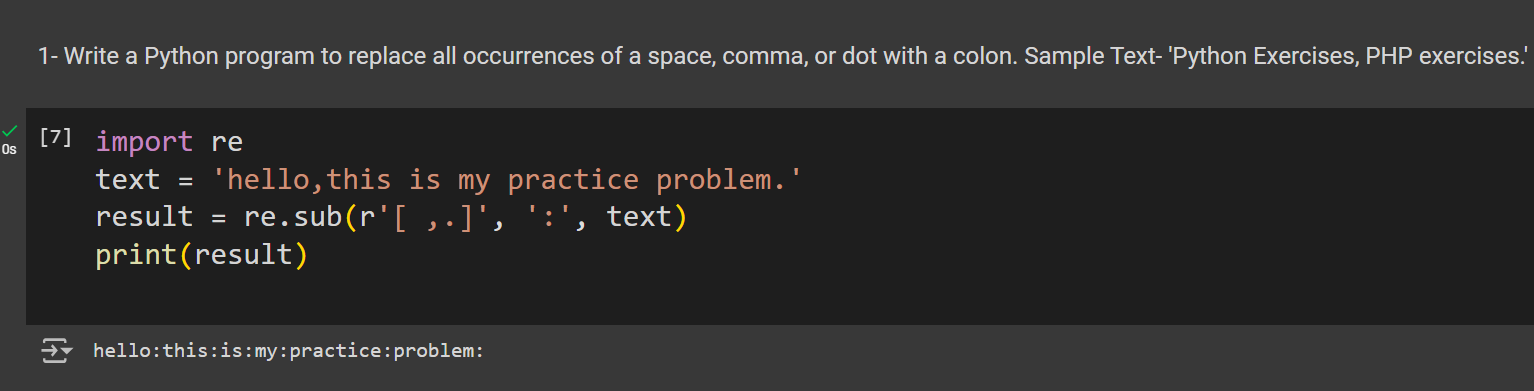
**CD SOLUTION**

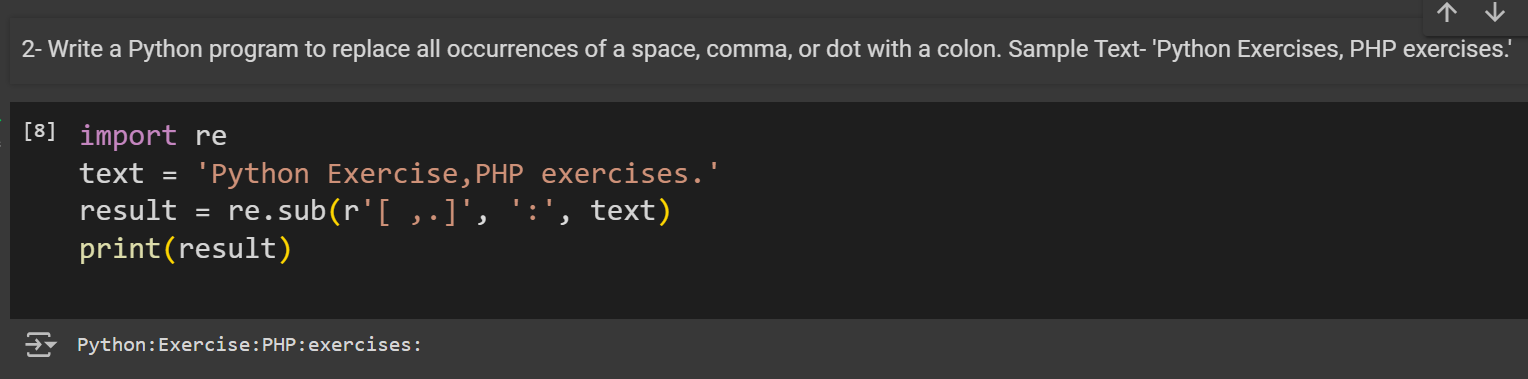
**1) Regular Expression:**

* r"[ ,.]"
  + This pattern matches any space ( ), comma (,), or dot (.)

****

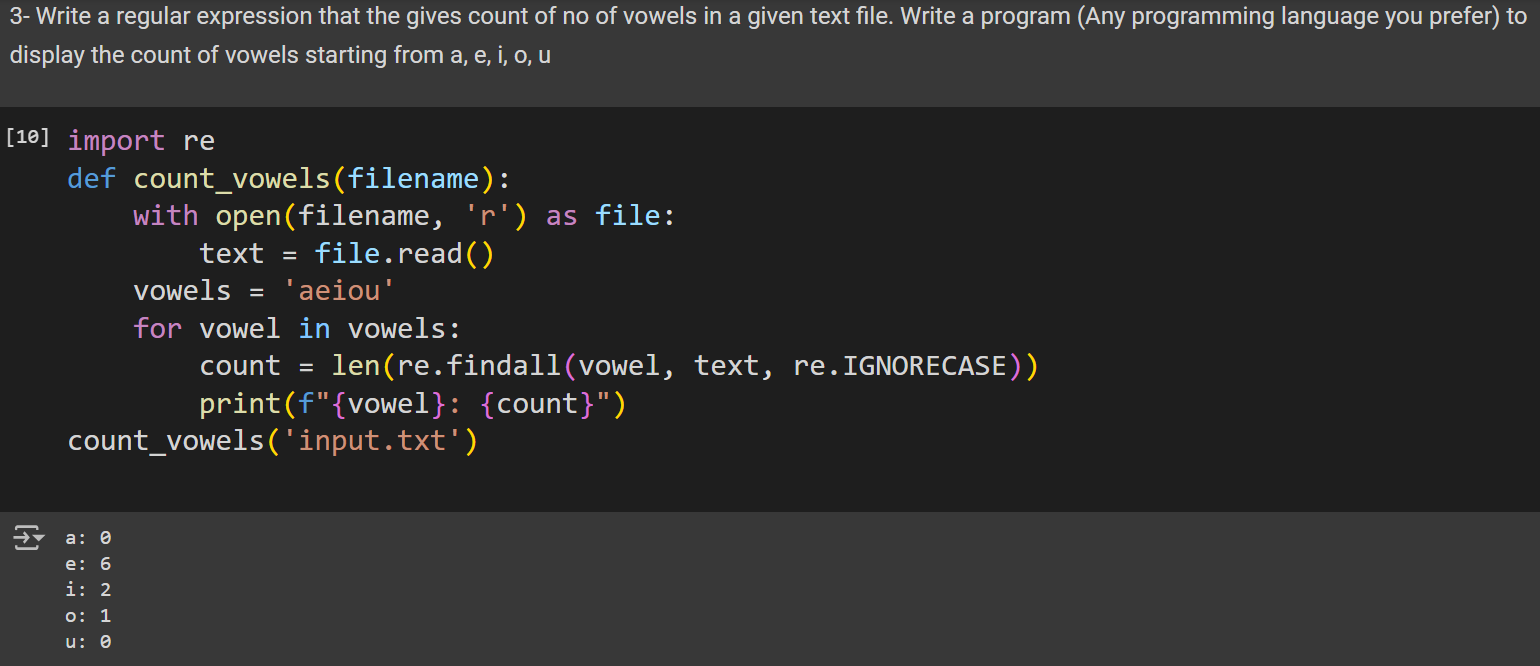
**2) Regular Expression:**

* r"[ ,.]"
  + This pattern matches any space, comma, or dot**.**

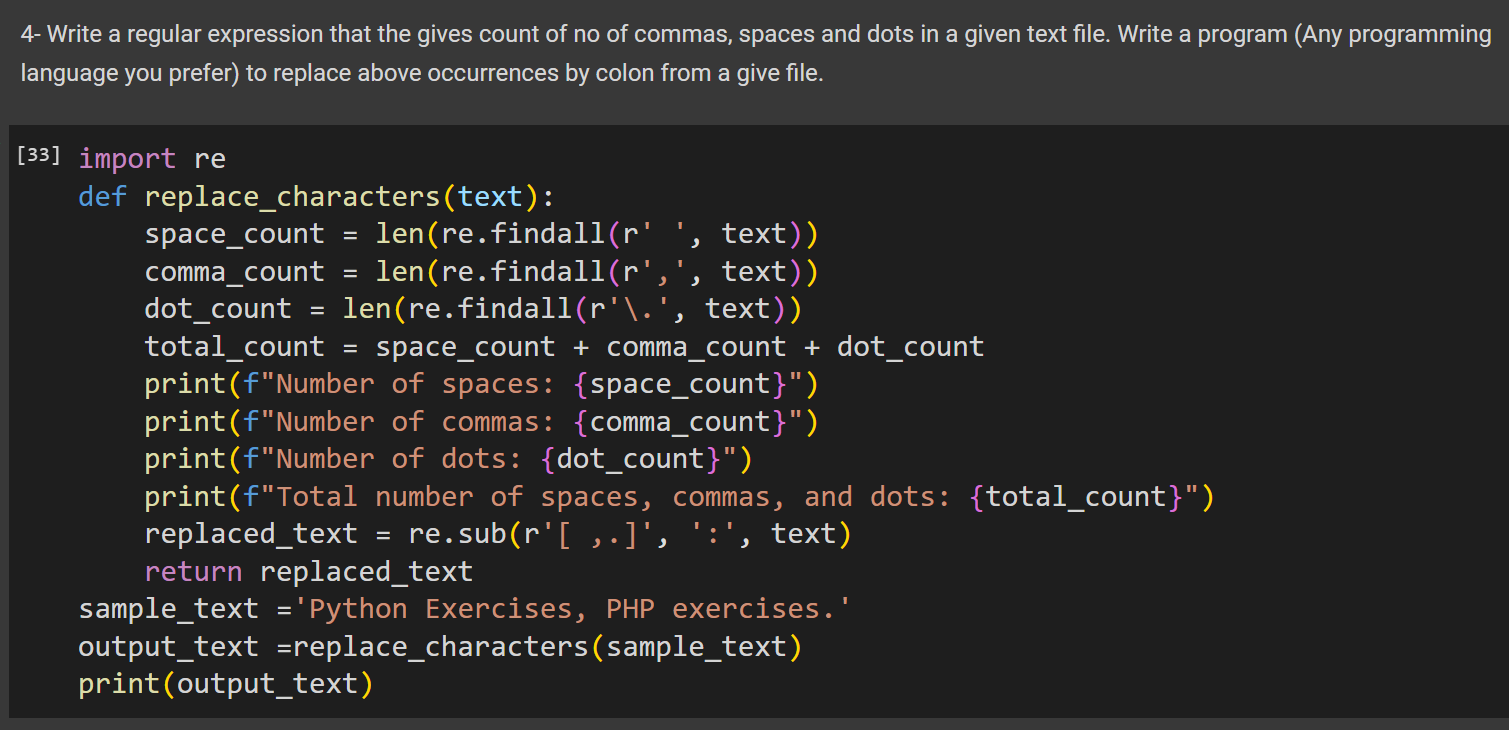
****

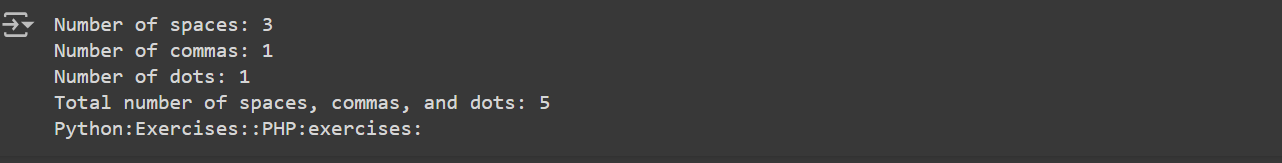
**3) Regular Expression:**

* r"[aeiouAEIOU]"
  + This pattern matches any vowel, both lowercase and uppercase.

****

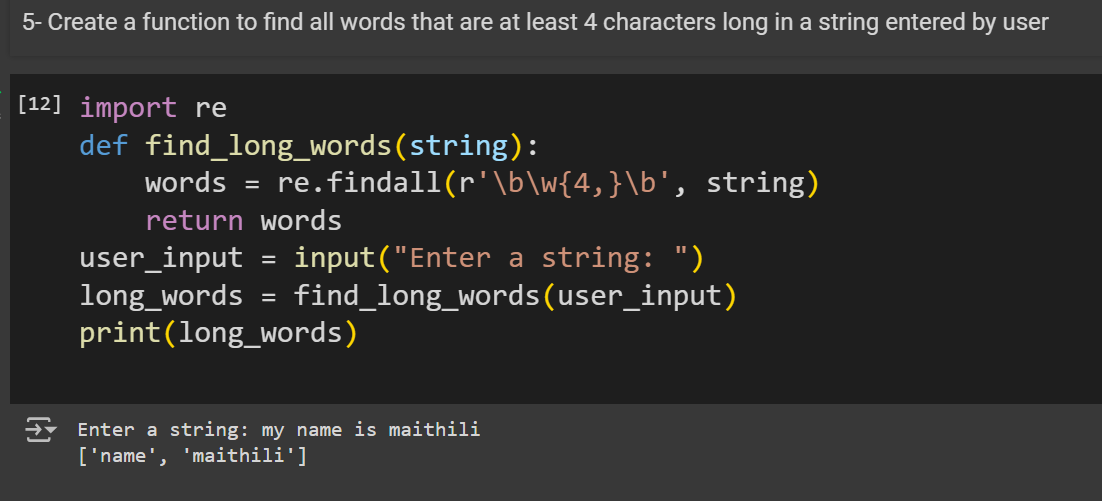
**4) Regular Expression:**

* r”[ ,.]”
  + This pattern that the gives count of no of commas, spaces and dots in a given text file.****

****

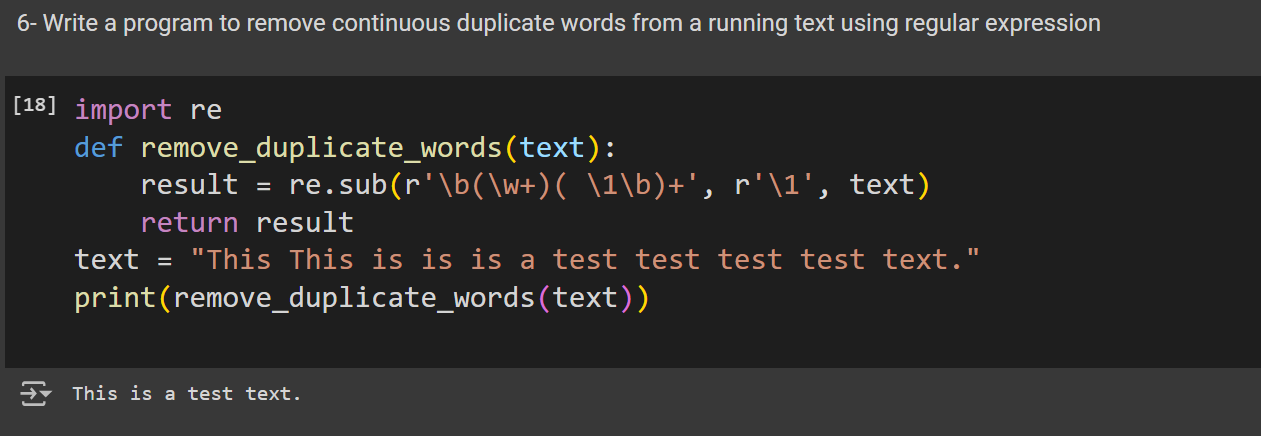
**5) Regular Expression:**

* r"\b\w{4,}\b"
  + This pattern matches words that are at least 4 characters long.

****

**6) Regular Expression:**

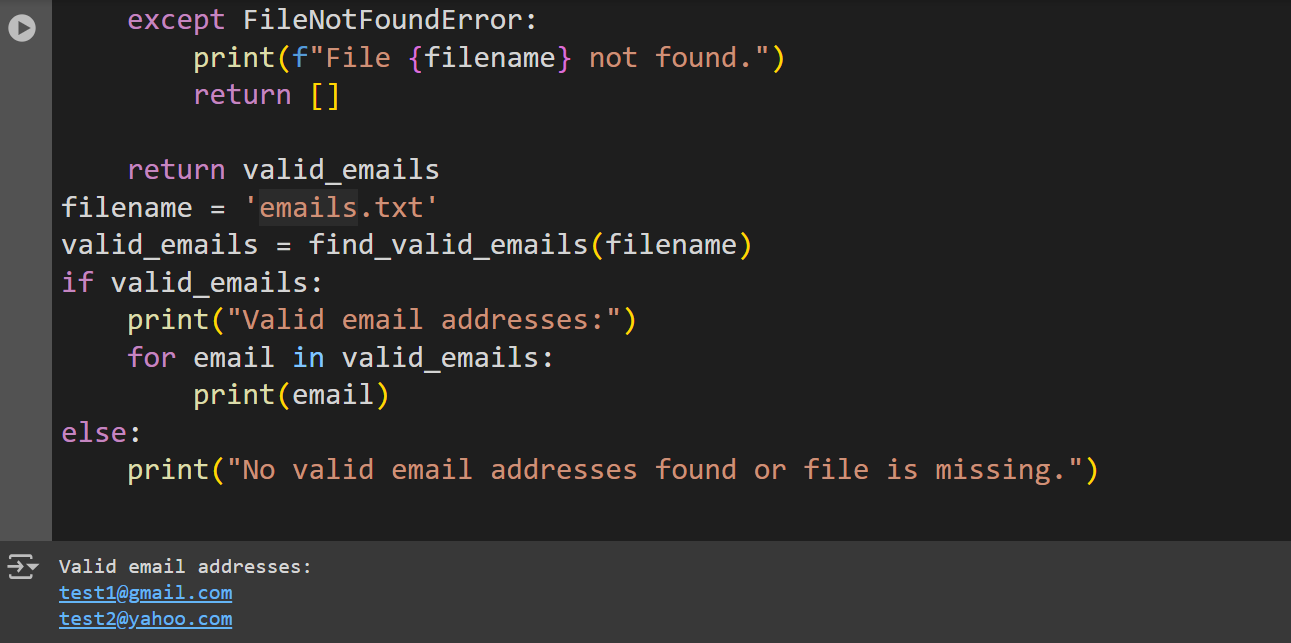
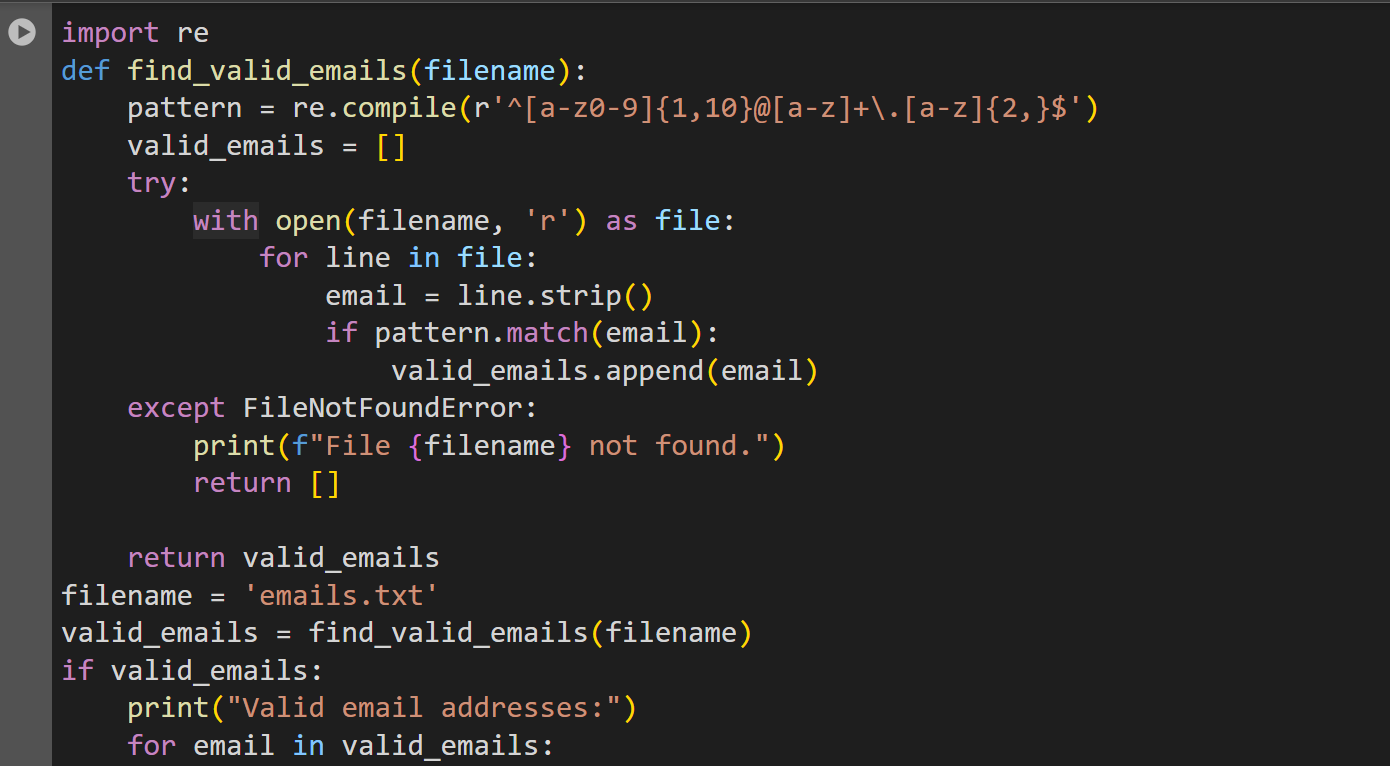
* r"\b(\w+)\s+\1\b"
  + This pattern matches a word followed by the same word.

****

**7) Regular Expression:**

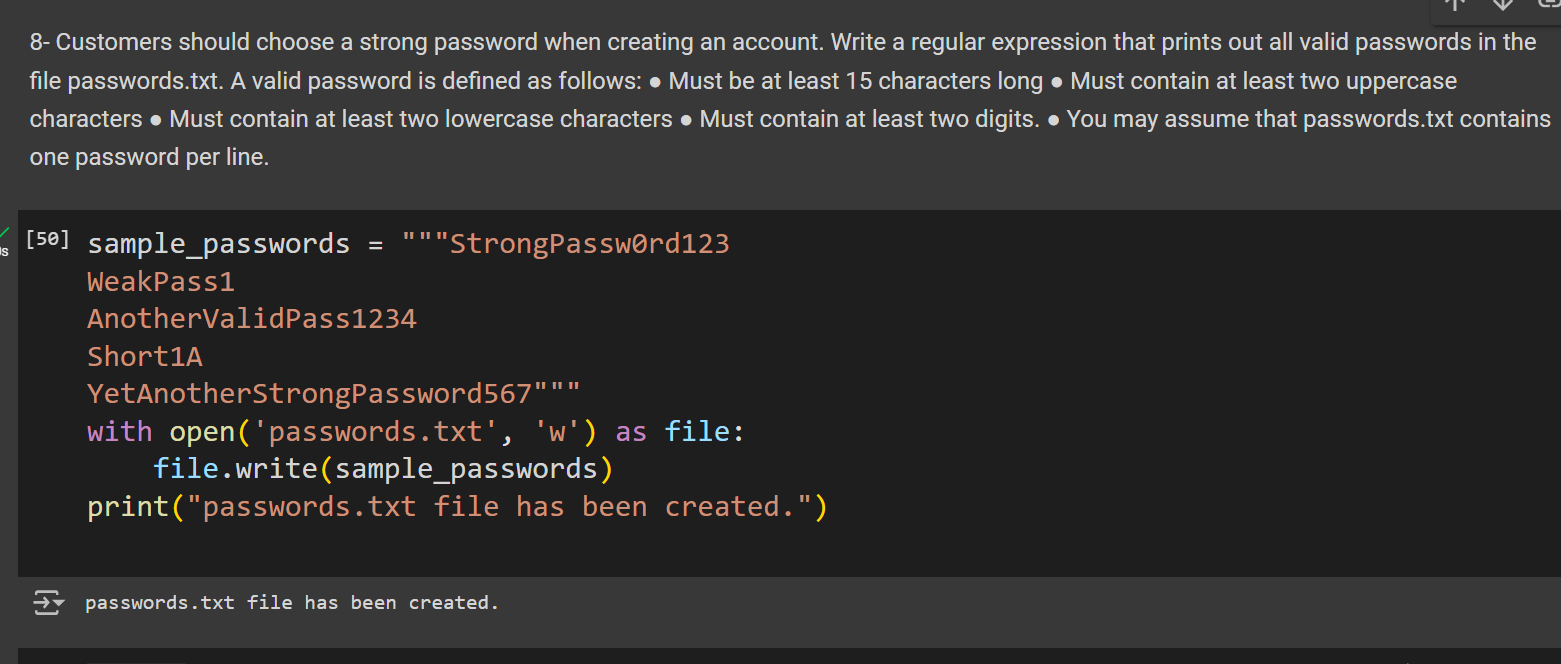
* r"\b[a-z0-9]{1,10}@[a-z]+\.[a-z]{2,}\b"
  + This pattern matches a valid email address as per the criteria given.

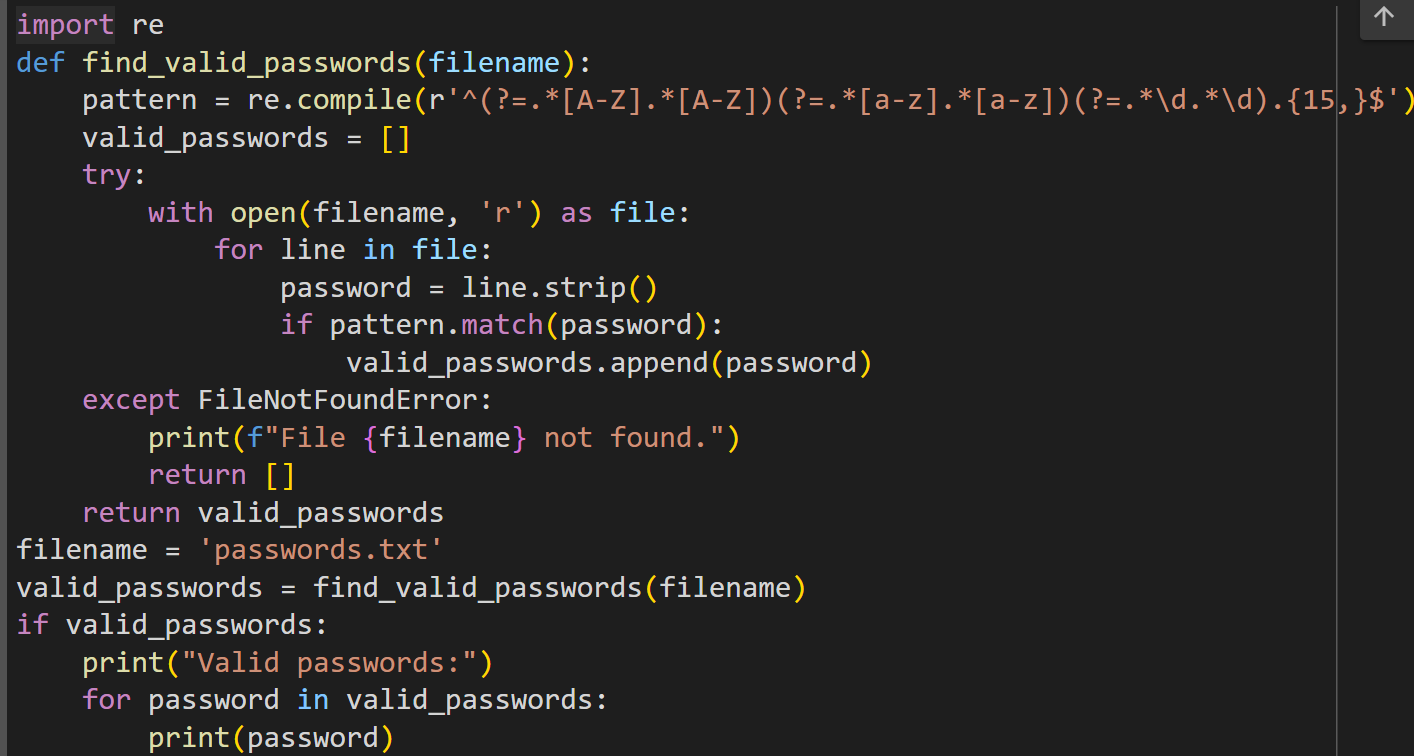
****

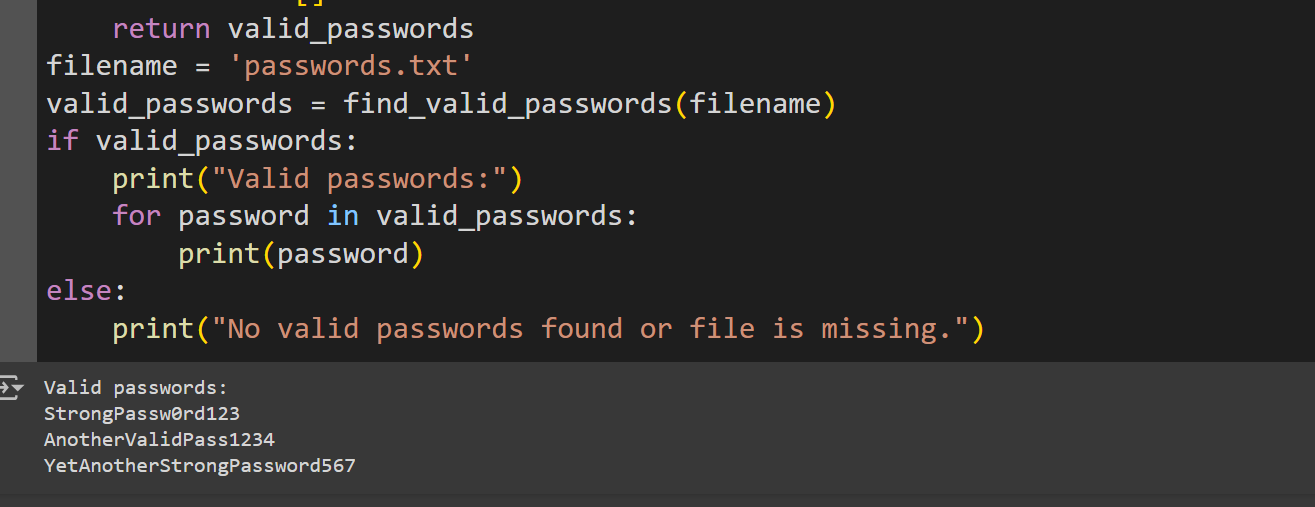
****

**8) Regular Expression:**

* r"(?=.\*[A-Z].\*[A-Z])(?=.\*[a-z].\*[a-z])(?=.\*\d.\*\d).{15,}"

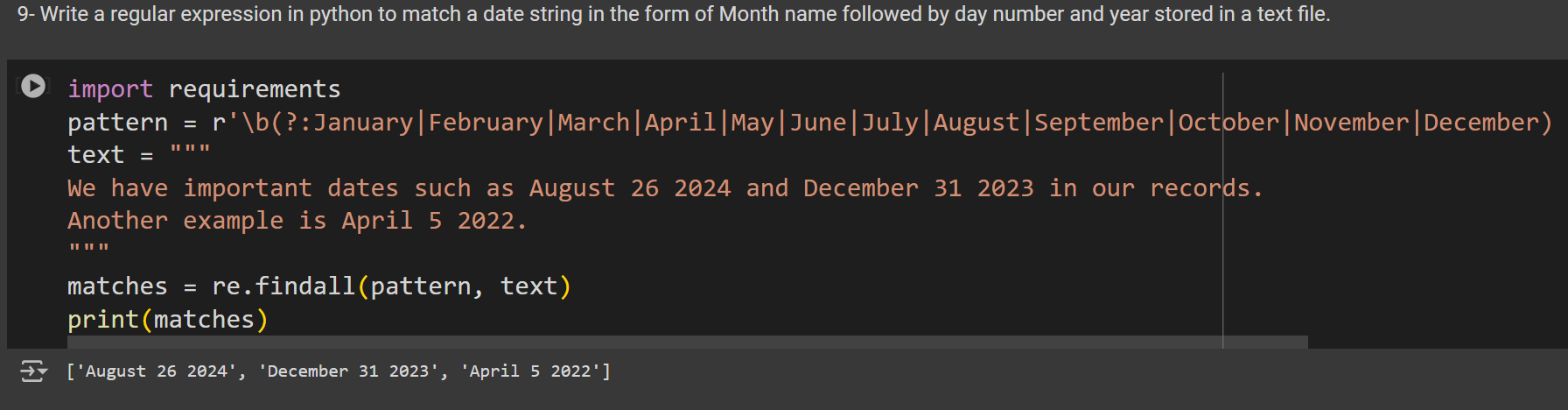
****

****

****

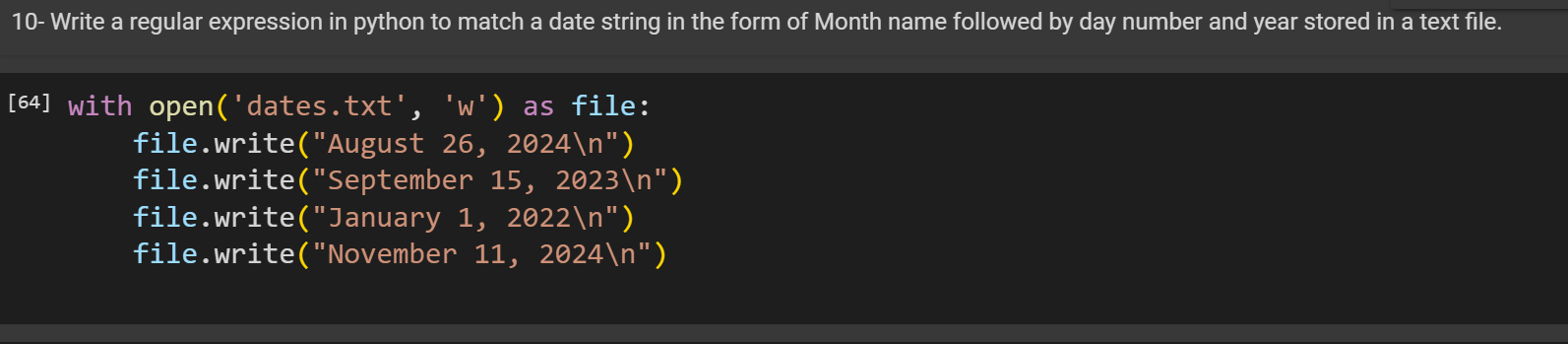
**9)Regular Expression:**

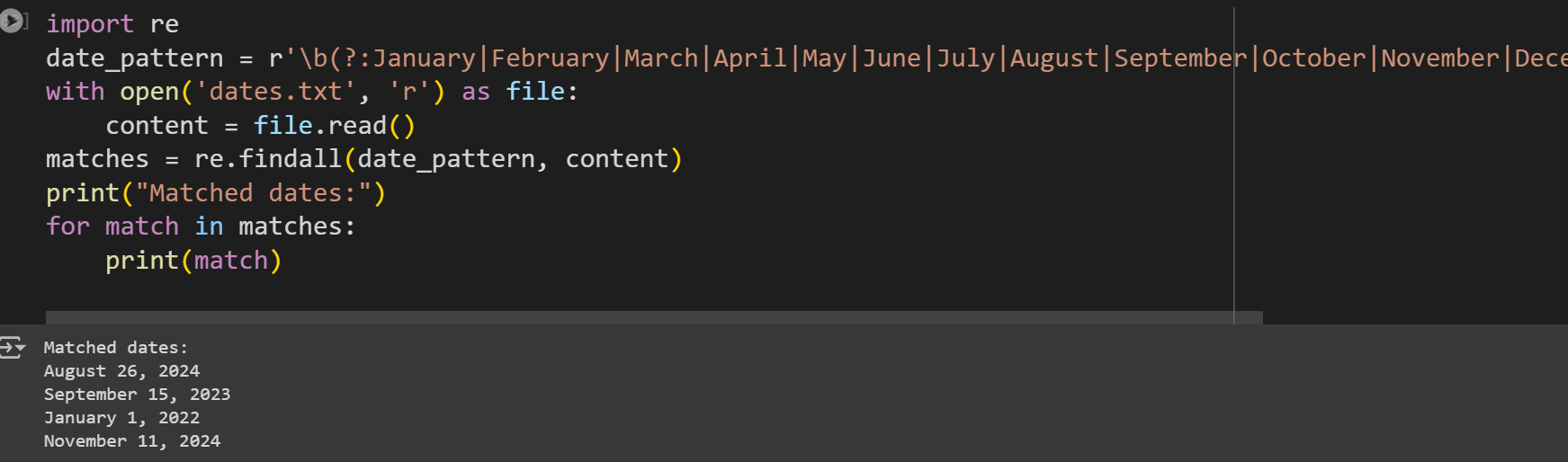
* r'\b(?:January|February|March|April|May|June|July|August|September|October|November|December) \d{1,2} \d{4}\b'

****

**10)Regular Expression:**

* r'\b(?:January|February|March|April|May|June|July|August|September|October|November|December) \d{1,2} \d{4}\b'

****

****

**CONCLUSION:**

By leveraging regular expressions in Python, you can effectively extract and validate dates from a text file, ensuring they adhere to a specific format. This method is particularly useful for parsing and processing date information in various applications.