Zadanie 1

## CEL ZADANIA

Celem zadania było zapoznanie się z podstawowymi operatorami i typami danych w języku Python oraz sprawdzenie ich działania przy użyciu funkcji type().

## PRZEBIEG ZADANIA

W interpreterze Pythona wykonano działania arytmetyczne, takie jak dodawanie, dzielenie, potęgowanie i reszta z dzielenia. Następnie zastosowano funkcje konwersji typów: int(), float(), str() i bool() w celu sprawdzenia ich efektu.

## KOD PROGRAMU

a)

Zad1.py

```
print("1. ",type(1 + 2)) # "+" -  Operator dodawania
print("2. ",type(1 + 4.5)) # "+" - Operator dodawania
print("3. ",type(3/2)) # "/" - Operator dzielenia (z resztą)
print("4. ",type(4/2)) # "/" - Operator dzielenia (z resztą)
print("5. ",type(3//2)) # "/" Operator dzielenia całkowitego (bez reszty)
print("6. ",type(-3//2)) # "/" - Operator dzielenia całkowitego (bez reszty)
print("7. ",type(11 % 2)) # "%" - Operator modulo (wypisanie tylko reszty z dzielenia podanych liczb)
print("8. ",type(2 ** 10)) # "**" - Operator potęgowania
print("9. ",type(8 ** (1/3))) # "**" -  Operator potęgowania, "/" - Operator potęgowania (z resztą), "()" - służy do grupowanina wyrażeń i określania kolejności działań
```

b)

Zad1.py  Zad1_b.py

```
print("1. ",int(3.0)) # Funkcja "int()" zmienia liczbę zmiennoprzecinkową (float) na liczbę całkowitą (int)
print("2. ",float(3)) # Funkcja "float()" zmienia liczbę całkowitą (int) na liczbę zmiennoprzecinkową (float)
print("3. ",float("3.0")) # Funkcja "float()" zamienia napis (string), który ma w sobie liczbę, na liczbę zmiennoprzecinkową (float)
print("4. ",str(12.4)) # Funkcja "str()" zamienia liczbę (w tym przypadku zmiennoprzecinkową) na ciąg znaków/tekst (string)
print("5. ",bool(0)) # Funkcja "bool()" to logiczny typ danych, przyjmuje tylko dwie wartości (true/false) - wartość 0 to fałsz (false) a każda inna to prawda (true)
```

## WYNIK PROGRAMU

a)

```
1.  <class 'int'>
2.  <class 'float'>
3.  <class 'float'>
4.  <class 'float'>
5.  <class 'int'>
6.  <class 'int'>
7.  <class 'int'>
8.  <class 'int'>
9.  <class 'float'>
PS C:\Users\wiki_>
```

b)

```
1.  3
2.  3.0
3.  3.0
4.  12.4
5.  False
PS C:\Users\wiki_>
```

## ANALIZA WYNIKÓW

Działania zwracały wyniki o odpowiednich typach (int, float). Funkcje konwersji poprawnie zmieniały typy danych.
Przykładowo:

- 1 + 2 → wynik: 3, typ int
- 1 + 4.5 → wynik: 5.5, typ float
- float("3") → konwersja tekstu na liczbę
- bool(0) → wynik False

## WNIOSKI

Python automatycznie rozpoznaje typ danych na podstawie wartości. Funkcje konwersji umożliwiają łatwą zmianę typów. Działanie operatorów jest intuicyjne i spójne z zasadami matematyki.

# Zadanie 2

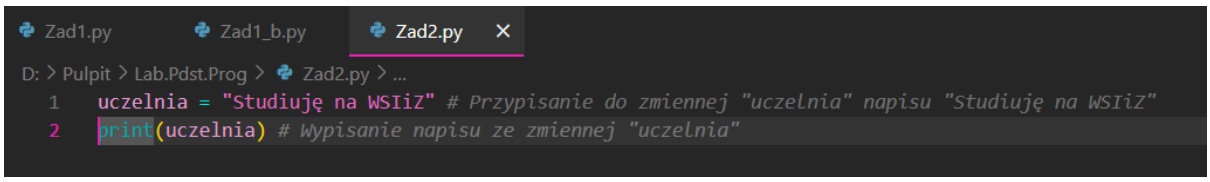
## CEL ZADANIA

Celem zadania było zapoznanie się z funkcją print() i sposobem wyświetlania tekstu w Pythonie.

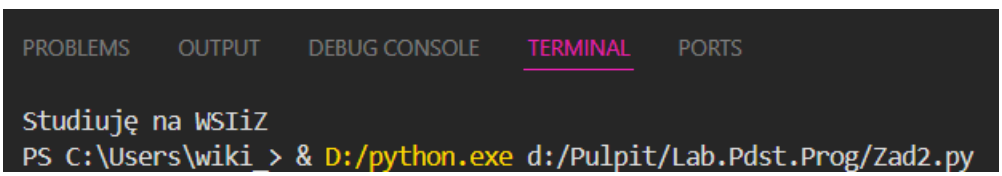
## PRZEBIEG ZADANIA

Utworzono zmienną uczelnia o wartości "Studiuję na WSIiZ", a następnie za pomocą print(uczelnia) wypisano jej zawartość w konsoli.

## KOD PROGRAMU




## WYNIK PROGRAMU



## ANALIZA WYNIKÓW

Program poprawnie wyświetlił tekst:
“ Studiuję na WSIiZ”
Nie wystąpiły żadne błędy składniowe ani logiczne.

## WNIOSKI

Instrukcja “print()” służy do prezentacji wyników programu i jest podstawową funkcją wyjścia w Pythonie.

# Zadanie 3

## CEL ZADANIA

Celem zadania było wykorzystanie zmiennych i funkcji print() do tworzenia dynamicznego tekstu.

## PRZEBIEG ZADANIA

Zdefiniowano zmienne:

imie = 'Jan'

wiek = 20

wzrost = 178

Następnie wydrukowano zdania przy pomocy print():

Nazywam się Jan i mam 20 lat.

Mój wzrost to 178 cm.

## KOD PROGRAMU

Zad1.py Zad1_b.py Zad2.py Zad3.py

D: > Pulpit > Lab.Pdst.Prog > Zad3.py > ...

```python
imie = 'Jan' # Przypisanie do zmiennej "imie" tekst "Jan"
wiek = 20 # Przypisanie do zmiennej "wiek" liczbe 20
wzrost = 178 # Przypisanie do zmiennej "wzrost" liczby 178

print("Nazywam się", imie, "i mam", wiek, "lat.\n Mój wzrost to", wzrost, "cm.") # Wypisanie tekstu z użyciem powyższych zmiennych oraz wykorzystanie "\n" (nowa linia)
```

WYNIK PROGRAMU

```
Nazywam się Jan i mam 20 lat.
 Mój wzrost to 178 cm.
PS C:\Users\wiki_>
```

ANALIZA WYNIKÓW

Program poprawnie połączył tekst ze zmiennymi liczbowymi. Wynik był zgodny z oczekiwaniami.

WNIOSKI

Python umożliwia łatwe łączenie tekstu i wartości zmiennych w jednym komunikacie. Umiejętność formatowania wyjścia jest niezbędna przy tworzeniu przejrzystych programów.

Zadanie 4

CEL ZADANIA

Celem zadania było zastosowanie działań arytmetycznych i formatowania tekstu do obliczenia ceny po rabacie.

PRZEBIEG ZADANIA

Zdefiniowano zmienne:

cena = 39.99

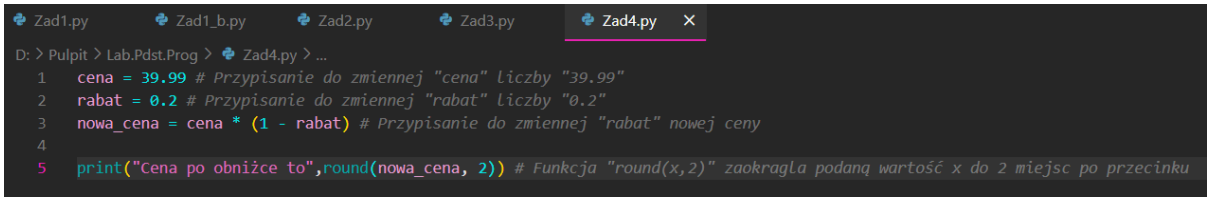
rabat = 0.2

Obliczono cenę po rabacie:

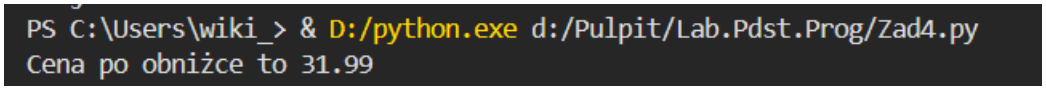
nowa_cena  = cena * (1 - rabat)

print("Cena po obniżce to”, round(nowa_cena, 2))

KOD PROGRAMU




WYNIK PROGRAMU




ANALIZA WYNIKÓW

Program obliczył i wyświetlił wynik 31.99. Formatowanie “round(x,2)”  ograniczyło wynik do dwóch miejsc po przecinku.

WNIOSKI

Zadanie 5

CEL ZADANIA

Celem zadania było stworzenie prostego programu obliczającego pole i obwód prostokąta na podstawie danych od użytkownika.

PRZEBIEG ZADANIA

Użytkownik wprowadzał długości boków:

a = float(input("Podaj długość boku a: "))

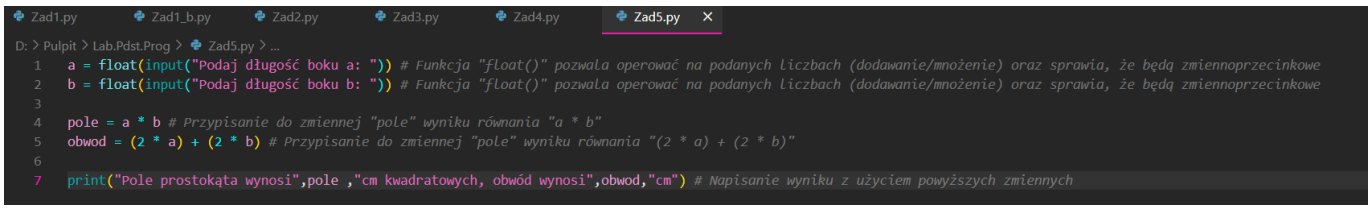
b = float(input("Podaj długość boku b: "))

pole = a * b

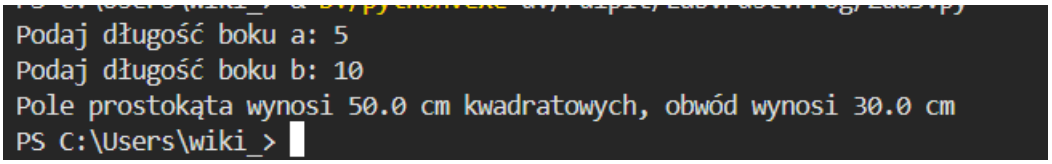
obwod = 2 * (a + b)

print("Pole prostokąta wynosi”,  pole, “cm kwadratowych, obwód wynosi”, obwod, “cm”)

KOD PROGRAMU




WYNIK PROGRAMU




ANALIZA WYNIKÓW

Program poprawnie obliczał wartości i zwracał wynik zgodny z danymi wejściowymi.

WNIOSKI

Instrukcja input() umożliwia interakcję z użytkownikiem. Prosty program pokazuje praktyczne zastosowanie podstaw arytmetyki i formatowania danych w Pythonie.

Zadanie 6

CEL ZADANIA

Celem zadania było stworzenie programu obliczającego spalanie i koszt podróży na podstawie danych użytkownika.

PRZEBIEG ZADANIA

Użytkownik podawał długość drogi, średnie spalanie i cenę paliwa. Program obliczał:

zuzycie = (srednie_spalanie / 100) * droga

koszt_podrozy = zuzycie* cena_paliwa

Wersja rozszerzona używała “random.randint()” do losowania długości drogi i “f-stringów” do formatowania wyników.

KOD PROGRAMU

```python
droga = float(input("Podaj pokonany dystans: ")) # Funkcja "float()" pozwala operować na podanych liczbach (dodawanie/mnożenie/dzielenie) oraz sprawia, że będą zmiennoprz
srednie_spalanie = float(input("Podaj średnie spalanie samochodu (w litrach na 100km): ")) # Funkcja "float()" pozwala operować na podanych liczbach (dodawanie/mnożenie/d
cena_paliwa = 6.5 # Cena paliwa za litr
zuzycie = (srednie_spalanie / 100) * droga # Obliczanie zużycia paliwa
koszt_podrozy = zuzycie * cena_paliwa # Obliczenie kosztu podróży

print("Przewidywane zużycie paliwa wynosi:",zuzycie, "litrów","\nSzacunkowy koszt podróży wynosi:",koszt_podrozy, "zł") # Wypisanie wyników z użyciem zmiennych
```

a)

```python
import random # Importowanie biblioteki "math" w celu użycia funkcji random.randint()

droga = random.randint(1, 10000) # Używamy funkcji random.randint() aby wylosować liczbę spośród podanego zakresu
srednie_spalanie = float(input("Podaj średnie spalanie samochodu (l/100km): ")) # Funkcja "float()" pozwala operować na podanych liczbach (dodawanie/mnożenie/dzielenie) o
cena_paliwa = float(input("Podaj cenę paliwa (zł/l): ")) # Funkcja "input()" pozwoli użytkownikowi wprowadzić cenę paliwa do zmiennej
zuzycie = round(srednie_spalanie / 100 * droga, 2) # Obliczanie zużycia paliwa, funkcja "round()" zaokrągli wynik do 2 miejsc po przecinku
koszt_podrozy = round(zuzycie * cena_paliwa, 2) # Obliczenie kosztu podróży, funkcja "round()" zaokrągli wynik do 2 miejsc po przecinku

print("Przewidywane zużycie paliwa na", droga, "km wynosi:",zuzycie, "litrów","\nSzacunkowy koszt podróży na", droga, "km wynosi:",koszt_podrozy, "zł") # Wypisanie wynikó
```

b)

```python
import random # Importowanie biblioteki "math" w celu użycia funkcji random.randint()

droga = random.randint(1, 10000) # Używamy funkcji random.randint() aby wylosować liczbę spośród podanego zakresu
srednie_spalanie = float(input("Podaj średnie spalanie samochodu (l/100km): ")) # Funkcja "float()" pozwala operować na podanych liczbach (dodawanie/mnożenie/dzielenie) o
cena_paliwa = float(input("Podaj cenę paliwa (zł/l): ")) # Funkcja "input()" pozwoli użytkownikowi wprowadzić cenę paliwa do zmiennej
zuzycie = round(srednie_spalanie / 100 * droga, 2) # Obliczanie zużycia paliwa, funkcja "round()" zaokrągli wynik do 2 miejsc po przecinku
koszt_podrozy = round(zuzycie * cena_paliwa, 2) # Obliczenie kosztu podróży, funkcja "round()" zaokrągli wynik do 2 miejsc po przecinku

print(f'Przewidywane zużycie paliwa wynosi: {zuzycie} litrów, a szacunkowy koszt podróży wynosi: {koszt_podrozy} zł') # Wypisanie wyniku z użyciem zmiennych oraz użycie f
```

WYNIK PROGRAMU

```
PS C:\Users\wiki_> & D:/python.exe d:/Pulpit/Lab.Pdst.Prog/Zad6.py
Podaj pokonany dystans: 30
Podaj średnie spalanie samochodu (w litrach na 100km): 7
Przewidywane zużycie paliwa wynosi: 2.1 litrów
Szacunkowy koszt podróży wynosi: 13.65 zł
PS C:\Users\wiki_>
```

a)

```
Podaj średnie spalanie samochodu (l/100km): 7
Podaj cenę paliwa (zł/l): 5.58
Przewidywane zużycie paliwa na 8139 km wynosi: 569.73 litrów
Szacunkowy koszt podróży na 8139 km wynosi: 3179.09 zł
PS C:\Users\wiki_>
```

b)

```
PS C:\Users\wiki_> & D:/python.exe c:/Users/wiki_/Downloads/zadanie_6b.py
Podaj średnie spalanie samochodu (l/100km): 11
Podaj cenę paliwa (zł/l): 2.69
Przewidywane zużycie paliwa wynosi: 673.31 litrów, a szacunkowy koszt podróży wynosi: 1811.2 zł
PS C:\Users\wiki_>
```

## ANALIZA WYNIKÓW

Program poprawnie obliczał zużycie paliwa i koszty podróży. Losowanie działało prawidłowo, a formatowanie zapewniało czytelność wyników.

## WNIOSKI

Ćwiczenie pokazało zastosowanie biblioteki random, obliczeń matematycznych i dynamicznego wyświetlania wyników przy użyciu f-stringów

## Zadanie 7

## CEL ZADANIA

Celem zadania było napisanie programu rozwiązującego równanie liniowe ax + b = 0 oraz narysowanie schematu blokowego algorytmu.

## PRZEBIEG ZADANIA

Użytkownik wprowadzał współczynniki a i b. Program obliczał:

if a == 0:

  if b == 0

    print(“Równanie tożsamościowe”)

  else:

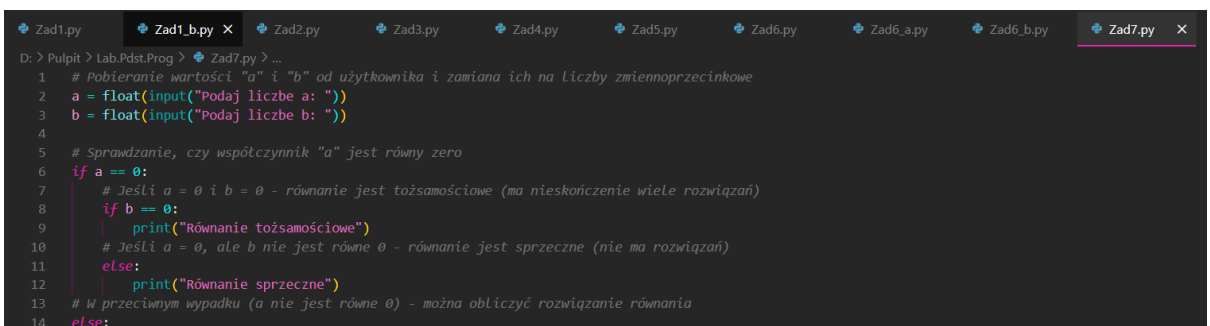
    print("Równanie sprzeczne")

else:

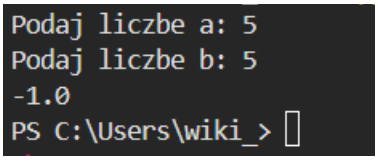
    X = (b * -1)/a

Schemat blokowy wykonano w draw.io zgodnie z logiką warunkową.

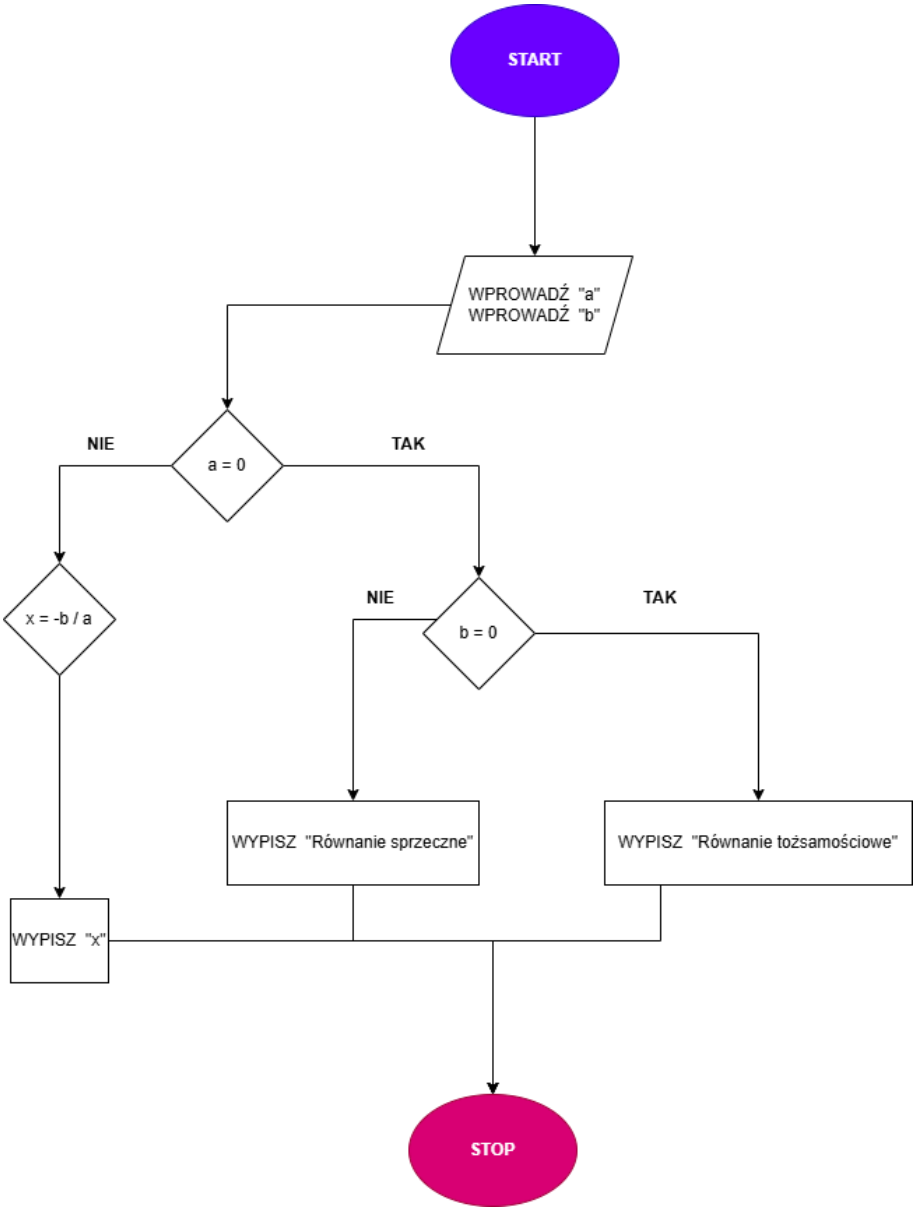
## KOD PROGRAMU



WYNIK PROGRAMU




SHCEMAT BLOKOWY




ANALIZA WYNIKÓW

Dla poprawnych danych program zwracał jedno rozwiązanie. Przy a = 0 wyświetlał odpowiedni komunikat.

WNIOSKI

Zadanie pozwoliło zrozumieć strukturę warunkową if-else i sposób reprezentacji algorytmu na schemacie blokowym.

Zadanie 8

## CEL ZADANIA

Celem zadania było napisanie programu rozwiązującego równanie kwadratowe $ax^2 + bx + c = 0$ oraz opracowanie jego schematu blokowego.

## PRZEBIEG ZADANIA

Program pobierał wartości a, b, c, następnie obliczał deltę i pierwiastki.

Schemat blokowy wykonano w draw.io zgodnie z logiką warunkową.

## KOD PROGRAMU

Zad1.py | Zad1_b.py | Zad2.py | Zad3.py | Zad4.py | Zad5.py | Zad6.py | Zad6_a.py | Zad6_b.py | Zad7.py | Zad8.py

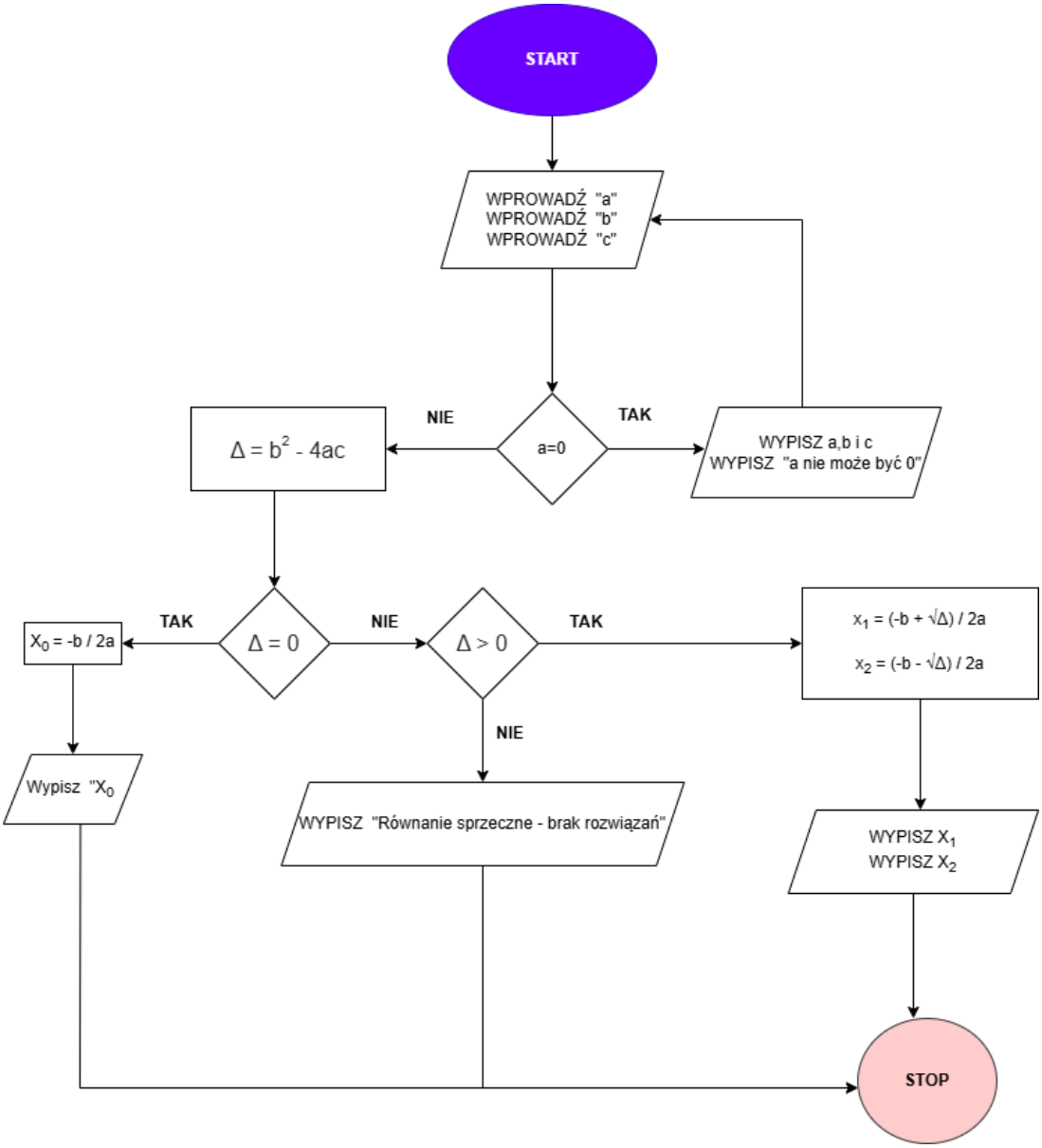
D: > Pulpit > Lab.Pdst.Prog > Zad8.py > ...

```python
from math import sqrt # Importujemy funkcje "sqrt" (pierwiastek kwadratowy) z modułu "math"
# Pobranie współczynników równania kwadratowego od użytkownika
a = float(input("Podaj a: "))
b = float(input("Podaj b: "))
c = float(input("Podaj c: "))

# Sprawdzenie czy a = 0 (bo wtedy równanie nie jest kwadratowe)
if a == 0:
    print("a nie może być równe 0")
else:
    delta = (b ** 2) - (4 * a * c) # Obliczanie delty
    # Jeśli delta = 0 - równanie ma jedno rozwiązanie
    if delta == 0:
        x_zero = (b * -1) / (2 * a)
        print(f"X0 = {x_zero}")
    else:
        # Jeśli delta > 0 - równanie ma dwa rozwiązania
        if delta > 0:
            pierwiastek_z_delty = sqrt(delta)
            x_jeden = round(((b * -1) + pierwiastek_z_delty) / (2 * a),2) # Obliczanie pierwszego pierwiastka równiania
            x_dwa = round(((b * -1) - pierwiastek_z_delty) / (2 * a),2) # Obliczanie drugiego pierwiastka równania
            print(f"x1 = {x_jeden} x2 = {x_dwa}") # Wypisanie wyników
        # Jeśli delta < 0 - brak rozwiązań
        else:
            print("Równanie sprzeczne")

```

## WYNIK PROGRAMU

```
PS C:\Users\wiki_> & D:/python.exe c:/Users/wiki_/Downloads/zadanie_8.py
Podaj a: 7
Podaj a: 7
Podaj b: 5
Podaj c: 2
Równanie sprzeczne
PS C:\Users\wiki_> & D:/python.exe c:/Users/wiki_/Downloads/zadanie_8.py
Podaj a: 16
Podaj b: 8
Podaj c: 1
X0 = -0.25
PS C:\Users\wiki_>
```

SCHEMAT BLOKOWY




ANALIZA WYNIKÓW

Program poprawnie rozpoznawał liczbę rozwiązań i obliczał je z użyciem wzoru kwadratowego. Schemat blokowy był zgodny z logiką programu.

WNIOSKI

Zadanie utrwaliło wiedzę o instrukcjach warunkowych i działaniach matematycznych w Pythonie. Pozwoliło zrozumieć zastosowanie algorytmu w praktyce.

## Zadanie 9

### CEL ZADANIA

Celem zadania było stworzenie prostego kalkulatora wykonującego podstawowe działania matematyczne.

### PRZEBIEG ZADANIA

Użytkownik podawał dwie liczby. Program wykonywał i wyświetlał:

dodawanie = a + b

odejmowanie = a – b

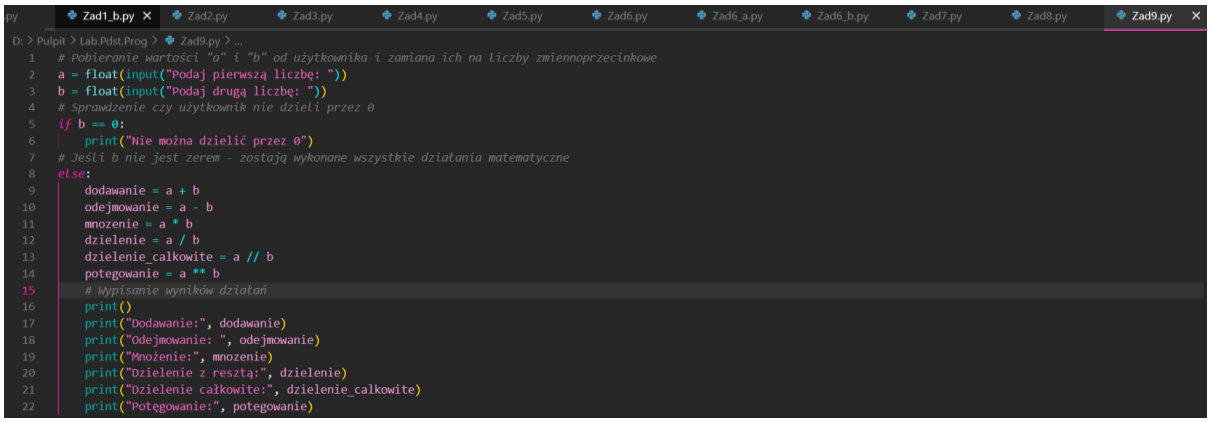
mnozenie = a * b

dzielenie = a / b

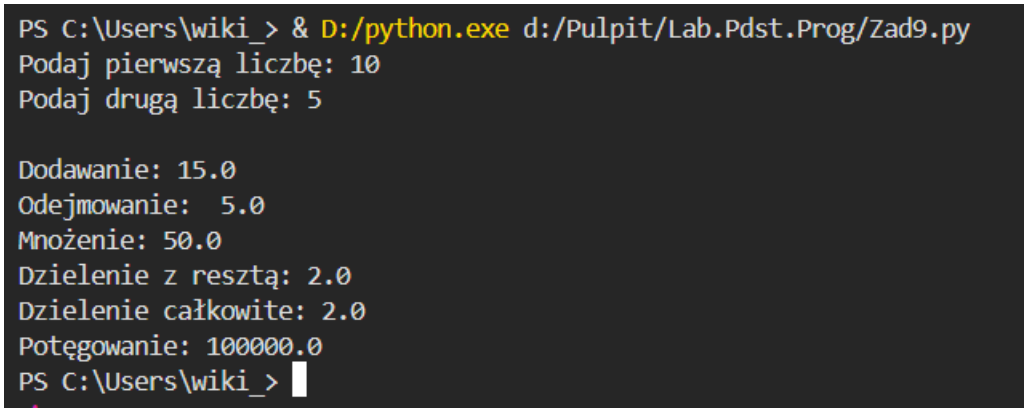
dzielenie_calkowite = a // b

potegowanie = a ** b

### KOD PROGRAMU




### WYNIK PROGRAMU




ANALIZA WYNIKÓW

Program poprawnie wykonywał obliczenia dla wszystkich operatorów. Przy dzieleniu przez 0 zwracał błąd, co pozwoliło zrozumieć znaczenie obsługi wyjątków.

WNIOSKI

Kalkulator jest praktycznym przykładem wykorzystania operatorów arytmetycznych. Pokazuje znaczenie walidacji danych i obsługi błędów.