4) Circular Queue Program

```
#include <stdio.h>
#include <stdlib.h>
#define S 3
int front = -1;
int rear = -1;
int queue[5];
void Enque (int, int);
int Deque (int);
void display (int);
int main () {
int choice, SIZE;
int item;
printf ("Enter the size of queue : \n");
scanf ("%d", &SIZE);
do {
printf ("\n<---- Circular Queue ----> \n");
printf ("\n 1. Insert to Queue (Enqueue)");
printf ("\n 2. Delete from Queue (DeQueue)");
printf ("\n 3. Display the contents ");
printf ("\n 4. Exit\n");
printf ("\n<---- ---------- ---->\n");
printf ("Enter your choice: ");
scanf ("%d", &choice);
switch (choice) {
case 1: if (((front == 0 && rear == SIZE-1)) || (front == rear+1)) {
        printf ("Queue is Full !! \n");
        break;
        }
    printf ("Enter the element you want to Insert : \n");
```

```
Scanf("%d", &item);
Enque(SIZE, item);
break;
case 2: item = Deque(SIZE);
if (item == -999)
printf("Queue is Empty !! \n");
else
printf("\n Removed element from the queue %d \n", item);
break;
case 3: display(SIZE);
break;
case 4: printf("Exiting.... \n");
exit(0);
default: printf("Invalid Choice");
break;
}
} while (choice != 4);
return 0;
}

void Enque (int SIZE, int ele) {
if (((front == 0 && rear == SIZE-1)) || (front == rear+1)){
printf("--- Queue is full .... \n");
return;
}else {
rear = (rear + 1) % SIZE;
queue[rear] = ele;
if (front == -1)
front = 0;
}
}
int Deque (int SIZE) {
```

```
int item;
if ((front == -1) && (rear == -1)) {
return (-999);
} else {
item = queue[front];
if (front == rear){
 front = -1;
  rear = -1;
}else{
front =(front +1) % SIZE;
}
return item;
}
}
void display(int SIZE) {
int i;
if (((front == -1) && (rear == -1))) {
printf("--- Queue is Empty ----\n");
return;
} else {
printf("\n Queue Contents: \n");
for (i = front; i != rear; i =(i+1) % SIZE)
printf("%d \t", queue[i]);
printf("%d \t", queue[i]);
}
}
```