[Name:Valaki Jaymin D]
[RollNo: MA075]
[Date of Experiment: 29/03/2023]
[

Practical-6
Singly Linked List Operations and Applications

HEADER FLE

NAME: Linklist.h

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
   char data;
   struct node *next;
} *head = NULL, *temp, *newnode;

void create()
{
   newnode = (struct node *)malloc(sizeof(struct node));
   if (newnode == NULL)
   {
      printf("No memory\n");
      exit(0);
   }
   newnode->next = NULL;
   scanf(" %c", &newnode->data);
}

void insertFirst()
{
   create();
   if (head == NULL)
   {
      head = newnode;
      return;
   }
   newnode->next = head;
   head = newnode;
}

void insertEnd()
{
   create();
   if (head == NULL)
```

```
    {
        head = newnode;
        return;
    }
    temp = head;
    while (temp->next != NULL)
    {
        temp = temp->next;
    }
    temp->next = newnode;
}

void insertSpecific(int pos)
{
    struct node *pred;
    int count=1;
    if (pos == 1)
    {
        insertFirst();
    }
    else
    temp = head;
    {
        while (temp->next != NULL && count != pos)  //insertend condtion check
        {
            count++;
            temp = temp->next;
        }
        create();
        newnode->next = temp->next;
        temp->next = newnode;
    }
}

void deletefirst()
{

    if(head==NULL)
    {
        printf("list is empty\n");
    }

    else{
        temp=head;
        head=temp->next;
        free(temp);
    }
```

```
}

void deleteend()
{
   struct node *previous;
     if(head==NULL)
    {
        printf("list is empty\n");
    }
    else{
        temp=head;

     while (temp->next != NULL)
    {
      previous = temp;
      temp = temp->next;
    }
    previous->next = NULL;
    free(temp);
    }
}

void deletespecific(int pos)
{
    int count=1;
    if(pos==1)
    {
        deletefirst();
    }
    else{
        temp=head;
        while(temp->next!=NULL && count!=pos)
        {
            count++;
            temp=temp->next;
        }
            temp->next=temp->next->next;
    }
}

void display()
{
    if (head == NULL)
    {
        printf("List is empty\n");
    }
    else
    {
        temp = head;
        while (temp != NULL)
        {
            printf("%c-> ", temp->data);
            temp = temp->next;
        }
        printf("\n");
    }
}


int main()
{
    insertFirst();
    insertFirst();
    insertFirst();
    insertEnd();
    insertSpecific(3);
    display();
    deleteend();
    display();
    return 0;
}
```

1. Write program for all operations of singly link list. (store character value in list)

- Creation of List
- Inserting Node – as First Node, as Last Node, at desired location
- Deleting Node – at First, at Last, Specific Node
- Display List

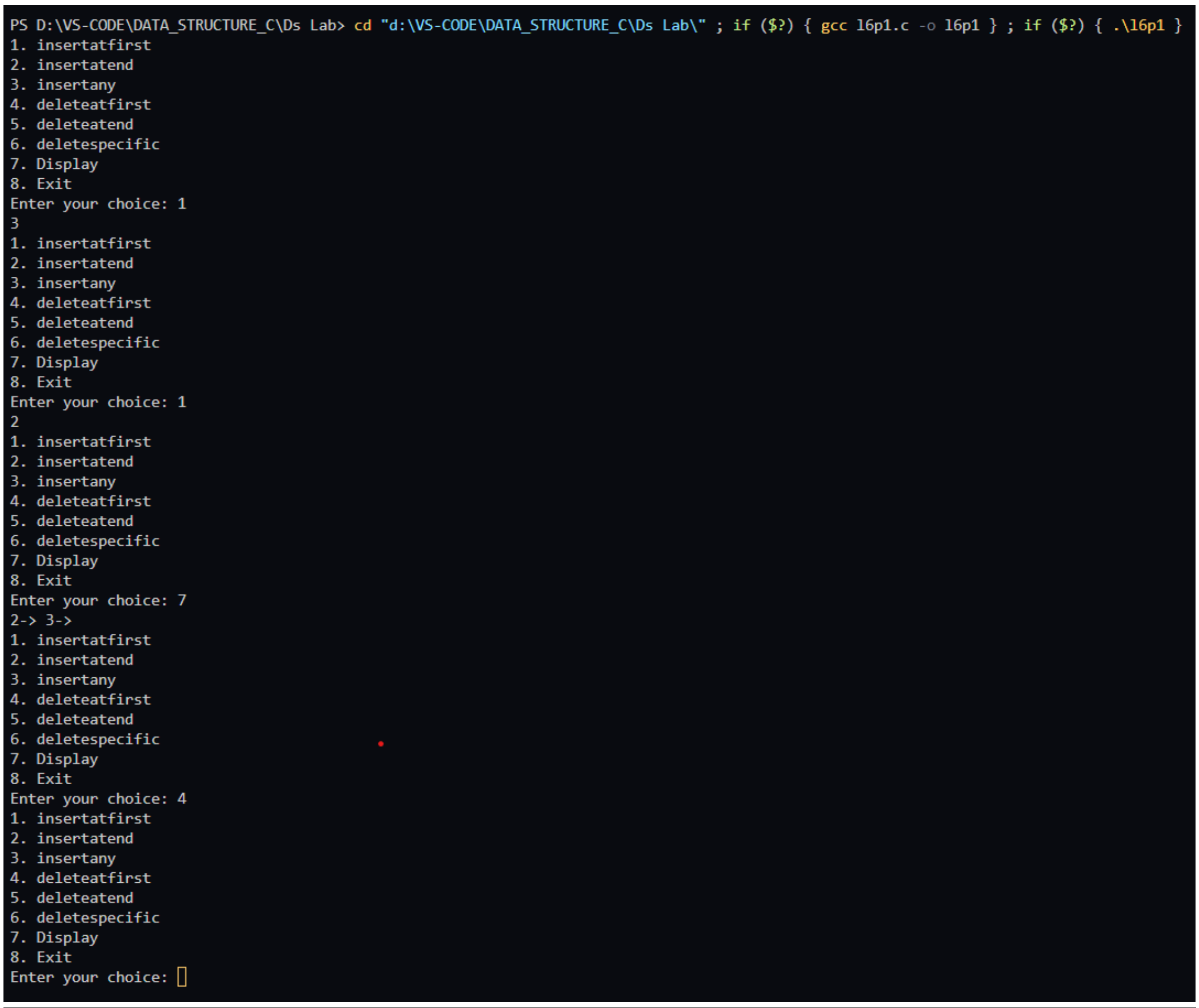
Source Code:-

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

int main() {

    int choice;
    int pos;
```

```
while (1) {
    printf("1. insertatfirst\n");
    printf("2. insertatend\n");
    printf("3. insertany\n");
    printf("4. deleteatfirst\n");
    printf("5. deleteatend\n");
    printf("6. deletespecific\n");
    printf("7. Display\n");
    printf("8. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);


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

        case 2:
            insertEnd();
            break;

        case 3:
            printf("give position:\n");
            scanf("%d",&pos);
            insertSpecific(pos);
            // display();
            break;
        case 4:
              deletefirst();
            break;
        case 5:
              deleteend();
            break;

        case 6:
            printf("give position:\n");
            scanf("%d",&pos);
            deletespecific(pos);
            display();
```

```
          break
        case 7:
                display();
                break;
        case 8:
            exit(0);
            break;
         default:
            printf("Invalid choice!!\n");
            break;
         }
      }
   }
```

Output:-



```
PS D:\VS-CODE\DATA_STRUCTURE_C\Ds Lab> cd "d:\VS-CODE\DATA_STRUCTURE_C\Ds Lab\" ; if ($?) { gcc 16p1.c -o 16p1 } ; if ($?) { .\16p1 }
1. insertatfirst
2. insertatend
3. insertany
4. deleteatfirst
5. deleteatend
6. deletespecific
7. Display
8. Exit
Enter your choice: 1
3
1. insertatfirst
2. insertatend
3. insertany
4. deleteatfirst
5. deleteatend
6. deletespecific
7. Display
8. Exit
Enter your choice: 1
2
1. insertatfirst
2. insertatend
3. insertany
4. deleteatfirst
5. deleteatend
6. deletespecific
7. Display
8. Exit
Enter your choice: 7
2-> 3->
1. insertatfirst
2. insertatend
3. insertany
4. deleteatfirst
5. deleteatend
6. deletespecific
7. Display
8. Exit
Enter your choice: 4
1. insertatfirst
2. insertatend
3. insertany
4. deleteatfirst
5. deleteatend
6. deletespecific
7. Display
8. Exit
Enter your choice: 
```

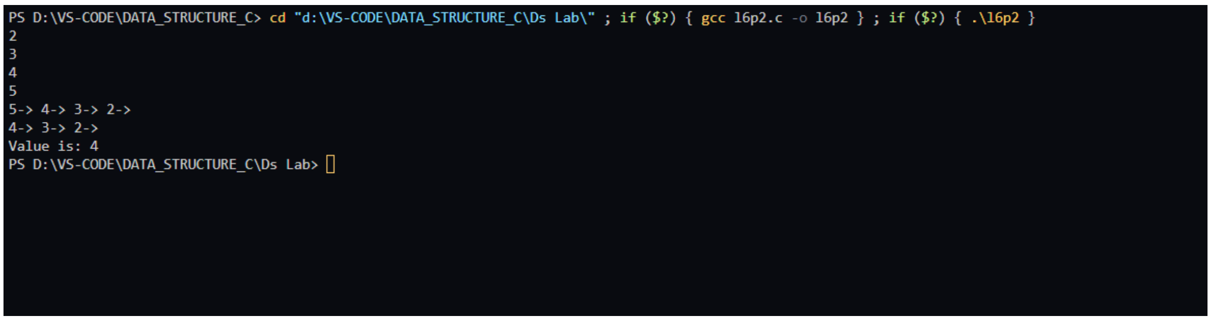
2. Write an algorithm and implement program to perform all stack operations using singly link list. *Implement PUSH, POP, PEEP, Change and DISPLAY.*

Source code :

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

int peep();
void change(int x,int y);

int main()
{
   insertFirst();
   insertFirst();
   insertFirst();
   insertFirst();
   display();
   deletefirst();
   display();
   peep();
   change(2,4);
   return 0;
}

int peep()
{
   if (head == NULL)
   {
      printf("Stack is empty\n");
      return -1;
   }
   printf("Value is: %c\n",head->data);
   return head->data;

}
```

```
void change(int pos,int val)
{
        int cnt = 1;
        temp=head;
        while(temp->next!=NULL&&cnt!=pos)
        {
                cnt++;
                temp=temp->next;
        }
        temp->data=val;
}
```

Output:



```
PS D:\VS-CODE\DATA_STRUCTURE_C> cd "d:\VS-CODE\DATA_STRUCTURE_C\Ds Lab\" ; if ($?) { gcc 16p2.c -o 16p2 } ; if ($?) { .\16p2 }
2
3
4
5
5-> 4-> 3-> 2->
4-> 3-> 2->
Value is: 4
PS D:\VS-CODE\DATA_STRUCTURE_C\Ds Lab> 
```

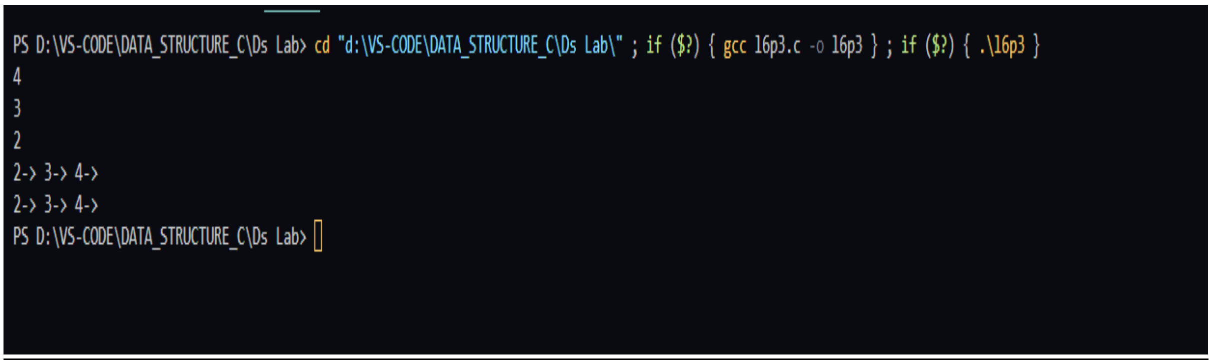
3. Write a program to perform sort on an integer linked list.

Source code:

```
#include<stdio.h>
#include<stdlib.h>
#include"linklist.h"
void sortList();

int main()
{
  struct node *tail=NULL;
        insertFirst();
   insertFirst();
   insertFirst();
   display();
        sortList();
        display();
        return 0;}
```

```
void sortList()
{
            struct node *temp=head, *temp1=NULL;
    //temp1 is temprory
            int item;
            if(head==NULL)
            {
                        return;
            }

            else
            {
                        while(temp!=NULL)
                        {
                                    temp1=temp->next;
                                    while(temp1!=NULL)
                                    {
                                                if(temp->data > temp1->data)
                                                {
                                                            item=temp->data;
                                                            temp->data= temp1->data;
                                                            temp1->data=item;
                                                }
                                                temp1=temp1->next;
                                    }
                                    temp=temp->next;
}
}
}
```

Output:



```
PS D:\VS-CODE\DATA_STRUCTURE_C\Ds Lab> cd "d:\VS-CODE\DATA_STRUCTURE_C\Ds Lab\" ; if ($?) { gcc 16p3.c -o 16p3 } ; if ($?) { .\16p3 }
4
3
2
2-> 3-> 4->
2-> 3-> 4->
PS D:\VS-CODE\DATA_STRUCTURE_C\Ds Lab>
```