

**Symbiosis Institute of Technology**

**Faculty of Engineering**

**CSE- Academic Year 2024-25**

**Data Structures – Lab Batch 2023-27**

|  |  |
| --- | --- |
| **Lab Assignment No:- 5** | |
|  | |
| **Name of Student** | Faheemuddin Sayyed |
| **PRN No.** | 23070122196 |
| **Batch** | 23-27 |
| **Class** | CSE C-1 |
| **Academic Year & Semester** | SY 24-25 |
| **Date of Performance** | 12/08/24 |
|  | |
| **Title of Assignment:** | * Menu-driven program for: * Creation of Doubly Linked list * Insertion at beginning * Insertion at end * Insertion after specific node * Display |
| **Source Code/Algorithm/Flow Chart:** | #include <stdio.h>  #include <stdlib.h>  struct Node{  struct Node \*prev;  int data;  struct Node \*next;  }\*first = NULL, \*last = NULL;  typedef struct Node node;  void display(){  node \*temp = first;  printf("\n");  while(temp){  printf("%d ", temp -> data);  temp = temp -> next;  }  printf("\n");  }  void create(){  node \*temp;  int data;  char choice;  do{  temp = (node \*)malloc(sizeof(node));  printf("\nEnter data: ");  scanf("%d", &data);  getchar();  if(!first){  temp -> data = data;  temp -> next = temp -> prev = NULL;  first = temp;  last = temp;  } else {  temp -> data = data;  temp -> next = NULL;  temp -> prev = last;  last -> next = temp;  last = temp;  }  printf("\nDo you wish to continue? (Y/N): ");  scanf("%c", &choice);  } while(choice == 'Y' || choice == 'y');  printf("\nList created!\n");  display();  }  void displayReverse(){  node \*temp = last;  printf("\n");  while(temp){  printf("%d ",temp -> data);  temp = temp -> prev;  }  printf("\n");  }  int count(){  node \*temp = first;  int count = 0;  while(temp){  count++;  temp = temp -> next;  }  return count;  }  void insertBeg(int data){  node \*temp;  temp = (node \*)malloc(sizeof(node));  temp -> data = data;  temp -> next = first;  temp -> prev = NULL;  first -> prev = temp;  first = temp;  display();  }  void insertEnd(int data){  node \*temp;  temp = (node \*)malloc(sizeof(node));  temp -> data = data;  temp -> next = NULL;  temp -> prev = last;  last -> next = temp;  last = temp;  display();  }  void insertMid(int data, int pos){  if(pos < 0 || pos > count()) return;  node \*temp = first;  node \*new;  for(int i = 0; i < pos - 1; i++){  temp = temp -> next;  }  new = (node \*)malloc(sizeof(node));  new -> data = data;  new -> next = temp -> next;  new -> prev = temp;  temp -> next = new;  temp -> next -> prev = new;  display();  }  void deleteBeg(){  if(first == NULL) return;  node \*temp = first;  first = first -> next;  if(first != NULL) first -> prev = NULL;  free(temp);  display();  }  void deleteEnd(){  if(last == NULL) return;  node \*temp = last;  last = last -> prev;  if(last != NULL) last -> next = NULL;  free(temp);  display();  }  void deleteMid(int pos){  if(pos < 0 || pos >= count()) return;  node \*temp = first;  for(int i = 0; i < pos; i++){  temp = temp -> next;  }  temp -> prev -> next = temp -> next;  if(temp -> next != NULL) temp -> next -> prev = temp -> prev;  free(temp);  display();  }  int main() {  int choice, data, pos;  while(1){  printf("\nDoubly Linked List Operations\n");  printf("1. Create\n");  printf("2. Display\n");  printf("3. Display in Reverse\n");  printf("4. Count\n");  printf("5. Insert at Beginning\n");  printf("6. Insert at End\n");  printf("7. Insert at Position\n");  printf("8. Delete from Beginning\n");  printf("9. Delete from End\n");  printf("10. Delete from Position\n");  printf("11. Exit\n");  printf("Enter your choice: ");  scanf("%d", &choice);  switch(choice){  case 1:  create();  break;  case 2:  display();  break;  case 3:  displayReverse();  break;  case 4:  printf("\nCount: %d\n", count());  break;  case 5:  printf("\nEnter data to insert at beginning: ");  scanf("%d", &data);  insertBeg(data);  break;  case 6:  printf("\nEnter data to insert at end: ");  scanf("%d", &data);  insertEnd(data);  break;  case 7:  printf("\nEnter data to insert: ");  scanf("%d", &data);  printf("Enter position: ");  scanf("%d", &pos);  insertMid(data, pos);  break;  case 8:  deleteBeg();  break;  case 9:  deleteEnd();  break;  case 10:  printf("\nEnter position to delete: ");  scanf("%d", &pos);  deleteMid(pos);  break;  case 11:  exit(0);  default:  printf("\nInvalid choice! Please try again.\n");  }  }  return 0;  } |
| **Output Screenshots** | Doubly Linked List Creation:    1. Insertion at beginning:      1. Insertion at end:      1. Deletion at beginning:      1. Deletion at end:      1. Insertion at specific position:      1. Deletion at specific position:      1. Display and Reverse Display: |
| **Practice questions** | N/A |
| **Conclusion** | Thus we have explored all the different doubly linked list operations and implemented them using a menu driven program. |