DAA - LAB 3

Algorithm:

CSV file format

serial number; Basic salary; House Rent Allowance; Other Allowances; Income tax; Provident fund; Professional Tax

eg. 1; 56223; 5511; 2314; 6327; 4276; 1321

Gross Salary

Gross Salary = Basic Salary - House Rent Allowance - Other Allowance

Net Salary

Net Salary = Gross Salary - Income tax - Provident fund - Professional tax

// Input: Taken from csv file with above format

file_path = dataset.csv

dataset = readcsv(file_path)

GrossSalary (dataset)

// Input: csv file dataset

// Output: Value of Gross salary

return BasicSalary + HouseRentAllowance + OtherAllowance

NetSalary (dataset)

// Input: csv file dataset

// Output: value of Net Salary

return GrossSalary(dataset) - IncomeTax - ProvidentFund - Professional Tax

```
LinearMinMax(dataset)
// Input: csv file dataset
// Output: Id of employee with min &
            max salary
maxs = -inf
mins = inf
maxid = 0
minid = 0
for row in dataset:
  net = NetSalary(row)
  if net > maxs
    maxs = net
    maxid = row
  if net < mins
    mins = net
    minid = row
return maxid, minid

RecursiveMinMax(dataset, start, end)
// Input: csv file dataset
// Output: Id of employee with min & max salary
 if (start == end)
    return dataset.iloc[start], dataset.iloc[start]
mid = (start + end) / 2

maxL = RecursiveMinMax(dataset, start, mid)
maxR = RecursiveMinMax(dataset, mid+1, end)
minL = RecursiveMinMax(dataset, stard, mid)
maxL, minL = RecursiveMinMax(dataset, start, mid)
maxR, minR = RecursiveMinMax(dataset, mid+1, end)
```

```
if NetSalary[maxL] > NetSalary[maxR]
    maxid = maxL
else
    maxid = maxR
if NetSalary(minL) > NetSalary(minR)
    minid = minR
else
    minid = minL
return maxid, minid
```

TESTCASES:

- For positive testcases will give appropriate answer
- For Negative testcases:
  - If any Salary is negative:
    Output: INVALID INPUT
  - If ~~Salary~~ any column is left unfilled
    output: INVALID SIZE OF COLUMNS
  - If dataset is empty:
    output: EMPTY FILE

TIME COMPLEXITY:

1) LINEAR:

- Some Initialisations at the start take time complexity $O(1)$, ∴, there are 4 total time will be

$$4O(1)$$

- Now in the for loop we iterate over the entire dataset so, the time for that is, $\sum_{i=1}^{n} 1$

$$T(n) = \sum_{i=1}^{n} 1 + 4O(1)$$

$$T(n) = n + 4O(1)$$

$$T(n) = O(n)$$

∴, Time complexity is $O(n)$

2) DIVIDE & CONQUER

- we divide the problem in half and then find min & max in subarrays of size $\frac{n}{2}$ &

- ~~Then we recombine these arrays and~~
- Then we compare the values of min & max of left & right, ∴, there are 2 if statements, this takes $2O(1)$ time

- ∴, Eqn of Total time will be

$$T(n) = 2T\left(\frac{n}{2}\right) + 2O(1)$$

where $2T\left(\frac{n}{2}\right)$ as we subdivide array to 2 equal halves of size $\frac{n}{2}$

Now comparing this to

$T(n) = aT(n/b) + cn^d \in O(n^d)$

we get

$a = 2$

$b = 2$

$d = 0$

Now ∵,

$b^d = 2^0 = 1$

Now, ∵, $2 > 1$, ∴, $a > b^d$

∴, $T(n) \in O(n^{\log_2 2})$

$T(n) \in O(n)$

$T(n) = O(n)$

∴, Time complexity by divide & conquer is also $O(n)$