Assignment-6

**#Implement linked list.**

**Code:**

// Linked list operations in C

#include <stdio.h>

#include <stdlib.h>

// Create a node struct Node {

int data; struct Node\* next;

};

// Insert at the beginning void insertAtBeginning(struct Node\*\* head\_ref, int new\_data) {

// Allocate memory to a node struct Node\* new\_node = (struct Node\*)malloc(sizeof(struct Node));

// insert the data

new\_node->data = new\_data;

new\_node->next = (\*head\_ref);

// Move head to new node

(\*head\_ref) = new\_node;

}

// Insert a node after a node

void insertAfter(struct Node\* prev\_node, int new\_data) { if (prev\_node == NULL) { printf("the given previous node cannot be NULL");

return;

}

struct Node\* new\_node = (struct Node\*)malloc(sizeof(struct Node)); new\_node->data = new\_data; new\_node->next = prev\_node->next; prev\_node->next = new\_node;

}

// Insert the the end void insertAtEnd(struct Node\*\* head\_ref, int new\_data) { struct Node\* new\_node = (struct Node\*)malloc(sizeof(struct Node)); struct Node\* last = \*head\_ref; /\* used in step 5\*/

new\_node->data = new\_data; new\_node->next = NULL;

if (\*head\_ref == NULL) { \*head\_ref = new\_node;

return;

}

while (last->next != NULL) last = last->next;

last->next = new\_node; return;

}

// Delete a node void deleteNode(struct Node\*\* head\_ref, int key) { struct Node \*temp = \*head\_ref, \*prev;

if (temp != NULL && temp->data == key) { \*head\_ref = temp->next; free(temp); return;

}

// Find the key to be deleted while (temp != NULL && temp->data != key) { prev = temp; temp = temp->next;

}

// If the key is not present if (temp == NULL) return;

// Remove the node prev->next = temp->next;

free(temp);

}

// Search a node int searchNode(struct Node\*\* head\_ref, int key) { struct Node\* current = \*head\_ref;

while (current != NULL) { if (current->data == key) return 1; current = current->next;

}

return 0;

}

// Sort the linked list void sortLinkedList(struct Node\*\* head\_ref) { struct Node \*current = \*head\_ref, \*index = NULL; int temp;

if (head\_ref == NULL) { return;

} else {

while (current != NULL) {

// index points to the node next to current index = current->next;

while (index != NULL) { if (current->data > index->data) { temp = current->data; current->data = index->data; index->data = temp;

}

index = index->next;

}

current = current->next;

}

}

}

// Print the linked list void printList(struct Node\* node) { while (node != NULL) { printf(" %d ", node->data); node = node->next;

}

}

// Driver program

int main() { struct Node\* head = NULL;

insertAtEnd(&head, 1); insertAtBeginning(&head, 2); insertAtBeginning(&head, 3); insertAtEnd(&head, 4); insertAfter(head->next, 5);

printf("Linked list: "); printList(head);

printf("\nAfter deleting an element: "); deleteNode(&head, 3); printList(head);

int item\_to\_find = 3; if (searchNode(&head, item\_to\_find)) { printf("\n%d is found", item\_to\_find);

} else {

printf("\n%d is not found", item\_to\_find);

}

sortLinkedList(&head);

printf("\nSorted List: "); printList(head);

}

**Output:**

