# Observations regarding OCR on various documents:

## 1. Table of Observations:

| Doc No. | Doc Description | Font Type | Observation |
|---|---|---|---|
| 1 | Anandabazar Patrika's News column | Anandabazar | 1. A specific line was always getting ignored, irrespective of its position in the document.<br>2. The heading of the column was ignored, along with the first line.<br>3. When this heading was erased, then the first line of the news column was recognized appropriately.<br>4. "purnochhed" (Full-stop symbol in Bengali) was confused as "aa"-kar and vice versa in some scenarios.<br>5. "Chandrabindu" (ঁ) was ignored in all of the cases<br>6. Difficulty in recognizing some of the "juktakkhor" (e.g. স্র, ঙ্গ) |
| 2 | PDF version of "Feluda Somogro" (Ananda Publishers) - Text excerpt | Anandabazar | 1. Accuracy better than Doc No. 1<br>2. Frequent paragraph change (when the story was in form of multiple speakers' conversation) caused some of the line misses. |
| 3 | A small story in Bengali Font, exported as a PDF from Microsoft Word | Vrinda | 1. Accuracy better than Doc No. 1<br>2. Again, position independent line miss |
| 4 | Do. | Anandabazar | 1. As compared to the Vrinda font version of the same Document, recognition was better.<br>2. The immediate word after a 'hyphen' mark was missed. |
| 5 | Wikipedia Bengali article PDF export (Indian National Cricket Team) | Vrinda | 1. Best accuracy achieved so far.<br>2. One of the rows in the tabular data of the doc (invloving a mixture of Bengali and English alphanumeric text) was missed. |
| 6 | Do. | Calibri | Do. |
| 7 | Bengali Map Marking (Location of Stadiums in India) | Vrinda | 1. Out of many map markings, only "Wankhede" (ওয়াংখেড়ে) was recognized as text. (Probably because it is the leftmost marking on the map) |

## 2. Codes used for OCR:

### a. For OCR (For upto ~80 pages) (Python):

```
from multilingual_pdf2text.pdf2text import PDF2Text
from multilingual_pdf2text.models.document_model.document import
Document
import logging
import fpdf
logging.basicConfig(level=logging.INFO)

def main():
    # create document for extraction with configurations
    pdf_document = Document(
        document_path='C:\\CODING\\Bangla_OCR\\Test.pdf',
        language='Bengali'
        )
    pdf2text = PDF2Text(document=pdf_document)
    content = pdf2text.extract()
    # get size of content
    print(len(content))
    with open("file.txt", "w", encoding="utf-8") as f:
        for i in range(len(content)):
            f.write(content[i]['text'])
            f.write("\n")

if __name__ == "__main__":
    main()
```

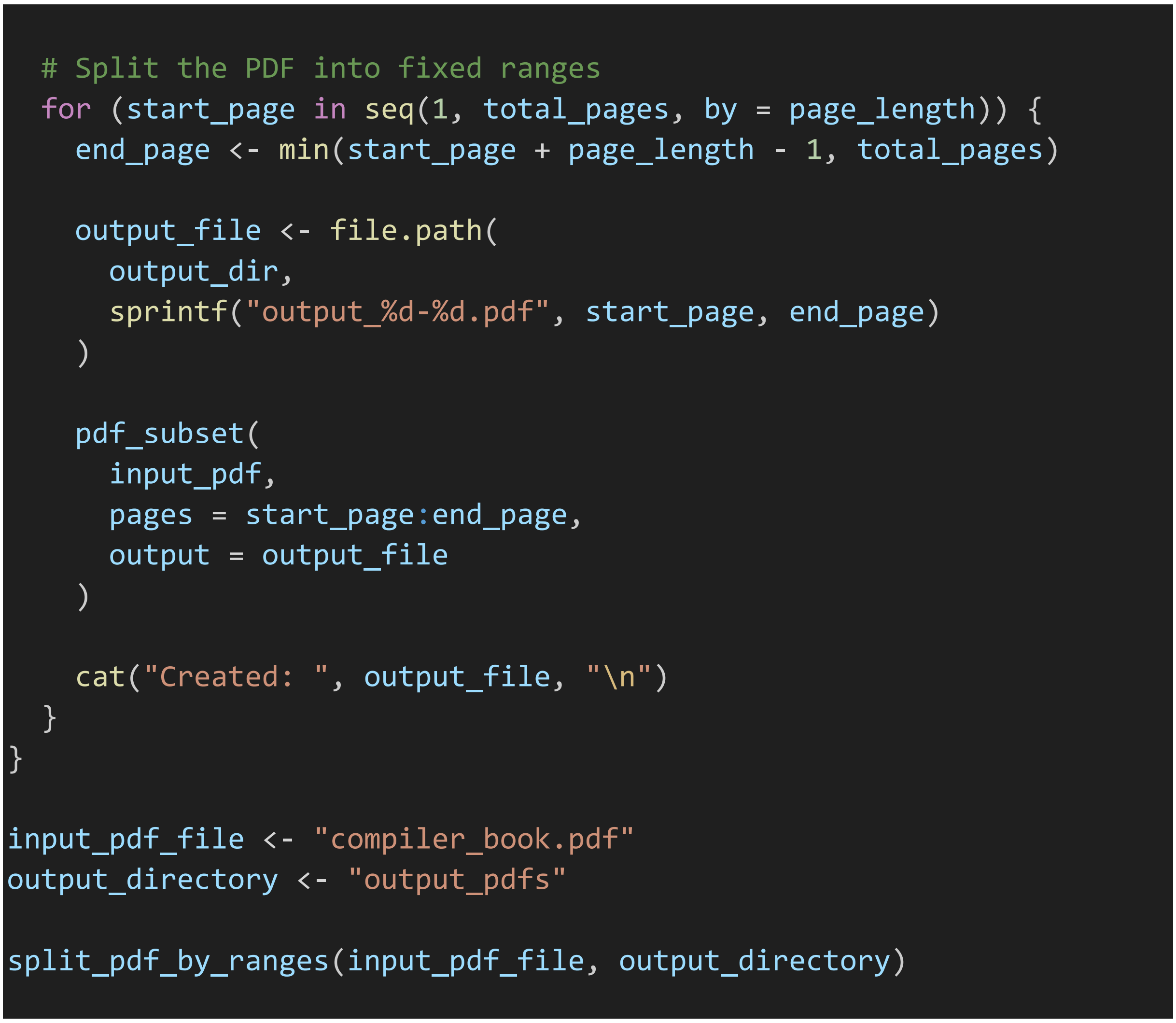
### b. For PDF Splitting (R Language):

```
# Install and load the pdftools package
if (!requireNamespace("pdftools", quietly = TRUE)) {
  install.packages("pdftools")
}

library(pdftools)

# Function to split PDF into fixed ranges
split_pdf_by_ranges <- function(input_pdf, output_dir, page_length =
50) {
  pdf_info <- pdf_info(input_pdf)
  total_pages <- pdf_info[["pages"]]

  # Create the output directory if it doesn't exist
  dir.create(output_dir, showWarnings = FALSE)
```



```r
  # Split the PDF into fixed ranges
  for (start_page in seq(1, total_pages, by = page_length)) {
    end_page <- min(start_page + page_length - 1, total_pages)

    output_file <- file.path(
      output_dir,
      sprintf("output_%d-%d.pdf", start_page, end_page)
    )

    pdf_subset(
      input_pdf,
      pages = start_page:end_page,
      output = output_file
    )

    cat("Created: ", output_file, "\n")
  }
}

input_pdf_file <- "compiler_book.pdf"
output_directory <- "output_pdfs"

split_pdf_by_ranges(input_pdf_file, output_directory)
```