



**BUSINESS  
PROFESSIONALS**  
of **AMERICA**  
Giving Purpose to Potential



# **JAVA PROGRAMMING**

## **(340)**

### **REGIONAL 2022**

#### **Production:**

Program 1: Tollway Customer Database \_\_\_\_\_ (445 points)

**TOTAL POINTS** \_\_\_\_\_ **(445 points)**

**Test Time: 90 minutes**

### **GENERAL GUIDELINES:**

*Failure to adhere to any of the following rules will result in disqualification:*

1. Member must hand in this test booklet and all printouts if any.
2. No equipment, supplies, or materials other than those specified for this event are allowed in the testing area. No previous BPA tests and/or sample tests (handwritten, photocopied, or keyed) are allowed in the testing area.
3. Electronic devices will be monitored according to ACT standards.

You will have ninety (90) minutes to complete your work.

Your name and/or school name should *not* appear on work you submit for grading.

1. Create a folder on the flash drive provided using your contestant number as the name of the folder.
2. Copy your entire solution/project into this folder.
3. Submit your entire solution/project so that the graders may open your project to review the source code.
4. Ensure that the files required to run your program are present and will execute on the flash drive provided.

\*Note that the flash drive letter may *not* be the same when the program is graded as it was when you created the program.

\*It is recommended that you use relative paths rather than absolute paths to ensure that the program will run regardless of the flash drive letter.

The graders will *not* compile or alter your source code to correct for this.  
Submissions that do *not* contain source code will *not* be graded.

### **Assumptions to make when taking this assessment:**

- There will only be one record created from the supplied data fields in the comments section.
- Users can attempt to enter in erroneous information during an input prompt.
- The program will only exit at a single point which is when they decline to retrieve the single record created.
- All getter and setter methods for the Customer and Address object have been created; the source code for these classes is not accessible. \
- If the user does not want to update the records the program will ask if they want to retrieve the records. If they answer “no” the program terminates. If they answer yes then the output (supplied in this document) will have NO reference ID.

JAVA PROGRAMMING  
REGIONAL KEY 2022

**Development Standards:**

- Your Code must use a consistent variable naming convention.
- All subroutines (if any), functions (if any), and methods (if any) must be documented with comments explaining the purpose of the method, the input parameters (if any), and the output (if any).
- If you create a class, then you must use Javadoc comments.

**Note to Graders:**

- Output will be static for the customer name and address; however, the data will vary for the monies deposited and the random reference ID generated.
- The output format should be very similar in its organization.
- Error message may differ from sample output.

**Test Case #1.a with Update Entry Error (repeats entry request until Yes or No entered)**

Customer record import successful.

Type in "Yes" if you want **update** this record:  
ENTER: Yes or No: *sssss*

Type in "Yes" if you want update this record:  
ENTER: Yes or No:

**Test Case #1.b with Retrieval Entry Error (repeats entry request until Yes or No entered)**

Type in "Yes" if you want update this record:  
ENTER: Yes or No: *no*

Do you want to **retrieve** this record?  
ENTER: Yes or No: *dgsdg*

Do you want to retrieve this record?  
ENTER: Yes or No:

**Test Case #2.a Update ("Yes") with Numerical Entry Error (repeats entry request until double or integer values entered)**

Type in "Yes" if you want update this record:  
ENTER: Yes or No: *yes*

All new records require a new deposit. How much will the customer be depositing?  
Please enter in a value between \$1.00 to \$9,999.99: *hsfhs*

Please enter a correct value.  
Please enter in a value between \$1.00 to \$9,999.99:

**Test Case #2.b Update (“Yes”) with Numerical Entry Error (program truncates all digits past hundredth place value)**

Type in "Yes" if you want update this record:

ENTER: Yes or No: *yes*

All new records require a new deposit. How much will the customer be depositing?

Please enter in a value between \$1.00 to \$9,999.99: *45.559*

Jose Montana deposited a total of \$45.55.

Reference ID: WVB28039

Do you want to retrieve this record?

ENTER: Yes or No:

**Test Case #2.c Update (“Yes”) with Numerical Entry Error Beyond Parameters (forces required entry)**

Type in "Yes" if you want update this record:

ENTER: Yes or No: *yes*

All new records require a new deposit. How much will the customer be depositing?

Please enter in a value between \$1.00 to \$9,999.99: *33333*

Please enter in a value between \$1.00 to \$9,999.99: *.01*

Please enter in a value between \$1.00 to \$9,999.99: *345.55*

Jose Montana deposited a total of \$345.55.

Reference ID: QIT44464

Do you want to retrieve this record?

ENTER: Yes or No:

JAVA PROGRAMMING  
REGIONAL KEY 2022

<b>Solution and Project</b>		
The project is present on the flash drive		10 points
The projects main class is named <b>TollwayCustomerDataBase</b>		10 points
The class helper method is named <b>setReferenceID(Customer c)</b>		10 points
The class helper method is named <b>getDepositMessage(Customer c)</b>		10 points
The class helper method is named <b>getUserStringInput()</b>		10 points
The class helper method is named <b>getUserNumberInput()</b>		10 points
The class helper method is named <b>consoleRecordCheck(Customer c)</b>		10 points
<b>Program Execution</b>		
The program runs from the USB flash drive		15 points
<i>If the program does not execute, then the remaining items in this section receive a score of zero.</i>		
The program displays a message declaring successful record import.		10 points
The program prompts and forces user to enter “yes” or “no” if they want to update the record & it is not case sensitive.		10 points
If “no” is entered for update: the program prompts and forces user to enter “yes” or “no” if they want to update the record & it is not case sensitive.		10 points
If “yes” is entered for update: the program prompts and forces user to enter “\$1.00 to \$9,999.99” the deposit amount. All data entry errors are caught.		10 points
Program displays customer name and how much was deposited properly formatted: i.e. <i>Jose Montana deposited a total of \$56.00.</i>		20 points
Transaction reference ID is randomly generated with the first three elements being letters and the remaining five are 0 to 9. Note: the letter “O” must be omitted. i.e. <i>Reference ID: KMU43187</i>		20 points
The program prompts and forces user to enter “yes: or “no” if they want to retrieve the record & it is not case sensitive.		10 points
If “no” is entered for retrieval: the program prompts says “Goodbye” and the terminates.		10 points
If “yes” is entered for retrieval: the program prints the entire record, including the formatted deposit, and the same randomly generated reference ID.		20 points
Output matches required format.		20 points
<b>Source Code Review</b>		
The source code is properly commented		
A comment containing the contestant number is present		10 points
Methods and code sections are commented		20 points

JAVA PROGRAMMING  
REGIONAL KEY 2022

Code uses try... catch for exception handling for <b>getUserNumberInput()</b> when entering numbers. All values entered beyond given range are not accepted; and all values entered into the thousandths decimal place are truncated.		30 points
<b>getUserNumberInput()</b> : All values entered beyond given range are not accepted; and all values entered into the thousandths decimal place are truncated.		20 points
<b>main (String args [ ])</b> : Customer object is correctly constructed from data entry into its given attributes, and also properly passes data into attributes of Address object (attribute in Customer).		10 points
<b>getDepositMessage(Customer c)</b> method retrieves required fields from Customer and also formats the deposit amount to US currency including "\$" and "," plus the cents.		30 points
<b>setDepositCustomerRecord(Customer c)</b> : calls on <i>getUserNumberInput()</i> for data entry and deposits values into customer object; and calls <i>setReferenceID(Customer c)</i> to create reference ID.		20 points
<b>setDepositCustomerRecord(Customer c)</b> : calls on <i>getDepositMessage(c)</i> for String to print out for feedback to the user.		10 points
<b>setReferenceID(Customer c)</b> : randomly generates three capital letters, and omits the latter "O".		20 points
<b>setReferenceID(Customer c)</b> : randomly generates an integer greater than 9,999 and less than 100,000; concatenates with the three letters and returns the value. The reference ID must be in the following format(L: letter & N: number): LLLNNNNNN		20 points
<b>consoleRecordCheck(Customer c)</b> : retrieves all data from the customer object using its getter methods. Places information in proper format		10 points
<b>consoleRecordCheck(Customer c)</b> : formats the deposit retrieved from customer object into US currency including "\$" and "," plus the cents.		20 points

**Total Points = \_\_\_\_ / 445 points**

# JAVA PROGRAMMING

## REGIONAL KEY 2022

### Suggested Solution

```
1 import java.util.*;
2 import java.text.NumberFormat;    //They can also use other formatting methodology
3 import java.util.Locale;
4 import java.util.Random;          //They can also use other random number generation tactics
5
6 public class TollwayCustomerDataBase
7 {
8
9     static Scanner sc = new Scanner(System.in);
10    public static void main (String args [])
11    {
12
13        //Pre loaded customer data: students will only get the string literals
14        Customer cust1 = new Customer("Montana","Jose", "Ford", "F150", "ABC-123", 55.00);
15        cust1.address = new MailingAddress ("3445 Rockhill Rd.", "Santa Fe", "New Mexico", "77777");
16
17
18        String yesNo= "";
19        //Checks that the customer object was created //////////////////////////////////////
20        if(cust1 != null && cust1.address != null)
21            System.out.println("Customer record import successful.");
22
23
24        //Prompts the user to see if they want to update the record //////////////////////////////////////
25        do{
26            System.out.print("\nType in \"Yes\" if you want update this record:\nEnter: Yes or No: ");
27            yesNo = getUserStringInput();    //helper method call
28            if(yesNo.equals("yes"))
29                setDepositCustomerRecord(cust1);    //helper method call
30            }while (yesNo.equals("yes") == false && yesNo.equals("no") == false);
31
32        //Prompts the user to see if they want to retrieve the record //////////////////////////////////////
33        do{
34            System.out.print("\nDo you want to retrieve this record?\nEnter: Yes or No: ");
35            yesNo = getUserStringInput();    //helper method call
36            if(yesNo.equals("yes"))
37                consoleRecordCheck(cust1);    //helper method call
38            else if(yesNo.equals("no")){
39                System.out.print("\nGoodbye!");
40                System.exit(0);    //this is the only program exit
41            }
42            } while (yesNo.equals("yes") == false && yesNo.equals("no") == false);
43
44    }
45
46    //Prompts the user to submit a deposit: calls other helper methods //////////////////////////////////////
47
48    private static void setDepositCustomerRecord(Customer c)
49    {
50        System.out.println("\nAll new records require a new deposit. How much will the customer be depositing? ");
51        double newDeposit = getUserNumberInput();    //helper method call
52        c.setDeposit(newDeposit);
53        setReferenceID(c);    //helper method call
54        String message = getDepositMessage(c);    //helper method call
55        System.out.println(message);
56
57    }
```

## JAVA PROGRAMMING

### REGIONAL KEY 2022

```
59 //Creates and stores the reference ID (random letters and numbers) //////////////////////////////////
60 //helper method
61 private static void setReferenceID(Customer c)
62 {
63     String referenceString = "";
64     char randomReferenceLetter;
65     Random rand = new Random();
66
67     for (int i=0; i<3; i++)
68     {
69         do
70         {
71             randomReferenceLetter = (char)(rand.nextInt(26)+'A');
72         } while (randomReferenceLetter == '0');
73         referenceString += randomReferenceLetter;
74     }
75     int randomReferenceNumber;
76     do{ randomReferenceNumber = rand.nextInt(100000);
77     } while(randomReferenceNumber<10000);
78     referenceString += randomReferenceNumber;
79     c.setReferenceTicket(referenceString);
80 }
81
82 //Displays the message in console after a successful deposit entry //////////////////////////////////
83 //helper method
84 private static String getDepositMessage(Customer c)
85 {
86
87     NumberFormat d = NumberFormat.getCurrencyInstance(new Locale("en", "US"));
88     return "\n" + c.getFN() + " " + c.getLN() + " deposited a total of " + d.format(c.getDeposit())+" \nReference ID: " +
89     c.getReferenceTicket();
90 }
91
92
93 //Gets the user input for the yes or no prompts and turns it into LC //////////////////////////////////
94 //helper method
95 private static String getUserStringInput()
96 {
97     String temp = sc.nextLine().toLowerCase();
98
99     return temp;
100 }
101
102 //Gets the user input for the deposit prompt //////////////////////////////////
103 //helper method
104 private static double getUserNumberInput()
105 {
106     double temp;
107     while(true){
108         try{
109             do{
110                 System.out.print("Please enter in a value between $1.00 to $9,999.99: ");
111                 temp = sc.nextDouble();
112                 sc.nextLine();
113                 temp = 0.01 * Math.floor(temp *100);
114             }while(temp >9999 || temp <1);
115             return temp;
116
117         }
118         catch(InputMismatchException e)
119         {
120             sc.next();
121             System.out.println("\nPlease enter a correct value.");
122         }
123     }
124 }
125
126 }
```



## JAVA PROGRAMMING

### REGIONAL KEY 2022

```
126
127 //Prints the final record to the console by using data from the object //////////////////////////////////////
128 private static void consoleRecordCheck(Customer c)
129 {
130     NumberFormat d = NumberFormat.getCurrencyInstance(new Locale("en", "US"));
131     System.out.println("\n"+c.getFN() + " " + c.getLN());
132     System.out.println(c.address.getAdd());
133     System.out.println("Deposit: " + d.format(c.getDeposit()));
134     System.out.println("Car Information: " + c.getMake() + " " + c.getModel());
135     System.out.println("Reference ID: " + c.getReferenceTicket());
136     //System.out.println(c); //This prints the record from its toString method
137 }
138 }
139
140
141
142
143
144
145
146
147 class Customer
148 {
149     private String last_Name;
150     private String first_Name;
151     private String carMake;
152     private String carModel;
153     private String carLicensePlate;
154     private double deposit;
155     private String reference = "";
156     //private List<String> references = new ArrayList<>(); //This is in the next round
157
158     public MailingAddress address;
159
160     public Customer (String ln, String fn, String cMa, String cMo, String lp, Double de)
161     {
162         this.last_Name = ln;
163         this.first_Name = fn;
164         this.carMake = cMa;
165         this.carModel = cMo;
166         this.carLicensePlate = lp;
167         this.address = new MailingAddress();
168         this.deposit = de;
169     }
170
171     public String getLN() //gets last name
172     {
173         return this.last_Name;
174     }
175
176     public void setLN(String ln) //sets last name
177     {
178         this.last_Name = ln;
179     }
180
181     public String getFN() //gets first name
182     {
183         return this.first_Name;
184     }
185
186     public void setFM(String fn) //sets first name
187     {
188         this.first_Name = fn;
189     }
190
191     public String getMake() //gets make of car
192     {
193         return this.carMake;
194     }
195
196     public void setMake(String ma) //sets make of car
197     {
198         this.carMake = ma;
199     }
200
```

## JAVA PROGRAMMING

### REGIONAL KEY 2022

```
200
201 public String getModel() //gets model of car
202 {
203     return this.carModel;
204 }
205
206 public void setModel(String mo) //sets model of car
207 {
208     this.carModel = mo;
209 }
210
211 public Double getDeposit() //gets deposit
212 {
213     return this.deposit;
214 }
215
216 public void setDeposit(Double d) //sets deposit amount
217 {
218     this.deposit = d;
219 }
220
221 public String getLicensePlate() //gets plate #
222 {
223     return this.carLicensePlate;
224 }
225
226 public void setLicensePlate(String lp) //sets plate #
227 {
228     this.carLicensePlate = lp;
229 }
230
231 public void setReferenceTicket(String d)
232 {
233
234     reference = d;
235 }
236
237 public String getReferenceTicket()
238 {
239     return reference;
240 }
241
242
243
244 //This is not used in the program. If a student tries to print the object we will get a similiar output and its wrong
245 /*
246     Customer[ Jose Montana
247     Ford F150
248     ABC-123
249     456.0]
250 */
251 public String toString()
252 {
253     return this.getClass().getName() + "[ " + this.getFN() + " "
254         + this.getLN() + "\n" + this.getMake() + " " + this.getModel() +
255         "\n" + this.getLicensePlate() + "\n" + this.getDeposit() + " ]";
256 }
257 }
258
-----
```

## JAVA PROGRAMMING

### REGIONAL KEY 2022

```
265
266 class MailingAddress
267 {
268     private String street;
269     private String city;
270     private String state;
271     private String zipCode;
272
273     public MailingAddress()
274     {
275         this.street = " ";
276         this.city = " ";
277         this.state = " ";
278         this.zipCode = " ";
279     }
280
281     public MailingAddress(String s, String c, String st, String zc)
282     {
283         this.street = s;
284         this.city = c;
285         this.state = st;
286         this.zipCode = zc;
287     }
288
289     public String getAdd()
290     {
291         return this.street + "\n" + this.city + ", " + this.state + " " + this.zipCode;
292     }
293
294 }
295
296
```