

# Assignment 3



## Shri G.S Institute of Technology & Science

### Data Structure

### Assignment 3 – INDEX

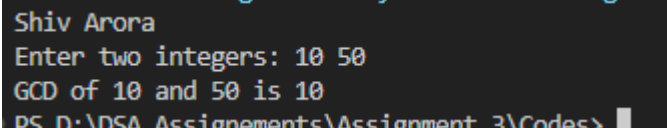
Sr. No.	Program	P. No.	Remarks
1	Write a program in C to find the GCD of two numbers using recursion.	1	
2	Write a program in C to get the largest element of an array using recursion.	1	
3	Write the Algorithm for enqueue and dequeue operation.	1-2	
4	Write a C program to implement a queue using an array. Programs should contain functions for inserting elements into the queue, deletion elements into the queue, displaying queue elements, and checking whether the queue is empty or not.	2-4	
5	Write a C program to count the number and sum of elements in a queue.	4-6	

P1 Write a program in C to find the GCD of two numbers using recursion.

```
#include <stdio.h>
int findGCD(int a, int b) {
    if (b == 0)
        return a;
    return findGCD(b, a % b);
}

void main() {
    printf("Shiv Arora\n");
    int n1, n2;
    printf("Enter two integers: ");
    scanf("%d %d", &n1, &n2);
    int gcd = findGCD(n1, n2);
    printf("GCD of %d and %d is %d\n", n1, n2, gcd);
}
```

OUTPUT:



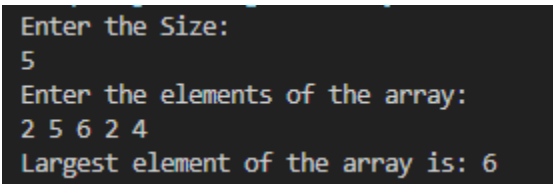
```
Shiv Arora
Enter two integers: 10 50
GCD of 10 and 50 is 10
PS D:\DSA Assignments\Assignment 3\Codes>
```

P2 Write a program in C to get the largest element of an array using recursion.

```
#include <stdio.h>
int findLargest(int arr[], int size) {
    if (size == 1)
        return arr[0];
    int maxRest = findLargest(arr, size-1);
    return (arr[size-1] > maxRest) ? arr[size-1] : maxRest;
}

void main() {
    int size;
    printf("Enter the Size:\n");
    scanf("%d", &size);
    printf("Enter the elements of the array:\n");
    int arr[size];
    for (int i=0; i<size; i++) {
        scanf("%d", &arr[i]);
    }
    int largest = findLargest(arr, size);
    printf("Largest element of the array is: %d\n", largest);
}
```

OUTPUT:



```
Enter the Size:
5
Enter the elements of the array:
2 5 6 2 4
Largest element of the array is: 6
PS D:\DSA Assignments\Assignment 3\Codes>
```

P3 Write the Algorithm for enqueue and dequeue operation.

ENQUEUE

Step1 -> START

Step2 -> Check if the queue is full (overflow and exit).

Step3 -> Increment rear pointer to point the next empty space.  
 Step4 -> Add data element to the queue location, where the rear is pointing.  
 Step5 -> END

#### DEQUEUE

Step1 -> START  
 Step2 -> Check if the queue is empty (underflow and exit).  
 Step3 -> Access the data where front is pointing.  
 Step4 -> Increment front pointer to point to the next available data element.  
 Step5 -> END

P4 Write a C program to implement a queue using an array. Programs should contain functions for inserting elements into the queue, deletion elements into the queue, displaying queue elements, and checking whether the queue is empty or not.

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 100

typedef struct Queue {
    int items[MAX];
    int front, rear;
} Queue;

void initializeQueue(Queue *q) {
    q->front = -1;
    q->rear = -1;
}

int isEmpty(Queue *q) {
    return (q->front == -1);
}

void enqueue(Queue *q, int value) {
    if (q->rear == MAX - 1) {
        printf("Queue Overflow!");
        return;
    }
    if (isEmpty(q)) {
        q->front = 0;
    }
    q->items[++q->rear] = value;
}

int dequeue(Queue *q) {
    if (isEmpty(q)) {
        printf("Queue Underflow!\n");
        return -1;
    }
    if (q->front == q->rear) {
        q->front = q->rear = -1;
    } else {
        q->front++;
    }
    return -1;
}

void display(Queue *q) {
```

```

    if (isEmpty(q)) {
        printf("Queue is empty.\n");
        return;
    }
    printf("Queue elements: ");
    for (int i = q->front; i <= q->rear; i++) {
        printf("|%d ", q->items[i]);
    }
    printf("\n");
}
void main() {
    Queue q;
    initializeQueue(&q);
    int choice, value;
    int flag = 1;
    printf("\nQueue Operations:\n");
    printf("1. Enqueue\n");
    printf("2. Dequeue\n");
    printf("3. Display\n");
    printf("4. Check if Empty\n");
    printf("5. Exit\n");
    while (flag) {
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Value to Enqueue: ");
                scanf("%d", &value);
                enqueue(&q, value);
                break;
            case 2:
                dequeue(&q);
                break;
            case 3:
                display(&q);
                break;
            case 4:
                if (isEmpty(&q)) {
                    printf("Queue is empty.\n");
                } else {
                    printf("Queue is not empty.\n");
                }
                break;
            case 5:
                printf("Exiting...\n");
                flag = 0;
        }
    }
}

```

OUTPUT:

```
Queue Operations:
1. Enqueue
2. Dequeue
3. Display
4. Check if Empty
5. Exit
Enter your choice: 1
Value to Enqueue: 14
Enter your choice: 1
Value to Enqueue: 65
Enter your choice: 1
Value to Enqueue: 46
Enter your choice: 2
Enter your choice: 1
Value to Enqueue: 87
Enter your choice: 3
Queue elements: |65 |46 |87 |
Enter your choice: 4
Queue is not empty.
Enter your choice: 5
Exiting...
```

P6 Write a C program to count the number and sum of elements in a queue.

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 100

typedef struct Queue {
    int items[MAX];
    int front, rear;
} Queue;

void initializeQueue(Queue *q) {
    q->front = -1;
    q->rear = -1;
}

int isEmpty(Queue *q) {
    return (q->front == -1);
}

void enqueue(Queue *q, int value) {
    if (q->rear == MAX - 1) {
        printf("Queue Overflow!");
        return;
    }
    if (isEmpty(q)) {
        q->front = 0;
    }
    q->items[++q->rear] = value;
}

int dequeue(Queue *q) {
    if (isEmpty(q)) {
        printf("Queue Underflow!\n");
        return -1;
    }
}
```

```

    }
    if (q->front == q->rear) {
        q->front = q->rear = -1;
    } else {
        q->front++;
    }
    return -1;
}

int countElements(Queue *q) {
    if (isEmpty(q)) {
        return 0;
    }
    return (q->rear - q->front + 2);
}

int sumElements(Queue *q) {
    if (isEmpty(q)) {
        return 0;
    }
    int sum = 0;
    for (int i = q->front; i <= q->rear; i++) {
        sum += q->items[i];
    }
    return sum;
}

void main() {
    Queue q;
    initializeQueue(&q);
    int choice, value;
    int flag = 1;
    printf("\nQueue Operations:\n");
    printf("1. Enqueue\n");
    printf("2. Dequeue\n");
    printf("3. Number of Elements\n");
    printf("4. Sum of Elements\n");
    printf("5. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);
    while (flag) {
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Value to Enqueue: ");
                scanf("%d", &value);
                enqueue(&q, value);
                break;
            case 2:
                dequeue(&q);
                break;
            case 3:
                printf("Number of elements in the queue: %d\n", countElements(&q));

```

```
        break;
    case 4:
        printf("Sum of elements in the queue: %d\n", sumElements(&q));
        break;
    case 5:
        printf("Exiting...\n");
        flag = 0;
    }
}
```

OUTPUT:

```
Queue Operations:
1. Enqueue
2. Dequeue
3. Number of Elements
4. Sum of Elements
5. Exit
Enter your choice: 1
Enter your choice: 13
Enter your choice: 1
Value to Enqueue: 45
Enter your choice: 1
Value to Enqueue: 56
Enter your choice: 4
Sum of elements in the queue: 101
Enter your choice: 3
Number of elements in the queue: 3
Enter your choice: 2
Enter your choice: 3
Number of elements in the queue: 2
Enter your choice: 4
Sum of elements in the queue: 56
Enter your choice: 5
Exiting...
```