**Name: ZISHNENDU SARKER**

**Roll: 2K19/CO/450**

**Java Programming**

**Lab assignment 08**

**28/11/2021**

**Java Program to Join Two ListsJava Program to Implement the queue data structure**

Theory: The queue is a linear data structure that follows the FIFO rule (first in first out). We can implement Queue for not only Integers but also Strings, Float, or Characters. There are 5 primary operations in Queue:

1. enqueue() adds element x to the front of the queue
2. dequeue() removes the last element of the queue
3. front() returns the front element
4. rear() returns the rear element
5. empty() returns whether the queue is empty or not

Code:

public class Queue {

int SIZE = 5;

int items[] = new int[SIZE];

int front, rear;

Queue() {

front = -1;

rear = -1;

}

// check if the queue is full

boolean isFull() {

if (front == 0 && rear == SIZE - 1) {

return true;

}

return false;

}

// check if the queue is empty

boolean isEmpty() {

if (front == -1)

return true;

else

return false;

}

// insert elements to the queue

void enQueue(int element) {

// if queue is full

if (isFull()) {

System.out.println("Queue is full");

}

else {

if (front == -1) {

// mark front denote first element of queue

front = 0;

}

rear++;

// insert element at the rear

items[rear] = element;

System.out.println("Insert " + element);

}

}

// delete element from the queue

int deQueue() {

int element;

// if queue is empty

if (isEmpty()) {

System.out.println("Queue is empty");

return (-1);

}

else {

// remove element from the front of queue

element = items[front];

// if the queue has only one element

if (front >= rear) {

front = -1;

rear = -1;

}

else {

// mark next element as the front

front++;

}

System.out.println( element + " Deleted");

return (element);

}

}

// display element of the queue

void display() {

int i;

if (isEmpty()) {

System.out.println("Empty Queue");

}

else {

// display the front of the queue

System.out.println("\nFront index-> " + front);

// display element of the queue

System.out.println("Items -> ");

for (i = front; i <= rear; i++)

System.out.print(items[i] + " ");

// display the rear of the queue

System.out.println("\nRear index-> " + rear);

}

}

public static void main(String[] args) {

// create an object of Queue class

Queue q = new Queue();

// try to delete element from the queue

// currently queue is empty

// so deletion is not possible

q.deQueue();

// insert elements to the queue

for(int i = 1; i < 6; i ++) {

q.enQueue(i);

}

// 6th element can't be added to queue because queue is full

q.enQueue(6);

q.display();

// deQueue removes element entered first i.e. 1

q.deQueue();

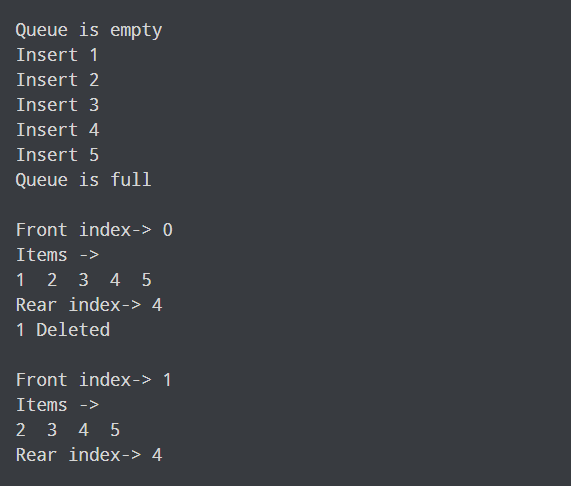
// Now we have just 4 elements

q.display();

}

}

Output:



Learning Outcome: We have implemented the queue data structure in Java.

**Java Program to Remove duplicate elements from ArrayList**

Theory: To remove duplicate elements from the arraylist, we have Here, we have used the LinkedHashSet to create a set. It is because it removes the duplicate elements and maintains insertion order.

Code:

import java.util.ArrayList;

import java.util.Arrays;

import java.util.LinkedHashSet;

import java.util.Set;

class Main {

public static void main(String[] args) {

// create an arraylist from the array

// using asList() method of the Arrays class

ArrayList<Integer> numbers = new ArrayList<>(Arrays.asList(1, 2, 3, 4, 1, 3));

System.out.println("ArrayList with duplicate elements: " + numbers);

// convert the arraylist into a set

Set<Integer> set = new LinkedHashSet<>();

set.addAll(numbers);

// delete al elements of arraylist

numbers.clear();

// add element from set to arraylist

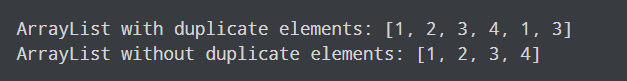
numbers.addAll(set);

System.out.println("ArrayList without duplicate elements: " + numbers);

}

}

Output:



Learning Outcome:

We have created an arraylist named numbers. The arraylist contains duplicate elements. To remove duplicate elements from the arraylist, we have add all elements from arraylist to set

empty the arraylist using clear() method to add all elements from set to arraylist. Here, we have used the LinkedHashSet to create a set. It is because it removes the duplicate elements and maintains insertion order.