

```

1  #include<stdio.h>
2  #include<stdlib.h>
3  #include<malloc.h>
4  struct node
5  {
6  int data;
7  struct node * next;
8
9  };
10 struct node *head=NULL;
11 void stacksinglypush();
12 void stacksinglypop();
13 void stackdisplay();
14
15 void stacksinglypush()
16 {
17 struct node *newnode;
18 newnode=(struct node*)malloc(sizeof(struct node));
19 printf("enter data:");
20 scanf("%d",&newnode->data);
21 newnode->next=NULL;
22 if(head==NULL)
23 {
24 head=newnode;
25 }
26 else
27 {
28 newnode->next=head;
29 head=newnode;
30 }
31 }
32
33 void stacksinglypop()
34 {
35 {
36 struct node*temp;
37
38 if(head==NULL)
39 {
40 printf("list is empty");
41 }
42 else if(head->next==NULL)
43 {
44 head=NULL;
45 //free(temp);
46 printf("\n stack deleted\n");
47 }
48 else
49 {
50 temp=head;
51 head=head->next;
52 free(temp);
53 //printf("\n begin node deleted\n");
54 }
55 }
56
57
58 }
59
60 void stackdisplay()
61 {
62 struct node *temp;
63 if(head==NULL)
64 {
65 printf("list is empty\n");
66 }

```

```

67 else
68 {
69 temp=head;
70 printf("List elements\n");
71 while(temp!=NULL)
72 {
73 printf("%d\t",temp->data);
74 temp=temp->next;
75 }
76 }
77 }
78
79 int main()
80 {
81     int op;
82     while(op !=4)
83     {
84         int op;
85         printf("\n*****manu*****\n");
86         printf("\n 1->stack as singly push\n");
87         printf("\n 2->stack as singly pop\n");
88         printf("\n 3->stack as singly display\n");
89         printf("\n 4->exit\n");
90         printf("\n enter your choice\n");
91         scanf("%d",&op);
92         switch(op)
93         {
94             case 1:
95                 stacksinglypush();
96                 break;
97             case 2:
98                 stacksinglypop();
99                 break;
100             case 3:
101                 stackdisplay();
102                 break;
103             case 4:
104                 exit(0);
105             default:
106                 printf("\n incorrect choice\n");
107             }
108         }
109         return 0;
110     }

```