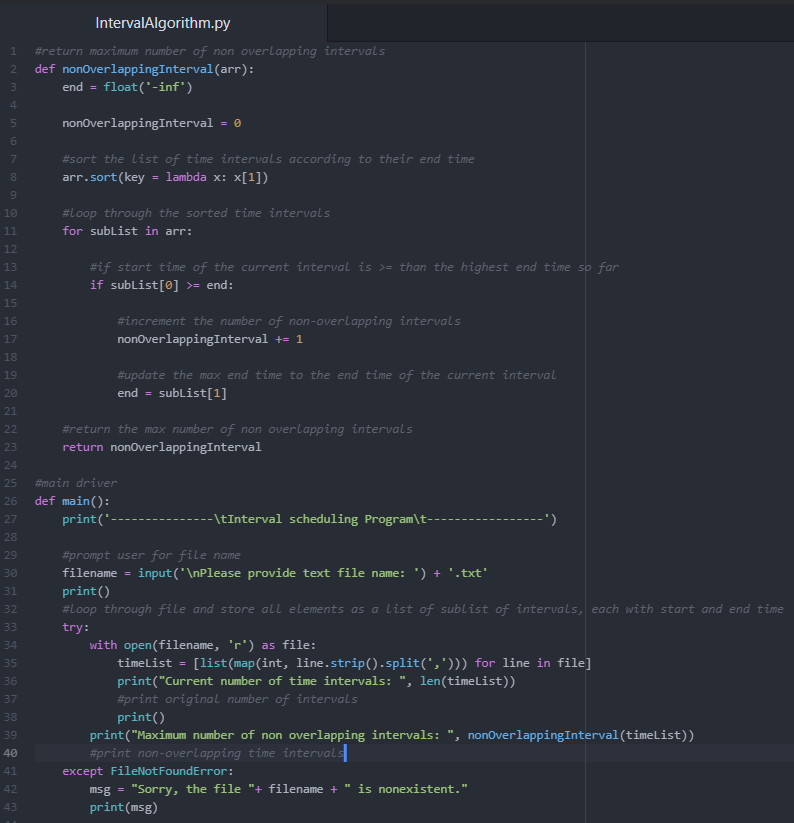
**CS3012-2 || Algorithm Analysis**

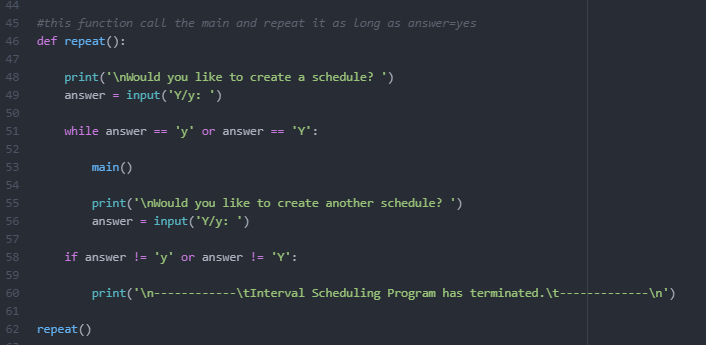
**Interval Scheduling Algorithm Assignment**

**Student name:** Razan Almahdi

**Student ID:** S20106649

**The Algorithm (Python):**





**Time Complexity:**

In the case of the algorithm, it can be broken up in the following parts:

• Reading the "input.txt" file and building the array of jobs takes **O(n)** time.

• Ordering arrays takes **O(nlogn)** time.

• Creating the array of machines and initializing it takes **O(m)** time.

• Outer loop takes **O(n)** time.

• Each step of the outer loop is an entire inner loop which takes **O(m)** time.

• Writing the "output.txt" file takes **O(n)** time.

Time complexity of the algorithm:

**O(nlogn + nm): O(n + nlogn + m + nm + n) = O(2n + m + nlogn + nm) = O(nlogn + nm)**

However, the algorithm can be also broken up in the following parts:

• Reading the "input.txt" file and building the array of jobs takes **O(n)** time.

• Ordering such array takes **O(nlogn)** time.

• Creating the array of machines and initializing it takes **O(m)** time.

• Outer loop takes **O(n)** time.

• Each step of the outer loop is takes **O(logm)** average time and **O(m)** worst-case time to search and update.

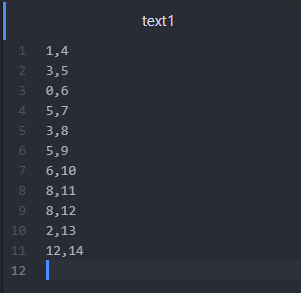
• Writing the "output.txt" file takes **O(n)** time.

Therefore, this algorithm will show an average, and worst-case time-complexities of **O(nlogn)** and **O(nlogn+ nm)**, respectively:

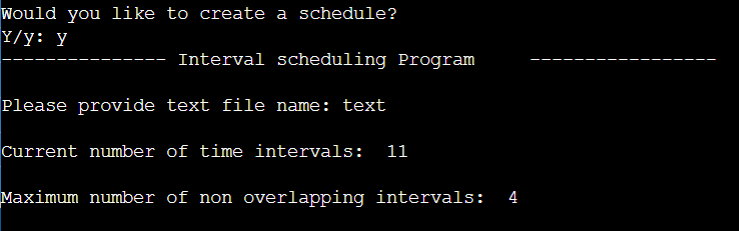
1. **O(n + nlogn + m + nlogm + n) = O(nlogn + nlogm) = O(nlogn)**
2. **O(n + nlogn + m + nm + n) = O(nlogn + nm)**

**Results:**

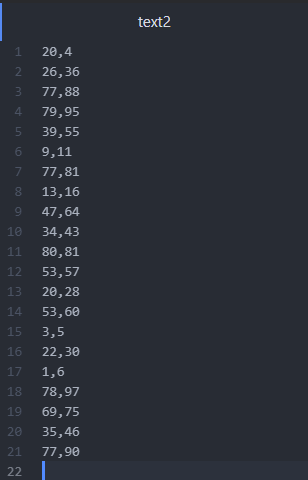
**First dataset/text file:**



**Results:**



**Second dataset/text file:**



**Results:**

