A picture containing text

Description automatically generated

**CS435**

*Prof. Emdad Khan*

*September 2019*

*Lab#5*

***Group 1***

*Group members:*

*Asad Ali Kanwal*

*Aser Ahmad Ibrahim Ahmad*

*Jean Wilbert Volcy*

*Zayed Hassan*

1. Problem 1

Steps of Quick sort for the array with leftmost value as pivot.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **6** | **2** | **4** | **3** | **5** |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **2** | **4** | **3** | **5** |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **6** | **2** | **4** | **3** | **5** |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **6** | **2** | **4** | **3** | **5** |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **2** | **4** | **3** | **5** |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **2** | **4** | **3** | **5** |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **6** | **2** | **4** | **3** | **5** | **1** | **2** | **3** | **4** | **5** | **6** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **2** | **4** | **3** | **5** |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **2** | **4** | **3** | **5** |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **4** | **3** | **5** |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **6** | **2** | **4** | **3** | **5** |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **2** | **4** | **3** | **5** |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **2** | **4** | **3** | **5** |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **4** | **3** | **5** |  |  |  |

|  |  |
| --- | --- |
| **3** | **3** |

|  |  |
| --- | --- |
| **5** | **5** |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **6** | **2** | **4** | **3** | **5** | ***1*** | ***2*** | ***3*** | ***4*** | ***5*** | ***6*** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **2** | **4** | **3** | **5** | ***2*** | ***3*** | ***4*** | ***5*** | ***6*** |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **2** | **4** | **3** | **5** | ***2*** | ***3*** | ***4*** | ***5*** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **4** | **3** | **5** | ***3*** | ***4*** | ***5*** |

|  |  |
| --- | --- |
| **3** | **3** |

|  |  |
| --- | --- |
| **5** | **5** |

1. Problem 2

This table contains the investigation of good and bad elements:

|  |  |  |  |
| --- | --- | --- | --- |
| Pivot, *x* | No. of elements ≥ *x* | No. of elements ≤ *x* | Comment |
| 5 | 2 | 6 | Good pivot |
| 1 | 8 | 0 | Bad pivot |
| 4 | 3 | 5 | Good pivot |
| 3 | 4 | 4 | Good pivot |
| 6 | 1 | 7 | Bad pivot |
| 2 | 6 | 2 | Good pivot |
| 7 | 0 | 8 | Bad pivot |
| 1 | 8 | 0 | Bad pivot |
| 3 | 4 | 4 | Good pivot |

Elements that are good pivots are: .

⸫ According to the example above, it is true that at least half of the elements are good pivots.

1. Problem 3

|  |  |
| --- | --- |
| **Algorithm** findElementEqualToItsIndex (A, start, end) | *Count of operations* |
| ***Input:*** sorted array A, starting position start, ending position end |  |
| ***Output:*** true if element A[m] = m is found, false otherwise |  |
| mid = (start + end) / 2 | *3* |
| **if** (A[mid] = mid) **then** | *3* |
| return true | *1* |
| **if** (A[mid] < mid and start != end) **then** | *3* |
| return *findElementEqualToItsIndex* (A, mid + 1, end) | *3 + T(n/2)* |
| **if** (A[mid] > mid and start != end) **then** | *3* |
| return *findElementEqualToItsIndex* (A, start, mid) | *2 + T(n/2)* |
| return false | *1* |

1. Problem 4

One pivot selection strategy that can be used to guarantee a Quick Sort with running time of is to use the Quick Select algorithm to select the smallest element, which is the median. The quick select algorithm has an expected running time of , and it guarantees a good pivot is found for the Quick Sort.

1. Problem 5
2. .

Check:

Check:

Check:

Check:

Check:

Check:

Check:

Check:

Check:

Check:

Check: