Create – Applications From Ideas
Written Response Submission Template

Please see Assessment Overview and Performance Task Directions for Student for the task directions and recommended word counts.

Program Purpose and Development

2a)

I created my interactive display using the JavaScript programming language. The purpose of my program is to display trends in winning lottery numbers in a way that is more digestible for humans to read. The video illustrates how someone might interact with the project (by clicking the bars to pop up a readout) if they were to collect data from it, and, in addition, it references the locations of other pieces of data that are important to the project.

2b)

I started my project by trying to find a data set that had enough elements in it each with enough variables to be useful to me. I eventually settled for the Lottery Number data set after 2 days of searching. The first challenge I encountered was how to separately parse out the first five numbers of the lottery numbers from their string so that they could be counted. I ended up coming up with a combination of the subString() method and the parseInt() method, both of which I had very little prior knowledge with. This solution worked great and was very concise and intuitive. This was done independently. Work continued on smoothly for the most part and I got some commentary from my partner on ways to make the graphs more readable. Towards the end of the project, I did have some trouble drawing the bar graphs in the correct positions, although my partner, who was also doing something with bar graphs, helped my figure out the correct way to draw them in the for loops; only change one position variable of the rectangle in the loop, not both.

2c)

If your captured code segment is an image, click the Picture icon to browse to the location of your saved image. If your captured code segment is text, include it with the rest of your 2c written response below.

```
var frontCount = 0;
  var backCount = array.length-1;
  console.time("Time Taken To Complete Sorting");
  for(var i = 0; i < array.length/2; i++){
   for(var j = frontCount; j < backCount; j++){
    compares++;
    if(array[j][11] > array[j+1][11] || array[j+1][11] === null){
     var temp = array[j];
     array[j] = array[j+1];
     array[j+1] = temp;
     swaps++;
    }
   }
   backCount--;
   for(var k = backCount; k > frontCount; k--){
    compares++;
    if(array[k][11] < array[k-1][11] || array[k][11] === null){
     var temp = array[k];
     array[k] = array[k-1];
     array[k-1] = temp;
     swaps++;
    }
   }
   frontCount++;
   }
console.timeEnd("Time Taken To Complete Sorting");
```

The algorithm I developed independently was one of the sorting routines I had available for my program. The segment uses for loops that iterate through 1/2 of the total array of lottery numbers each and that check if the current tie number is greater than or less then the preceding or next number in the array when needed. These logic statements are managed by math, since they check the numerical value of the strings. Inside these logic statements are each an algorithm that creates a temporary variable and sets its value to one of the two elements checked in the logic statement, then switches the two element's values with each other. The temporary variable needs to be created here because otherwise the two values would end up equaling the same one value and would not switch. Overall, there is a lot of math being used through out all of the algorithms. Without this sorting routine, I would not be able to iterate through the array correctly later on in the program when loading the smaller arrays and displaying the bar graphs.

2d)

If your captured code segment is an image, click the Picture icon to browse to the location of your saved image. If your captured code segment is text, include it with the rest of your 2d written response below.

```
for (var m = 0; m < firstFiveArray.length; m++){
   if (firstFiveArray[m] > firstFiveArray[mostCommon[0]] || m === 0){
     mostCommon[0] = m;
   }
   if (firstFiveArray[m] < firstFiveArray[leastCommon[0]] || m === 0){
     leastCommon[0] = m;
   }
   var x = m*10+100;
   var y = firstFiveArray[m];
   ctx.rect(x, canvas.height/2, 5, -y);
   ctx.fillStyle = 'rgb(0, 0, 0)';
   ctx.fill();
   if (m === 0 || m === firstFiveArray.length-1){
     ctx.fillText((m+1), x-2.5, canvas.height/2+10);
   }
   if (doListeners === 0){
     addListener(x, canvas.height/2, x+5, canvas.height/2-y, firstFiveArray,
firstFiveArray, megaArray, tiesArray, m, mostCommon, leastCommon);
   }
 }
```

This is one of the three for loops I have in my display() method for drawing the bar graphs. The abstraction integrates mathematical concepts to properly draw each bar the correct distance from each other and in the correct positions. The abstraction integrates logical concepts to check if the current element is the first or last element in the array, to check if it is the biggest or smallest number in the array, or to check if this is the first time through the for loop and, as such, an event listener should be created. The method this for loop is in is only called once at the beginning of the program and once each time a bar is unclicked. Because of this abstraction, I did not have to write the same if statements a total of over 100 times for each of the three different types of numbers present in the program.