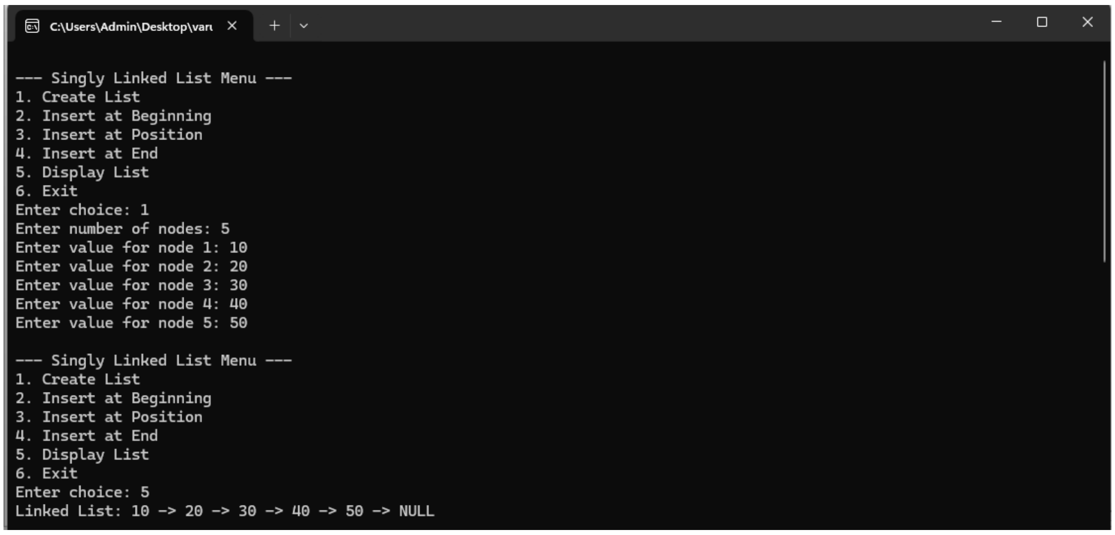
4. WAP to Implement Singly Linked List with following operations
a) Create a linked list.
b) b) Insertion of a node at first position, at any position and at end of list.
Display the contents of the linked list.

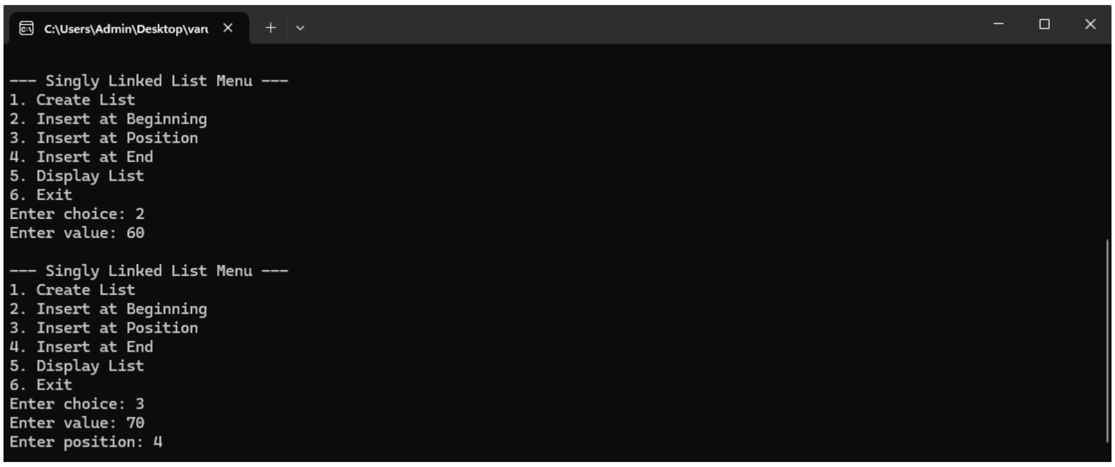
```
#include <stdio.h>
#include <stdlib.h>

// Structure of a Node
struct Node {
    int data;
    struct Node* next;
};

struct Node* head = NULL;

// Function to create a linked list (append nodes)
void createList() {
    int n, value;
    printf("Enter number of nodes: ");
    scanf("%d", &n);

    for (int i = 0; i < n; i++) {
        printf("Enter value for node %d: ", i + 1);
        scanf("%d", &value);

        struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
        newNode->data = value;
        newNode->next = NULL;

        if (head == NULL) {
            head = newNode;
        } else {
            struct Node* temp = head;
            while (temp->next != NULL)
                temp = temp->next;
            temp->next = newNode;
        }
    }
}

// Insert at beginning
void insertAtBeginning(int value) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = value;
    newNode->next = head;
    head = newNode;
}

// Insert at end
void insertAtEnd(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* temp = head;
   while (temp->next != NULL)
      temp = temp->next;

   temp->next = newNode;
}

// Insert at specific position
void insertAtPosition(int value, int pos) {
   struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
   newNode->data = value;

   if (pos == 1) {
      newNode->next = head;
      head = newNode;
      return;
   }

   struct Node* temp = head;
   for (int i = 1; i < pos - 1 && temp != NULL; i++) {
      temp = temp->next;
   }

   if (temp == NULL) {
      printf("Position out of range!\n");
      return;
   }

   newNode->next = temp->next;
   temp->next = newNode;
}
```

```
// Display list
void display() {
    if (head == NULL) {
        printf("List is empty.\n");
        return;
    }

    struct Node* temp = head;
    printf("Linked List: ");
    while (temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
}

// Main menu
int main() {
    int choice, value, pos;

    while (1) {
        printf("\n--- Singly Linked List Menu ---\n");
        printf("1. Create List\n");
        printf("2. Insert at Beginning\n");
        printf("3. Insert at Position\n");
        printf("4. Insert at End\n");
        printf("5. Display List\n");
        printf("6. Exit\n");
        printf("Enter choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                createList();
                break;

            case 2:
                printf("Enter value: ");
                scanf("%d", &value);
                insertAtBeginning(value);
                break;

            case 3:
                printf("Enter value: ");
                scanf("%d", &value);
                printf("Enter position: ");
                scanf("%d", &pos);
                insertAtPosition(value, pos);
                break;

            case 4:
                printf("Enter value: ");
                scanf("%d", &value);
                insertAtEnd(value);
                break;

            case 5:
                display();
                break;

            case 6:
                exit(0);

            default:
                printf("Invalid choice!\n");
        }
    }
}
```

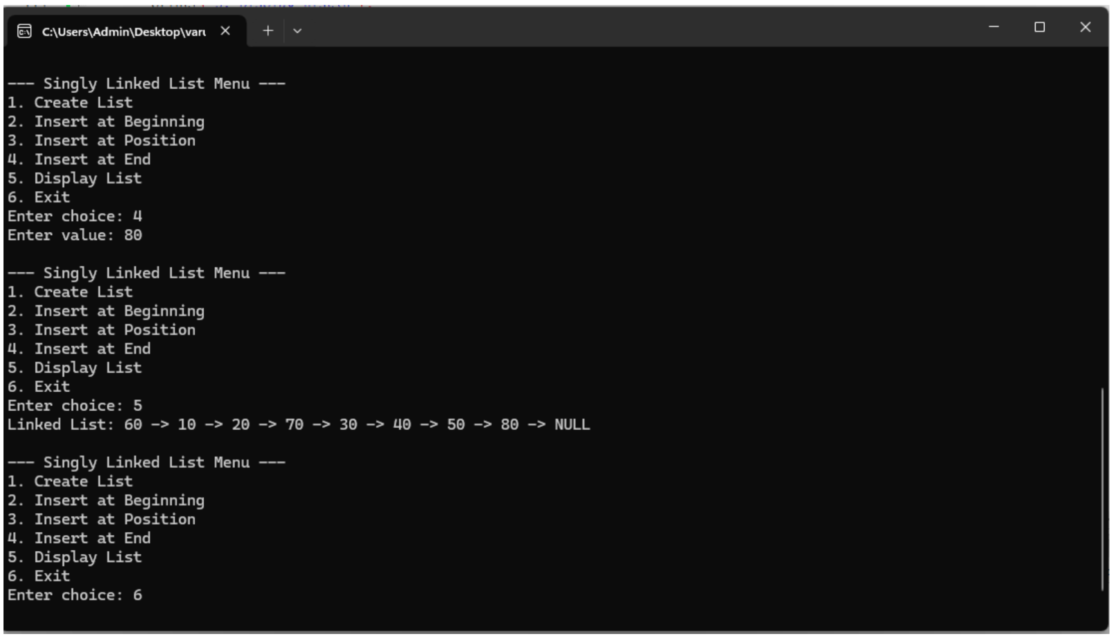
Output



```
--- Singly Linked List Menu ---
1. Create List
2. Insert at Beginning
3. Insert at Position
4. Insert at End
5. Display List
6. Exit
Enter choice: 1
Enter number of nodes: 5
Enter value for node 1: 10
Enter value for node 2: 20
Enter value for node 3: 30
Enter value for node 4: 40
Enter value for node 5: 50

--- Singly Linked List Menu ---
1. Create List
2. Insert at Beginning
3. Insert at Position
4. Insert at End
5. Display List
6. Exit
Enter choice: 5
Linked List: 10 -> 20 -> 30 -> 40 -> 50 -> NULL
```



```
--- Singly Linked List Menu ---
1. Create List
2. Insert at Beginning
3. Insert at Position
4. Insert at End
5. Display List
6. Exit
Enter choice: 2
Enter value: 60

--- Singly Linked List Menu ---
1. Create List
2. Insert at Beginning
3. Insert at Position
4. Insert at End
5. Display List
6. Exit
Enter choice: 3
Enter value: 70
Enter position: 4
```



```
--- Singly Linked List Menu ---
1. Create List
2. Insert at Beginning
3. Insert at Position
4. Insert at End
5. Display List
6. Exit
Enter choice: 4
Enter value: 80

--- Singly Linked List Menu ---
1. Create List
2. Insert at Beginning
3. Insert at Position
4. Insert at End
5. Display List
6. Exit
Enter choice: 5
Linked List: 60 -> 10 -> 20 -> 70 -> 30 -> 40 -> 50 -> 80 -> NULL

--- Singly Linked List Menu ---
1. Create List
2. Insert at Beginning
3. Insert at Position
4. Insert at End
5. Display List
6. Exit
Enter choice: 6
```