Wydział: **Zarządzania i Modelowania Komputerowego**
Katedra Technologii Informatycznych
Przedmiot: **Technologie informacyjne**
Rok **I**

PYTHON - ĆWICZENIE 1

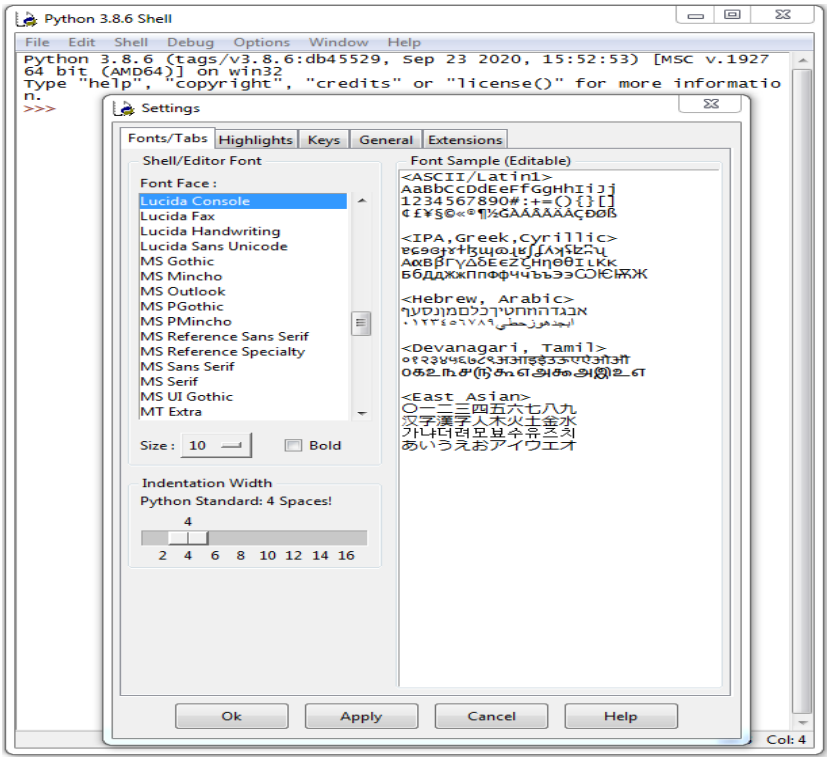
Wprowadzanie i uruchamianie programów w środowisku programistycznym PYTHON.
Proste algorytmy obliczeniowe.

Instrukcja dotyczy zasad uruchamiania prostych programów w języku Python.
Darmowa wersja jest dostępna na stronie: https://www.python.org/ Dla posiadaczy systemu Windows 10 dostępna jest wersja Python 3.9.0 , jeśli ktoś dysponuje tylko Windows 7, może wybrać wersję Python 3.8.6 – ścieżka do instalacji poniżej.
https://www.python.org/ftp/python/3.8.6/python-3.8.6-amd64.exe

Język PYTHON jest językiem skryptowym. Interpretacja i wykonywanie instrukcji odbywa się bez tworzenia programu wykonywalnego (z rozszerzeniem ***.exe***). Instrukcje wchodzące w skład programu zapisywane są w pliku tekstowym (skrypcie) z rozszerzeniem ***.py***
Możliwa jest również praca w trybie linii poleceń.

1. Uruchomienie IDLE(Python GUI)

Aby rozpocząć pracę w Pythonie należy wybrać aplikację ***IDLE*** Pythona. Otworzy się wówczas okno interaktywnej powłoki. Widoczny, po uruchomieniu trybu interaktywnego Pythona, znak zachęty **>>>** oznacza, że interpreter jest gotowy do wykonywania poleceń.
Możliwe jest dostosowanie środowiska programistycznego do własnych potrzeb po uruchomieniu funkcji Options/Configure IDLE (rys.1).



*Rys.1. Okno **IDLE** po uruchomieniu i wybraniu funkcji **Options/Configure IDLE***

W zakładce ***Fonts/Tabs*** można wybrać rodzaj używanej czcionki i jej wielkość, jak również zmienić rozmiar tabulatora (jak się okaże, wcięcie w tekstach pisanych programów ma istotne

znaczenie w tym języku; z tego też powodu warto wybrać font o stałej szerokości znaków). Zmiana pozostałych opcji nie jest zalecana.

2. Praca w trybie linii poleceń (tryb interaktywny)

Tryb linii poleceń można traktować jako rozbudowany kalkulator. Każde wyrażenie należy zakończyć przyciskiem ***Enter***. Przykładowo:

```
>>> 4-1
3
>>> 2+3
5
>>> 4*-5
-20
>>> 3**4
81
>>> 4/3
1.3333333333333333
>>> 4//3
1
>>> 7%3
1
>>>
```

3. Instrukcja przypisania

Możliwe jest przypisywanie wartości zmiennym. Instrukcja przypisania ma postać:

zmienna = wyrażenie

W jej wyniku zmienna występująca po lewej stronie znaku = przyjmuje wartość wyrażenia po prawej stronie. Warto przy tym pamiętać, że odczytanie wartości zmiennej (wykorzystanie jej w wyrażeniu po prawej stronie instrukcji przypisania) nie zmienia wartości tej zmiennej, natomiast nadanie zmiennej nowej wartości niszczy jej poprzednią wartość.

Nazwa zmiennej zaczyna się od litery lub podkreślnika, może zawierać litery, podkreślniki i cyfry.

Przykład:

```
>>> a=2
>>> a
2
>>> b=a+3
>>> b
5
>>> a=a-3
>>> a
-1
>>>
```

Ostatnia z przykładowych instrukcji modyfikuje warość zmiennej. Dla takich instrukcji możliwy jest skrócony zapis:

a+=4 jest równoważne zapisowi: a=a+4
a-=2 jest równoważne zapisowi: a=a-2
a*=5 jest równoważne zapisowi: a=a*5
a/=7 jest równoważne zapisowi: a=a/7
a//=3 jest równoważne zapisowi: a=a//3

4. Wyprowadzanie wyników

Do wyprowadzania wyników obliczeń służy instrukcja:

print (lista_wyjścia)

gdzie **lista_wyjścia** składa się z rozdzielonych przecinkami wyrażeń, których wartość ma być zaprezentowana. Na takiej liście mogą się więc znaleźć stałe (liczbowe, logiczne i łańcuchy znaków), zmienne oraz wyrażenia arytmetyczne i logiczne.

Przykład:

```
>>> a=2
>>> x=3.4
>>> print(a+x)
5.4
>>> print('a=',a,7.45,True,a<-5.6,3*x-a)
a= 2 7.45 True False 8.2
>>>
```

5. Wykorzystanie modułu matematycznego.

Większość funkcji matematycznych np. sin, exp, sqrt, log i inne nie są standardowo dostępne. Po wprowadzeniu instrukcji w trybie linii poleceń:

print (log(4))

nastąpi sygnalizacja błędu:

```
>>> print(log(4))
Traceback (most recent call last):
  File "<pyshell#67>", line 1, in <module>
    print(log(4))
NameError: name 'log' is not defined
>>>
```

wskazująca, że nie jest zdefiniowany identyfikator **log.** W celu zapewnienia dostępu do podstawowych funkcji matematycznych należy odwołać się do modułu **math** poleceniem:

import math

Od tego momentu można korzystać z biblioteki funkcji matematycznych. Należy jednak pamiętać, że nazwa funkcji musi być poprzedzona prefiksem **math.** Wydrukowanie wartości logarytmu dziesiętnego z liczby 4 realizuje zatem instrukcja:

```
>>> import math
>>> print(math.log(4))
1.3862943611198906
>>>
```

Pełny wykaz funkcji matematycznych z modułu **math** można uzyskać wprowadzając polecenie:

help (math)

Zadania do wykonania

Wprowadź polecenia (w trybie interaktywnym) przypisujące wartości zmiennym:

a=2.4
b=5.5
x=3
y=2

a następnie wydrukuj wartości wyrażeń (w nawiasie poprawny wynik):

a) $\frac{a+1}{b-1}$ (0.7555555555555555)

b) $\sqrt{x^2+y^2}$ (3.60555127546)

c) $|x-10|*y$ (14.0)

d) $e^x \sin 5x$ (13.06138042417573)

e) $\frac{\ln|x-5|}{\sqrt[3]{y}}$ (0.5501512817948243)

f) $sin\frac{\pi}{3} * cos\sqrt[3]{x}$ (0.11101841654054002)

g) $\sin 60^o$ (0.8660254037844386)

6. Wprowadzanie danych z klawiatury

Do wprowadzania danych z klawiatury służy instrukcja:

input ('prompt')

gdzie **prompt** jest łańcuchem zachęty (może nie występować):

Wprowadź w trybie interaktywnym polecenia:

```
>>> x=input('x=')
x=12
>>> y=input('y=')
y=4
>>> print(x+y)
124
>>>
```

Dlaczego wynik nie jest równy sumie 12+4?

Instrukcja **input** przypisuje podanej zmiennej **łańcuch tekstu** wprowadzony z klawiatury. Jeżeli zmienne x i y mają mieć wartości liczbowe, konieczna jest konwersja wprowadzonego łańcucha znaków na liczbę. W poniższym przykładzie dokonano konwersji na liczby całkowite.

```
>>> x=int(input('x='))
x=12
>>> y=int(input('y='))
y=4
>>> print(x+y)
16
>>>
```

7. Wprowadzanie tekstu nowego programu na przykładzie algorytmu wyznaczającego pole powierzchni i objętość kuli o promieniu r.

Utwórz nowe okienko ***File/NewWindow*** a następnie zapisz pustą aplikację do pliku **kula.py** (funkcja ***File/Save As***) w swoim katalogu stanowiskowym.

Uwaga: Przy wprowadzaniu nazwy pliku należy obowiązkowo dopisać rozszerzenie ***.py*** Ułatwi to otwieranie już utworzonych i zapisanych skryptów (funkcja ***File/Open***).

Wprowadź tekst skryptu (programu) aby uzyskać zapis przedstawiony poniżej. Zwróć uwagę na przyłączenie z modułu **math** tylko stałej **pi.** Analogicznie można przyłączyć wybrane funkcje tego modułu. Korzystanie z tak przyłączonych elementów modułu nie wymaga już użycia prefiksu **math.**

File Edit Format Run Options Window Help

```
print ('obliczanie pola i objętości kuli o promieniu r')
from math import pi
r=float(input('promień r='))
pole=4*pi*r**2
print('pole kuli = ', pole)
obj=pole*R/3
print('objętość= ', obj)
```

8. Uruchamianie i testowanie skryptu kula.py.

Uruchamianie aplikacji realizujemy wybierając funkcję ***Run|Run Module*** lub **F5** . Wprowadź wartość promienia, np. 1.

Jeżeli popełniłeś błąd przy wprowadzaniu tekstu programu, zostanie wyświetlony komunikat o wystąpieniu błędu składniowego i nastąpi przejście do tekstu programu, w którym kursor wskaże prawdopodobne miejsce wystąpienia błędu. Należy dokonać poprawki i ponownie uruchomić aplikację.

Dla zilustrowania takiej sytuacji dokonaj zmiany w tekście programu w trzecim wierszu wprowadzając błędny zapis:

r=*float(input("r="))

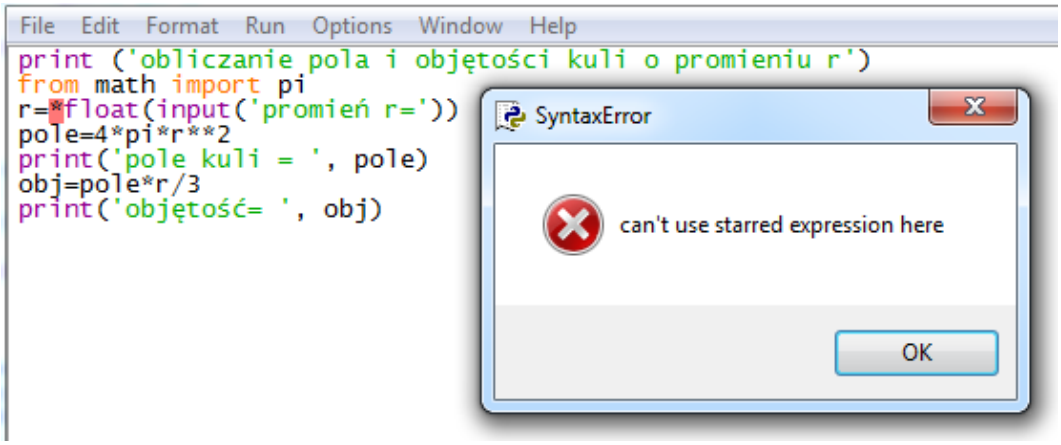
Przetestuj działanie aplikacji z błędem składniowym i wróć do poprzedniej (poprawnej wersji programu).

W opracowywanych programach można popełnić też innego rodzaju błędy wykrywane dopiero podczas ich wykonywania. Zobaczysz to to zmieniając instrukcję w czwartym wierszu na:

obj=pole*R/3

W zapisie tym jest błąd ponieważ w wyrażeniu po prawej stronie znaku przypisania występuje duża litera **R. (Python rozróżnia wielkość liter!)** Przetestuj działanie aplikacji z błędem wykonania. Zwróć uwagę na fakt, że aplikacja jest uruchamiana i sygnalizacja błędu następuje dopiero w momencie wykonywania błędnej instrukcji.

Poniżej przedstawiono okienka z informacją o błędzie składniowym (Rys.2) i błędzie wykonania (Rys. 3)



Rys.2. Okienko z informacją o błędzie składniowym

```
========== RESTART: F:\PYTHON\kula.py
obliczanie pola i objętości kuli o promieniu r
promień r=1
pole kuli =  12.566370614359172
Traceback (most recent call last):
  File "F:\PYTHON\kula.py", line 6, in <module>
    obj=pole*R/3
NameError: name 'R' is not defined
>>>
```

Rys.3. Informacja o błędzie wykonania

Zadania dodatkowe

Napisz, uruchom i przetestuj skrypty:

- obliczający pole rombu o boku a i wysokości h (nazwa skryptu: romb),
- obliczający długość odcinka dla danych dwóch punktów o współrzędnych (xa,ya) oraz (xb,yb) (nazwa skryptu: odcinek),
- wyznaczający pole powierzchni trójkąta zdefiniowanego przez trzy punkty (xa,ya), (xb,yb) oraz (xc,yc). Należy wyznaczyć długości trzech boków trójkąta a, b i c (patrz zadanie poprzednie) a następnie wykorzystać wzór Herona:

$s = \sqrt{p(p-a)(p-b)(p-c)}$ gdzie $p = \frac{a+b+c}{2}$ (nazwa aplikacji Trojkat).