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

In Python, when working with PdfFileReader() and PdfFileWriter() objects from the PyPDF2 module, the corresponding PDF files should be opened in binary mode ('rb' for reading and 'wb' for writing).

In [2]:
```
#Here's an example of how to open a PDF file using PdfFileReader() in binary mode:
import PyPDF2

with open('example.pdf', 'rb') as pdf_file:
    pdf_reader = PyPDF2.PdfReader(pdf_file)
```

In [3]:
```
#And here's an example of how to open a PDF file using PdfFileWriter() in binary mode:
import PyPDF2

with open('example.pdf', 'rb') as pdf_file:
    pdf_reader = PyPDF2.PdfReader(pdf_file)

with open('output.pdf', 'wb') as output_file:
    pdf_writer = PyPDF2.PdfWriter()
    pdf_writer.write(output_file)
```

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

To get a Page object for page 5 of a PDF file using a PdfFileReader object from the PyPDF2 module, you can use the getPage() method and pass the index of the page you want as an argument (remember that page indexes in PyPDF2 start at 0, not 1).

In [8]:
```
import PyPDF2

with open('example.pdf', 'rb') as pdf_file:
    pdf_reader = PyPDF2.PdfReader(pdf_file)
    page_five = pdf_reader.pages[4] # get the fifth page (index 4)
```

In this example, we open the PDF file in binary mode, create a PdfFileReader object from it, and then use the getPage() method to get the Page object for page 5 (which has an index of 4, since indexing starts at 0).

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

In the PyPDF2 module, the PdfFileReader object has a variable called numPages that stores the number of pages in the PDF document. You can access this variable directly as an attribute of the PdfFileReader object,

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 first decrypt it using the decrypt() method before you can obtain Page objects from it.

import PyPDF2 with open('example.pdf', 'rb') as pdf_file: pdf_reader = PyPDF2.PdfReader(pdf_file) pdf_reader.decrypt('swordfish') # decrypt the PDF file with the password # now you can access the pages in the PDF file page_one = pdf_reader.getPage(0) page_two = pdf_reader.getPage(1) 5. What methods do you use to rotate a page?

In PyPDF2, we can rotate a page by using the rotateClockwise() or rotateCounterClockwise() method of the Page object.

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

In the context of Python's python-docx library, a Run object and a Paragraph object are both used to represent text in a Word document, but they have different purposes and properties.

A Paragraph object represents a single paragraph of text in a Word document. It contains one or more Run objects, which represent contiguous runs of text with the same formatting properties (e.g., font size, boldness, etc.). A Paragraph object can have multiple Run objects if the text within the paragraph has different 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 that's stored in a variable named doc, we can use the paragraphs attribute of the Document object.

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

The Run object in the python-docx module has the following properties: bold, underline, italic, strike, and outline. These properties represent the formatting applied to the text contained in the Run object.

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

In the Run object of the python-docx module, the bold property can have three possible values: True, False, or None.

If bold is set to True, the text contained in the Run object will be displayed in bold font.

If bold is set to False, the text contained in the Run object will not be displayed in bold font.

If bold is set to None, the text contained in the Run object will inherit the bold setting from its style.

So, True and False values are used to explicitly set the bold property to the specified value, whereas None is used to allow the bold property to inherit its value from the style.

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 module, we can use the docx.Document() function with no argument

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 module, we can use the add_paragraph() method of the Document object.

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

In Word documents, the following integers represent the levels of headings:

Level 1: 0 Level 2: 1 Level 3: 2 Level 4: 3 Level 5: 4 Level 6: 5 Level 7: 6 Level 8: 7 Level 9: 8 These integers are used to specify the level of a heading when creating or modifying a Paragraph object using the python-docx module. For example, to create a new heading at level 2, you would use the add_heading() method of the Document object with the level argument set to 1,

In [ ]: