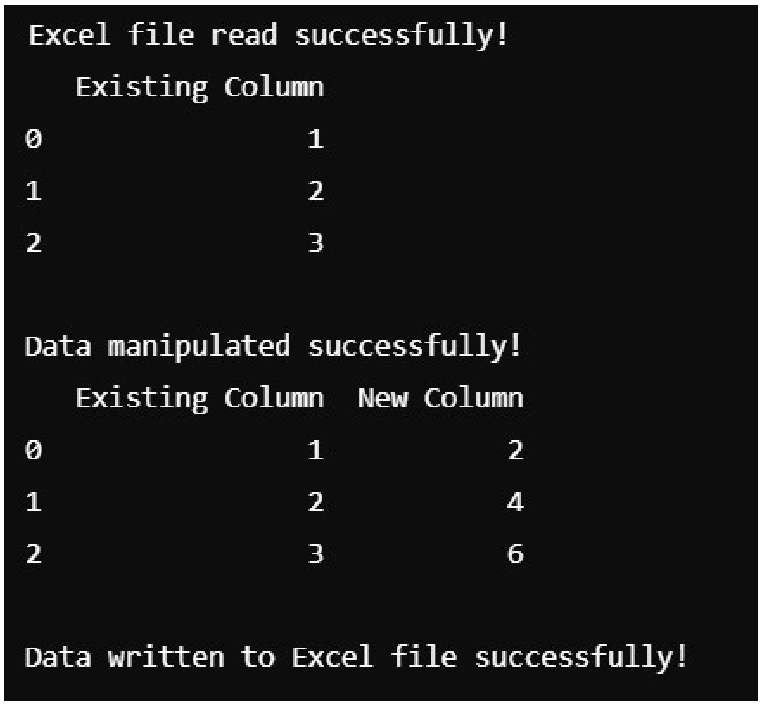
Practical-8

AIM:- Python program to perform file operation on Excel Data Sheet

```
import pandas as pd

def read("D:\Programming\Datasets\Dataset_1\dataset_1.xlsx"):
    try:
        data = pd.read_excel(file_path)
        print("Excel file read successfully!")
        print(data.head())
        return data
    except Exception as e:
        print("Error reading the Excel file:", e)

def write(data, output_path):
    try:
        data.to_excel(output_path, index=False)
        print("Data written to Excel file successfully!")
    except Exception as e:
        print("Error writing to the Excel file:", e)

def manipulate_data(data):
    data['New Column'] = data['Existing Column'] * 2
    print("Data manipulated successfully!")
    print(data.head())
    return data

def main():
    file_path = 'input.xlsx'
    output_path = 'output.xlsx'
    data = read(file_path)
    if data is not None:
        modified_data = manipulate_data(data)
        write(modified_data, output_path)

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

OUTPUT:-



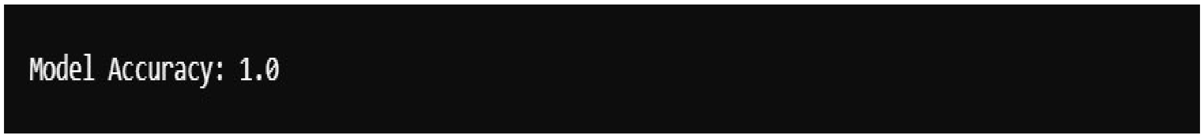
```
Excel file read successfully!
   Existing Column
0                1
1                2
2                3

Data manipulated successfully!
   Existing Column  New Column
0                1           2
1                2           4
2                3           6

Data written to Excel file successfully!
```

Practical-9

AIM: Python program to implement Python Sci Kit Learn & NLTK.

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
texts = [
    "I love programming in Python",
    "Python is great for data science",
    "I enjoy learning new languages",
    "Data science is fascinating",
    "I love studying machine learning"
]
labels = [1, 1, 0, 1, 1]

vectorizer = CountVectorizer(stop_words='english')
X = vectorizer.fit_transform(texts)

X_train, X_test, y_train, y_test = train_test_split(X, labels, test_size=0.2, random_state=42)

classifier = MultinomialNB()
classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print("Model Accuracy:", accuracy)
```

OUTPUT:-



```
Model Accuracy: 1.0
```

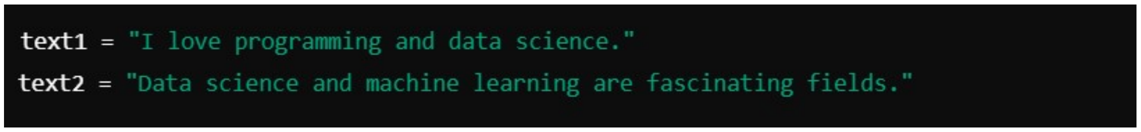
Practical-10

AIM: Python program to implement with python NLTK/Spicy/Py NLPI

```
import nltk
import spacy
from scipy.spatial.distance import cosine

nltk.download('stopwords')
nltk.download('punkt')

from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

nlp = spacy.load("en_core_web_md")

def preprocess_text(text):
    stop_words = set(stopwords.words('english'))
    words = word_tokenize(text.lower())
    return ' '.join(word for word in words if word.isalnum() and word not in stop_words)

def cosine_similarity(text1, text2):
    vec1 = nlp(text1).vector
    vec2 = nlp(text2).vector
    return 1 - cosine(vec1, vec2)

text1 = "I love programming and data science."
text2 = "Data science and machine learning are fascinating fields."

processed_text1 = preprocess_text(text1)
processed_text2 = preprocess_text(text2)
```

```
similarity = cosine_similarity(processed_text1, processed_text2)
print("Cosine Similarity between text1 and text2:", similarity)
```

OUTPUT:-



```
text1 = "I love programming and data science."
text2 = "Data science and machine learning are fascinating fields."
```