Problem 1. For the following problems use regular bubble sort

(a) Given the following list of numbers, $[20, 0, 10, 6, 4, 9, 14, 15, 7, 13]$, how would the list look like after three complete passes of bubble sort?

(b) How many comparisons occur after five passes of bubble sort are complete on the array in Part (a)? Show your work.

(c) Now consider an arbitrary unsorted array of $fifty$ values. How many comparisons would have occurred after $thirtyfive$ passes of bubble sort? How did you find it? Show your work.

Problem 2. *Collaborative filtering* is a technique for generating recommendations. Suppose a website wants to come up with suggestions for products, movies, songs, news stories, and so on. In collaborative filtering, the idea is to identify other users who have similar preferences, and to recommend to you things that have been that have been popular with them. Thus collaborative filtering algorithms require a formal notion of "similarity" between users, and the problem of computing inversions captures some of the essence of this problem. For example, an array, $A = [1, 3, 5, 2, 4, 6]$ requires three inversions, the 5, and 2, the 3, and the 2, and the 5, and the 4. Your task is to write a brute force pseudo-code to count the number of inversions in an arbitrary array of length n. How many exact number of comparisons does your algorithm take in the worst case? Show your work.

Problem 3. In computer vision we are always interested in finding the change of intensity values, usually for edge detection. These changes in intensities are obtained by convolving (or cross correlating) an image with a filter. The convolution results in a correlated image representing the changes in intensity values in the horizontal and vertical directions in the image. These changes in gradients or derivatives may contain positive, negative or zero values.

Suppose you are provided these images of gradients for edge detection. We are interested in finding the region of the brightest edge. Write a combination of pseudocode and English sentences (within pseudo-code) to find the region (rectangular area) of the maximum intensity(gradient) change. For each such region the gradient(or intensity) value is obtained by adding all the values of that subregion. For example, for the following image

6	-5	-7	4	-4
-9	3	-6	5	2
-10	4	7	-6	3
-8	9	-3	3	-7

the maximum intensity region is given by the subregion

4	7
9	-3

What is the asymptotic runtime of your algorithm? *Note*: The rectangular region can be single or more cells and not just 2×2 like the example and also, the input is an $n \times n$ matrix.