This is what I have so far for two programs. I had a fairly easy time with program 1 and program 7, but had a really hard time with challenge 11

Programming challenge #1 - Roman Numerals

```java
import java.awt.Font;
import java.util.Scanner;
/**
 * Description
 * ----------------------
 *  Programming Challenge #1 public static void romanNumeral()
 *  Write a program that prompts the user to enter a number within the range of 1
 *  through 10. The program should display the Roman numeral version of that
 *  number. If the number is outside the range of 1 through 10, the program
 *  should display an error message.
 *
 *  Programming Challenge #7 public static void sortedNames()
 *  Write a program that asks the user to enter three names, and then displays the
 *  names sorted in ascending order. For example, if the user entered "Charlie",
 *  "Leslie", and "Andy", the program would display:
 *  Andy
 *  Charlie
 *  Leslie
 *
 * Programming Challenge #11 public static void runningTheRace()
 * Write a program that asks for the names of three runners and the time, in minutes,
 * it took each of them to finish a race. The program should display the names of the
 * runners in the order that they finished.
 *
 * ----------------------
 * @author Vincent Nguyen
 * @version 09/14/2024
 */
public class HW3VincentNguyen
{
    public static void main(String[] args)
    {
    }
```

I initially thought of doing if else statements but since roman numerals were going to go off of numbers I thought it would be easier and cleaner with a case switch method from the book.

```
37      public static void romanNumeral()
38      {
39          Scanner input = new Scanner(System.in);
40          System.out.println("Enter a number within the range of 1 and 10:");
41          int num = input.nextInt();
42
43          String romanNumeral = "";
44
45          // Switch statement to determine which number is inputted
46          switch(num)
47          {
48              case 1:
49                  romanNumeral = "I";
50                  break;
51              case 2:
52                  romanNumeral = ("II");
53                  break;
54              case 3:
55                  romanNumeral = ("III");
56                  break;
57              case 4:
58                  romanNumeral = ("IV");
59                  break;
60              case 5:
61                  romanNumeral = ("V");
62                  break;
63              case 6:
64                  romanNumeral = ("VI");
65                  break;
66              case 7:
67                  romanNumeral = ("VII");
68                  break;
69              case 8:
70                  romanNumeral = ("VIII");
71                  break;
72              case 9:
73                  romanNumeral = ("IX");
74                  break;
75              case 10:
76                  romanNumeral = ("X");
77                  break;
78              default:
79                  System.out.println("Error: Invalid month");
80                  break;
81
82          }
```

```
83
84          // Setup graphics screen
85          Draw scr = new Draw();
86          scr.clear(Draw.LIGHT_GRAY);
87          scr.setXscale(0, 400);
88          scr.setYscale(400, 0);
89          scr.setTitle("Roman Numerals by Vincent Nguyen");
90
91          // Shapes for fun
92          scr.setPenColor(Draw.RED);
93          scr.filledSquare(100, 300, 100);
94          scr.filledSquare(300, 100, 100);
95
96          scr.setPenColor(Draw.MAGENTA);
97          scr.filledRectangle(100, 100, 50, 100); // (x, y, width, height);
98          scr.filledRectangle(100, 100, 100, 50);
99
100         scr.setPenColor(Draw.WHITE);
101         scr.filledEllipse(100, 300, 30, 100);   // (x, y, horizontal radius, vertical radius)
102         scr.filledEllipse(100, 300, 100, 30);   // (hori radius (half of ellipse's width)
103                                                 // (vert radius (half of ellipse's height)
104         // Font and Color for Roman Numeral
105         Font romanNumFnt = new Font("Helvetica", Font.BOLD, 100);
106         scr.setFont(romanNumFnt);
107         scr.setPenColor(Draw.MAGENTA);
108         scr.text(300, 300, "" + romanNumeral);
109
110         // Font and Color for Name
111         Font nameFnt = new Font("TimesNewRoman", Font.BOLD, 60);
112         scr.setFont(nameFnt);
113         scr.setPenColor(Draw.YELLOW);
114         String name = "Vincent Nguyen";
115         scr.text( 200, 100, "" + name);
116
117
118     }
```

Graphics Screen for Roman numerals

```
Enter a number within the range of 1 and 10:
4

Can only enter input while your program is ru
```

Programming challenge #7 - Sorted Names

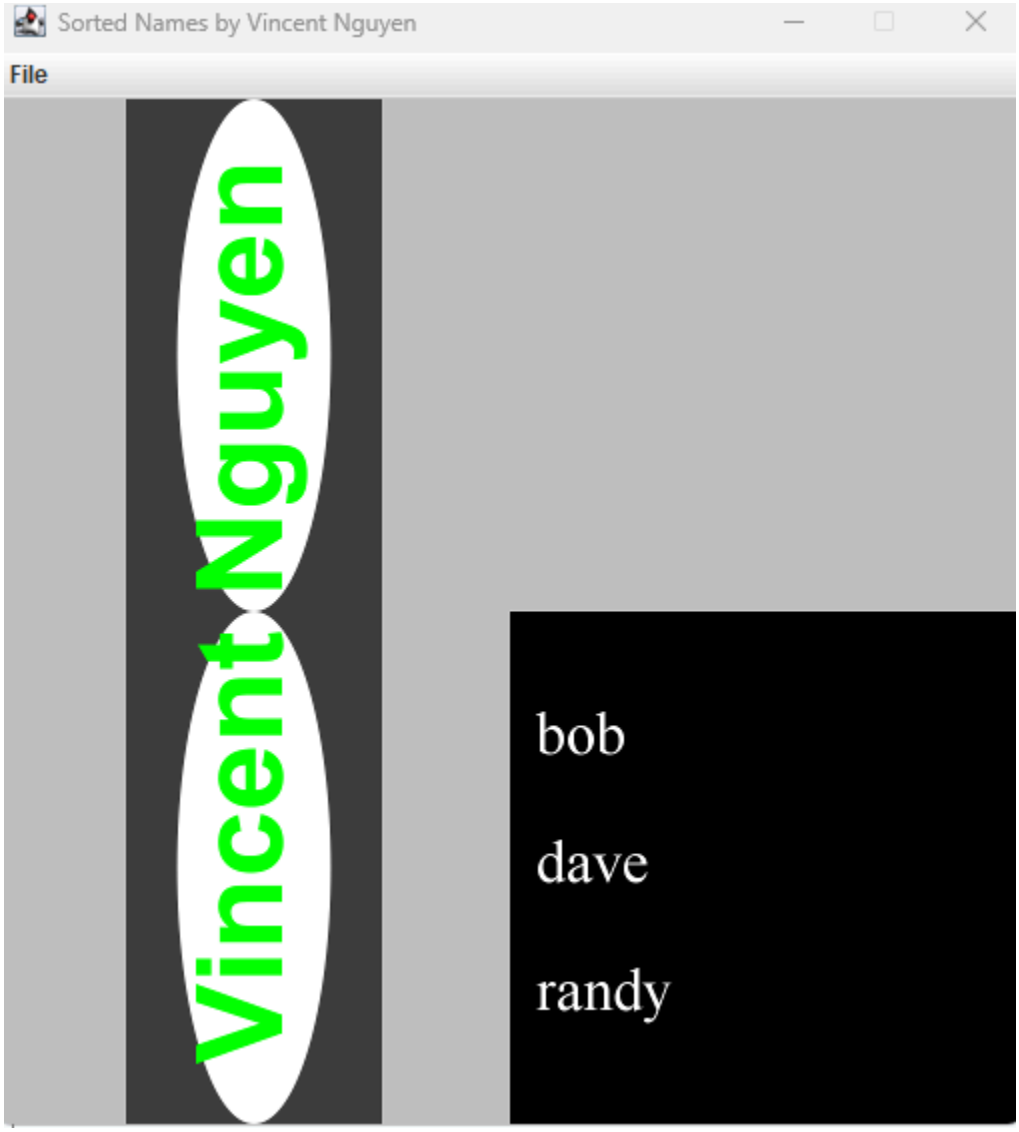
First by comparing if one variable is greater than the other two
Third by comparing the same variable is the lowest otherwise
Second place because its neither the smallest no the largest
Sorted Names
I used the .compareTo methodS
Code

```
120     public static void sortedNames()
121     {
122         // Get input from user
123         Scanner input = new Scanner(System.in);
124         System.out.println("Enter three names");
125         String name1 = input.nextLine(); // .nextline for string
126         String name2 = input.nextLine(); // .nextDouble for double
127         String name3 = input.nextLine(); // .nextFloat for float
128 
129         String first = "";
130         String second = "";
131         String third = "";
132 
133         // Checks if name1 is greater than name2 and name3
134         // If name1 is greater than name2 and name3 assign it third as highest value
135         // Since the desire is for ascending order we want the third value highest to
136         // lowest
137         if(name1.compareTo(name2) > 0 && name1.compareTo(name3) > 0)
138         {
139             third = name1;
140         }
141 
142         // if name1 is the lower than name2 and name3 assign it first as lowest value
143         else if(name1.compareTo(name2) < 0 && name1.compareTo(name3) < 0)
144         {
145             first = name1;
146         }
147         // if the first two conditions are not met then that means name1 is neither
148         // greater than both or lower than both
149         else
150         {
151             second = name1;
152         }
153 
154 
155         // Checks name2
156         // if name2 is greater than name1 and name3, assign to third
157         if(name2.compareTo(name1) > 0 && name2.compareTo(name3) > 0)
158         {
159             third = name2;
160         }
161 
162         else if(name2.compareTo(name1) < 0 && name2.compareTo(name3) < 0)
163         {
164             first = name2;
165         }
166         else
167         {
168             second = name2;
169         }
```

```
170
171         // Checks name3
172         // if name3 is greater than name1 and name 2, assign it as third
173         if(name3.compareTo(name1) > 0 && name3.compareTo(name2) > 0)
174         {
175             third = name3;
176         }
177         else if(name3.compareTo(name1) < 0 && name3.compareTo(name2) < 0)
178         {
179             first = name3;
180         }
181         else
182         {
183             second = name3;
184         }
185
186         // Graphics screen setup
187         Draw scr = new Draw();
188         scr.clear(Draw.LIGHT_GRAY);
189         scr.setXscale(0, 400);
190         scr.setYscale(400, 0);
191         scr.setTitle("Sorted Names by Vincent Nguyen");
192
193         // Shapes for fun
194         scr.setPenColor(Draw.BLACK);
195         scr.filledSquare(300, 300, 100);
196
197         scr.setPenColor(Draw.DARK_GRAY);
198         scr.filledRectangle(100, 100, 50, 100); // (x, y, width, height);
199         scr.filledRectangle(100, 300, 50, 100);
200
201         scr.setPenColor(Draw.WHITE);
202         scr.filledEllipse(100, 300, 30, 100);   // (x, y, horizontal radius, vertical radius)
203         scr.filledEllipse(100, 100, 30, 100);   // (hori radius (half of ellipse's width)
204                                                 // (vert radius (half of ellipse's height)
205         // Font and Color for names
206         Font romanNumFnt = new Font("Serif", Font.PLAIN, 30);
207         scr.setFont(romanNumFnt);
208         scr.setPenColor(Draw.WHITE);
209
210         int xPos = 210, yPos = 250;
211         scr.textLeft(xPos, yPos, "" + first);
212         scr.textLeft(xPos, yPos + 50, "" + second);
213         scr.textLeft(xPos, yPos + 100, "" + third);
214
215         // Font and Color for Name
216         Font nameFnt = new Font("TimesNewRoman", Font.BOLD, 60);
217         scr.setFont(nameFnt);
218         scr.setPenColor(Draw.GREEN);
219         String name = "Vincent Nguyen";
220         scr.text(100, 200, "" + name, 90);
221
222
223     }
```

Class compiled - no syntax errors

Graphics output for sorted names



Programming Challenge #11 - Running the Race

Running the names was the program I had the most trouble on. My first issue was the input

```
227
228         Scanner input = new Scanner(System.in);
229         System.out.println("\nEnter the name of 1st runner: ");
230         String runner1 = input.nextLine();
231         System.out.println("Enter their time (minutes): ");
232         double runTime1 = input.nextDouble();
233
234
235         System.out.println("\nEnter the name of 2nd runner: ");
236         String runner2 = input.nextLine();
237         System.out.println("Enter their time (minutes): ");
238         double runTime2 = input.nextDouble();
```

My input would be able to take the first name and their time, but "skipped" the second input and went to the third. This confused me for a while until I asked for some help.
I realized that line 232 which asks for a double input leaves a white space after hitting the enter key to input the double. This makes line 236 input read the whitespace as a name and "skips" that part of the code.

I had to add the code on line 233 to consume the whitespace that allows for the next input. (input.nextline();)

```
228         Scanner input = new Scanner(System.in);
229         System.out.println("\nEnter the name of 1st runner: ");
230         String runner1 = input.nextLine();
231         System.out.println("Enter their time (minutes): ");
232         double runTime1 = input.nextDouble();
233         input.nextLine(); // Consumes the space after the enter k
234                           // is hit
235
```

My first problem was the logic for determining who was first. I based my logic off of the second program

I changed this slightly keeping my first place comparison the same (If runTime1 was greater than the other 2)

Then realized I could compare the next variable (runTime2) with the another variable (runTime3) and if it was lower then it would get second place.

Otherwise that would leave the only option of runTime3 being larger and in second place instead. I applied this for the 3 different variables

```
257         if(runTime1 < runTime2 && runTime1 < runTime3)
258         {
259             first = runner1;
260             time1 = runTime1;
261             if (runTime2 < runTime3) // checks if the second runner
262             // is lower than third time if so make 2nd place the second runner
263             {
264                 second = runner2;
265                 time2 = runTime2;
266                 third = runner3;
267                 time3 = runTime3;
268             }
269             //
270             else
271             {
272                 second = runner3;
273                 time2 = runTime3;
274                 third = runner2;
275                 time3 = runTime2;
276             }
277         }S
```

I tried cleaning my code up a bit after

I compacted this code just like double

```
String first = "";
String Ssecond = "";
String third  = "";
String tied = "Tied!";

double time1, time2, time3;
```

->

```
249         String first = "", second = "", third  = "";
250         String tied = ""; // place holder
251
252         double time1, time2, time3;
```

After testing my code I realized that I had made two values even and decided that I should add an if statement that checks if they are tied first

Code

```
225     public static void runningTheRace()
226     {
227
228         Scanner input = new Scanner(System.in);
229         System.out.println("\nEnter the name of 1st runner: ");
230         String runner1 = input.nextLine();
231         System.out.println("Enter their time (minutes): ");
232         double runTime1 = input.nextDouble();
233         input.nextLine(); // Consumes the space after the enter key
234                           // is hit
235
236         System.out.println("\nEnter the name of 2nd runner: ");
237         String runner2 = input.nextLine();
238         System.out.println("Enter their time (minutes): ");
239         double runTime2 = input.nextDouble();
240         input.nextLine(); // Consume the space after the enter key
241                           // is hit
242
243         System.out.println("\nEnter the name of 3rd runner: ");
244         String runner3 = input.nextLine();
245         System.out.println("Enter their time (minutes): ");
246         double runTime3 = input.nextDouble();
247         input.nextLine(); // Consumes the whitespace
248
```

Class compiled - no syntax errors

```
247         input.nextLine(); // Consumes the whitespace
248
249         String first = "", second = "", third  = "";
250         String tiedForFirst = "", tiedForSecond = "";
251         String tied = ""; // place holder
252
253         double time1 = 0, time2 = 0, time3 = 0;
254
255         if (runTime1 == runTime2 && runTime1 == runTime3)
256         {
257             tied = "All runners are tied for first place!";
258         }
259         else if (runTime1 == runTime2)
260         {
261             tiedForFirst = runner1 + " and " + runner2 + " are tied for 1st Place";
262             first = runner3;
263             time1 = runTime3;
264         }
265         else if (runTime1 == runTime3)
266         {
267             tiedForFirst = runner1 + " and " + runner3 + " are tied for 1st place";
268             first = runner2;
269             time1  = runTime2;
270         }
```

```
271         else if (runTime2 == runTime3)
272         {
273            tiedForSecond = runner2 + " and " + runner3 + " are tied for 2nd place";
274            first = runner1;
275            time1 = runTime1;
276         }
277
278         if(tied.isEmpty())
279             {
280
281             // Checks if runner 1 is first, third, or second
282             // First if state checks if runner1 is the lowest time
283             if(runTime1 < runTime2 && runTime1 < runTime3)
284             {
285                 first = runner1;
286                 time1 = runTime1;
287                 if (runTime2 < runTime3) // checks if the second runner
288                 // is lower than third time if so make 2nd place the second runner
289                 {
290                     second = runner2;
291                     time2 = runTime2;
292                     third = runner3;
293                     time3 = runTime3;
294                 }
```

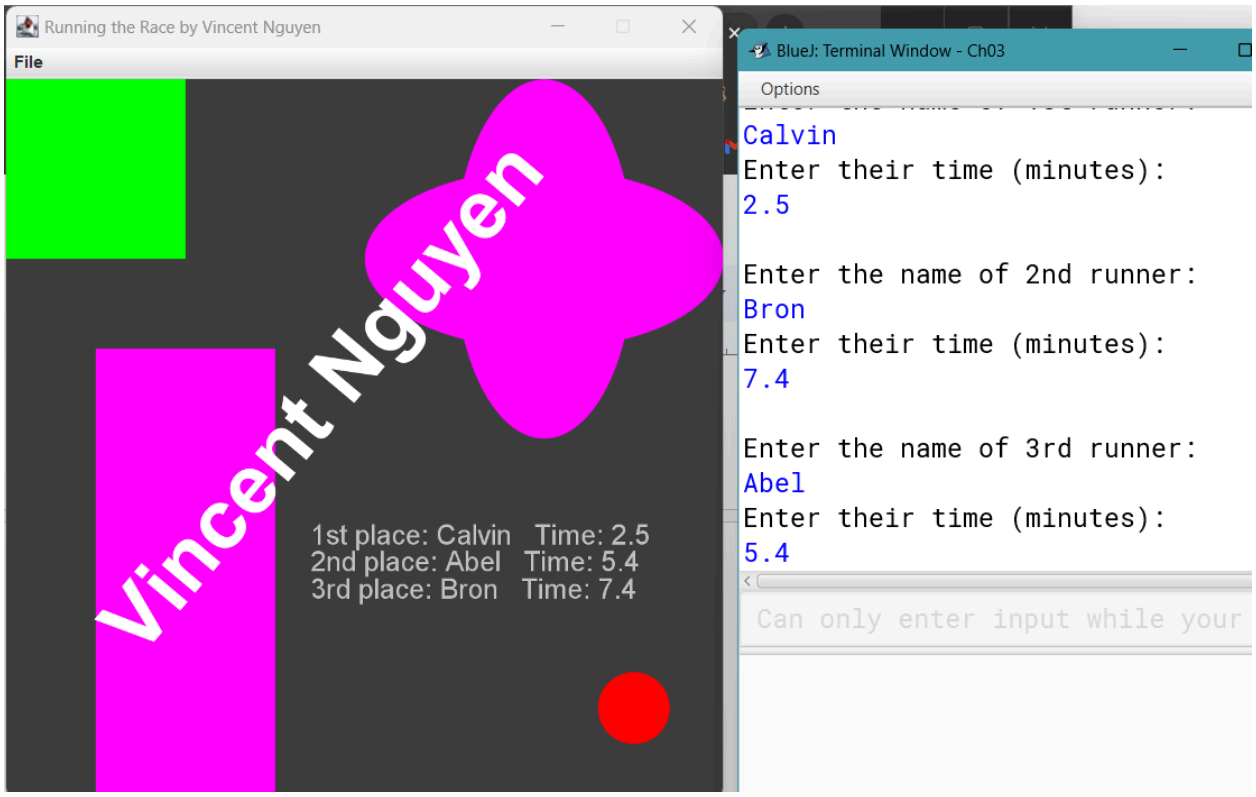
```
296                     else
297                     {
298                         second = runner3;
299                         time2 = runTime3;
300                         third = runner2;
301                         time3 = runTime2;
302                     }
303                 }
304                 else if(runTime2 < runTime1 && runTime2 < runTime3)
305                 {
306                     first = runner2;
307                     time1 = runTime2;
308                     if(runTime1 < runTime3)
309                     {
310                         second = runner1;
311                         time2 = runTime1;
312                         third = runner3;
313                         time3 = runTime3;
314                     }
315                     else
316                     {
317                         second = runner3;
318                         time2 = runTime3;
319                         third = runner1;
320                         time3 = runTime1;
```

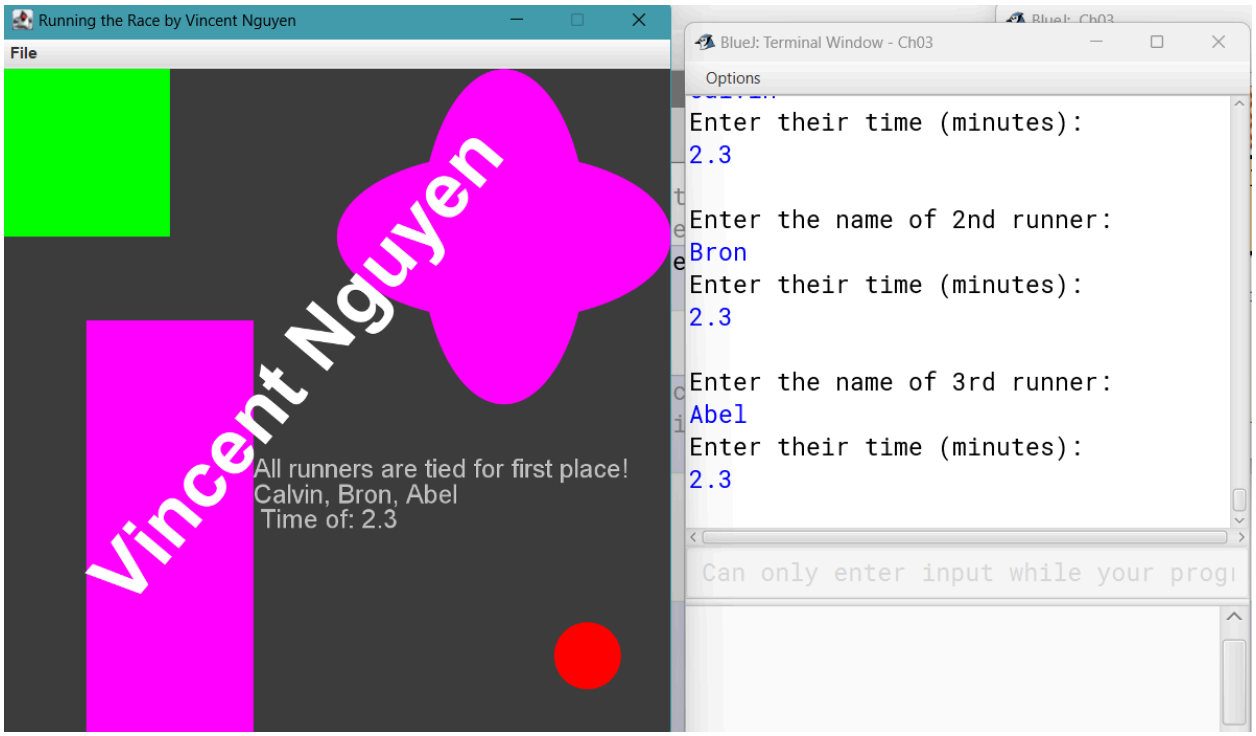
```
            }
        }
        else // that means runner 3 has to be the last option to be first
        {
            first = runner3;
            time1 = runTime3;
            if(runTime1 < runTime2)
            {
                second = runner1;
                time2 = runTime1;
                third = runner2;
                time3 = runTime2;
            }
            else
            {
                second = runner2;
                time2 = runTime2;
                third = runner1;
                time3 = runTime1;
            }
        }
    }
```

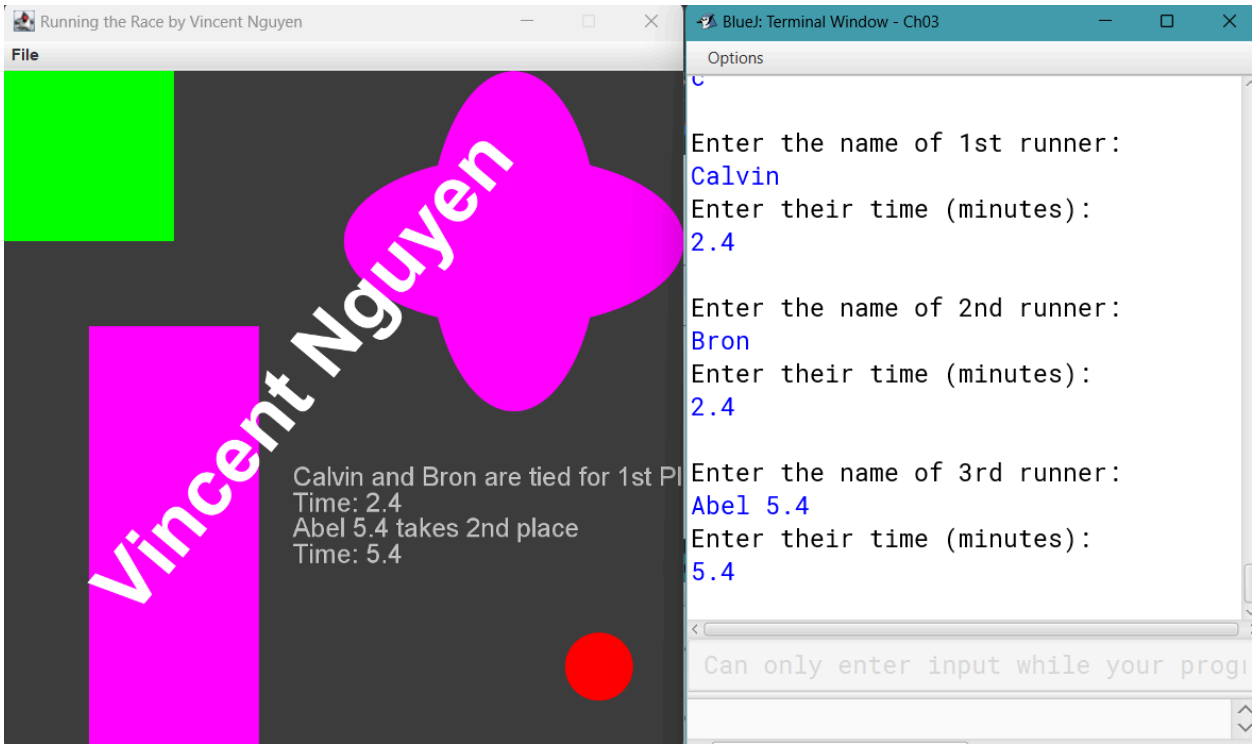
```
    // Draw Graphics Screen
    Draw box = new Draw();
    box.clear (Draw.DARK_GRAY);
    box.setXscale(0, 400);
    box.setYscale(400, 0);
    box.setTitle("Running the Race by Vincent Nguyen");

    // Shapes for background
    box.setPenColor(Draw.GREEN);
    box.filledSquare(50, 50, 50);

    box.setPenColor(Draw.RED);
    box.filledCircle(350, 350, 20);

    box.setPenColor(Draw.MAGENTA);
    box.filledEllipse (300, 100, 50, 100);
    box.filledEllipse (300, 100, 100, 50);

    box.filledRectangle (100, 300, 50, 150);


    // Font for name
    Font bigger = new Font("Mongolas", Font.BOLD, 60);
    box.setFont(bigger);
```

```
367         box.setPenColor(Draw.WHITE);
368         box.text(180, 180, "Vincent Nguyen", 50);
369
370         // Font for displaying results
371         Font big = new Font("Arial", Font.PLAIN, 20);
372         box.setFont(big);
373         box.setPenColor(Draw.LIGHT_GRAY);
374
375         int xPos = 170, yPos = 240;
376
377         if(!tied.isEmpty())
378         {
379             box.textLeft(xPos -20, yPos, "" + tied);
380             box.textLeft(xPos -20, yPos + 15, runner1 + ", " +
381             runner2 + ", " + runner3);
382             box.textLeft(xPos -20, yPos + 30, " Time of: " + runTime1);
383         }
384         else if(!tiedForFirst.isEmpty())
385         {
386             box.textLeft(xPos, yPos, tiedForFirst);
387             box.textLeft(xPos, yPos + 15,"Time: " + runTime1);
388             box.textLeft(xPos, yPos + 30,runner3 + " takes 2nd place");
389             box.textLeft(xPos, yPos + 45,"Time: " + runTime3);
390         }
391         else if (!tiedForSecond.isEmpty())
392         {
393             box.textLeft(xPos,yPos, "1st place: " + first + " with time: " + time1);
394             box.textLeft(xPos,yPos + 15, tiedForSecond);
395             box.textLeft(xPos,yPos + 30,"Time: " + runTime2);
396         }
397         else
398         {
399             box.textLeft(xPos, yPos + 15, "1st place: " + first +
400                                           "   Time: " + time1);
401             box.textLeft(xPos, yPos + 30, "2nd place: " + second +
402                                           "   Time: " + time2);
403             box.textLeft(xPos, yPos + 45, "3rd place: " + third +
404                                           "   Time: " + time3);
405         }
406     }
407
408 }
```

Graphics Screen






Running the Race by Vincent Nguyen
File
Vincent Nguyen
Calvin and Bron are tied for 1st Pl
Time: 2.4
Abel 5.4 takes 2nd place
Time: 5.4
BlueJ: Terminal Window - Ch03
Options
Enter the name of 1st runner:
Calvin
Enter their time (minutes):
2.4
Enter the name of 2nd runner:
Bron
Enter their time (minutes):
2.4
Enter the name of 3rd runner:
Abel 5.4
Enter their time (minutes):
5.4
Can only enter input while your progr