Stack:

DSARRAY.c

```c
#include<stdio.h>
#include<conio.h>
int stack[100];
int top = -1;
int size;
void push(){
    int value;
    printf("Enter Value: ");
    scanf("%d",&value);
    if(top >= size - 1){
        printf("Stack Is Full. Cannot Push.\n");
        return;
    }
    stack[++top] = value;
    printf("Pushed %d onto the stack.\n",value);
}
void pop(){
    if(top < 0){
        printf("Stack is Empty. Cannot Pop.\n");
        return;
    }
    int popvalue = stack[top--];
    printf("Poped %d from the stack.\n",popvalue);
}
void peek(){
    if(top < 0){
        printf("Stack Is Empty. Cannot Peek.\n");
        return;
    }
    printf("Top Element of the stack : %d\n",stack[top]);
}
```

DSARRAY.c

```c
void isEmpty(){
    if(top < 0){
        printf("Stack Is Empty.\n");
    }else{
        printf("Stack Is Not Empty.\n");
    }
}
void Fdisplay(){
    if(top < 0){
        printf("Stack Is Empty.\n");
        return;
    }
    printf("Stack Elements: \n");
    for(int i = top; i >= 0; i--){
        printf("%d ",stack[i]);
    }
    printf("\n");
}
```

```c
void Bdisplay(){
    if(top < 0){
        printf("Stack Is Empty.\n");
        return;
    }
    printf("Stack Element: \n");
    for(int i = 0; i <= top; i++){
        printf("%d ",stack[i]);
    }
    printf("\n");
}
```

```c
int main() {
    printf("Enter The Size Of The Stack : ");
    scanf("%d",&size);

    int choice;
    do{
        printf("Enter Choice For Operation : ");
        scanf("%d",&choice);
        switch(choice){
            case 1:
                push();
                break;
            case 2:
                pop();
                break;
            case 3:
                peek();
                break;
            case 4:
                isEmpty();
                break;
            case 5:
                Fdisplay();
                break;
            case 6:
                Bdisplay();
                break;
            default:
                printf("Invalid Choice!!!");
                break;
        }
    }while(choice != 0);
}
```

Queue:

DSARRAY.c

```c
#include<stdio.h>
#include<conio.h>
#include<stdbool.h>
int queue[100];
int front = -1;
int rear = -1;
void enqueue(){
    int element;
    printf("\nEnter Element For Queue : \n");
    scanf("%d",&element);
    if(rear == 100 - 1){
        printf("\nQueue Is Full. No Enqueue Process.\n");
    }
    if(front == -1){
        front = 0;
    }
    rear++;
    queue[rear] = element;
}
void dequeue(){
    if(front == -1 || front > rear){
        printf("\nQueue Is Empty\n");
    }
    int element = queue[front];
    front++;
    printf("\nRemoved Element Is : %d\n",element);
}
```

```c
void frontele(){
    if(front == -1){
        printf("\nQueue Is Empty.\n");
    }
    else{
        printf("\nFront Element : %d\n",queue[front]);
    }
}
void rearele(){
    if(front == -1){
        printf("\nQueue Is Empty.\n");
    }
    else{
        printf("\nRear Element : %d\n",queue[rear]);
    }
}
void display(){
    if(front == -1){
        printf("\nQueue Is Empty.\n");
    }
    for(int i = front; i <= rear; i++){
        printf("q[%d] : %d\n",i,queue[i]);
    }
    printf("\n");
}
```

```c
int main(){
    int choice;
    do{
        printf("\nEnter Choice For Operation : ");
        scanf("%d",&choice);
        switch(choice){
            case 1:
                enqueue();
                break;
            case 2:
                dequeue();
                break;
            case 3:
                frontele();
                break;
            case 4:
                rearele();
                break;
            case 5:
                display();
                break;
            default:
                printf("Invalid Options!!!");
                break;
        }
    }while(choice != 0);
}
```

Circular Queue:

DSARRAY.c

```c
#include<stdio.h>
#include<conio.h>
#include<stdbool.h>
int cqueue[6];
int front = -1;
int rear = -1;
void enqueue(){
    int element;
    printf("\nEnter Element : \n");
    scanf("%d",&element);
    if((front == rear + 1) || (front == 0 && rear == 6 - 1)){
        printf("\nQueue Is Full. No Enqueue Process.\n");
    }
    else{
        if(front == -1) front = 0;
        rear = (rear + 1) % 6;
        cqueue[rear] = element;
        printf("\nInserted Element : %d\n",element);
    }
}
void dequeue(){
    int element;
    if((front == -1) && (rear == -1)){
        printf("\nQueue Is Empty.\n");
    }
    element = cqueue[front];
    if(front == rear){
        front = rear -1;
    }else{
        front = (front + 1) % 6;
    }
    printf("\nDeleted Item Is : %d\n",element);
}
```

```c
void isfull(){
    if((front == rear + 1) || (front == 0 && rear == 6 - 1)){
        printf("\nCircular Queue Is Full.\n");
    }
}
void isempty(){
    if(front == -1){
        printf("\nCircular Queue Is Empty.\n");
    }
}
void frontele(){
    printf("\nCircular Queue Front Element : %d\n",cqueue[front]);
}
void rearele(){
    printf("\nCircular Queue Rear Element : %d\n",cqueue[rear]);
}
```

```c
void display(){
    if(front == -1){
        printf("\nCircular Queue Is Empty.\n");
    }int i = front;
    printf("\nCircular Queue Is : \n");
    while(i != rear){
        printf("%d ",cqueue[i]);
        i = (i + 1) % 6;
    }
    printf("%d ",cqueue[rear]);
}
void nextele(){
    if((front == -1) && (rear == -1)){
        printf("\nQueue Is Empty.\n");
    }else{
        printf("\nCurrent Element : %d\n",cqueue[rear]);
        int next = (front) % 6;
        printf("Next Element Is : %d\n",cqueue[next]);
    }
}
```

```c
int main() {
    int choice;
    do {
        printf("\nEnter Choice For Operation : ");
        scanf("%d",&choice);
        switch(choice) {
            case 1:
                enqueue();
                break;
            case 2:
                dequeue();
                break;
            case 3:
                isfull();
                break;
            case 4:
                isempty();
                break;
            case 5:
                frontele();
                break;
            case 6:
                rearele();
                break;
            case 7:
                display();
                break;
            case 8:
                nextele();
                break;
            default:
                printf("\nInvalid Option!!!\n");
                break;
        }
    } while(choice != 0);}
```