```
#installing dependency
!pip -q install pymupdf
```

```
#DOB:07/13/2002
from google.colab import files
import fitz
import os
from typing import Tuple
```

```
#uploading the PDF from my computer
#choosing harrypotter.pdf
uploaded = files.upload()
pdf_name = next(iter(uploaded.keys()))
pdf_path = os.path.abspath(pdf_name)
print("Uploaded PDF:", pdf_path)
```

Choose Files harrypotter.pdf

harrypotter.pdf(application/pdf) - 17338552 bytes, last modified: 2/23/2026 - 100% done

```
Saving harrypotter.pdf to harrypotter (1).pdf
Uploaded PDF: /content/harrypotter (1).pdf
```

```
#function to extract a range of pages from the PDF
#pages are treated as human-readable(1 based indexing)
def extract_pages_to_txt(
    pdf_path: str,
    start_page: int,
    num_pages: int,
    outer_txt_path: str
) -> Tuple[int, int]:

    if start_page < 1:
        raise ValueError("The starting page number should be 1.")

    with fitz.open(pdf_path) as doc:
        total_pages = doc.page_count

        # converting 1 based page number to a 0 based index
        start_index = start_page - 1
        end_index = start_index + num_pages - 1

        if start_index >= total_pages:
            raise ValueError(
                f"Start page {start_page} exceeds the total pages ({total_pages})."
            )

        #making sure we donot exceed the document length
        end_index = min(end_index, total_pages - 1)

        extracted_text = []

        for i in range(start_index, end_index + 1):
            page = doc.load_page(i)
            text = page.get_text("text")

            # adding page separators for clarity
            extracted_text.append(
                f"\n\n===== PAGE {i+1} =====\n{text}".strip()
            )

    # writing extracted content into output file
    with open(outer_txt_path, "w", encoding="utf-8") as f:
        f.write("\n".join(extracted_text))

    return start_page, end_index + 1
```

```
#extracting the pages 2980-2989
p1_start, p1_end = extract_pages_to_txt(
    pdf_path=pdf_path,
    start_page=2980,
    num_pages=10,
    outer_txt_path="file1.txt"
)

#extracting pages 102-111
p2_start, p2_end = extract_pages_to_txt(
    pdf_path=pdf_path,
    start_page=102,
    num_pages=10,
```

```
    outer_txt_path="file2.txt"
)

print(f"file1.txt created for pages {p1_start}–{p1_end}")
print(f"file2.txt created for pages {p2_start}–{p2_end}")
```

```
file1.txt created for pages 2980–2989
file2.txt created for pages 102–111
```

```
#quick previewing to verify if extraction worked correctly
def preview_txt(path: str, characters: int = 1000):
    with open(path, "r", encoding="utf-8") as f:
        content = f.read(characters)
    print(f"\nPreview of {path}:\n")
    print(content)

preview_txt("file1.txt")
preview_txt("file2.txt")
```

```
Preview of file1.txt:

===== PAGE 2980 =====
Many of those sitting around Yaxley looked impressed; his neighbor,
Dolohov, a man with a long, twisted face, clapped him on the back.
“It is a start,” said Voldemort. “But Thicknesse is only one man.
Scrimgeour must be surrounded by our people before I act. One failed attempt
on the Minister’s life will set me back a long way.”
“Yes — my Lord, that is true — but you know, as Head of the Department
of Magical Law Enforcement, Thicknesse has regular contact not only with
the Minister himself, but also with the Heads of all the other Ministry
departments. It will, I think, be easy now that we have such a high-ranking
official under our control, to subjugate the others, and then they can all work
together to bring Scrimgeour down.”
“As long as our friend Thicknesse is not discovered before he has
converted the rest,” said Voldemort. “At any rate, it remains unlikely that the
Ministry will be mine before next Saturday. If we cannot touch the boy at his
destination, th

Preview of file2.txt:

===== PAGE 102 =====
“Er — I don’t know any,” Harry confessed.
“What!” Ron looked dumbfounded. “Oh, you wait, it’s the best game in the
world —” And he was off, explaining all about the four balls and the
positions of the seven players, describing famous games he’d been to with his
brothers and the broomstick he’d like to get if he had the money. He was just
taking Harry through the finer points of the game when the compartment door
slid open yet again, but it wasn’t Neville the toadless boy, or Hermione
Granger this time.
Three boys entered, and Harry recognized the middle one at once: It was
the pale boy from Madam Malkin’s robe shop. He was looking at Harry with
a lot more interest than he’d shown back in Diagon Alley.
“Is it true?” he said. “They’re saying all down the train that Harry Potter’s
in this compartment. So it’s you, is it?”
“Yes,” said Harry. He was looking at the other boys. Both of them were
thickset and looked extremely mean. Standing on either side of the pale boy,
```

```
#downloading the generated text files
files.download("file1.txt")
files.download("file2.txt")
```

Second Task

```
#installing MRJob
!pip -q install mrjob
```

```
#grabbing a clean English word list
!wget -q -O english_words.txt https://raw.githubusercontent.com/dwyl/english-words/master/words_alpha.txt

#sanity check
!wc -l english_words.txt
```

```
370105 english_words.txt
```

```
%%writefile mr_wordcount.py
#!/usr/bin/env python3
#word count using MRJob (MapReduce)
#DOB: 07/13/2002
```

```
import re
from mrjob.job import MRJob

WORD_Regex = re.compile(r"[A-Za-z]+(?:'[A-Za-z]+)?")

class MRWordCount(MRJob):

    def mapper(self, _, line):
        for w in WORD_Regex.findall(line):
            yield w.lower(), 1

    def combiner(self, word, counts):
        yield word, sum(counts)

    def reducer(self, word, counts):
        yield word, sum(counts)

if __name__ == "__main__":
    MRWordCount.run()
```

```
Overwriting mr_wordcount.py
```

```
#running the task 1 (MRJob local runner)
!python mr_wordcount.py file1.txt > wordcount_file1_Output.txt

#showing the top lines (proof of output)
!head -n 30 wordcount_file1_Output.txt
```

```
No configs found; falling back on auto-configuration
No configs specified for inline runner
Creating temp directory /tmp/mr_wordcount.root.20260226.215206.098795
Running step 1 of 1...
job output is in /tmp/mr_wordcount.root.20260226.215206.098795/output
Streaming final output from /tmp/mr_wordcount.root.20260226.215206.098795/output...
Removing temp directory /tmp/mr_wordcount.root.20260226.215206.098795...
"a"     53
"aberforth"     3
"about" 6
"above" 2
"absently"      1
"accept"        1
"accorded"      1
"according"     1
"across"        4
"act"   1
"action"        1
"adalbert"      1
"addressing"    1
"admit" 1
"admitted"      1
"advantage"     1
"afraid"        1
"after" 4
"again" 9
"against"       1
"age"   1
"ago"   1
"ah"    1
"ahead" 1
"album" 1
"albus" 10
"alchemist"     1
"alike" 1
"all"   8
"allowed"       1
```

```
%%writefile mr_nonenglish.py
#!/usr/bin/env python3
#Task 2: non english word count using MRJob (MapReduce)
#DOB: 07/13/2002
import re
from mrjob.job import MRJob

WORD_Regex = re.compile(r"[A-Za-z]+(?:'[A-Za-z]+)?")

class MRNonEnglish(MRJob):

    def configure_args(self):
        super().configure_args()
        #shipping english_words.txt along with the job
        self.add_file_arg("--dict", default="english_words.txt", help="English dictionary file")

    def mapper_init(self):
        self.english = set()
```

```
            with open(self.options.dict, "r", encoding="utf-8", errors="ignore") as f:
                for line in f:
                    self.english.add(line.strip().lower())

        def mapper(self, _, line):
            for w in WORD_Regex.findall(line):
                w = w.lower()

                #filtering tiny junk tokens
                if len(w) < 3:
                    continue

                #skipping if it's in dictionary, it's english
                if w in self.english:
                    continue

                yield w, 1

        def combiner(self, word, counts):
            yield word, sum(counts)

        def reducer(self, word, counts):
            yield word, sum(counts)

    if __name__ == "__main__":
        MRNonEnglish.run()
```

```
Overwriting mr_nonenglish.py
```

```
#running task 2 (MRJob local runner)
!python mr_nonenglish.py file2.txt --dict english_words.txt > non_english_file2_output.txt

#showing top lines
!head -n 40 non_english_file2_output.txt
```

```
No configs found; falling back on auto-configuration
No configs specified for inline runner
Creating temp directory /tmp/mr_nonenglish.root.20260226.215206.855249
Running step 1 of 1...
job output is in /tmp/mr_nonenglish.root.20260226.215206.855249/output
Streaming final output from /tmp/mr_nonenglish.root.20260226.215206.855249/output...
Removing temp directory /tmp/mr_nonenglish.root.20260226.215206.855249...
"crabbe"        3
"diagon"        2
"dursleys"      2
"gringotts"     1
"gryffindor"    2
"gryffindors"   1
"malfoy"        8
"mcgonagall"    10
"oooooh"        1
"ravenclaw"     2
"scabbers"      6
"slytherin"     2
"weasleys"      2
"wizarding"     1
"hadn"  1
"hogwarts"      7
"hufflepuff"    3
"hufflepuffs"   1
```