Assignment_12

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

> The PdfFileReader() and PdfFileWriter() file objects should be opened in read-binary mode, which is indicated by the rb mode. This mode used to open files that are only to be read and prevents the file from being modified or corrupted.

```
import PyPDF2 as pdf
pdf_file = open("my_pdf.pdf", "rb")
pdf_reader = pdf.PdfFileReader(pdf_file)
print(pdf_reader)
```

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

> we can get get a Page object for page 5 as: `page_5 = pdf_reader.getPage(5)`
>
> ```
> import PyPDF2 as pdf
> pdf_file = open("my_pdf.pdf", "rb")
> pdf_reader = pdf.PdfFileReader(pdf_file)
> page_5 = pdf_reader.getPage(5)
> ```

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

> The numPages variable in a PdfFileReader object stores the number of pages in the PDF document.

```
import PyPDF2 as pdf
pdf_file = open("my_pdf.pdf", "rb")
pdf_reader = pdf.PdfFileReader(pdf_file)
num_pages = pdf_reader.numPages
print(num_pages)
```

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?

We should provide the password before we can obtain Page objects from it. we can do this by using the decrypt() method of the PdfFileReader object.
`pdf_reader.decrypt("swordfish")`

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

There are two methods that you can use to rotate a page in PyPDF2:

- The rotate() method of the Page object.
- The rotatePages() method of the PdfFileWriter object.

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

A Run object and a Paragraph object are both objects that represent text in a PDF file.

Feature	Run object	Paragraph object
Represents	A single piece of text	A group of Run objects
Can contain	Only text	Text, fonts, colors, sizes, and other formatting information
Can be used to	Create text boxes, labels, and other text-based objects	Create paragraphs, headings, and other complex text-based objects

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

- We can use the getParagraphs() method that returns a list of Paragraph objects for the Document object.
- Now, open the PDF file in read-binary mode & create a PdfFileReader object. The getDocument() method will then be used to get a Document object for the PDF file. The getParagraphs() method will be used to get a list of Paragraph objects for the Document object.
- The output of the code will be a list of Paragraph objects.

```
import PyPDF2
pdf_file = open("my_pdf.pdf", "rb")
pdf_reader = PyPDF2.PdfFileReader(pdf_file)
doc = pdf_reader.getDocument()
paragraphs = doc.getParagraphs()
```

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

> The TextObject class in the PyPDF2 module has bold, underline, italic, strike, and outline variables. These variables control the formatting of text in a PDF file.

```
import PyPDF2 as pdf

pdf_file = open("my_pdf.pdf", "rb")
pdf_reader = pdf.PdfFileReader(pdf_file)
page = pdf_reader.getPage(5)
text_object = page.get_text_object()

text_object.bold = True
text_object.underline = True
text_object.italic = True

pdf_writer = pdf.PdfFileWriter()
pdf_writer.addPage(page)

with open("new_pdf.pdf", "wb") as output_file:
    pdf_writer.write(output_file)
```

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

> The bold variable can be assigned one of three values: False, True, or None.

- **False** is a Boolean value that represents the logical concept of "falsehood." It is the opposite of True. In Python, False is equal to 0, an empty string, an empty list, an empty tuple, an empty dictionary, or the special value None.
- **True** is a Boolean value that represents the logical concept of "truth." It is the opposite of False. In Python, True is equal to anything that is not False.
- **None** is a special value that represents the absence of a value. It is not a Boolean value, and it is not equal to False or True.

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

> To create a Document object for a new Word document, we can use the code:

```
import docx
document = docx.Document()
```

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

```
import docx
doc = docx.Document()
paragraph = doc.add_paragraph("Hello, there!")
```

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

> There are 9 levels of headings available in Word documents. The integers that represent these levels are 0 to 8.

```
Level 0: This is the highest level heading and is typically used f
or the title of the document.
Level 1: This is the second-highest level heading and is typically
used for the main headings in the document.
Level 2: This is the third-highest level heading and is typically
used for the subheadings in the document.
Level 3: This is the fourth-highest level heading and is typically
used for the sub-subheadings in the document.
Level 4: This is the fifth-highest level heading and is typically
used for the sub-sub-subheadings in the document.
Level 5: This is the sixth-highest level heading and is typically
used for the sub-sub-sub-subheadings in the document.
Level 6: This is the seventh-highest level heading and is typicall
y used for the sub-sub-sub-sub-subheadings in the document.
Level 7: This is the eighth-highest level heading and is typically
used for the sub-sub-sub-sub-sub-subheadings in the document.
Level 8: This is the ninth-highest level heading and is typically
used for the sub-sub-sub-sub-sub-sub-subheadings in the document.
```