



JAVA PROGRAMMING

(340)

REGIONAL 2023

PRODUCTION:

NFT Market Place

_____ (425 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. Failure to do so will result in disqualification.
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.

Note to Graders:

- Output will be static for the username and wallet address; however, the data will vary for the floor price, rarity, token type, and the NFT address generated.
- NOTE: the NFT address should have the first three letters (not case sensitive) of the username and have the non hyphenated alphanumeric values of the token type at the end
- The output format should be very similar in its organization.
- Error message may differ from sample output.
- Ignore spacing differences or grammar errors
- *When reading code: you might consider using the “Regional Solutions.rtf” file due to ease of readability compared to the embedded solution at the end of this document which is heavily effected by the template margins.*

Test Case #1.1: program executes and displays welcome message with the username (given) and wallet address (given):

Welcome Ana1234 to the NFT Market Place form. Your wallet is now connected (verify below):

1Awyd1QWR5gcfrn1UmL8dUBj2H1eVKtQhg

Test Case #1.2 Error Entry: program displays prompt asking user to create a new NFT; enter a number to force program to ask prompt again:

Type in "Yes" if you want create a new NFT:

ENTER: Yes or No: **4**

Type in "Yes" if you want create a new NFT:

ENTER: Yes or No:

Test Case #1.3 Error Entry: program displays prompt asking user to create a new NFT; enter wrong text to force program to ask prompt again:

Type in "Yes" if you want create a new NFT:

ENTER: Yes or No: **ffffff**

Type in "Yes" if you want create a new NFT:

ENTER: Yes or No:

Test Case #1.4 Correct Entry: program displays prompt asking user to enter in floor price, if a ‘y’, ‘Y’, ‘yes’, or ‘YES’ is entered (NOTE: this needs to noncase sensitive):

Type in "Yes" if you want create a new NFT:

ENTER: Yes or No: **y**

All new NFT's require a new Floor Price. How much will be the NFT's floor price?

Please enter in a value between \$0.50 to \$99,999.99:

Test Case #1.5 Error Entry (Text, High Breach, Low Breach): program displays prompt asking user to create a new NFT, enter wrong text to force program to ask prompt again:

All new NFT's require a new Floor Price. How much will be the NFT's floor price?

Please enter in a value between \$0.50 to \$99,999.99: *fghd*

Please enter a correct value.

Please enter in a value between \$0.50 to \$99,999.99: **222333**

Please enter in a value between \$0.50 to \$99,999.99:

Please enter in a value between \$0.50 to \$99,999.99: **.33**

Please enter in a value between \$0.50 to \$99,999.99:

Test Case #1.6 Correct Entry: program displays prompt asking user to enter 1 or 2 for token type:

Please enter in a value between \$0.50 to \$99,999.99: **456.23**

Please enter in a 1 for ERC-721 or a 2 for a BEP-721 token type (NOTE all rational values will be truncated to the whole number):

Test Case #1.7 Error Entry (Text, High Breach, Low Breach): program displays prompt and keeps asking user to choose token type:

Please enter in a 1 for ERC-721 or a 2 for a BEP-721 token type (NOTE all rational values will be truncated to the whole number): *hgfhf*

Please enter a correct value.

Please enter in a 1 for ERC-721 or a 2 for a BEP-721 token type (NOTE all rational values will be truncated to the whole number): **4**

Please enter in a 1 for ERC-721 or a 2 for a BEP-721 token type (NOTE all rational values will be truncated to the whole number): **0**

Please enter in a 1 for ERC-721 or a 2 for a BEP-721 token type (NOTE all rational values will be truncated to the whole number):

Test Case #1.8 Correct Entry: program displays prompt asking user to enter rarity type:

Please enter in a 1 for ERC-721 or a 2 for a BEP-721 token type (NOTE all rational values will be truncated to the whole number): **2**

Your NFT contract address: Anax4066xx5246xx8387xx9971xBEP721

All new NFT's require a rarity. Choose the number for the level of rarity:

(6)Mythic, (5)Legendary, (4)Epic, (3)Rare, (2)Uncommon, (1)Common

Please enter in a value between 1 to 6 (NOTE all rational values will be truncated to the whole number):

Test Case #1.9 Error Entry (Text, High Breach, Low Breach): program displays prompt and keeps asking user to choose rarity type:

All new NFT's require a rarity. Choose the number for the level of rarity:

(6)Mythic, (5)Legendary, (4)Epic, (3)Rare, (2)Uncommon, (1)Common

Please enter in a value between 1 to 6 (NOTE all rational values will be truncated to the whole number): **ghjghj**

Please enter a correct value.

Please enter in a value between 1 to 6 (NOTE all rational values will be truncated to the whole number): **7**

Please enter in a value between 1 to 6 (NOTE all rational values will be truncated to the whole number): **0**

Test Case #1.8 Correct Entry: program displays prompt asking user if they want to see the profile (NOTE: rational numbers within the expected range are tolerable; they will be truncated to the ones place value; in this case it will be 5):

Please enter in a value between 1 to 6 (NOTE all rational values will be truncated to the whole number): **5.5**

Do you want to see the profile for this user?

ENTER: Yes or No:

Test Case #1.10 Error Entry (wrong entry types): program displays prompt and keeps asking user if they would like to profile:

Do you want to see the profile for this user?

ENTER: Yes or No: **hjkkjh**

Do you want to see the profile for this user?

ENTER: Yes or No: **678687**

Do you want to see the profile for this user?

ENTER: Yes or No:

Test Case #1.11 Correct Entry: program displays creator records (NOTE: yes or no prompts should except all lowercase and uppercase entries; expected text entries can only be 'y','n','no' or "yes"):

Do you want to see the profile for this user?

ENTER: Yes or No: **y**

***** CREATOR RECORDS *****

Username: Ana1234

Wallet Address: 1Awyd1QWR5gcfrn1UmL8dUBj2H1eVKtQhg

Collection Name: Bored BPA Yacht Club

NFT Address: Anax4066xx5246xx8387xx9971xBEP721

Floor Price: \$456.23

Rarity/Collection: Legendary

Test Case #2 Correct Entry to End Program: entering negative to the following prompts the program terminates and gives a goodbye message “Have a great day and see you on the Moon!”

Welcome Ana1234 to the NFT Market Place form. Your wallet is now connected (verify below):

1Awyd1QWR5gcfrn1UmL8dUBj2H1eVKtQhg

Type in "Yes" if you want create a new NFT:

ENTER: Yes or No: **N**

Do you want to see the profile for this user?

ENTER: Yes or No: **N**

Have a great day and see you on the Moon!

Test Case #3 Correct Entry for NO to NFT and YES to see Creator Records: entering negative to the NFT creation prompt and then affirmative to the prompt to see the record. Notice the record does not have any information about NFTs per the requirement.

Welcome Ana1234 to the NFT Market Place form. Your wallet is now connected (verify below):

1Awyd1QWR5gcfrn1UmL8dUBj2H1eVKtQhg

Type in "Yes" if you want create a new NFT:

ENTER: Yes or No: *n*

Do you want to see the profile for this user?

ENTER: Yes or No: *y*

***** CREATOR RECORDS *****

Username: Ana1234

Wallet Address: 1Awyd1QWR5gcfrn1UmL8dUBj2H1eVKtQhg

Collection Name: Bored BPA Yacht Club

Test Case #4 Correct Entry for YES to NFT and NO to see Creator Records: entering affirmative to the NFT creation prompt and the subsequent NFT requirements. The negative response to see the record causes the program to terminate with the goodbye message.

Welcome Ana1234 to the NFT Market Place form. Your wallet is now connected (verify below):

1Awyd1QWR5gcfrn1UmL8dUBj2H1eVKtQhg

Type in "Yes" if you want create a new NFT:

ENTER: Yes or No: *y*

All new NFT's require a new Floor Price. How much will be the NFT's floor price?

Please enter in a value between \$0.50 to \$99,999.99: **456**

Please enter in a 1 for ERC-721 or a 2 for a BEP-721 token type (NOTE all rational values will be truncated to the whole number): **2**

Your NFT contract address: Anax4194xx1981xx1166xx3073xBEP721

All new NFT's require a rarity. Choose the number for the level of rarity:

(6)Mythic, (5)Legendary, (4)Epic, (3)Rare, (2)Uncommon, (1)Common

Please enter in a value between 1 to 6 (NOTE all rational values will be truncated to the whole number): **4**

Do you want to see the profile for this user?

ENTER: Yes or No: **n**

Have a great day and see you on the Moon!

Solution and Project		
The project is present on the flash drive	_____	10 points
The projects main class is named NFTMarketPlace	_____	10 points
The class helper method is named setFloorPriceNFTDataRecord(NFTData c)	_____	10 points
The class helper method is named setNFTRarityRecord(NFTData c)	_____	10 points
The class helper method is named setContractAddress(NFTData trader)	_____	10 points
The class helper method is named getUserStringInput()	_____	10 points
The class helper method is named getUserNumericalInput(double low, double high, String message)	_____	10 points
The class helper method is named consoleRecordCheck(NFTData c)	_____	10 points
Program Execution		
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 welcoming with UserName and confirming the wallet is now connected.	_____	10 points
The program prompts and forces user to enter “yes” or “no” if they want to create a new NFT.	_____	10 points
If “no” or ‘n’ is entered for update: the program prompts and forces user to enter “yes”, ‘y’, ‘n’, or “no” if they want to view the profile for the user.	_____	10 points
If “yes” or ‘y’ is entered for update: the program prompts and forces user to enter “\$0.50 to \$99,999.99” the deposit amount. All data entry errors are caught.	_____	10 points
Program displays “Creator Records”(aka profile) with NFT information when appropriate, no NFT information when appropriate.	_____	30 points
NFT Address is a combination between first three letters of username, randomly generated four sets of 4 digits that use the delimiter of ‘x’ (‘xx’ is permissible). The Address has the non hyphenated token type concatenated at the end . i.e. NFT Address: <i>Anax7061xx7769xx1404xx1748xBEP721</i>	_____	30 points
The program prompts and forces user to enter “yes: or “no” if they want to view the profile & it is not case sensitive.	_____	10 points
If “no” or ‘n’ is entered for retrieval: the program prompts says “Have a great day and see you on the Moon!” and then terminates.	_____	10 points
Floor price is displayed for US Dollars	_____	10 points
Output matches required format.	_____	20 points

Source Code Review		
The source code is properly commented		
A comment containing the contestant number is present		10 points
Methods and code sections are commented		20 points
Code uses try... catch for exception handling for getUserNumericalInput(double, double, String) when entering numbers. All values entered beyond given range are not accepted based upon high and low parameters; and all rational values		30 points
All rational numbers entered via getUserNumericalInput(double, double, String) are truncated when being passed back to integer variable (NOTE: easiest method is to cast the returned value)		10 points
Appropriate messages appear (3 total: floor price double, token type int, rarity type int) via getUserNumericalInput(double, double, String) via passed String		10 points
Code uses getUserStringInput() to gather all “yes”, “no”, ‘n’ or ‘y’ entries and passes result back as String.		10 points
main (String args []) : NFTData object is correctly constructed from data entry into its given attributes, and also properly passes data into attribute of WalletAddress object (attribute in NFTData).		10 points
main (String args []) : program checks to see if NFTData object and the WalletAddress have been created. Displays a welcome message with getting the Username and the wallet address from the getter methods		10 points
setContractAddress(NFTData trader) : method creates the NFT address based upon the given criteria (first three letters of username, four sets of 4 random digits separated by ‘x’ delimiter, non hyphenated token type concatenated at the end)		30 points
consoleRecordCheck(NFTData c) : formats floor price to US dollars using NumberFormat		20 points
consoleRecordCheck(NFTData c) : prints all required data via accessing getter methods of the NFTData object.		20 points
Total Points		/425 points

Suggested Solution

```
1 import java.util.*;
2 import java.text.NumberFormat;
3 import java.util.Locale;
4 import java.util.Random;
5
6 public class NFTMarketPlace
7 {
8     //VARIABLES ARE GIVEN
9     public static String rarity [] =
10 {"Common","Uncommon","Rare","Epic","Legendary","Mythic"} ;
11 private static double lowFloor = 0.5;
12 private static double highFloor = 99999.99;
13 private static String floorMssg = "Please enter in a value between $0.50 to $99,999.99: ";
14 private static int lowRarity = 1;
15 private static int highRarity = 6;
16 private static String rarityMssg = "Please enter in a value between 1 to 6"
17     +(NOTE all rational values will be truncated to the whole number): ";
18 private static int lowToken = 1;
19 private static int highToken = 2;
20 private static String tokenMssg = "Please enter in a 1 for ERC-721 or a 2 for a BEP-721
21 token type"+
22     "(NOTE all rational values will be truncated to the whole number): ";
23 static Scanner sc = new Scanner(System.in);
24
25 public static void main (String args [])
26 {
27     //STUDENT WILL MAKE THIS
28     //Pre loaded customer data: students will only get the string literals
29     NFTData client = new NFTData("Ana1234","Bored BPA Yacht Club");
30     client.walletAddress.setAdd("1Awyd1QWR5gcfrn1UmL8dUBj2H1eVKtQhg");
31     String yesNo= "";
32
33     //STUDENT WILL MAKE THIS
34     //Checks that the customer object was created
35     if(client != null && client.walletAddress != null)
36         System.out.println("Welcome " + client.getUN()+ " to the NFT Market Place form."+
37             "Your wallet is now connected (verify below):\n"+client.walletAddress.getAdd());
38
39     //THIS IS GIVEN
40     //Prompts the user to see if they want to update the record
41     do{
```

```

40      System.out.print("\nType in \"Yes\" if you want create a new NFT:\nEnter: Yes or
No: ");
41      yesNo = getUserStringInput(); //helper method created by the student
42      if(yesNo.equals("yes")|| yesNo.equals("y"))
43      {
44          setFloorPriceNFTDataRecord(client); //helper method call given
45          setNFTRarityRecord(client);
46      }
47      }while (yesNo.equals("yes") == false && yesNo.equals("no") == false &&
48              yesNo.equals("y") == false && yesNo.equals("n") == false);
49
50      //THIS IS GIVEN
51      //Prompts the user to see if they want to retrieve the record
52      do{
53          System.out.print("\nDo you want to see the profile for this user?\nEnter: Yes or No:
");
54          yesNo = getUserStringInput(); //helper method created by the student
55          if(yesNo.equals("yes")|| yesNo.equals("y"))
56          {
57              consoleRecordCheck(client); //helper method call created by student
58          }
59
60          else if(yesNo.equals("no")|| yesNo.equals("n")){
61              System.out.print("\nHave a great day and see you on the Moon!");
62              System.exit(0);
63          }
64      } while (yesNo.equals("yes") == false && yesNo.equals("no") == false &&
65              yesNo.equals("y") == false && yesNo.equals("n") == false);
66
67      }
68      //THIS IS GIVEN
69      //Prompts the user to submit a floorPrice: calls other helper methods
70      private static void setFloorPriceNFTDataRecord(NFTData c)
71      {
72          System.out.println("\nAll new NFT's require a new Floor Price."+
73              "How much will be the NFT's floor price? ");
74          double numberInput = getUserNumericalInput(lowFloor, highFloor, floorMssg);
75          c.setFloorPrice(numberInput);
76          setContractAddress(c);
77
78      }
79      //THIS IS GIVEN
80      //Sets the rarity of the NFT
81      //Students will need to make the get method to have user input for the NFT rarity value

```

```
82 private static void setNFTRarityRecord(NFTData c)
83 {
84     System.out.println("\nAll new NFT's require a rarity. Choose the number for the level of
rarity:\n"+
85         "(6)Mythic, (5)Legendary, (4)Epic, (3)Rare, (2)Uncommon, (1)Common");
86     int rarityInt = (int)getUserNumericalInput(lowRarity, highRarity, rarityMssg);
87     c.setRarity(rarity[rarityInt-1]);
88 }
89
90 //STUDENT WILL MAKE THIS
91 //Creates and stores the reference ID (random letters and numbers)
92 //Object Needs to store either a ERC-721 or BEP-721 token.
93 //Address is four parts, x's are delimiters
94 private static void setContractAddress(NFTData trader)
95 {
96     String referenceString = "";
97     char randomReferenceLetter = 'x';
98     Random rand = new Random();
99     int temp = 0;
100     String token = "";
101
102     do{
103
104         temp = (int)getUserNumericalInput(lowToken, highToken, tokenMssg);
105
106     }while(temp >2 || temp <1);
107     if(temp == 1)
108         token = "ERC721";
109     else token = "BEP721";
110     // Blocks of four digit codes for the address name that is enclosed wiht 'x' as delimiters
111     referenceString = trader.getUN().substring(0,3);
112     for (int i=0; i<4; i++)
113     {
114         int randomReferenceNumber = 0;
115         do{ randomReferenceNumber = rand.nextInt(10000);
116             } while(randomReferenceNumber<1000);
117         referenceString += "x" + randomReferenceNumber + "x";
118     }
119     referenceString += token;
120     System.out.print("Your NFT contract address: " +referenceString +"\n");
121     trader.setNFTAddress(referenceString);
122
123 }
124
```

```
125 //STUDENT WILL MAKE THIS
126 //Gets the user input for the yes or no prompts and turns it into LC
127 //helper method
128 private static String getUserStringInput()
129 {
130     String temp = sc.nextLine().toLowerCase();
131
132     return temp;
133 }
134
135 //STUDENT WILL MAKE THIS
136 //Gets use to input numerical responses.
137 //The return value is double for simplicity purposes.
138 //The return value will need to be cast to int where appropriate
139 //An example is given in the initial code
140 private static double getUserNumericalInput(double low, double high, String message)
141 {
142     double temp;
143     while(true){
144         try{
145             do{
146                 System.out.print(message);
147                 temp = sc.nextDouble();
148                 sc.nextLine();
149                 temp = 0.01 * Math.floor(temp * 100);
150             }while(temp > high || temp < low);
151             return temp;
152         }
153     }
154     catch(InputMismatchException e)
155     {
156         sc.next();
157         System.out.println("\nPlease enter a correct value.");
158     }
159 }
160
161 }
162
163
164 //STUDENT WILL MAKE THIS
165 //Prints the final record to the console by using data from the object
166 private static void consoleRecordCheck(NFTData c)
167 {
168     NumberFormat d = NumberFormat.getCurrencyInstance(new Locale("en", "US"));
```

```

169     System.out.println("\n*****"+
170         "*****");
171     System.out.println("***** CREATOR RECORDS "+
172         "*****");
173     System.out.println("\nUsername: " + c.getUN());
174     System.out.println("Wallet Address: " + c.walletAddress.getAdd());
175     System.out.println("Collection Name: " + c.getCollection());
176     if(c.getNFTAddress() != null) //This will determine to print NFT information or not
177     {
178         System.out.println("NFT Address: " + c.getNFTAddress());
179         System.out.println("Floor Price: " + d.format(c.getFloorPrice()));
180         System.out.println("Rarity/Collection: " + c.getRarity());
181     }
182     System.out.println("\n*****"+
183         "*****");
184     System.out.println("*****"+
185         "*****");
186 }
187 }
188
189
190
191 //////////////////////////////////////
192 //////////////////////////////////////
193 /*****NFTData*****/
194 /*Student will only get the .class files
195 for the following object classes */
196 //////////////////////////////////////
197 //////////////////////////////////////
198 class NFTData
199 {
200     private String userName;
201     private String productName;
202     private String rarity;
203     private String collectionName;
204     private double floorPrice;
205     private String tokenType;
206     private String nftAddress = null;
207
208
209     public WalletAddress walletAddress;
210
211     public NFTData (String un, String co)
212     {

```

```
213     this.userName = un;
214     this.productName = null;
215     this.rarity = null;
216     this.collectionName = co;
217     this.tokenType = null;
218     this.floorPrice = 0;
219     this.walletAddress = new WalletAddress();
220
221 }
222 //LONG CONTRUCTORVERSION NOT USED IN TEST
223 public NFTData (String un, String pn, String ra,
224                 String co, String bc, Double fp)
225 {
226     this.userName = un;
227     this.productName = pn;
228     this.rarity = ra;
229     this.collectionName = co;
230     this.tokenType = bc;
231     this.floorPrice = fp;
232     this.walletAddress = new WalletAddress();
233 }
234
235 public String getUN() //gets username
236 {
237     return this.userName;
238 }
239
240 public void setUN(String un) //sets username
241 {
242     this.userName = un;
243 }
244
245 public String getPN() //gets productname
246 {
247     return this.productName;
248 }
249
250 public void setPN(String pn) //sets productname
251 {
252     this.productName = pn;
253 }
254
255 public String getRarity() //gets the rarity
256 {
```



```
257     return this.rarity;
258 }
259
260 public void setRarity(String ra) //sets rarity
261 {
262     this.rarity = ra;
263 }
264
265 public String getCollection() //gets collection name
266 {
267     return this.collectionName;
268 }
269
270 public void setCollection(String co) //sets collection name
271 {
272     this.collectionName = co;
273 }
274
275 public Double getFloorPrice() //gets floorPrice
276 {
277     return this.floorPrice;
278 }
279
280 public void setFloorPrice(Double f) //sets floorPrice amount
281 {
282     this.floorPrice = f;
283 }
284
285 public String getTokenType() //gets plate #
286 {
287     return this.tokenType;
288 }
289
290 public void setTokenType(String bc) //sets plate #
291 {
292     this.tokenType = bc;
293 }
294
295 public void setNFTAddress(String a)
296 {
297
298     nftAddress = a;
299 }
300
```

```

301 public String getNFTAddress()
302 {
303     return nftAddress;
304
305 }
306
307
308 public String toString() //not used; wrong result
309 {
310     return this.getClass().getName() + "WRONG " + " " + this.getUN() + "\n WRONG" +
311         this.getRarity() + " " + this.getCollection() +
312         "\n WRONG" + this.getTokenType() + "\n WRONG" + this.getFloorPrice() + "]";
313 }
314 }
315
316 //////////////////////////////////////
317 //////////////////////////////////////
318 /*****WalletAddress*****/
319 /*Student will only get the .class files for this */
320 //////////////////////////////////////
321 //////////////////////////////////////
322
323 class WalletAddress
324 {
325     private String publicKeyWallet;
326
327     public WalletAddress()
328     {
329         this.publicKeyWallet = "NA";
330     }
331
332     public WalletAddress(String a)
333     {
334         this.publicKeyWallet = a;
335
336     }
337     public String getAdd()
338     {
339         return this.publicKeyWallet;
340     }
341
342     public void setAdd(String s)
343     {
344         publicKeyWallet = s;

```

345

346 }

347

348 }