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

Ans-The PdfFileReader() and PdfFileWriter() File objects should be opened in binary mode, using the 'rb' (read binary) mode for PdfFileReader() and the 'wb' (write binary) mode for PdfFileWriter().

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

Ans-To get a Page object for page 5 from a PdfFileReader object, you can use the following code:

page = pdf\_reader.getPage(4)

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

Ans-The number of pages in the PDF document is stored in the numPages attribute of the PdfFileReader object. You can access it using:

num\_pages = pdf\_reader.numPages

**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?**

Ans-If a PdfFileReader object's PDF is encrypted with the password "swordfish", you need to decrypt it by calling the decrypt() method and passing the correct password before you can obtain Page objects. Here's an example:

pdf\_reader.decrypt('swordfish') # Decrypts the PDF with the password 'swordfish'

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

Ans-To rotate a page in a PDF document, you can use the rotateClockwise() or rotateCounterClockwise() methods of the Page object. Here are the methods and their usage:

page.rotateClockwise(90) # Rotates the page 90 degrees clockwise

page.rotateCounterClockwise(180)

# Rotates the page 180 degrees counterclockwise

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

Ans-A Run object represents a contiguous run of text within a paragraph that has the same character style. A Paragraph object represents a single paragraph of text within a document, which can contain one or more Runs.

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

Ans-To obtain a list of Paragraph objects for a Document object stored in a variable named "doc," you can access the paragraphs attribute of the Document object:

paragraphs = doc.paragraphs

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

Ans-The Run object in python-docx has bold, underline, italic, strike, and outline variables. It represents a run of text with specific character formatting.

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

Ans-In python-docx, the bold variable can have three possible values:

True: The text is explicitly marked as bold.

False: The text is explicitly marked as not bold.

None: The text does not have an explicitly set bold formatting. The default formatting specified by the style or the inherited formatting is used.

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

Ans-To create a Document object for a new Word document, you can use the following code:

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?**

Ans-To add a paragraph with the text 'Hello, there!' to a Document object stored in a variable named "doc," you can use the following code:

doc.add\_paragraph('Hello, there!')

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

Ans-The integer levels of headings available in Word documents are 0 to 8, with 0 being the title of the document, 1 being the main heading, 2 being the subheading, and so on up to 8.