```
#include<stdio.h>
#include<stdlib.h>
struct node
{
int info;
struct node *ptr;
}*top,*top1,*temp;

void push(int data);
void pop();
void display();
void create();


int main()
{
int no, ch, e;

printf("\n 1 - Push");
printf("\n 2 - Pop");
printf("\n 3 - Dipslay");
printf("\n 4 - Exit");

create();

while (1)
{
printf("\n Enter choice : ");
scanf("%d", &ch);

switch (ch)
{
```

```
case 1:
printf("Enter data : ");
scanf("%d", &no);
push(no);
break;
case 2:
pop();
break;
case 3:
display();
break;
case 4:
exit(0);
default :
printf(" Wrong choice, Please enter correct choice  ");
}
}
}


void create()
{
top = NULL;
}


void push(int data)
{
if (top == NULL)
{
top =(struct node *)malloc(1*sizeof(struct node));
top->ptr = NULL;
top->info = data;
}
```

```
else
{
temp =(struct node *)malloc(1*sizeof(struct node));
temp->ptr = top;
temp->info = data;
top = temp;
}

}


void display()
{
top1 = top;

if (top1 == NULL)
{
printf("Stack is empty");
return;
}

while (top1 != NULL)
{
printf("%d ", top1->info);
top1 = top1->ptr;
}
}

void pop()
{
top1 = top;

if (top1 == NULL)
{
```

```
printf("\n Error : Trying to pop from empty stack");
return;
}
else
top1 = top1->ptr;
printf("\n Popped value : %d", top->info);
free(top);
top = top1;

}
```

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

struct node
{
int info;
struct node *ptr;
}*front,*rear,*temp,*front1;

void enq(int data);
void deq();

void display();
void create();

int main()
{
int no, ch, e;

printf("\n 1 - Enque");
printf("\n 2 - Deque");
printf("\n 3 - Display");
printf("\n 4 - Exit");
create();
while (1)
{
printf("\n Enter choice : ");
scanf("%d", &ch);
switch (ch)
```

```
{
case 1:
printf("Enter data : ");
scanf("%d", &no);
enq(no);
break;
case 2:
deq();
break;
case 3:
display();
break;
case 4:
exit(0);
default:
printf("Wrong choice, Please enter correct choice  ");
break;
}
}
return 0;
}


void create()
{
front = rear = NULL;
}


void enq(int data)
{
if (rear == NULL)
{
rear = (struct node *)malloc(1*sizeof(struct node));
```

```
rear->ptr = NULL;
rear->info = data;
front = rear;
}
else
{
temp=(struct node *)malloc(1*sizeof(struct node));
rear->ptr = temp;
temp->info = data;
temp->ptr = NULL;

rear = temp;
}

}


void display()
{
front1 = front;

if ((front1 == NULL) && (rear == NULL))
{
printf("Queue is empty");
return;
}
while (front1 != rear)
{
printf("%d ", front1->info);
front1 = front1->ptr;
}
if (front1 == rear)
printf("%d", front1->info);
}

```

```
void deq()
{
front1 = front;

if (front1 == NULL)
{
printf("\n queue is empty");
return;
}
else
if (front1->ptr != NULL)
{
front1 = front1->ptr;
printf("\n Dequed value : %d", front->info);
free(front);
front = front1;
}
else
{
printf("\n Dequed value : %d", front->info);
free(front);
front = NULL;
rear = NULL;
}

}
```

```
 1 - Enque
 2 - Deque
 3 - Display
 4 - Exit
 Enter choice : 1
Enter data : 22

 Enter choice : 1
Enter data : 33

 Enter choice : 1
Enter data : 55

 Enter choice : 2

 Dequed value : 22
 Enter choice : 3
33 55
 Enter choice : 4


...Program finished with exit code 0
Press ENTER to exit console.
```