package Assignment14;

import java.util.\*;

public class QueueArrays {

static class Node{

int data;

Node next;

Node(int data){

this.data = data;

this.next = null;

}

}

static Node createNode(int [] arr){

Node head = new Node(arr[0]);

Node move = head ;

for(int i = 1 ; i < arr.length ; i++){

Node temp = new Node(arr[i]);

move.next = temp ;

move = move.next;

}

return head;

}

static Node enqueue(int data , Node head){

Node newnode = new Node(data);

Node temp = head;

while (temp.next != null){

temp = temp.next;

}

temp.next = newnode;

return head;

}

static Node dequeue(Node head){

if(head == null) return head;

return head.next;

}

static void display(Node head){

Node temp = head;

while(temp != null){

System.***out***.print(" "+temp.data);

temp = temp.next;

}

}

public static void main(String[] args) {

int [] arr = {10, 20, 30};

Node head = *createNode*(arr);

Node head1 = *enqueue*(40,head);

//Node head2 = dequeue(head);

*display*(head1);

}

}

package Assignment14;

import java.util.\*;

public class QueueLinkedList {

public class QueueUsingArrays {

static int[] *queue* = new int[100]; // fixed-size queue

static int *front* = 0;

static int *rear* = 0;

// Enqueue operation

public static void enqueue(int data) {

if (*rear* == 100) {

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

return;

}

*queue*[*rear*] = data;

*rear*++;

System.***out***.println(data + " enqueued");

}

// Dequeue operation

public static void dequeue() {

if (*front* == *rear*) {

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

return;

}

System.***out***.println(*queue*[*front*] + " dequeued");

*front*++;

}

// Display operation

public static void display() {

if (*front* == *rear*) {

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

return;

}

System.***out***.print("Queue elements: ");

for (int i = *front*; i < *rear*; i++) {

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

}

System.***out***.println();

}

// Main method for testing

public static void main(String[] args) {

Scanner sc = new Scanner(System.***in***);

boolean running = true;

while (running) {

System.***out***.println("\nChoose operation:\n1. Enqueue\n2. Dequeue\n3. Display\n4. Exit");

int choice = sc.nextInt();

switch (choice) {

case 1:

System.***out***.print("Enter data to enqueue: ");

int data = sc.nextInt();

*enqueue*(data);

break;

case 2:

*dequeue*();

break;

case 3:

*display*();

break;

case 4:

running = false;

break;

default:

System.***out***.println("Invalid option");

}

}

sc.close();

}

}

}

package Assignment14;

import java.util.\*;

public class ReverseKqueue {

public static void main(String[] args) {

int [] arr = {1,2,3,4,5};

int k = 2;

System.***out***.println(*firstRotate*(arr, k));

}

static String firstRotate (int [] arr , int k){

Deque<Integer> q1 = new ArrayDeque<>();

Deque<Integer> q2 = new ArrayDeque<>();

for(int i = 1 ; i <= arr.length ; i++){

q1.offer(i);

}

while(k > 0){

q2.offer(q1.poll());

k--;

}

while(!q2.isEmpty()) {

q1.addFirst(q2.poll());

}

return q1.toString();

}

}