Objective: Develop a Python program that analyzes text input, counts word frequencies, and identifies patterns using regular expressions.

Features to implement: 1. Read text from a file or user input 2. Count word frequencies 3. Find specific patterns using regular expressions 4. Display results in a user-friendly format

Suggested Implementation Steps:

1. Text Input:
    - Allow users to input text directly or specify a file to read from
2. Word Frequency Counter:
    - Tokenize the text into words
    - Remove punctuation and convert to lowercase for consistency
    - Use a dictionary to count occurrences of each word
    - Sort words by frequency
3. Pattern Finder:
    - Implement several pre-defined regex patterns (e.g., email addresses, URLs, dates)
    - Allow users to input custom regex patterns
    - Use `re` module to find all matches in the text
4. Results Display:
    - Show top N most frequent words
    - Display found patterns with their counts

Example Usage:

```
Welcome to the Text Analyzer!

1. Enter text manually
2. Read text from file
Enter your choice: 2

Enter file name: sample_text.txt

Text loaded successfully!

1. Count word frequencies
2. Find patterns
3. Exit
Enter your choice: 1

Top 10 most frequent words:
1. the (50 occurrences)
2. and (30 occurrences)
3. to (25 occurrences)
```

```
...

Enter your choice: 2

Select pattern to find:
1. Email addresses
2. URLs
3. Dates (YYYY-MM-DD format)
4. Custom regex pattern
Enter your choice: 1

Found 5 email addresses:
1. user@example.com
2. info@company.com
...

Enter a custom regex pattern or press Enter to go back: \b\d{3}-\d{3}-\d{4

Found 3 matches:
1. 123-456-7890
2. 987-654-3210
...

Enter your choice: 3

Thank you for using the Text Analyzer!
```

Key Components:

1. Text Processing:

```
import re
from collections import Counter

def process_text(text):
    # Remove punctuation and convert to lowercase
    text = re.sub(r'[^\w\s]', '', text.lower())
    return text.split()

def count_words(words):
    return Counter(words)
```

2. Pattern Matching:

```
def find_pattern(text, pattern):
    return re.findall(pattern, text)

# Predefined patterns
patterns = {
    'email': r'\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b',
    'url': r'http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[!*\\(\\),]|(?:%[0
```

```
    'date': r'\d{4}-\d{2}-\d{2}'
}
```

3. Results Display:

```
def display_word_frequencies(word_counts, n=10):
    for word, count in word_counts.most_common(n):
        print(f"{word}: {count}")

def display_pattern_matches(matches):
    for i, match in enumerate(matches, 1):
        print(f"{i}. {match}")
```

Learning Outcomes: - Practice file I/O operations in Python - Gain experience with text processing and tokenization - Learn to use the `collections` module, specifically `Counter` - Develop skills in using regular expressions with the `re` module - Implement a command-line interface for user interaction - Work with dictionaries to store and manipulate data - Practice sorting and displaying data in a formatted manner