```python
from statsmodels.stats.outliers_influence import variance_inflation_factor

def calc_vif(X):

    # Calculating VIF
    vif = pd.DataFrame()
    vif["variables"] = X.columns
    vif["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]

    return(vif)
```

1. Import Statement:

    This line imports the `variance_inflation_factor` function from the `statsmodels.stats.outliers_influence` module. This function is [1] the core of the VIF calculation

2. Function Definition:

    This defines a function named `calc_vif` that takes one argument, `X`.

3. Initialization of DataFrame:

    `Inside the function, an empty Pandas DataFrame named vif is created. This DataFrame will store the calculated VIF values for each feature.`

4. Assigning Feature Names:

    A new column named "variables" is added to the `vif` DataFrame.

    `X.columns` provides a list of the column names.

5. Calculating VIFs:

- **List Comprehension:** The right-hand side uses a list comprehension to efficiently calculate the VIF for each feature.
- **`range(X.shape[1])`:**
- `X.shape[1]` gives the number of columns (features) in the DataFrame `X`. The `range` function generates a sequence of numbers from 0 to the number of features - 1.

- **`variance_inflation_factor(X.values, i)`:**
- For each `i` (representing the index of a feature), the `variance_inflation_factor` function is called.
    - `X.values` converts the Pandas DataFrame X into a NumPy array, which is required by the `variance_inflation_factor` function.
    - `i` specifies the index of the feature for which the VIF is to be calculated.

6. Return Statement:

`Finally, the function returns the vif DataFrame, which now contains the feature names and their corresponding VIF values.`