****

**Project - XML editor**

**Report**

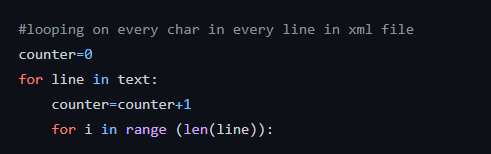
**Prepared by :**

**1st : error checking :**

**Complexity of operations:**

**O(n\*m)**

**Where n is number of lines in the xml file and n is number of characters per line**

****

**1-First for loop: lines O(n)**

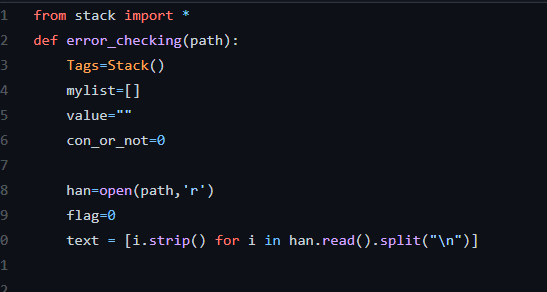
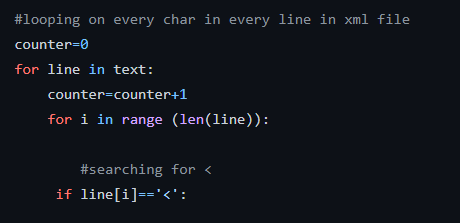
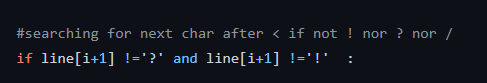
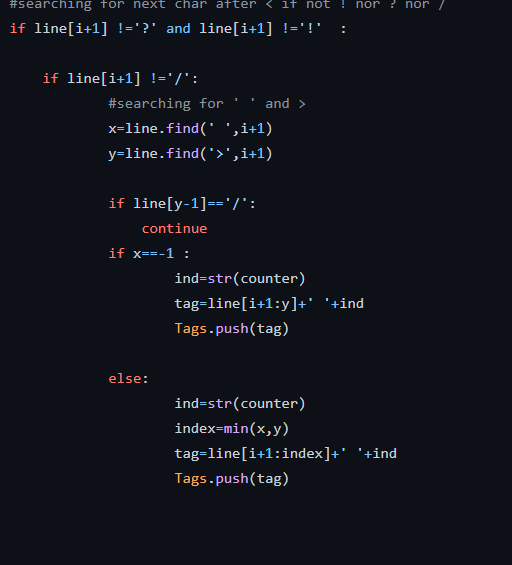
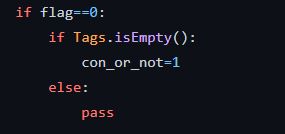
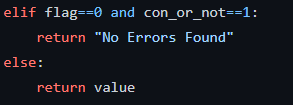
**Second for loop: characters in line O(m)**

**O(n)\*O(m)**

**2-anything else will be O(1) or O(name of characters in tag name) that is also is less than O(m)**

**So overall O(n)\*O(m) = O(n\*m)**

**Implementation details:**

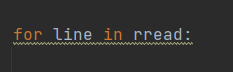
1. **Defining variables ,importing stack and initializing flags to be used later inside algorithm**
2. **Make sure that text is left stripped and splitting it into a list of lines** 
3. **Looping in the characters of each line**
4. **Checking that the character is a start of tag**
5. **Making sure that the tag is not a comment (<? Or <!)**
6. **If the tag is open tag:  
   find index of first space and “>” (to determine name of the tag)  
   checking if the tag is open and close in the same time(will not be pushed as it doesn’t cause a problem)  
   pushing the tag into stack **
7. **If the tag is close:  
   popping the top most tag from stack and comparing it with the closed one to determine error**
8. **Checking that stack is empty to determine error(no extra open tags)  
   **
9. **Returning no error found  
   or returning the error  
   **

**2nd prettifying:**

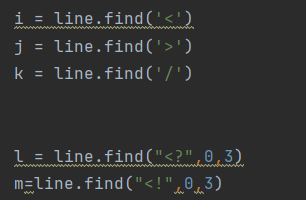
**Complexity of operations:**

**In worst case:**

**O(n\*m)**

**1-O(n):from each line in file  
**

**2-O(m):from operations like line.find() that will be o(m) in its worst case**

****

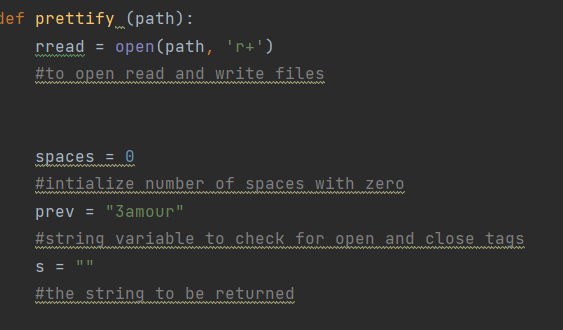
**3-anything else will be O(1)**

**So overall O(n)\*O(m) = O(n\*m)**

**Implementation details:**

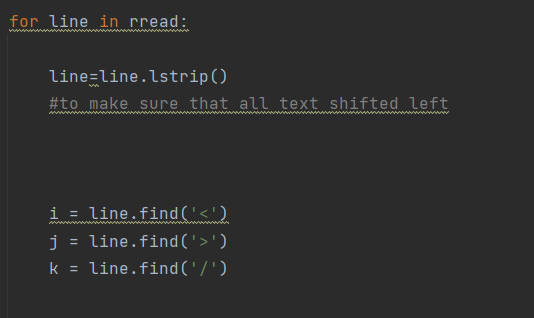
**1-opening file and initializing variables:  
reading the file  
create spaces variable to store in it number of spaces for each line  
creating previous variable: to determine if the previous tag is open or close**

**Creating string to store in it the new prettified content**

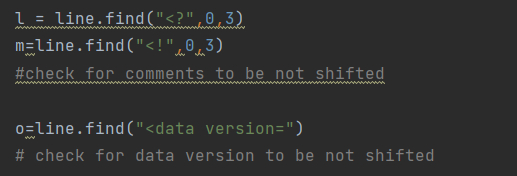
****

**2-looping in every line in file and make sure that it is left stripped**

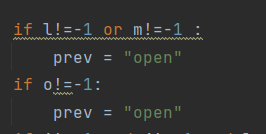
**3- determining indexes of “<” , “/” and”>” to use them later**

****

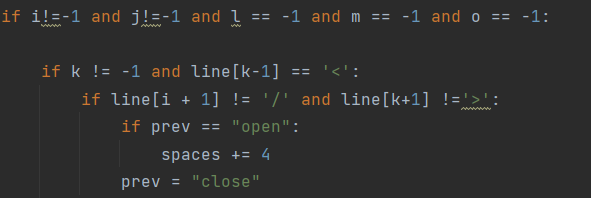
**4-checking if the line is a comment or a data version tag to be not shifted**

****

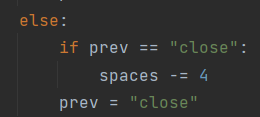
**5- setting prev to open(to run the algorithm for them) for the next line if the tag is comment or data**

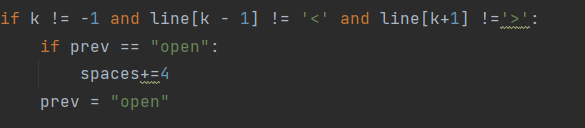


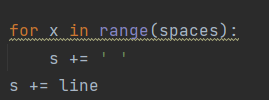
**6- if line contains open and close tags in the same time: it will be shifted if the previous one is open only**

****

**7-if line contains close tag only: it will be shifted backwards if the previous one is close only**



**8-if line contains open tag only: it will be shifted forward if the previous one is open only** 

**9-writing spaces before each line according to algorithm checks:  
  
10-returning the new prettified string to be written in the file** ****