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

ANSWER.

When working with the `PdfFileReader()` and `PdfFileWriter()` objects in Python's `PyPDF2` library, the File objects passed to these functions should be opened in binary mode (`'rb'` for reading and `'wb'` for writing).

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

ANSWER.

with open('example.pdf', 'rb') as file:

pdf\_reader = PdfFileReader(file)

page\_5 = pdf\_reader.getPage(4)

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

ANSWER.

The PdfFileReader variable that stores the number of pages in the PDF document is `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?

ANSWER.

If a PdfFileReader object's PDF is encrypted with the password "swordfish," you need to decrypt the PDF by passing the password as an argument to the `decrypt()` method before you can obtain Page objects from it.

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

ANSWER.

To rotate a page in a PDF document using the PyPDF2 library, you can use the `rotateClockwise()` or `rotateCounterClockwise()` methods of the Page object. These methods allow you to rotate the page clockwise or counterclockwise by 90 degrees, respectively.

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

ANSWER.

A Paragraph object represents a single paragraph of text in a Word document, while a Run object represents a contiguous range of text within a paragraph with the same formatting attributes. Paragraphs contain Runs, and you can access and manipulate both to programmatically modify the content and formatting of a Word document using the Word object model.

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

ANSWER.

To obtain a list of Paragraph objects for a Document object stored in a variable named `doc` in the python-docx library, you can use the `doc.paragraphs` attribute.

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

ANSWER.

The object in the python-docx library that has attributes such as bold, underline, italic, strike, and outline is the `Run` object.

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

ANSWER.

`False` and `True` are used to explicitly set the bold formatting of text, while `None` allows the text to inherit its formatting from the parent style or default settings.

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

ANSWER.

To create a `Document` object for a new Word document using the python-docx library, you can simply use the `Document()` constructor without passing any arguments.

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

ANSWER.

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.

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

ANSWER.

In Word documents, the levels of headings are typically represented by integers ranging from 1 to 9. Each integer corresponds to a specific level of heading, with 1 being the highest level (main heading or title) and 9 being the lowest level (sub-sub-sub-sub-sub-sub-sub-subheading).

.