This is a coding question. You may use C, C++, or Python. Include all your source code, including any tests you write! In addition to submitting source code, comment on the algorithmic complexity of each of the API methods you implement. Clearly state any assumptions you make.

You are to design and implement a data structure that can be used to keep track of and manipulate “ranges”. A “range” is defined by a starting point (inclusive) and an ending point (exclusive), for example the range (1, 4) contains the values 1, 2, and 3. You may assume the ranges can only be defined with integer values.

Implement the following three methods.

a) Add(start, end): Adds a new range into the data structure from “start” to “end”. If the range

already exists, or is contained within an already existing range, then nothing needs to be done. If the new range partially overlaps an existing range, then the two ranges should be merged into a new range that covers the union of the new and old ranges.

Examples:

i) Original state [(1, 2)], Add(3, 5), New state: [(1, 2), (3, 5)]

ii) Original state [(1, 6)], Add(3, 5), New state: [(1, 6)]

iii) Original state [(1, 4)], Add(3, 5), New state: [(1, 5)]

b) Delete(start, end): Deletes the range from “start” to “end” from the data structure. If the range does not exist, then nothing needs to be done. If the range partially overlaps an existing range, then the existing range should be truncated.

Examples:

i) Original state [(1, 6)], Delete(-3, -1), New state: [(1, 6)]

ii) Original state [(1, 6)], Delete(-1, 10), New state: []

iii) Original state [(1, 6)], Delete(4, 10), New state: [(1, 4)]

c) Get(start, end): Returns a list of ranges contained in the data that intersect with “start” and

“end”.

Examples:

i) State [(1, 3), (5, 7)]. Get(4, 5). Returns []

ii) State [(1, 3), (5, 6)]. Get(4, 6). Returns [(5, 6)]

iii) State [(1, 3), (5, 6)]. Get(2, 9). Returns [(1, 3), (5, 6)]