CSA0318 - LAB EXPERIMENTS

EXP NO.21: WRITE A C PROGRAM TO IMPLEMENT STACK OPERATION IN LINKED LIST[POP]

```
#include <stdio.h>
#include <stdlib.h>
struct node {
   int data;
   struct node *next;
};
int main() {
   struct node *top = NULL, *temp, *newnode;
   int n, i, item;
   printf("Enter number of elements to push: ");
   scanf("%d", &n);
   for(i = 0; i < n; i++) {
      newnode = (struct node*)malloc(sizeof(struct node));
      printf("Enter element to push: ");
      scanf("%d", &item);
      newnode->data = item;
      newnode->next = top;
      top = newnode;
   }
   if(top == NULL)
      printf("\nStack Underflow! No elements to pop.\n");
```

```
    else {
        temp = top;
        printf("\nPopped element = %d\n", temp->data);
        top = top->next;
        free(temp);
    }
    printf("\nStack after POP (Top to Bottom): ");
    temp = top;
    if(temp == NULL)
        printf("Stack is empty\n");
    else {
        while(temp != NULL) {
            printf("%d ", temp->data);
            temp = temp->next;
        }
    }
    return 0;
}
```

Output

```
Enter number of elements to push: 3
Enter element to push: 6
Enter element to push: 9
Enter element to push: 1

Popped element = 1

Stack after POP (Top to Bottom): 9 6

=== Code Execution Successful ===
```

EXP NO22: WRITE A C PROGRAM TO IMPLEMENT STACK OPERATION IN LINKED LIST[DISPLAY]

```
#include <stdio.h>

#include <stdlib.h>

struct node {

   int data;

   struct node *next;

};

int main() {

   struct node *top = NULL, *newnode, *temp;

   int n, i, item;

   printf("Enter number of elements to push: ");

   scanf("%d", &n);

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

      newnode = (struct node*)malloc(sizeof(struct node));

      printf("Enter element: ");

      scanf("%d", &item);

      newnode->data = item;

      newnode->next = top;

      top = newnode;

   }

   printf("\n--- Stack Display ---\n");

   if(top == NULL)

      printf("Stack is empty.\n");
```

```
    else {
        temp = top;
        printf("Stack elements (Top to Bottom): ");
        while(temp != NULL) {
            printf("%d ", temp->data);
            temp = temp->next;
        }
        printf("\n");
    }
    return 0;
}
```

```
Output
Enter number of elements to push: 4
Enter element: 7
Enter element: 9
Enter element: 0
Enter element: 5

--- Stack Display ---
Stack elements (Top to Bottom): 5 0 9 7


=== Code Execution Successful ===
```

EX.NO:23 IMPLEMENT A C PROGRAM TO PRINT NO OF NODES IN THE GIVEN LINKED LIST

```
#include <stdio.h>
#include <stdlib.h>
struct node {
    int data;
    struct node *next;
```

```
};
int main() {
    struct node *head = NULL, *newnode, *temp;
    int n, i, count = 0;
    printf("Enter number of nodes: ");
    scanf("%d", &n);
    for(i = 0; i < n; i++) {
        newnode = (struct node*)malloc(sizeof(struct node));
        printf("Enter data for node %d: ", i + 1);
        scanf("%d", &newnode->data);
        newnode->next = NULL;
        if(head == NULL)
            head = temp = newnode;
        else {
            temp->next = newnode;
            temp = newnode;
        }
    }
    temp = head;
    while(temp != NULL) {
        count++;
        temp = temp->next;
    }
```

```
    printf("\nTotal number of nodes = %d\n", count);
    return 0;
}
```

Output

```
Enter number of nodes: 3
Enter data for node 1: 10
Enter data for node 2: 20
Enter data for node 3: 30

Total number of nodes = 3


=== Code Execution Successful ===
```

EX.NO:24 IMPLEMENT A C PROGRAM TO PERFORM PALINDROME USING SLL

```
#include <stdio.h>
#include <stdlib.h>
struct node {
    int data;
    struct node *next;
};
int main() {
    struct node *head = NULL, *newnode, *temp;
    int n, i, arr[100], count = 0, flag = 1;
    printf("Enter number of nodes: ");
    scanf("%d", &n);
```

```
for(i = 0; i < n; i++) {
    newnode = (struct node*)malloc(sizeof(struct node));
    printf("Enter data for node %d: ", i + 1);
    scanf("%d", &newnode->data);
    newnode->next = NULL;

    if(head == NULL)
        head = temp = newnode;
    else {
        temp->next = newnode;
        temp = newnode;
    }
}
temp = head;
while(temp != NULL) {
    arr[count++] = temp->data;
    temp = temp->next;
}
for(i = 0; i < count / 2; i++) {
    if(arr[i] != arr[count - i - 1]) {
        flag = 0;
        break;
    }
```

```
    }
    if(flag == 1)
        printf("\nThe linked list is a Palindrome.\n");
    else
        printf("\nThe linked list is NOT a Palindrome.\n");

    return 0;
}
```

```
Output
Enter number of nodes: 3
Enter data for node 1: 2
Enter data for node 2: 1
Enter data for node 3: 2

The linked list is a Palindrome.

=== Code Execution Successful ===
```

EX.NO:25 IMPLEMENT A C PROGRAM TO INTERSECT SLL

```
#include <stdio.h>
#include <stdlib.h>
struct node {
    int data;
    struct node *next;
};
struct node* createList(int n) {
    struct node *head = NULL, *newnode, *temp;
    int i, val;
    for(i = 0; i < n; i++) {
```

```
        newnode = (struct node*)malloc(sizeof(struct node));
        printf("Enter data for node %d: ", i + 1);
        scanf("%d", &val);
        newnode->data = val;
        newnode->next = NULL;
        if(head == NULL)
            head = temp = newnode;
        else {
            temp->next = newnode;
            temp = newnode;
        }
    }
    return head;
}
int main() {
    struct node *head1 = NULL, *head2 = NULL, *temp1, *temp2;
    int n1, n2;
    printf("Enter number of nodes in 1st list: ");
    scanf("%d", &n1);
    head1 = createList(n1);
    printf("Enter number of nodes in 2nd list: ");
    scanf("%d", &n2);
    head2 = createList(n2);
```

```
    printf("\nIntersection elements: ");
    temp1 = head1;
    while(temp1 != NULL) {
        temp2 = head2;
        while(temp2 != NULL) {
            if(temp1->data == temp2->data)
                printf("%d ", temp1->data);
            temp2 = temp2->next;
        }
        temp1 = temp1->next;
    }
    return 0;
}
```

Output

```
Enter number of nodes in 1st list: 2
Enter data for node 1: 5
Enter data for node 2: 8
Enter number of nodes in 2nd list: 2
Enter data for node 1: 8
Enter data for node 2: 3

Intersection elements: 8

=== Code Execution Successful ===
```

EX.NO:26 IMPLEMENT A C PROGRAM TO PERFORM LINKED LIST – INSERTION

```
#include <stdio.h>
#include <stdlib.h>
struct node {
   int data;
   struct node *next;
```

```
};
int main() {
   struct node *head = NULL, *newnode, *temp;
   int n, i, val;
   printf("Enter number of nodes: ");
   scanf("%d", &n);
   for(i = 0; i < n; i++) {
      newnode = (struct node*)malloc(sizeof(struct node));
      printf("Enter data for node %d: ", i + 1);
      scanf("%d", &val);
      newnode->data = val;
      newnode->next = NULL;
      if(head == NULL)
         head = temp = newnode;
      else {
         temp->next = newnode;
         temp = newnode;
      }
   }
   newnode = (struct node*)malloc(sizeof(struct node));
   printf("Enter data to insert: ");
   scanf("%d", &newnode->data);
   newnode->next = NULL;
   temp->next = newnode;
   printf("\nLinked List after insertion: ");
   temp = head;
   while(temp != NULL) {
      printf("%d ", temp->data);
```

```
        temp = temp->next;
    }
    return 0;
}
```

```
Output

Enter number of nodes: 3
Enter data for node 1: 14
Enter data for node 2: 16
Enter data for node 3: 18
Enter data to insert: 20

Linked List after insertion: 14 16 18 20

=== Code Execution Successful ===
```

EX.NO:27 IMPLEMENT A C PROGRAM TO REVERSE – SLL

```
#include <stdio.h>
#include <stdlib.h>
struct node {
    int data;
    struct node *next;
};
int main() {
    struct node *head = NULL, *newnode, *temp;
    struct node *prev = NULL, *next = NULL, *current;
    int n, i;
    printf("Enter number of nodes: ");
    scanf("%d", &n);
    for(i = 0; i < n; i++) {
```

```
    newnode = (struct node*)malloc(sizeof(struct node));
    printf("Enter data for node %d: ", i + 1);
    scanf("%d", &newnode->data);
    newnode->next = NULL;
    if(head == NULL)
        head = temp = newnode;
    else {
        temp->next = newnode;
        temp = newnode;
    }
}
printf("\nOriginal Linked List: ");
temp = head;
while(temp != NULL) {
    printf("%d ", temp->data);
    temp = temp->next;
}
current = head;
while(current != NULL) {
    next = current->next;
    current->next = prev;
    prev = current;
    current = next;
}
head = prev;
printf("\nReversed Linked List: ");
temp = head;
while(temp != NULL) {
```

```
        printf("%d ", temp->data);
        temp = temp->next;
    }
    return 0;
}
```

Output

```
Enter number of nodes: 3
Enter data for node 1: 5
Enter data for node 2: 90
Enter data for node 3: 8

Original Linked List: 5 90 8
Reversed Linked List: 8 90 5

=== Code Execution Successful ===
```