1) Write a parallel program using OpenMP to implement the Odd-even transposition sort. Vary the input size and analyse the program efficiency.
Hint: Odd-even transposition sort is a sorting algorithm that's similar to bubble sort. The list **a** stores **n** integers, and the algorithm sorts them into increasing order. During an *"even phase"* (phase % 2 == 0), each odd-subscripted element, a[i], is compared to the element to its *"left"* a[i−1], and if they're out of order, they're swapped. During an *"odd"* phase, each odd-subscripted element is compared to the element to its right, and if they're out of order, they're swapped. A theorem guarantees that after n phases, the list will be sorted.

	Subscript in Array					
Phase	**0**		**1**		**2**	**3**
0	9	↔	7		8	↔ 6
	7		9		6	8
1	7		9	↔	6	8
	7		6		9	8
2	7	↔	6		9	↔ 8
	6		7		8	9
3	6		7	↔	8	9
	6		7		8	9

2) Write a parallel program using OpenMP to implement multi-threaded tokenizer for a text file.
Hint: The tokens are just contiguous sequences of characters separated from the rest of the text by white space—spaces, tabs, or newlines. Assume that, the text file contains English text. A simple approach to this problem is to divide the input file into lines of text and assign the lines to the threads in a round-robin fashion. The first line goes to thread 0, the second goes to thread 1, . . . , the t^{th} goes to thread t, the t + 1^{st} goes to thread 0, and so on. Each thread then tokenizes the input line and prints each of the tokens along with the ThreadID

3) Write an OpenMP program to find the Summation of integers from a given interval. Analyze the performance of various iteration scheduling strategies.

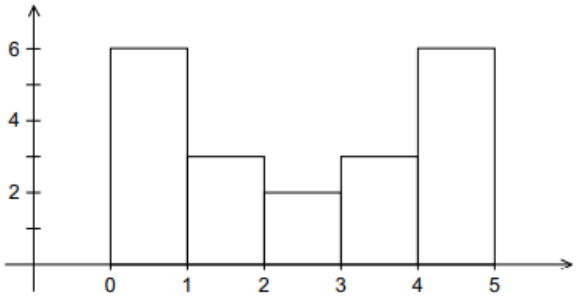
4) Write a parallel program using OpenMP to compute π using random shooting.
Hint: Shoot randomly into a square [0, 1]×[0, 1] and count how many shots hit inside the unit circle and how many do not. Then calculate the ratio of number of points lied inside the circle and total number of generated points. We know that, area of the circle is πr^2 and area of the square is $4r^2$. Now, for a very large number of generated points,

$$\pi = 4 * \frac{No.of\ points\ generated\ inside\ the\ circle}{Number\ of\ points\ generated\ inside\ the\ square}$$

5) Write a parallel program using OpenMP to generate the histogram of the given array A.

Hint: To generate histogram, we simply divide the range of the data up into equal sized sub intervals, or bins and determine the number of measurements (frequency) in each bin.
Example: suppose our data are

1.3, 2.9, 0.4, 0.3, 1.3, 4.4, 1.7, 0.4, 3.2, 0.3, 4.9, 2.4, 3.1, 4.4, 3.9, 0.4, 4.2, 4.5, 4.9, 0.9.



Where, Y axis represents the frequency of occurrence of the values and the x axis represents the bins.