CSCI 115 Lab

Lab 6- Quick Sort

- Professor: Dr. Matin Pirouz
  Email: mpirouz@csufresno.edu

- TA: Shreeja Miyyar
  Email:shreejarao12@mail.fresnostate.edu

Table of Contents

Quick Sort

- It is a sorting algorithm which uses the divide and conquer approach.
- Using an element of the array as a pivot, it partitions the array around the pivot.
- Usually the pivot is picked as follows:
  - Pick the first element of the array.
  - Pick the last element of the array.
  - Pick a random element of the array.
  - Pick the median of the array.

- The important process in this algorithm is ***partition.*** The array is divided into two subarrays – Array[i.. q] and Array[q+1..j], such that Array[i.. q] is less than the pivot and Array[q+1.. j] is greater than pivot. This process is done in linear time.

- This process is done recursively until the entire array is sorted and it is sorted in place.

- Time Complexity:
  - Best case: The partition produces two equal sized subarrays -> **O(nlogn)**
  - Worst case: The partition produces a subarray with size 1 and another subarray with size n-1 -> **$O(n^2)$**

Quick Sort algorithm

This is the algorithm where the pivot is the first element of the array.

Initially set p = 0 and r = n-1; where n is the length of the array

QuickSort (array, p, r)

If (p < r)

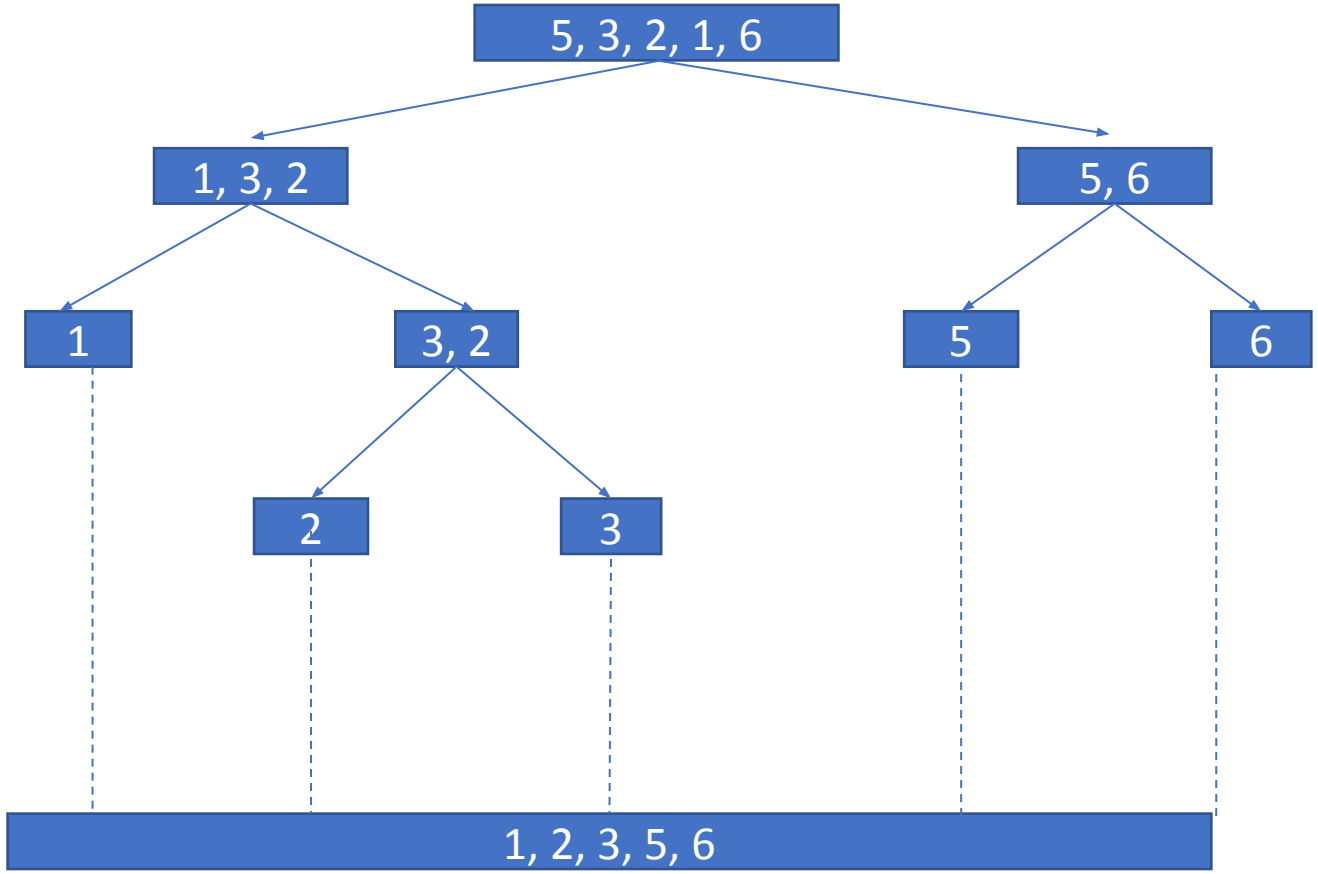
- Q = Partition(array, p, r)
- QuickSort(array, p, Q)
- QuickSort(array, Q+1, r)

Partition (array, low, high)

- pivot = array[low]
- i=low-1
- j=high+1
- While true
  - Keep decrementing j until A[j] <= pivot
  - Keep incrementing i until A[i] >= pivot
  - If i is less than j;
    - swap A[i] and A[j]
  - Else;
    - Return j

Partition example

The pivot is the first element of the array.



Lab Assignment

1. Write a program that takes a list and sorts it using Quick Sort for different pivots
   - Pivot Choice 1: The first element in the list
   - Pivot Choice 2: A random element in the array.
   - Pivot Choice 3: The median of the first, middle, and last elements in the array.

   Hint: Use the Quick sort algorithm mentioned in the slides to write the program.

   Use rand()%(size of the array) to pick a random index in the array.

2. Compare the time complexity and execution time for all three pivot values:

Hint: You can use clock() function to record execution time.

3. Fill out the report sheet.

   Hint: Write a detailed report as per the template and provide an elaborate explanation on how the execution time is different for each algorithm.

Coding guidelines

- In the main function provide the input array to be sorted.
- Create a function quicksort() which takes 3 arguments - input array, left index and right index.
- Create a function partition() which takes 3 arguments - input array, left index and right index and implement the partition algorithm in that function.

Questions?