Algorithms Assignment

Daniel Tilley

C14337041

Q1 a:

```
Start Program
    Prompt user for number of files
    Get number of files

    For i = 0, i < number of files do
        Prompt user for file name
        Get File name
        If Open file != NULL Then
            Print "File Opened Successfully"
        End If
        For j=Last_pos, j<n do
            Read Record [j]
            Convert Record.Key to upper case // comparison reason
            If Read = EOF Then
                Break
            End If
            Last_pos = j
            Call Insertion Sort Function (data, last_pos)
        End For
        Else
            Print "File Cannot Be Opened"
        End Else
        Close File
    End For
    Call Bubble Sort Function (data)
```

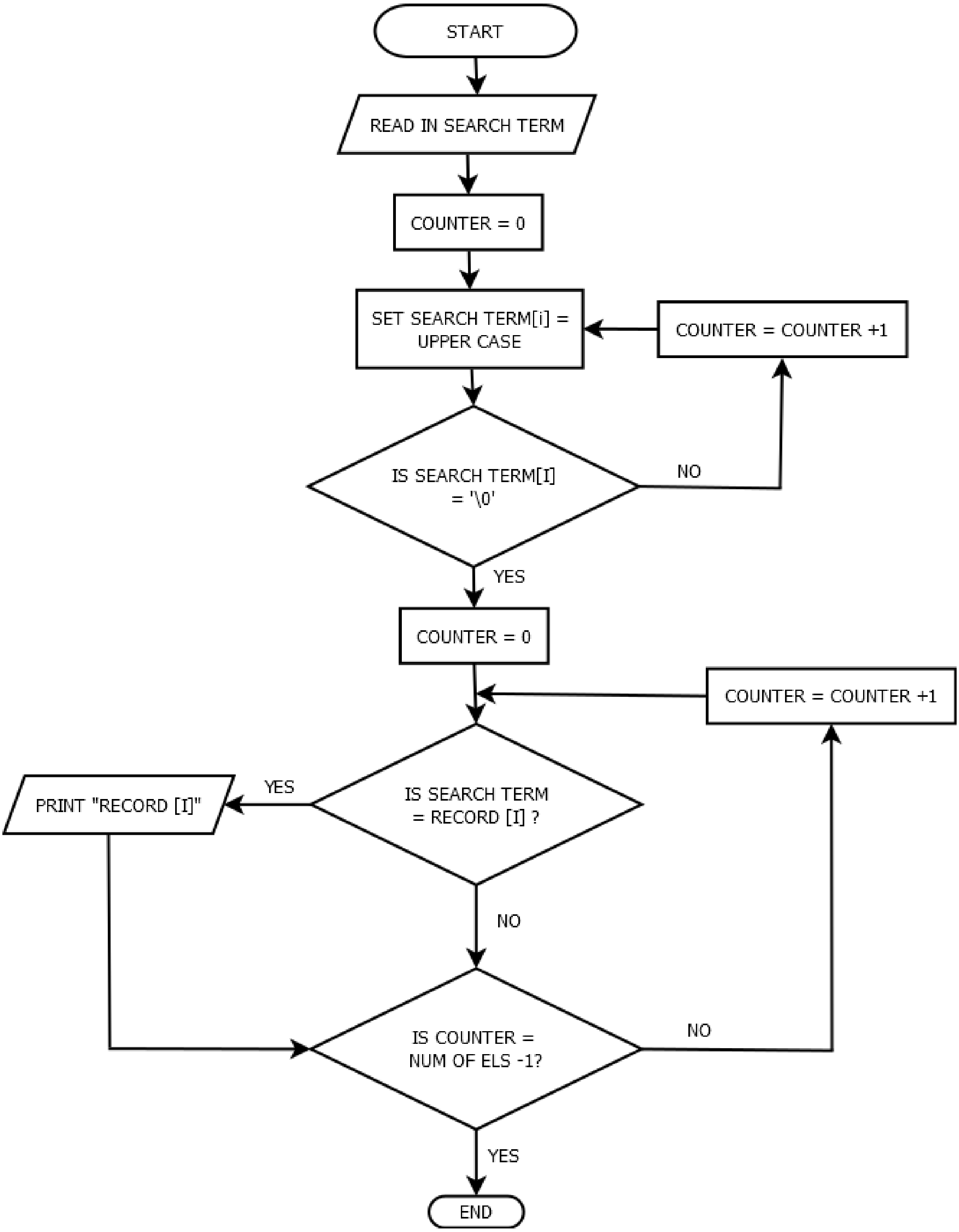
```
//Insertion sort function
For i = last_pos-10,  i < last_pos do
          Current = data[i]
           j = i
          While A[j-1] > current
                    Data[j] = data[j-1]
                     j --
          End while
          A[j] = current
End for


//Bubble sort function
While not sorted
          Flag = 1
          For j = 0 to j < N-1 do
                    If Data[j] > Data[j+1]
                              Flag = 0
                              Temp = Data[j]
                              Data[j] = Data[j+1]
                              Data[j+1] = temp
                    End if
          End for
End for
```

End Program

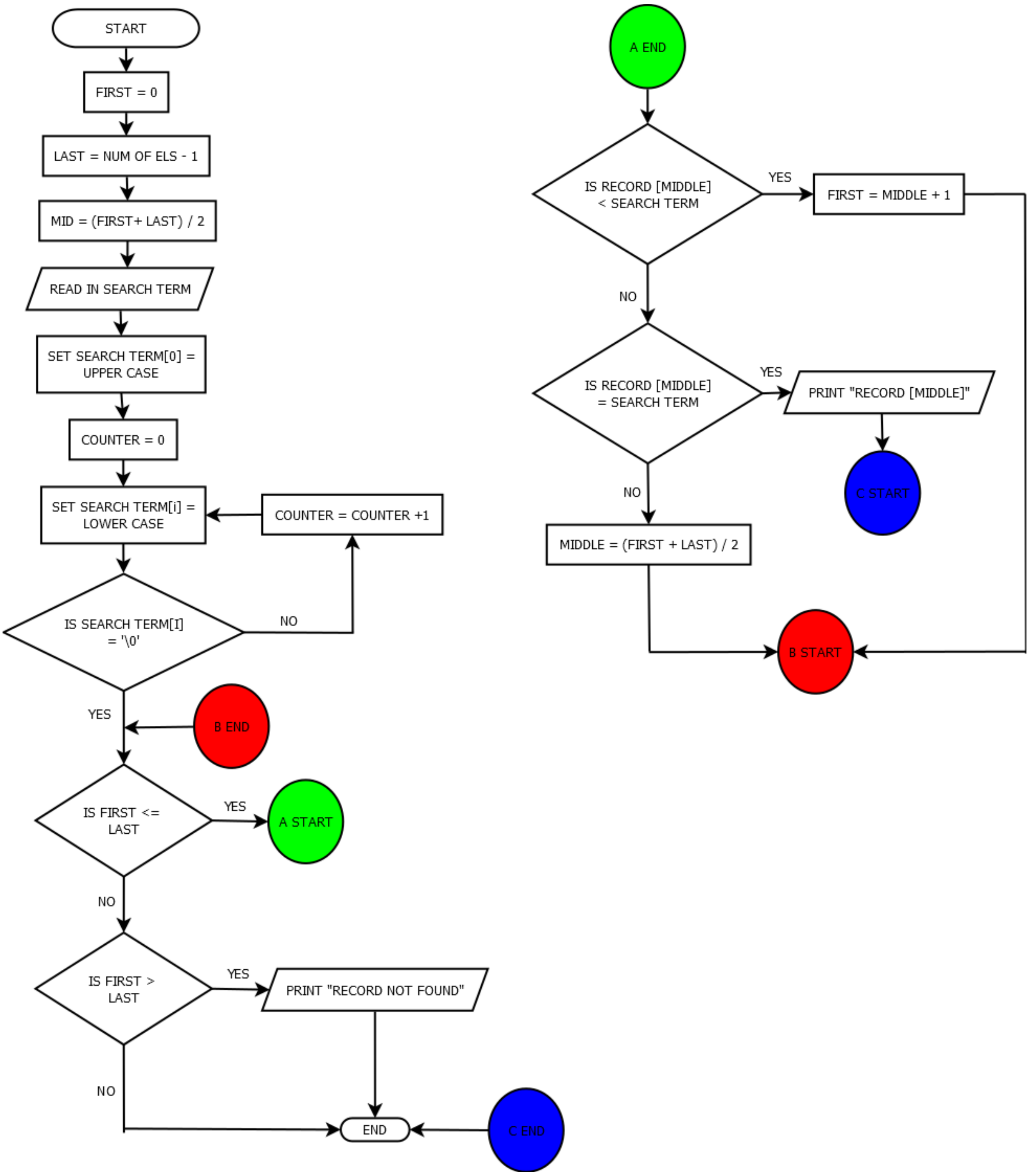
Q1 b: The big O for this algorithm can been seen in separate ways. If the data is read in is already sorted then the big O in each case of the insertion sort will be O(n). However if the data is not already pre-sorted, the the big O for each insertion sort will be O(n^2) as the data will have to be sorted. For Bubble sort, the big O will always be O(n^2) as the combined list will have to be sorted. Overall the big O will always be O(n^2) due to the fact that both algorithms use two loops to sort the data and bubble sort will always have to sort the data.

Q2 a: Flowchart used for searching colleges.



Q2 b: The big O of this algorithm in O(n) as it uses one loop to search. The algorithm converts the users search term so that it matches the term it is searching for in records. It then searches through the array one by one using a loop and prints the data one by one if a match is found.

Q3 a: Flowchart used for searching names.



Q3 b: The big O of this algorithm in O(log n) as it uses a divide and conquer approach and on each call it half's the search time taken. The algorithm uses a binary search to divide the list up and make searching for a single person's name a lot quicker.