This lab concerns how we might devise an algorithm to find out whether or not a list of numbers contains a duplication; of course, if it doesn’t, then the elements are unique.

Assignment The assignment is to turn in your answers to each of the following three problems. Note: I consider an algorithm which you have written as either discrete steps or as pseudocode complete if I can take it and write a C/C++ program which would implement it. If there is anything ambiguous or unspecified in your procedure I will take off points. Please:

i. Email me your results as in-line text.
ii. Keep line length under 80 characters per line.
iii. Do not send any attachments.

1. First suppose that the list contains \( n \) real numbers, for example:

```c
double array[n];
```

Describe in precise terms (step 1, step 2, step 3, etc.) how you could use a carefully chosen sorting algorithm to solve the problem of finding out if the list contains a duplication. What is the complexity of your solution and why? Do you think you can do any better? Why or why not?

2. Now suppose that the list contains \( n \) positive integers (i.e. \( \text{array}[i] > 0.0 \) for all \( i = 0, 1, 2, \ldots, n-1 \)). Does your algorithm in Part 1. still work? What is the complexity of your solution? Do you think you can do any better? Why or why not?

3. Now suppose that the list contains \( n \) unsigned integers (in the sense of C/C++, each of which is smaller than \( 5n \)), for example:

```c
unsigned array[n];
```

where \( \text{array}[i] < 5n \) for all \( i = 0, 1, 2, \ldots, n-1 \). Does your algorithm in Part 1. still work? Can you produce a better algorithm, one which has a strictly smaller complexity? Why or why not? If, so, describe you algorithm in C/C++ pseudocode and explain why it has smaller complexity. You do not have to write a full program but it should be possible for a proficient C/C++ programmer to take your pseudocode and unambiguously produce a working program in a reasonable amount of time.