**CSC 3020 – Java Programming**

**Homework 6 – [your name]**

**25 points – Due April 18, 10am**

**Late deadline is April 20, 11:59pm, but 20% off**

**a)** Save this document with your name and the homework number somewhere in the file name.

**b)** Type/paste your answers into the document.

**c)** Submit this document and your .java file(s) to the Blackboard item where you downloaded this document. Do not submit a zip file but individually attach your files.

You have been hired by *Millennial Mart* to write a Java console application that processes customers through their store checkout line. The application has the following two classes:

**Customer.java**

Each object created from this class represents one customer and includes the following fields and methods:

**Fields**

● (static) totalCustomers – count of all distinct customers; initialize to 0 in declaration.

● (static) totalItems – total items purchased by all customers; initialize to 0 in declaration.

● (static) totalCost – total cost of all items purchased by all customers; initialize to 0 in declaration.

● ID – unique number assigned to each customer.

● items – number of items in customer shopping cart.

● cost – cost of all items in customer shopping cart.

● next – pointer to next customer.

**Methods**

● A constructor with no parameters that sets the fields, respectively, to these values:

totalCustomers = totalCustomers + 1

ID = -1

items = -1

cost = -1

next = null

● A constructor with two parameters that sets the fields, respectively, to these values:

totalCustomers = totalCustomers + 1

totalItems = totalItems + items

totalCost = totalCost + cost

ID – set to totalCustomers

items – set from parameter

cost – set from parameter

next = null

● Getter methods for each field (declare the getters for totalCustomers, totalItems, and totalCost static).

● Setter methods for each field (declare the setters for totalCustomers, totalItems, and totalCost static).

● *equals* method that compares ID for equality.

● *toString* method for returning ID, items, and cost values only.

**HW6.java**

This class contains the main method and uses the Customer class to represent each customer in the store checkout line. The line is represented as a queue (first-in first-out). Begin with an empty queue. Create the following three methods:

● **void insertCustomer(Customer ptr)** – this method adds a customer to the queue.

● **Customer removeCustomer()** – this method removes a customer from the queue.

● **void printCustomers()** – this method prints any customers in the queue. It prints the ID, items, and cost for each customer formatted in three columns.

In the **main method**, loop twenty times. Within each loop, either add a customer to the queue or remove a customer from the queue, and then print the customers in the queue. Add a customer if the queue is empty or if a randomly generated value is true (use random function **nextBoolean** for this). Otherwise, remove a customer. Print a message showing the ID of the customer added or removed. When adding a customer (creating an object of the Customer class):

● Generate a random number between 1 and 40 for their number of items.

● Generate a random number between 1 and 200 for their cost.

Keep track of the number of customers in the queue, and the number of customers that have left the queue. At the bottom of each loop, pause execution for one second with **Thread.sleep(1000);** After the loop, print:

● The number of customers that have left the queue.

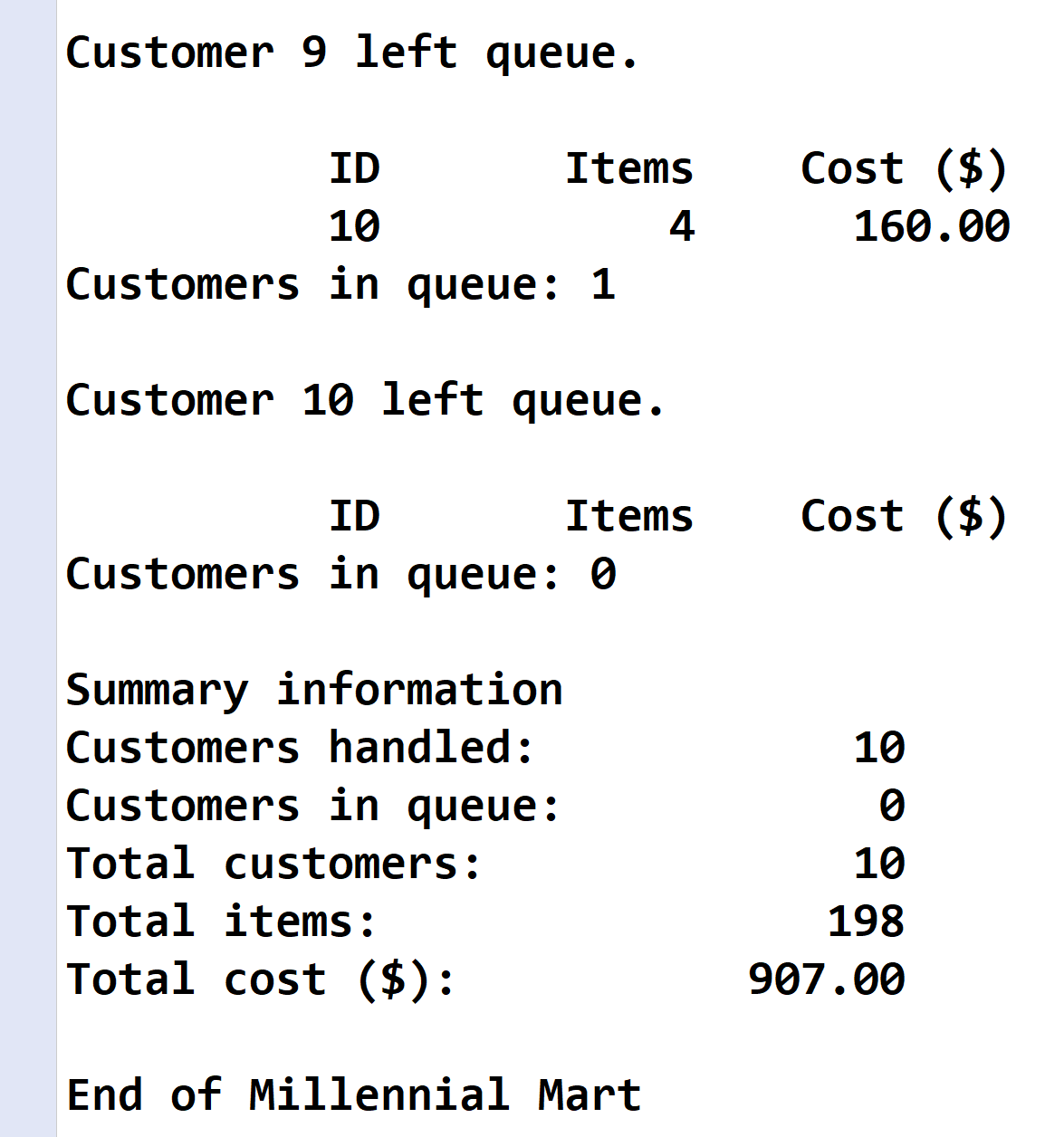
● The number still in the queue.

● The total number of customers handled.

● The total items purchased by all customers.

● The total cost of all items purchased by all customers.

Each run will generate different results. Here is sample output from the end of one run:



**Customer.java**

**//======================================================================**

**//**

**// Class: Customer**

**// Description:**

**// This class represents a Customer.**

**//**

**//======================================================================**

**package wsu.HW06\_01;**

**//======================================================================**

**// class Customer**

**//======================================================================**

**public class Customer**

**{**

**//==================================================================**

**// Fields**

**//==================================================================**

**private static int totalCustomers = 0;**

**private static int totalItems = 0;**

**private static double totalCost = 0;**

**private int ID;**

**private int items;**

**private double cost;**

**private Customer next;**

**//==================================================================**

**// Constructors**

**//==================================================================**

**//------------------------------------------------------------------**

**// No-parameter constructor**

**//------------------------------------------------------------------**

**public Customer()**

**{**

**totalCustomers = totalCustomers + 1;**

**ID = -1;**

**items = -1;**

**cost = -1;**

**next = null;**

**}**

**//------------------------------------------------------------------**

**// Two-parameter constructor**

**//------------------------------------------------------------------**

**public Customer(int items, double cost)**

**{**

**totalCustomers = totalCustomers + 1;**

**totalItems = totalItems + items;**

**totalCost = totalCost + cost;**

**ID = totalCustomers;**

**this.items = items;**

**this.cost = cost;**

**next = null;**

**}**

**//==================================================================**

**// Methods**

**//==================================================================**

**//------------------------------------------------------------------**

**// get methods**

**//------------------------------------------------------------------**

**public static int getTotalCustomers()**

**{**

**return totalCustomers;**

**}**

**public static int getTotalItems()**

**{**

**return totalItems;**

**}**

**public static double getTotalCost()**

**{**

**return totalCost;**

**}**

**public int getID()**

**{**

**return ID;**

**}**

**public int getItems()**

**{**

**return items;**

**}**

**public double getCost()**

**{**

**return cost;**

**}**

**public Customer getNext()**

**{**

**return next;**

**}**

**//------------------------------------------------------------------**

**// set methods**

**//------------------------------------------------------------------**

**public static void setTotalCustomers(int totalCustomers)**

**{**

**Customer.totalCustomers = totalCustomers;**

**}**

**public static void setTotalItems(int totalItems)**

**{**

**Customer.totalItems = totalItems;**

**}**

**public static void setTotalCost(double totalCost)**

**{**

**Customer.totalCost = totalCost;**

**}**

**public void setID(int ID)**

**{**

**this.ID = ID;**

**}**

**public void setItems(int items)**

**{**

**this.items = items;**

**}**

**public void setCost(double cost)**

**{**

**this.cost = cost;**

**}**

**public void setNext(Customer next)**

**{**

**this.next = next;**

**}**

**//------------------------------------------------------------------**

**// equals and toString methods**

**//------------------------------------------------------------------**

**public boolean equals(Customer b)**

**{**

**if (this.ID == b.ID)**

**return true;**

**else**

**return false;**

**}**

**public String toString ()**

**{**

**return "Code: " + ID +**

**"\nItems: " + items +**

**"\nCost ($): " + String.format("%,1.2f", cost);**

**}**

**}**

**HW6.java**

**//======================================================================**

**//**

**// Title: Millennial Mart**

**// Course: CSC 3020**

**// Homework: 6-1**

**// Author: Dan Ouellette**

**// Date: 18 April 2018**

**// Description:**

**// This Java console application processes customers through the store**

**// checkout line. For each customer, it maintains an ID, the number of**

**// items in the shopping cart, and cost of all items in the shopping**

**// cart. It randomly adds and removes customers from the queue. When**

**// done, the number of customers that have left the queue, the number**

**// still in the queue, the total number of customers handled, the total**

**// items purchased by all customers, and the total cost of all items**

**// purchased by all customers.**

**//**

**//======================================================================**

**package wsu.HW06\_01;**

**//Import classes**

**import java.util.Random;**

**//======================================================================**

**// class HW06\_01**

**//======================================================================**

**public class HW06\_01**

**{**

**//------------------------------------------------------------------**

**// Constants**

**//------------------------------------------------------------------**

**// Declare formatting constants**

**private static final String COLFMTS1 = "%12s";**

**private static final String COLFMTS2 = "%-20s";**

**private static final String COLFMTD = "%12d";**

**private static final String COLFMTF = "%12.2f";**

**private static final int ITEM\_MAX = 40;**

**private static final int COST\_MAX = 200;**

**//------------------------------------------------------------------**

**// Variables**

**//------------------------------------------------------------------**

**private static Customer head = null; // Pointer to queue head**

**private static Customer tail = null; // Pointer to queue tail**

**//------------------------------------------------------------------**

**// insertCustomer**

**//------------------------------------------------------------------**

**public static void insertCustomer(Customer ptr)**

**{**

**// Test if queue is empty**

**if (head == null)**

**head = ptr;**

**else**

**tail.setNext(ptr);**

**// Set tail**

**tail = ptr;**

**}**

**//------------------------------------------------------------------**

**// removeCustomer**

**//------------------------------------------------------------------**

**public static Customer removeCustomer()**

**{**

**// Declare variables**

**Customer ptr = null;**

**// Test if queue is empty**

**if (head != null)**

**{**

**// Remove node from queue**

**ptr = head;**

**if (head == tail)**

**tail = null;**

**head = head.getNext();**

**}**

**return ptr;**

**}**

**//------------------------------------------------------------------**

**// printCustomers**

**//------------------------------------------------------------------**

**public static void printCustomers()**

**{**

**// Declare variables**

**Customer ptr = head;**

**int count = 0;**

**// Loop to print customers in line**

**System.out.printf("%n" + COLFMTS1 + COLFMTS1 + COLFMTS1 + "%n",**

**"ID", "Items", "Cost ($)");**

**while (ptr != null)**

**{**

**System.out.printf(COLFMTD + COLFMTD + COLFMTF + "%n",**

**ptr.getID(), ptr.getItems(), ptr.getCost());**

**count = count + 1;**

**ptr = ptr.getNext();**

**}**

**System.out.println("Customers in queue: " + count);**

**}**

**//------------------------------------------------------------------**

**// main**

**//------------------------------------------------------------------**

**public static void main (String[] args)**

**{**

**// Declare variables**

**int customersInQueue;**

**int customersHandled;**

**Customer ptr;**

**Random rand = new Random();**

**// Show application header**

**System.out.println("Welcome to Millennial Mart");**

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

**// Initialize queue**

**head = null;**

**tail = null;**

**customersInQueue = 0;**

**customersHandled = 0;**

**// Loop to process customer checkout**

**for (int i = 1; i <= 20; i++)**

**{**

**try**

**{**

**// Test if customers in queue**

**if (customersInQueue == 0 || rand.nextBoolean())**

**{**

**// Create customer and add to queue**

**ptr = new Customer(**

**rand.nextInt(ITEM\_MAX) + 1,**

**rand.nextInt(COST\_MAX) + 1);**

**insertCustomer(ptr);**

**customersInQueue = customersInQueue + 1;**

**System.out.println("\nCustomer " + ptr.getID() +**

**" entered queue.");**

**}**

**else**

**{**

**// Remove customer from queue**

**ptr = removeCustomer();**

**customersInQueue = customersInQueue - 1;**

**customersHandled = customersHandled + 1;**

**System.out.println("\nCustomer " + ptr.getID() +**

**" left queue.");**

**}**

**// Print queue**

**printCustomers();**

**// Pause thread**

**Thread.sleep(1000);**

**}**

**catch (InterruptedException e)**

**{**

**System.out.println("InterruptedException error while " +**

**"running customer checkout.");**

**System.out.println("Error message: " + e.getMessage());**

**}**

**}**

**// Show summary and application close**

**System.out.println("\nSummary information");**

**System.out.printf(COLFMTS2 + COLFMTD + "%n",**

**"Customers handled:", customersHandled);**

**System.out.printf(COLFMTS2 + COLFMTD + "%n",**

**"Customers in queue:", customersInQueue);**

**System.out.printf(COLFMTS2 + COLFMTD + "%n",**

**"Total customers:", Customer.getTotalCustomers());**

**System.out.printf(COLFMTS2 + COLFMTD + "%n",**

**"Total items:", Customer.getTotalItems());**

**System.out.printf(COLFMTS2 + COLFMTF + "%n",**

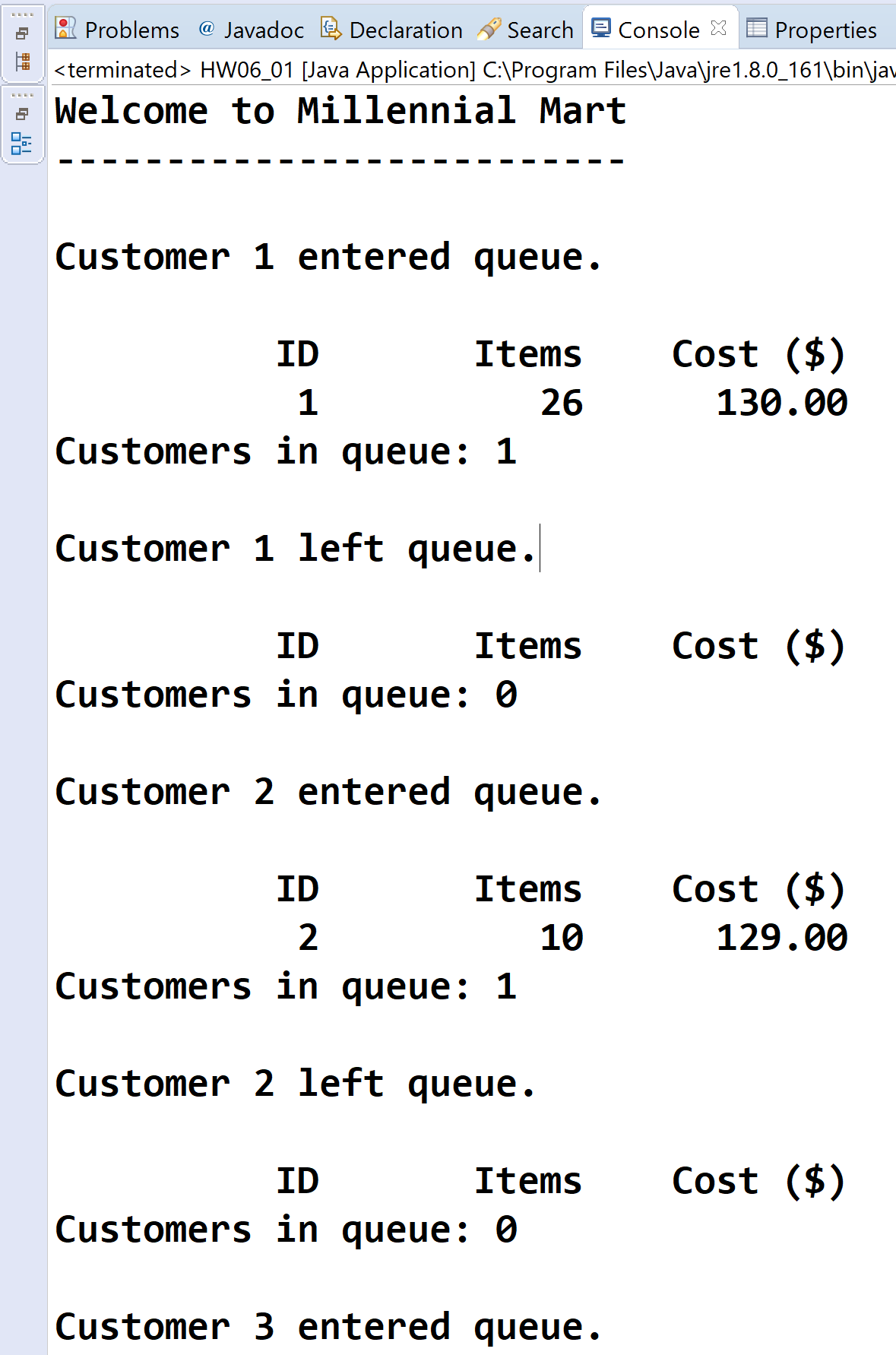
**"Total cost ($):", Customer.getTotalCost());**

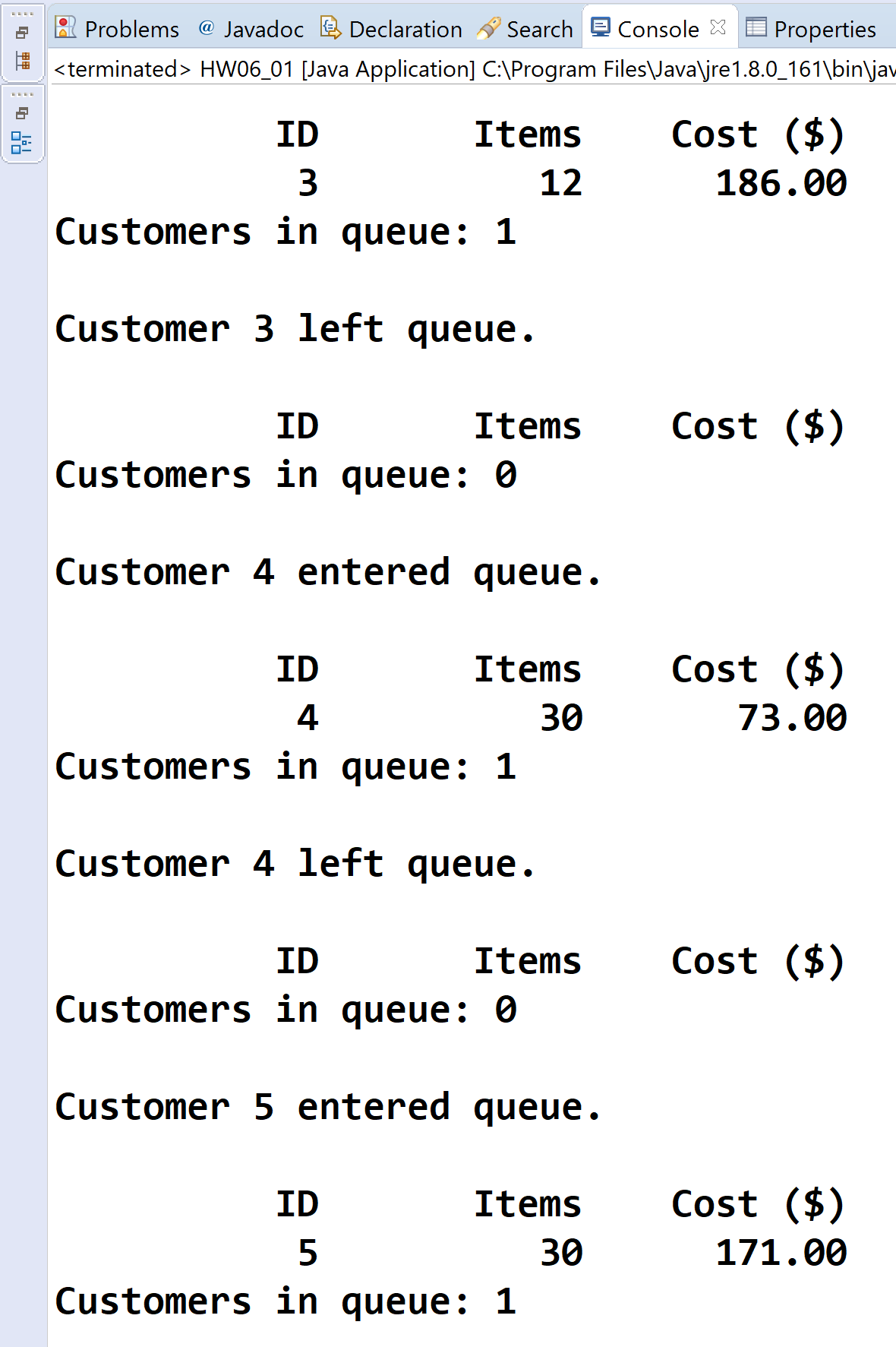
**System.out.println("\nEnd of Millennial Mart");**

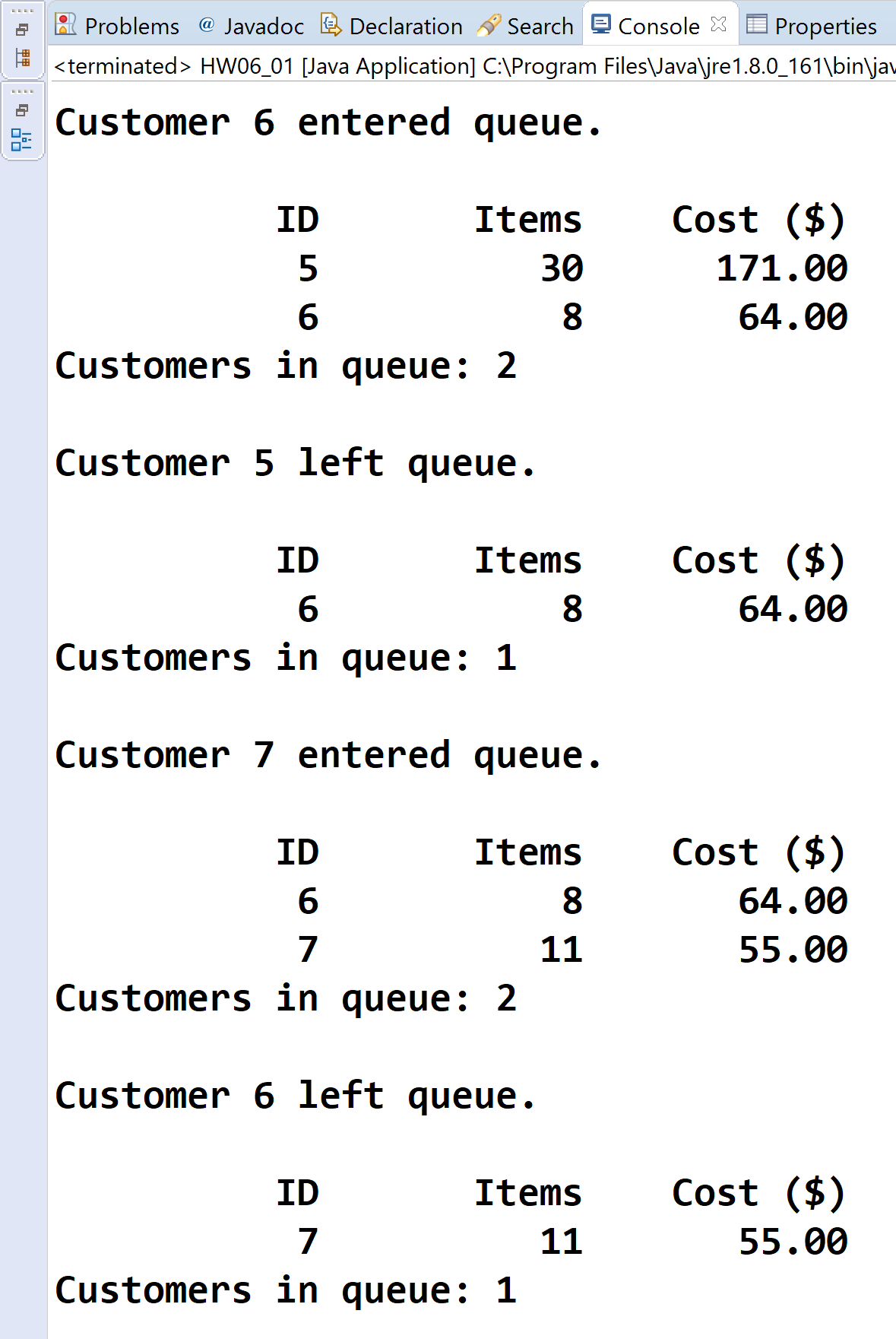
**}**

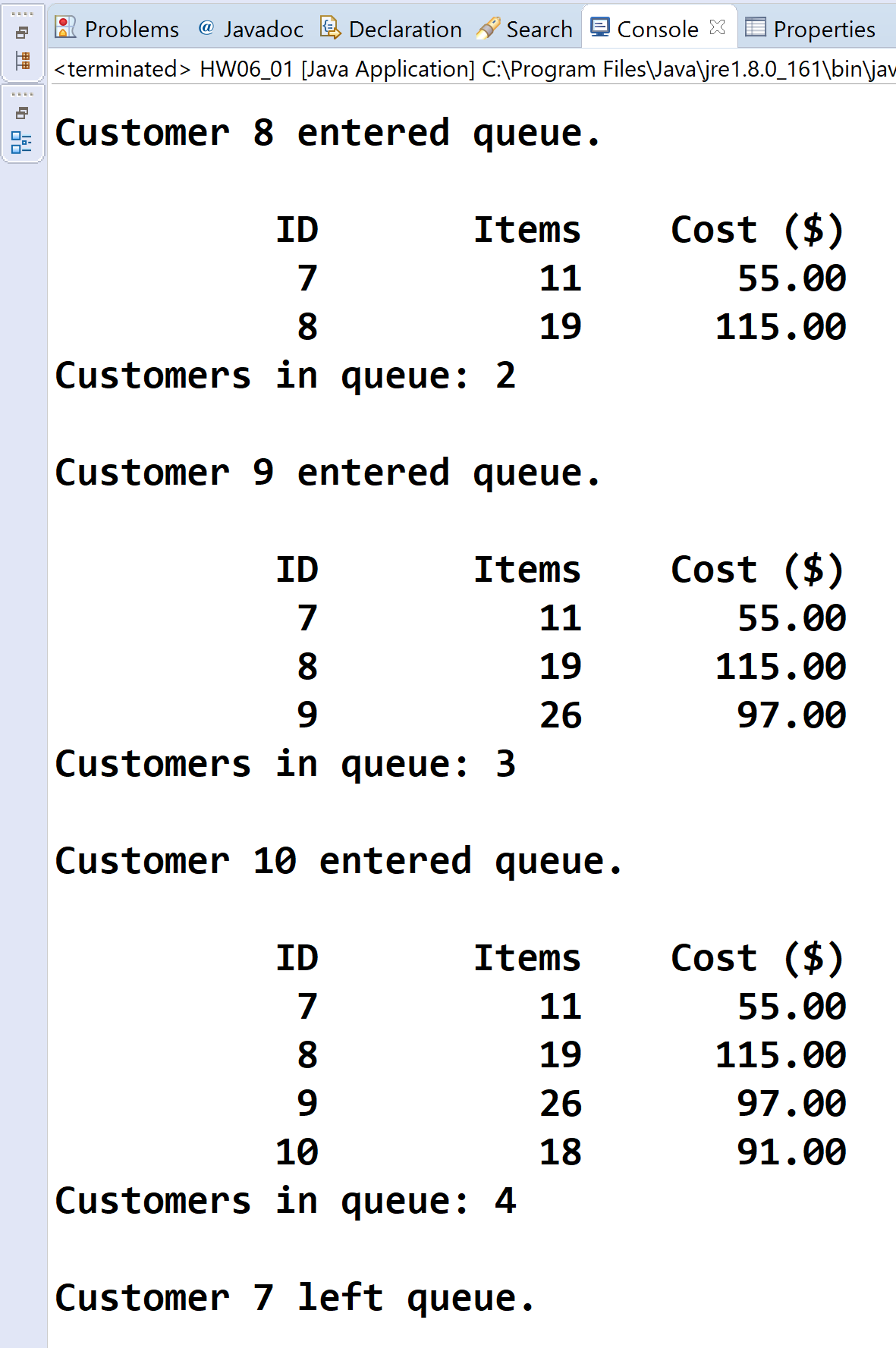
**}**

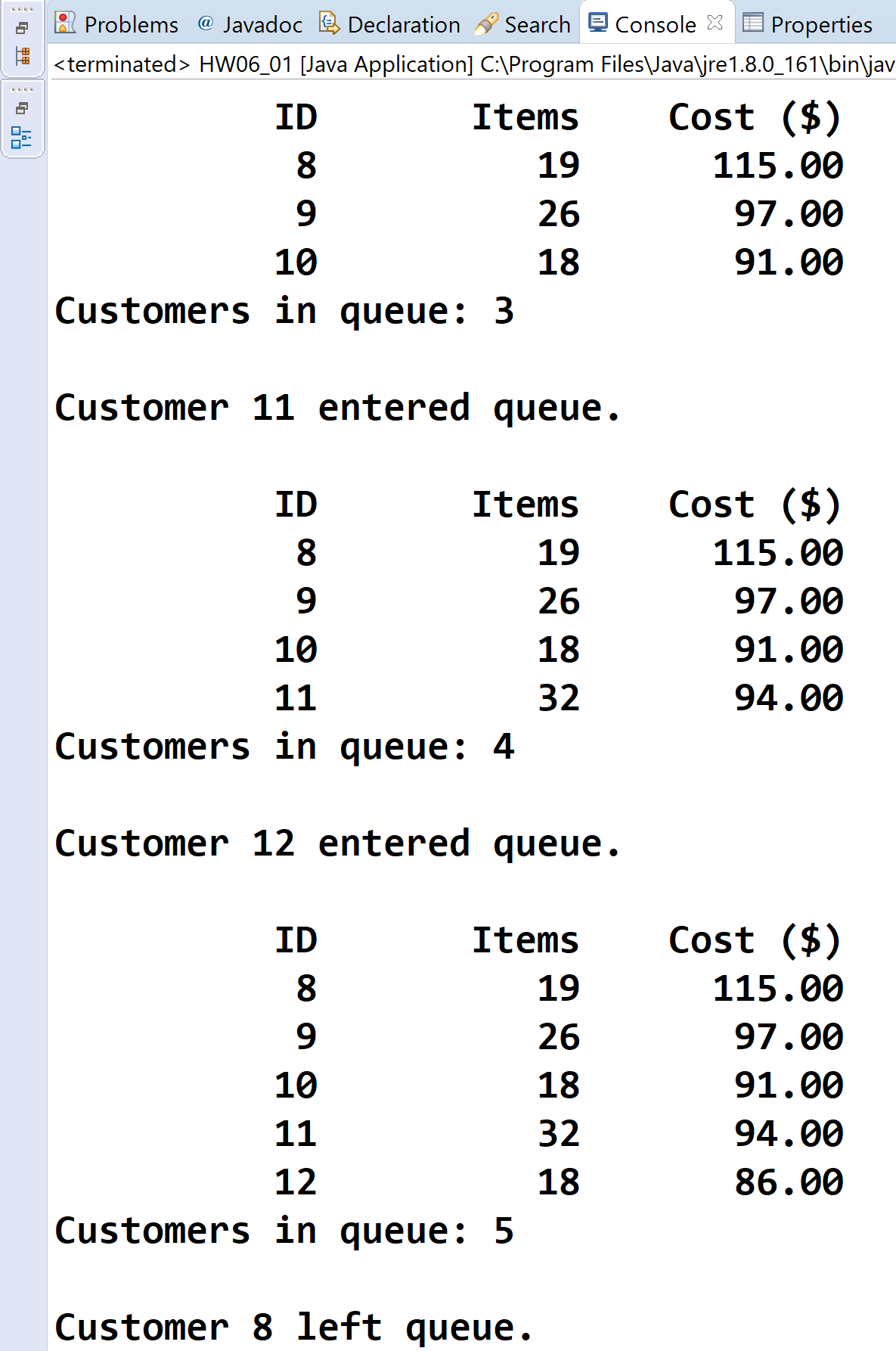
**Program output**

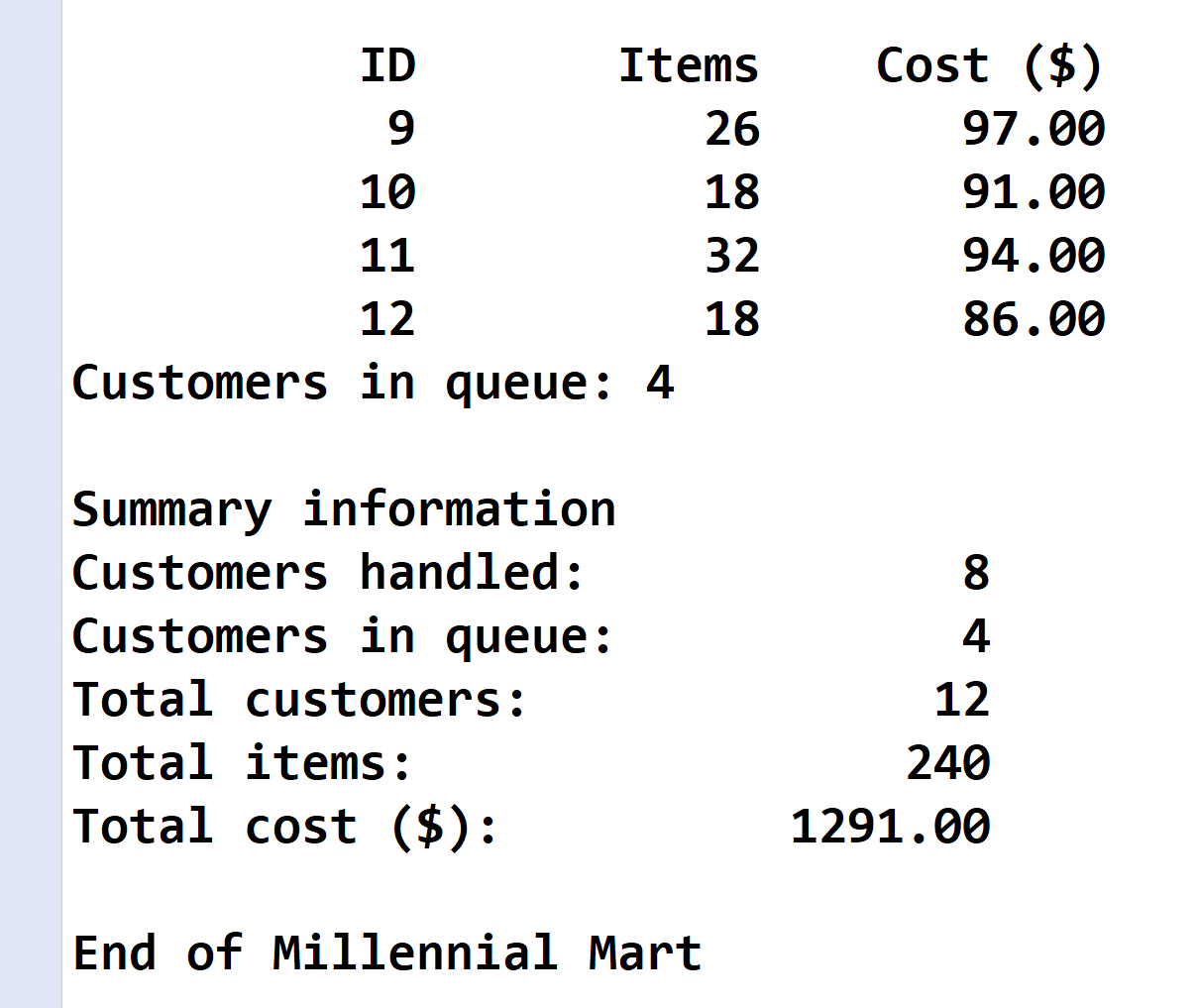












\* **Copying-and-pasting application code to a Word document**

1) From the program editor window, press **CTRL-A** and press **CTRL-C**.

2) From within the Word document, press **CTRL-V**.

\*\* **Copying-and-pasting application output to a Word document**

1) From the Eclipse main screen, maximize the Console window.

2) From the Console window, press **ALT-PrintScreen**.

3) From within the Word document, press **CTRL-V**.