Python Assignment - 12

**1. In what modes should the PdfFileReader() and PdfFileWriter() File objects will be opened?**

=> When working with these functions, you should open the file objects in the following modes:

**1. PdfFileReader**(): When using PdfFileReader(), you should open the PDF file in the "rb" mode, which stands for "read binary". This mode is specifically designed for reading binary files like PDFs. Opening the file in "rb" mode ensures that the file is opened for reading as binary data, preserving the integrity of the PDF file.

**2. PdfFileWriter**(): When using PdfFileWriter(), you should open the output PDF file in the "wb" mode, which stands for "write binary". This mode is used for writing binary data to a file. Opening the file in "wb" mode ensures that the file is opened for writing binary data.

**2. From a PdfFileReader object, how do you get a Page object for page 5?**

=> To get a Page object for page 5 from a PdfFileReader object in the PyPDF2 library, you can use the ‘getPage()’ method and pass the index of the page you want (zero-based index).

**3. What PdfFileReader variable stores the number of pages in the PDF document?**

=> The PdfFileReader class in the PyPDF2 library provides a variable named ‘numPages’ that stores the number of pages in the PDF document. You can access this variable to retrieve the total page count of the PDF.

**4. If a PdfFileReader object’s PDF is encrypted with the password swordfish, what must you do before you can obtain Page objects from it?**

=> If a PdfFileReader object's PDF is encrypted with the password "swordfish," you must provide the password to decrypt the PDF before you can obtain Page objects from it. To do so, you can use the decrypt() method of the PdfFileReader object.

Example:

From PyPDF2 import PdfFileReader

With open(‘input.pdf’, ‘rb’) as file:

pdf\_reader = PdfFileReader(file)

page\_count = pdf\_reader.numPages

print(“Total number of pages ”, page\_count)

**5. What methods do you use to rotate a page?**

=> To rotate a page in a PDF using the PyPDF2 library, you can utilize the following methods:

**1. rotateClockwise**(): This method rotates the page clockwise by the specified angle. The angle parameter accepts an integer value that represents the rotation angle in degrees.

**2. rotateCounterClockwise**(): This method rotates the page counter-clockwise by the specified angle.

**6. What is the difference between a Run object and a Paragraph object?**

=> The differences between Run object and Paragraph object are:

1. **Paragraph object**: A Paragraph object represents a paragraph-level element in a document. It typically contains one or more Run objects and represents a block of text with common formatting properties. A Paragraph object can include multiple lines or sentences, and it is the fundamental unit for organizing and formatting text within a document.

**2. Run object**: A Run object represents a contiguous run of text within a paragraph. It is a sequence of characters in a specific style or formatting within the paragraph. Runs can have different formatting attributes such as font style, size, color, and emphasis (e.g., bold, italic). A paragraph may contain multiple Run objects, each with its own formatting properties.

**7. How do you obtain a list of Paragraph objects for a Document object that’s stored in a variable named doc?**

=> To obtain a list of Paragraph objects for a Document object in python-docx library, you can use the paragraphs attribute of the Document object.

Example:

from docx import Document

paragraphs = doc.paragraphs

for paragraph in paragraphs:

print(paragraph.text)

**8. What type of object has bold, underline, italic, strike, and outline variables?**

=> The ‘Font’ object in the ‘python-docx’ library has all the above variables.

**9. What is the difference between False, True, and None for the bold variable?**

=> Differences between False, True and None:

1. **False**: When the bold variable is set to False, it means that the text should not be rendered in bold. The text will be displayed with the regular or default font weight.

**2. True**: Setting the bold variable to True indicates that the text should be rendered in bold. The text will appear with a bold font weight.

**3. None**: If the bold variable is set to None, it means that the bold formatting for the text is not explicitly specified. In this case, the text will inherit the default bold formatting based on the surrounding styles or the default style of the document.

**10. How do you create a Document object for a new Word document?**

=> To create a Document object for a new Word document using the python-docx library, you can use the Document() constructor.

Example:

from docx import Document

doc = Document()

**11. How do you add a paragraph with the text 'Hello, there!' to a Document object stored in a variable named doc?**

=> To add a paragraph with the text 'Hello, there!' to a Document object stored in a variable named doc using the python-docx library, you can use the add\_paragraph() method.

Example:

from docx import Document

doc.add\_paragraph(‘Hello, there!’)

**12. What integers represent the levels of headings available in Word documents?**

=> commonly used integer values for representing the levels of headings in Word documents are:

**1. Heading 1**: Level 1 headings are represented by the integer value 1.

**2. Heading 2**: Level 2 headings are represented by the integer value 2.

**3. Heading 3**: Level 3 headings are represented by the integer value 3.

**4. Heading 4**: Level 4 headings are represented by the integer value 4.

**5. Heading 5**: Level 5 headings are represented by the integer value 5.

**6. Heading 6**: Level 6 headings are represented by the integer value 6.

**7. Heading 7**: Level 7 headings are represented by the integer value 7.

**8. Heading 8**: Level 8 headings are represented by the integer value 8.

**9. Heading 9**: Level 9 headings are represented by the integer value 9.