|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test końcowy | | | *PS-06-01-Z1/03*  Ważne od: 2021.07.02 | |
| ***Python dla administratorów systemów LUW***  ***SK-PROPYT-AKAZAA***  ***Data: 24.03.2025 – 28.03.2025***  ***Wykładowca: Marcin Albiniak***  ***TES\_241002\_2551500PYT\_140408AK\_*** | | |  | |
| ***Dane uczestnika testu*** | ***Wyniki – punktacja*** | |
| ***Imię:*** | ***Maksymalna: 14*** | |
| ***Nazwisko:*** | ***Uzyskana:14*** | |
| ***Data wypełnienia testu:*** Kliknij lub naciśnij, aby wprowadzić datę. |  | |
| ***Miejscowość:*** |  | |

1. **Cechy języka Python to**

Wsparcie paradygmatów programowania obiektowego

Implementacja składni C++

Typowanie dynamiczne

Wieloplatformowość

***Liczba punktów: \_\_\_\_\_***

1. **Środowiska deweloperskie dla języka Python to:**

IntelliJ

Visual Studio 2019

PyCharm

Adobe Dreamweaver

***Liczba punktów: \_\_\_\_\_***

1. **Biblioteki służące do analizy danych to**

Matlab

numpy

pandas

gallileo

***Liczba punktów: \_\_\_\_\_***

1. **Metaklasa:**

dziedziczy po klasie type

jest odmianą klasy abstrakcyjnej

jest implementowana w klasach pochodnych za pomocą argumentu metaclass

definiuje przyszłe klasy i ich zachowanie

***Liczba punktów: \_\_\_***

1. **Metoda listy, która konwertuje sekwencję s na listę to:**

s.count(x)

List(s)

s.append(x)

s.reverse()

***Liczba punktów: \_\_\_\_\_***

1. **Jedna z definicji funkcji typu getter jest realizowana poprzez:**

użycie dekoratora @property

użycie generatora

implementacja metody set()

import settter

***Liczba punktów: \_\_\_\_\_***

1. **Polecenie import dotyczy:**

platform

modułów

rejestrów

bibliotek

***Liczba punktów: \_\_\_\_\_***

1. **Podstawowa definicja klasy to:**

class nazwa:

class: nazwa

\_class() nazwa:

Nazwa():

***Liczba punktów: \_\_\_\_\_***

1. **Moduły dają możliwości:**

Wykonywania operacji na ciągach znaków

Obsługi wyrażeń regularnych

Dostępu do stron www przez URL

Przeprowadzenie ataku sieciowego na serwer www

***Liczba punktów: \_\_\_\_\_***

1. **Prawidłowy zapis dziedziczenia klas to:**

class Szescian(Kwadrat):

class Szescian:Kwadrat

class Szescian extends Kwadrat:

class Szescian import Kwadrat

***Liczba punktów: \_\_\_\_\_***

1. **Obsługa wyjątków realizowana jest instrukcją:**

try-catch

try−except−else

if-else

exception

***Liczba punktów: \_\_\_\_***

1. **Proces optymalizacji może być realizowany w Pythonie poprzez:**

zwiększenie liczby wcięć w kodzie

stosowanie bibliotek zewnętrznych napisanych w języku C np. .numpy

stosowanie przetwarzania asynchronicznego

]pisanie nazw klas dużymi literami

***Liczba punktów: \_\_\_\_\_***

1. **biblioteka pozwalająca na zarządzanie wątkami asynchronicznymi:**

async

AsyncTask

concurrent.features

worker

***Liczba punktów: \_\_\_\_\_***

1. **Klasę abstrakcyjną tworzymy przez:**

abstract class Nazwa

class Nazwa: abstract

class nazwa(ABC)

class Nazwa:ABC

***Liczba punktów: \_\_\_\_\_***