

```

1 package org.bwagner;
2
3 import java.util.*;
4
5 /*
6     This class is the program's user interface. It is responsible for interacting
7     with the user through a menu system. It contains the program's main method.
8 */
9
10 public class WeightTraining
11 {
12     //instance variables
13     private Scanner keyboard;
14
15     // constructor
16     public WeightTraining()
17     {
18         keyboard = new Scanner(System.in);
19
20         mainMenu();
21     }
22
23     /*
24         This is the main menu for the program. All interaction with the user
25         originates from this menu.
26     */
27     public void mainMenu()
28     {
29         int ans = 0;
30
31         do
32         {
33             System.out.println();
34             System.out.println("=====");
35             System.out.println("        Main Menu        ");
36             System.out.println("=====");
37             System.out.println("  1. Add Player");
38             System.out.println("  2. Update Player Maxes");
39             System.out.println("  3. View List of Player Names");
40             System.out.println("  4. View a Player's Maxes");
41             System.out.println("  5. Delete Players");
42             System.out.println("  6. Print");
43             System.out.println("  7. Save");
44             System.out.println("  8. Exit");
45
46             ans = validateIntegerInput("Selection -->");
47             System.out.println();
48             if(ans == 1)
49                 addPlayer();
50             if(ans == 2)
51                 updatePlayers();
52             if(ans == 3)

```

```

53         viewAllPlayers();
54         if(ans == 4)
55             searchForPlayer();
56         if(ans == 5)
57             delete();
58         if(ans == 6)
59             print();
60         if(ans == 7)
61             saveDataFile();
62     }
63     while(ans != 8);
64
65     System.out.println();
66     System.out.println("Good Bye!");
67     System.out.println();
68     System.exit(0);        // close terminal window
69 }
70
71 /*
72     This method allows the user to enter an integer value. It then verifies
73     that the input value is an integer. If it is not an integer the method
74     prompts the user to re-enter the value again.
75     @return the input value
76     @param prompt the input prompt
77 */
78 public int validateIntegerInput(String prompt)
79 {
80     int ans = 0;
81     boolean flag;
82
83     do
84     {
85         flag = true;
86         System.out.print(prompt); // display input prompt
87         try
88         {
89             ans = keyboard.nextInt();
90
91         }
92         catch (Exception e)
93         {
94             System.out.println("Invalid Entry. Try again.");
95             flag = false;
96         }
97         keyboard.nextLine(); // clear buffer
98     }
99     while(flag == false);
100
101     return ans;
102 }
103
104 /*

```

```

105 This method validates that the parameter week is between
106 1 <= week <= 10. If it is not it requires the user to enter
107 a valid number.
108 @param the week value(1-10)
109 */
110 public void validateWeekNum(int week)
111 {
112     while(week < 1 || week > 10)
113     {
114         week = validateIntegerInput("Enter Program Week (1, 10) -->");
115     }
116 }
117
118 /*
119 This method prompts the user to enter a player's info and then adds
120 the player to the database.
121 */
122 public void addPlayer()
123 {
124     String ans = "";
125     do
126     {
127         System.out.println("=====");
128         System.out.println("    Add Player");
129         System.out.println("=====");
130         System.out.print("Enter Player Name (lastname, firstname)-->");
131         String name = keyboard.nextLine();
132
133         int bench = validateIntegerInput("Enter Bench Max -->");
134         int squat = validateIntegerInput("Enter Squat Max -->");
135         int incline = validateIntegerInput("Enter Incline Max -->");
136         int power = validateIntegerInput("Enter Power Clean Max -->");
137
138         System.out.println();
139         System.out.print("Add another player[Y/N]?");
140
141         ans = keyboard.nextLine();
142     }
143     while(ans.equalsIgnoreCase("y"));
144 }
145
146 /*
147 This method allows a user to modify all players or single player
148 max values.
149 */
150
151 public void updatePlayers()
152 {
153     System.out.println("=====");
154     System.out.println("    Update Players Max");
155     System.out.println("=====");
156     System.out.println("1. Update a Player's Max");

```

```

157 System.out.println("2. Update All Players' Max");
158 int ans = validateIntegerInput("Selection -->");
159
160 if(ans == 1)
161 {
162     String response = "";
163     do
164     {
165         System.out.println();
166         System.out.print("Enter Player Name (lastname, firstname)-->");
167         String name = keyboard.nextLine();
168
169         System.out.println();
170
171         int bench = validateIntegerInput("Enter new Bench Max -->");
172         int squat = validateIntegerInput("Enter new Squat Max -->");
173         int incline = validateIntegerInput("Enter new Incline Max -->");
174         int power = validateIntegerInput("Enter new Power Clean Max -->");
175
176         System.out.println();
177         System.out.print("Update Another Player[Y/N]-->");
178         response = keyboard.nextLine();
179
180     }
181     while(response.equalsIgnoreCase("y"));
182 }
183 if(ans == 2)
184 {
185     System.out.println("=====");
186     System.out.println("    Update Players Max");
187     System.out.println("=====");
188 }
189 }
190
191 /* This method allows the user to remove a player from the database or
192    clear the database of all players.
193 */
194 public void delete()
195 {
196     System.out.println("=====");
197     System.out.println("    Delete Player");
198     System.out.println("=====");
199     System.out.println("    1. Delete a Player");
200     System.out.println("    2. Clear Database");
201     int ans = validateIntegerInput("Selection -->");
202 }
203
204 /*
205    This method displays a list in alphabetical of all players in the
206    database. It displays each player's name and classification.
207 */
208 public void viewAllPlayers()

```

```

209 {
210     System.out.println("=====");
211     System.out.println("    View All Players");
212     System.out.println("=====");
213 }
214
215 /* This method searches the database by player name. If the player is found
216    it displays the Player's exercise maxes.
217 */
218 public void searchForPlayer()
219 {
220     System.out.println("=====");
221     System.out.println("    Search For Player");
222     System.out.println("=====");
223     System.out.print("Enter Player Name (lastname, firstname)-->");
224     String name = keyboard.nextLine();
225 }
226
227 /* This method allows the user to print two documents.
228    1. A player or players workout program.
229    2. A list of players organized in groups of four by
230       their bench max.
231 */
232 public void print()
233 {
234     System.out.println("=====");
235     System.out.println("        Print");
236     System.out.println("=====");
237 }
238
239 /*
240    This method saves the databasse to the data file.
241 */
242 public void saveDataFile()
243 {
244     System.out.println("=====");
245     System.out.println("    Save Data File");
246     System.out.println("=====");
247 }
248
249 /*
250    This is the program's main menu.
251 */
252 public static void main(String[] args)
253 {
254     WeightTraining app = new WeightTraining();
255 }
256 }

```