Multicore computing with Java
Due: Monday, March 6
20 points

Using the example program that was presented, write a Java program that will use multiple cores to add the elements of an array of int values.

Your main method should create the array and randomly generate the values for the array. It should then pass that array to an object that is created to handle the computation.

Classes you will create

- The class that handles the computation will be similar to the Multicore class in the example. This class should retrieve the sums from the Future's and add those together to get the overall sum. You can do most of the computation in the constructor method the way my example does, or you can not have the constructor do much and put the code in a method that you call in main after creating the instance of this class.
- You will also need to create a class similar to the Compute class that implements the Callable interface. This class should use it's ID, the number of processors involved, and the size of the array to determine which part of the array it is supposed to sum. The call method of this class should return its sum

Experiments to run

1. Start with a fairly small array – say 1000 int values. This will likely be faster in the single process version. Keep increasing the size of the array – can you determine a point where the multiprocessor approach becomes faster?
2. Try generating more tasks than the number of available processors. Is this code slower or faster than when you match the number of tasks to the number of processors?
3. If you are able, run your code on a different computer that has a different number of cores. Do your results from the first experiment change at all?