Algorithm Investigator

Your Professor stayed up a little too late working on her latest project and as a result the finished product has a rather significant bug. While the following algorithms were implemented correctly, they were not labeled properly. It is your job to match the correct label with each algorithm using what you know about algorithm analysis.

You must work in groups of two or three on this lab. Submit only one write-up with all names (either person may submit).

**public** **static** **int**[] copyArray(**int**[] input){

**int**[] answer = **new** **int**[***MAX\_VALUE***];

**for**(**int** i = 0; i < ***MAX\_VALUE***; i++){

**if** (i < input.length){

answer[i] = input[i];

} **else** {

answer[i] = 0;

}

}

**return** answer;

}

**public** **static** **int** maxConsecSum(**int**[] input){

**int** answer = Integer.***MIN\_VALUE***;

**for**(**int** i = 0; i < input.length; i++){

**for**(**int** j = i; j < input.length; j++){

**int** thisSum = Integer.***MIN\_VALUE***;

**for**(**int** k = i; k <= j; k++){

thisSum += input[k];

}

**if**( thisSum > answer){

answer = thisSum;

}

}

}

**return** answer;

}

**public** **static** **int**[] sort(**int**[] input){

**int**[] answer = **new** **int**[input.length];

**for**(**int** i = 0; i < answer.length; i++){

**int** max = Integer.***MIN\_VALUE***;

**int** pos = -1;

**for**(**int** j = 0; j < input.length; j++){

**if** (input[j] > max){

pos = j;

max = input[j];

}

}

answer[i] = max;

input[pos] = Integer.***MIN\_VALUE***;

}

**return** answer;

}

**public** **static** **int** findMax(**int**[] input){

**int** answer = Integer.***MIN\_VALUE***;

**for**(**int** i = 0; i < input.length; i++){

**if** (input[i] > answer){

answer = input[i];

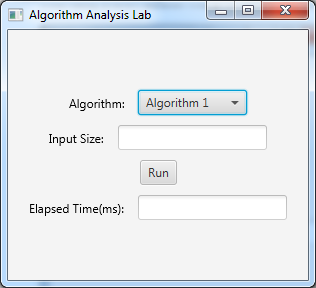
}

}

**return** answer;

}

Begin by downloading the COMSC111Lab3 program (this is an executable .jar file). You can run this as you would a .exe file. If you run the program, you should see this:



The drop down menu allows you to select one of the four algorithms that correspond to the four pieces of code above. In the Input Size box, you can enter the size of the input. This will be the length of the array that is the input to each method above. Then click the Run button and you will see the time the method took in the Elapsed Time box.

Next devise a plan to match each algorithm (findMax, copyArray, maxConsecSum, sort) to its label in the program (Algorithm 1, Algorithm 2, Algorithm 3, Algorithm 4). Then implement your plan. Be sure to take notes and record all of your results. Finally, write up your findings in a lab report. (See the What to turn in section below for the format and requirements of the report.)

You should create a table and/or graph of the runtime of each algorithm, for several input sizes (these will be your results). Your graphs should have input size on the x-axis and runtime on the y-axis (see examples in the textbook). Create the graphs with a program like Excel.

You do not have to use the same input sizes for every algorithm, since this might not produce the most useful data; but if you intend to compare two algorithms directly by how fast they run, you should use the same input size for that comparison.

**Note 1**: For several reasons, the data will be noisy – just like all “real” data (don’t expect a smooth curve). This means you may need many data points or repeated trials of each input size in order to determine the form of the growth function.

**Note 2**: Start with small inputs (Don’t try 1,000,000 first! Inputs sizes less than 10,000 will probably be enough.)

What to turn in

A written report including the following three sections

1. Methods: A description of your experimental design. This is your plan to match the algorithms to the labels. If you found that you needed to change your original plan include a description of what needed to change and why.
2. Results: The results (time data) you obtained. This section should include the tables and graphs. It may not have very much writing.
3. Discussion: A discussion of your results in which you tell me your conclusions (which algorithm matches with which label) and how you have arrived at your conclusions. Put your final matching in a table so that I can easily find it. You must also describe your thought process in reaching your answers. Be sure to support any conclusion that you make by referencing both your data and the provided code.

Your submission should be either a Word document or pdf file.

Suggestions for collaboration

You will need to run each of the algorithms several times, determine the asymptotic time complexity of each piece of code, determine a matching, and write a report during this lab. All members of the group will need to communicate and participate in a hands on way in order to finish in a timely manner. Here are some suggestions for organizing your group.

* When collecting data consider having one person run the program on one computer while another group member enters the results into Excel on another computer.
* Consider having each group member analyze and determine the time complexity of one or two pieces of code individually, then reconvene as a group and have each group member explain their answers to the group.
* The Discussion section is critical. To receive a high score on this lab you must have detailed and well supported reasons for your matching. Rather than assigning one group member to summarize the group’s conclusions while others work on other tasks, consider writing the Discussion section as a group. Have one person type while everyone contributes to the ideas and the wording.

Grading

There is no coding in this lab. Therefore a significant portion of the grade will be based on the quality of your written report. This includes completeness, clarity, and general presentation (neatness).

Grade breakdown is as follows:

Correctly matching the algorithms: 5 points

Neatness and presentation (including grammar and spelling): 5 points

Your process and explanation: 10 points