**CIS 500 – Fundamentals of Software Practice**

**Weekly Exercise #11**

**Topic: Searching and Sorting Techniques**

**Assignment:**

* Complete the given exercises on *binary search*, *selection sort*, *bubble sort*, *insertion sort*, and *quick sort* in the following pages.

**What to turn in on Blackboard?**

* Upload this file with your answers on Blackboard by midnight of due date.

**Binary Search Exercises:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 10 | 12 | 19 | 21 | 23 | 35 | 50 | 65 | 84 | 90 |

**Perform “binary search” on the list above for value 84. The first row shows the index positions.**

1. from\_pos = 0

to\_pos = 9

mid\_pos = (0+9)//2 = 4

1. from\_pos = (mid\_pos+1) = 4 + 1 = 5

to\_pos = 9

mid\_pos = (5+9)//2 = 7

1. from\_pos = (mid\_pos+1) = 7 + 1 = 8

to\_pos = 9

mid\_pos = (8+9)//2 = 8

1. from\_pos = mid\_pos = 8

to\_pos = 8

mid\_pos = 8 **Success! Value 84 found at index 8 in the list**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 10 | 12 | 19 | 21 | 23 | 35 | 50 | 65 | 84 | 90 |

**Perform “binary search” on the list above for value 35. The first row shows the index positions.**

1. from\_pos = 0

to\_pos = 9

mid\_pos = (0+9)//2 = 4

1. from\_pos = 4 + 1 = 5

to\_pos = 9

mid\_pos = (5+9) // 2 = 7

1. from\_pos = 5

to\_pos = 7 – 1 = 6

mid\_pos = (5+6)//2 = 5 **Success! Value 35 found at index 5 in the list**

1. from\_pos = \_\_\_\_\_\_

to\_pos = \_\_\_\_\_\_

mid\_pos = \_\_\_\_\_\_

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 10 | 12 | 19 | 21 | 23 | 35 | 50 | 65 | 84 | 90 |

**Perform “binary search” on the list above for value 95. The first row shows the index positions.**

1. from\_pos = 0

to\_pos = 9

mid\_pos = (0+9)//2 = 4

1. from\_pos = 4+1 = 5

to\_pos = 9

mid\_pos = (5+9)//2 = 7

1. from\_pos = 7+1 = 8

to\_pos = 9

mid\_pos = (8+9)//2 = 8

1. from\_pos = 8+1 = 9

to\_pos = 9

mid\_pos = (9+9)//2 = 9 **search stops when from\_pos not <= to\_pos i.e., (value 95 is not in the list)**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 10 | 12 | 19 | 21 | 23 | 35 | 50 | 65 | 84 | 90 |

**Perform “binary search” on the list above for value 21. The first row shows the index positions.**

1. from\_pos = 0

to\_pos = 9

mid\_pos = (0+9)//2 = 4

1. from\_pos = 0

to\_pos = 4 – 1 = 3

mid\_pos = (0+3)//2 = 1

1. from\_pos = 1+1 = 2

to\_pos = 3

mid\_pos = (2+3)//2 = 2

1. from\_pos = 2+1 = 3

to\_pos = 3

mid\_pos = (3+3)//2 = 3 **Success! Value 21 found at index 3 in the list**

**Apply “Selection Sort” to sort the list below.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **18** | **46** | **10** | **82** | **67** | **72** | **12** | **31** | **22** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **46** | **18** | **82** | **67** | **72** | **12** | **31** | **22** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **82** | **67** | **72** | **46** | **31** | **22** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **22** | **67** | **72** | **46** | **31** | **82** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **22** | **31** | **72** | **46** | **67** | **82** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **22** | **31** | **46** | **72** | **67** | **82** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **22** | **31** | **46** | **59** | **67** | **82** | **72** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **22** | **31** | **46** | **59** | **67** | **82** | **72** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **22** | **31** | **46** | **59** | **67** | **72** | **82** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **22** | **31** | **46** | **59** | **67** | **72** | **82** |

**Apply “Bubble Sort” to sort the list below.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **18** | **46** | **10** | **82** | **67** | **72** | **12** | **31** | **22** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **18** | **10** | **46** | **67** | **72** | **12** | **31** | **22** | **59** | **82** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **18** | **46** | **67** | **12** | **31** | **22** | **59** | **72** | **82** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **18** | **46** | **12** | **31** | **22** | **59** | **67** | **72** | **82** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **18** | **12** | **31** | **22** | **46** | **59** | **67** | **72** | **82** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **22** | **31** | **46** | **59** | **67** | **72** | **82** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **22** | **31** | **46** | **59** | **67** | **72** | **82** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |

**Apply “Insertion Sort” to sort the list below.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **18** | **46** | **10** | **82** | **67** | **72** | **12** | **31** | **22** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **18** | **46** | **10** | **82** | **67** | **72** | **12** | **31** | **22** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **18** | **10** | **46** | **82** | **67** | **72** | **12** | **31** | **22** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **18** | **46** | **82** | **67** | **72** | **12** | **31** | **22** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **18** | **46** | **67** | **82** | **72** | **12** | **31** | **22** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **18** | **46** | **67** | **82** | **72** | **12** | **31** | **22** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **46** | **67** | **72** | **82** | **31** | **22** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **31** | **46** | **67** | **72** | **82** | **22** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **22** | **31** | **46** | **67** | **72** | **82** | **59** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** | **12** | **18** | **22** | **31** | **46** | **67** | **72** | **82** | **59** |

**Demonstrate one application of the partition technique of “Quick Sort” to partition the list below. Use the first value (72) as the pivot value.**

**You are not required to sort the entire list (i.e., you do not have to go through the entire process of quick sort).**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **72(P)** | **82** | **46** | **98** | **59** | **22** | **12** | **10** | **34** | **67** | **18** | **31** |

Partition the list such that all elements less than the pivot are placed to its left and all elements greater than the pivot are placed to its right. We can do this by comparing each element with the pivot and swapping it with the element at the leftmost index if it is smaller than the pivot. Similarly, we can swap elements with the element at the rightmost index if it is greater than the pivot.

After performing one partitioning step, the list becomes:

* 18 82(L) 46 98 59 22 12 10 34 67 72(P) 31(R)

Now, we need to recursively sort the left and right sublists. For this, we can repeat the above two steps on each sublist.

* Left sublist: [18, 82, 46, 98, 59, 22, 12, 10, 34, 67]

We can choose 18 as the pivot.

After partitioning, the list becomes:

* 10 82(L) 46 98 59 22 12 18(P) 34 67 31(R)
* 10 18 12 98 59 22 46 82(L) 34 67 31(R)
* 10 12 18 22 59 46 82(L) 34 67 98 31(R)
* 10 12 18 22 31 46 82(L) 34 67 98 59(R)

The left sublist is now sorted.

* Right sublist: [67, 31]

We can choose 67 as the pivot.

After partitioning, the list becomes:

* 31 67(P)

The right sublist is now sorted.

Finally, concatenate the sorted left and right sublists along with the pivot to get the sorted list:

* 10 12 18 22 31 34 46 59 67 72 82 98

Left Half: [10,12,18,22,31,34,46,59,67]

Right Half: [82,98]

**Demonstrate one application of the partition technique of “Quick Sort” to partition the list below. Use the first value (75) as the pivot value.**

**You are not required to sort the entire list (i.e., you do not have to go through the entire process of quick sort).**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **75(P)** | **25** | **10** | **5** | **60** | **55** | **30** | **80** | **45** | **40** | **98** | **34** |

Partition the list such that all elements less than the pivot are placed to its left and all elements greater than the pivot are placed to its right. We can do this by comparing each element with the pivot and swapping it with the element at the leftmost index if it is smaller than the pivot. Similarly, we can swap elements with the element at the rightmost index if it is greater than the pivot.

34 25 10 5 60 55 30 45 40 75(P) 98(R) 80(R)

Now, we need to recursively sort the left and right sublists. For this, we can repeat the above two steps on each sublist.

* Left sublist: [34, 25, 10, 5, 60, 55, 30, 45, 40]

We can choose 34 as the pivot.

After partitioning, the list becomes:

* 5 25 10 34(P) 60 55 30 45 40

We can choose 5 as the pivot for the left sublist.

After partitioning, the list becomes:

* 5(P) 25 10 34 30 55 45 40 60

The left sublist is now sorted.

* Right sublist: [98, 80]

We can choose 98 as the pivot.

After partitioning, the list becomes:

* 34 25 10 5 60 55 30 45 40 75 98(P) 80

We can choose 80 as the pivot for the right sublist.

After partitioning, the list becomes:

* 34 25 10 5 60 55 30 45 40 75 98 80(P)

The right sublist is now sorted.

Finally, concatenate the sorted left and right sublists along with the pivot to get the sorted list:

* 5 10 25 30 34 40 45 55 60 75 80 98

Left Half: [5,10,25,30,34,40,45,55,60]

Right Half: [80,98]