

Laboratorium 6 – Programowanie obiektowe w Python.

Cele dydaktyczne

1. Zapoznanie z paradygmatem obiektowym w Python.
2. Zapoznanie z kaczym typowaniem w Python.

Program można zgłosić jako zrobiony, jeśli spełnione są poniższe warunki:

1. Program jest zgodny z podaną specyfikacją.
2. Został przetestowany.
3. Student go rozumie i potrafi wyjaśnić.

UWAGA:

Wykonywanie zadań przy użyciu czatu GPT będzie traktowane jako praca niesamodzielna i będzie skutkować oceną niedostateczną.

- **należy zabezpieczyć funkcje przed błędnymi danymi wejściowymi i wyjątkami. (np. dzieleniem przez zero).**
- **do zadań przygotować kilka testów sprawdzających poprawność działania**
- **do prowadzącego wystać pliki z kodem źródłowym oraz rzuty ekranu z przygotowany testami**

Wprowadzenie

Celem niniejszego laboratorium jest przećwiczenie elementów programowania obiektowego w języku Python. Język ten wspiera (choć nie wymusza) paradygmat klasowo-obiektowy m.in. zakresie związków klasa-instancja, generalizacja-specjalizacja (w celu realizacji mechanizmu dziedziczenia), w tym daje możliwość wielodziedziczenia. Polimorfizm możliwy jest przez tzw. [kacze typowanie](#). Zadania z niniejszej listy można i warto rozwiązać, korzystając z własnej implementacji struktur danych oraz funkcji do reprezentacji i przetwarzania logów SSH z poprzedniej listy. Opracowane na tej liście programy powinny być również kompatybilne z formatem [przykładowych logów SSH z poprzedniej listy](#).

Zadania

1. Skonstruuj klasę `SSHLogEntry` reprezentującą pojedynczy wpis dziennika SSH. Niech klasa pozwala na reprezentację co najmniej informacji o czasie, opcjonalnej nazwie hosta, surowej treści wpisu, numerze PID. W ramach klasy `SSHLogEntry` zaimplementuj:
 - a. konstruktor
 - b. metodę przekształcającą obiekt w ciąg znaków,
 - c. metodę zwracającą [obiekt klasy IPv4Address](#), jeśli we wpisie wystąpił adres, w przeciwnym wypadku `None`

Punkty: 1

2. Skonstruuj klasy dziedziczące po `SSHLogEntry`. Klasy powinny reprezentować:
 - a. Odrzucenie hasła¹
 - b. Akceptację hasła²
 - c. Błąd³
 - d. Inną informację.

Przeanalizuj wpisy reprezentujące poszczególne rodzaje informacji i wydobądź z nich atrybuty specyfikujące konkretne klasy. Pamiętaj o wywołaniu funkcji `__init__` klasy bazowej.

Punkty: 1

3. Korzystając z modułu [abc](#), zdefiniuj **`SSHLogEntry`** jako klasę abstrakcyjną. Utwórz w niej metodę abstrakcyjną **`validate()`**. Zaimplementuj tę metodę w każdej z klas dziedziczących w taki sposób, aby weryfikowała, czy zawartość surowej treści wpisu jest zgodna z pozostałymi, wyekstrahowanymi atrybutami. Funkcja w klasie reprezentującej inną informację powinna zawsze zwracać prawdę.
4. W klasie `SSHLogEntry`, określ atrybut reprezentujący surową treść wpisu jako część niepublicznego API.
5. W klasie `SSHLogEntry`, zdefiniuj właściwość ([property](#)) tylko do odczytu o nazwie `has_ip`, która będzie miała wartość `True`, gdy we wpisie występuje adres IP, w przeciwnym wypadku `False`.
6. W klasie `SSHLogEntry` zaproponuj implementację metod magicznych `__repr__`, `__eq__`, `__lt__`, `__gt__`.

Punkty: 4

Utwórz klasę `SSHLogJournal`, która będzie służyła do agregowania `SSHLogEntry` w wewnętrznej liście. Zdefiniuj w niej magiczne metody, takie jak `__len__`, `__iter__`, `__contains__` tak, aby można było po niej iterować jak po

¹ Przykład wiersza reprezentującego odrzucone hasło:

Dec 10 06:55:48 LabSZ sshd[24200]: Failed password for invalid user webmaster from 173.234.31.186 port 38926 ssh2

² Przykład wiersza reprezentującego zaakceptowane hasło:

Dec 10 09:32:20 LabSZ sshd[24680]: Accepted password for fztu from 119.137.62.142 port 49116 ssh2

³ Przykład wiersza reprezentującego błąd:

Dec 10 11:03:44 LabSZ sshd[25455]: error: Received disconnect from 103.99.0.122: 14: No more user authentication methods available. [preauth]

sekwencji. Zdefiniuj metodę **append()**, która przyjmie na wejściu ciąg znaków, utworzy z niego odpowiedni obiekt SSHLogEntry, dokona jego walidacji, i doda do wewnętrznej listy. Zdefiniuj metodę, pozwalającą pobrać fragment listę logów wg. wybranego samodzielnie kryterium (np. w wybranym przedziale dat, dla danego IP, etc.).

Punkty: 1

Dla ambitnych (punkty 1):

- a. Wykorzystaj wzorzec projektowy [metody fabrycznej](#).
- b. Zdefiniuj metodę **__getattr__** w taki sposób, by z jej użyciem pozyskiwać wpisy po IP, indeksie i dacie. Czy wymaga to zmiany wewnętrznej struktury klasy? Zadbaj, by klasa działała z obiektami typu [slice](#).

7. **Kacze typowanie:** Utwórz klasę SSHUser reprezentującą użytkownika. Niech klasa pozwala na przechowywanie informacji o nazwie użytkownika i dacie ostatniego logowania. Zdefiniuj w niej metodę **validate()**⁴, która będzie walczyć poprawność nazwy użytkownika. Następnie pobierz kilka instancji SSHLogEntry z SSHLogJournal. Utwórz kilka instancji klasy SSHUser. Przechowaj wszystkie instancje na wspólnej liście. Zademonstruj działanie kaczego typowania poprzez iterację po tej liście i wywoływanie metody **validate()**.

Punkty: 2

⁴ np. przy pomocy wyrażenia regularnego `r'^[a-z_][a-z0-9_-]{0,31}$'`