William Carey

C16315253

Algorithms and problem solving Project Report

Course Code

DT-228

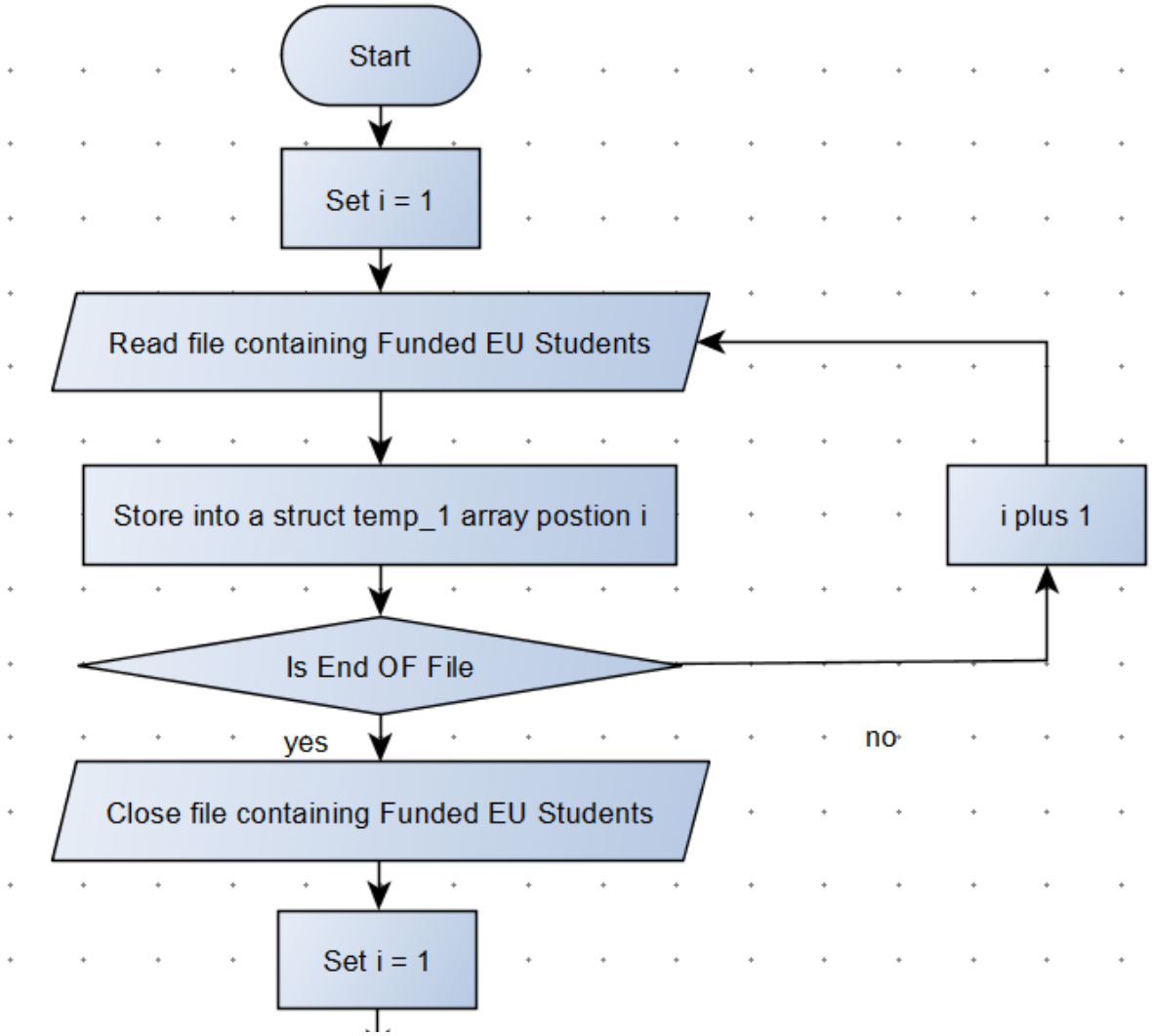
Year 1

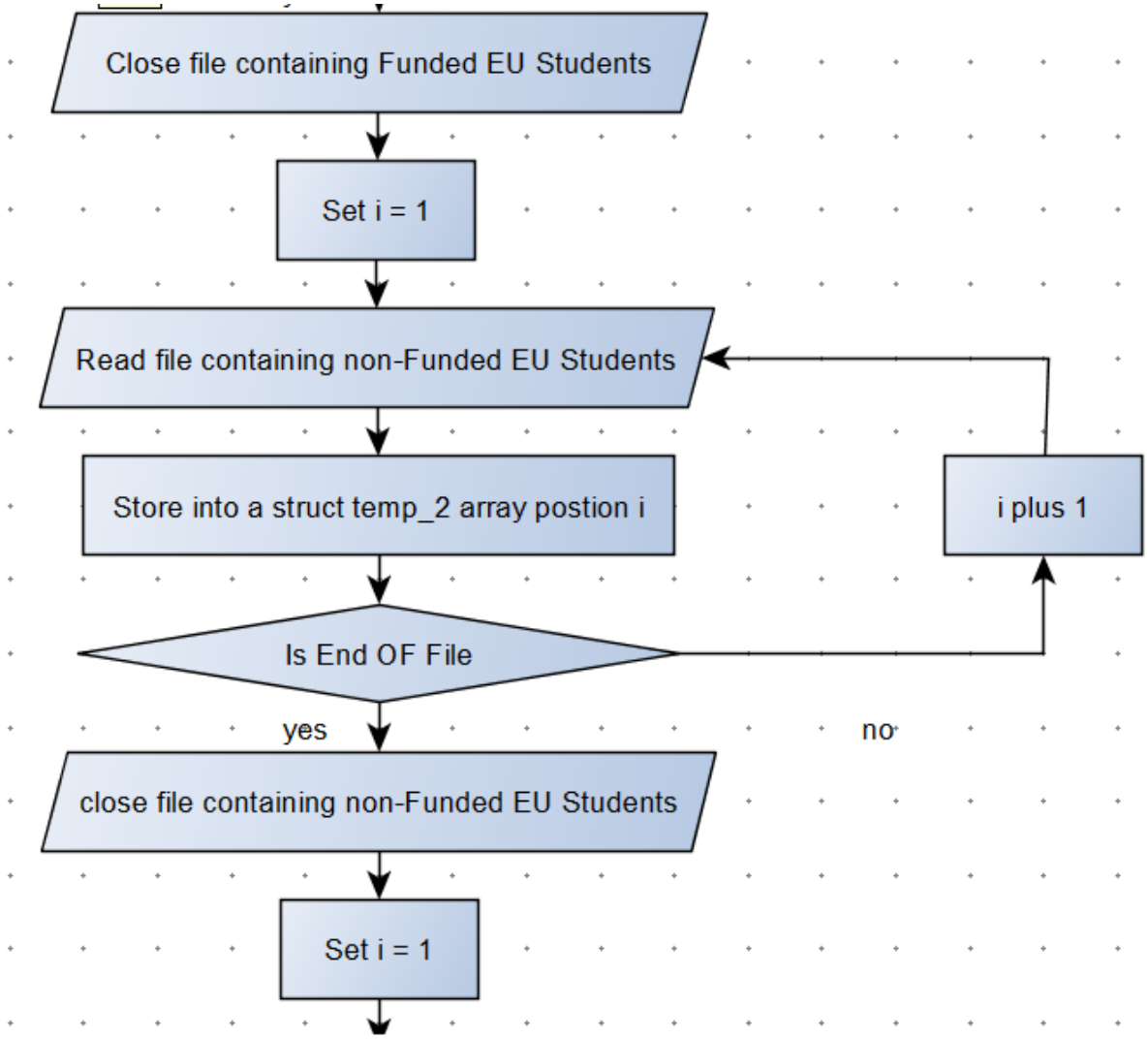
Computer Science

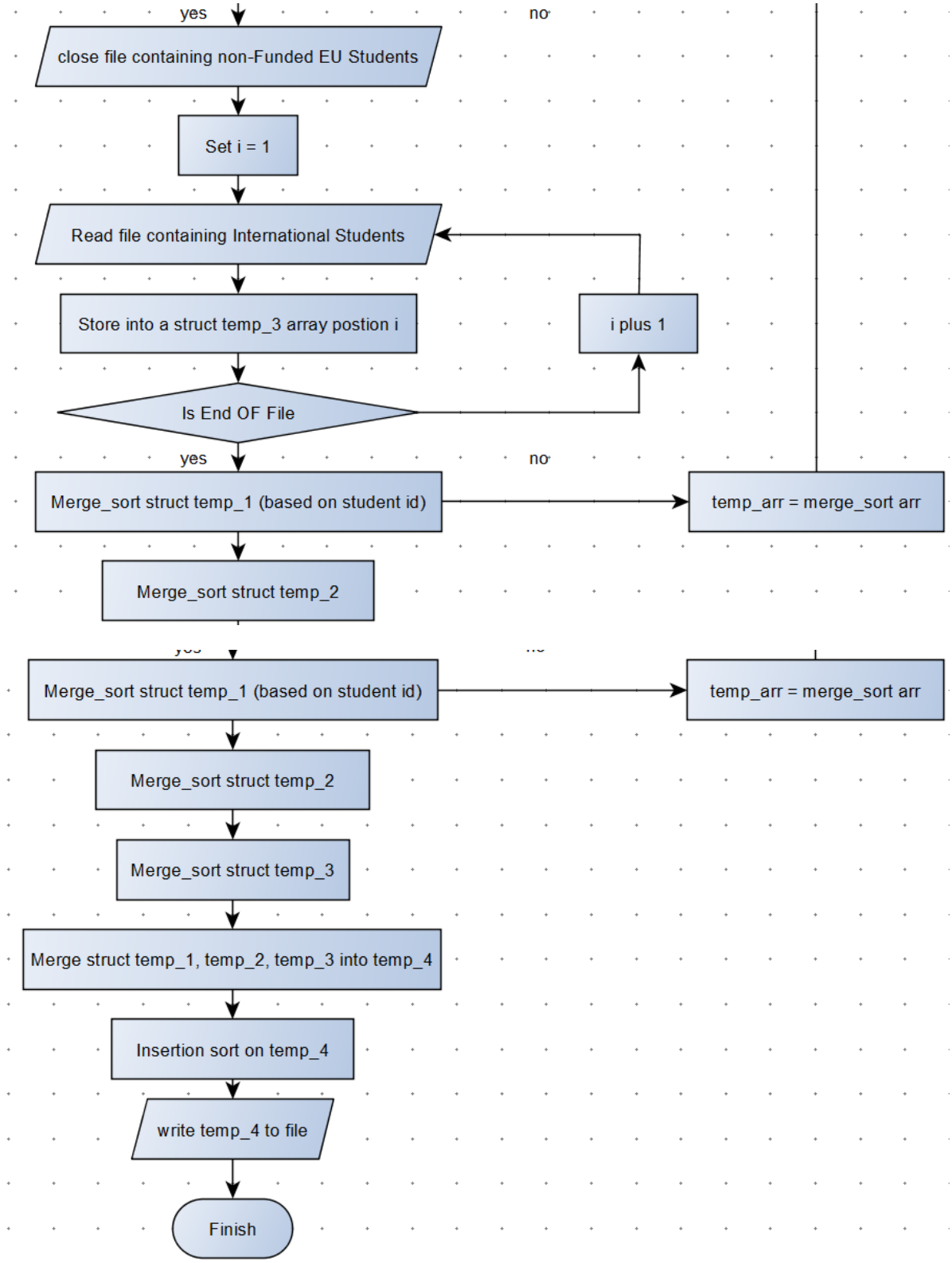
Contents

Question 1 – Flowchart that sorts out three list into one

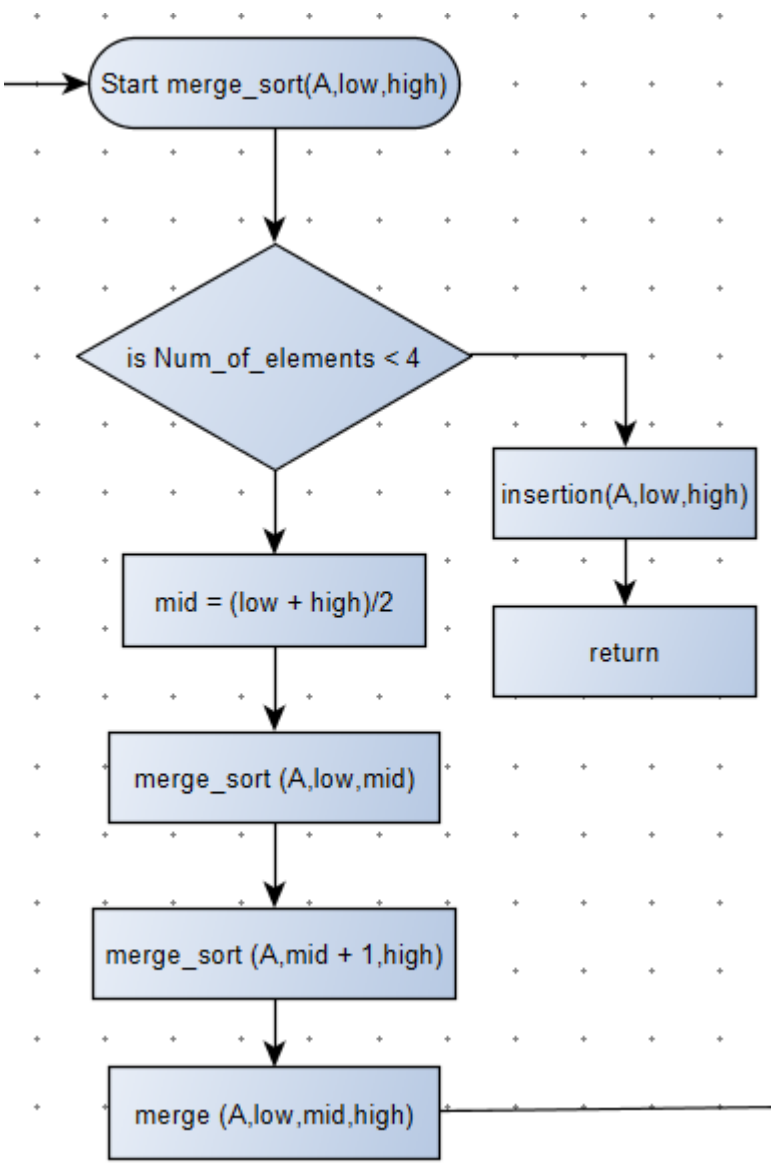
Within the merge sort, there is a insertion sort. The base case was changed to “if N < 8” to allow this change. Because of this, the big O of this algorithm is O ((N ^ 3)(log (N))) as the insertion sort has the big O of O (N ^ 2) whereas the merge sort has a worst case of O (N (log (N))). (N x N ^ 2) = (N ^ 3). This means within the algorithm there is a loop within a loop within a loop. All the students are sorted by student ID as that between all the student is the only thing that is unique between each student.

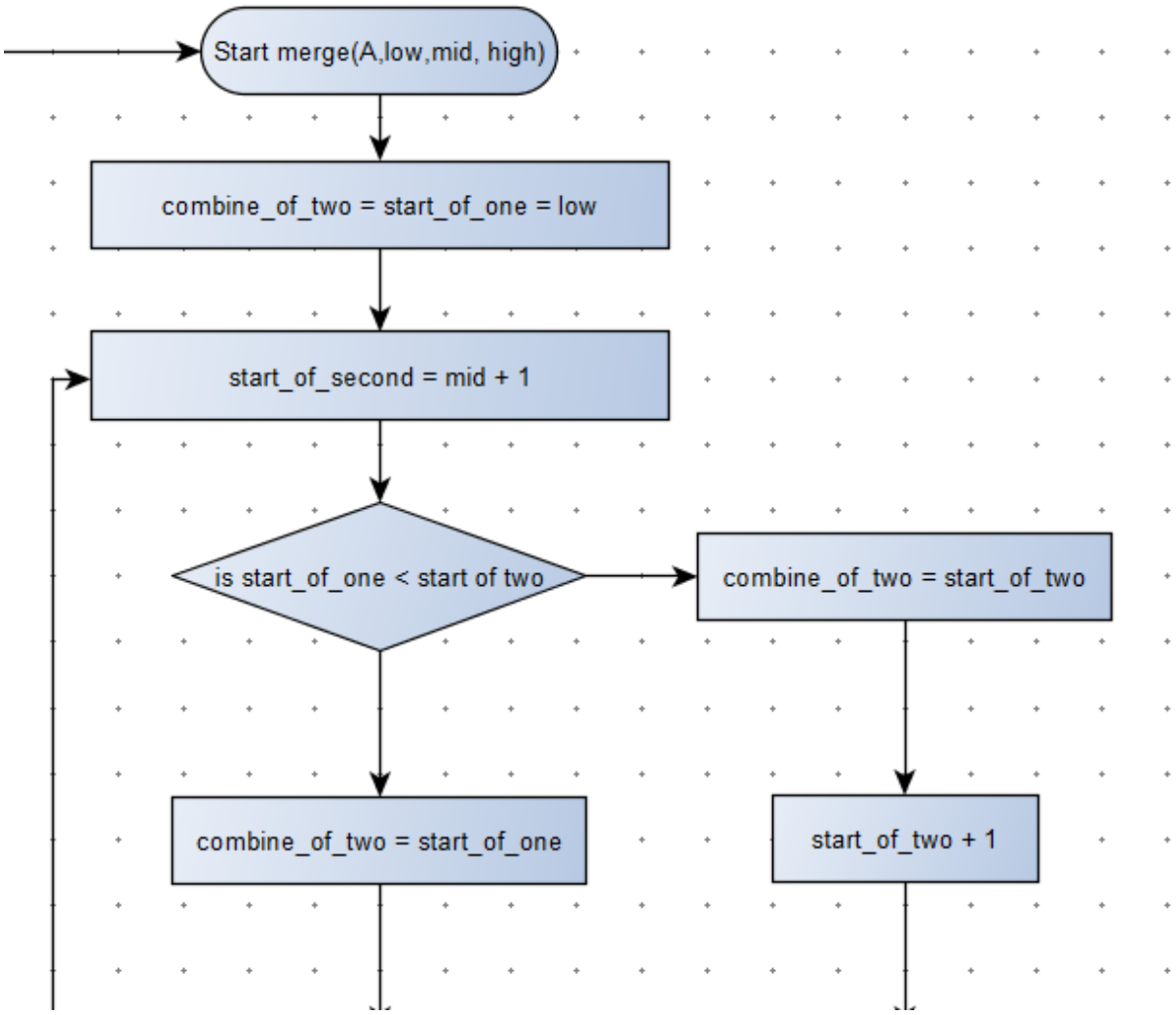


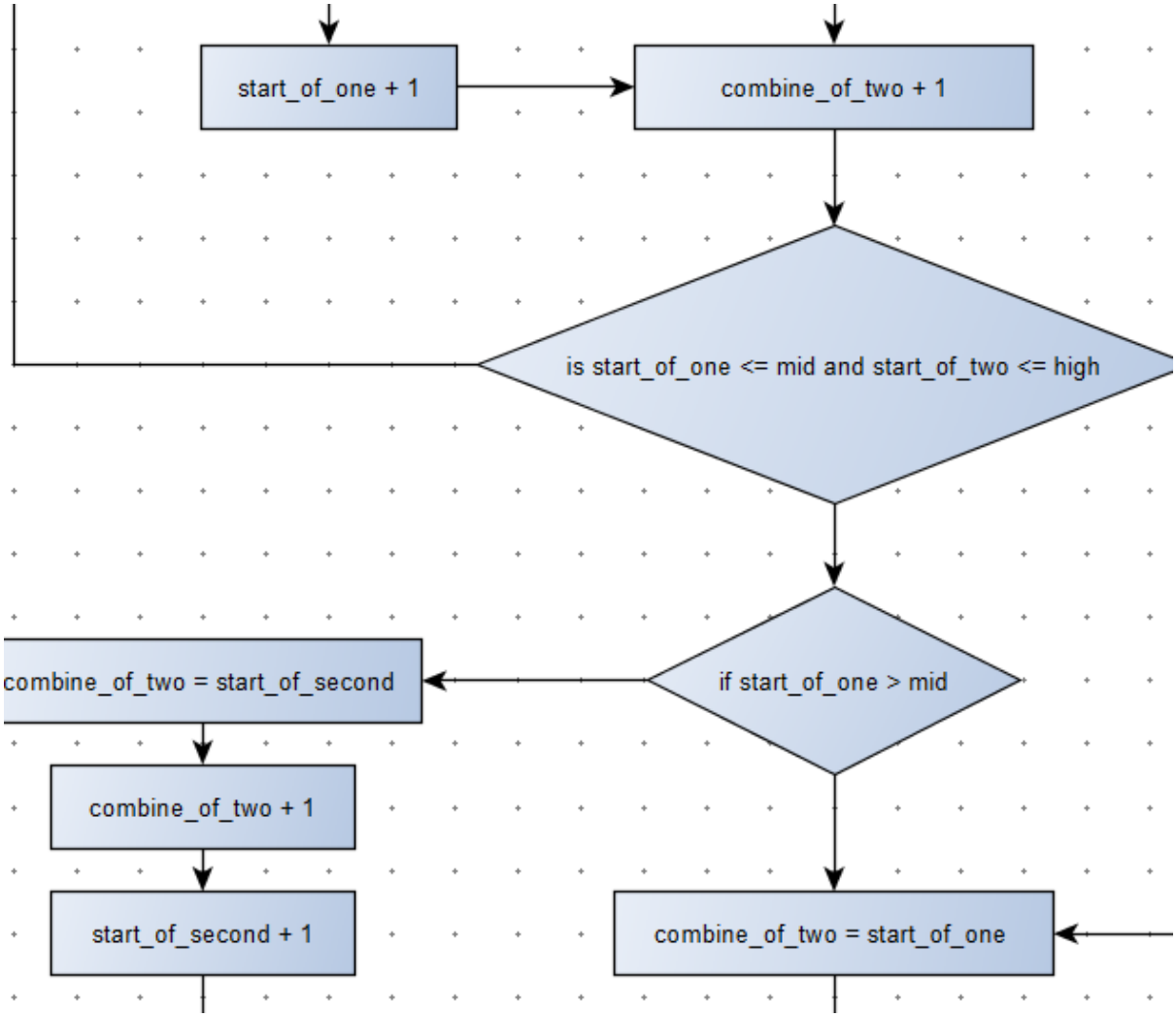

Close file containing Funded EU Students
Set i = 1
Read file containing non-Funded EU Students
Store into a struct temp_2 array postion i
i plus 1
Is End OF File
yes
no
close file containing non-Funded EU Students
Set i = 1

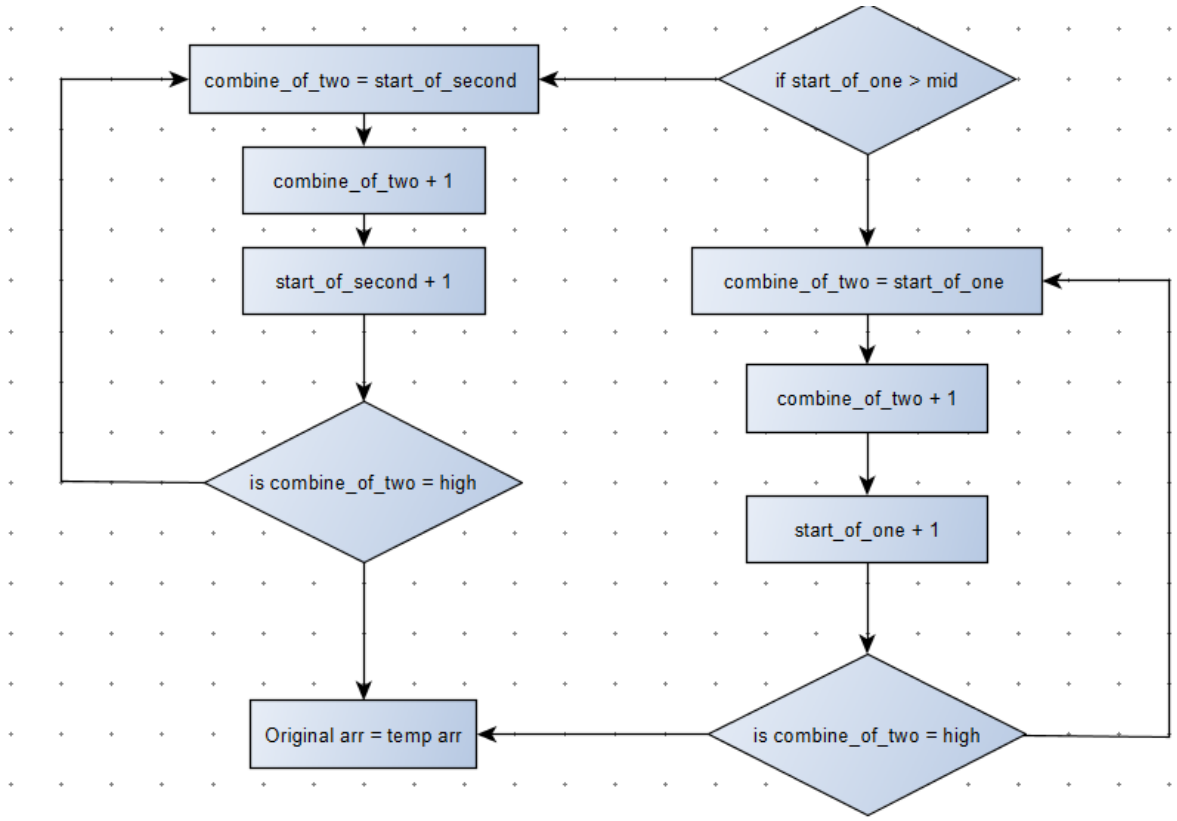


The “temp arr = merge_sort arr” points to the merge_sort algorithm


Start merge_sort(A,low,high)
is Num_of_elements < 4
insertion(A,low,high)
return
mid = (low + high)/2
merge_sort (A,low,mid)
merge_sort (A,mid + 1,high)
merge (A,low,mid,high)


Start merge(A,low,mid, high)
combine_of_two = start_of_one = low
start_of_second = mid + 1
is start_of_one < start of two
combine_of_two = start_of_two
combine_of_two = start_of_one
start_of_two + 1


start_of_one + 1
combine_of_two + 1
is start_of_one <= mid and start_of_two <= high
if start_of_one > mid
combine_of_two = start_of_second
combine_of_two + 1
start_of_second + 1
combine_of_two = start_of_one


combine_of_two = start_of_second
if start_of_one > mid
combine_of_two + 1
start_of_second + 1
combine_of_two = start_of_one
combine_of_two + 1
is combine_of_two = high
start_of_one + 1
Original arr = temp arr
is combine_of_two = high

Question 2 pseudo-code on searching for international students

```
START PROGRAMME

CREATE counter, file pointer

PROMPT for file and point file pointer to it

IF file does not exist, tell the user

ELSE
    WHILE it is not the end of file
    DO
        IF it is international student
        THEN copy it into structure array
        print the student details to the screen
        postion plus 1
        counter plus 1
    END WHILE

CLOSE file

END PROGRAMME
```

The Big O for this algorithm is O(N) as the previous algorithm sorts the students based on student ID. So in the case of searching for international, we don't actually know where each student is. We also know it is not sorted in student status, so we would have no idea where each international is.

There is one loop happening, which is why is has a Big O of N

Question 3 pseudo-code on searching for a particular student by surname

```
START PROGRAMME

CREATE counter, file pointer

PROMPT for file and point file pointer to it

IF file does not exist, tell the user

ELSE
    WHILE it is not the end of file
    DO
        IF it is the person we want
        print the student details to the screen
        end while loop

        ELSE
            postion plus 1
            counter plus 1

    END WHILE

CLOSE file

END PROGRAMME
```

The big O is O(N) as again we don't have a hundred percent certain the location of the student by surname, so we must check each student linear. There is one loop to check all the students surname.