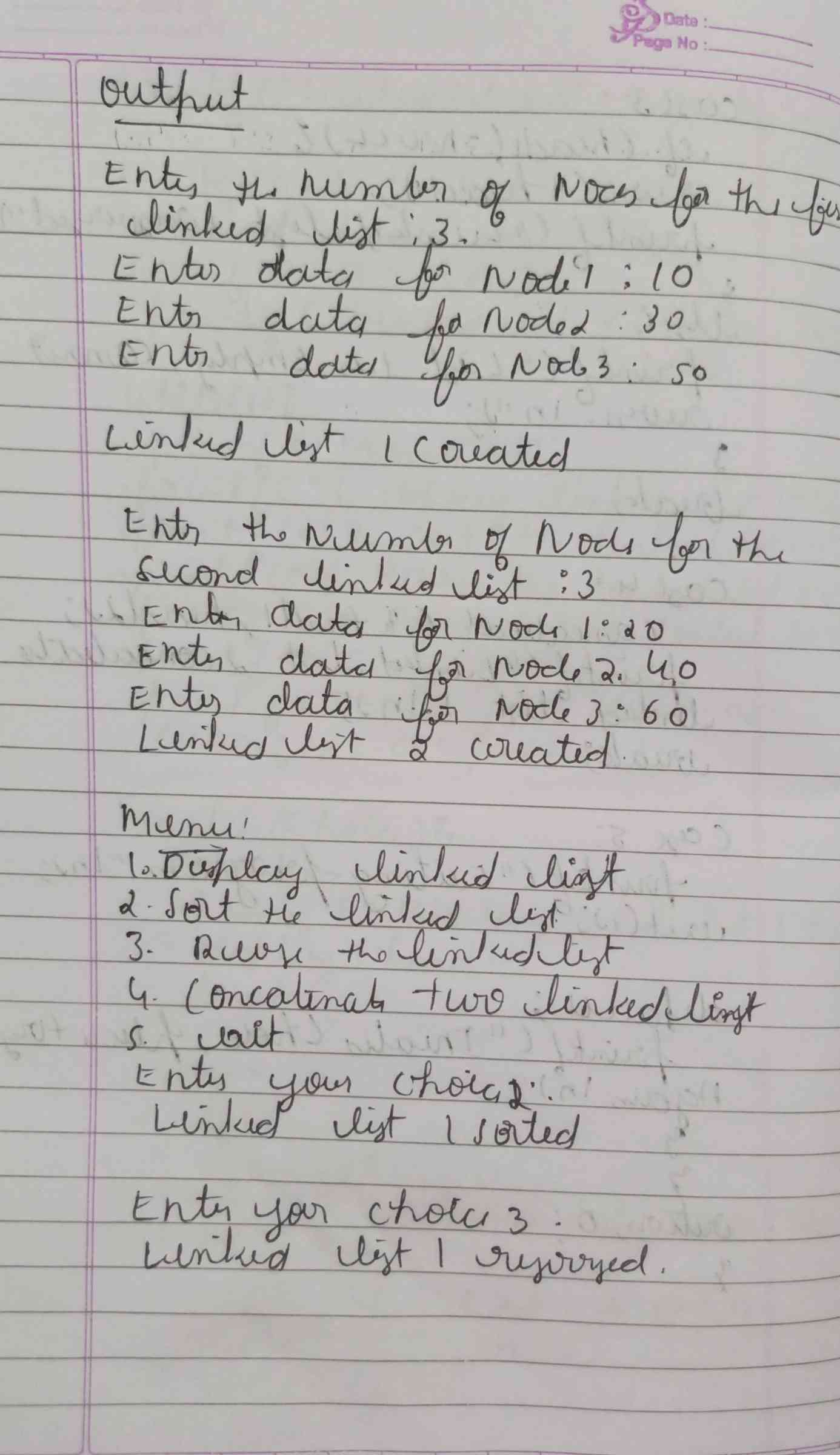
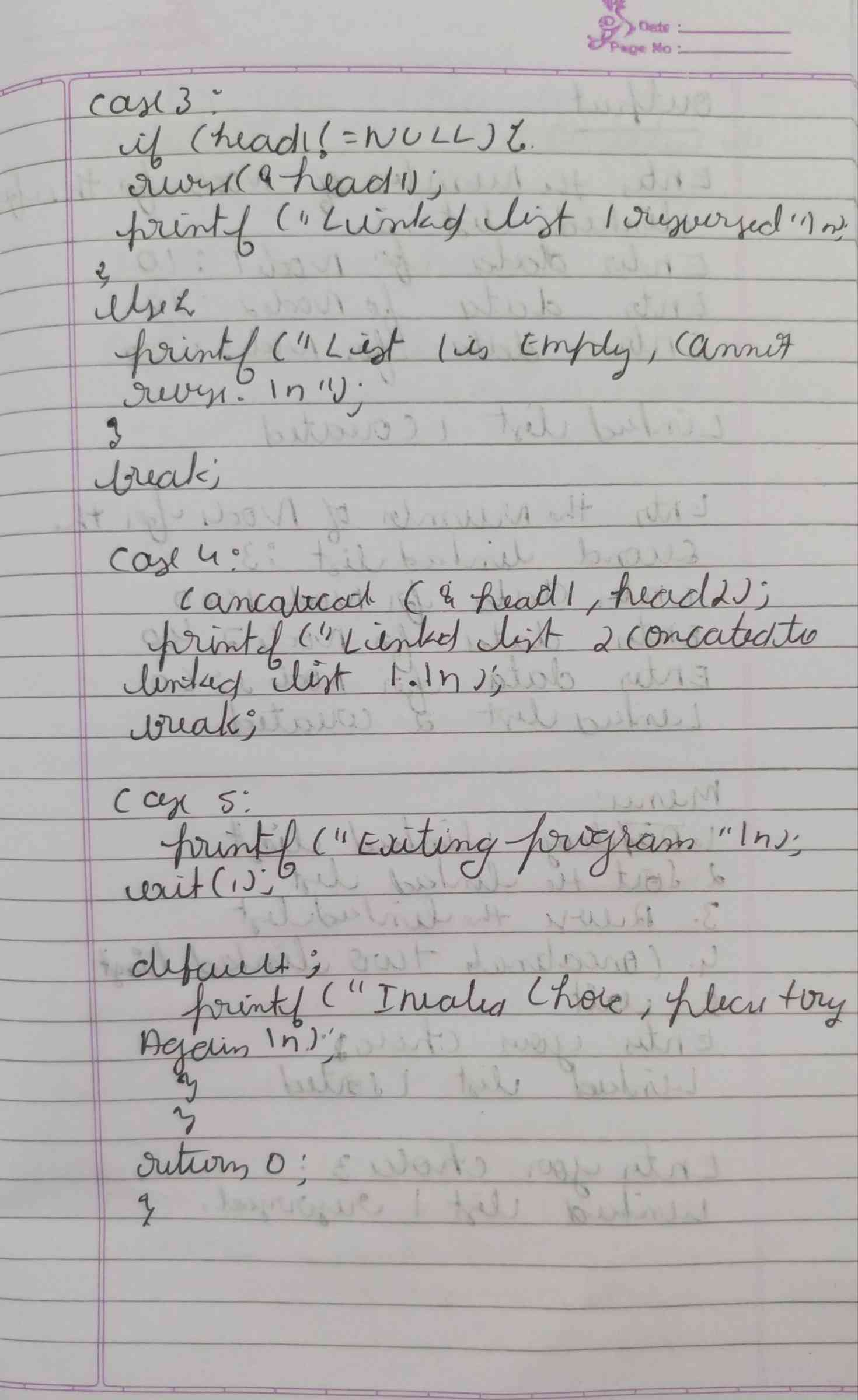
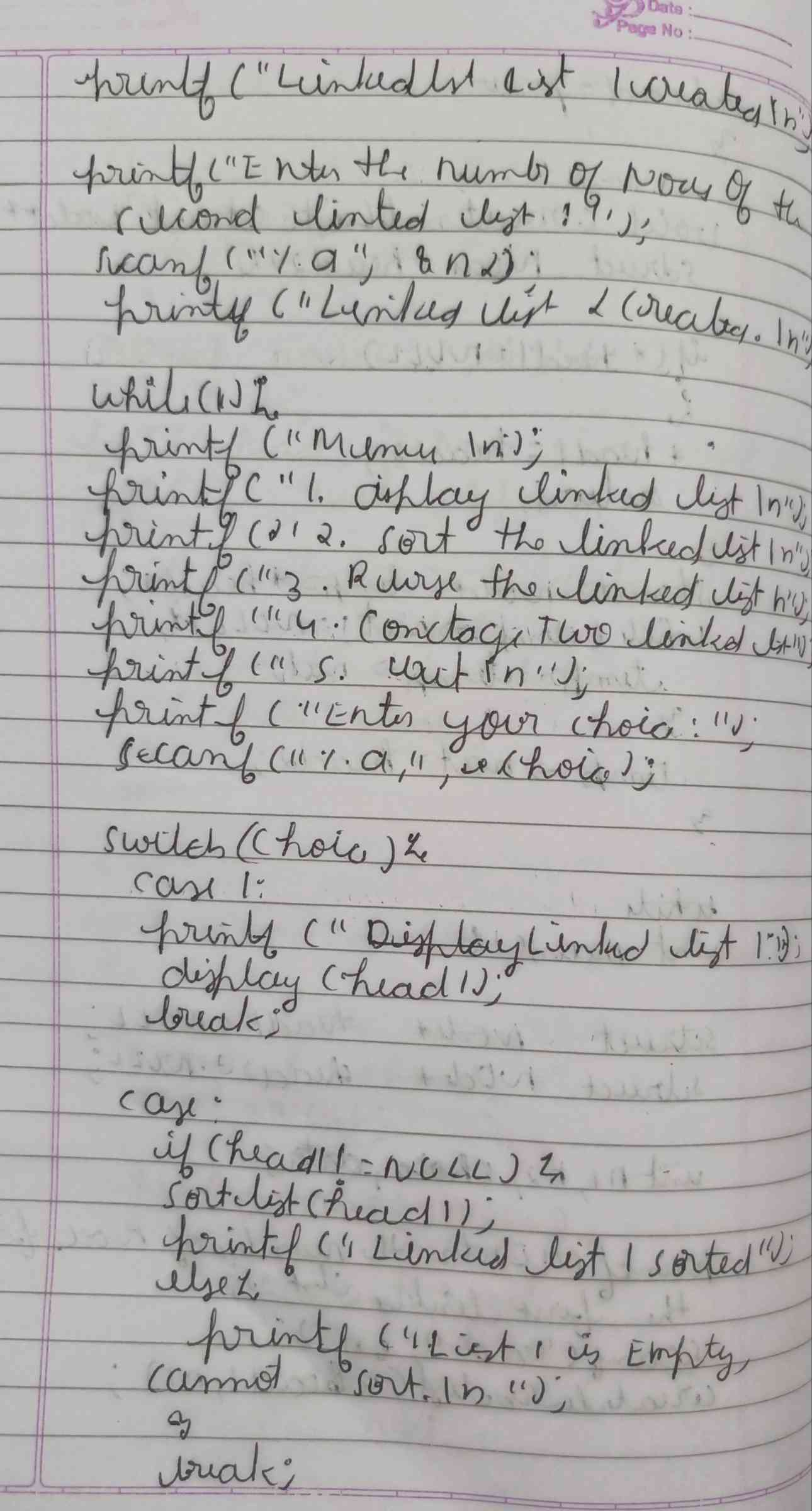
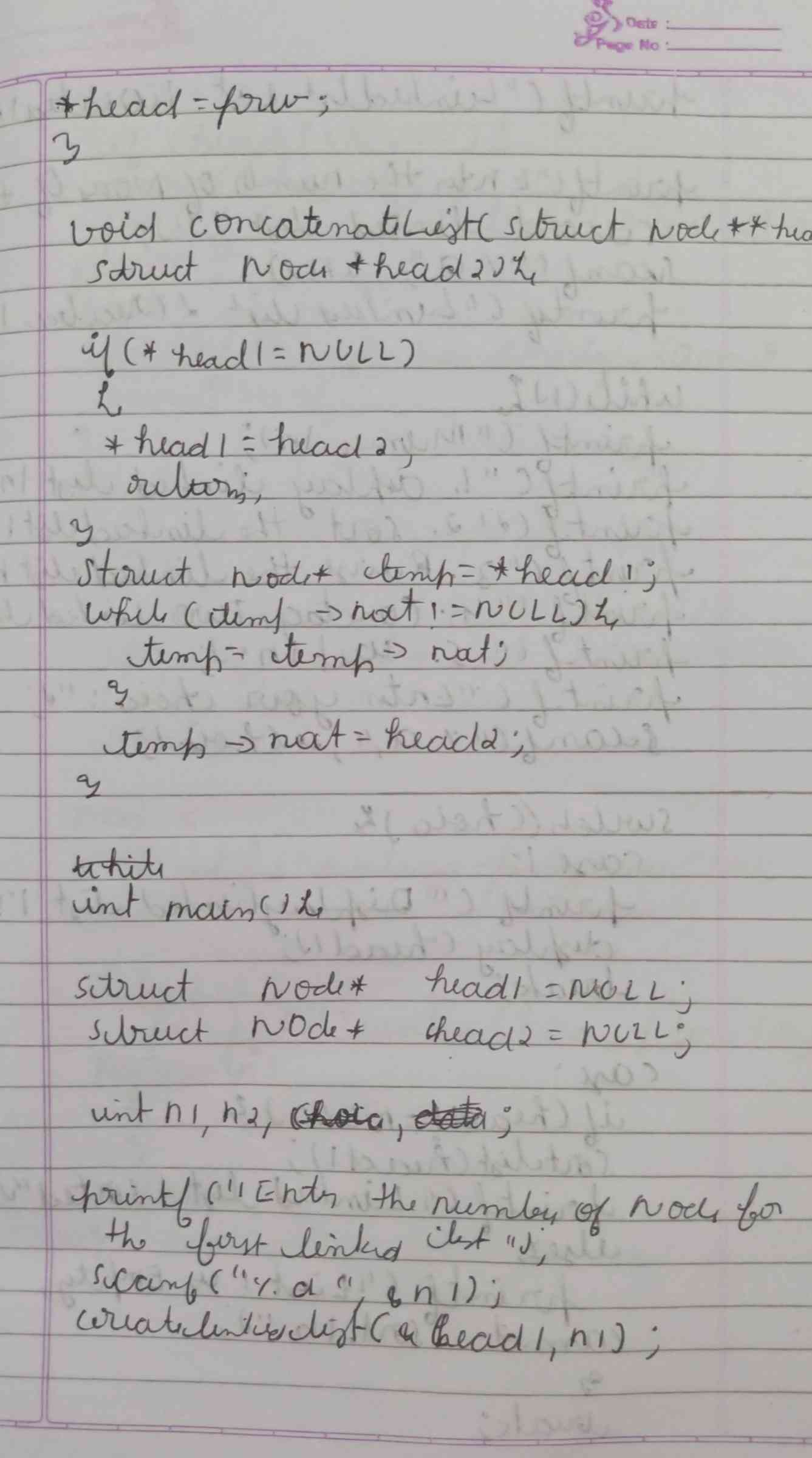
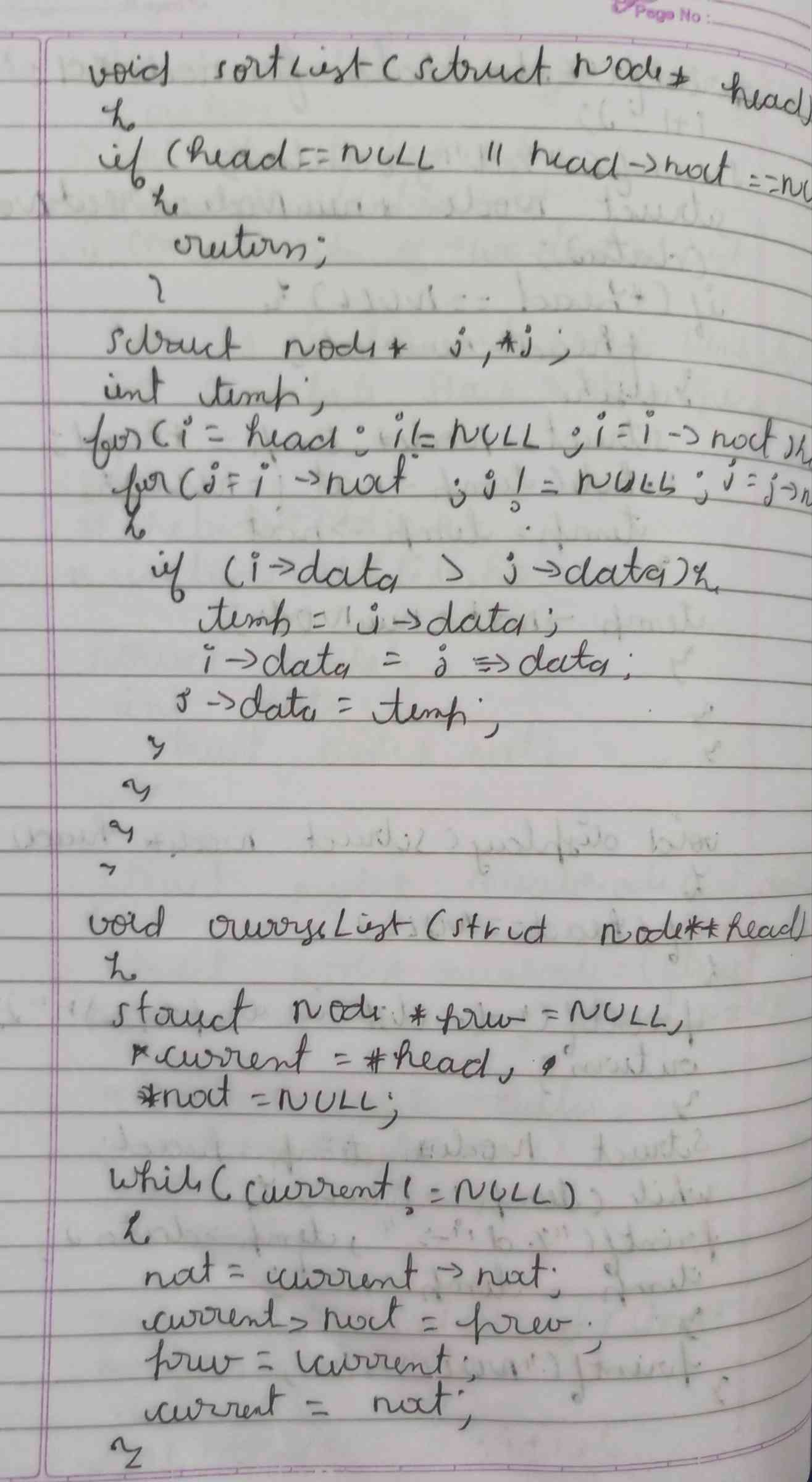
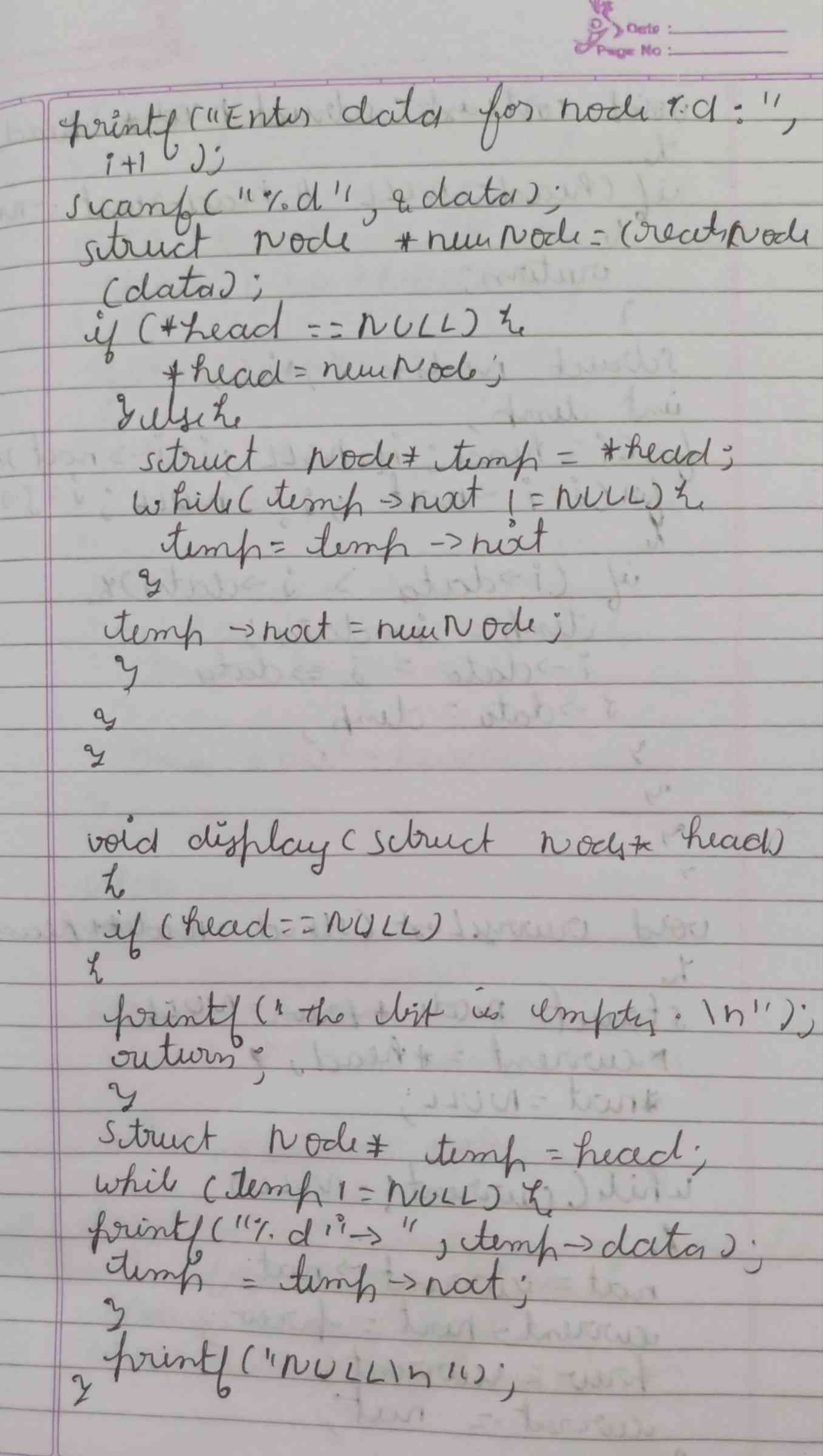
**LABORATORY PROGRAM – 7**

WAP to Implement Single Link List with following operations: Sort the linkedlist, Reverse the linkedlist, Concatenation of two linked lists.

**OBSERVATION :**

****

**CODE :**

**#include <stdio.h>**

**#include <stdlib.h>**

**struct Node {**

**int data;**

**struct Node\* next;**

**};**

**struct Node\* createNode(int data) {**

**struct Node\* newNode = (struct Node\*) malloc(sizeof(struct Node));**

**newNode->data = data;**

**newNode->next = NULL;**

**return newNode;**

**}**

**void createLinkedList(struct Node\*\* head, int n) {**

**int data;**

**struct Node\* temp;**

**for (int i = 0; i < n; i++) {**

**printf("Enter data for node %d: ", i + 1);**

**scanf("%d", &data);**

**struct Node\* newNode = createNode(data);**

**if (\*head == NULL) {**

**\*head = newNode;**

**} else {**

**temp = \*head;**

**while (temp->next != NULL) {**

**temp = temp->next;**

**}**

**temp->next = newNode;**

**}**

**}**

**}**

**void display(struct Node\* head) {**

**struct Node\* temp = head;**

**if (temp == NULL) {**

**printf("The list is empty.\n");**

**return;**

**}**

**while (temp != NULL) {**

**printf("%d -> ", temp->data);**

**temp = temp->next;**

**}**

**printf("NULL\n");**

**}**

**void sortList(struct Node\* head) {**

**if (head == NULL) return;**

**struct Node \*i, \*j;**

**int temp;**

**for (i = head; i != NULL; i = i->next) {**

**for (j = i->next; j != NULL; j = j->next) {**

**if (i->data > j->data) {**

**temp = i->data;**

**i->data = j->data;**

**j->data = temp;**

**}**

**}**

**}**

**}**

**void reverse(struct Node\*\* head) {**

**struct Node \*prev = NULL, \*current = \*head, \*next = NULL;**

**while (current != NULL) {**

**next = current->next;**

**current->next = prev;**

**prev = current;**

**current = next;**

**}**

**\*head = prev;**

**}**

**void concatenate(struct Node\*\* head1, struct Node\* head2) {**

**if (\*head1 == NULL) {**

**\*head1 = head2;**

**return;**

**}**

**struct Node\* temp = \*head1;**

**while (temp->next != NULL) {**

**temp = temp->next;**

**}**

**temp->next = head2;**

**}**

**int main() {**

**struct Node\* head1 = NULL;**

**struct Node\* head2 = NULL;**

**int choice, n1, n2;**

**printf("Enter the number of nodes for the first linked list: ");**

**scanf("%d", &n1);**

**createLinkedList(&head1, n1);**

**printf("Linked List 1 created.\n");**

**printf("Enter the number of nodes for the second linked list: ");**

**scanf("%d", &n2);**

**createLinkedList(&head2, n2);**

**printf("Linked List 2 created.\n");**

**while (1) {**

**printf("\nMenu:\n");**

**printf("1. Display Linked List\n");**

**printf("2. Sort Linked List\n");**

**printf("3. Reverse Linked List\n");**

**printf("4. Concatenate Two Linked Lists\n");**

**printf("5. Exit\n");**

**printf("Enter your choice: ");**

**scanf("%d", &choice);**

**switch (choice) {**

**case 1:**

**printf("Linked List 1:\n");**

**display(head1);**

**break;**

**case 2:**

**if (head1 != NULL) {**

**sortList(head1);**

**printf("Linked List 1 sorted.\n");**

**} else {**

**printf("List 1 is empty, cannot sort.\n");**

**}**

**break;**

**case 3:**

**if (head1 != NULL) {**

**reverse(&head1);**

**printf("Linked List 1 reversed.\n");**

**} else {**

**printf("List 1 is empty, cannot reverse.\n");**

**}**

**break;**

**case 4:**

**concatenate(&head1, head2);**

**printf("Linked List 2 concatenated to Linked List 1.\n");**

**break;**

**case 5:**

**printf("Exiting program.\n");**

**exit(1);**

**default:**

**printf("Invalid choice. Please try again.\n");**

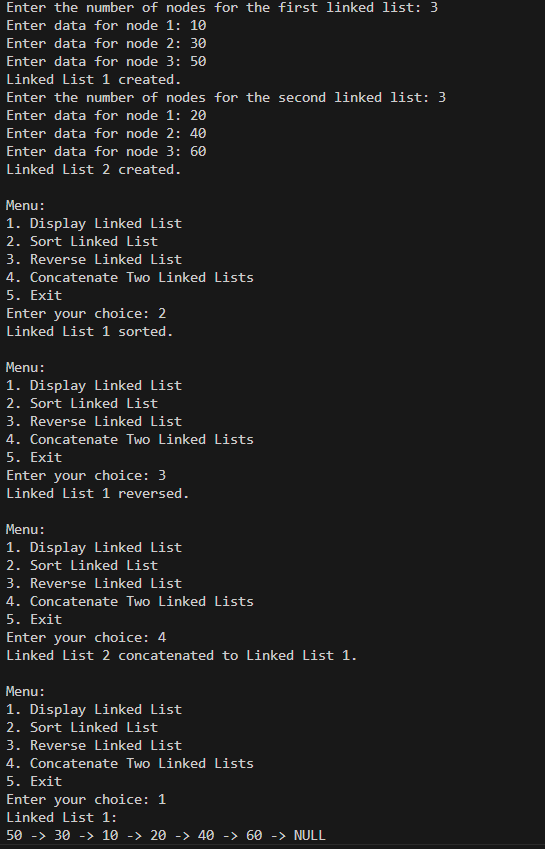
**}**

**}**

**return 0;**

**}**

**OUTPUT :**

****