
	<p>Politechnika Bydgoska im. J. J. Śniadeckich</p> <p><b>Wydział Telekomunikacji, Informatyki i Elektrotechniki</b></p>		
<b>Przedmiot</b>	Skryptowe języki programowania		
<b>Prowadzący</b>	mgr inż. Martyna Tarczewska		
<b>Temat</b>	<i>Więcej liczb</i>		
<b>Student</b>	Marcin Ogórkiewicz		
<b>Nr ćw.</b>	5	<b>Data wykonania</b>	06.11.2023
<b>Ocena</b>		<b>Data oddania spr.</b>	06.11.2023

#### Zadanie 1

```
#!C:\Users\koksu\AppData\Local\Programs\Python\Python310

# importy
import typing

# stałe i zmienne globalne

# funkcje

def get_base() -> int:
    while True:
        try:
            base = int(input("Podaj podstawę systemu liczbowego (2, 8, 10
lub 16): "))
            if base in [2, 8, 10, 16]:
                return base
            else:
                print("Podana wartość nie jest prawidłową podstawą systemu
liczbowego.")
        except ValueError:
            print("Podaj liczbę całkowitą (2, 8, 10 lub 16).")

def convert_number(number: str, base: int) -> str:
    if base == 2:
        return bin(number)
    elif base == 8:
        return oct(number)
    elif base == 10:
        return str(number)
    elif base == 16:
```

```

        return hex(number)

def main() -> None:
    base = get_base()
    number = input(f"Podaj liczbę w systemie o podstawie {base}: ")
    try:
        number = int(number, base)
        print(f"W postaci dwójkowej: {convert_number(number, 2)}")
        print(f"W postaci ósemkowej: {convert_number(number, 8)}")
        print(f"W postaci dziesiętnej: {convert_number(number, 10)}")
        print(f"W postaci heksadecymalnej: {convert_number(number, 16)}")
    except ValueError:
        print("Podana liczba nie jest prawidłowa w wybranym systemie
liczbowym.")

main()

```

```

C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\Lab05\Zadanie1.py
Podaj podstawę systemu liczbowego (2, 8, 10 lub 16): 10
Podaj liczbę w systemie o podstawie 10: 10
W postaci dwójkowej: 0b1000
W postaci ósemkowej: 0o10
W postaci dziesiętnej: 8
W postaci heksadecymalnej: 0x8

Process finished with exit code 0

```

## Zadanie 2

```

#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing

# stałe i zmienne globalne

# funkcje

def get_bit_value(number: int, bit_index: int) -> int:
    if 0 <= number <= 255 and 0 <= bit_index < 8:
        bit_mask = 1 << bit_index
        bit_value = (number & bit_mask) >> bit_index
        return bit_value
    else:
        print("Podano nieprawidłowe wartości. Zakres liczb to 0-255, a
indeks bitu od 0 do 7.")

def main() -> None:
    number = int(input("Podaj liczbę (0-255): "))
    bit_index = int(input("Podaj indeks bitu (0-7): "))
    result = get_bit_value(number, bit_index)
    if result is not None:
        print(f"Wartość bitu na bicie o indeksie {bit_index} to: {result}")

```

```
main()
```

```
C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\Lab05\Zadanie2.py
Podaj liczbę (0-255): 13
Podaj indeks bitu (0-7): 2
Wartość bitu na bicie o indeksie 2 to: 1

Process finished with exit code 0
```

### Zadanie 3

```
#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing
import random
# stałe i zmienne globalne

# funkcje

def calculate_triangle_area() -> tuple:
    side1 = random.randint(3, 10)
    side2 = random.randint(3, 10)
    side3 = random.randint(3, 10)
    if (side1 + side2 > side3) and (side1 + side3 > side2) and (side2 +
side3 > side1):
        perimeter = side1 + side2 + side3
        half_perimeter = perimeter / 2
        area = (half_perimeter * (half_perimeter - side1) * (half_perimeter
- side2) * (half_perimeter - side3))**0.5
        return side1, side2, side3, area

def main() -> None:
    result = calculate_triangle_area()
    if result:
        side1, side2, side3, area = result
        print(f"Losowe boki trójkąta: {side1}, {side2}, {side3}")
        print(f"Pole trójkąta wynosi: {area:.2f}")
    else:
        print("Nie można utworzyć trójkąta z wylosowanych boków.")

main()
```

```
C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\Lab05\Zadanie3.py
Losowe boki trójkąta: 6, 7, 7
Pole trójkąta wynosi: 18.97

Process finished with exit code 0
```

## Zadanie 4

```
#!C:\Users\koksu\AppData\Local\Programs\Python\Python310

# importy
import typing
import random
# stałe i zmienne globalne

# funkcje

def flip_coin() -> int:
    result = random.randint(0, 1)
    return result

def main() -> None:
    games_played = 0
    games_won = 0

    while True:
        user_choice = input("Obstaw orła (O) lub reszkę (R), lub wprowadź  
'X' aby zakończyć: ").upper()

        if user_choice == 'X':
            break # Zakończ grę

        if user_choice not in ('O', 'R'):
            print("Nieprawidłowy wybór. Wprowadź 'O', 'R' lub 'X'.")
            continue

        # Rzut monetą
        coin_result = flip_coin()
        if (coin_result == 0 and user_choice == 'O') or (coin_result == 1
and user_choice == 'R'):
            print("Wygrałeś!")
            games_won += 1
        else:
            print("Przegrałeś.")

        games_played += 1

    print(f"Liczba rozegranych gier: {games_played}")
    print(f"Liczba wygranych gier: {games_won}")

main()
```

```

C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\Lab05\Zadanie4.py
Obstaw orła (O) lub reszkę (R), lub wprowadź 'X' aby zakończyć: R
Wygrałeś!
Obstaw orła (O) lub reszkę (R), lub wprowadź 'X' aby zakończyć: R
Wygrałeś!
Obstaw orła (O) lub reszkę (R), lub wprowadź 'X' aby zakończyć: R
Przegrałeś.
Obstaw orła (O) lub reszkę (R), lub wprowadź 'X' aby zakończyć: X
Liczba rozegranych gier: 3
Liczba wygranych gier: 2
Process finished with exit code 0

```

## Zadanie 5

```

#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing
import random

# stałe i zmienne globalne

# funkcje

def main() -> None:
    games_played = 0
    games_won = 0
    while True:
        user_choice = input("Wybierz: papier (P), kamień (R), nożyce (S),  
lub 'X' aby zakończyć: ").upper()
        if user_choice == 'X':
            break
        if user_choice not in ('P', 'R', 'S'):
            print("Nieprawidłowy wybór. Wprowadź 'P', 'R', 'S' lub 'X'.")
            continue
        computer_choice = random.choice(['P', 'R', 'S'])
        print(f"Twój wybór: {user_choice}")
        print(f"Wybór komputera: {computer_choice}")
        if user_choice == computer_choice:
            print("Remis!")
        elif (user_choice == 'P' and computer_choice == 'R') or  
(user_choice == 'R' and computer_choice == 'S') or (user_choice == 'S' and  
computer_choice == 'P'):
            print("Wygrałeś!")
            games_won += 1
        else:
            print("Komputer wygrał.")
            games_played += 1
        print(f"Liczba rozegranych gier: {games_played}")
        print(f"Liczba wygranych gier: {games_won}")

main()

```

```

C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\Lab05\Zadanie5.py
Wybierz: papier (P), kamień (R), nożyce (S), lub 'X' aby zakończyć: 
Twój wybór: P
Wybór komputera: P
Remis!
Wybierz: papier (P), kamień (R), nożyce (S), lub 'X' aby zakończyć: 
Twój wybór: R
Wybór komputera: R
Remis!
Wybierz: papier (P), kamień (R), nożyce (S), lub 'X' aby zakończyć: 
Twój wybór: S
Wybór komputera: R
Komputer wygrał.
Wybierz: papier (P), kamień (R), nożyce (S), lub 'X' aby zakończyć: 
Liczba rozegranych gier: 3
Liczba wygranych gier: 0

Process finished with exit code 0

```

## Zadanie 6

```

#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing
import math
# stałe i zmienne globalne

# funkcje

def calculate_ladder_height(length: float, angle_degrees: float) -> float
or str:
    if length > 0 and angle_degrees >= 0:
        angle_radians = math.radians(angle_degrees)
        ladder_height = length * math.sin(angle_radians)
        return ladder_height
    else:
        return "Nieprawidłowa długość drabiny bądź kąt"

def main() -> None:
    length = float(input("Podaj długość drabiny: "))
    angle_degrees = float(input("Podaj kąt drabiny względem poziomu (w stopniach): "))
    ladder_height = calculate_ladder_height(length, angle_degrees)
    print(f"Wysokość, na jaką sięga koniec drabiny, wynosi:
{ladder_height:.2f}")

main()

```

```

C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\Lab05\Zadanie6.py
Podaj długość drabiny: 
Podaj kąt drabiny względem poziomu (w stopniach): 
Wysokość, na jaką sięga koniec drabiny, wynosi: 2.50

Process finished with exit code 0

```

## Zadanie 7

```

#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing
import math
# stałe i zmienne globalne

# funkcje

def main() -> None:
    for angle_degrees in range(91):
        angle_radians = math.radians(angle_degrees)
        sin_alfa = math.sin(angle_radians)
        cos_alfa = math.cos(angle_radians)
        if pow(sin_alfa, 2)+pow(cos_alfa, 2) != 1:
            print(f"Równanie nie zostało spełnione dla kąta
{angle_degrees}.")
        elif angle_degrees == 90:
            print("Koniec sprawdzania.")

main()

```

```

C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\env\Scripts\python.exe C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\Lab05\Zadanie7.py
Równanie nie zostało spełnione dla kąta 3.
Równanie nie zostało spełnione dla kąta 4.
Równanie nie zostało spełnione dla kąta 6.
Równanie nie zostało spełnione dla kąta 7.
Równanie nie zostało spełnione dla kąta 8.
Równanie nie zostało spełnione dla kąta 10.
Równanie nie zostało spełnione dla kąta 12.
Równanie nie zostało spełnione dla kąta 18.
Równanie nie zostało spełnione dla kąta 24.
Równanie nie zostało spełnione dla kąta 29.
Równanie nie zostało spełnione dla kąta 34.
Równanie nie zostało spełnione dla kąta 35.
Równanie nie zostało spełnione dla kąta 40.
Równanie nie zostało spełnione dla kąta 43.
Równanie nie zostało spełnione dla kąta 45.
Równanie nie zostało spełnione dla kąta 47.
Równanie nie zostało spełnione dla kąta 55.
Równanie nie zostało spełnione dla kąta 58.
Równanie nie zostało spełnione dla kąta 63.
Równanie nie zostało spełnione dla kąta 66.
Równanie nie zostało spełnione dla kąta 70.
Równanie nie zostało spełnione dla kąta 71.
Równanie nie zostało spełnione dla kąta 72.
Równanie nie zostało spełnione dla kąta 80.
Równanie nie zostało spełnione dla kąta 82.
Równanie nie zostało spełnione dla kąta 84.
Równanie nie zostało spełnione dla kąta 86.
Równanie nie zostało spełnione dla kąta 87.
Koniec sprawdzania.

```

Jedynka trygonometryczna nie sprawdza się w Pythonie.

## Zadanie 8

```

#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing

```

```

# stałe i zmienne globalne

# funkcje

def xor_cipher(text: str, key: str) -> str:
    encrypted_text = ""
    for i in range(len(text)):
        char = text[i]
        key_char = key[i]
        encrypted_char = chr(ord(char) ^ ord(key_char))
        encrypted_text += encrypted_char
    return encrypted_text

def main() -> None:
    plaintext = input("Podaj tekst:")
    keyword = input("Podaj słowo szyfrowe:")
    xor_text = xor_cipher(plaintext, keyword)
    xor_text2 = xor_cipher(xor_text, keyword)
    print("Tekst po operacji xor:", xor_text, xor_text2)

main()

```

```

C:\Users\koku\PycharmProjects\SkrzytoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkrzytoweJęzykiProgramowania\Lab05\Zadanie8.py
Podaj tekst: algorytm
Podaj słowo szyfrowe: algorytm
Tekst po operacji xor:
0000000 algorytm
Process finished with exit code 0

```

Użycie funkcji ze słowem niejawnym i tym samym słowem kodującym powoduje odszyfrowanie. Program nie zawsze jest w stanie wyświetlić niektóre znaki, ponieważ nie posiadają one graficznej reprezentacji.

## Zadanie 9

```

#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing

# stałe i zmienne globalne

# funkcje

def power_of_two_using_bit_shift(p: int) -> int:
    if p < 0:
        print("Potęgi liczb ujemnych nie są obsługiwane")
    return 1 << p

```



```
def main() -> None:
    try:
        exponent = int(input("Podaj wykładnik potęgi (p): "))
        result = power_of_two_using_bit_shift(exponent)
        print(f"2^{exponent} = {result}")
    except ValueError:
        print("Nieprawidłowe dane wejściowe.")

main()
```

```
C:\Users\koku\PycharmProjects\SkrypoweJęzykiProgramowania\env\Scripts\python.exe C:\Users\koku\PycharmProjects\SkrypoweJęzykiProgramowania\Lab05\Zadanie9.py
Podaj wykładnik potęgi (p): 5
2^5 = 32

Process finished with exit code 0
```

## Zadanie 10

```
#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing
import math
import sys

# stałe i zmienne globalne

# funkcje

def main() -> None:
    a = float(input("Podaj liczbę zmiennoprzecinkową (a): "))
    print(f"Math.trunc(a): {math.trunc(a)}")
    print(f"Math.floor(a): {math.floor(a)}")
    print(f"Math.ceil(a): {math.ceil(a)}")
    print(f"Math.fabs(a): {math.fabs(a)}")
    if sys.version_info >= (3, 9):
        b = float(input("Podaj drugą liczbę zmiennoprzecinkową (b): "))
        print(f"Math.lcm(a, b): {math.lcm(math.ceil(a), math.ceil(b))}")
        print(f"Math.gcd(a, b): {math.gcd(math.ceil(a), math.ceil(b))}")

main()
```

```

C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\Lab05\Zadanie10.py
Podaj liczbę zmiennoprzecinkową (a): 5
Math.trunc(a): 5
Math.floor(a): 5
Math.ceil(a): 5
Math.fabs(a): 5.0
Podaj drugą liczbę zmiennoprzecinkową (b): 10
Math.lcm(a, b): 10
Math.gcd(a, b): 5
Process finished with exit code 0

```

Działanie funkcji:

**Math.trunc** - podaje tylko część całkowitą liczby

**Math.floor** - podaje liczbę w zaokrągleniu do liczby mniejszej

**Math.ceil** - podaje liczbę w zaokrągleniu do liczby większej

**Math.abs** - podaje wartość bezwzględną liczby

**Math.lcm** - działa tylko na liczbach całkowitych, podaje najmniejszą wspólną wielokrotność

**Math.gcd** - działa tylko na liczbach całkowitych, podaje największy wspólny dzielnik

Zadanie 11

```

#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing
import pytest
import math

# stałe i zmienne globalne

# funkcje

def calculate_ladder_height(length: float, angle_degrees: float) -> float:
    angle_radians = math.radians(angle_degrees)
    ladder_height = length * math.sin(angle_radians)
    return ladder_height

def test_calculate_ladder_height_positive_angle():
    result = calculate_ladder_height(5.0, 30.0)
    assert pytest.approx(result, 0.01) == 2.5

```

```
def test_calculate_ladder_height_zero_length():
    result = calculate_ladder_height(0.0, 45.0)
    assert "Nieprawidłowa długość drabiny bądź kat"

def test_calculate_ladder_height_negative_length():
    result = calculate_ladder_height(-3.0, 60.0)
    assert "Nieprawidłowa długość drabiny bądź kat"
```

```
C:\Users\koku\PycharmProjects\SkrzytweJęzykiProgramowania\venv\Scripts\python.exe "C:/Program Files/JetBrains/PyCharm 2023.2.2/plugins/python/helpers/pycharm/_jb_pytest_run
Testing started at 01:39 ...
Launching pytest with arguments C:\Users\koku\PycharmProjects\SkrzytweJęzykiProgramowania\Lab05\Zadanie11.py --no-header --no-summary -q in C:\Users\koku\PycharmProjects\S

===== test session starts =====
collecting ... collected 3 items

Zadanie11.py::test_calculate_ladder_height_positive_angle PASSED      [ 33%]
Zadanie11.py::test_calculate_ladder_height_zero_length PASSED        [ 66%]
Zadanie11.py::test_calculate_ladder_height_negative_length PASSED     [100%]

===== 3 passed in 0.01s =====

Process finished with exit code 0
```

## Wnioski

Ćwiczenie pozwoliło mi przetrenować operacje na liczbach w języku Python. Zadania nie sprawiły mi problemu.