Абстрактні класи, інтерфейси, класи-домішки

Завдання:

Абстрактні класи:

1. Реалізувати абстрактний клас "аналізатор послідовностей" (`SequenceAnalyzer`), який обов'язково матиме:

   - абстрактний метод `terms_gen`. Його призначення - повертати генератор, із допомогою якого можна отримати номери та значення усіх ненульових елементів послідовності (починаючи з найпершого і далі).

   - метод друку перших n елементів послідовності (бажано реалізувати його як `__str__`)

   Бажано також, щоб цей клас мав:

   - методи пошук найменшого та найбільшого з-поміж перших n елементів послідовності,

   - методи визначення найменшого номера елемента послідовності, який є меншим за задане ε

2. Реалізувати класи-нащадки `SequenceAnalyzer`, в яких метод `terms_gen` обчислюватиме послідовності, задані такими рекурентними співвідношеннями:

   - $a_0 = 0, a_1 = 1, a_{n+2} = -\frac{a_n}{(n+1)(n+2)}$ $(n \geqslant 1)$ – клас `SinSeq`

   - $a_0 = 1, a_{n+2} = -\frac{a_n}{(n+1)(n+2)}$ $(n \geqslant 0)$ – клас `CosSeq`

3. Реалізувати в класі `SequenceAnalyzer` метод `eval(self, x)`, який трактуватиме елементи послідовності a_n як коефіцієнти у сумі $\sum_{n=0}^{N} a_n \cdot x^n$ (частина доданків ряда Маклорена), та повертатиме цю суму для N, знайденого як найменший номер доданка, який задовольняє умові $|a_n \cdot x^n| < \varepsilon$. За замовчуванням параметр ε обрати 10^{-5}.

   Інтерфейси:

4. Створити класи `ExactSin` та `ExactCos`, які також матимуть метод `eval(self, x)`, який зводитиметься до виклику `math.sin` та `math.cos` відповідно.

5. Створити функцію `print_table(objs, a=0, b=1, npoints=10)`, яка друкуватиме "табличку" значень у форматі:

$x_1 = a$	$f_1(x_1)$	$f_2(x_1)$	...
$x_2 = x_1 + s$	$f_1(x_2)$	$f_2(x_2)$	...
$x_3 = x_2 + s$	$f_1(x_3)$	$f_2(x_3)$	...
...	...	...	...
$x_P = b$	$f_1(x_P)$	$f_2(x_P)$	...

де f_1, f_2, ... відповідають об'єктам, переданим у аргументі `objs`, кожен з яких має метод `eval`. Передати функції `print_table` пари об'єктів, які є екземплярами класів `SinSeq`, `ExactSin` або `CosSeq`, `ExactCos` та таким чином перевірити правильність реалізації рекурентних співвідношень.

6. Реалізувати інтерфейс (як абстрактний клас, в якому є *лише* абстрактні методи) `Evaluatable`, в якому є метод `eval`. Використовуючи множинне наслідування, зробити класи `SinSeq`, `CosSeq`, `ExactSin` та `ExactCos` класами-нащадками класа `Evaluatable`. Внести в код функції `print_table` перевірку, чи всі об'єкти, передані у списку `objs` реалізовують інтерфейс `Evaluatable`.

   Класи-домішки:

7. Виділити метод `eval(self, x)` окремо від класа `SequenceAnalyzer`. Для цього створити новий клас `EvalTaylorSeries` ("ряд Маклорена, який за свої коефіцієнти приймає елементи заданої послідовності"), який:

   - міститиме реалізацію вказаного методу,
   - буде нащадком класу `SequenceAnalyzer`,
   - реалізовуватиме інтерфейс `Evaluatable`.

   Створити класи `MySin`, `MyCos`, унаслідувані від `SinSeq`, `CosSeq` відповідно, а також – від класа `EvalTaylorSeries`, який в цьому випадку відіграватиме роль класа-домішки.

8. Використати функцію `print_table` для перевірки коректності роботи класів `MySin`, `MyCos` як наближених обчислювачів відповідних функцій.