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

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

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

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?

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

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

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

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

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

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

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

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

**ANS:  
Here are the answers to your questions:**

**1. `PdfFileReader()` and `PdfFileWriter()` File objects should be opened in binary mode (`'rb'` for reading and `'wb'` for writing), as PDF files are binary files.**

**2. To get a Page object for page 5 from a `PdfFileReader` object, you can use the `getPage()` method:**

**```python**

**page = pdf\_reader.getPage(4) # Page numbers are zero-based, so page 5 corresponds to index 4**

**```**

**3. The `numPages` variable of a `PdfFileReader` object stores the number of pages in the PDF document.**

**4. If a `PdfFileReader` object's PDF is encrypted with the password 'swordfish', you must use the `decrypt()` method with the password as an argument before obtaining Page objects from it:**

**```python**

**pdf\_reader.decrypt('swordfish')**

**```**

**5. To rotate a page, you can use the `rotateClockwise()` or `rotateCounterClockwise()` methods of the Page object.**

**6. In the context of Python-docx library:**

**- A Run object represents a contiguous run of text with the same character style within a paragraph.**

**- A Paragraph object represents a single paragraph of text.**

**7. To obtain a list of Paragraph objects for a Document object stored in a variable named `doc`, you can use the `paragraphs` attribute:**

**```python**

**paragraphs = doc.paragraphs**

**```**

**8. The object type that has bold, underline, italic, strike, and outline variables is a Run object. These variables represent the formatting applied to the text in the run.**

**9. In the context of a Run object in Python-docx:**

**- `False`: The text is not formatted with the respective style attribute (e.g., not bold, not italic).**

**- `True`: The text is formatted with the respective style attribute (e.g., bold, italic).**

**- `None`: The formatting of the respective style attribute is inherited from the style of the parent paragraph.**

**10. To create a Document object for a new Word document, you can use the `Document()` constructor:**

**```python**

**from docx import Document**

**doc = Document()**

**```**

**11. To add a paragraph with the text 'Hello, there!' to a Document object stored in a variable named `doc`, you can use the `add\_paragraph()` method:**

**```python**

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

**```**

**12. The integers representing the levels of headings available in Word documents are 1 to 9, where 1 represents the highest level of heading and 9 represents the lowest level.**