Laboratorium 01

Temat: Wprowadzenie do metod numerycznych i obliczeń inżynierskich z wykorzystaniem języka programowania Python

Celem ćwiczenia jest zapoznanie się z:

- językiem programowania Python
- zintegrowanym środowiskiem programistycznym: **DrPython**, Geany, **PyCharm**, itp.

Język programowania Python został stworzony w latach 90 przez holendra Guido van Rossumema. Język ten był następcę języka ABC. Założeniem twórcy było opracowanie języka interpretowanego, interaktywnego i skryptowego, jak również posiadającego dynamiczny system typów danych i automatyczne zarządzanie pamięcią, przez co podobny do takich języków, jak: Tcl czy Perl. Aktualnie Python jest rozwijany na zasadzie projektu Open Source i jest zarządzany przez Python Software Foundation, będącą organizacją non-profit. Python doczekał się aktualnie kilku implementacji, np. standardowa to tzw. CPython (pisany w C) oraz inne jak: Jython (pisany w Javie), IronPython (opracowany na platformę .NET). Jedną z najważniejszych cech użytecznych języka Python jest możliwość tworzenia aplikacji z wykorzystaniem podstawowych paradygmatów programowania, tzn.:

- programowanie strukturalne,
- programowanie funkcyjne,
- programowanie obiektowe.

Ponadto ważną cechą Pythona jest dynamiczne sprawdzanie typów danych oraz zarządzania pamięcią z wykorzystaniem tzw. „garbage collection“, np. znanego z języka Java. Aktualnie korzysta się z dwóch interpreterów języka Python tj. wersji 2.x (najczęściej 2.7) i wersji 3.x (np. 3.5). Obie wersje są stosowane równolegle, przy czym wersja 2.7 jest uznawana za stabilną i dopracowaną, natomiast wersja 3.x zawiera wiele modyfikacji i usprawnień.

Python został zaprojektowany tak, aby zapewnić możliwie dużą czytelność kodu źródłowego. Posiada prosty układ graficzny, używa angielskich słów tam, gdzie inne języki korzystają ze znaków interpunkcyjnych i posiada zdecydowanie mniej konstrukcji składniowych niż wiele języków strukturalnych, takich jak C czy Perl. Cechą wyróżniającą Pythona spośród innych języków jest stosowanie wcięć do wydzielania bloków kodu (cecha odziedziczona z języka ABC). Jest to cecha unikatowa wśród powszechnie stosowanych języków programowania, jako pierwsza rzucająca się w oczy programistom niepiszącym w Pythonie.

Tab.1. Przykład funkcji silnia

C	Python

int silnia(int x) {
if (x == 0) return 1;
else return x * silnia(x-1);
}
	def silnia(x):
	  if x == 0:
	  return 1
	  else:
	  return x * silnia(x-1)

W celu uruchomienia tzw. maszyny wirtualnej (interpretatora) języka Python w systemie Linux, należy uruchomić konsolę oraz wpisać słowo „python“ dla wersji 2.7 lub "python3" dla wersji 3.x.

```
artur@HP:~$ python
Python 2.7.6 (default, Mar 22 2014, 22:59:56)
[GCC 4.8.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

```
artur@pirx:~$ python3
Python 3.5.3 (default, Sep 14 2017, 22:58:41)
[GCC 6.3.0 20170406] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

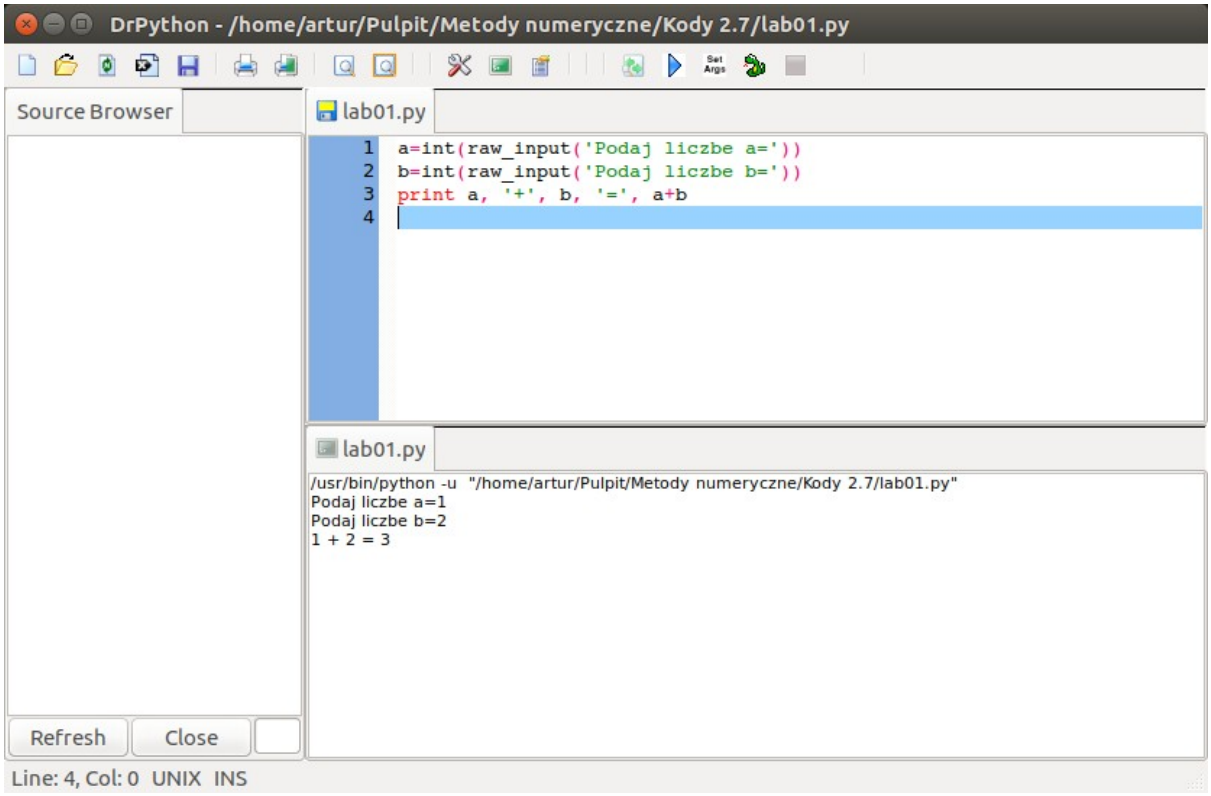
Interpreter języka Python pozwala na pracę interaktywną, jak to pokazano na rysunku poniżej.

```
Python 2.7.6 (default, Mar 22 2014, 22:59:56)
[GCC 4.8.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> a=2
>>> b=2
>>> a+b
4
>>> a="Lab."
>>> b=" z metod numerycznych"
>>> a+b
'Lab. z metod numerycznych'
>>>
```

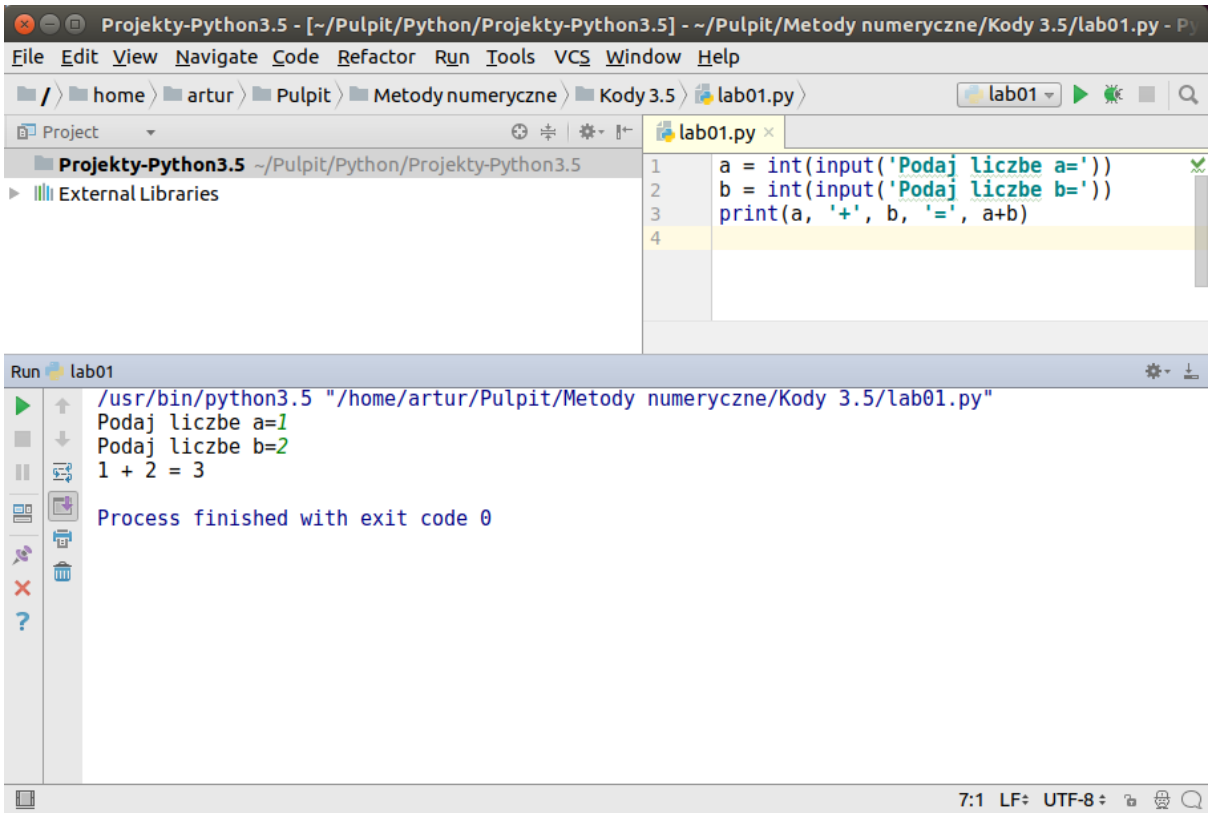
Inny rodzaj pracy to uruchamianie wcześniej napisanych skryptów w postaci programu. Standardowe rozszerzenie programów skryptowych w Pythonie to *.py. Skrypt można uruchomić korzystając ze zintegrowanego środowiska programistycznego IDE (Integrated Development Environment). Język Python doczekał się wielu implementacji zintegrowanych środowisk programistycznych IDE, np.:

- Idle - wbudowany
- Geany – niewielki
- **DrPython** – przydatny w przypadku małych projektów
- **PyCharm** – przydatny w przypadku dużych projektów
- itp.

DrPython jest edytorem prostym i wysoce konfigurowalnym, który został opracowany z myślą o tworzenie krótkich programów w języku Python. Jest napisany w Pythonie z wykorzystaniem biblioteki wxPython. W przypadku ćwiczeń laboratoryjnych proponuje się korzystanie ze środowiska DrPython dla projektów wykonanych dla interpretera Pythona w wersji 2.7, który jest interpreterm domyślnym w systemie operacyjnym Linux. DrPython pozwala zarówno pisać, sprawdzać składnię, jak i uruchamiać programy skryptowe, ale także pracować w sposób interaktywny, korzystając z konsoli Pythona, ikona: 



PyCharm jest środowiskiem rozbudowanym i stosowanym powszechnie do tworzenia dużych projektów. Posiada wiele zalet jak rozbudowany system pomocy, praca w trybie debugowania i możliwość korzystania z różnych interpreterów Pythona, np. wersji 2.7, 3.5, itp. poprzez zmianę ustawień konfiguracyjnych. PyCharm posiada także rozbudowaną konsolę Pythona.



Jedną z istotnych zalet języka Python jest bogactwo dostępnych bibliotek oraz otwarty kod. Należy jednak pamiętać, że wybrane biblioteki są kompilowane pod określoną wersję interpretera Pythona, tj. np. 2.7 lub 3.5. Typowe biblioteki Pythona wykorzystywane w zastosowaniach naukowych i inżynierskich to:

- numpy - obliczenia numeryczne
- scipy - obliczenia naukowe
- matplotlib – wykresy (Uwaga: wymaga zainstalowanej biblioteki graficznej tk)
- tk - grafika
- wxPython - grafika z wykorzystaniem biblioteki wx
- PyQt - grafika z wykorzystaniem biblioteki Qt
- PyGame - tworzenie gier

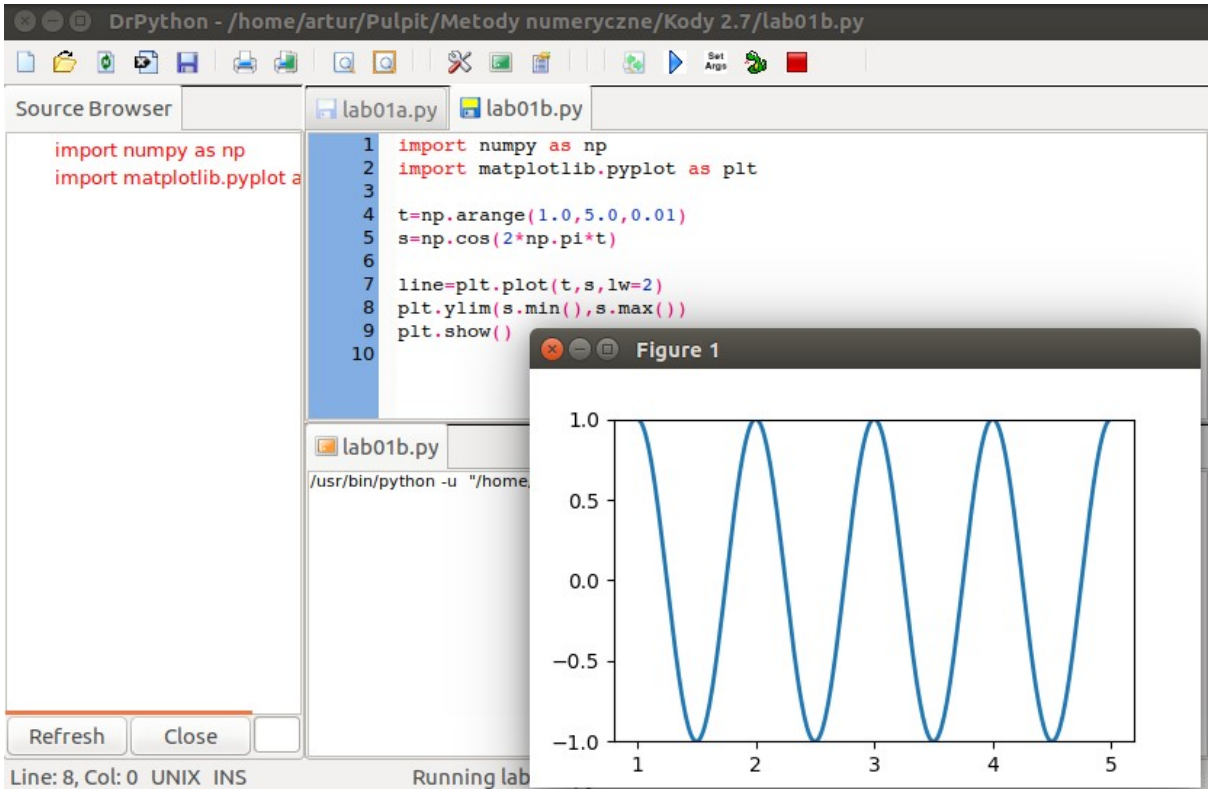
Biblioteki można instalować korzystając z programu / skryptu pip (dla wersji 2.7) lub pip3 (dla wersja 3.x). np.:

- pip install numpy
- pip3 install numpy.

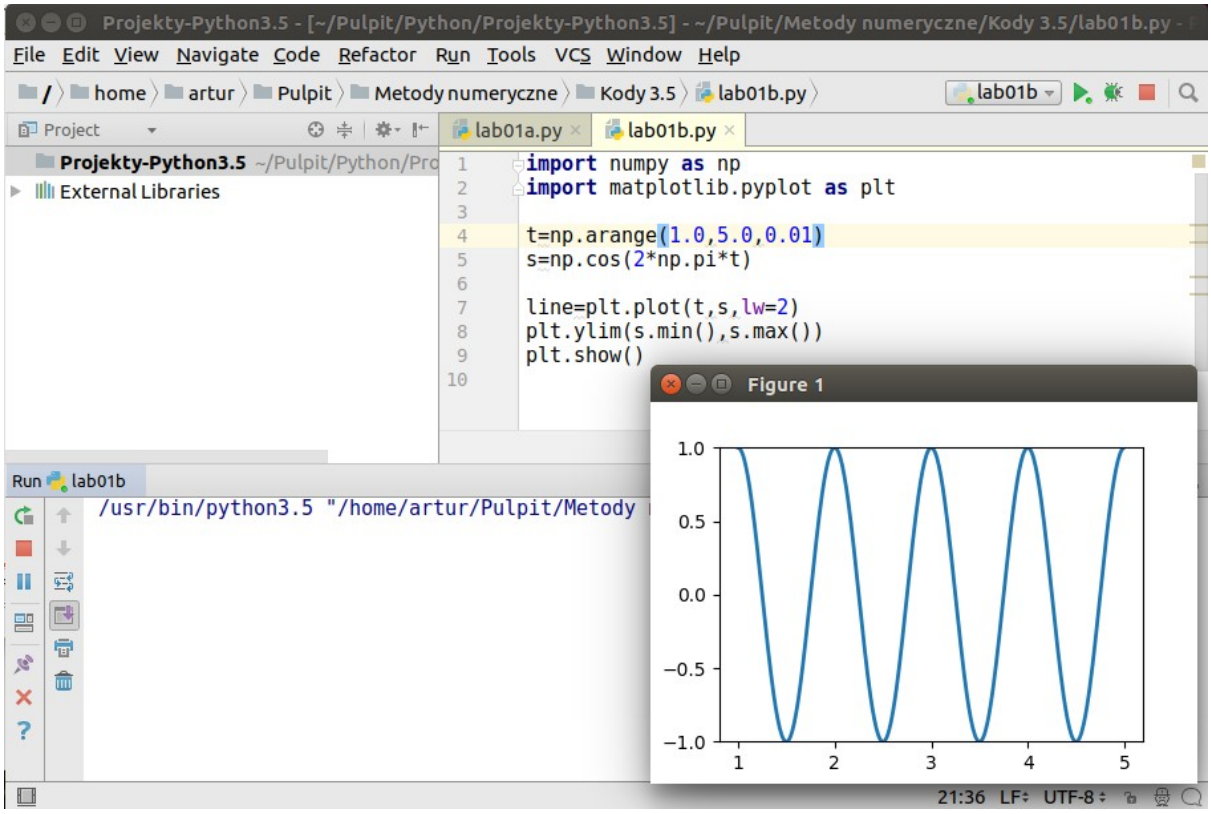
W celu skorzystania z wybranej biblioteki wewnątrz interpretera Pythona korzysta się z instrukcji „import“, np.:

- import numpy
- import numpy as np
- from numpy import *

Poniżej przedstawiono kod programu oraz wynik w przypadku środowiska DrPython i interpretera Pythona w wersji 2.7.



Poniżej przedstawiono kod programu oraz wynik w przypadku środowiska PyCharm i interpretera Pythona w wersji 3.5.



Opis programu (Python w wersji 2.7 i 3.5):

```
import numpy as np
import matplotlib.pyplot as plt

t=np.arange(1.0,5.0,0.01)
s=np.cos(2*np.pi*t)

line=plt.plot(t,s,lw=2)
plt.ylim(s.min(),s.max())
plt.show()
```

Import bibliotek: numpy i matplolib

Definicja wartości punktów dla osi X i Y

Definicja danych oraz parametrów wykresu

<u>Ćwiczenia:</u>

1. Zapoznaj się z różnymi trybami pracy z Pythonem w postaci:
   - interaktywnej
   - skryptowej - programowej
2. Zapoznaj się z różnymi paradygmatami programowania w języku Python, np. na przykładzie silni, tzn.:
   - funkcyjnym
   - strukturalnym
3. Wykonaj wykres wybranej funkcji, np. kwadratowej, trygonometrycznej, itp. w wybranym przez użytkownika zakresie z wykorzystaniem biblioteki numpy i matplotlib.