Kevin Shah Batch-C 23107006O

DAA Lab-3

Aim: To understand how visual studio code works and its installation process, code editing, productivity tips and learning about its extensions. Writing an algorithm to find gross and net salary of 2000 employees of ABC Co. Ltd and find maximum and minimum salary using divide and conquer.

Input: csv file having salary information of 2000 employees along with the breakup.

Theory: According to salary literature
- Gross salary → Base salary + HRA + other Allowances
- Net salary → Gross salary - Professional tax - Employee provident fund - Income Tax

Columns of csv file contain all these elements and row represents each employee.

Gross and net salaries of employees:

function calculate_salaries(employee):

// input: employee contains: basic_salary, HRA, other allowances, Income_Tax, Professional_Tax, Employee_Provident_Fund.

If any value < 0 in any column:
    employee.gross_salary = "Invalid"
    employee.net_salary = "Invalid"
Else:
    employee.gross_salary = employee.Basic_Salary + employee.HRA + employee.Other_Allowances

    employee.total_deductions = employee.Income_Tax + employee.Employee_Provident_Fund + employee.Professional_Tax

    employee.net_salary = employee.gross_salary - employee.total_deductions

→ linear algorithm

```
function minmax (arr[]) {
// searches for maximum and minimum salary value
// from every given array by using iterative approach
// ie by comparing every element present in array.
// input : an array arr[] containing the salaries of
//         the employees
// output: maximum and minimum values of the
//         employee's salary

    max_salary = arr[0]
    min_salary = arr[0]
    for (int i=1 ; i < arr.size ; i++) {
        if (a[i] > max_salary)
            max_salary = a[i];
        if (a[i] < min_salary)
            min_salary = a[i];

    return min_salary, max_salary; }
```

Time Complexity: (Linear approach)

base case is only 1 element present in array ie. O(1)

$$\sum_{i=0}^{n} 1 = (n-1) + 1$$

$$= O(n)$$

1) Algorithm (Divide-and-conquer Approach)

```
function findminmax (int arr[], low, high) {

// searches for maximum and minimum salary value
// from arr[] by using recursive divide and conquer
// input : an array arr[] and indices high, low
// output : The maximum and minimum value present
//          at that particular index.
int low = 0
int high = arr.size() - 1
if (low == high) {
    return {arr[low], arr[low]};
}
else if (high == low + 1) {
    if (arr[low] < arr[high]) {
        return {arr[low], arr[high]};
    } else {
        return {arr[high], arr[low]};
    }
}
else {
    int mid = low + (high - low) / 2;
    leftResult = findminmax(arr[], low, mid);
    rightResult = findminmax(arr[], mid+1, high);

    minimum = min(leftResult.min, rightResult.min);
    maximum = max(leftResult.max, rightResult.max);

    return minimum, maximum;
}
```

Test cases:

(Positive)

1) Input: employee_valid_1.csv

Output:

Linear search -> Min: 23167 (ID: 3), Max -> 101218 (ID: 6)
Divide and conquer -> Min: 23167 (ID: 3), Max: 101218 (ID: 6)

2) Input: employee_valid_2.csv

Output:

Linear search -> Min: 33923 (ID: 10), Max: 96456 (ID: 16)
Divide and conquer -> Min: 33923 (ID: 10), Max: 96456 (ID: 16)

3) Input: employee_valid_3.csv

Output:

Linear search -> Min: 34613 (ID: 23), Max: 122970 (ID: 18)
Divide and conquer -> Min: 34613 (ID: 23), Max: 122970 (ID: 18)

4) Input: employee_valid_4.csv

Output:

Linear search -> Min: 48907 (ID: 19), Max: 69370 (ID: 24)
Divide and conquer -> Min: 48907 (ID: 19), Max: 69370 (ID: 24)

5) Input: employee_valid_5.csv

Output:

Linear Search → Min: 30372 (ID: 19), Max: 128537 (ID 18)
Divide and Conquer → Min: 30372 (ID: 19), Max: 128537 (ID 18)

Negative:

6) Input: employee_invalid-1.csv //ID missing

output: Employee ID is missing for an entry

7) Input: employee_invalid_2.csv //Field Heads missing

output: Error: Missing required fields in input data

8) Input: employee_invalid_3.csv

ID 32 has alphabets as data and not no.

output: Invalid Data for Employee ID 32

9) Input: employee_invalid_4.csv

Output: Invalid on missing Employee ID

10) Input: employee_invalid_5.csv

output: No employee data available

For divide-and-conquer

Time complexity using masters theorem

$T(n) = 2T(\frac{n}{2}) + O(1)$ {O(1) is the base case ie. only one element present in array}

$a = 2$ {since we are using the recursive call of the function twice}

$b = 2$ {since we are using divide and conquer recursively each time f" is called the array is split into half by using this method}

general formula: $T(n) = aT(\frac{n}{b}) + f(n)$

$n^k = 1 \Rightarrow k = 0$

$a > b^k$
$2 > 2^0$
$2 > 1$

} This satisfies one of the master theorem's case

Hence we use this formula: $n^{\log_b a}$

$$\log_b a = \log_2 2 = 1$$

$$n^1 = n$$

Thus the time complexity of the algorithm is $O(n)$

Date
Page

Conclusion:

Hence we computed the gross and net salaries of employees found employees with minimum and maximum net salaries using two different functions.

Gross-Net salary is calculated by

Gross salary = Basic salary + HRA + Other allowance

HRA = 4% of Basic Salary

Net salary = Gross Salary - Deductions

Linear search was implemented - it traverses through the list of employees and keeps track of employee with minimum and maximum net salary

Time Complexity = O(n)

Divide and Conquer was implemented too.
It recursively divides into smaller sub arrays
Find min and max in each sub arrays and combine

Using master theorem we found $T(n) = O(n)$

Both achieve same complexity i.e. $O(n)$
∴ Linear Search is preferable due to simplicity.