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

The PdfFileReader() and PdfFileWriter() classes in Python's PyPDF2 library are used for reading and writing PDF files, respectively.

To create a PdfFileReader object, you should open the input PDF file in the 'rb' (read binary) mode, since a PDF file is a binary file. For example:

In [ ]:

```
from PyPDF2 import PdfFileReader

with open('input.pdf', 'rb') as file:
    reader = PdfFileReader(file)
```

Here, we first open the input.pdf file in 'rb' mode using a with block, and then pass the file object to the PdfFileReader() constructor to create a reader object.

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

Ans: **`PdfFileReader`** class provides a method called **`getPage(page_no)`** to get a page object.

In [2]:

```
# Example Code:
from PyPDF2 import PdfFileReader
pdf_reader = PdfFileReader(file_path)
for page in pdf_reader.getNumPages():
    pdf_reader.getPage(page)
```

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

Ans: **`getNumPages()`** method of **`PdfFileReader`** class stores the no pages in a PDF document

In [ ]:

```
#Example Code:
from PyPDF2 import PdfFileReader
pdf_reader = PdfFileReader(file_path)
print(pdf_reader.getNumPages()) # Prints the no of pages in a input 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?

Ans: If a `PdfFileReader` object’s PDF is encrypted with the password `swordfish` and you're not aware of it. first read the Pdf using the PdfFileReader Class. PdfFileReader class provides a attribute called `isEncrypted` to check whether a pdf is encrypted or not. the method returns true if a pdf is encrypted and vice versa.
if pdf is encrypted use the `decrypt()` method provided by PdfFileReader class first then try to read the contents/pages of the pdf, else PyPDF2 will raise the following error
`PyPDF2.utils.PdfReadError: file has not been decrypted`

In [ ]:

```python
#Example Code:
from PyPDF2 import PdfFileReader
pdf_reader = PdfFileReader(file_path)
if pdf_reader.isEncrypted: # to check whether the pdf is encrypted or not
    pdf_reader.decrypt("swordfish")
for page in pdf_reader.pages:
    print(page.extractText()) # to print the text data of a page from pdf
```

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

Ans: PyPDF2 Package provides 2 methods to rotate a page:

1. `rotateClockWise()` -> For Clockwise rotation
2. `rotateCounterClockWise()` -> For Counter Clockwise rotation

The PyPDF2 package only allows you to rotate a page in increments of 90 degrees. You will receive an AssertionError otherwise.

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

Ans: In Python's python-docx library, both Run and Paragraph are classes that represent different parts of a Word document.

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 in the paragraph that share the same set of character formatting.

The main difference between a Run and a Paragraph object is that a Run represents a continuous span of text with the same formatting, whereas a Paragraph can contain one or more Run objects and represents a block of text with potentially different formatting between runs.

In other words, a Run object is used to apply formatting to a specific section of text within a

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

In [ ]:

```
# Example Program
from docx import Document
doc = Document("sample_file.docx") # Path of the Docx file
print(doc.paragraphs) # Prints the list of Paragraph objects for a Document
for paragraph in doc.paragraphs:
    print(paragraph.text) # Prints the text in the paragraph
```

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

Ans: `Run` object has bold, underline, italic, strike, and outline variables. The text in a Word document is more than just a string. It has font, size, color, and other styling information associated with it.

A style in Word is a collection of these attributes. A Run object is a contiguous run of text with the same style. A new Run object is needed whenever the text style changes.

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

In [ ]:

```
bold = True  # Style Set to Bold
bold = False # Style Not Set to Bold
bold = None # Style is Not Applicable
```

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

In [2]:

```
# Example Program
from docx import Document
document = Document()
document.add_paragraph("iNeuron Full Stack DataScience Course")
document.save('mydocument.docx')
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_10044/2689855423.py in <module>
      1 # Example Program
----> 2 from docx import Document
      3 document = Document()
      4 document.add_paragraph("iNeuron Full Stack DataScience Course")
      5 document.save('mydocument.docx')

ModuleNotFoundError: No module named 'docx'
```

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

In [ ]:

```
# Example Program
from docx import Document
doc = Document()
doc.add_paragraph('Hello, there!')
doc.save('hello.docx')
```

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

Ans: The levels for a heading in a word document can be specified by using the `level` attribute inside the `add_heading` method. There are a total of 5 levels statring for 0 t0 4. where level 0 makes a headline with the horizontal line below the text, whereas the heading level 1 is the main heading. Similarly, the other headings are sub-heading with their's font-sizes in decreasing order.

In [ ]:

```

```