

```

1  package App;
2
3  public class AdminPanel {
4      String id, pass;
5      String c;
6
7      AdminPanel() {
8          // admin privilege starts here
9          System.out.println("\tLogin Successful");
10
11         // Use AppScanner instead of Scanner
12         try {
13             // 2ND PART
14             System.out.println("\t=====");
15             System.out.println("\t\tAdmin Panel");
16             System.out.println("\t=====");
17             System.out.println("\t'→Navigate by inserting index←'");
18             System.out.println("\t1. Change volume of parts");
19             System.out.println("\t2. Add new parts");
20             System.out.println("\t3. Go back to the menu");
21             System.out.println("\t=====");
22             System.out.print("\tInsert a number between 1 to 3: ");
23
24             int x;
25             while (true) {
26                 try {
27                     x = AppScanner.nextInt();
28                     break;
29                 } catch (Exception e) {
30                     System.out.println("Invalid input. Please enter a
31 number.");
32                 }
33
34                 while (x < 1 || x > 3) {
35                     System.out.println("\tInvalid range. \n\tPlease insert a
36 number between 1 to 3.");
37                     System.out.print("\tSelect range: ");
38
39                     while (true) {
40                         try {
41                             x = AppScanner.nextInt();
42                             break;
43                         } catch (Exception e) {
44                             System.out.println("Invalid input. Please enter a
45 number.");
46                         }
47                     }
48                     System.out.println("\t-----");
49
50                     switch (x) {
51                         case 1:

```

```

51         // looks for a part. if found, shows the current volume
    then changes it to the target;
52         Selector.printPartAll();
53         System.out.println("\tSelect a part ID from above to
change its volume.");
54         System.out.print("\tChosen index :");
55         int t;
56
57         while (true) {
58             try {
59                 t = AppScanner.nextInt();
60                 break;
61             } catch (Exception e) {
62                 System.out.println("Invalid input. Please enter
a number.");
63             }
64         }
65
66         System.out.print("\tInput the new volume : ");
67         int v;
68
69         while (true) {
70             try {
71                 v = AppScanner.nextInt();
72                 break;
73             } catch (Exception e) {
74                 System.out.println("Invalid input. Please enter
a number.");
75             }
76         }
77
78         CSVWriter.updateVolume(t, v);
79         new AdminPanel();
80         break;
81     case 2:
82         // looks for the highest ID, does +1, and adds parts
    based on type;
83         CSVWriter.addNewPart();
84         new AdminPanel();
85         break;
86     case 3:
87         // creates a new Menu instance
88         new Menu();
89         break;
90     default:
91         break;
92     }
93     } catch (Exception e) {
94         e.printStackTrace();
95     }
96 }
97 }
98

```

```

1  package App;
2
3  public class LocationFinder {
4      int n;
5
6      public LocationFinder() {
7          System.out.println("\t=====");
8          System.out.println("\t\tDealership Finder");
9          System.out.println("\t=====");
10         System.out.println("\t'→Navigate by inserting index←'");
11         System.out.println("\t1.IN Barishal");
12         System.out.println("\t2.IN Chittagong");
13         System.out.println("\t3.IN Dhaka");
14         System.out.println("\t4.IN Khulna");
15         System.out.println("\t5.IN Rajshahi");
16         System.out.println("\t6.IN Rangpur");
17         System.out.println("\t7.IN Sylhet");
18         System.out.println("\t=====");
19         System.out.print("\tInsert number between 1 to 5: ");
20         // Use AppScanner for user input
21         this.n = AppScanner.nextInt();
22         switch (n) {
23             case 1:
24
25                 System.out.println("\t=====");
26                 System.out.println("\t\tBarishal Dealership");
27
28                 System.out.println("\t=====");
29                 System.out.println("\tAddress : Fake address\n\tCall : Fake
30 number");
31
32                 System.out.println("\t=====");
33                 break;
34             case 2:
35
36                 System.out.println("\t=====");
37                 System.out.println("\t\tChittagong Dealership");
38
39                 System.out.println("\t=====");
40                 System.out.println("\tAddress : Fake address\n\tCall : Fake
41 number");
42
43                 System.out.println("\t=====");
44                 break;
45             case 3:
46
47                 System.out.println("\t=====");
48                 System.out.println("\t\tDhaka Dealership");
49
50                 System.out.println("\t=====");
51                 System.out.println("\tAddress : Fake address\n\tCall : Fake
52 number");
53
54                 System.out.println("\t=====");
55                 break;
56             case 4:
57
58                 System.out.println("\t=====");
59                 System.out.println("\t\tKhulna Dealership");
60
61                 System.out.println("\t=====");
62                 System.out.println("\tAddress : Fake address\n\tCall : Fake
63 number");
64
65                 System.out.println("\t=====");
66                 break;
67             case 5:
68
69                 System.out.println("\t=====");
70                 System.out.println("\t\tRajshahi Dealership");
71
72                 System.out.println("\t=====");
73                 System.out.println("\tAddress : Fake address\n\tCall : Fake
74 number");
75
76                 System.out.println("\t=====");
77                 break;
78             case 6:
79
80                 System.out.println("\t=====");
81                 System.out.println("\t\tRangpur Dealership");
82
83                 System.out.println("\t=====");
84                 System.out.println("\tAddress : Fake address\n\tCall : Fake
85 number");
86
87                 System.out.println("\t=====");
88                 break;
89             case 7:
90
91                 System.out.println("\t=====");
92                 System.out.println("\t\tSylhet Dealership");
93
94                 System.out.println("\t=====");
95                 System.out.println("\tAddress : Fake address\n\tCall : Fake
96 number");
97
98                 System.out.println("\t=====");
99                 break;
100            default:
101                System.out.println("\tInvalid input");
102            }
103        }
104    }

```

```

System.out.println("\t=====");
        break;
        case 4:

System.out.println("\t=====");
        System.out.println("\t\tKhulna Dealership");

System.out.println("\t=====");
        System.out.println("\tAddress : Fake address\n\tCall : Fake
number");

System.out.println("\t=====");
        break;
        case 5:

System.out.println("\t=====");
        System.out.println("\t\tRajshahi Dealership");

System.out.println("\t=====");
        System.out.println("\tAddress : Fake address\n\tCall : Fake
number");

System.out.println("\t=====");
        break;
        case 6:

System.out.println("\t=====");
        System.out.println("\t\tRangpur Dealership");

System.out.println("\t=====");
        System.out.println("\tAddress : Fake address\n\tCall : Fake
number");

System.out.println("\t=====");
        break;
        case 7:

System.out.println("\t=====");
        System.out.println("\t\tSylhet Dealership");

System.out.println("\t=====");
        System.out.println("\tAddress : Fake address\n\tCall : Fake
number");

System.out.println("\t=====");
        break;
        default:
            // dummy
            break;
    }
    System.out.println("\t—————→Going back to menu←————");
    new Menu();
}
}

```

```

1  package App;
2
3  import java.io.BufferedReader;
4  import java.io.FileReader;
5  import java.io.IOException;
6  import java.util.Set;
7  import java.util.HashSet;
8
9  public class Main{
10     public static void main(String[] args) {
11         new Menu();
12     }
13
14 }
15
16 public class Menu {
17     int x;
18
19     public Menu() {
20         System.out.println("\t=====");
21         System.out.println("\t\t\tMenu");
22         System.out.println("\t=====");
23         System.out.println("\t'→Navigate by inserting index←'");
24         System.out.println("\t1. Browse parts");
25         System.out.println("\t2. Find nearest dealership");
26         System.out.println("\t3. Contact");
27         System.out.println("\t4. Admin login");
28         System.out.println("\t=====");
29         System.out.print("\tInsert number between 1 to 4: ");
30
31         x = AppScanner.nextInt();
32
33         while (x < 1 || x > 4) {
34             System.out.println("\tInvalid range. \n\tPlease insert a
number between 1 to 4.");
35             System.out.print("\tSelect range: ");
36             x = AppScanner.nextInt();
37         }
38
39         System.out.println("\t-----");
40
41         switch (x) {
42             case 1:
43                 // browse parts
44
45                 System.out.println("\t=====");
46                 System.out.println("\t\tBrowsing Window");
47
48                 System.out.println("\t=====");
49                 System.out.println("\t'→Navigate by inserting
index←'");
50                 System.out.println("\t1. Browse all ");
51                 System.out.println("\t2. Browse by type ");

```

```

50         System.out.println("\t3. Browse by brand ");
51         System.out.println("\t4. Search by name");
52         System.out.println("\t5. Go back to menu");
53
54         System.out.println("\t=====");
55         System.out.print("\tInsert number between 1 to 5: ");
56
57         int x;
58
59         while (true) {
60             try {
61                 x = AppScanner.nextInt();
62                 AppScanner.nextLine(); // Consume the newline
63                 character
64                 break;
65             } catch (Exception e) {
66                 System.out.println("Invalid input. Please enter a
67                 number.");
68                 AppScanner.nextLine(); // Consume the invalid
69                 input
70             }
71         }
72
73         System.out.println("\t-----");
74
75         switch (x) {
76             case 4:
77                 // search using user input
78                 System.out.print("\tInput parts name(must be 1:1):
79                 ");
80                 String s = AppScanner.nextLine();
81
82                 System.out.println("\t-----");
83                 // Call the printPartName method from Selector
84                 class
85                 Selector.printPartName(s);
86                 break;
87             case 5:
88                 // creates new Menu instance
89                 new Menu();
90                 break;
91             default:
92                 // creates new constructor from Selector class
93                 // works when input 1 to 3
94                 new Selector(x);
95                 break;
96         }
97     }
98     break;
99 case 2:
100     new LocationFinder();
101
102     break;
103 case 3:
104     System.out.println("\tEmail : fake@fake.com");

```

```

97         System.out.println("\tCall : fake number");
98
99         System.out.println("\t-----");
100         new Menu();
101         break;
102     case 4:
103         // admin panel
104         new LoginVerification();
105         break;
106
107     default:
108         break;
109 }
110 }
111
112
113
114 public class Selector extends Parts {
115     int x;
116
117     public Selector(int x) {
118         super(0, "", "", "", 0, 0);
119         this.x = x;
120         switch (x) {
121             case 1:
122                 printPartAll();
123                 break;
124             case 2:
125                 printPartType();
126                 break;
127             case 3:
128                 printPartBrand();
129                 break;
130             default:
131                 break;
132         }
133         new CallBack();
134     }
135
136     public static void printPartAll() {
137         String csvFilePath = "Database/PartsData.csv";
138
139         try (BufferedReader br = new BufferedReader(new
140 FileReader(csvFilePath))) {
141             String line;
142             boolean isFirstLine = true;
143
144             while ((line = br.readLine()) != null) {
145                 if (isFirstLine) {
146                     isFirstLine = false;
147                     continue;
148                 }
149
150                 Parts part = Parts.createFromCSVLine(line);

```

```

150         printPart(part);
151     }
152     } catch (IOException e) {
153         e.printStackTrace();
154     }
155 }
156
157 private static void printPartAll(String s) {
158     String csvFilePath = "Database/PartsData.csv";
159     boolean found = false;
160
161     try (BufferedReader br = new BufferedReader(new
162     FileReader(csvFilePath))) {
163         String line;
164         boolean isFirstLine = true;
165
166         while ((line = br.readLine()) != null) {
167             if (isFirstLine) {
168                 isFirstLine = false;
169                 continue;
170             }
171
172             Parts part = Parts.createFromCSVLine(line);
173             if (part.type.equalsIgnoreCase(s) ||
174             part.brand.equalsIgnoreCase(s) || part.name.equalsIgnoreCase(s)) {
175                 printPart(part);
176                 found = true;
177             }
178         } catch (IOException e) {
179             e.printStackTrace();
180         }
181
182         if (!found) {
183             System.out.println("No parts found for the specified criteria:
184             " + s);
185         }
186     }
187
188     public void printPartType() {
189         // Use AppScanner instead of direct Scanner
190         System.out.println("Please select which type of product you want
191         to see by, inserting number(1-6)");
192         System.out.println("'→ 1 for Engine.");
193         System.out.println("'→ 2 for Wheels.");
194         System.out.println("'→ 3 for Turbo.");
195         System.out.println("'→ 4 for ECU.");
196         System.out.println("'→ 5 for Rear Wing.");
197         System.out.println("'→ 6 for Aero Kit.");
198         System.out.print("Select range: ");
199         int y = AppScanner.nextInt();
200
201         while (y < 1 || y > 6) {
202             System.out.println("Invalid range. Please insert a number
203             between 1 to 6: ");

```



```

200         System.out.print("Select range: ");
201         y = AppScanner.nextInt();
202     }
203
204     switch (y) {
205         case 1:
206             printPartAll("Engine");
207             break;
208         case 2:
209             printPartAll("Wheels");
210             break;
211         case 3:
212             printPartAll("Turbo");
213             break;
214         case 4:
215             printPartAll("ECU");
216             break;
217         case 5:
218             printPartAll("Rear Wing");
219             break;
220         case 6:
221             printPartAll("Aero Kit");
222             break;
223         default:
224             break;
225     }
226 }
227
228 private void printPartBrand() {
229     String csvFilePath = "Database/PartsData.csv";
230     Set<String> uniqueStrings =
UniqueStringGenerator.generateUniqueStrings(csvFilePath);
231
232     System.out.println("Unique Strings:");
233     int i = 1;
234     for (String str : uniqueStrings) {
235         System.out.println(i++ + " for " + str);
236     }
237
238     int y = AppScanner.nextInt();
239     while (y < 1 || y > (i - 1)) {
240         System.out.println("Invalid range. Please insert a number
between 1 to " + (i - 1) + "");
241         System.out.print("Select range: ");
242         y = AppScanner.nextInt();
243     }
244
245     i = 1;
246     for (String str : uniqueStrings) {
247         if (i == y) {
248             System.out.println("\t====You chose " + str + "====");
249             System.out.println("====The available parts from " + str
+ " are====");
250             System.out.println();
251             printPartAll(str);

```

```

252         } else {
253             // do nothing
254         }
255         i++;
256     }
257 }
258
259 public static void printPartName(String s) {
260     printPartAll(s);
261     new Callback();
262 }
263
264 private static void printPart(Parts part) {
265     PartPrinter.printPartInfo(part);
266 }
267 }
268
269
270 public class UniqueStringGenerator {
271     //checks the different brand name and act likes a dictionary that
272     //holds unique string
273     public static Set<String> generateUniqueStrings(String csvFilePath) {
274         Set<String> uniqueStrings = new HashSet<>();
275
276         try (BufferedReader br = new BufferedReader(new
277             FileReader(csvFilePath))) {
278             String line;
279             boolean isFirstLine = true;
280
281             while ((line = br.readLine()) != null) {
282                 if (isFirstLine) {
283                     isFirstLine = false;
284                     continue;
285                 }
286
287                 String[] data = line.split(",");
288                 // index 3 contains the string of interest
289                 String stringValue = data.length > 3 ? data[3].trim() :
290                 "";
291
292                 if (!stringValue.isEmpty()) {
293                     uniqueStrings.add(stringValue);
294                 }
295             } catch (IOException e) {
296                 e.printStackTrace();
297             }
298
299             return uniqueStrings;
300         }
301     }

```

```

1  package App;
2
3  public class Parts {
4      protected int id;
5      protected String name;
6      protected String type;
7      protected String brand;
8      protected int price;
9      protected int volume;
10
11     // Constructor
12     public Parts(int id, String name, String type, String brand, int
price, int volume) {
13         this.id = id;
14         this.name = name;
15         this.type = type;
16         this.brand = brand;
17         this.price = price;
18         this.volume = volume;
19     }
20
21     // Factory method to create Parts object from CSV line
22     public static Parts createFromCSVLine(String line) {
23         String[] data = line.split(",");
24         int id = Integer.parseInt(data[0].trim());
25         String name = data[1].trim();
26         String type = data[2].trim();
27         String brand = data[3].trim();
28         int price = Integer.parseInt(data[4].trim());
29         int volume = Integer.parseInt(data[5].trim());
30
31         if (type.equalsIgnoreCase("Engine")) {
32             // Parse additional data for Engine
33             int horsepower = Integer.parseInt(data[6].trim());
34             return new Engine(id, name, type, brand, price, volume,
horsepower);
35         }
36         else if(type.equalsIgnoreCase("Wheels")){
37             int diameter = Integer.parseInt(data[6].trim());
38             return new Wheels(id, name, type, brand, price, volume,
diameter);
39         }
40         else if(type.equalsIgnoreCase("Turbo")){
41             String boost = (data[6].trim());
42             return new Turbo(id, name, type, brand, price, volume, boost);
43         }
44         else if(type.equalsIgnoreCase("Ecu")){
45             return new Ecu(id, name, type, brand, price, volume);
46         }
47         else if(type.equalsIgnoreCase("Rear wing")){
48             String material = (data[6].trim());
49             return new RearWing(id, name, type, brand, price, volume,
material);

```

```

50     }
51     else if(type.equalsIgnoreCase("Aero Kit")){
52         String color = (data[6].trim());
53         return new AeroKit(id, name, type, brand, price, volume,
color);
54     }
55     else {
56         // For other types, create a regular Parts object
57         return new Parts(id, name, type, brand, price, volume);
58     }
59 }
60 }
61
62 public class PartPrinter {
63
64     public static void printPartInfo(Parts part) {
65         System.out.println("      +----->");
66         System.out.println("      |      " +
part.getClass().getSimpleName() );
67         System.out.println("      +----->");
68         System.out.println("      +----->");
69         System.out.println("      | - ID: " + part.id);
70         System.out.println("      | - Name: " + part.name);
71         System.out.println("      | - Type: " + part.type);
72         System.out.println("      | - Brand: " + part.brand);
73         System.out.println("      | - Price: " + part.price+"$");
74
75         if (part instanceof Engine) {
76             System.out.println("      | - Horse Power: " + ((Engine)
part).getHorsepower()+" BHP");
77         }
78         else if(part instanceof Wheels){
79             System.out.println("      | - Wheel Size: " + ((Wheels)
part).getDiameter()+" inches");
80         }
81         else if(part instanceof Turbo){
82             System.out.println("      | - Boost max: " + ((Turbo)
part).getBoost()+" psi");
83         }
84         else if(part instanceof RearWing){
85             System.out.println("      | - Wing material: " + ((RearWing)
part).getMaterial());
86         }
87         else if(part instanceof AeroKit){
88             System.out.println("      | - Color available: " +
((AeroKit) part).getColor());
89         }
90         System.out.println("      | - Available parts: " + part.volume);
91         System.out.println("      +----->");
92     }
93 }
94
95 package App;
96
97 public class CallBack {

```

```

98     public Callback() {
99         try {
100             System.out.println("\tInsert 1 to find the Nearest
dealer\n\t\t2 to go back to menu");
101             System.out.print("\tInsert here: ");
102             int n;
103             while (true) {
104                 try {
105                     n = AppScanner.nextInt();
106                     break;
107                 } catch (Exception e) {
108                     System.out.println("Invalid input. Please enter a
number.");
109                 }
110             }
111
112             if (n == 1) {
113                 new LocationFinder();
114             } else if (n == 2) {
115                 new Menu();
116             } else {
117                 while (n != 1 && n != 2) {
118                     System.out.println("\tInvalid range. \n\tPlease insert
either 1 or 2.");
119                     System.out.print("\tSelect range: ");
120
121                     while (true) {
122                         try {
123                             n = AppScanner.nextInt();
124                             break;
125                         } catch (Exception e) {
126                             System.out.println("Invalid input. Please
enter a number.");
127                         }
128                     }
129                 }
130             }
131         } catch (Exception e) {
132             e.printStackTrace();
133         }
134     }
135 }
136
137 //java file for writing on CSV logic
138 package App;
139
140 import java.io.BufferedReader;
141 import java.io.BufferedWriter;
142 import java.io.FileReader;
143 import java.io.FileWriter;
144 import java.io.IOException;
145 import java.io.PrintWriter;
146 import java.nio.file.Files;
147 import java.nio.file.Paths;
148 import java.util.List;

```

```

149
150 public class CSVWriter {
151     private static final String CSV_FILE_PATH = "Database/PartsData.csv";
152
153     // Method to update the volume of a part in the CSV file
154     public static void updateVolume(int partId, int newVolume) {
155         try {
156             List<String> lines =
157 Files.readAllLines(Paths.get(CSV_FILE_PATH));
158
159             // Adjust the part index based on the starting ID (1000 in
160 your case)
161             int partIndex = partId - 999;
162
163             if (partIndex ≥ 0 && partIndex < lines.size()) {
164                 String[] parts = lines.get(partIndex).split(",");
165                 parts[5] = String.valueOf(newVolume);
166                 lines.set(partIndex, String.join(",", parts));
167
168                 Files.write(Paths.get(CSV_FILE_PATH), lines);
169                 System.out.println("Volume updated successfully!");
170             } else {
171                 System.out.println("Invalid part ID.");
172             }
173         } catch (IOException e) {
174             e.printStackTrace();
175         }
176
177     public static void addNewPart() {
178         // Read existing lines to find the highest index
179         int highestIndex = findHighestIndex();
180         int newIndex = highestIndex + 1;
181
182         System.out.println("Adding a new part with index: " + newIndex);
183
184         // Gather information from the user
185         System.out.print("Enter part name: ");
186         String name = AppScanner.nextLine();
187
188         System.out.print("Enter part type: ");
189         String type = AppScanner.nextLine();
190
191         System.out.print("Enter part brand: ");
192         String brand = AppScanner.nextLine();
193
194         System.out.print("Enter part price: ");
195         int price = AppScanner.nextInt();
196
197         System.out.print("Enter part volume: ");
198         int volume = AppScanner.nextInt();
199
200         String additionalInfo = "";
201
202         // Based on part type, ask for additional information

```

```

202         switch (type.toLowerCase()) {
203             case "engine":
204                 System.out.print("Enter horsepower: ");
205                 additionalInfo = AppScanner.nextLine();
206                 break;
207             case "wheels":
208                 System.out.print("Enter diameter: ");
209                 additionalInfo = AppScanner.nextLine();
210                 break;
211             case "turbo":
212                 System.out.print("Enter boost: ");
213                 additionalInfo = AppScanner.nextLine();
214                 break;
215             case "ecu":
216                 // No additional information for ECU
217                 break;
218             case "rear wing":
219                 System.out.print("Enter material: ");
220                 additionalInfo = AppScanner.nextLine();
221                 break;
222             case "aero kit":
223                 System.out.print("Enter color: ");
224                 additionalInfo = AppScanner.nextLine();
225                 break;
226             default:
227                 System.out.println("Invalid part type.");
228                 return;
229         }
230
231         // Construct the new part line
232         String newPartLine = newIndex + "," + name + "," + type + "," +
brand + "," + price + "," + volume + "," + additionalInfo;
233
234         // Append the new line to the CSV file
235         try (PrintWriter writer = new PrintWriter(new BufferedWriter(new
FileWriter(CSV_FILE_PATH, true)))) {
236             writer.println(newPartLine);
237             System.out.println("New part added successfully.");
238
239         } catch (IOException e) {
240             System.out.println("Error writing to the CSV file.");
241             e.printStackTrace();
242         }
243     }
244
245     private static int findHighestIndex() {
246         int highestIndex = 0;
247
248         try (BufferedReader br = new BufferedReader(new
FileReader(CSV_FILE_PATH))) {
249             String line;
250             boolean isFirstLine = true;
251
252             while ((line = br.readLine()) != null) {
253                 if (isFirstLine) {

```

```

254         isFirstLine = false;
255         continue;
256     }
257
258     String[] parts = line.split(",");
259     int currentId = Integer.parseInt(parts[0]);
260     highestIndex = Math.max(highestIndex, currentId);
261 }
262 } catch (IOException e) {
263     e.printStackTrace();
264 }
265
266     return highestIndex;
267 }
268 }
269
270 package App;
271
272 public class LoginVerification {
273
274     public LoginVerification() {
275         boolean loginSuccessful = false;
276
277         do {
278             System.out.print("\tPlease input User Id: ");
279             String id = AppScanner.nextLine();
280             System.out.print("\tPlease input Password: ");
281             String pass = AppScanner.nextLine();
282
283             // Default login admin admin
284             if (id.equals("admin") && pass.equals("admin")) {
285                 // Login successful
286                 loginSuccessful = true;
287                 new AdminPanel();
288             } else {
289                 System.out.println("\tPress R to retry or M to go back to
menu :");
290                 String c = AppScanner.nextLine();
291
292                 if (c.equalsIgnoreCase("M")) {
293                     new Menu();
294                     break;
295                 }
296             }
297         } while (!loginSuccessful);
298     }
299 }
300
301

```



```
1 package App;
2
3 import java.util.Scanner;
4
5 public class AppScanner {
6     private static final Scanner scanner = new Scanner(System.in);
7
8     public static String nextLine() {
9         return scanner.nextLine();
10    }
11
12    public static int nextInt() {
13        while (true) {
14            try {
15                return Integer.parseInt(scanner.nextLine());
16            } catch (NumberFormatException e) {
17                // Do nothing or handle the exception as needed
18            }
19        }
20    }
21 }
22
23 public class AeroKit extends Parts {
24     private String color; // Additional attribute for Aero Kit
25
26     public AeroKit(int id, String name, String type, String brand, int
price, int volume, String color) {
27         super(id, name, type, brand, price, volume);
28         this.color = color;
29     }
30
31     public String getColor() {
32         return color;
33     }
34 }
35
36 public class Ecu extends Parts {
37     // No additional attributes specific to ECU
38
39     public Ecu(int id, String name, String type, String brand, int price,
int volume) {
40         super(id, name, type, brand, price, volume);
41     }
42 }
43
44 public class Engine extends Parts {
45     private int horsepower;
46
47     public Engine(int id, String name, String type, String brand, int
price, int volume, int horsepower) {
48         super(id, name, type, brand, price, volume);
49         this.horsepower = horsepower;
50     }
51 }
```

```
51
52     public int getHorsepower() {
53         return horsepower;
54     }
55 }
56
57 public class RearWing extends Parts {
58     private String material; // Additional attribute for Rear Wing
59
60     public RearWing(int id, String name, String type, String brand, int
price, int volume, String material) {
61         super(id, name, type, brand, price, volume);
62         this.material = material;
63     }
64
65     public String getMaterial() {
66         return material;
67     }
68 }
69
70 public class Turbo extends Parts {
71     private String boost; // Additional attribute for Turbo
72
73     public Turbo(int id, String name, String type, String brand, int price,
int volume, String boost) {
74         super(id, name, type, brand, price, volume);
75         this.boost = boost;
76     }
77
78     public String getBoost() {
79         return boost;
80     }
81 }
82
83 public class Turbo extends Parts {
84     private String boost; // Additional attribute for Turbo
85
86     public Turbo(int id, String name, String type, String brand, int price,
int volume, String boost) {
87         super(id, name, type, brand, price, volume);
88         this.boost = boost;
89     }
90
91     public String getBoost() {
92         return boost;
93     }
94 }
95
96
```