
	Politechnika Bydgoska im. J. J. Śniadeckich Wydział Telekomunikacji, Informatyki i Elektrotechniki Zakład Systemów Teleinformatycznych		
Przedmiot	Skryptowe języki programowania		
Prowadzący	mgr inż. Martyna Tarczewska		
Temat	Więcej liczb		
Student	Paweł Jońca		
Nr lab.	5	Data wykonania	11.11.2024r
Ocena		Data oddania spr.	11.11.2024r

Zad 1

```

1  def get_base(): 1usage new *
2      while True:# Funkcja pobiera podstawę systemu liczbowego od użytkownika i sprawdza poprawność
3          try:
4              base = int(input("Enter the base of the number system (2, 8, 10, or 16): "))
5              if base in [2, 8, 10, 16]:
6                  return base
7              else:
8                  print("Invalid base. Choose 2, 8, 10, or 16.")
9          except ValueError:
10             print("Invalid input. Please enter an integer.")
11
12 def get_number(base): 1usage new *
13     while True:# Funkcja pobiera liczbę od użytkownika w wybranej podstawie i konwertuje ją do systemu dziesiętnego
14         number = input(f"Enter a number in base {base}: ")
15         try:# Konwersja liczby do systemu dziesiętnego przy użyciu podanej podstawy
16             decimal_val = int(number, base)
17             return decimal_val
18         except ValueError:
19             print(f"Invalid number for base {base}. Please try again.")
20
21 def main(): 1usage new *
22     base = get_base()
23     decimal_number = get_number(base)
24     print("Number in binary system:", bin(decimal_number))
25     print("Number in octal system:", oct(decimal_number))
26     print("Number in decimal system:", decimal_number)
27     print("Number in hexadecimal system:", hex(decimal_number))
28
29 if __name__ == "__main__":
30     main()# Wywołanie głównej funkcji

```

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skryp
Enter the base of the number system (2, 8, 10, or 16): 1
Invalid base. Choose 2, 8, 10, or 16.
Enter the base of the number system (2, 8, 10, or 16): 2.1
Invalid input. Please enter an integer.
Enter the base of the number system (2, 8, 10, or 16): 2
Enter a number in base 2: 011011001
Number in binary system: 0b11011001
Number in octal system: 0o331
Number in decimal system: 217
Number in hexadecimal system: 0xd9

Process finished with exit code 0
```

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Do
Enter the base of the number system (2, 8, 10, or 16): 8
Enter a number in base 8: 767
Number in binary system: 0b111110111
Number in octal system: 0o767
Number in decimal system: 503
Number in hexadecimal system: 0x1f7

Process finished with exit code 0
|
```

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokun
Enter the base of the number system (2, 8, 10, or 16): 16
Enter a number in base 16: A02B
Number in binary system: 0b1010000000101011
Number in octal system: 0o120053
Number in decimal system: 41003
Number in hexadecimal system: 0xa02b

Process finished with exit code 0
|
```

Zad 2

```
1 def get_bit(num, index): 1usage new *
2     # Sprawdzenie, czy liczba jest w zakresie 0-255
3     if not (0 <= num <= 255):
4         raise ValueError("Liczba musi być w zakresie 0-255.")
5     # Sprawdzenie, czy indeks bitu jest nieujemny i mieści się w zakresie 0-7
6     if not (0 <= index <= 7):
7         raise ValueError("Indeks bitu musi być w zakresie 0-7.")
8
9     # Przesunięcie bitowe
10    return (num >> index) & 1
11
12 def main(): 1usage new *
13     try:#pobranie liczby oraz indeksu od użytkownika
14         num = int(input("Podaj liczbę (0-255): "))
15         index = int(input("Podaj indeks bitu (0-7): "))
16         result = get_bit(num, index)
17         print(f"Wartość bitu na pozycji {index} w liczbie {num} wynosi: {result}")
18     except ValueError as e:
19         print("Błąd:", e)
20 if __name__ == "__main__":
21     main()
22
```

Run zad2 x

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skryptowe J
Podaj liczbę (0-255): 7
Podaj indeks bitu (0-7): 2
Wartość bitu na pozycji 2 w liczbie 7 wynosi: 1

Process finished with exit code 0
```

Zad 3

```
import random
import math
def triangle(a, b, c): 1 usage new *
    # Sprawdza czy z wylosowanych boków można stworzyć trójkąt
    return a + b > c and a + c > b and b + c > a

def calculate(a, b, c): 1 usage new *
    # Oblicza pole trójkąta
    s = (a + b + c) / 2 # Obliczenie połowy obwodu
    return math.sqrt(s * (s - a) * (s - b) * (s - c))

def generate_calculate(): 1 usage new *
    # Losowanie trzech liczb z zakresu <3, 10>
    a = random.randint(a: 3, b: 10)
    b = random.randint(a: 3, b: 10)
    c = random.randint(a: 3, b: 10)
    print(f"Wylosowane boki: {a}, {b}, {c}")

    # Sprawdzenie, czy z wylosowanych boków można zbudować trójkąt
    if triangle(a, b, c):
        area = calculate(a, b, c)
        print(f"Z boków {a}, {b} i {c} można zbudować trójkąt. Pole trójkąta wynosi: {area:.2f}")
    else:
        print("Nie można utworzyć trójkąta z tych boków.")

def main(): 1 usage new *
    generate_calculate()
> if __name__ == "__main__":
    main()
```

zad3 x

⏏

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki progra
Wylosowane boki: 8, 4, 6
Z boków 8, 4 i 6 można zbudować trójkąt. Pole trójkąta wynosi: 11.62

Process finished with exit code 0
|
```

Zad 4

```
import random
def flip_coin(): 1 usage new *
    # Funkcja losująca wynik rzutu monetą: 0 oznacza orzeł, 1 oznacza reszkę
    return random.choice(["orzeł", "reszka"])
def coin_toss_game(): 1 usage new *
    games_played = 0
    wins = 0
    print("Hej, miło jest cię tu widzieć!")
    print("Wybierz 'orzeł' lub 'reszka'. Wpisz 'X', aby zakończyć grę.")
    while True:
        # Pobieranie wyboru użytkownika
        user_choice = input("Wybierz orzeł/reszka lub X, aby zakończyć: ").lower()
        if user_choice == "x":
            print("Koniec gry.")
            break
        elif user_choice not in ["orzeł", "reszka"]:
            print("Nieprawidłowy wybór. Wybierz 'orzeł', 'reszka' lub 'X'.")
            continue
        # Wykonanie rzutu monetą
        result = flip_coin()
        games_played += 1 # Zliczanie liczby gier
        print(f"Wynik rzutu: {result}")
        # Sprawdzanie, czy użytkownik wygrał
        if user_choice == result:
            print("Gratulacje, wygrałeś!")
            wins += 1 # Zliczanie wygranych
        else:
            print("Niestety, przegrałeś.")
        print(f"Liczba gier: {games_played}, Liczba wygranych: {wins}")
    # Wywołanie funkcji gry
if __name__ == "__main__":
    coin_toss_game()
```

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\G
Hej, miło jest cię tu widzieć!
Wybierz 'orzeł' lub 'reszka'. Wpisz 'X', aby zakończyć grę.
Wybierz orzeł/reszka lub X, aby zakończyć: orzeł
Wynik rzutu: reszka
Niestety, przegrałeś.
Wybierz orzeł/reszka lub X, aby zakończyć: reszka
Wynik rzutu: reszka
Gratulacje, wygrałeś!
Wybierz orzeł/reszka lub X, aby zakończyć: ress
Nieprawidłowy wybór. Wybierz 'orzeł', 'reszka' lub 'X'.
Wybierz orzeł/reszka lub X, aby zakończyć: x
Koniec gry.
Liczba gier: 2, Liczba wygranych: 1

Process finished with exit code 0
```

Zad 5

```
1 import random
2 def get_computer_choice(): 1 usage new *
3     return random.choice(["papier", "kamień", "nożyce"]) # Losowanie wyboru komputera
4 def determine_winner(user_choice, computer_choice): 1 usage new *
5     if user_choice == computer_choice: # Sprawdzenie wyników gry
6         return "remis"
7     elif (user_choice == "papier" and computer_choice == "kamień") or \
8           (user_choice == "kamień" and computer_choice == "nożyce") or \
9           (user_choice == "nożyce" and computer_choice == "papier"):
10        return "wygrana"
11    else:
12        return "przegrana"
13 def rock_paper_scissors_game(): 1 usage new *
14     games_played = 0
15     wins = 0
16     print("Witaj w grze 'Papier, Kamień, Nożyce'!")
17     print("Wybierz 'papier', 'kamień' lub 'nożyce'. Wpisz 'X', aby zakończyć grę.")
18     while True:
19         user_choice = input("Wybierz papier/kamień/nożyce lub X, aby zakończyć: ").lower()
20         if user_choice == "x":
21             print("Koniec gry.")
22             break
23         elif user_choice not in ["papier", "kamień", "nożyce"]:
24             print("Nieprawidłowy wybór. Wybierz 'papier', 'kamień', 'nożyce' lub 'X'.")
25             continue
26         # Losowanie wyboru komputera
27         comp_choice = get_computer_choice()
28         print(f"Komputer wybrał: {comp_choice}")
29         # Określenie wyniku gry
30         result = determine_winner(user_choice, comp_choice)
31         games_played += 1 # Zliczanie liczby gier
32         if result == "wygrana":
33             print("Gratulacje, wygrześ!")
34             wins += 1 # Zliczanie wygranych użytkownika
35         elif result == "przegrana":
36             print("Niestety, przegrałeś.")
37         else:
38             print("Remis.")
39         print(f"Liczba gier: {games_played}, Liczba wygranych: {wins}")
40 if __name__ == "__main__":
41     rock_paper_scissors_game()
42
```

```
↑ "C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skrypt
↓ Witaj w grze 'Papier, Kamień, Nożyce'!
Wpisz 'X', aby zakończyć grę.
Wybierz 'papier', 'kamień' lub 'nożyce'. Wpisz 'X', aby zakończyć grę.
Wybierz papier/kamień/nożyce lub X, aby zakończyć: papier
Komputer wybrał: kamień
Gratulacje, wygrześ!
Wybierz papier/kamień/nożyce lub X, aby zakończyć: nożyce
Komputer wybrał: papier
Gratulacje, wygrześ!
Wybierz papier/kamień/nożyce lub X, aby zakończyć: kamień
Komputer wybrał: kamień
Remis.
Wybierz papier/kamień/nożyce lub X, aby zakończyć: ka
Nieprawidłowy wybór. Wybierz 'papier', 'kamień', 'nożyce' lub 'X'.
Wybierz papier/kamień/nożyce lub X, aby zakończyć: x
Koniec gry.
Liczba gier: 3, Liczba wygranych: 2

Process finished with exit code 0
```

Zad 6

```
1 import math
2 def ladder_height(ladder_length, angle_degrees): 1 usage new *
3     angle_radians = math.radians(angle_degrees) # Konwersja kąta z stopni na radiany
4     height = ladder_length * math.sin(angle_radians) # Obliczenie wysokości
5     return height
6 def main(): 1 usage new *
7     print("Test na jaką wysokość sięgnie drabina")
8     try:
9         ladder_length = float(input("Podaj długość drabiny (w metrach): "))
10        angle_degrees = float(input("Podaj kąt między drabiną a poziomem (w stopniach): "))
11        if ladder_length <= 0 or angle_degrees <= 0 or angle_degrees >= 90:
12            print("Błędne dane. Długość drabiny musi być dodatnia, a kąt musi być z zakresu (0, 90) stopni.")
13            return
14        height = ladder_height(ladder_length, angle_degrees)
15        print(f"Wysokość, na jaką sięga koniec drabiny, wynosi: {height:.2f} metra/ów")
16    except ValueError:
17        print("Podaj poprawne wartości.")
18 > if __name__ == "__main__":
19     main()
20
```

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Documents\03 - studia
Test na jaką wysokość sięgnie drabina
Podaj długość drabiny (w metrach): 8
Podaj kąt między drabiną a poziomem (w stopniach): 45
Wysokość, na jaką sięga koniec drabiny, wynosi: 5.66 metra/ów

Process finished with exit code 0
```

Zad 7

```

import math
def check_trig_identity(): 1 usage new*
    for angle_degrees in range(91): # Zakres od 0 do 90 stopni
        # Konwersja kąta na radiany
        angle_radians = math.radians(angle_degrees)
        sin_square = math.sin(angle_radians) ** 2 # Obliczenie sin^2(theta) + cos^2(theta)
        cos_square = math.cos(angle_radians) ** 2
        result = sin_square + cos_square
        if round(result, 6) == 1.0: # Sprawdzenie, czy wynik jest równy 1 z dokładnością do 6 miejsc po przecinku
            print(f"Kąt: {angle_degrees}°, Wynik: {result:.6f} - Zgadza się")
        else:
            print(f"Kąt: {angle_degrees}°, Wynik: {result:.6f} - Nie zgadza się")
if __name__ == "__main__":
    check_trig_identity()

```

```

"C:\Program Files\Python312\python.exe" *C
Kąt: 0°, Wynik: 1.000000 - Zgadza się
Kąt: 1°, Wynik: 1.000000 - Zgadza się
Kąt: 2°, Wynik: 1.000000 - Zgadza się
Kąt: 3°, Wynik: 1.000000 - Zgadza się
Kąt: 4°, Wynik: 1.000000 - Zgadza się
Kąt: 5°, Wynik: 1.000000 - Zgadza się
Kąt: 6°, Wynik: 1.000000 - Zgadza się
Kąt: 7°, Wynik: 1.000000 - Zgadza się
Kąt: 8°, Wynik: 1.000000 - Zgadza się
Kąt: 9°, Wynik: 1.000000 - Zgadza się
Kąt: 10°, Wynik: 1.000000 - Zgadza się
Kąt: 11°, Wynik: 1.000000 - Zgadza się
Kąt: 12°, Wynik: 1.000000 - Zgadza się
Kąt: 13°, Wynik: 1.000000 - Zgadza się
Kąt: 14°, Wynik: 1.000000 - Zgadza się
Kąt: 15°, Wynik: 1.000000 - Zgadza się
Kąt: 16°, Wynik: 1.000000 - Zgadza się
Kąt: 17°, Wynik: 1.000000 - Zgadza się
Kąt: 18°, Wynik: 1.000000 - Zgadza się
Kąt: 19°, Wynik: 1.000000 - Zgadza się
Kąt: 20°, Wynik: 1.000000 - Zgadza się
Kąt: 21°, Wynik: 1.000000 - Zgadza się
Kąt: 22°, Wynik: 1.000000 - Zgadza się
Kąt: 23°, Wynik: 1.000000 - Zgadza się
Kąt: 24°, Wynik: 1.000000 - Zgadza się
Kąt: 25°, Wynik: 1.000000 - Zgadza się
Kąt: 26°, Wynik: 1.000000 - Zgadza się
Kąt: 27°, Wynik: 1.000000 - Zgadza się
Kąt: 28°, Wynik: 1.000000 - Zgadza się
Kąt: 29°, Wynik: 1.000000 - Zgadza się
Kąt: 30°, Wynik: 1.000000 - Zgadza się
Kąt: 31°, Wynik: 1.000000 - Zgadza się
Kąt: 32°, Wynik: 1.000000 - Zgadza się
Kąt: 33°, Wynik: 1.000000 - Zgadza się
Kąt: 34°, Wynik: 1.000000 - Zgadza się
Kąt: 35°, Wynik: 1.000000 - Zgadza się
Kąt: 36°, Wynik: 1.000000 - Zgadza się
Kąt: 37°, Wynik: 1.000000 - Zgadza się

```

```

Kąt: 38°, Wynik: 1.000000 - Zgadza się
Kąt: 39°, Wynik: 1.000000 - Zgadza się
Kąt: 40°, Wynik: 1.000000 - Zgadza się
Kąt: 41°, Wynik: 1.000000 - Zgadza się
Kąt: 42°, Wynik: 1.000000 - Zgadza się
Kąt: 43°, Wynik: 1.000000 - Zgadza się
Kąt: 44°, Wynik: 1.000000 - Zgadza się
Kąt: 45°, Wynik: 1.000000 - Zgadza się
Kąt: 46°, Wynik: 1.000000 - Zgadza się
Kąt: 47°, Wynik: 1.000000 - Zgadza się
Kąt: 48°, Wynik: 1.000000 - Zgadza się
Kąt: 49°, Wynik: 1.000000 - Zgadza się
Kąt: 50°, Wynik: 1.000000 - Zgadza się
Kąt: 51°, Wynik: 1.000000 - Zgadza się
Kąt: 52°, Wynik: 1.000000 - Zgadza się
Kąt: 53°, Wynik: 1.000000 - Zgadza się
Kąt: 54°, Wynik: 1.000000 - Zgadza się
Kąt: 55°, Wynik: 1.000000 - Zgadza się
Kąt: 56°, Wynik: 1.000000 - Zgadza się
Kąt: 57°, Wynik: 1.000000 - Zgadza się
Kąt: 58°, Wynik: 1.000000 - Zgadza się
Kąt: 59°, Wynik: 1.000000 - Zgadza się
Kąt: 60°, Wynik: 1.000000 - Zgadza się
Kąt: 61°, Wynik: 1.000000 - Zgadza się
Kąt: 62°, Wynik: 1.000000 - Zgadza się
Kąt: 63°, Wynik: 1.000000 - Zgadza się
Kąt: 64°, Wynik: 1.000000 - Zgadza się
Kąt: 65°, Wynik: 1.000000 - Zgadza się
Kąt: 66°, Wynik: 1.000000 - Zgadza się
Kąt: 67°, Wynik: 1.000000 - Zgadza się
Kąt: 68°, Wynik: 1.000000 - Zgadza się
Kąt: 69°, Wynik: 1.000000 - Zgadza się
Kąt: 70°, Wynik: 1.000000 - Zgadza się
Kąt: 71°, Wynik: 1.000000 - Zgadza się
Kąt: 72°, Wynik: 1.000000 - Zgadza się
Kąt: 73°, Wynik: 1.000000 - Zgadza się
Kąt: 74°, Wynik: 1.000000 - Zgadza się
Kąt: 75°, Wynik: 1.000000 - Zgadza się
Kąt: 76°, Wynik: 1.000000 - Zgadza się
Kąt: 77°, Wynik: 1.000000 - Zgadza się
Kąt: 78°, Wynik: 1.000000 - Zgadza się
Kąt: 79°, Wynik: 1.000000 - Zgadza się
Kąt: 80°, Wynik: 1.000000 - Zgadza się
Kąt: 81°, Wynik: 1.000000 - Zgadza się
Kąt: 82°, Wynik: 1.000000 - Zgadza się
Kąt: 83°, Wynik: 1.000000 - Zgadza się
Kąt: 84°, Wynik: 1.000000 - Zgadza się
Kąt: 85°, Wynik: 1.000000 - Zgadza się
Kąt: 86°, Wynik: 1.000000 - Zgadza się
Kąt: 87°, Wynik: 1.000000 - Zgadza się
Kąt: 88°, Wynik: 1.000000 - Zgadza się
Kąt: 89°, Wynik: 1.000000 - Zgadza się
Kąt: 90°, Wynik: 1.000000 - Zgadza się

```


Zad 8

```
1 def xor_encrypt_decrypt(plain_text, key): 2 usages new *
2     if len(plain_text) != len(key):
3         raise ValueError("Długość tekstu jawnego i słowa szyfrującego musi być taka sama.")
4     encrypted_text = ""
5     for i in range(len(plain_text)):
6
7         encrypted_char = ord(plain_text[i]) ^ ord(key[i]) # Pobieramy kod ASCII dla obu znaków i wykonujemy operację XOR
8
9         encrypted_text += chr(encrypted_char) # Konwertujemy wynik do znaku i dodajemy do wyniku
10    return encrypted_text
11 def main(): 1 usage new *
12    plain_text = "Pawełjon"
13    key = "kodykody"
14    encrypted_text = xor_encrypt_decrypt(plain_text, key) # Szyfrowanie
15    print(f"Tekst zaszyfrowany: {encrypted_text}")
16    decrypted_text = xor_encrypt_decrypt(encrypted_text, key) # Deszyfrowanie
17    print(f"Tekst odszyfrowany: {decrypted_text}")
18
19 > if __name__ == "__main__":
20     main()
21
```

```
Tekst zaszyfrowany: ;0000000
Tekst odszyfrowany: Pawełjon
```

Process finished with **exit** code 0

Zad 9

```
def power_of_two(p): 1 usage new *
    # Przesunięcie bitowe w lewo o p pozycji, co odpowiada 2^p
    return 1 << p
def main(): 1 usage new *
    try:
        p = int(input("Podaj wykładnik potęgi (p): "))
        result = power_of_two(p)
        print(f"2^{p} = {result}")
    except ValueError:
        print("Podano nieprawidłową wartość. Proszę podać liczbę całkowitą.")
> if __name__ == "__main__":
    main()
```

```
Podaj wykładnik potęgi (p): 5
2^5 = 32
```

Zad 10

```
1 import math
2 import sys
3 > def test_math_functions(): 1 usage new *
4     """Testuje metody math.trunc, math.floor, math.ceil dla przykładowej liczby zmiennoprzecinkowej."""
5     a = 5.7 # Przykładowa liczba do testów
6     print(f"math.trunc({a}) = {math.trunc(a)}") # Zaokrąglenie do 0
7     print(f"math.floor({a}) = {math.floor(a)}") # Zaokrąglenie w dół
8     print(f"math.ceil({a}) = {math.ceil(a)}") # Zaokrąglenie w górę
9
10 > def test_python_version_for_lcm_gcd(): 1 usage new *
11     """Sprawdza wersję Pythona i testuje metody math.lcm oraz math.gcd dla przykładowych liczb, jeśli wersja >= 3.9."""
12     if sys.version_info >= (3, 9):
13         print("\nWersja Pythona 3.9 lub wyższa wykryta.")
14         a, b = 15, 20 # Przykładowe liczby do testów
15         print(f"math.lcm({a}, {b}) = {math.lcm(*integers: a, b)}") # Najmniejsza wspólna wielokrotność
16         print(f"math.gcd({a}, {b}) = {math.gcd(*integers: a, b)}") # Największy wspólny dzielnik
17     else:
18         print("\nWersja Pythona jest poniżej 3.9, funkcje math.lcm() i math.gcd() nie będą testowane.")
19
20 > def test_check_abs_function(): 1 usage new *
21     """Sprawdza, czy moduł math zawiera funkcję do obliczania wartości bezwzględnej."""
22     if hasattr(math, "fabs"):
23         print(f"math.fabs(-10.5) = {math.fabs(-10.5)}") # Wartość bezwzględna liczby zmiennoprzecinkowej
24     else:
25         print("Moduł math nie zawiera funkcji do obliczania wartości bezwzględnej.")
26
27 def main(): 1 usage new *
28     test_math_functions() # Testowanie funkcji trunc, floor i ceil
29     test_python_version_for_lcm_gcd() # Testowanie wersji Pythona i funkcji lcm oraz gcd
30     test_check_abs_function() # Sprawdzenie funkcji wartości bezwzględnej
31 > if __name__ == "__main__":
32     main()
33
```

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

zad10.py::test_math_functions PASSED [ 33%]math.trunc(5.7) = 5
math.floor(5.7) = 5
math.ceil(5.7) = 6

zad10.py::test_python_version_for_lcm_gcd PASSED [ 66%]
Wersja Pythona 3.9 lub wyższa wykryta.
math.lcm(15, 20) = 60
math.gcd(15, 20) = 5

zad10.py::test_check_abs_function PASSED [100%]math.fabs(-10.5) = 10.5

===== 3 passed in 0.06s =====

Process finished with exit code 0
```

Zad 11

```

1 > import ...
3 # Test 1: Sprawdzamy liczbę całkowitą
4 def test_trunc_integer(): new *
5     assert math.trunc(5) == 5, "Test nie przeszedł! Oczekiwano 5."
6
7 # Test 2: Sprawdzamy liczbę zmiennoprzecinkową
8 def test_trunc_float(): new *
9     assert math.trunc(5.67) == 5, "Test nie przeszedł! Oczekiwano 5."
10
11 # Test 3: Sprawdzamy liczbę ujemną
12 def test_trunc_negative(): new *
13     assert math.trunc(-5.67) == -5, "Test nie przeszedł! Oczekiwano -5."
14

```

```

ms ✓ Tests passed: 3 of 3 tests - 0ms
"C:\Program Files\Python312\python.exe" "C:/users/pawel/AppData/Local/Programs/Python/Python312/Scripts/pycharm_professional_plugins/p
Testing started at 15:00 ...
Launching pytest with arguments C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\lab05\pyt

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

zad11.py::test_trunc_integer PASSED [ 33%]
zad11.py::test_trunc_float PASSED [ 66%]
zad11.py::test_trunc_negative PASSED [100%]

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

Process finished with exit code 0

```

Wnioski:

Wszystkie zadania nauczyły praktycznego podejścia do stosowania wbudowanych funkcji matematycznych. Należało skorzystać z modułu `math` i testowania różnych funkcji matematycznych które są dostępne w tym module. Przetestowane zostały `trunc`, `floor`, `ceil`, `lcm`, `gcd`, `fabs`. Zadania pokazały jak można operować na liczbach w pythonie co rozszerzyło moją wiedzę.