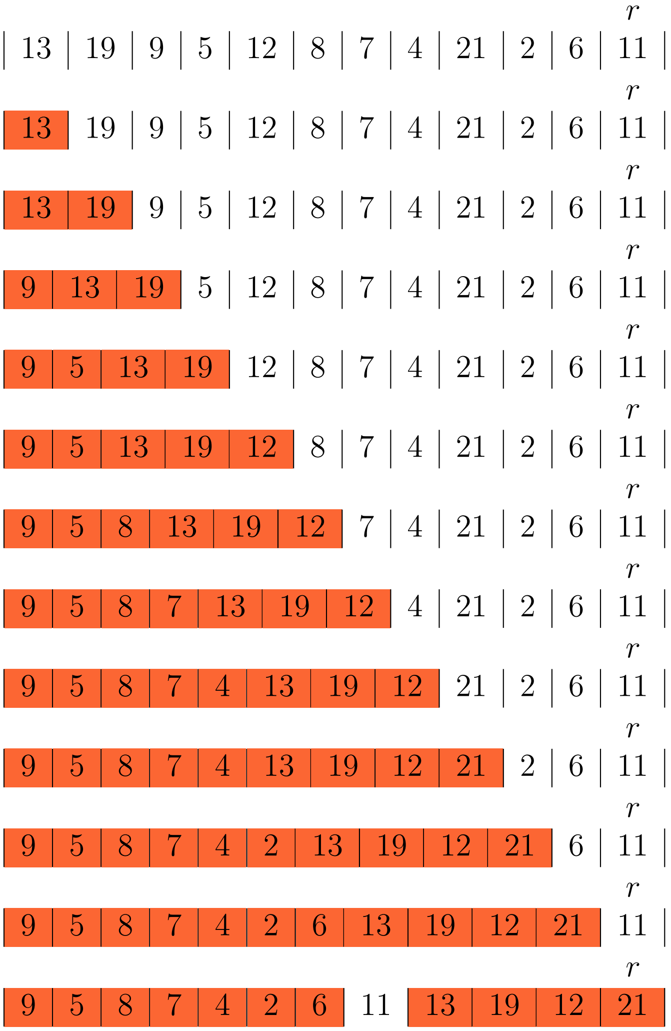
Name: Sabyasachi Sahoo
COSC 40403 - Analysis of Algorithms: Fall 2018: Homework 4
Due: 23:59:59 on September 25, 2018

Question	Points	Score
1	5	
2	5	
3	30	
Total:	40	

1. (5 points) Exercise 7.1-1: Using Figure 7.1 as a model, illustrate the operation PARTITION on the array $A = \langle 13, 19, 9, 5, 12, 8, 7, 4, 21, 2, 6, 11 \rangle$

Solution:



r
| 13 | 19 | 9 | 5 | 12 | 8 | 7 | 4 | 21 | 2 | 6 | 11 |

r
| 13 | 19 | 9 | 5 | 12 | 8 | 7 | 4 | 21 | 2 | 6 | 11 |

r
| 13 | 19 | 9 | 5 | 12 | 8 | 7 | 4 | 21 | 2 | 6 | 11 |

r
| 9 | 13 | 19 | 5 | 12 | 8 | 7 | 4 | 21 | 2 | 6 | 11 |

r
| 9 | 5 | 13 | 19 | 12 | 8 | 7 | 4 | 21 | 2 | 6 | 11 |

r
| 9 | 5 | 13 | 19 | 12 | 8 | 7 | 4 | 21 | 2 | 6 | 11 |

r
| 9 | 5 | 8 | 13 | 19 | 12 | 7 | 4 | 21 | 2 | 6 | 11 |

r
| 9 | 5 | 8 | 7 | 13 | 19 | 12 | 4 | 21 | 2 | 6 | 11 |

r
| 9 | 5 | 8 | 7 | 4 | 13 | 19 | 12 | 21 | 2 | 6 | 11 |

r
| 9 | 5 | 8 | 7 | 4 | 13 | 19 | 12 | 21 | 2 | 6 | 11 |

r
| 9 | 5 | 8 | 7 | 4 | 2 | 13 | 19 | 12 | 21 | 6 | 11 |

r
| 9 | 5 | 8 | 7 | 4 | 2 | 6 | 13 | 19 | 12 | 21 | 11 |

r
| 9 | 5 | 8 | 7 | 4 | 2 | 6 | 11 | 13 | 19 | 12 | 21 |

2. (5 points) Exercise 7.1-2: What value of q does Partition return when all elements in the array $A[p..r]$ have the same value? Modify Partition so that $q = \lfloor (p+r)/2 \rfloor$ when all elements in the array $A[p..r]$ have the same value.

Solution:
1.) When all the elements are same in an array, the value of q returned is r.

2.)

```
PARTITION(A, p, r)
x = A[r]
k = p  1
l = p  1
for j = p to r - 1
   if A[j] ¡ x
     k = k + 1
     exchange A[j] with A[k]
     l = l + 1
   else
     if A[j] = x
       l = l + 1
       exchange A[j] with A[l]
exchange A[l + 1] with A[r]
return (A, floor((l+k)/2) +1 )
```

3. (30 points) Using a Jupyter notebook (Python 3 programming language), create an array of 1000 random integers with each integer in the range [10000..99999] (inclusive). You can use this array to make deep-copies for each of the sorting methods. Implement the Randomized-Quicksort(), Counting-Sort(), and Radix-Sort() as presented in our textbook. Time each algorithm and rank them by their sorting times (lowest to highest) on sorting the same (one) instance of the array of 1000 random integers.

Solution: Submit solution in Jupyter notebook.