# DAA LAB 3

Amit Rajkumar Ingle

S.Y.B.Tech computer

231070020

## AIM:

Write an algorithm to find gross and net salary of employees.

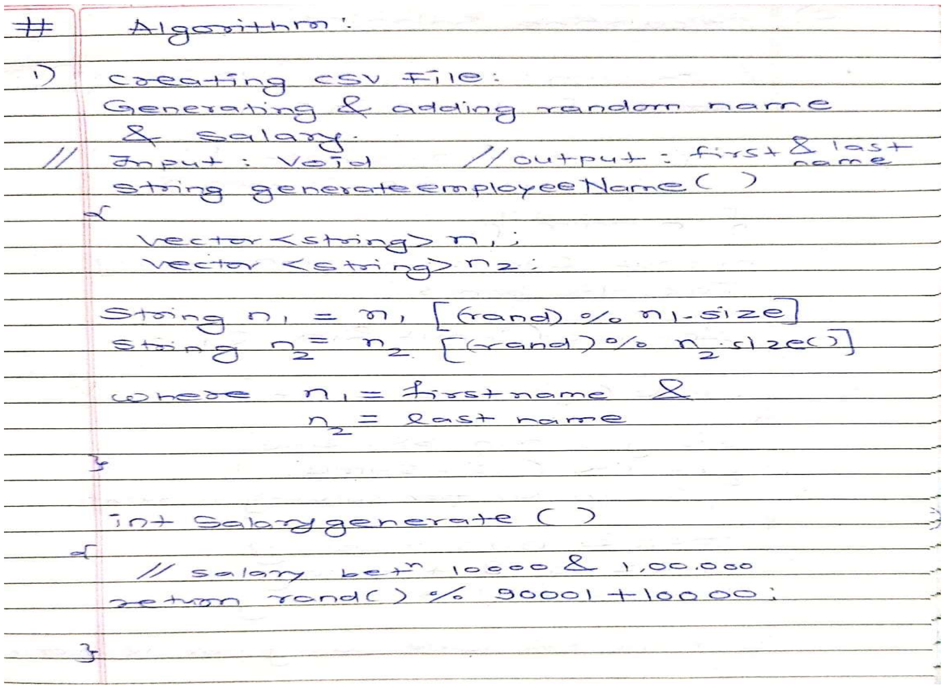
ABC co. ltd. has 2000 employees.

your task is to calculate each employees salary and find employee with minimum salary and maximum salary.

Do the above task using divide and conquer technique.

Find the improvement in the complexity using divide and conquer method.

## ALGORITHM:



```
# Algorithm:

1) creating csv File:
   Generating & adding random name
   & salary.
// Input : Void          // output : first & last name
   string generateemployeeName( )
   {
      vector<string> n1;
      vector <string> n2;

   String n1 = n1 [(rand) % n1.size]
   String n2 = n2 [(rand) % n2.size()]

   where   n1 = firstname &
           n2 = last name

   }

   int Salarygenerate ( )
   {
      // salary betn 10000 & 1,00,000
   return rand( ) % 90001 + 10000;

   }
```

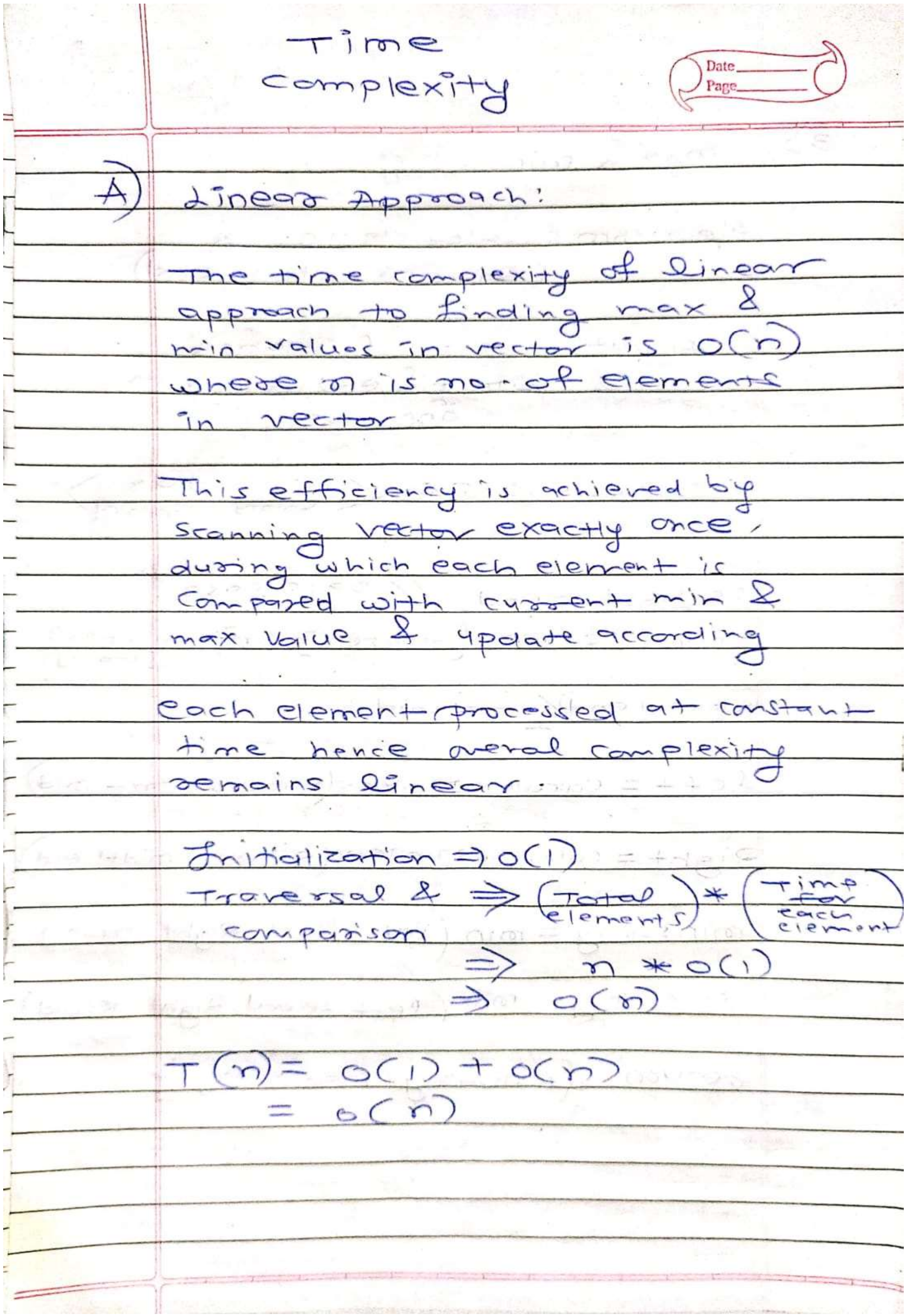
2) Calculating Salaries :

```
// To calculate Taxes, HR ( House rent )
bonus & store in CSV

if ( ! inputfile.is-open ||
     ! output file.is-open )
{ error }
output file = << "name, Salary,
Tax, HR, bonus"
getline ( input file, line)

while ( getline( input file, line))
{
 string stream ss (line)
 getline (ss, name, )
   // similar for solving as name

 tax = 0.15 * Salary
 HR = 0.18 * salary
 bonus = 0.11 * salary

outfile << name, Salary, tax, HR, bonus

}
```

3) max & min salary

```
algorithm : calculateMinMax
            (Salaries , start, end)

// Input : vector of salaries .
           start & end pointer to
           Input vector.

// output : Pair < min salary - max salary >

if start==end   // Base case
then return {salaries [start], salaries [start]}
(start + end)/2 ← mid
left = calculateMinMax(salaries , start , mid)
Right = calculateMinMax(salaries, mid+1, end)
minsalary = min (left.first , Right.first)
maxSalary = max (left.second , Right.second)
return {minsalary, maxsalary}
```

## TIME COMPLEXITY:



Time complexity

Date

Page

A) Linear Approach:

The time complexity of linear approach to finding max & min values in vector is O(n) where n is no. of elements in vector

This efficiency is achieved by scanning vector exactly once, during which each element is compared with current min & max value & update according

each element processed at constant time hence overal complexity remains linear.

$$\text{Initialization} \Rightarrow O(1)$$

$$\text{Traversal \& comparison} \Rightarrow \left(\text{Total elements}\right) * \left(\text{Time for each element}\right)$$

$$\Rightarrow n * O(1)$$

$$\Rightarrow O(n)$$

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

$$= O(n)$$

B) Divide & conquer method:

CalculateMinMax function Breakdown into 2 parts:

→ Base case : start == end
it means there is only one element & fun return single element as both max and min.
Time for operation ⇒ O(1)

→ Recursive case : for range [start, end] with more than 1 element
fun Divides range into Two halves & recursively find min & max for each half

Let, T(n) as Time complexity

① Divide step : In this step, range is Divided into 2 halves of n/2
This Division takes O(1) time

② Recursive calls : fun make 2 Recursive calls for each $n/2$ elements

③ Combine step : In this step, we are combining result by min & max of 2 pairs
This operation also takes O(1) time.

The Recurrence relation is

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

Using Master theorem,

given relⁿ in typical form
$aT\left(\frac{n}{b}\right) + f(n)$

Here, $a = 2$, $b = 2$, $f(n) = O(1)$

compare $f(n)$ with $n^{\log_b a}$

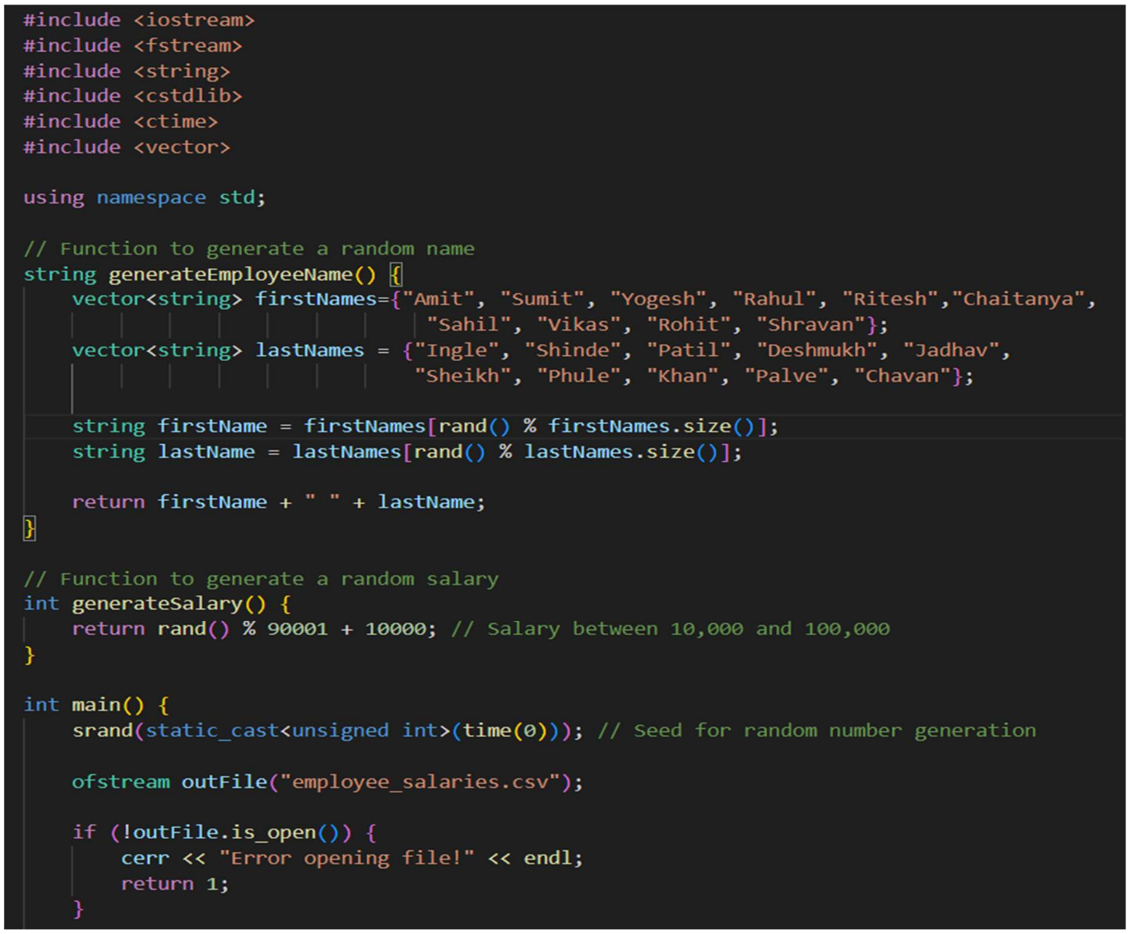
$\therefore \log_b a = \log_2 2 = 1$

Since $f(n) = O(1)$ which is $O(n^0)$
The master theorem tells that

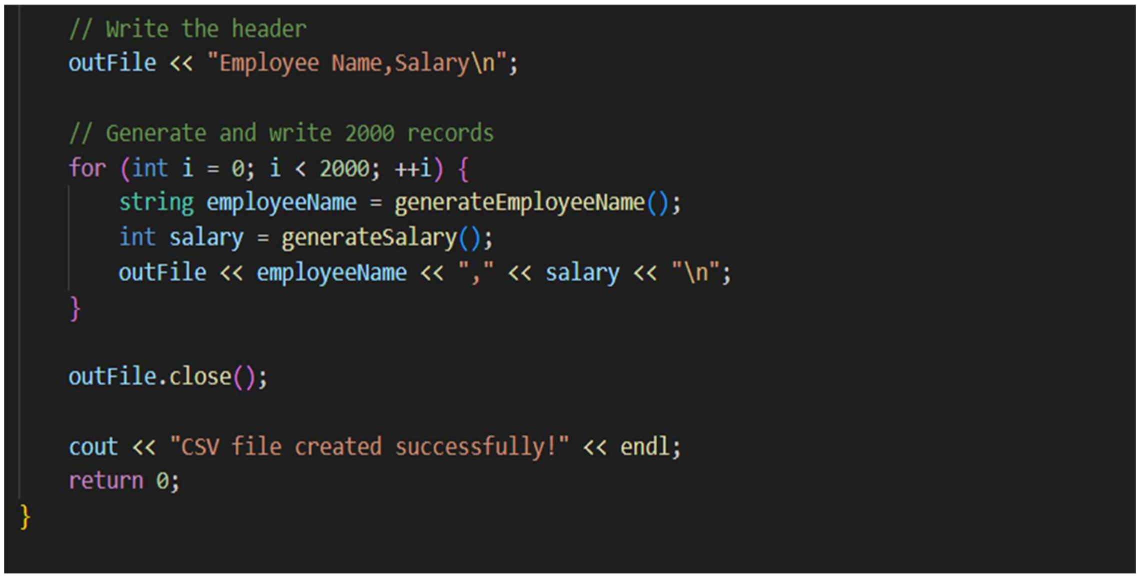
$T(n) = O(n^{\log_b a})$
$= O(n^1)$
$= O(n)$

CODE:

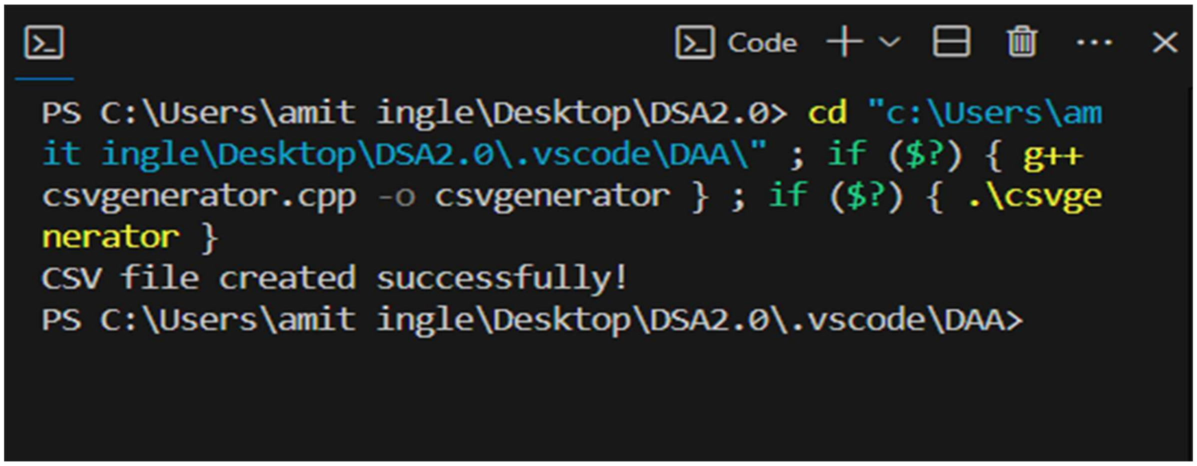
## 1)CSV FILE GENERATOR:



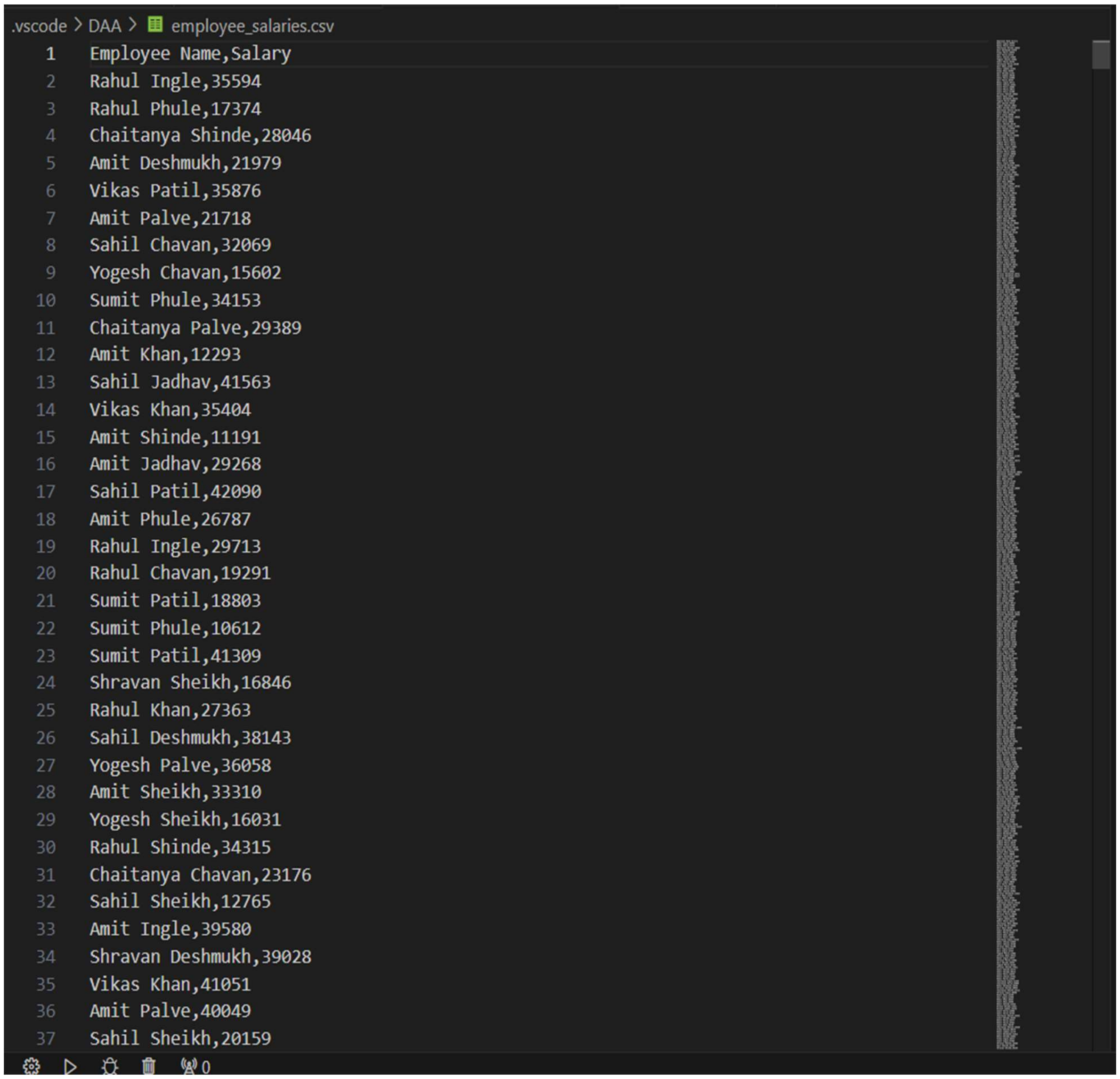
```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <cstdlib>
#include <ctime>
#include <vector>

using namespace std;

// Function to generate a random name
string generateEmployeeName() {
    vector<string> firstNames={"Amit", "Sumit", "Yogesh", "Rahul", "Ritesh","Chaitanya",
                               "Sahil", "Vikas", "Rohit", "Shravan"};
    vector<string> lastNames = {"Ingle", "Shinde", "Patil", "Deshmukh", "Jadhav",
                               "Sheikh", "Phule", "Khan", "Palve", "Chavan"};

    string firstName = firstNames[rand() % firstNames.size()];
    string lastName = lastNames[rand() % lastNames.size()];

    return firstName + " " + lastName;
}

// Function to generate a random salary
int generateSalary() {
    return rand() % 90001 + 10000; // Salary between 10,000 and 100,000
}

int main() {
    srand(static_cast<unsigned int>(time(0))); // Seed for random number generation

    ofstream outFile("employee_salaries.csv");

    if (!outFile.is_open()) {
        cerr << "Error opening file!" << endl;
        return 1;
    }
```



```cpp
    // Write the header
    outFile << "Employee Name,Salary\n";

    // Generate and write 2000 records
    for (int i = 0; i < 2000; ++i) {
        string employeeName = generateEmployeeName();
        int salary = generateSalary();
        outFile << employeeName << "," << salary << "\n";
    }

    outFile.close();

    cout << "CSV file created successfully!" << endl;
    return 0;
}
```



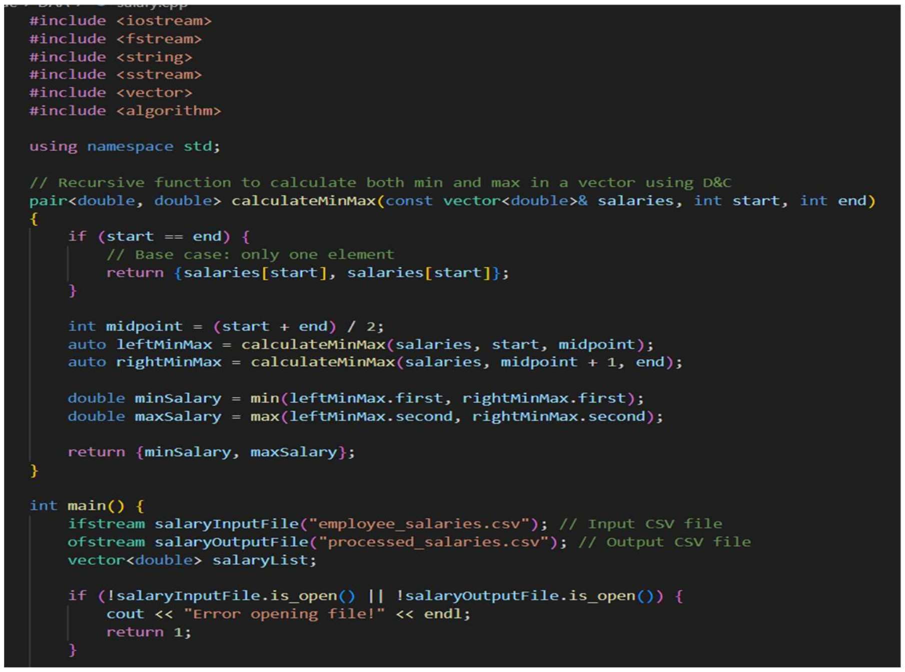
```
PS C:\Users\amit ingle\Desktop\DSA2.0> cd "c:\Users\amit ingle\Desktop\DSA2.0\.vscode\DAA\" ; if ($?) { g++ csvgenerator.cpp -o csvgenerator } ; if ($?) { .\csvgenerator }
CSV file created successfully!
PS C:\Users\amit ingle\Desktop\DSA2.0\.vscode\DAA>
```



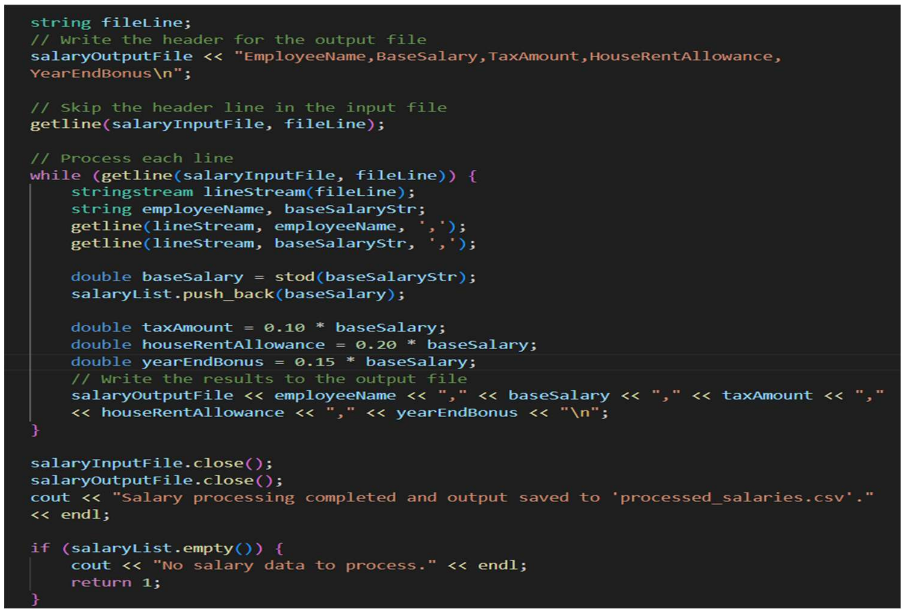
.vscode > DAA > employee_salaries.csv

```
Employee Name,Salary
Rahul Ingle,35594
Rahul Phule,17374
Chaitanya Shinde,28046
Amit Deshmukh,21979
Vikas Patil,35876
Amit Palve,21718
Sahil Chavan,32069
Yogesh Chavan,15602
Sumit Phule,34153
Chaitanya Palve,29389
Amit Khan,12293
Sahil Jadhav,41563
Vikas Khan,35404
Amit Shinde,11191
Amit Jadhav,29268
Sahil Patil,42090
Amit Phule,26787
Rahul Ingle,29713
Rahul Chavan,19291
Sumit Patil,18803
Sumit Phule,10612
Sumit Patil,41309
Shravan Sheikh,16846
Rahul Khan,27363
Sahil Deshmukh,38143
Yogesh Palve,36058
Amit Sheikh,33310
Yogesh Sheikh,16031
Rahul Shinde,34315
Chaitanya Chavan,23176
Sahil Sheikh,12765
Amit Ingle,39580
Shravan Deshmukh,39028
Vikas Khan,41051
Amit Palve,40049
Sahil Sheikh,20159
```

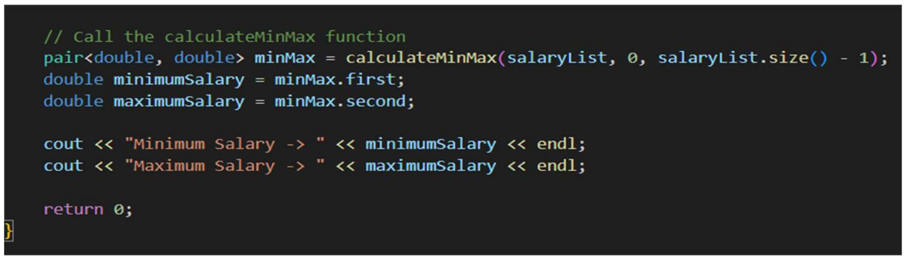
## 2)salary calculator:



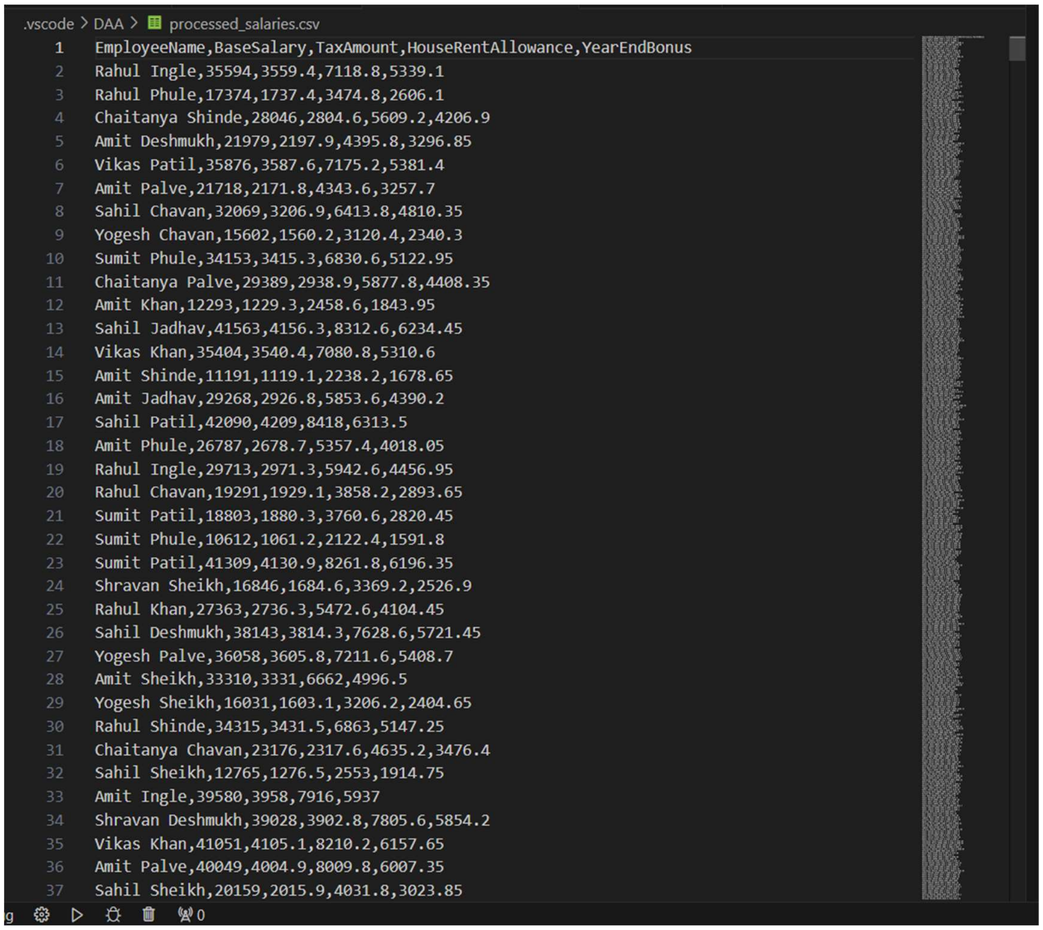
```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <sstream>
#include <vector>
#include <algorithm>

using namespace std;

// Recursive function to calculate both min and max in a vector using D&C
pair<double, double> calculateMinMax(const vector<double>& salaries, int start, int end)
{
    if (start == end) {
        // Base case: only one element
        return {salaries[start], salaries[start]};
    }

    int midpoint = (start + end) / 2;
    auto leftMinMax = calculateMinMax(salaries, start, midpoint);
    auto rightMinMax = calculateMinMax(salaries, midpoint + 1, end);

    double minSalary = min(leftMinMax.first, rightMinMax.first);
    double maxSalary = max(leftMinMax.second, rightMinMax.second);

    return {minSalary, maxSalary};
}

int main() {
    ifstream salaryInputFile("employee_salaries.csv"); // Input CSV file
    ofstream salaryOutputFile("processed_salaries.csv"); // Output CSV file
    vector<double> salaryList;

    if (!salaryInputFile.is_open() || !salaryOutputFile.is_open()) {
        cout << "Error opening file!" << endl;
        return 1;
    }
```



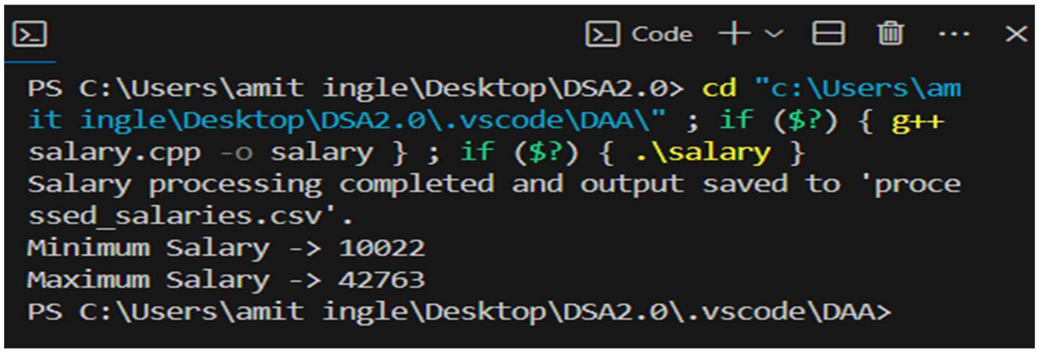
```cpp
string fileLine;
// Write the header for the output file
salaryOutputFile << "EmployeeName,BaseSalary,TaxAmount,HouseRentAllowance,
YearEndBonus\n";

// Skip the header line in the input file
getline(salaryInputFile, fileLine);

// Process each line
while (getline(salaryInputFile, fileLine)) {
    stringstream lineStream(fileLine);
    string employeeName, baseSalaryStr;
    getline(lineStream, employeeName, ',');
    getline(lineStream, baseSalaryStr, ',');

    double baseSalary = stod(baseSalaryStr);
    salaryList.push_back(baseSalary);

    double taxAmount = 0.10 * baseSalary;
    double houseRentAllowance = 0.20 * baseSalary;
    double yearEndBonus = 0.15 * baseSalary;
    // Write the results to the output file
    salaryOutputFile << employeeName << "," << baseSalary << "," << taxAmount << ","
    << houseRentAllowance << "," << yearEndBonus << "\n";
}

salaryInputFile.close();
salaryOutputFile.close();
cout << "Salary processing completed and output saved to 'processed_salaries.csv'."
<< endl;

if (salaryList.empty()) {
    cout << "No salary data to process." << endl;
    return 1;
}
```



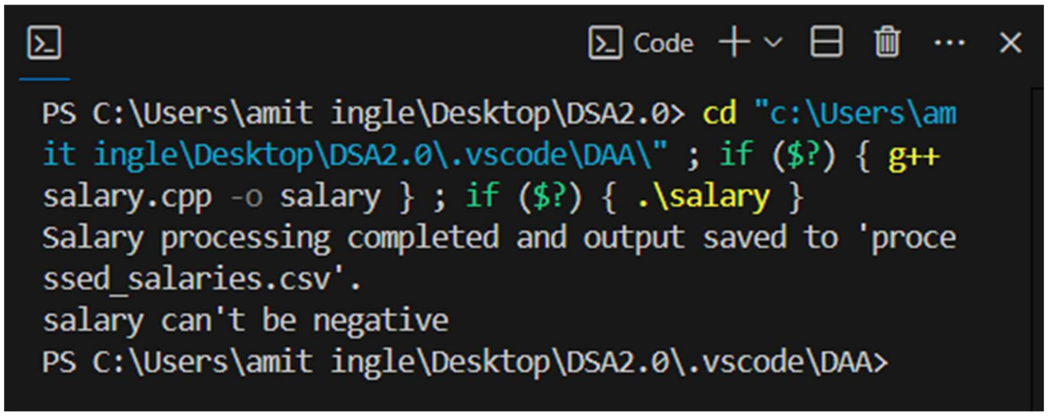
```cpp
    // Call the calculateMinMax function
    pair<double, double> minMax = calculateMinMax(salaryList, 0, salaryList.size() - 1);
    double minimumSalary = minMax.first;
    double maximumSalary = minMax.second;

    cout << "Minimum Salary -> " << minimumSalary << endl;
    cout << "Maximum Salary -> " << maximumSalary << endl;

    return 0;
}
```



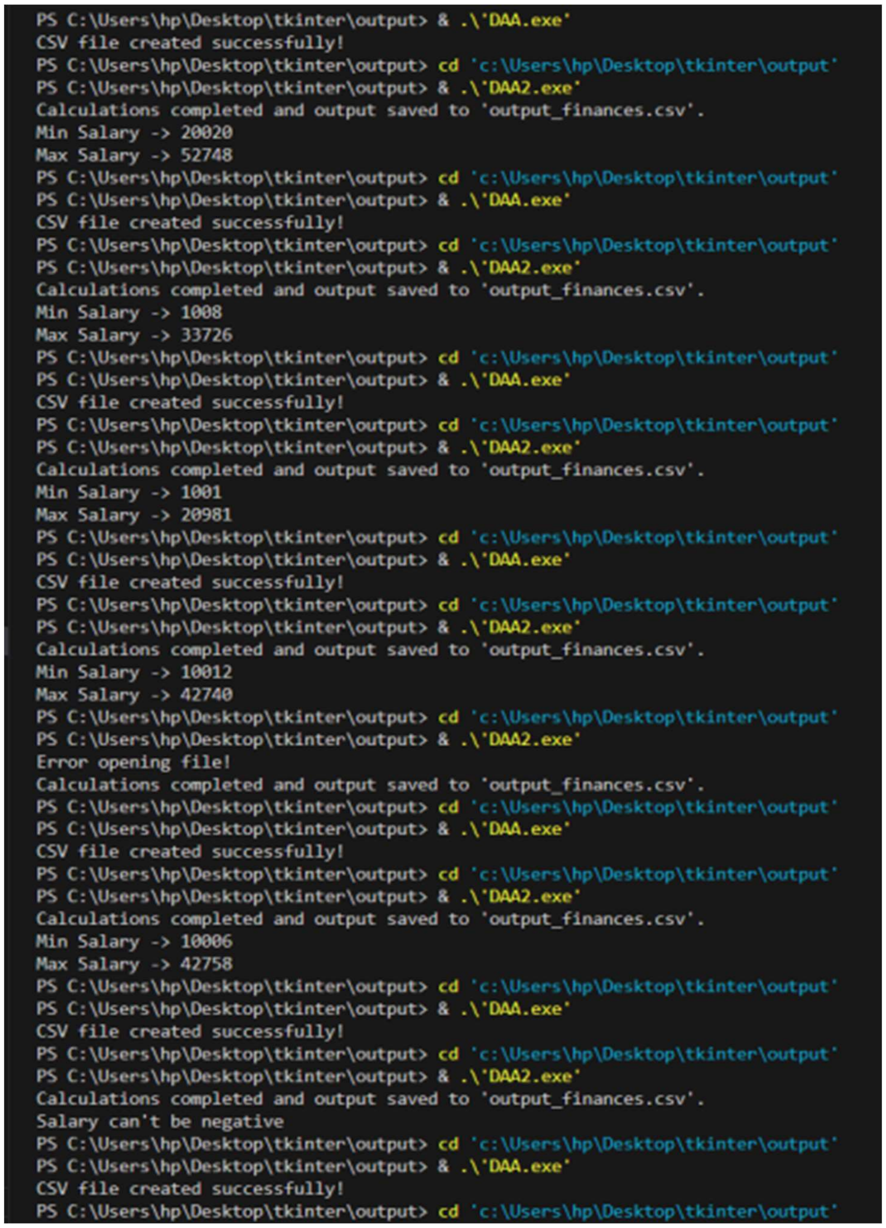
```
EmployeeName,BaseSalary,TaxAmount,HouseRentAllowance,YearEndBonus
Rahul Ingle,35594,3559.4,7118.8,5339.1
Rahul Phule,17374,1737.4,3474.8,2606.1
Chaitanya Shinde,28046,2804.6,5609.2,4206.9
Amit Deshmukh,21979,2197.9,4395.8,3296.85
Vikas Patil,35876,3587.6,7175.2,5381.4
Amit Palve,21718,2171.8,4343.6,3257.7
Sahil Chavan,32069,3206.9,6413.8,4810.35
Yogesh Chavan,15602,1560.2,3120.4,2340.3
Sumit Phule,34153,3415.3,6830.6,5122.95
Chaitanya Palve,29389,2938.9,5877.8,4408.35
Amit Khan,12293,1229.3,2458.6,1843.95
Sahil Jadhav,41563,4156.3,8312.6,6234.45
Vikas Khan,35404,3540.4,7080.8,5310.6
Amit Shinde,11191,1119.1,2238.2,1678.65
Amit Jadhav,29268,2926.8,5853.6,4390.2
Sahil Patil,42090,4209,8418,6313.5
Amit Phule,26787,2678.7,5357.4,4018.05
Rahul Ingle,29713,2971.3,5942.6,4456.95
Rahul Chavan,19291,1929.1,3858.2,2893.65
Sumit Patil,18803,1880.3,3760.6,2820.45
Sumit Phule,10612,1061.2,2122.4,1591.8
Sumit Patil,41309,4130.9,8261.8,6196.35
Shravan Sheikh,16846,1684.6,3369.2,2526.9
Rahul Khan,27363,2736.3,5472.6,4104.45
Sahil Deshmukh,38143,3814.3,7628.6,5721.45
Yogesh Palve,36058,3605.8,7211.6,5408.7
Amit Sheikh,33310,3331,6662,4996.5
Yogesh Sheikh,16031,1603.1,3206.2,2404.65
Rahul Shinde,34315,3431.5,6863,5147.25
Chaitanya Chavan,23176,2317.6,4635.2,3476.4
Sahil Sheikh,12765,1276.5,2553,1914.75
Amit Ingle,39580,3958,7916,5937
Shravan Deshmukh,39028,3902.8,7805.6,5854.2
Vikas Khan,41051,4105.1,8210.2,6157.65
Amit Palve,40049,4004.9,8009.8,6007.35
Sahil Sheikh,20159,2015.9,4031.8,3023.85
```



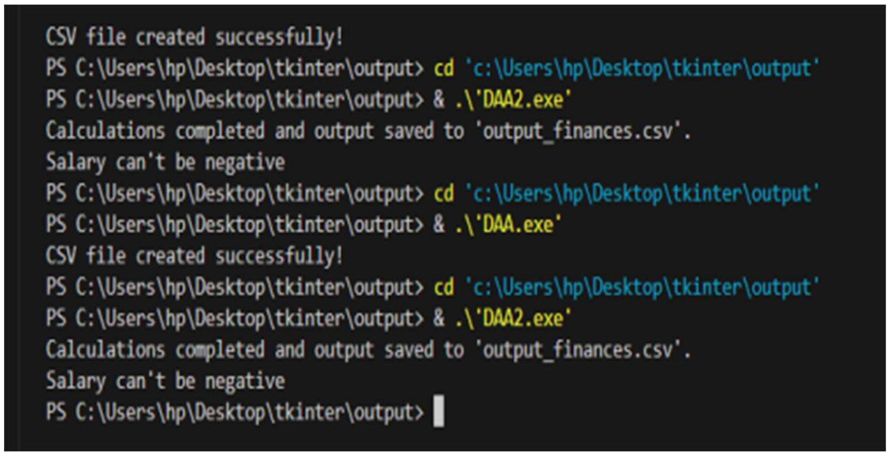
```
PS C:\Users\amit ingle\Desktop\DSA2.0> cd "c:\Users\am
it ingle\Desktop\DSA2.0\.vscode\DAA\" ; if ($?) { g++
salary.cpp -o salary } ; if ($?) { .\salary }
Salary processing completed and output saved to 'proce
ssed_salaries.csv'.
Minimum Salary -> 10022
Maximum Salary -> 42763
PS C:\Users\amit ingle\Desktop\DSA2.0\.vscode\DAA>
```



```
PS C:\Users\amit ingle\Desktop\DSA2.0> cd "c:\Users\am
it ingle\Desktop\DSA2.0\.vscode\DAA\" ; if ($?) { g++
salary.cpp -o salary } ; if ($?) { .\salary }
Salary processing completed and output saved to 'proce
ssed_salaries.csv'.
salary can't be negative
PS C:\Users\amit ingle\Desktop\DSA2.0\.vscode\DAA>
```

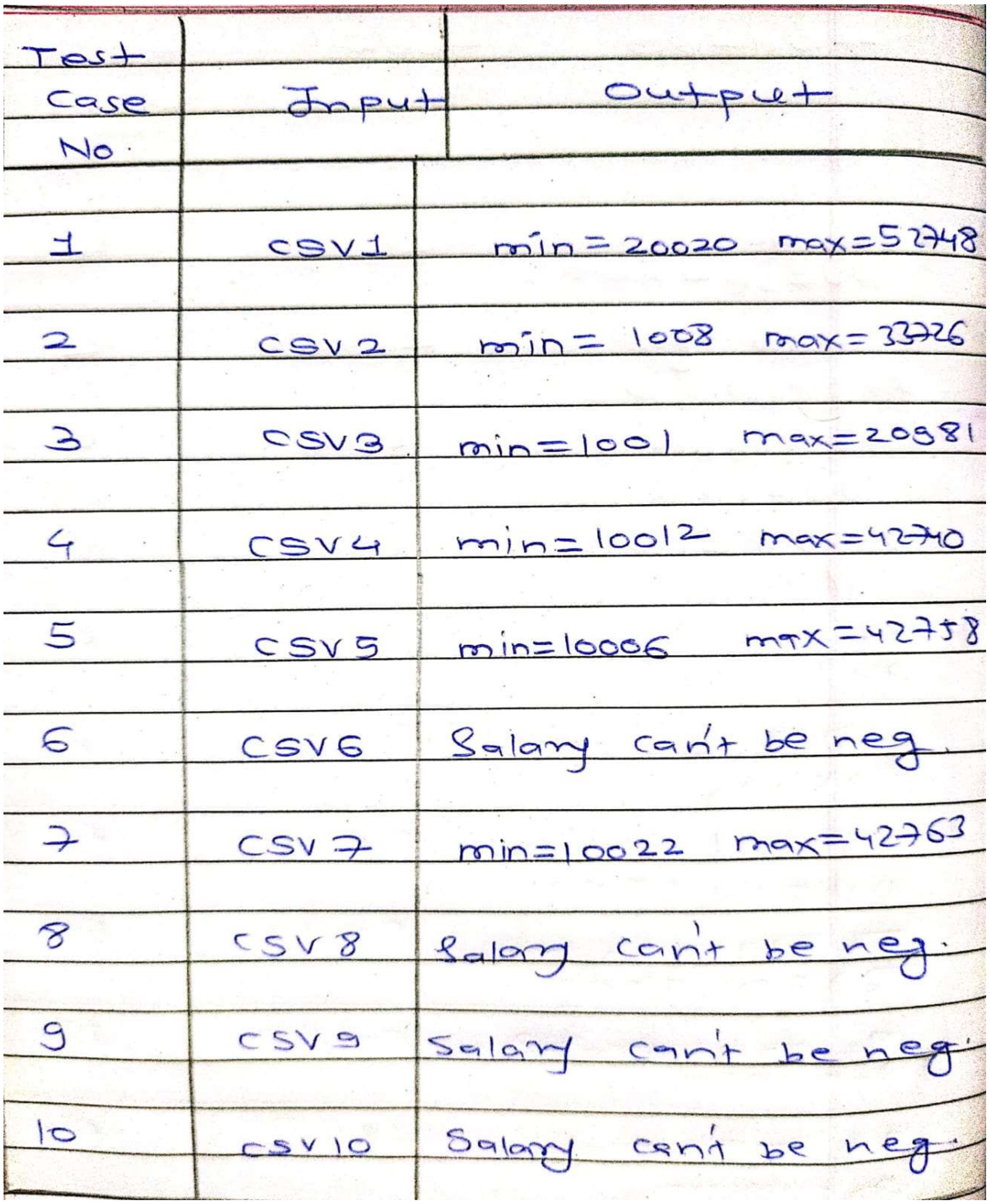


```
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA.exe'
CSV file created successfully!
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA2.exe'
Calculations completed and output saved to 'output_finances.csv'.
Min Salary -> 20020
Max Salary -> 52748
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA.exe'
CSV file created successfully!
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA2.exe'
Calculations completed and output saved to 'output_finances.csv'.
Min Salary -> 1008
Max Salary -> 33726
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA.exe'
CSV file created successfully!
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA2.exe'
Calculations completed and output saved to 'output_finances.csv'.
Min Salary -> 1001
Max Salary -> 20981
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA.exe'
CSV file created successfully!
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA2.exe'
Calculations completed and output saved to 'output_finances.csv'.
Min Salary -> 10012
Max Salary -> 42740
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA2.exe'
Error opening file!
Calculations completed and output saved to 'output_finances.csv'.
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA.exe'
CSV file created successfully!
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA2.exe'
Calculations completed and output saved to 'output_finances.csv'.
Min Salary -> 10006
Max Salary -> 42758
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA.exe'
CSV file created successfully!
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA2.exe'
Calculations completed and output saved to 'output_finances.csv'.
Salary can't be negative
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA.exe'
CSV file created successfully!
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
```



```
CSV file created successfully!
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA2.exe'
Calculations completed and output saved to 'output_finances.csv'.
Salary can't be negative
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA.exe'
CSV file created successfully!
PS C:\Users\hp\Desktop\tkinter\output> cd 'c:\Users\hp\Desktop\tkinter\output'
PS C:\Users\hp\Desktop\tkinter\output> & .\'DAA2.exe'
Calculations completed and output saved to 'output_finances.csv'.
Salary can't be negative
PS C:\Users\hp\Desktop\tkinter\output> 
```

TEST CASES:



| Test Case No. | Input | Output |
|---|---|---|
| 1 | CSV1 | min = 20020 max = 52748 |
| 2 | CSV2 | min = 1008 max = 33726 |
| 3 | CSV3 | min = 1001 max = 20981 |
| 4 | CSV4 | min = 10012 max = 42740 |
| 5 | CSV5 | min = 10006 max = 42758 |
| 6 | CSV6 | Salary can't be neg. |
| 7 | CSV7 | min = 10022 max = 42763 |
| 8 | CSV8 | Salary can't be neg. |
| 9 | CSV9 | Salary can't be neg. |
| 10 | CSV10 | Salary can't be neg. |

## CALCULATION:

In this experiment, we calculated the gross and net salaries for 2,000 employees at ABC Co. Ltd., and identified those with the minimum and maximum salaries using a divide-and-conquer approach. This method divides the salary list into smaller segments, recursively computes the minimum and maximum for each segment, and then combines the results, achieving a time complexity of ( O(n) ). This is comparable to the linear scan approach but offers benefits in modularity and potential parallelization. Although both methods have the same time complexity, divide-and-conquer can be advantageous for handling larger datasets or more complex problems efficiently.

## END