```
#importing needed libraries and defining symbolic function 'x' using 'symbols' function
import sympy as sp
x = sp.symbols('x')
print(x)
```

```
x
```

```
#defining the list of functions for which we will generate taylor expansions up to 4th degree
functions = [sp.sin(x), sp.cos(x), sp.exp(x), sp.log(x + 1)]

print("Original Functions:")
for function in functions:
    print(function)
```

```
Original Functions:
sin(x)
cos(x)
exp(x)
log(x + 1)
```

```
#defining the point at which we want to henerate taylor expansions and list to store the tokenized data
point = 0
data = []


#loop over the list of functions and generate their tayler expansion up to the 4th order.
for function in functions:
    taylor_expansion = function.series(x, point, n=5).removeO()
    data.append((str(function), str(taylor_expansion)))

print("\nTaylor Expansions:")
for _, taylor_expansion in data:
    print(taylor_expansion)
```

```
Taylor Expansions:
-x**3/6 + x
x**4/24 - x**2/2 + 1
x**4/24 + x**3/6 + x**2/2 + x + 1
-x**4/4 + x**3/3 - x**2/2 + x
```

```
#tokenizing the data using 'tokenize' function from 'nltk' library
import nltk
tokenizer = nltk.RegexpTokenizer('\w+')
tokenized_data = [(tokenizer.tokenize(function), tokenizer.tokenize(taylor_expansion)) for function, taylor_expansion in data]
print("\nTokenized Data:")
for function, taylor_expansion in tokenized_data:
    print(function, "->", taylor_expansion)
```

```
Tokenized Data:
['sin', 'x'] -> ['x', '3', '6', 'x']
['cos', 'x'] -> ['x', '4', '24', 'x', '2', '2', '1']
['exp', 'x'] -> ['x', '4', '24', 'x', '3', '6', 'x', '2', '2', 'x', '1']
['log', 'x', '1'] -> ['x', '4', '4', 'x', '3', '3', 'x', '2', '2', 'x']
```