print("Ala ma", 10, "lat", sep=" ", end="\n") #przecinek łączy liczby i ciągi

print("Ala " + "ma " + "kota") #plus tylko dla łączenia ciągów znaków

print() #pusta linia

ciag = input("Podaj jakis tekst") #tylko tekt, iczba wymaga kowersji

#znak uieczki \\

print("\nNaciśnij klawisz Enter, aby zakończyć program.")

input()

print("KONIEC.")

----------------------------------------

print(0o123) #liczba ósemkowa

print(0x123) #liczba szesnastkowa

\*\* potęga

% modulo

// cześć całkowita z dzielenia

----------------------------------------

tekst = "x" \* 3 #xxx

----------------------------------------

x, y, z = 5, 10, 8

----------------------------------------

liczba = int(ciag) konwersja na int

liczba = float(ciag) na float

ciag = str(liczba) na string

print(type(zmienna)) typ zmiennej

----------------------------------------

if ładnaPogoda:

if znajdziemyDobrąRestaurację:

zjemyLunch()

else:

zjemyKanapki()

else:

if biletyDostępne:

pójdziemyDoKina()

else:

pójdziemyNaZakupy()

----------------------------------------

if ładnaPogoda:

pójdziemyNaSpacer()

elif biletyDostępne:

pójdziemyDoKina()

elif dostaniemyStolik:

zjemyKolacje()

else:

zagramyWSzachyWDomu()

----------------------------------------

if liczba1 > liczba2: większaLiczba = liczba1

else: większaLiczba = liczba2

----------------------------------------

największaLiczba = max(liczba1, liczba2, liczba3)

----------------------------------------

podatek = round(podatek, 0) #zaokrąglanie

----------------------------------------

while wyrażenie\_warunkowe:

instrukcja1

instr2

----------------------------------------

for i in range(100):

instr1

instr2

for i in range(2, 8):

print("Wartość i wynosi obecnie", i)

for i in range(2, 8, 3):

print("Wartość i wynosi obecnie", i)

break i continue dla for

----------------------------------------

import time

time.sleep(1) #czeka 1 sek

----------------------------------------

slowo\_uzytkownika = slowo\_uzytkownika.upper()

----------------------------------------

slowo = "kaloryfer"

for litera in slowo:

print(litera)

----------------------------------------

i = 1

while i < 5:

print(i)

i += 1

else:

print("else:", i) #5

----------------------------------------

for i in range(5):

print(i)

else:

print("else:", i) #4

------------------------------------------------

#petla w dół

for i in range(6, 1, -2):

print(i, end=" ") # wyjście: 6, 4, 2

-----------------------------------------

n = range(4)

for num in n:

print(num - 1)

else:

print(num)

----------------------------------------

**Operatory bitowe**

& (et) - koniunkcja bitowa;

| (kreska pionowa) - alternatywa bitowa;

~ (tylda) - negacja bitowa;

^ (daszek) - alternatywa rozłączna bitowa (xor).

----------------------------------------

zwrotka = print("rutututu") #rutututu

print(zwrotka) #None - literał brak wartości

----------------------------------------

przesuniecie bitowe

var = 17

var\_w\_prawo = var >> 1 #o 1 bit w prawp /2

var\_w\_lewo = var << 2 #o 2 bity w lewo \*4

print(var, var\_w\_lewo, var\_w\_prawo)

----------------------------------------

**Tabelka z priorytetami operatorow**

Priorytet Operator

1 ~, +, - jednoargumentowe

2 \*\*

3 \*, /, //, %

4 +, - dwuargumentowe

5 <<, >>

6 <, <=, >, >=

7 ==, !=

8 &

9 |

10 =, +=, -=, \*=, /=, %=, &=, ^=, |=, >>=, <<=

--------------------------------------

**listy**

moja\_lista = [] # Tworzenie pustej listy.

liczby = [10, 5, 7, 2, 1]

liczby[0] = 111

liczby = [10, 5, 7, 2, 1]

print("Zawartość listy:", liczby) # wyświetlanie oryginalnej zawartości listy

print(len(liczby)) # wyświetlanie długości listy

del liczby[1] #brak indexu 4, liczby maja nowe indexy

liczby = [111, 7, 2, 1]

print(liczby[-1]) #od tylu liczymu od 1

print(liczby[-2])

liczby.append(4) #dodanie na koncu listy

liczby.insert(0, 222) #wstaw na pozycji 0 wartość 222, inne przesuńw prawo

liczby.sort()

liczby.reverse()

-----------------------------------

moja\_lista = [1, None, True, "Jestem łańcuchem znaków", 256, 0]

moja\_lista = [1, 'a', ["list", 64, [0, 1], False]]

moja\_lista = [10, 1, 8, 3, 5]

suma = 0

for i in range(len(moja\_lista)):

suma += moja\_lista[i]

print(suma)

lub

for i in moja\_lista:

sum += i

--------------------------------------

**odwracanie kolejnosci elementow listy**

lista = [10, 1, 8, 3, 5]

d = len(lista)

for i in range(d // 2):

lista[i], lista[d - i - 1] = lista[d - i - 1], lista[i]

print(lista)

--------------------------------------

del moja\_lista # usuwa całą listę

--------------------------------------

**sortowanie babelkowe**

lista = [8, 10, 6, 2, 4] # lista do posortowania

swapped = True # to trochę fałszywe - potrzebujemy tego, by wprowadzić pętlę while

while swapped:

swapped = False # jak na razie bez zamian

for i in range(len(lista) - 1):

if lista[i] > lista[i + 1]:

swapped = True # zamiana wystąpiła!

lista[i], lista[i + 1] = lista[i + 1], lista[i]

print(lista)

----------------------------------------

lista1 = [1]

lista2 = lista1 #obie nazwy dotyczą tej samej listy

lista1[0] = 2

print(lista2)

----------------------------------------

# Kopiowanie całej listy.

list1 = [1]

list2 = list1[:] #wycinek listy, zupełnie nowa lista

list1[0] = 2

print(list2)

# Kopiowanie części listy.

lista = [10, 8, 6, 4, 2]

nowa\_lista = lista[1:3] #lista[start:end +1] od 1 do 2 > [8,6]

print(nowa\_lista)

-----------------------------------------

lista = [10, 8, 6, 4, 2]

nowa\_lista = lista[1:-1] #od indexu 1 do przedostatniego

print(nowa\_lista)

Wynik tego fragmentu: [8, 6, 4].

----------------------------------------

lista = [10, 8, 6, 4, 2]

nowa\_lista = lista[-1:1]

print(nowa\_lista)

Wynik działania programu powyżej to: [].

---------------------------------------

lista = [10, 8, 6, 4, 2]

nowa\_lista = lista[:3] #od 0

print(nowa\_lista)

Oto dlaczego wynik tego fragmentu to: [10, 8, 6].

---------------------------------------

lista = [10, 8, 6, 4, 2]

nowa\_lista = lista[3:] #do konca

print(nowa\_lista)

Wynik tego fragmentu kodu to: [4, 2].

--------------------------------------

lista = [10, 8, 6, 4, 2]

del lista[1:3]

print(lista)

--------------------------------------

lista = [10, 8, 6, 4, 2]

del lista[:] #pusta lista

print(lista)

Lista staje się pusta, a wynik to: [].

--------------------------------------

lista = [10, 8, 6, 4, 2]

del lista #usuwa liste z pamieci

print(lista) #blad, nie ma listy

--------------------------------------

lista = [0, 3, 12, 8, 2]

print(5 in lista) #czy 5 jest na liscie

print(5 not in lista) #czy nie ma na liscie

print(12 in lista)

---------------------------------------

**Znajdowanie najwiekszego elementu na liscie**

lista = [17, 3, 11, 5, 1, 9, 7, 15, 13]

najwieksza = lista[0]

for i in range(1, len(lista)):

if lista[i] > najwieksza:

najwieksza = lista[i]

print(najwieksza)

--------------------------------------

**szukanie elementu w liscie**

lista = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

do\_znalezienia = 5

znaleziono = False

for i in range(len(lista)):

znaleziono = lista[i] == do\_znalezienia

if znaleziono:

break

if znaleziono:

print("Element znaleziony pod indeksem", i)

else:

print("brak")

------------------------------------------

**sprawdzenie lotka**

wylosowano = [5, 11, 9, 42, 3, 49]

obstawiono = [3, 7, 11, 42, 34, 49]

trafiono = 0

for liczba in obstawiono:

if liczba in wylosowano:

trafiono += 1

print(trafiono)

--------------------------------------------

**Bez powtorzen**

moja\_lista = [1, 2, 4, 4, 1, 4, 2, 6, 2, 9]

#

# tu wstaw swój kod

#

temp = moja\_lista[:]

print(temp)

bez\_powtorzen = []

for i in range(len(temp)):

temp = moja\_lista[:]

liczba = temp[i]

del temp[i]

print(temp,liczba)

if liczba not in(temp):

bez\_powtorzen.append(liczba)

print("Lista tylko z unikalnymi elementami:")

#print(moja\_lista)

print(bez\_powtorzen)

---------------------------------------

**Wyrażenia listowe**

kwadraty = [x \*\* 2 for x in range(10)] # (0, 1, 4, 9, 16, 25, 36, 49, 64, 81)

dwojki = [2 \*\* i for i in range(8)] #(1, 2, 4, 8, 16, 32, 64, 128)

nieparzyste = [x for x in kwadraty if x % 2 != 0 ] #

---------------------------------------

**Szachownica**

szachownica = []

for i in range(8):

rzad = [PUSTY for i in range(8)]

szachownica.append(rzad)

lub prościej

szachownica = [[PUSTY for i in range(8)] for j in range(8)]

---------------------------------------

**Srenia temp w poludnie**

temp = [[0.0 for h in range(24)] for d in range(31)] #31wierszy po 24 kolumny

suma = 0.0

for dzien in temps:

suma += dzien[11]

srednia = suma / 31

print("Srednia temperatura w poludnie:", srednia)

---------------------------------------

**Najwyzsza temp w tablicy dwuwymiarowej**

najwyzsza = -100.0

for dzien in temps:

for temp in dzien:

if temp > najwyzsza:

najwyzsza = temp

print("Najwyzsza temperatura tego dnia:", najwyzsza)

---------------------------------------

**Tablica 3-wymiarowa,  trzy budynki, 15 pięter, 20 pokoi.**

pokoje = [[[False for r in range(20)] for f in range(15)] for t in range(3)]

pokoje[1][9][13] = True

**Sprawdzić, czy są jakieś wolne miejsca na 15. piętrze trzeciego budynku:**

wolne = 0

for numer\_pokoju in range(20):

if not pokoje[2][14][numer\_pokoju]:

wolne += 1

-----------------------------------------

#**definiowanie funkcji**

def podziel(dzielna, dzielnik):

print(dzilna / dzielnik)

def dzielenie(dzielna, dzielnik = 2)

print(dzielna / dzielnik)

podziel(dzielnik=2,dzielna=4)

podziel(2,4)

dzielenie(4)

dzielenie(4,3)

def przedstawienie(imie="Jan", nazwisko="Kowalski"):

print("Cześć, nazywam się", imie, nazwisko)

przedstawienie()

przedstawienie(nazwisko="Hopkins")

---------------------------------------

def odliczanie(zyczenia = True):

print("Trzy...")

print("Dwa...")

print("Jeden...")

if not zyczenia:

return

print("Szczęśliwego Nowego Roku!")

odliczanie()

odliczanie(Flse)

---------------------------------------

def nudna\_funkcja():

return 123

x = nudna\_funkcja()

print("nudna\_funkcja zwróciła swój wynik:", x)

---------------------------------------

def funkcja():

x = 5

a = funkcja()

print(a) #None

---------------------------------------

Argument lista

def suma\_listy(lst): #argumentem jest lista

suma = 0

for elem in lst:

suma += elem

return suma

print(suma\_listy([5, 4, 3]))

---------------------------------------

Funkcja zwraca listę

def lista\_funkcja(n):

lista = []

for i in range(0, n):

lista.insert(0, i)

return lista

print(lista\_funkcja(5))

---------------------------------------

def czy\_przestepny(rok):

if (rok%4==0 and rok%100!=0) or rok%400==0:

return True

else:

return False

dane\_testowe = [1900, 2000, 2016, 1987]

wyniki\_testow = [False, True, True, False]

for i in range(len(dane\_testowe)):

r = dane\_testowe[i]

print(r,"->",end="")

wynik = czy\_przestepny(r)

if wynik == wyniki\_testow[i]:

print("OK")

else:

print("Nie powiodło się")

---------------------------------------

def czy\_pierwsza(liczba):

for i in range(2, int(liczba\*\*0.5)+1):

if liczba % i == 0:

return False

return True

for i in range(1, 20):

if czy\_pierwsza(i + 1):

print(i + 1, end=" ")

print()

---------------------------------------

**Zasięg zmiennych**

def moja\_funkcja():

print("Czy znam tę zmienną?", var) #funkcja widzi zmienna var

var = 1

moja\_funkcja()

print(var)

---------------------------------------

def moja\_funkcja():

print("Czy znam tę zmienną?", var #zmienna już zaje print("Czy znam tę zmienną?", vara wewnątrz funkcji

var = 2 #tworzenie nowej zmiennj

print("Czy znam tę zmienną?", var)

var = 1

moja\_funkcja()

print(var)

---------------------------------------

def moja\_funkcja():

global var

print("Czy znam tę zmienną?", var)

var = 2 #tworzenie nowej zmiennj

print("Czy znam tę zmienną?", var)

var = 1

moja\_funkcja()

print(var)

---------------------------------------

Kontynuacja kodu

def bmi(waga, wzrost):

if wzrost < 1.0 or wzrost > 2.5 or \ #kontynuacja w następnej linii

waga < 20 or waga > 200:

return None

return waga / wzrost \*\* 2

print(bmi(352.5, 1.65))

---------------------------------------

**KROTKI – niemodyfikowalne listy**

pusta\_krotka = ()

krotka1 = (1, 2, 4, 8)

krotka2 = 1., .5, .25, .125

krotka\_jednoelementowa1 = (1, )

krotka\_jednoelementowa2 = 1.,

moja\_krotka = (1, 10, 100, 1000)

print(moja\_krotka[0])

print(moja\_krotka[-1])

print(moja\_krotka[1:])

print(moja\_krotka[:-2])

for elem in moja\_krotka:

print(elem)

---------------------------------------

moja\_krotka = (1, 10, 100)

t1 = moja\_krotka + (1000, 10000)

t2 = moja\_krotka \* 3

print(len(t2)) # 9

print(t1) #(1, 10, 100, 1000, 10000)

print(t2) #(1, 10, 100, 1, 10, 100, 1, 10, 100)

print(10 in moja\_krotka) #True

print(-10 not in moja\_krotka) #True

---------------------------------------

var = 123

t1 = (1, )

t2 = (2, )

t3 = (3, var)

t1, t2, t3 = t2, t3, t1

print(t1, t2, t3)

moja\_krotka = tuple((1, 2, "lancuch znakow")) #tworzenie krotki

print(moja\_krotka)

lst = [2, 4, 6]

print(lst) # daje na wyjściu: [2, 4, 6]

print(type(lst)) # daje na wyjściu: <class 'list'>

tup = tuple(lst) #konwersja na krotke

print(tup) # daje na wyjściu: (2, 4, 6)

print(type(tup)) # daje na wyjściu: <class 'tuple'>

tup = 1, 2, 3,

lst = list(tup) #konwersja na liste

print(type(lst)) # daje na wyjściu: <class 'list'>

tup = 1, 2, 3

a, b, c = tup

print(a \* b \* c) #6, krotka wypakowana do zmiennych abc

tup = 1, 2, 3, 2, 4, 5, 6, 2, 7, 2, 8, 9

duplicates = tup.count(2) #ilość wystąpień wartości 2 w krotce

print(duplicates) # wyjście: 4

---------------------------------------

**SŁOWNIKI**

slownik = {"kot" : "gato", "pies" : "perro", "koń" : "caballo"}

numery\_telefonow = {'szef' : 5551234567, 'Marian' : 22657854310}

pusty\_slownik = {}

print(slownik)

print(numery\_telefonow)

print(pusty\_slownik)

print(slownik['kot'])

print(numery\_telefonow['Marian'])

slownik['kot'] = 'gatito' #zmiana wartości

slownik['świnka'] = 'paperas' #dodanie nowego klucza i wartości

slownik.update({"kurczak" : "pollo"}) #dodawanie

del slownik['pies'] #usuwanie

slownik.popitem() #usuwanie ostatniego elementu

slownik.clear() # usuwa wszystkie elementy

kopia\_pol\_ang = pol\_ang.copy() #kopia całego slownika

slowa = ['kot', 'lew', 'koń']

for slowo in slowa:

if slowo in slownik:

print(slowo, "->", slownik[slowo])

else:

print("W słowniku nie istnieje słowo:", slowo)

for klucz in slownik.keys(): #iteracja po kluczach slownika

print(klucz, "->", slownik[klucz]

for klucz in sorted(slownik.keys()): #mozna posortować klucze

slownik = {"kot" : "gato", "pies" : "perro", "koń" : "caballo"}

for polski, hiszpanski in slownik.items(): #zwracane krotki

print(polski, "->", hiszpanski)

for hiszpanski in slownik.values(): #zwraca wartości

print(hiszpanski)

---------------------------------------

d1 = {'Adam Smith':'A', 'Judy Paxton':'B+'}

d2 = {'Mary Louis':'A', 'Patrick White':'C'}

d3 = {}

for item in (d1, d2): #iteracja po kilku słownikach

d3.update(item)

print(d3)

---------------------------------------

colors = (("green", "#008000"), ("blue", "#0000FF"))

col\_dict = dict(colors) #konwersja na slownik

print(col\_dict)

**LOSOWANIE liczb**

from random import randrange

print(randrange(8)) #losowa liczba 0-8

---------------------------------------

**MODUŁY**

import math, sys

import math

print(math.sin(math.pi/2))

---------------------------------------

**Przestrzeń nazw**

import math

def sin(x): #nazwa możliwa do wykorzystania

if 2 \* x == pi:

return 0.99999999

else:

return None

pi = 3.14

print(sin(pi/2))

print(math.sin(math.pi/2)) #nazwa sin z przestrzeni nazw math

---------------------------------------

from math import pi #import TYLKO konkretnego elementu modułu

from math import sin, pi

from modul import \*

from math import e as euler

import math as m

pi = 3

print(m.sin(pi/2))

print(m.pi)

---------------------------------------

**Zawartość modułu**

import math

for name in dir(math):

print(name, end="\t")

---------------------------------------

Print(pow(3,2) #pow funkcja wbudowana, nie wymaga importowania math

* ceil(x) → sufit liczby x (najmniejsza liczba całkowita większa od lub równa liczbie x)
* floor(x) → podłoga liczby x (największa liczba całkowita mniejsza od lub równa liczbie x)
* trunc(x) → wartość liczby x podana w formie liczby całkowitej (uwaga - to nie jest to samo, co funkcja ceil lub funkcja floor)
* factorial(x) → zwraca x! (x musi być całką i nie może być liczbą ujemną)
* hypot(x, y) → zwraca długość przeciwprostokątnej trójkąta prostokątnego, którego długości ramion są równe wartościom x oraz y (podobnie jak sqrt(pow(x, 2) + pow(y, 2)), ale dokładniejsze)

---------------------------------------

**LOSOWANIE**

from random import random, seed

seed() #ustalenie ziarna jako bieżącego czasu

for i in range(5):

print(random())

print(randrange(10), end=' ')

print(randrange(8, 15), end=' ')

print(randrange(5, 15, 2), end=' ') #od 5 do 15 z krokiem 2

print(randint(0, 10)) #losowanie <0,10> #przedział zamknięty

---------------------------------------

from random import choice, sample

lst = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

print(choice(lst)) #losowy element z listy

print(sample(lst, 5)) #nowa uporządkowana lista z elementów

print(sample(lst, 10))

---------------------------------------