Practice Problem 6

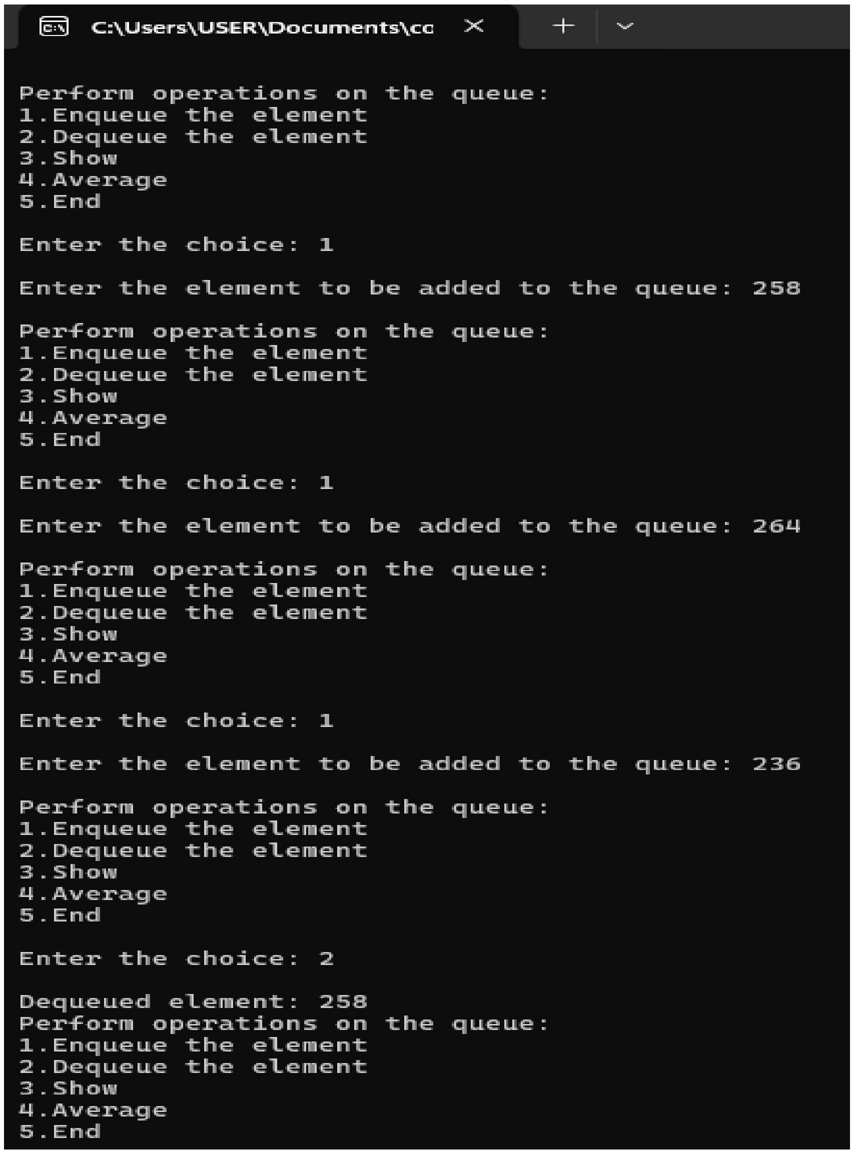
Data Structure

Topic Queue

ID 2023000000033

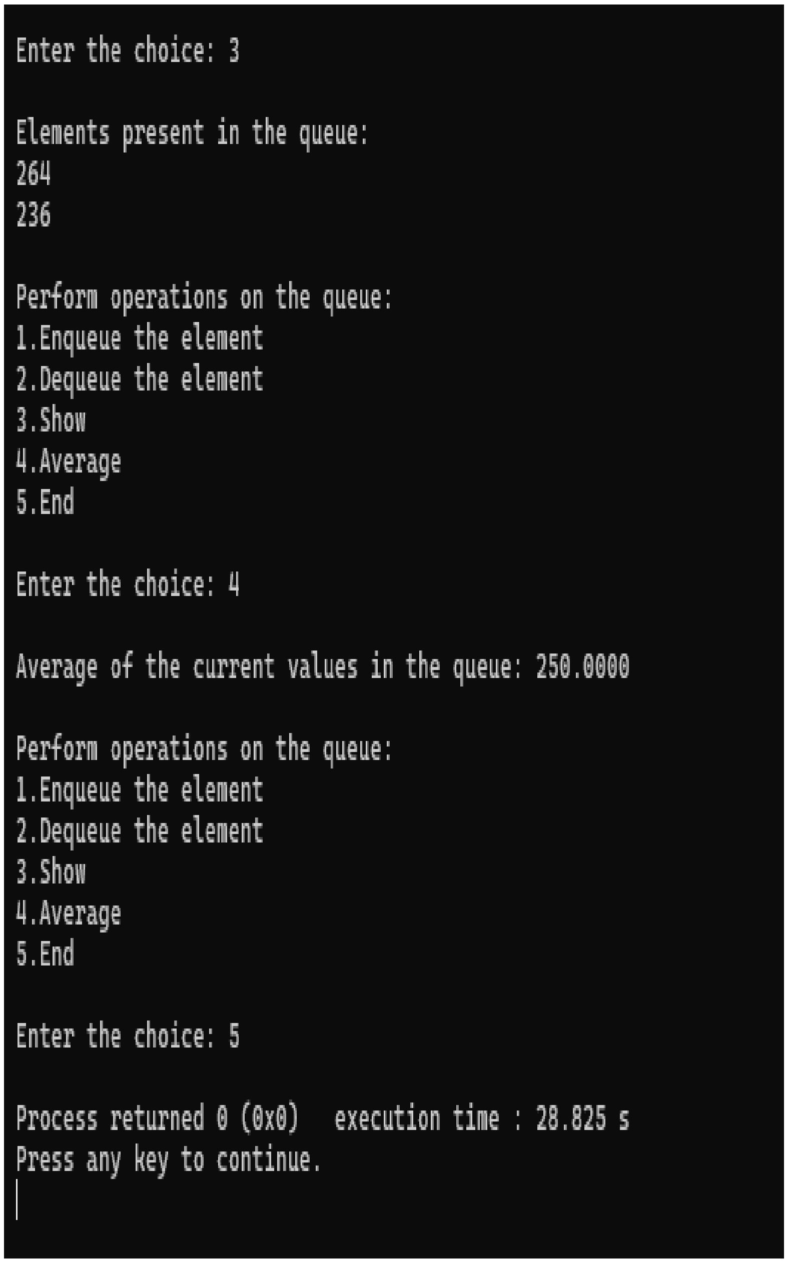
1. Queue implementation code is given in your Google classroom. Modify the code in such a way so that it provides an option Average. When someone chooses this option, your code should provide the average of the current values of the queue. Solution:

```
#include <stdio.h>
#include <stdlib.h>
#define SIZE 4
int front = -1, rear = -1, inp_array[SIZE];
void enqueue();    void dequeue();    void show();    void average();
int main() {
   int choice;
   while (1) {
      printf("\nPerform operations on the queue:");
      printf("\n1.Enqueue the element\n2.Dequeue the element\n3.Show\n4.Average\n5.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:
        average();
         break;
      case 5:
            exit(0);
         default:
            printf("\nInvalid choice!!");
      }
 }
 }
void enqueue()
{
   int x;
   if (rear == SIZE - 1)
   {
      printf("\nOverflow!!");
   }
   else
   {
```



```
C:\Users\USER\Documents\cc

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

Enter the choice: 1

Enter the element to be added to the queue: 258

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

Enter the choice: 1

Enter the element to be added to the queue: 264

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

Enter the choice: 1

Enter the element to be added to the queue: 236

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

Enter the choice: 2

Dequeued element: 258
Perform operations on the queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.Average
5.End
```

```
        printf("\nEnter the element to be added to the
queue: ");
        scanf("%d", &x);
        if (front == -1)
            front = 0;
        rear = rear + 1;
        inp_array[rear] = x;
    }
}
void dequeue()
{
    if (front == -1 || front > rear)
    {
       printf("\nUnderflow!!");
    }
    else
    {
        printf("\nDequeued element: %d", inp_array[front]);
        front = front + 1;
    }
 }
void show()
{
    if (front == -1 || front > rear)
    {
        printf("\nQueue is empty!!");
    }
    else
    {
        printf("\nElements present in the queue: \n");
        for (int i = front; i <= rear; i++)
            printf("%d\n", inp_array[i]);
      }
}
 void average()
 {
    if (front == -1) {
        printf("\nQueue is empty!!");
    } else {
        int sum = 0;
        int count = 0;
        int i =  front;

        while (1) {
            sum += inp_array[i];
            count++;
            if (i == rear) break;
            i = (i + 1) ;  }
        double avg = (double)sum / count;
        printf("\nAverage of the current values in the queue: %.4f\n", avg);  }
 }
```



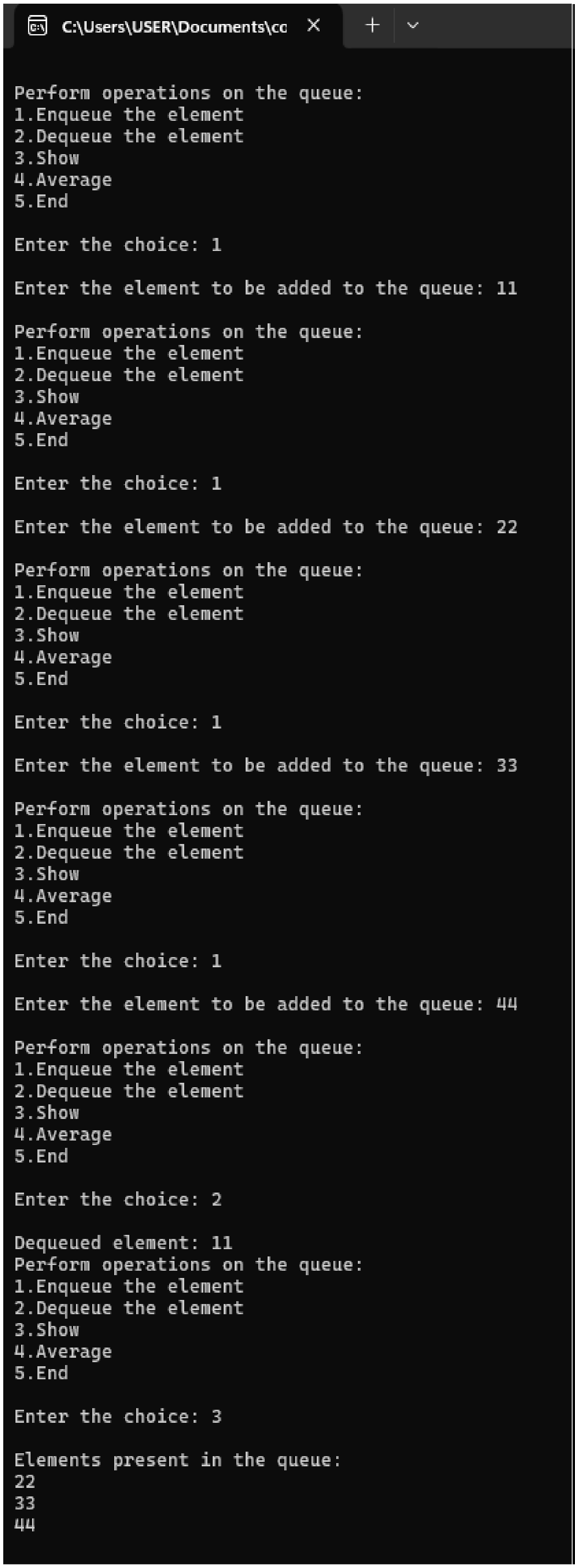
```
Enter the choice: 3

Elements present in the queue:
264
236

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

Enter the choice: 4

Average of the current values in the queue: 250.0000

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

Enter the choice: 5

Process returned 0 (0x0)   execution time : 28.825 s
Press any key to continue.
```

2. Queue implementation code is given in your Google classroom. Also, the pseudo code of Circular Queue is also given. Based on these two, You are asked to write the code that implements a circular queue.

Solution:

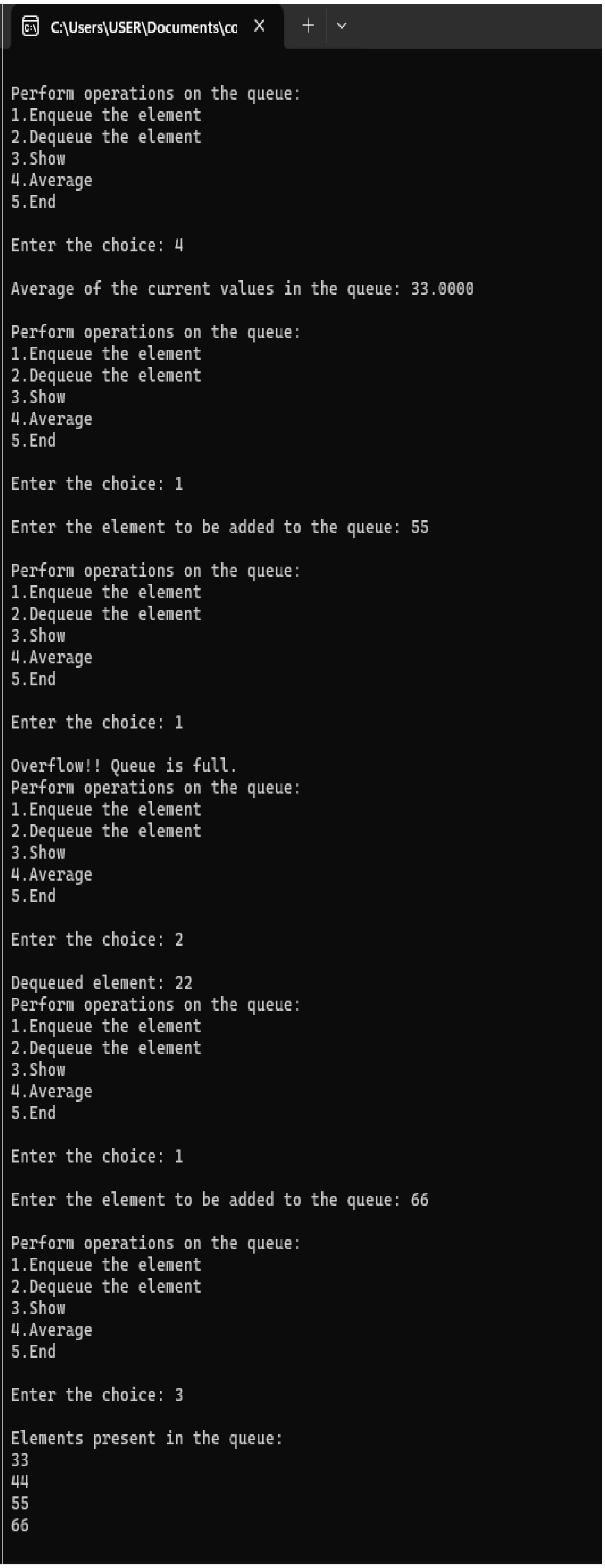
```
#include <stdio.h>

#include <stdlib.h>
#define SIZE 4
int front = -1, rear = -1, inp_array[SIZE];
void enqueue();
void dequeue();
void show();
void average();
int main() {
    int choice;
    while (1) {
        printf("\nPerform operations on the queue:");
        printf("\n1.Enqueue the element\n2.Dequeue the element\n3.Show\n4.Average\n5.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:
           average();
            break;
        case 5:
            exit(0);
        default:
            printf("\nInvalid choice!!"); }
}
 }
void enqueue() {
    int x;
    if ((rear + 1) % SIZE == front) {
        printf("\nOverflow!! Queue is full.");
    } else {
        printf("\nEnter the element to be added to the queue: ");
        scanf("%d", &x);
        if (front == -1)
            front = 0;
        rear = (rear + 1) % SIZE;
        inp_array[rear] = x; } }
```

```
void dequeue()
 {
    if (front == -1)
 {
        printf("\nUnderflow!! Queue is empty.");
    } else {
        printf("\nDequeued element: %d", inp_array[front]);
        if (front == rear) {

            front = rear = -1;
        } else {
            front = (front + 1) % SIZE;  }  } }
void show() {
    if (front == -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;  } } }
void average() {
    if (front == -1) {  printf("\nQueue is empty!!");
    } else {
        int sum = 0;
        int count = 0;
        int i = front;
        while (1) {
            sum += inp_array[i];
            count++;
            if (i == rear) break;
            i = (i + 1) % SIZE;
        }
        double avg = (double)sum / count;
        printf("\nAverage of the current values in the queue: %.4f\n", avg);
    }
}
```



C:\Users\USER\Documents\co

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

Enter the choice: 1

Enter the element to be added to the queue: 11

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

Enter the choice: 1

Enter the element to be added to the queue: 22

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

Enter the choice: 1

Enter the element to be added to the queue: 33

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

Enter the choice: 1

Enter the element to be added to the queue: 44

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

Enter the choice: 2

Dequeued element: 11
Perform operations on the queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.Average
5.End

Enter the choice: 3

Elements present in the queue:
22
33
44
```



C:\Users\USER\Documents\co

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

Enter the choice: 4

Average of the current values in the queue: 33.0000

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

Enter the choice: 1

Enter the element to be added to the queue: 55

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

Enter the choice: 1

Overflow!! Queue is full.
Perform operations on the queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.Average
5.End

Enter the choice: 2

Dequeued element: 22
Perform operations on the queue:
1.Enqueue the element
2.Dequeue the element
3.Show
4.Average
5.End

Enter the choice: 1

Enter the element to be added to the queue: 66

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

Enter the choice: 3

Elements present in the queue:
33
44
55
66
```