Tuesday, March 15, 2022 11:31 AM

Exam:

↳attempt all exercises ↳insertion sort
↳HW ↳binary search
↳lectures
↳how to sort data set / list → binary search for value
↳give algorithm for finding max value
  ↳evaluate complexity

↳a lot of complexity analysis
  ↳O → # cells memory algorithm needs
    ↳based on n

Sorted(data) → Timsort
 ↳O(n log n) → best complexity for sorting algorithm

How to sort data set on your own
↳sorted → easier to find min, max, binary search

↳ Insertion Sort
- •insert( ) → make room by shifting everything else right
  - ↳ O(n)
- ↳ most basic sort

#Algorithm → explain, not write in python

InsertionSort(A):

#input: array A of n elements

| 5 | -2 | 3 | 0 |

#output: array A of elements rearranged in (increasing) order
(ascending)

↳ | 5 | -2 | 3 | 0 |

↑ ↑ 1

compare

if -2 < 5 → swap values

Write loop → go through $\underbrace{\text{all elements}}_{n}$

| -2 | 5 | 3 | 0 |

↑ ↑ 2

| -2 | 3 | 5 | 0 |

↑ ↑ 3

| -2 | 3 | 0 | 5 |

↑ ↑ 4

$O(n^2)$

↳ worts case → opposite order

need another loop

↳ nested loops → n^2

↳ every check point → stop & go backward

| -2 | 0 | 3 | 5 |

↑ ↑

✓ 5 $< n^2$

```
def insertion_sort(A):
    for i in range(1, len(A)):
        for j in range(i,0,-1):
            if A[j] < A[j-1]:
                temp = A[j-1]    # to swap 2 variables, need 3rd temp variable
                A[j-1]= A[j]
                A[j] = temp
    return A

Insertion_sort([2,-2,3,-3,4,-4])
```

```
# create matrix 2 by 3
matrix1=[[0] *3]* 2            ← not matrix
matrix1[1][2]=3  # modifies both
```

Actual matrix

```
matrix2 =[[0] *3 for j in range(2)]
matrix[1][2] = 3  ✓ → [[0,0,0],[0,0,3]]
```