**4.Using pandas, write a Python function to clean and preprocess a given DataFrame, which involves handling missing values, normalizing numerical columns, and encoding categorical columns.**

import pandas as pd

from sklearn.preprocessing import StandardScaler, LabelEncoder

data = {

'Name': ['John', 'Anna', 'Peter', 'Linda', 'Tom'],

'Age': [28, 35, None, 41, 29],

'Gender': ['M', 'F', 'M', 'F', 'M'],

'Height': [175.2, 162.5, 180.0, None, 170.8],

'Weight': [70.0, None, 85.5, 68.0, 75.5],

'Income': [50000, 75000, 60000, 80000, 45000]

}

df = pd.DataFrame(data)

def preprocess\_data(df):

# Handling missing values

df.fillna(method='ffill', inplace=True)

# Normalizing numerical columns

numeric\_cols = df.select\_dtypes(include=['float64', 'int64']).columns

scaler = StandardScaler()

df[numeric\_cols] = scaler.fit\_transform(df[numeric\_cols])

# Encoding categorical columns

categorical\_cols = df.select\_dtypes(include=['object']).columns

label\_encoders = {}

for col in categorical\_cols:

le = LabelEncoder()

df[col] = le.fit\_transform(df[col])

label\_encoders[col] = le

return df, label\_encoders

# Preprocess the DataFrame

df\_cleaned, encoders = preprocess\_data(df)

# Display the cleaned and preprocessed DataFrame

print("Cleaned and Preprocessed DataFrame:")

print(df\_cleaned)

# Example of using label encoders to inverse transform

print("\nExample of inverse transformation:")

for col, encoder in encoders.items():

print(f"{col}: {list(encoder.inverse\_transform(df\_cleaned[col]))}")

**OUTPUT**:

