XML Validation using XML Schema

Name: Ankur Sarda

USN: 1PI10CS139

Objective: Given an XML Schema as an Input, verify the given XML Document

Language: Python

Input: XML Schema File (.xsd file), XML File (.xml file)

STEPS:

1. The input from XML and XSD file is taken in the form of a string line by line
2. The string of line received from the file every time it is accessed is appended to a string variable.
3. The string variable has the complete string in the XML and XSD file.
4. All the extra spaces from the given document are removed i.e., unwanted spaces
5. An object of a class consist of a valid tag, i.e., within (< > tags, e.g. : <html>)
6. The tags consist of an element name, namespace; attribute key value pairs, data associated with it and depth of the tag.
7. Use regular expression features of Python to take a string within <> tags. The character ‘<’ or ‘>’ with a previous character as ‘\’ is not to be taken in. (eg: the object for tag : <abc name = ‘\<’ > as a whole will b a object.
8. If the tag is of type ‘<…/>’, it is converted to type: ‘<></>’. And two objects are created.
9. Class structure is as follow: class: node, data members: namespace, element name, data, attribute (dictionary-key value pair), depth, list of child elements (list of nodes).
10. The tag received from the regular expression is then again subject to another regular expression based on spaces. Using the function findall of regular expression (re), a list of matched strings is obtained.
11. The space between the quotes of attributes is not to be considered. (Please note extra spaces have been removed before).
12. Using the function findall of regex (re), a list of matched strings is obtained same as before.
13. The first element is namespace and element name in a combined manner.
14. From the second element to the last element is the dictionary of attributes.
15. The first element is again split based on character ‘:’.
16. The attributes are split based on character ‘=’.
17. The data member ‘data ‘of the object is anything that comes after character ‘>’ of the given tag and the next opening tag ‘<’. If not present, this will not contain anything.
18. The same procedure is repeated for the whole line of input for both XML and XSD document. Since both are in the same XML format, same operations are performed on both the string of XML and XSD.
19. Hence a list of objects of nodes is created and passed on for tree making.