

# Data Structures using C

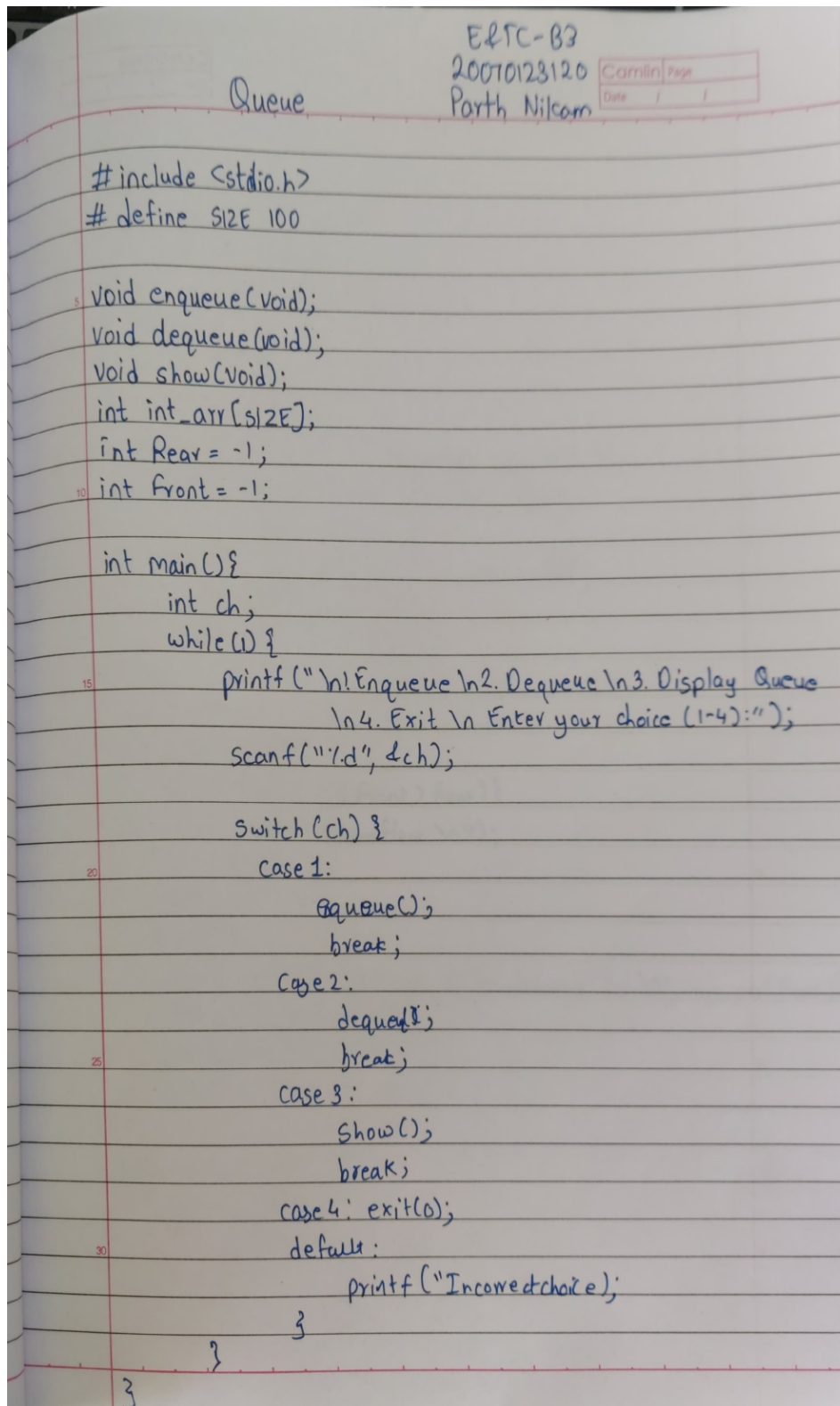
25<sup>th</sup> March 2022

Parth Nikam  
20070123120  
E&TC - B3

Aim: - Studying and coding on queue.

Objective: - To perform enqueue, dequeue, and display operations on queue in C language

Code: -



The image shows a handwritten C program for a queue implemented as an array. The code is written on lined paper with a red margin. At the top right, there is a header with the text 'E&TC-B3', '20070123120', and 'Parth Nikam'. To the left of this header, the word 'Queue' is written. Below the header, there is a small table with the columns 'Carrollin', 'Page', and 'Date'. The code itself starts with '#include <stdio.h>' and '#define SIZE 100'. It then defines three functions: 'void enqueue(void);', 'void dequeue(void);', and 'void show(void);'. It also declares an array 'int int\_arr[SIZE];' and two integers 'int Rear = -1;' and 'int Front = -1;'. The main function 'int main()' contains a 'while(1)' loop. Inside the loop, it prints a menu with four options: '1. Enqueue', '2. Dequeue', '3. Display Queue', and '4. Exit'. It then scans for an integer choice. A switch statement follows, with cases for each option. Case 1 calls 'enqueue()' and breaks. Case 2 calls 'dequeue()' and breaks. Case 3 calls 'show()' and breaks. Case 4 calls 'exit(0)'. The default case prints 'Incorrect choice'. The program ends with a closing brace for the switch statement, a brace for the while loop, and a brace for the main function.

```
#include <stdio.h>
#define SIZE 100

void enqueue(void);
void dequeue(void);
void show(void);
int int_arr[SIZE];
int Rear = -1;
int Front = -1;

int main() {
    int ch;
    while(1) {
        printf("\n1. Enqueue\n2. Dequeue\n3. Display Queue\n4. Exit\nEnter your choice (1-4):");
        scanf("%d", &ch);

        switch(ch) {
            case 1:
                enqueue();
                break;
            case 2:
                dequeue();
                break;
            case 3:
                show();
                break;
            case 4:
                exit(0);
            default:
                printf("Incorrect choice");
        }
    }
}
```

# Data Structures using C

25<sup>th</sup> March 2022

Parth Nikam  
20070123120  
E&TC - B3

E&TC-B3  
20070123120  
Parth Nikam

Camlin	Page
Date	/ /

Queue

```
void enqueue() {
    int insert_item;
    if (Rear == SIZE-1)
        printf("Overflow\n");
    else {
        if (front == -1) {
            front = 0;
            printf("Element to be inserted: ");
            scanf("%d", &insert_item);
            Rear = Rear + 1;
            inp_arr[Rear] = insert_item;
        }
    }
}

void dequeue() {
    if (front == -1 || front > Rear) {
        printf("Underflow\n");
        return;
    }
    else {
        printf("Element to be deleted %d\n", inp_arr[front]);
        front = front + 1;
    }
}

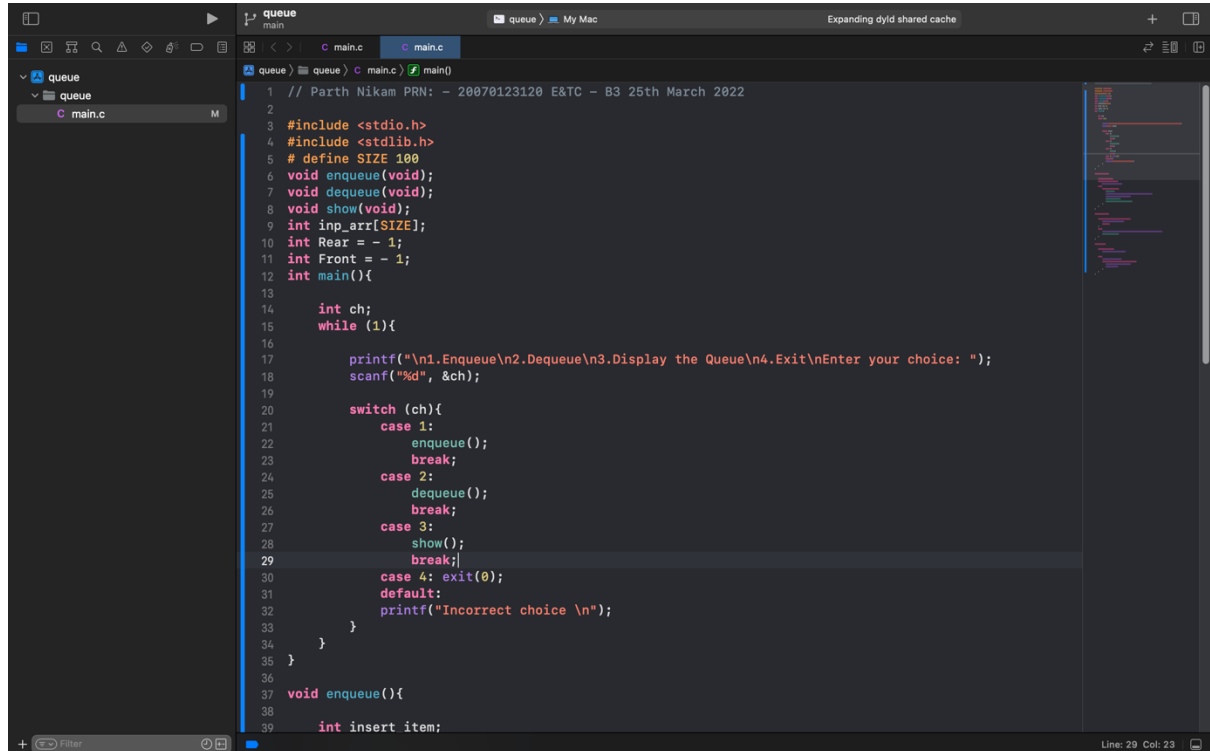
void show() {
    if (front == -1)
        printf("Empty Queue");
    else {
        printf("Queue\n");
        for (int i = front; i <= Rear; i++) {
            printf("%d ", inp_arr[i]);
        }
    }
}
```

# Data Structures using C

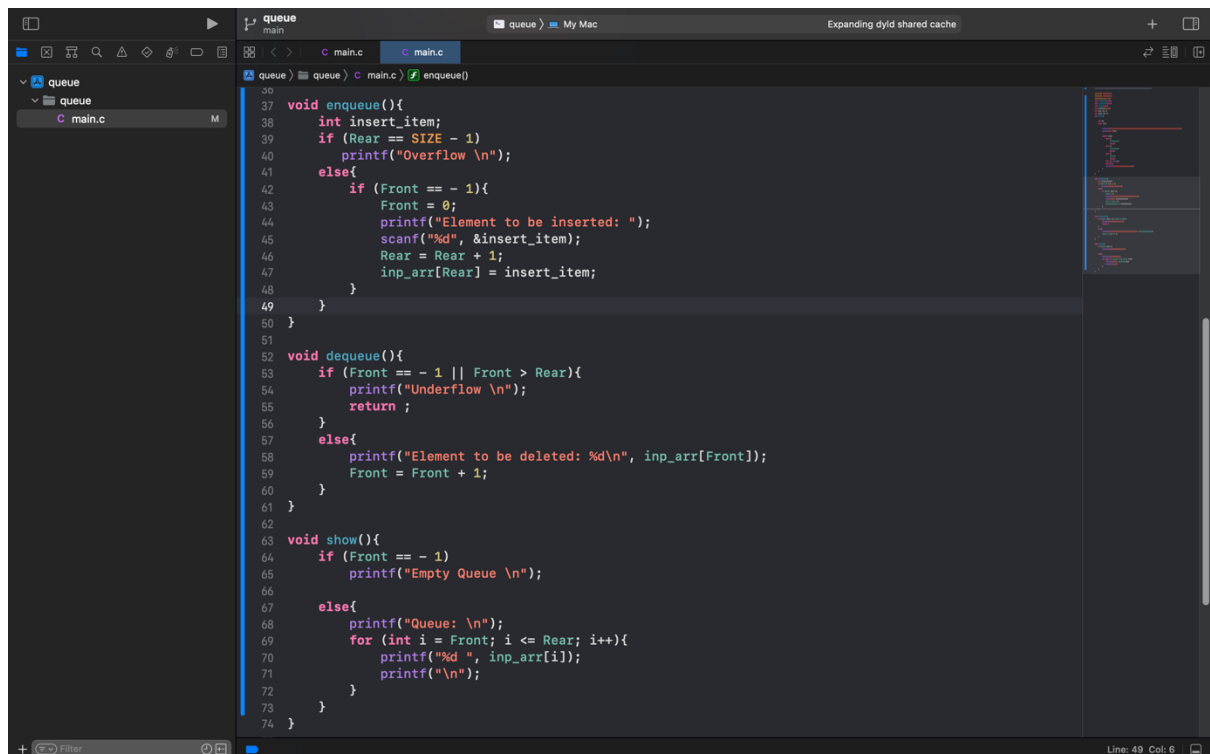
25<sup>th</sup> March 2022

Parth Nikam  
20070123120  
E&TC – B3

Screenshot: -



```
1 // Parth Nikam PRN: - 20070123120 E&TC - B3 25th March 2022
2
3 #include <stdio.h>
4 #include <stdlib.h>
5 # define SIZE 100
6 void enqueue(void);
7 void dequeue(void);
8 void show(void);
9 int inp_arr[SIZE];
10 int Rear = - 1;
11 int Front = - 1;
12 int main(){
13
14     int ch;
15     while (1){
16
17         printf("\n1.Enqueue\n2.Dequeue\n3.Display the Queue\n4.Exit\nEnter your choice: ");
18         scanf("%d", &ch);
19
20         switch (ch){
21             case 1:
22                 enqueue();
23                 break;
24             case 2:
25                 dequeue();
26                 break;
27             case 3:
28                 show();
29                 break;
30             case 4: exit(0);
31             default:
32                 printf("Incorrect choice \n");
33         }
34     }
35 }
36
37 void enqueue(){
38
39     int insert item;
```



```
30
31
32 void enqueue(){
33     int insert_item;
34     if (Rear == SIZE - 1)
35         printf("Overflow \n");
36     else{
37         if (Front == - 1){
38             Front = 0;
39             printf("Element to be inserted: ");
40             scanf("%d", &insert_item);
41             Rear = Rear + 1;
42             inp_arr[Rear] = insert_item;
43         }
44     }
45 }
46
47 void dequeue(){
48     if (Front == - 1 || Front > Rear){
49         printf("Underflow \n");
50         return ;
51     }
52     else{
53         printf("Element to be deleted: %d\n", inp_arr[Front]);
54         Front = Front + 1;
55     }
56 }
57
58 void show(){
59     if (Front == - 1)
60         printf("Empty Queue \n");
61     else{
62         printf("Queue: \n");
63         for (int i = Front; i <= Rear; i++){
64             printf("%d ", inp_arr[i]);
65             printf("\n");
66         }
67     }
68 }
69 }
```

# Data Structures using C

25<sup>th</sup> March 2022

Parth Nikam  
20070123120  
E&TC – B3

```
30
31
32
33
34
35
36
37 void enqueue(){
38     int insert_item;
39     if (Rear == SIZE - 1)
40         printf("Overflow \n");
41     else{
42         if (Front == - 1){
43             Front = 0;
44             printf("Element to be inserted: ");
45             scanf("%d", &insert_item);
46             Rear = Rear + 1;
47             inp_arr[Rear] = insert_item;
48         }
49     }
50 }
51
```

```
1.Enqueue
2.Dequeue
3.Display the Queue
4.Exit
Enter your choice: 1
Element to be inserted: 5

1.Enqueue
2.Dequeue
3.Display the Queue
4.Exit
Enter your choice: 3
Queue:
5

1.Enqueue
2.Dequeue
3.Display the Queue
4.Exit
Enter your choice: 4
Program ended with exit code: 0
```

Result: - A queue in C is basically a linear data structure to store and manipulate the data elements. It follows the order of First In First Out (FIFO). In queues, the first element entered into the array is the first element to be removed from the array.