CS 104

ÖZYEĞİN ÜNİVERSİTESİ

LAB WEEK 13

1. In Bubble sort, you repeatedly swap the adjacent elements that are not in correct order until there are no possible swaps. (if there are no possible swaps, it means that the list is sorted.)
   Create a random array of size 10 and fill it with integers between 1-100.

   Write a method called bubbleSort to sort the array you have created. Call the method and print your array after the method call to check whether you have successfully sorted the array.

2. Counting sort is a sorting technique that can be applied to an array holding integers from a certain range. Suppose that you are given an array of 10 integers and those integers are from the range 0-9. Instead of sorting those numbers directly, you count the frequency of each number and create a frequency array. Then you should use the frequency array to print the numbers in ascending order.

   Input array: {1, 4, 1, 2, 7, 5, 2, 3, 9, 8}
   Frequency array: {0, 2, 2, 1, 1, 1, 0, 1, 1, 1}

   Then looking at the frequencies in frequency array, you should print the numbers in the original array in ascending order.

   Two 1's: Print 1 1

   Two 2's: Print 2 2

   One 3: Print 3

   One 4: Print 4

   …

   1 1 2 2 3 4 …

   Write a method called printSorted which takes an integer array and range where range is the range of the numbers inside the array. Your method should print the numbers inside the array in ascending order.

3. Implement recursive and iterative binary search methods.

4. Implement recursive version of insertion sort. Remember its definition from the lectures. Insertion sort is a simple sorting algorithm that works similar to the way you sort playing cards in your hands. The array is virtually split into a sorted and an unsorted part. Values from the unsorted part are picked and placed at the correct position in the sorted part. To sort an array of size n in ascending order:
   1: Iterate from arr[1] to arr[n] over the array.
   2: Compare the current element (key) to its predecessor.
   3: If the key element is smaller than its predecessor, compare it to the elements before. Move the greater elements one position up to make space for the swapped element.

   Recursion idea:
   1: Base Case: If array size is 1 or smaller, return.
   2: Recursively sort first n-1 elements.
   3: Insert last element at its correct position in sorted array.

5. Write a Python program to sort a list of elements using Bogosort. We will implement its randomized version. See details below.

   Bogosort is a particularly ineffective sorting algorithm based on the generation and test paradigm. The algorithm successively generates permutations of its input until it finds one that is sorted. It is not useful for sorting, but may be used for educational purposes, to contrast it with other more realistic algorithms. Two versions of the algorithm exist: a deterministic version that enumerates all permutations until it hits a sorted one, and a randomized version that randomly permutes its input. An analogy for the working of the latter version is to sort a deck of cards by throwing the deck into the air, picking the cards up at random, and repeating the process until the deck is sorted. Its name comes from the word bogus.