Laboratorium 1 – Wprowadzenie do Python

Cele dydaktyczne

1. Uruchamianie interpretera Python
2. Uruchamianie prostych aplikacji
3. Instalacja dodatkowych modułów Python
4. Zarządzanie zależnościami

Wprowadzenie

Python jest językiem skryptowym, którego programy wykonywane są przez interpreter. W uproszczeniu, podstawowym sposobem uruchamiania aplikacji napisanych w tym języku jest przekazanie ścieżki do skryptu jako argument interpreterowi, czyli programowi odpowiedzialnemu za wykonywanie poleceń "linia po linii". W dowolnym momencie można dokonać edycji skryptu i uruchomić program ponownie bez konieczności przebudowania (ponownej kompilacji, linkowania, itd.). Z tego powodu, języki skryptowe są dobrymi narzędziami do takich zagadnień jak prototypowanie, automatyzacja zadań, przetwarzanie danych, etc.

Zadania

W salach laboratoryjnych 127b i 127c dostępna jest maszyna wirtualna VirtualBox, na której zainstalowany jest Linux Ubuntu[1], w którym znajdują się narzędzia niezbędne do wykonania zadań. Niezależnie, poniższe zadania są mogą zostać wykonane w ramach dowolnego systemu operacyjnego dla komputerów klasy PC.

1. Uruchamianie interpretera w trybie REPL

Tryb REPL (Read-Eval-Print Loop) pozwala na uruchomienie interpretera w trybie powłoki. Po uruchomieniu, użytkownik wprowadza wyrażenia do ewaluacji (po symbolu zachęty `>>>`), które następnie są ewaluowane przez interpreter, a wynik ewaluacji jest wyświetlany.

a. Aby poznać wersję interpretera Python dostępną w aktualnej ścieżce, uruchom konsolę/wiersz poleceń, a następnie wpisz: `python --version`.

b. Uruchom Interpreter Pythona w trybie REPL i skonstruuj program, wypisujący napis "`Hello world!`".

[1] użytkownik/hasło: `student/student`, dostęp do konta root'a przez sudo.

```
python

>>> print("Hello world!")
```

c. Wykorzystaj zmienną, aby zmodyfikować komunikat.

```
>>> name = "Python"

>>> print("Hello, {0}!".format(name))

>>> print(f"Hello, {name}!")
```

d. Wykonaj kilka działań arytmetycznych, poprzez skonstruowanie wyrażeń i zmiennych

- ✓ dodawanie (`+`), odejmowanie (`-`), mnożenie (`*`),
- ✓ dzielenie(`/`), dzielenie bez reszty(`//`), reszta z dzielenia (`%`),
- ✓ potęgowanie (`**`).
- ✓ wyświetl zawartości zmiennych (operator print)

```
# Przykład 1
5 + 3.2
10 - 4
2.5 * 6

# Przykład 2
wynik_dod_2 = -7 + 2.8
wynik_od_2 = 15.5 - 3
wynik_mn_2 = 4 * 9.2
```

```
# Przykład 1
15 / 3
17 // 4
20 % 7

# Przykład 2
wynik_dz_2 = 8.5 / 2
wynik_dz_bez_reszty_2 = 23 // 6
reszta_z_dzielenia_2 = 14 % 3
```

```
2 ** 3
5.5 ** 2

# Przykład 2
wynik_poteg_3 = (-3) ** 4
wynik_poteg_4 = 4.1 ** 1.5
```

e. Następnie, skonstruuj kilka wyrażenie arytmetycznych wykorzystujących symbol `_` jako jeden z operandów, np.

```
>>> _ * 2.0

>>> _ + 1
```

Zastanów się, jaką wartość reprezentuje ten symbol.

f. Zakończ sesję interaktywną wywołaniem wyrażeniem `exit()`.

Punkty: 2

2. Uruchamianie notesów Jupyter.

Jupyter Notebook – graficzne narzędzie uruchamiane w przeglądarce internetowej, które pozwala na mieszanie tekstów ze skryptami w języku Python, pozwalające na tworzenie dokumentów z obliczalnymi elementami (np. wizualizacjami).

a. Zainstaluj Jupyter Notebook wpisując w terminalu/wierszu poleceń.

```
pip install notebook
```

b. Uruchom Jupyter notebook.

```
jupyter notebook
```

c. Kliknij przycisk listy rozwijanej 

z prawej strony i wybierz opcję “Python 3 (ipykernel)”.

d. Wprowadź poniższy kod do pola tekstowego obok napisu `In [ ]:`.

```
name = "World"

print(f"Hello, {name}")
```

Wciśnij przycisk “Run”, w celu uruchomienia kodu.

e. W kolejnym polu tekstowym, wpisz

```
# Moja aplikacja w języku Python
  Oto mój pierwszy notes Jupyter.
```

Korzystając z listy rozwijanej umieszczonej pod menu, zmień typ bloku z Code na Markdown. Wciśnij przycisk “Run”.

f. Wciśnij przycisk zapisu  . Przeanalizuj zawartość pliku *.ipynb w edytorze tekstowym.

Punkty: 2

3. Uruchamianie skryptów w konsoli.

a. Otwórz wybrane IDE (np. VSCode).

b. Utwórz plik app.py z następującą treścią:

```
name = input("Enter your name: ")

print(f'Hello, {name}!')
```

c. Uruchom plik z poziomu terminala(wiersza poleceń) przy pomocy komendy: `python app.py`

d. .Uruchom plik w VSCode.

e. Utwórz plik app_par.py z następującą treścią:

```
import sys
name = sys.argv[1]
print("Hello", name)
```

f. Uruchom plik z poziomu terminala/wiersza poleceń przy pomocy komendy: `python app.py Imię_Studenta`

Punkty: 2

4. **Prosty program w Python.**

Napisz program w języku Python, który obliczy pole trójkąta o wysokości i długości podstawy zadanych przez użytkownika. Aby dokonać konwersji pomiędzy napisem a liczbą, należy wykorzystać funkcję `int()`.

Punkty: 2

5. **Korzystanie z menedżera zależności**

Python posiada bogate repozytorium modułów i narzędzi: The Python Package Index (PyPI).

Moduły można instalować z terminala/wiersza poleceń przy użyciu polecenia pip, tak jak robiono to w poprzednich zadaniach. Po zainstalowaniu modułu, w skrypcie należy go zaimportować przy użyciu instrukcji `import`. Dla przykładu, dla modułu wikipedia, pozwalającego na pobieranie treści stron z Wikipedii:

```
pip install wikipedia

…

>>> import wikipedia

>>> wikipedia.page("Script language").summary
```

W przypadku pracy nad większymi projektami. aby ułatwić zarządzanie modułami oraz ich wersjami, warto wykorzystać do tego odpowiednie narzędzie. Do wyboru są:

- Venv + pip https://docs.python.org/3/library/venv.html
- Poetry https://python-poetry.org
- Conda https://conda.io/en/latest/
- Pipenv https://pipenv.pypa.io/en/latest/

Zapoznaj się z dokumentacją wybranego narzędzia. Następnie, z jego wykorzystaniem utwórz program, który będzie zależny od modułu wikipedia. Niech program przyjmuje od użytkownika nazwę artykułu na wikipedii, a na wyjściu zwraca jego podsumowanie oraz URL do strony na angielskiej wikipedii.

Punkty: 2