```
import numpy as np
import pandas as pd

def calculate_fire_danger_nfdrs4(temperature, Tmax, relative_humidity,
RHmin, precipitation, wind_speed_mph):
    """
    Calculate fire danger indices based on input variables following
NFDRS4.

    Parameters:
        temperature (float): Temperature in degrees Celsius.
        Tmax (float): Maximum daily temperature in degrees Celsius.
        relative_humidity (float): Relative humidity as a percentage.
        RHmin (float): Minimum relative humidity as a percentage.
        precipitation (float): Precipitation in inches.
        wind_speed_mph (float): Wind speed in miles per hour.

    Returns:
        pd.DataFrame: A dataframe containing calculated fire danger
indices.
    """
    # Fine Fuel Moisture Code (FFMC) Calculation
    ffmc = 0.0
    if precipitation > 0:
        ffmc = max(0.0, 101.0 - (0.5 * precipitation))
    else:
        ffmc = max(0.0, 101.0 - (0.25 * (101.0 - 85.0)))  # Default
drying rate example

    # Duff Moisture Code (DMC)
    dmc = max(0.0, 0.92 * temperature * (100 - relative_humidity) /
100.0)

    # Drought Code (DC) Calculation
    # Initialize DC to 250 on the first day
    dc = 250.0
    dc = dc + (0.025 * (Tmax - 10) * (100 - RHmin))  # Formula for DC
based on Tmax and RHmin

    # Function to calculate Keetch-Byram Drought Index (KBDI)
    def calculate_kbdi(Tmax, rainfall, prev_kbdi=0):
        """
        Function to calculate KBDI (Keetch-Byram Drought Index).

        :param Tmax: Maximum daily temperature in Celsius (°C)
        :param rainfall: Daily rainfall in millimeters (mm)
        :param prev_kbdi: KBDI from the previous day (defaults to 0 if
first day)

        :return: Updated KBDI for the day
```

```
        """
        # Initialize constants
        max_kbdi = 800  # Maximum KBDI value, changeable based on
region

        if rainfall > 0:
            # If there is rainfall, KBDI is reduced based on rainfall
            new_kbdi = max(0, prev_kbdi - (0.2 * rainfall))
        else:
            # If no rainfall, KBDI increases based on Tmax
            new_kbdi = max(0, prev_kbdi + ((Tmax - 10) * 0.3))  #
Adjusted for Celsius temperature

        # Cap the KBDI at the maximum value
        new_kbdi = min(new_kbdi, max_kbdi)

        return new_kbdi

    keetch_byram_drought_index = calculate_kbdi(Tmax, precipitation,
prev_kbdi=250)

    # Drought Factor (DF) Calculation from KBDI
    def kbdi_to_df(kbdi):
        """Estimates Drought Factor (DF) from KBDI. This is an
approximation."""
        df = round(kbdi / 100)
        return max(0, min(8, df))  # Restrict DF to 0-8 range

    df = kbdi_to_df(keetch_byram_drought_index)

    # Spread Component (SC)
    spread_component = 0.0
    if ffmc >= 85.0:
        spread_component = 0.208 * wind_speed_mph * np.exp(0.05039 *
ffmc)

    # Apply scaling factor to the spread component to prevent large
values
    spread_component *= 0.01  # Adjust this scaling factor if needed

    # Buildup Index (BI)
    buildup_index = dmc + dc
    # Apply scaling factor to avoid extremely large BI
    buildup_index *= 0.05  # Adjust scaling factor for BI

    # Burning Index (BI)
    burning_index = 0.1 * spread_component * buildup_index
    # Apply scaling factor to prevent the Burning Index from being too
large
    burning_index *= 0.5  # Adjust this scaling factor if needed
```

```
    # Function to calculate FFDI (Forest Fire Danger Index) using
correct formula
    def calculate_ffdi(Tmax, RHmin, wind_speed, fuel_moisture=0.12):
        """
        Function to calculate FFDI (Forest Fire Danger Index).

        :param Tmax: Maximum daily temperature in Celsius (°C)
        :param RHmin: Minimum daily relative humidity (%)
        :param wind_speed: Wind speed in kilometers per hour (km/h)
        :param fuel_moisture: Fuel moisture content (default 12%)

        :return: FFDI value for the day
        """
        # Constants used in FFDI calculation
        a = 0.027
        b = 0.075

        # Convert wind speed from mph to km/h
        wind_speed_kmh = wind_speed * 1.60934

        # Step 2: Calculate FFDI using the correct formula
        ffdi = a * (Tmax ** 2) * (100 - RHmin) * (wind_speed_kmh **
0.5) * (fuel_moisture ** b)

        # Apply scaling factor to FFDI to bring it within a realistic
range
        ffdi *= 0.01  # Adjust this scaling factor for FFDI

        return ffdi

    forest_fire_danger_index = calculate_ffdi(Tmax, RHmin,
wind_speed_mph)

    # Create a dataframe to structure the results
    data = {
        "FFMC": [round(ffmc, 2)],
        "DMC": [round(dmc, 2)],
        "DC": [round(dc, 2)],
        "KBDI": [round(keetch_byram_drought_index, 2)],
        "Wind Speed (mph)": [round(wind_speed_mph, 2)],
        "Buildup Index": [round(buildup_index, 2)],
        "Spread Component (SC)": [round(spread_component, 2)],
        "Burning Index (BI)": [round(burning_index, 2)],
        "Forest Fire Danger Index (FFDI)":
[round(forest_fire_danger_index, 2)]
    }

    return pd.DataFrame(data)
```

```python
# Example usage in Jupyter Notebook or VS Code
# Input dataframe (temperature, Tmax, relative_humidity, RHmin,
precipitation, wind_speed_mph)
weather_data = pd.DataFrame({
    "Date": ["2024-07-15", "2024-07-16", "2024-07-17", "2024-07-28"],
    "Temperature (°C)": [25, 28, 30, 25],
    "Maximum Temperature (°C)": [27, 29, 30, 27],
    "Relative Humidity (%)": [60, 55, 50, 50],
    "Minimum Relative Humidity (%)": [50, 45, 40, 45],
    "Precipitation (inches)": [0, 0, 0.1, 0],
    "Wind Speed (mph)": [15, 20, 25, 10]
})

# Process each row and calculate indices
results = []
for _, row in weather_data.iterrows():
    temperature = row["Temperature (°C)"]
    Tmax = row["Maximum Temperature (°C)"]
    relative_humidity = row["Relative Humidity (%)"]
    RHmin = row["Minimum Relative Humidity (%)"]
    precipitation = row["Precipitation (inches)"]
    wind_speed_mph = row["Wind Speed (mph)"]

    result = calculate_fire_danger_nfdrs4(temperature, Tmax,
relative_humidity, RHmin, precipitation, wind_speed_mph)
    results.append(result)

# Combine results into a single dataframe
results_df = pd.concat(results, ignore_index=True)

# Add weather data to results
final_df = pd.concat([weather_data, results_df], axis=1)

# Display results
final_df
```

```
         Date  Temperature (°C)  Maximum Temperature (°C)  \
0  2024-07-15                25                        27
1  2024-07-16                28                        29
2  2024-07-17                30                        30
3  2024-07-28                25                        27

   Relative Humidity (%)  Minimum Relative Humidity (%)  \
0                     60                             50
1                     55                             45
2                     50                             40
3                     50                             45

   Precipitation (inches)  Wind Speed (mph)    FFMC    DMC      DC
KBDI  \
```

```
0                      0.0                 15   97.00   9.20  271.25
255.10
1                      0.0                 20   97.00  11.59  276.12
255.70
2                      0.1                 25  100.95  13.80  280.00
249.98
3                      0.0                 10   97.00  11.50  273.38
255.10

   Wind Speed (mph)  Buildup Index  Spread Component (SC)  Burning
Index (BI)  \
0                15          14.02                    4.14
2.90
1                20          14.39                    5.52
3.97
2                25          14.69                    8.42
6.18
3                10          14.24                    2.76
1.97

   Forest Fire Danger Index (FFDI)
0                           41.24
1                           60.44
2                           78.88
3                           37.04
```