ASSIGNMENT-1

Q.1. Performance analysis of Bubble Sort Write the program to analyze the performance of two different versions of bubble sort for randomized data sequences of integers. (i) BUBBLE SORT that terminates if the array is sorted before n-1th Pass. (ii) BUBBLE SORT that always completes the n-1th Pass.

CODE:

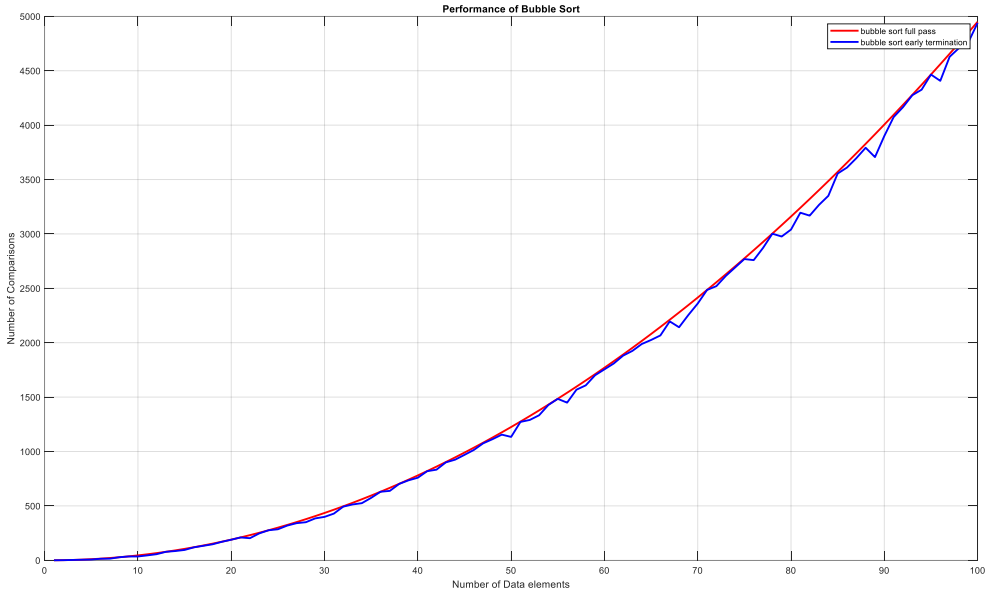
```
max_size = 100;
x_axis = 1:max_size;
y_axis = zeros(1,max_size);

for n= 1:max_size
    arr = round(rand(1, n) * 100);
    num_comparisons = 0;

    for i=1:n-1
        for j=1:n-i
            num_comparisons = num_comparisons+1;
            if arr(j) > arr(j+1)
                t = arr(j);
```

```
                arr(j) = arr(j+1);
                arr(j+1) = t;
            end
        end
    end
    y_axis(n) = num_comparisons;
end
figure;
plot(x_axis, y_axis, '-', 'LineWidth', 2, Color='r');
title("Performance of Bubble Sort");
xlabel("Number of Data elements");
ylabel("Number of Comparisons");
grid on;



y_axis2 = zeros(1, max_size);
% Bubble Sort with Early Terminations
for n= 1:max_size
    arr2 = round(rand(1, n) * 100);
    num = 0;
```

```
    for i=1:n-1
        swapped = false;
        for j=1:n-i
            num = num+1;
            if arr2(j) > arr2(j+1)
                t = arr2(j);
                arr2(j) = arr2(j+1);
                arr2(j+1) = t;
                swapped = true;
            end
        end
        if ~swapped
            break;
        end
    end
    y_axis2(n) = num;
end

hold on;
```

```
plot(x_axis, y_axis2, Color='b', LineWidth=2);
legend('bubble sort full pass', 'bubble sort early termination');
```

SIMULATION:



Observations:

From the graph, we see that the bubble sort that has early terminations shows better performance than the traditional bubble sort algorithm (the area under the curve is less) in the average case analysis.

Q.2. Performance analysis of Bubble and Insertion Sort Write the program to compare the performance of insertion sort and bubble sort for randomized data sequence of integers. Analyse their time and space complexities theoretically and then validate these complexities by running experiments with different sizes of input data

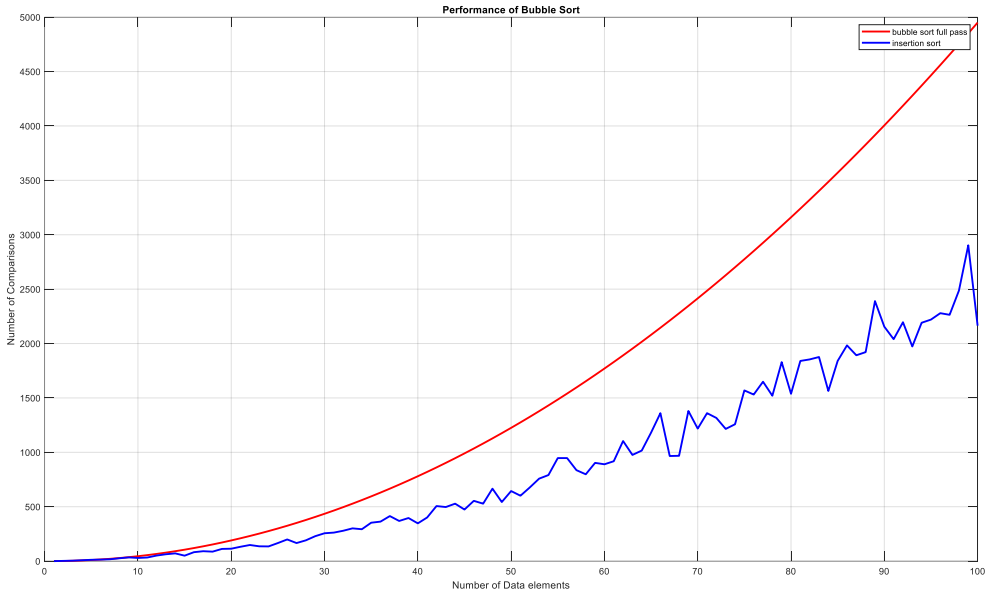
CODE:

```
max_size = 100;
x_axis = 1:max_size;
y_axis = zeros(1,max_size);

for n= 1:max_size
    arr = round(rand(1, n) * 100);
    num_comparisons = 0;

    for i=1:n-1
        for j=1:n-i
            num_comparisons = num_comparisons+1;
            if arr(j) > arr(j+1)
                t = arr(j);
```

```
                arr(j) = arr(j+1);
                arr(j+1) = t;
            end
        end
    end
    y_axis(n) = num_comparisons;
end
figure;
plot(x_axis, y_axis, '-', 'LineWidth', 2, Color='r');
title("Performance of Bubble Sort");
xlabel("Number of Data elements");
ylabel("Number of Comparisons");
grid on;



y_axis2 = zeros(1, max_size);
% Insertion Sort
for n= 1:max_size
    arr2 = round(rand(1, n) * 100);
    num = 0;
```

```
    for i=2:n
        current = arr2(i);
        j = i-1;
        while j >= 1 && current < arr2(j)
            arr2(j+1) = arr2(j);
            j = j-1;
            num = num +1;
        end
        arr2(j+1) = current;
        num = num +1;
    end

    y_axis2(n) = num;
end

hold on;
plot(x_axis, y_axis2, Color='b', LineWidth=2);
legend('bubble sort full pass', 'insertion sort');
```

SIMULATION:



OBSERVATIONS:

The insertion shows the same irregular behavior similar to the bubble sort with early terminations but performs much better on all different sizes of inputs in average case analysis compared to the traditional bubble sort algorithm.

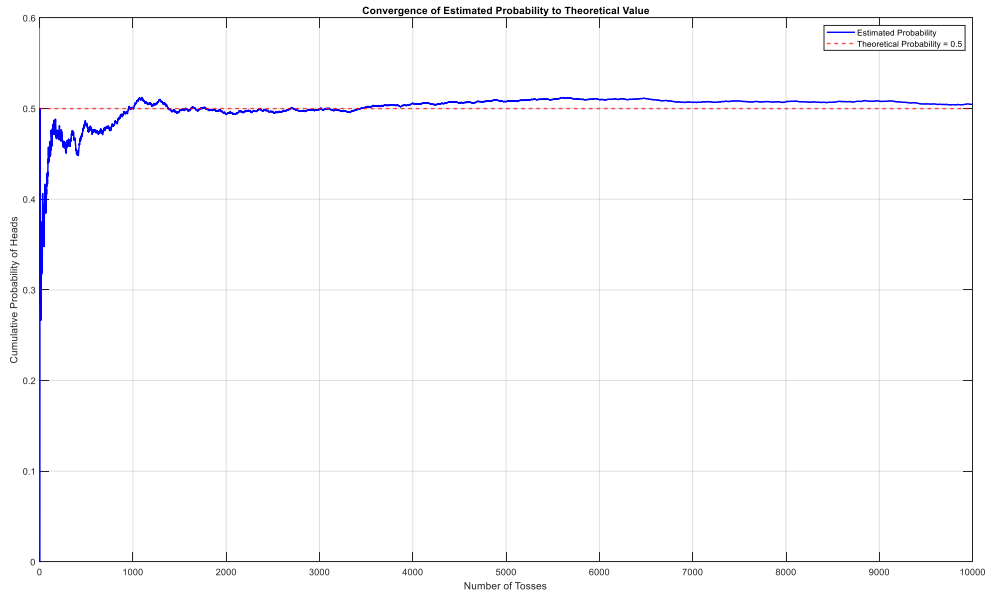
Q5. Through the simulation, show that probability of getting HEAD by tossing a fair coin is about 0.5. Write your observation from the simulation run.

CODE:

```
% Number of trials (tosses)
numTosses = 10000;

% Simulate the coin tosses
% Heads = 1 if random number < 0.5, otherwise Tails = 0
tosses = rand(1, numTosses) < 0.5;

% Calculate cumulative probability of heads
cumulativeHeads = cumsum(tosses);
cumulativeProbability = cumulativeHeads ./ (1:numTosses);

% Plotting the line plot
figure;
```

```
plot(1:numTosses, cumulativeProbability, 'b-',
'LineWidth', 1.5);

hold on;

yline(0.5, 'r--', 'LineWidth', 1.5); % Probability of 0.5

hold off;


% Customize plot

xlabel('Number of Tosses');

ylabel('Cumulative Probability of Heads');

title('Convergence of Estimated Probability to
Theoretical Value');

legend('Estimated Probability', 'Theoretical Probability
= 0.5');

grid on;
```

SIMULATION:



OBSERVATIONS:

The values that are generated using the standard distribution initially start showing more probability of tails but as the no. of epochs increases the probability reaches 0.5 (theoretical probability). Although the probability in theory should be 0.5 as there as 2 possible out of which we desire had. But the graph here says otherwise.