Wykład 2

- Podstawy języka Python

1

Język Python

- Stworzony przez holendra Guido van Rossum w 1991 roku
- Open Source
- Interpretowany, interaktywny, zorientowany obiektowo
- Przenośny
- Bogata biblioteka funkcji
- Dobra dokumentacja
- Łatwy do nauki, przejrzysty w zapisie
- Funkcje, moduły, klasy, pakiety
- Dynamiczne typowanie
- Automatyczne zarządzanie pamięcią
- Obsługa wyjątków
- Struktury wysokiego poziomu: zbiory, krotki, listy, słowniki
- Dostępny na wiele urządzeń i systemów operacyjnych

2

Popularność języków programowania

Sep 2024	Sep 2023	Change	Programming Language	Ratings	Change
1	1		Python	20.17%	+6.01%
2	3	↑	C++	10.75%	+0.09%
3	4	↑	Java	9.45%	-0.04%
4	2	↓	C	8.89%	-2.38%
5	5		C#	6.08%	-1.22%
6	6		JavaScript	3.92%	+0.62%
7	7		Visual Basic	2.70%	+0.48%
8	12	↑	Go	2.35%	+1.16%
9	10	↑	SQL	1.94%	+0.50%
10	11	↑	Fortran	1.78%	+0.49%
11	15	↑	Delphi/Object Pascal	1.77%	+0.75%
12	13	↑	MATLAB	1.47%	+0.28%
13	8	↓	PHP	1.46%	-0.09%
14	17	↑	Rust	1.32%	+0.35%
15	18	↑	R	1.20%	+0.23%

https://www.tiobe.com/tiobe-index/

3

Micro Python

Układ	Raspberry Pi RP2040
Rdzeń	ARM Cortex-M0+ Dual-Core 133 MHz
Pamięć SRAM	512 kB
Pamięć Flash	4 MB
Wyprowadzenia	40-pin / 23 GPIO cyfrowe + 3 piny ADC
Interfejsy	WiFi IEEE 802.11 b/g/n, 2x UART, 2x I2C, 2x SPI, do 16 kanałów PWM
Napięcie zasilania	5 V (USB)
Wymiary	51 x 21 mm

4

Instrukcja przypisania

<zmienna> = <wyrażenie>

```
>>> a = 23
>>> a = ”to jest napis”
>>> print(a)
to jest napis
>>> b, c = 6, ’kotek’
>>> print(b, c)
6 kotek
>>> c, b = b, c
>>> print(b, c)
kotek 6
>>> x = y = z = 0
```

5

Typowanie statyczne

```
n : int = 23
imie : str = ’Jola’
pierwsze = list[int] = [2,3,5,7]

def add(a: int, b: int) -> int:
    return a + b
```

Narzędzia analizy kodu:
- mypy
- pyright
- pylint

6

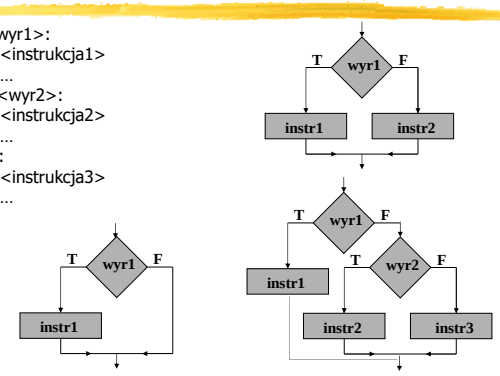
Operacje wejścia/wyjścia

```
>>> a = input(”Jak masz na imie: ”)
Jak masz na imie: Marek
>>> print(a)
Marek
>>> b = int(input(”Podaj liczbę całkowitą: ”))
Podaj liczbę całkowitą: 23
>>> print(b)
23
>>> c = float(input(”Podaj liczbę rzeczywistą: ”))
Podaj liczbę rzeczywistą: 0.1428
>>> print(c)
0.1428
```

7

Instrukcja skoku warunkowego

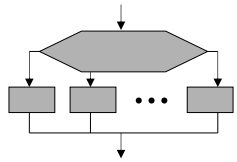
```
if <wyr1>:
    <instrukcja1>
    ...
elif <wyr2>:
    <instrukcja2>
    ...
else:
    <instrukcja3>
    ...
```



8

Instrukcja wyboru

```
match <term>:
  case <pattern_1>:
     <action_1>
  case <pattern_2>:
     <action_2>
  ...
  case _:
     <action-default>
```



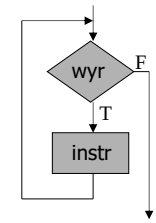
Przykład:

```
match dzien_tygodnia:
  case 0 : print(”niedziela”)
  case 1 : print(”poniedziałek”)
  ...
  case 6 : print(”sobota”)
  case _ : print(”zla wartosc”)
```

9

Instrukcja pętli while

```
while <wyrażenie>:
  <instrukcja>
  ...
```



Przykład:

```
n = int(input(”podaj liczbę”))
while n!=1:
  if n%2==0:
    n = n//2
  else:
    n = 3*n+1

print(”koniec”)
```

10

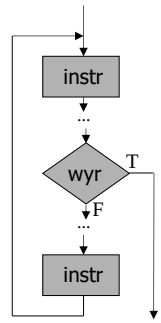
Przerywanie pętli - break

Przykład:

Wczytaj ciąg liczb zakończony zerem i oblicz ich średnią arytmetyczną

```
suma = 0
licznik = 0
while True:
  a=int(input(”podaj liczbę”))
  if a==0: break
  suma = suma+a
  licznik = licznik+1

print(suma/licznik)
```



11

Przerywanie pętli - break

Wczytaj ciąg liczb zakończony zerem, którego suma nie przekracza 100 i oblicz ich średnią arytmetyczną

```
suma = 0
licznik = 0
while suma<=100:
  a=int(input("podaj liczbe"))
  if a==0: break
  suma += a
  licznik += 1
else:
  print('Uwaga suma>100’)

print('srednia ', suma/licznik)
```

12

Instrukcja pętli for

for <zmienna> in <sekwencja>:
  <instrukcja>
  …

Zastosowanie:
Iterowanie po liczbach, napisach, listach, krotkach, zbiorach, słownikach, plikach

Przyklady:

```
for x in range(1, 6):
    print(x,"*",x," = ",x*x)
```

```
1 * 1  =  1
2 * 2  =  4
3 * 3  =  9
4 * 4  =  16
5 * 5  =  25
```

```
for x in "Python":
    print(x)
```

```
P
y
t
h
o
n
```

13

Assert

assert <warunek> [, <komunikat>]

Przykład:

```
def nwd(a,b):
  assert a>0 and b>0, 'a,b musza być większe od zera'

  while a!=b:
    if a>b: a = a-b
    else: b = b-a

  return a
```

Uwaga:
Uruchomienie Pythona z opcją -O powoduje generowanie kodu bez asercji.

14

Wyjątki

try:
  # kod w którym może wystąpić błąd
except [<typ błędu>]:
  # kod wykonywany po wystąpieniu błędu

Przykład:

```
while True:
    try:
        a = float(input("a="))
        b = float(input("b="))
        c = float(input("c="))
        break
     except ValueError:
        print("Podaj 3 liczby !!!")
  # end while
  d = b*b-4*a*c
```

15

Procedury i funkcje

Cel stosowania:
- dekompozycja problemu
- wielokrotne wykonanie
- poziomy abstrakcji
- oddzielna kompilacja
- możliwość użycia rekurencji

def <name> (<arg1>, <arg2>, ...):
  <instrukcja>
  …
end def

return # procedura
return *expression* # funkcja

16

Procedury i funkcje

```
def nwd(a, b):
    """Funkcja oblicza największy wspólny dzielnik.
        Nie jest najszybsza ale działa."""
    while a != b:
        if a>b: a=a-b
        else: b=b-a
    return a
```

```
>>> help(nwd)
Help on function nwd in module my_lib:
nwd(a, b)
    Funkcja oblicza największy wspólny dzielnik.
    Nie jest najszybsza ale działa.

>>>nwd(23,6)
1
```

17

Przekazywanie parametrów

- Argumenty typów niemodyfikowalnych (bool, int, float, string, krotka) są przekazywane przez wartość.
- Przykład
-
```
def cube(x):
    x = x*x*x
    return x
# end def

n=2
w=cube(n)
print(n,w)      # 2 8
```

18

Przekazywanie parametrów

- Argumenty typów modyfikowalnych (zbiory, listy, słowniki) są przekazywane przez referencję.
- Przykład
- ```
  def zeruj(lista):
      for i in range(len(lista)):
          lista[i] = 0
      return
  # end def

  l=[2,3,5,7]
  zeruj(l)
  print(l)        # [0,0,0,0]
  ```

19

Pytania i zadania

- Czym różni się notacja BNF od notacji EBNF?
- Zapisać w notacji EBNF składnię instrukcji warunkowej oraz pętli.
- Zapisać w notacji EBNF składnię wyrażenia arytmetycznego w którym mogą wystąpić zmienne a,b,c, operatory +, * oraz nawiasy ( ).
- Tylko 7 liczb pierwszych spełnia warunek:

  sum_p(N)=N

  gdzie:

  sum_p(N) to suma p-tych potęg cyfr p-cyfrowej liczby N

  np. sum_p(2016)=16+0+1+1296=1313

  sum_p(2017)=16+0+1+2401=2418

  Należy napisać program odnajdujący wszystkie takie liczby.

24
  ```