*LAB # 12*

queues implementation

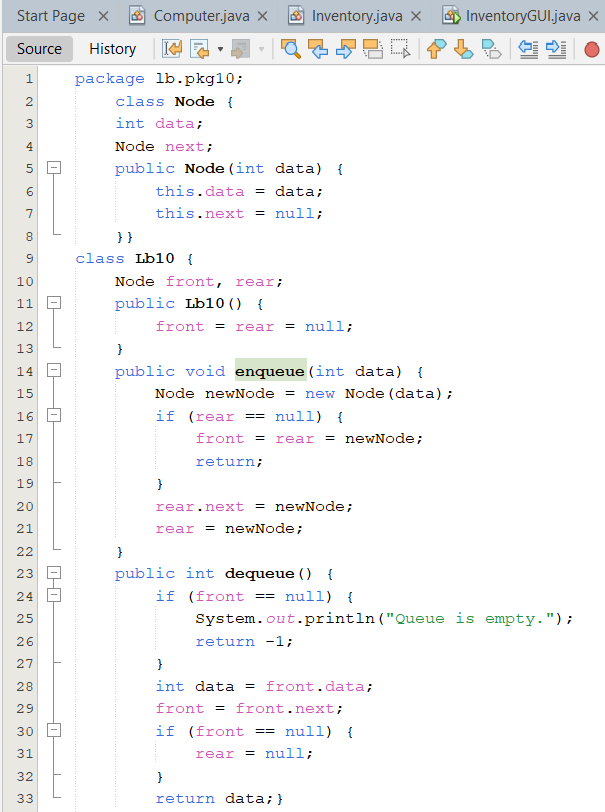
# *OBJECTIVE:*

*Fifo Queue implementation using stack and linkedlist*

*LAB task*

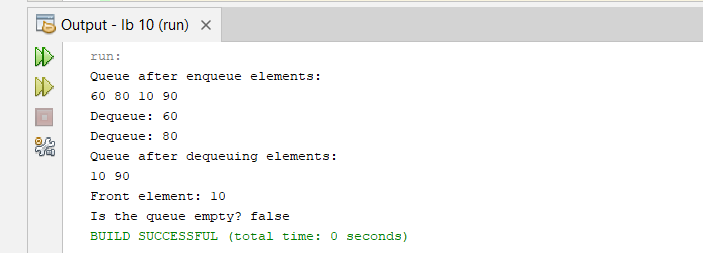
1. *Write a program to implement queue using link list and perform operations on it.*

**Code:**

****

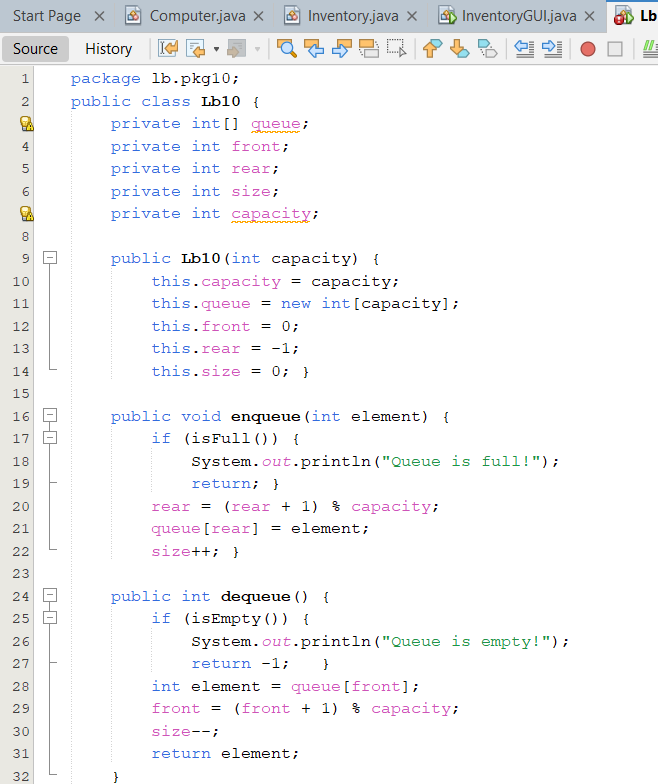
****

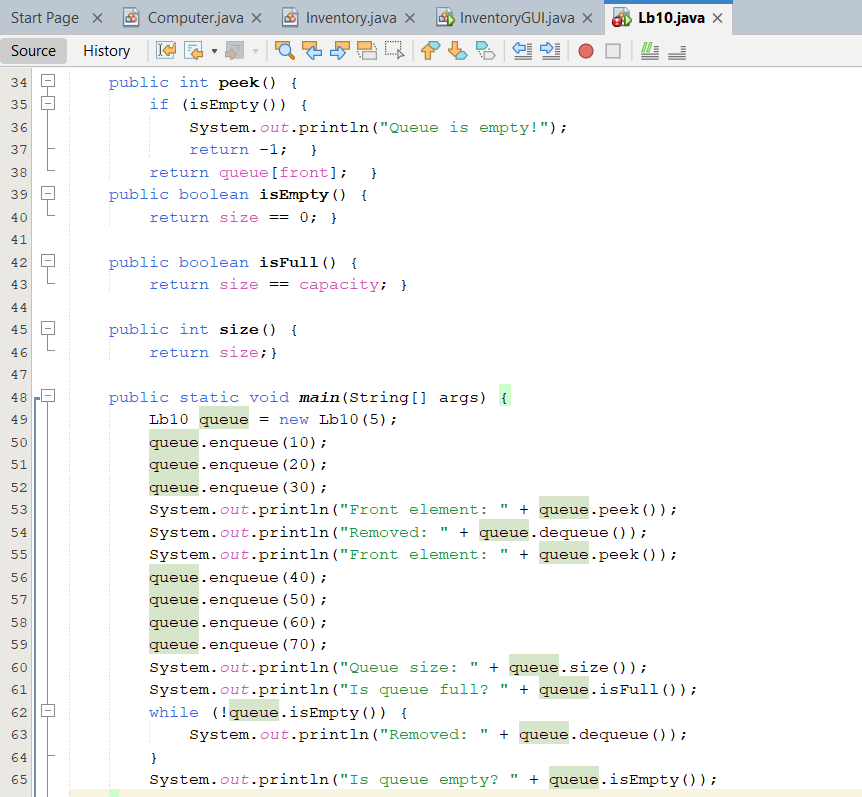
**Output:**

**

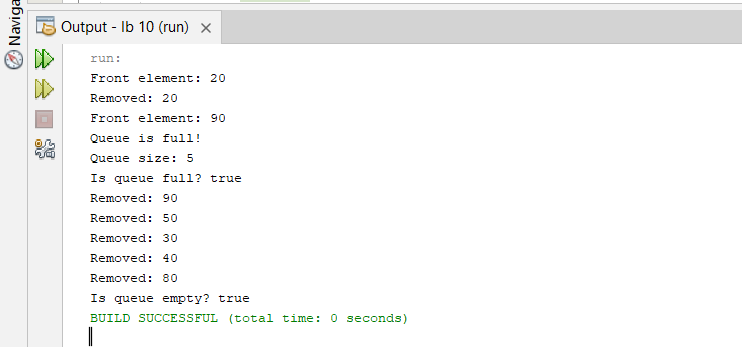
1. *In this problem, you need to implement a* ***Queue ADT*** *using an* ***array****. Your queue will support the following operations:*
2. ***Enqueue****: Add an element to the rear of the queue.*
3. ***Dequeue****: Remove an element from the front of the queue.*
4. ***Peek****: Retrieve the front element of the queue without removing it.*
5. ***IsEmpty****: Check if the queue is empty.*
6. ***IsFull****: Check if the queue is full.*
7. ***Size****: Return the number of elements currently in the queue.*

**Code:**

****

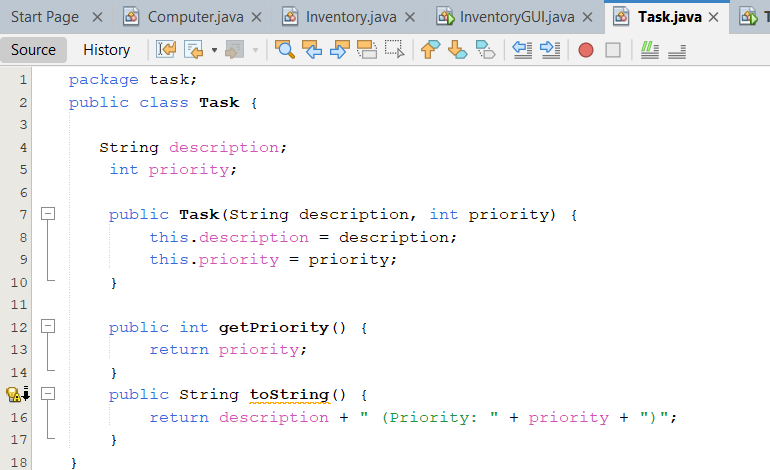
****

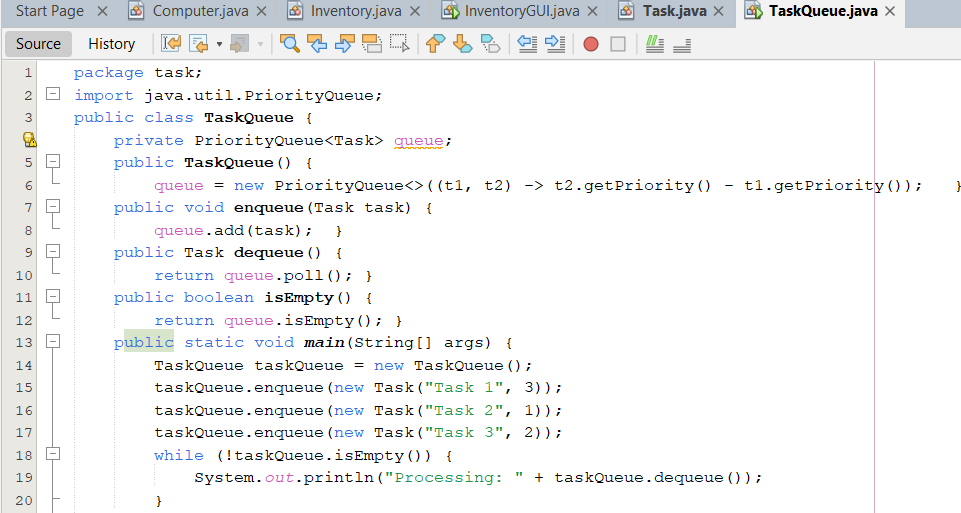
**Output:**

****

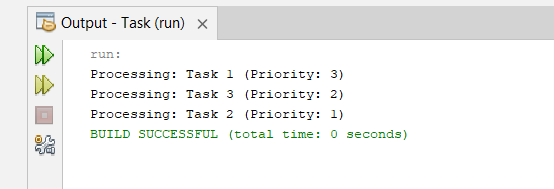
1. *You are managing a queue of tasks to be processed by a machine. Each task is assigned a priority, and you must process them in order of priority, with higher priorities processed first.Implement a queue where tasks are added with a priority, and each time you process (dequeue) a task, you must remove the task with the highest priority*

**Code:**

****

****

**Output:**

**

*HOME task*

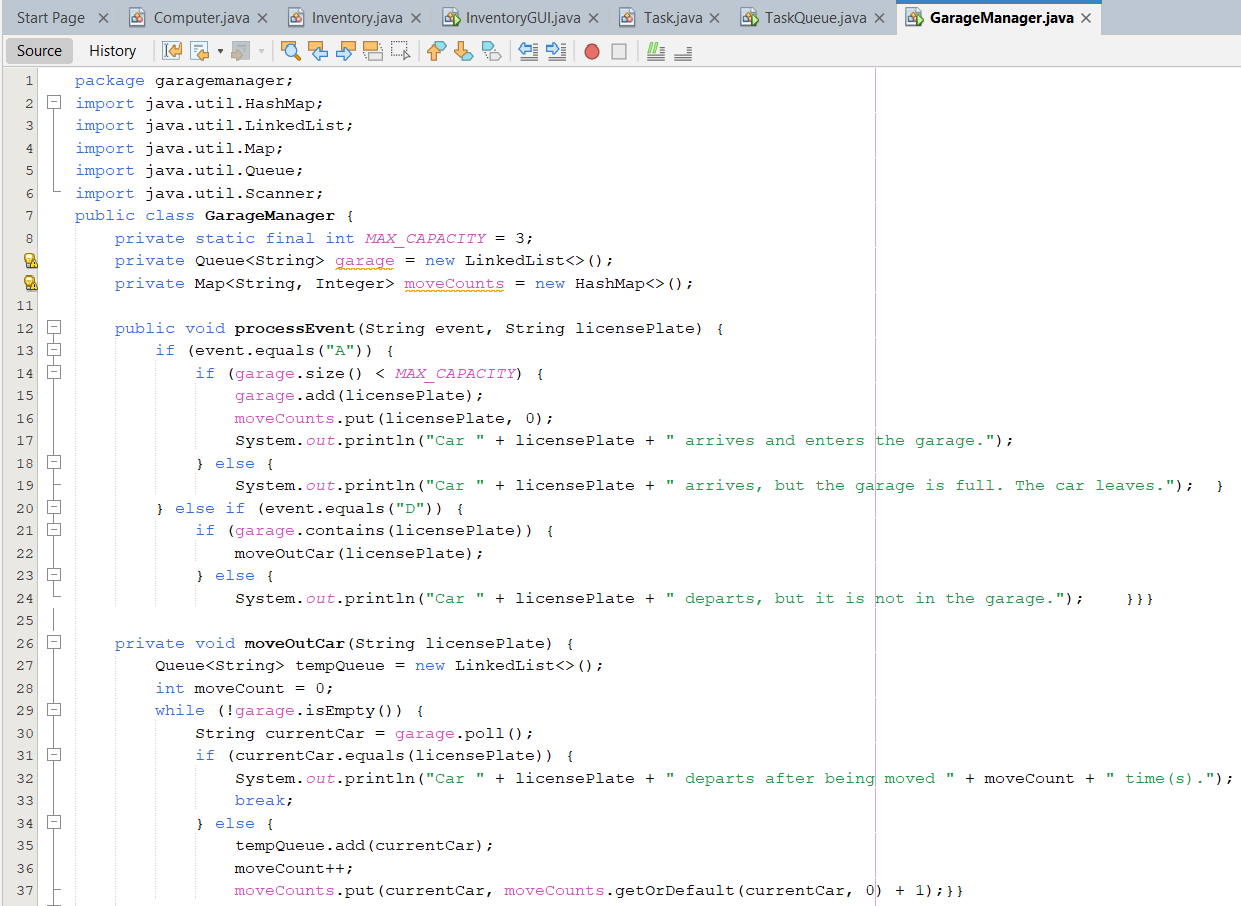
1. *Write a program that processes a group of input lines. Each input line contains an 'A' for arrival and a 'D' for departure, and a license plate number. Cars are assumed to arrive and depart in the order specified by the input. The program should:*

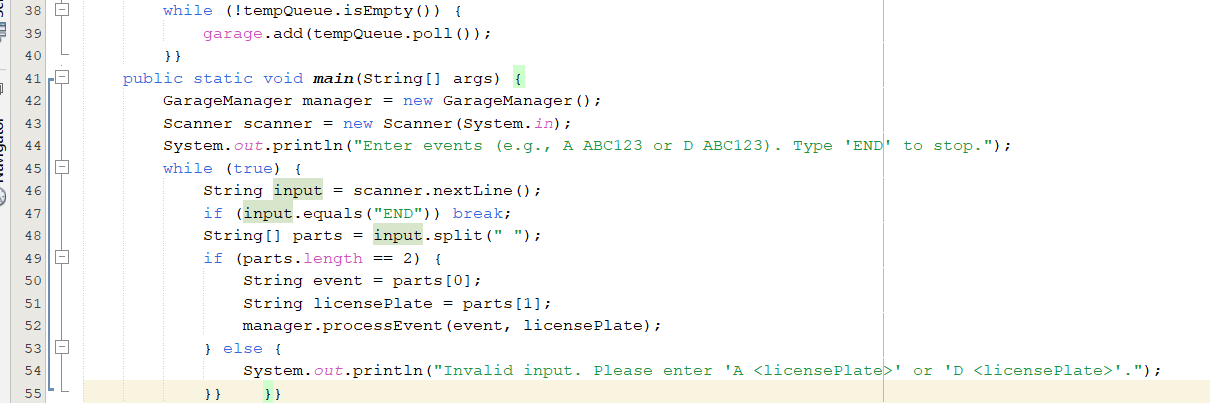
*a) Print a message whenever a car arrives or departs.*

*b) When a car arrives, the message should specify whether or not there is a room for the car in the garage. If there is no room, the car leaves without entering the garage.*

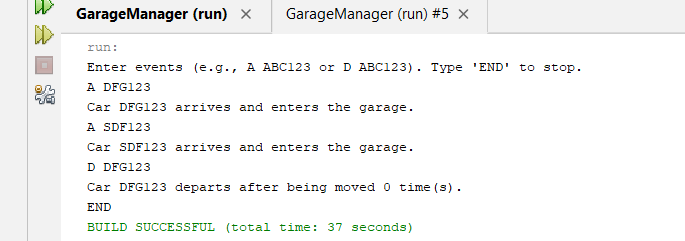
*c) When a car departs, the message should include the number of times that the car was moved out of the garage to allow other cars to depart.*

**Code:**

****

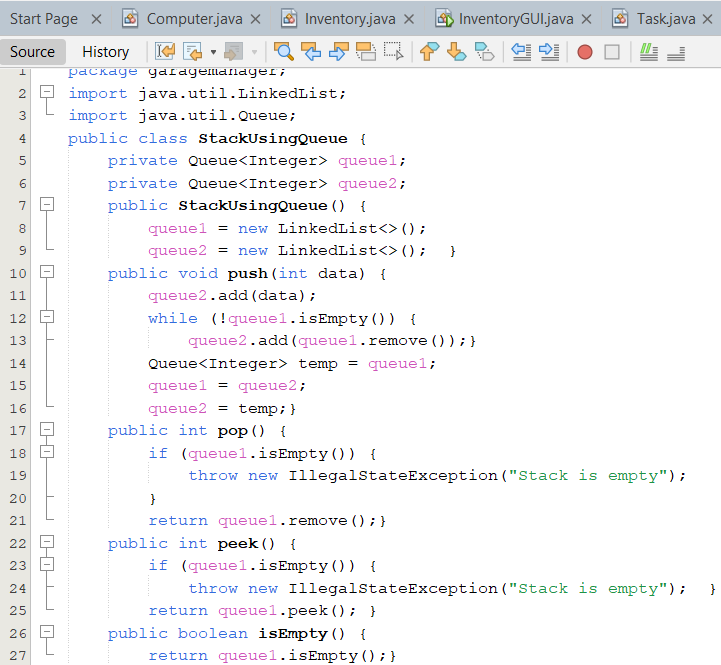
****

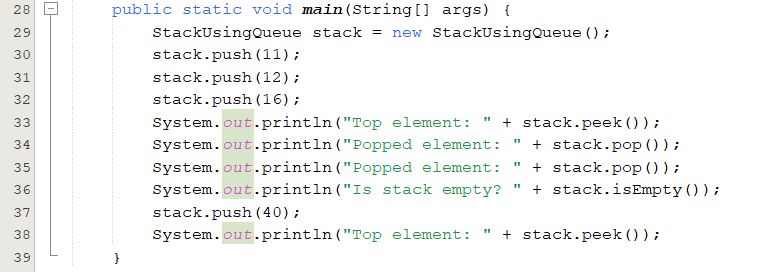
**Output:**

**

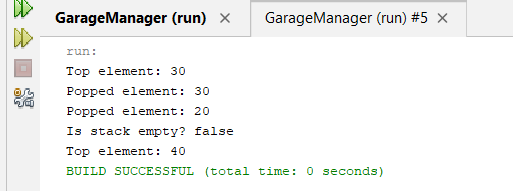
1. *Write a program to implement stack using queue.*

**Code:**

****

****

**Output:**

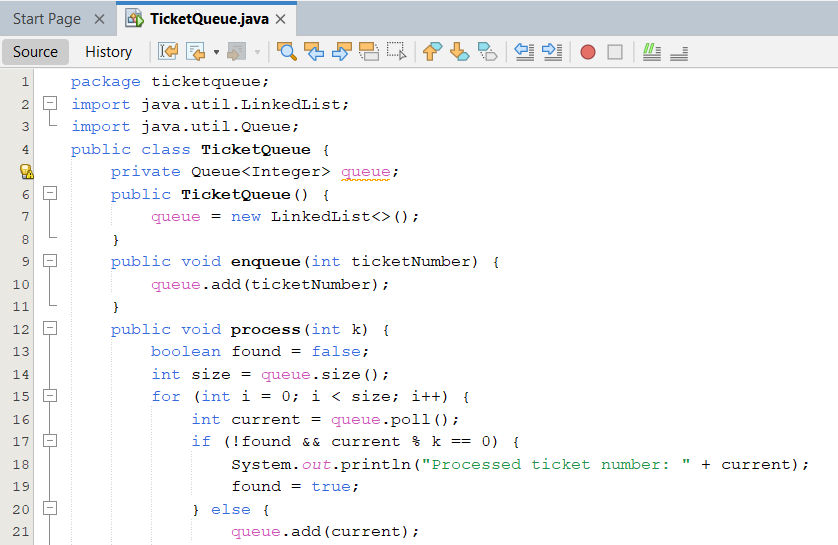
**

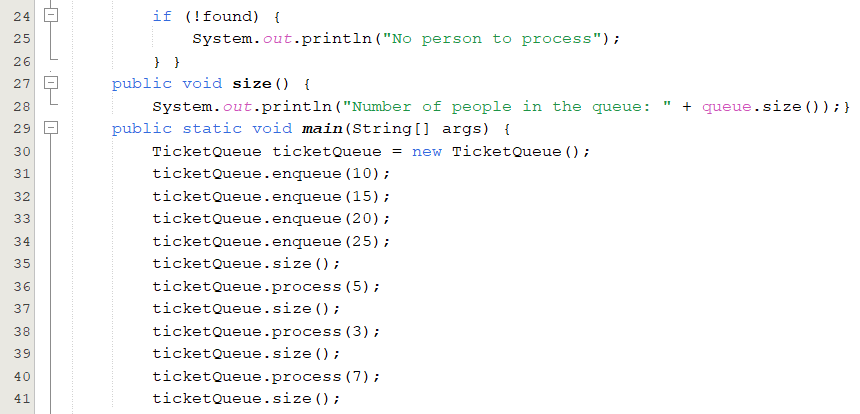
1. *You are simulating a waiting line for a ticket counter. Each person in the queue has a ticket number, and the queue will be processed based on the order in which people arrive. If a person’s ticket number is a multiple of a given number k, they will be processed (dequeued) before others, otherwise they will remain in the queue.*

*Implement the following operations:*

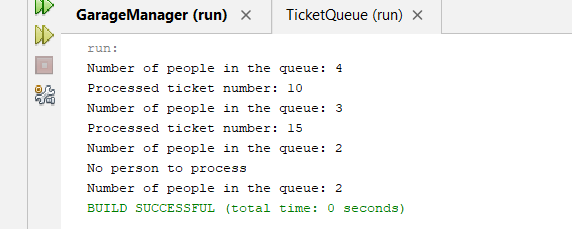
1. ***ENQUEUE ticket\_number****: Add a person with the given ticket number to the queue.*
2. ***PROCESS k****: Process the person with the ticket number that is a multiple of k. If no such person is found, print "No person to process".*
3. ***SIZE****: Print the number of people currently in the queue*

**Code:**

****

****

**Output:**

****