Zaawansowane techniki programowania w języku Python

Lista zadań – Dekoratory

Zadanie 1 – Hello

Napisz własny dekorator o nazwie hello. Wynikiem jego działania powinno być:

- wypisanie na ekranie napisu hello
- wywołanie funkcji przekazanej jako argument.

Przykład kodu źródłowego z zastosowaniem dekoratora:

```
@hello
def moja_funkcja():
    print("moja funkcja")

moja_funkcja()
moja_funkcja()
```

```
hello
moja_funkcja
hello
moja_funkcja
```

Zadanie 2 – Tetranacci

Zaimplementuj własną funkcję o nazwie `tetranacci`, zwracający określony element ciągu Tetranacciego[1]. Funkcja powinna posiadać parametr o nazwie `n` określający numer wyrazu ciągu do obliczenia. Obliczenia wykonuj rekurencyjnie.

Zadanie 3 – Tetranacci Cache

Wykorzystując dekoratory, napisz cache dla funkcji `tetranacci` z poprzedniego zadania. Ten dekorator powinien zapobiegać przed ponownym obliczaniem tych samych wartości.

1 https://pl.wikipedia.org/wiki/Ci%C4%85g_Fibonacciego#Ci.C4.85g_.E2.80.9ETetranacciego.E2.80.9D

Notatki:

Zadanie 4 – Login required

Napisz program, który do uruchomienia będzie wymagał nazwy użytkownika i hasła lub wybrania opcji logowanie anonimowe. Napisz dekorator o nazwie `login_required`, który zaaplikowany do dowolnej funkcji zweryfikuje czy użytkownik zalogował się poprawnie. W przypadku, gdy:

- użytkownik zalogował się poprawnie, dekorator powinien wywoływać otaczaną funkcję,
- użytkownik nie został zalogowany, dekorator powinien rzucić wyjątek o nazwie `NotLoggedIn`.

Przykład działania:

```
Zaloguj się
uzytkownik: jan
haslo: kowalski
Menu
1. Wypisz hello
2. Zakoncz
Wybor: 1
Hello
```

Przykład działania:

```
Zaloguj się
uzytkownik: anonim
Menu
1. Wypisz hello
2. Zakoncz
Wybor: 1
Musisz sie zalogowac
```

Notatki:

.....

.....