Exercise 7.1. *Make a function class.*

Make a class `F` that implements the function

$$f(x; a, w) = e^{-ax} \sin(wx) .$$

A `value(x)` method computes values of f, while a and w are class attributes. Test the class with the following main program:

```
from math import *
f = F(a=1.0, w=0.1)
print f.value(x=pi)
f.a = 2
print f.value(pi)
```

Name of program file: `F.py`. ◇

Exercise 7.2. *Make a very simple class.*

Make a class `Simple` with one attribute `i`, one method `double`, which replaces the value of `i` by `i+i`, and a constructor that initializes the attribute. Try out the following code for testing the class:

```
s1 = Simple(4)
for i in range(4):
    s1.double()
print s1.i

s2 = Simple('Hello')
s2.double();  s2.double()
print s2.i
s2.i = 100
print s2.i
```

Before you run this code, convince yourself what the output of the `print` statements will be. Name of program file: `Simple.py`. ◇

Q3: This is for advanced super programmer who get the above 2 problems done.

Exercise 7.18. *Make a class for summation of series.*

Our task in this exercise is to calculate a sum $S(x) = \sum_{k=M}^{N} f_k(x)$, where $f_k(x)$ is a term in a sequence which is assumed to decrease in absolute value. In class `Sum`, for computing $S(x)$, the constructor requires the following three arguments: $f_k(x)$ as a function `f(k, x)`, M as an `int` object `M`, and N as an `int` object `N`. A `__call__` method computes and returns $S(x)$. The next term in the series, $f_{N+1}(x)$, should be computed and stored as an attribute `first_neglected_term`. Here is an example where we compute $S(x) = \sum_{k=0}^{N}(-x)^k$:

```
def term(k, x):  return (-x)**k

S = Sum(term, M=0, N=100)
x = 0.5
print S(x)
# Print the value of the first neglected term from last S(x) comp.
print S.first_neglected_term
```

Calculate by hand what the output of this test becomes, and use it to verify your implementation of class `Sum`.

Apply class `Sum` to compute the Taylor polynomial approximation for $\sin x$ at $x = \pi, 30\pi$ and $N = 5, 10, 20$. Compute the error and compare with the first neglected term $f_{N+1}(x)$. Present the result in nicely formatted tables. Repeat such calculations for the Taylor polynomial for e^{-x} at $x = 1, 3, 5$ and $N = 5, 10, 20$. Also demonstrate how class `Sum` can be used to calculate the sum (3.1) on page 98 (choose $x = 2, 5, 10$ and $N = 5, 10, 20$). Formulas for the Taylor polynomials can be looked up in Exercise 5.27. Name of program file: `Sum.py`. ◇