SOUTHEAST UNIVERSITY

Meeting the Challenges of Time

Department Of CSE

Cse Assignment

Assignment No:06

Date of submission:

Course name : CSE LAB

Course code : 241

Section : 9

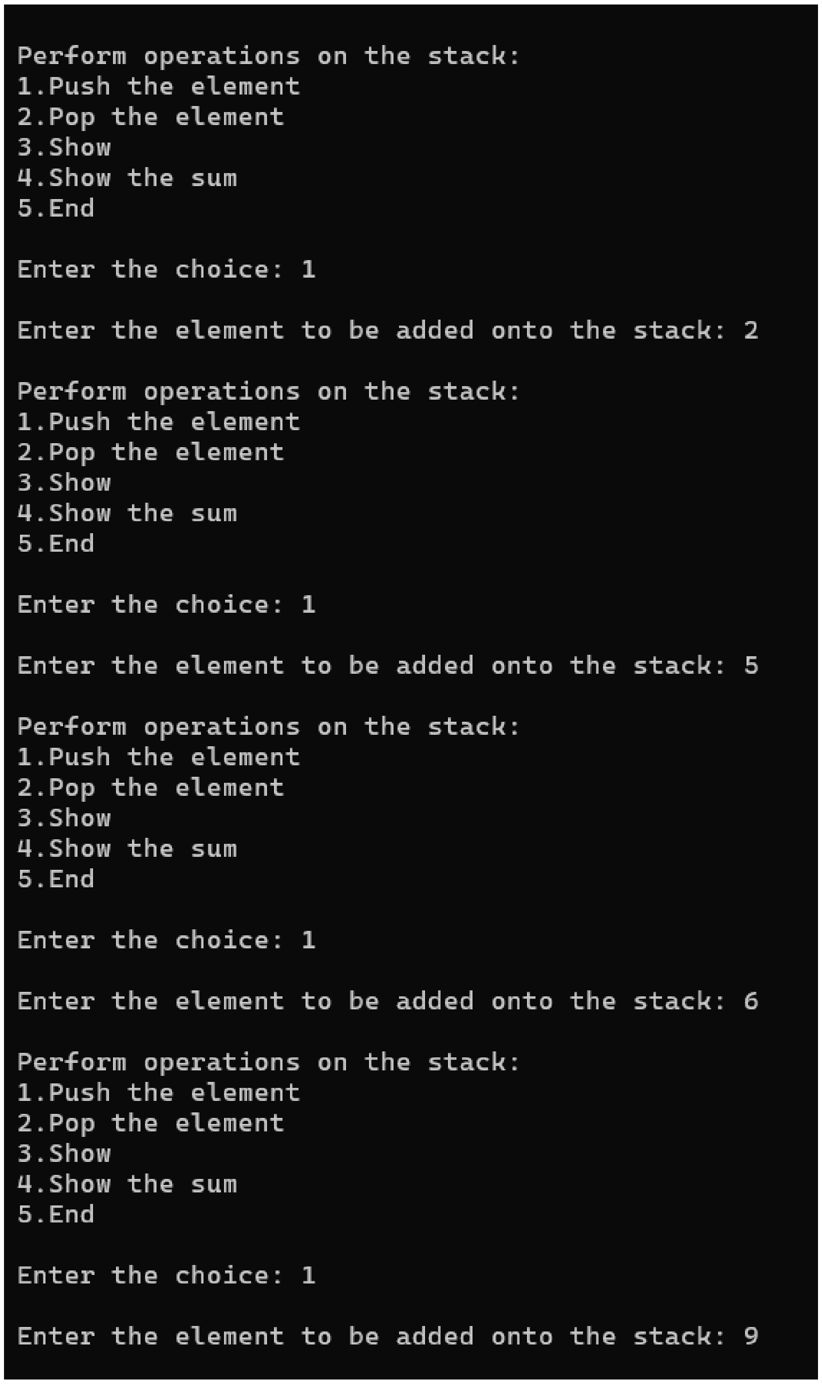
Student's Name	**:Nazmul Hasan**
Student's ID	**:2023100000130**
Submitted To	**:Mr. Muhammed Yeaseen Morshed Abid**
	[lecturer, Department of CSE]
Initial	**: YMA**

Code 1:

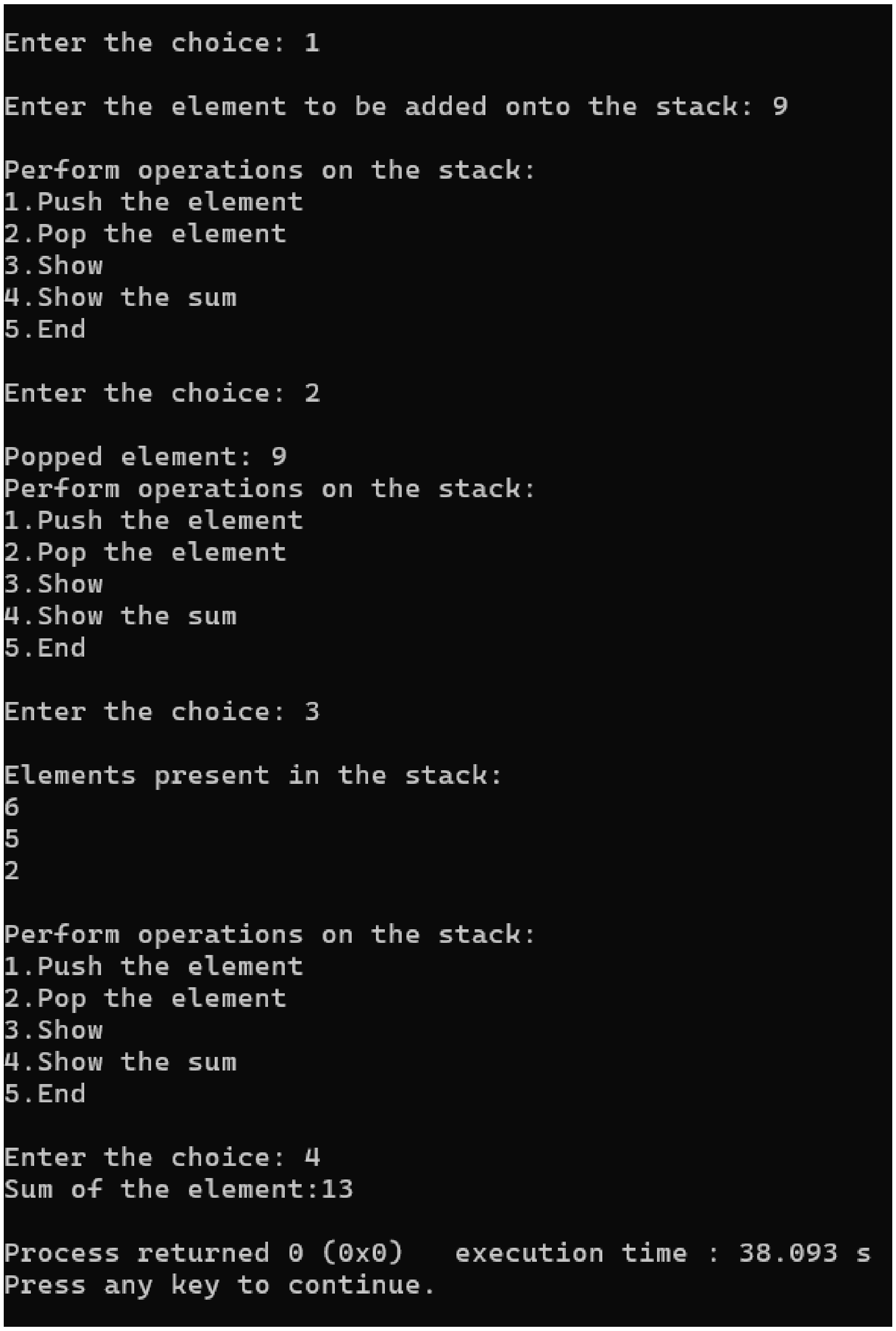
```
#include <stdio.h>

#include <stdlib.h>


#define SIZE 5


int top = -1, inp_array[SIZE];

void push();

void pop();

void show();

void sum();


int main()

{

   int choice;


   while (1)

   {

      printf("\nPerform operations on the stack:");

      printf("\n1.Push the element\n2.Pop the element\n3.Show\n4.Show the sum\n5.End");

      printf("\n\nEnter the choice: ");

      scanf("%d", &choice);


      switch (choice)

      {

      case 1:

         push();
```

```
            break;
        case 2:
            pop();
            break;
        case 3:
            show();
            break;
        case 4:
            sum();
        case 5:
            exit(0);

        default:
            printf("\nInvalid choice!!");
        }
    }
}

void push()
{
    int x;

    if (top == SIZE - 1)
    {
        printf("\nOverflow!!");
    }
    else
    {
        printf("\nEnter the element to be added onto the stack: ");
        scanf("%d", &x);
        top = top + 1;
        inp_array[top] = x;
    }
}

void pop()
{
    if (top == -1)
    {
        printf("\nUnderflow!!");
    }
    else
    {
        printf("\nPopped element: %d", inp_array[top]);
        top = top - 1;
    }
}

void show()
{
    if (top == -1)
    {
        printf("\nUnderflow!!");
    }
    else
    {
        printf("\nElements present in the stack: \n");
        for (int i = top; i >= 0; --i)
            printf("%d\n", inp_array[i]);
    }
}
void sum()
{
    int sum = 0;
    for(int i = top; i >= 0; i--){
        sum = sum + inp_array[i];
    }
    printf("Sum of the element:%d\n", sum);
}
```

Terminal:



```
Perform operations on the stack:
1.Push the element
2.Pop the element
3.Show
4.Show the sum
5.End

Enter the choice: 1

Enter the element to be added onto the stack: 2

Perform operations on the stack:
1.Push the element
2.Pop the element
3.Show
4.Show the sum
5.End

Enter the choice: 1

Enter the element to be added onto the stack: 5

Perform operations on the stack:
1.Push the element
2.Pop the element
3.Show
4.Show the sum
5.End

Enter the choice: 1

Enter the element to be added onto the stack: 6

Perform operations on the stack:
1.Push the element
2.Pop the element
3.Show
4.Show the sum
5.End

Enter the choice: 1

Enter the element to be added onto the stack: 9
```



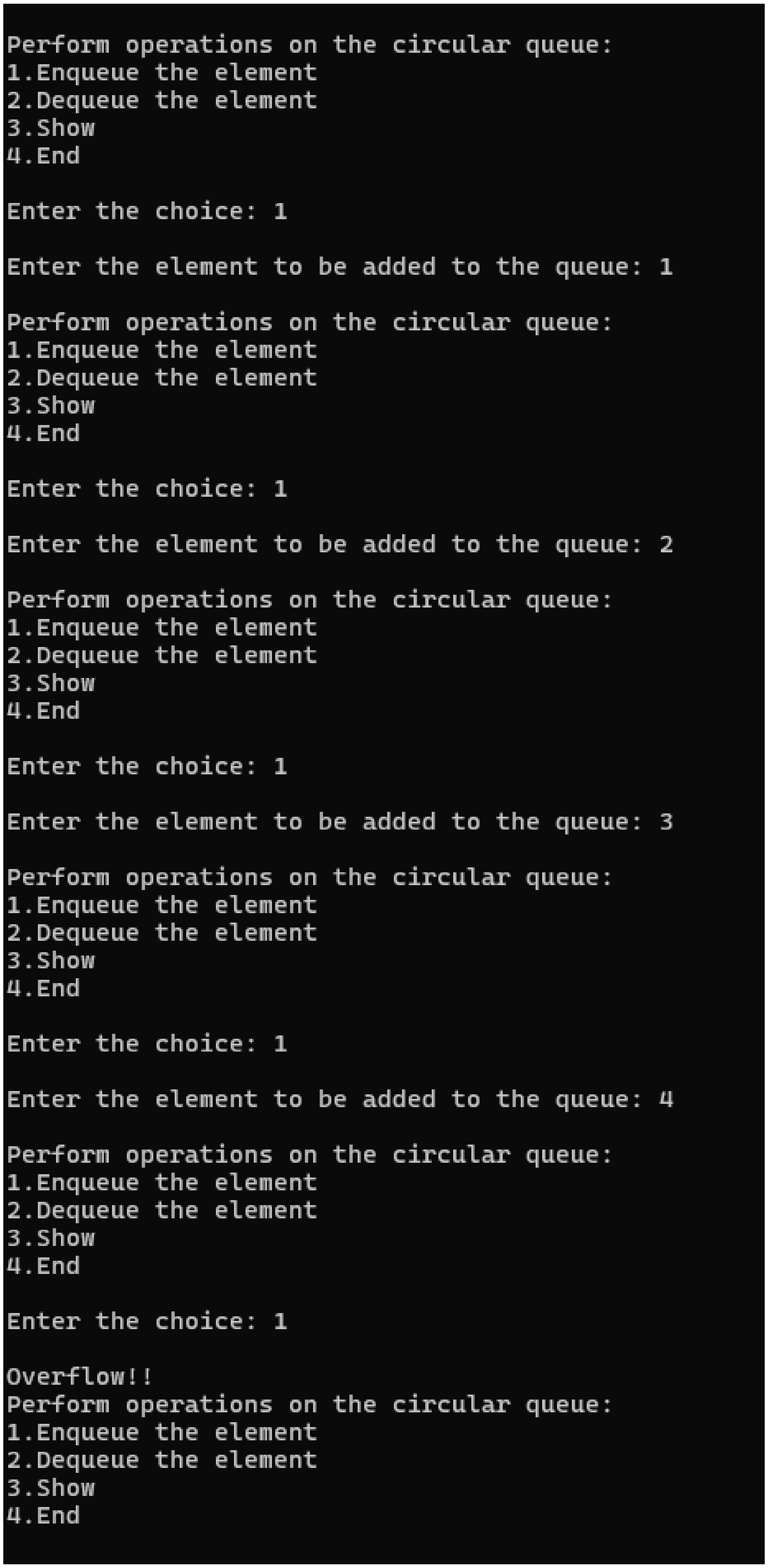
```
Enter the choice: 1

Enter the element to be added onto the stack: 9

Perform operations on the stack:
1.Push the element
2.Pop the element
3.Show
4.Show the sum
5.End

Enter the choice: 2

Popped element: 9
Perform operations on the stack:
1.Push the element
2.Pop the element
3.Show
4.Show the sum
5.End

Enter the choice: 3

Elements present in the stack:
6
5
2

Perform operations on the stack:
1.Push the element
2.Pop the element
3.Show
4.Show the sum
5.End

Enter the choice: 4
Sum of the element:13

Process returned 0 (0x0)   execution time : 38.093 s
Press any key to continue.
```

Code 2:

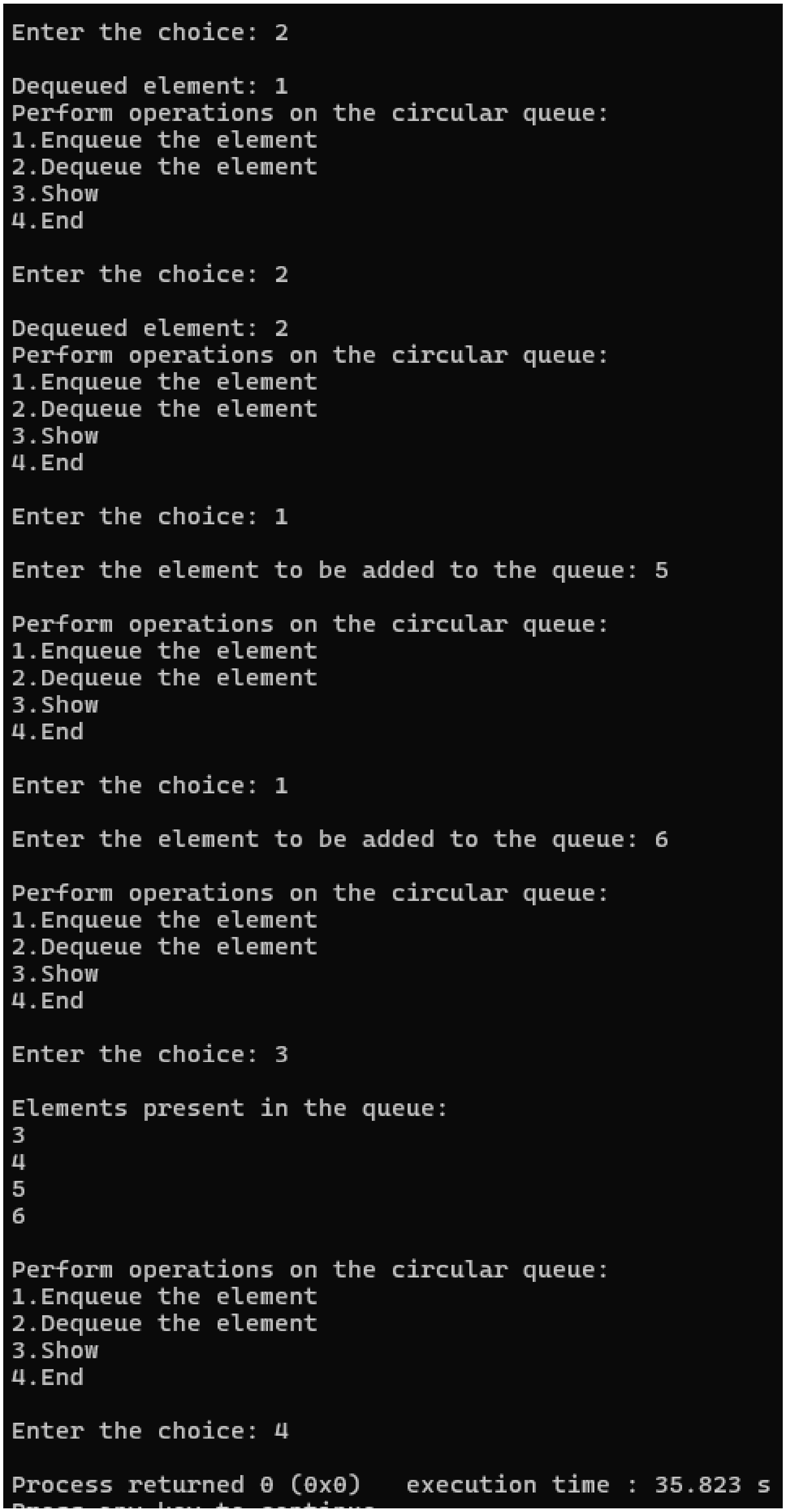
```
#include <stdio.h>
#include <stdlib.h>

#define SIZE 4

int front = -1, rear = -1, inp_array[SIZE];

void enqueue();
void dequeue();
void show();

int main()
{
    int choice;

    while (1)
    {
        printf("\nPerform operations on the circular queue:");
        printf("\n1.Enqueue the element\n2.Dequeue the element\n3.Show\n4.End");
        printf("\n\nEnter the choice: ");
        scanf("%d", &choice);

        switch (choice)
        {
        case 1:
            enqueue();
            break;
        case 2:
            dequeue();
            break;
        case 3:
            show();
            break;
        case 4:
            exit(0);

        default:
            printf("\nInvalid choice!!");
        }
    }
}

void enqueue()
```

```
{
    int x;

    if ((rear + 1) % SIZE == front)
    {
        printf("\nOverflow!!");
    }
    else
    {
        printf("\nEnter the element to be added to the queue: ");
        scanf("%d", &x);
        if (front == -1 && rear == -1)
        {
            front = rear = 0;
        }
        else
        {
            rear = (rear + 1) % SIZE;
        }
        inp_array[rear] = x;
    }
}

void dequeue()
{
    if (front == -1 && rear == -1)
    {
        printf("\nUnderflow!!");
    }
    else if (front == rear)
    {
        printf("\nDequeued element: %d", inp_array[front]);
        front = rear = -1;
    }
    else
    {
        printf("\nDequeued element: %d", inp_array[front]);
        front = (front + 1) % SIZE;
    }
}

void show()
{
    if (front == -1 && rear == -1)
    {
        printf("\nQueue is empty!!");
    }
    else
    {
        printf("\nElements present in the queue: \n");
        int i = front;
        while (1)
        {
            printf("%d\n", inp_array[i]);
            if (i == rear)
                break;
            i = (i + 1) % SIZE;
        }
    }
}
```

Terminal:



```
Perform operations on the circular queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.End

Enter the choice: 1

Enter the element to be added to the queue: 1

Perform operations on the circular queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.End

Enter the choice: 1

Enter the element to be added to the queue: 2

Perform operations on the circular queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.End

Enter the choice: 1

Enter the element to be added to the queue: 3

Perform operations on the circular queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.End

Enter the choice: 1

Enter the element to be added to the queue: 4

Perform operations on the circular queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.End

Enter the choice: 1

Overflow!!
Perform operations on the circular queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.End
```



```
Enter the choice: 2

Dequeued element: 1
Perform operations on the circular queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.End

Enter the choice: 2

Dequeued element: 2
Perform operations on the circular queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.End

Enter the choice: 1

Enter the element to be added to the queue: 5

Perform operations on the circular queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.End

Enter the choice: 1

Enter the element to be added to the queue: 6

Perform operations on the circular queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.End

Enter the choice: 3

Elements present in the queue:
3
4
5
6

Perform operations on the circular queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.End

Enter the choice: 4

Process returned 0 (0x0)   execution time : 35.823 s
```