# **ASSIGNMENT 4 DATA STRUCTURES**

Q1. Develop a menu driven program demonstrating the following operations on simple Queues: enqueue(), dequeue(), isEmpty(), isFull(), display(), and peek().

```
G Q1.cpp > ...
      //1) Develop a menu driven program demonstrating the following operations on simple Queues:
      // enqueue(), dequeue(), isEmpty(), isFull(), display(), and peek().
      #include <iostream>
 3
      using namespace std;
 4
 5
 6
      #define SIZE 5
 7
      class Queue {
 8
 9
          int arr[SIZE];
          int front, rear;
10
      public:
11
          Queue() {
12
              front = -1;
13
              rear = -1;
14
15
16
          bool isEmpty() {
17
              return (front == -1 || front > rear);
18
19
20
          bool isFull() {
21
              return (rear == SIZE - 1);
22
23
24
          void enqueue(int x) {
25
               if (isFull()) {
26
                   cout << "Queue is Full\n";</pre>
27
28
                   return;
29
              if (front == -1) front = 0;
30
               arr[++rear] = x;
31
               cout << x << " inserted\n";</pre>
32
33
34
          void dequeue() {
35
36
               if (isEmpty()) {
                   cout << "Queue is Empty\n";</pre>
37
```

```
G Q1.cpp > ...
      class Queue {
 8
35
          void dequeue() {
               if (isEmpty()) {
36
                   return;
38
39
               cout << arr[front++] << " removed\n";</pre>
40
41
42
           void peek() {
43
44
               if (isEmpty()) cout << "Queue is Empty\n";</pre>
               else cout << "Front element: " << arr[front] << endl;</pre>
45
46
47
           void display() {
48
49
               if (isEmpty()) {
                   cout << "Queue is Empty\n";</pre>
50
51
                   return;
52
               cout << "Queue: ";
53
               for (int i = front; i <= rear; i++)</pre>
54
55
               cout << arr[i] << " ";
               cout << endl;</pre>
56
57
58
      };
59
      int main() {
60
61
          Queue q;
           int ch, val;
62
63
           do {
               cout << "\n1.Enqueue 2.Dequeue 3.Peek 4.Display 5.Exit\n";</pre>
64
               cin >> ch;
65
66
               switch (ch) {
               case 1: cout << "Enter value: "; cin >> val; q.enqueue(val); break;
67
               case 2: q.dequeue(); break;
68
               case 3: q.peek(); break;
69
70
               case 4: q.display(); break;
71
          } while (ch != 5);
72
73
          return 0;
74
75
```

```
1.Enqueue 2.Dequeue 3.Peek 4.Display 5.Exit
1
Enter value: 23
23 inserted
Enter value: 23
23 inserted
23 inserted
1.Enqueue 2.Dequeue 3.Peek 4.Display 5.Exit
Enter value: 55
55 inserted
1.Enqueue 2.Dequeue 3.Peek 4.Display 5.Exit
Queue: 23 55
Enter value: 55
55 inserted
1. Enqueue 2. Dequeue 3. Peek 4. Display 5. Exit
Queue: 23 55
1. Enqueue 2. Dequeue 3. Peek 4. Display 5. Exit
Queue: 23 55
1.Enqueue 2.Dequeue 3.Peek 4.Display 5.Exit
Front element: 23
1. Enqueue 2. Dequeue 3. Peek 4. Display 5. Exit
23 removed
4 Engueur 2 Degueur 2 Deals 4 Display E Freit
```

Q2. Develop a menu driven program demonstrating the following operations on Circular Queues: enqueue(), dequeue(), isEmpty(), isFull(), display(), and peek().

```
G Q2.cpp > ...
      #include <iostream>
 1
      using namespace std;
 2
 3
     #define SIZE 5
 4
 5
      class CircularQueue {
 6
          int arr[SIZE];
 7
          int front, rear;
 8
 9
      public:
          CircularQueue() {
10
11
             front = -1;
              rear = -1;
12
13
14
          bool isEmpty() {
15
16
              return (front == -1);
17
18
          bool isFull() {
19
              return ((rear + 1) % SIZE == front);
20
21
22
23
          void enqueue(int x) {
              if (isFull()) {
24
                  cout << "Queue is Full\n";
25
                  return;
26
27
              if (isEmpty())
28
                 front = rear = 0;
29
              else
30
                  rear = (rear + 1) % SIZE;
31
32
              arr[rear] = x;
              cout << x << " inserted\n";</pre>
33
34
35
          void dequeue() {
36
           if (isEmpty()) {
37
```

```
€ Q2.cpp > ...
      class CircularQueue {
 6
           void dequeue() {
36
               if (isEmpty()) {
37
                   cout << "Queue is Empty\n";</pre>
38
39
                   return;
40
               cout << arr[front] << " removed\n";</pre>
41
               if (front == rear)
42
                   front = rear = -1;
43
               else
44
45
                   front = (front + 1) % SIZE;
46
47
           void peek() {
48
               if (isEmpty()) cout << "Queue is Empty\n";</pre>
49
               else cout << "Front element: " << arr[front] << endl;</pre>
50
51
52
53
           void display() {
               if (isEmpty()) {
54
                   cout << "Queue is Empty\n";</pre>
55
56
                   return;
57
               cout << "Oueue: ";
58
               int i = front;
59
               while (true) {
60
                   cout << arr[i] << " ";
61
                   if (i == rear) break;
62
                   i = (i + 1) \% SIZE;
63
64
               cout << endl;
65
66
      };
67
68
      int main() {
69
           CircularQueue q;
70
71
          int ch. val:
```

```
LI CULUI YUCUC 4,
          int ch, val;
71
72
          do {
              cout << "\n1.Enqueue 2.Dequeue 3.Peek 4.Display 5.Exit\n";</pre>
73
              cin >> ch;
74
75
              switch (ch) {
              case 1: cout << "Enter value: "; cin >> val; q.enqueue(val); break;
76
              case 2: q.dequeue(); break;
77
              case 3: q.peek(); break;
78
              case 4: q.display(); break;
79
80
          } while (ch != 5);
81
          return 0;
82
83
84
```

```
1. Enqueue 2. Dequeue 3. Peek 4. Display 5. Exit
Enter value: 23
23 inserted
1. Enqueue 2. Dequeue 3. Peek 4. Display 5. Exit
Enter value: 44
44 inserted
1. Enqueue 2. Dequeue 3. Peek 4. Display 5. Exit
23 removed
1. Enqueue 2. Dequeue 3. Peek 4. Display 5. Exit
Front element: 44
1. Enqueue 2. Dequeue 3. Peek 4. Display 5. Exit
4
Queue: 44
1. Enqueue 2. Dequeue 3. Peek 4. Display 5. Exit
Front element: 44
1. Enqueue 2. Dequeue 3. Peek 4. Display 5. Exit
44 removed
1. Enqueue 2. Dequeue 3. Peek 4. Display 5. Exit
Queue is Empty
1. Enqueue 2. Dequeue 3. Peek 4. Display 5. Exit
Queue is Empty
```

#### Q3. Write a program interleave the first half of the queue with second half.

Sample I/P: 4 7 11 20 5 9

Sample O/P: 4 20 7 5 11 9

```
G Q3.cpp > ...
 1 #include <iostream>
 2 #include <queue>
     using namespace std;
 3
 4
      void interleaveQueue(queue<int>& q) {
 5
          int n = q.size();
 6
          queue<int> firstHalf;
 7
 8
          for (int i = 0; i < n / 2; i++) {
 9
               firstHalf.push(q.front());
10
               q.pop();
11
12
13
          while (!firstHalf.empty()) {
14
              q.push(firstHalf.front());
15
16
               firstHalf.pop();
               q.push(q.front());
17
               q.pop();
18
19
20
21
22
      int main() {
          queue<int> q;
23
24
          int n, val;
          cout << "Enter number of elements: ";
25
          cin >> n:
26
          cout << "Enter elements: ";</pre>
27
          for (int i = 0; i < n; i++) {
28
              cin >> val;
29
              q.push(val);
30
31
32
          interleaveQueue(q);
33
34
          cout << "Interleaved Queue: ";</pre>
35
          while (!q.empty()) {
36
              cout << q.front() << " ";
37
```

```
PS C:\Users\hxrle\OneDrive\Dokumen\ASSIGNMENT 4 DS> g++ Q3.cpp -o Q3
PS C:\Users\hxrle\OneDrive\Dokumen\ASSIGNMENT 4 DS> g++ Q3.cpp -o Q3
PS C:\Users\hxrle\OneDrive\Dokumen\ASSIGNMENT 4 DS> ./Q3
Enter number of elements: 6
Enter elements: 4 7 11 20 5 9
Interleaved Queue: 4 20 7 5 11 9
```

#### Q4. Write a program to find first non-repeating character in a string using Queue.

#### Sample I/P: a a b c Sample O/P: a -1 b b

#### **Code with output:**

```
G Q4.cpp > ...
 1 #include <iostream>
  2 #include <queue>
 3 #include <string>
 4
     using namespace std;
  5
      int main() {
  6
          string s;
 7
           cout << "Enter characters (use space separated like a a b c): ";</pre>
 8
           getline(cin, s); // 🗹 takes full line input
 9
10
          queue<char> q;
 11
          int freq[26] = \{0\};
12
13
14
           cout << "Output: ";
15
           for (int i = 0; i < s.length(); i++) {
16
              if (s[i] == ' ') continue; // skip spaces
17
18
19
              char c = s[i];
              freq[c - 'a']++;
 20
              q.push(c);
 21
22
              while (!q.empty() && freq[q.front() - 'a'] > 1)
 23
 24
                   q.pop();
 25
               if (q.empty()) cout << "-1 ";</pre>
 26
               else cout << q.front() << " ";</pre>
 27
 28
 29
 30
           cout << endl;
           return 0;
 31
 32
PROBLEMS
          OUTPUT
                   DEBUG CONSOLE
                                  TERMINAL
PS C:\Users\hxrle\OneDrive\Dokumen\ASSIGNMENT 4 DS> ./Q4
Enter characters (use space separated like a a b c): a a b c e e
Output: a -1 b b b b
```

## Q5. Write a program to implement a stack using (a) Two queues and (b) One Queue.

#### Code:

a)

```
G Q5A.cpp > ...
      #include <iostream>
  1
      #include <queue>
  2
      using namespace std;
  3
 4
  5
      class Stack {
 6
          queue<int> q1, q2;
      public:
 7
          void push(int x) {
 8
               q2.push(x);
 9
               while (!q1.empty()) {
10
                   q2.push(q1.front());
11
                   q1.pop();
12
13
14
               swap(q1, q2);
15
16
17
          void pop() {
               if (q1.empty()) cout << "Stack Empty\n";</pre>
18
               else q1.pop();
19
20
21
22
          int top() {
               if (q1.empty()) return -1;
23
               return q1.front();
24
25
26
          bool empty() {
27
               return q1.empty();
28
29
30
      };
31
32
      int main() {
33
          Stack s;
          s.push(10);
34
          s.push(20);
35
          s.push(30);
36
37
          cout << s.top() << endl; // 30
```

```
38     s.pop();
39     cout << s.top() << endl; // 20
40     return 0;
41 }</pre>
```

```
PS C:\Users\hxrle\OneDrive\Dokumen\ASSIGNMENT 4 DS> ./Q5A 30 20
```

## b) Using One queue

```
G Q5B.cpp > ...
 1 #include <iostream>
     #include <queue>
      using namespace std;
 4
  5
      class Stack {
  6
          queue<int> q;
 7
      public:
          void push(int x) {
 8
 9
               int n = q.size();
               q.push(x);
10
               for (int i = 0; i < n; i++) {
11
                   q.push(q.front());
12
13
                   q.pop();
14
15
16
17
          void pop() {
              if (q.empty()) cout << "Stack Empty\n";</pre>
18
19
               else q.pop();
20
21
22
          int top() {
              if (q.empty()) return -1;
23
24
              return q.front();
25
26
      };
27
      int main() {
28
29
          Stack s;
30
          s.push(1);
31
          s.push(2);
32
          s.push(3);
          cout << s.top() << endl; // 3
33
34
          s.pop();
          cout << s.top() << endl; // 2</pre>
35
36
          return 0;
37
```

PS C:\Users\hxrle\OneDrive\Dokumen\ASSIGNMENT 4 DS> ./Q5B
3
3
2