

Programowanie 1R – Ćwiczenia 1

Podstawy Pythona

Mikołaj Marcinkowski

24 lutego 2026

Zadania możesz wykonywać jako oddzielne skrypty `.py` lub w notatniku jupytera `.ipynb`, chyba że treść zadania wskazuje inaczej.

Zadania wzorowane na Serii 1 zadań autorstwa Bartłomieja Zglinickiego (<https://www.fuw.edu.pl/~bzglinicki/teaching/plr/zadania/seria-1/>), udostępnionej na licencji CC BY 4.0.

Zadanie 1. Prostokąt

Napisz program, który prosi użytkownika o podanie dwóch boków prostokąta a i b , po czym zwraca pole powierzchni tego prostokąta.

Zadanie 2. Kula

Pole powierzchni P i objętość V kuli o promieniu r dane są wzorami:

$$P = 4\pi r^2,$$

$$V = \frac{4}{3}\pi r^3.$$

Napisz program, który prosi użytkownika o podanie promienia r kuli i zwraca jej pole powierzchni P i objętość V .

Zadanie 3. BMI

Wskaźnik masy ciała (Body Mass Index, BMI) to wielkość stosowana do sprawdzania poprawności masy ciała. Definiuje go wzór:

$$\text{BMI} = \frac{\text{masa ciała w kilogramach}}{(\text{wzrost w metrach})^2}.$$

Interpretuje się go w następujący sposób:

BMI	interpretacja
poniżej 18,5	niedowaga
18,5 – 25	waga prawidłowa
25 – 30	nadwaga
ponad 30	otyłość

Napisz program, który prosi użytkownika o podanie masy ciała w kilogramach i wzrostu w metrach, po czym oblicza i zwraca współczynnik BMI i interpretację.

Zadanie 4. Równanie kwadratowe

Napisz program, który prosi użytkownika o podanie współczynników $a \neq 0$, b i c równania kwadratowego:

$$ax^2 + bx + c = 0,$$

po czym zwraca rozwiązania tego równania (w tym, gdy są zespolone).

Zadanie 5. Silnia

Silnię liczby naturalnej n definiuje się wzorem:

$$n! = n \cdot (n - 1) \cdot (n - 2) \cdot \dots \cdot 2 \cdot 1.$$

Dodatkowo przyjmuje się, że $0! = 1$.

Napisz funkcję `rsilnia`, która przyjmuje liczbę naturalną n i zwraca jej silnię obliczoną w sposób rekurencyjny.

Napisz funkcję `isilnia`, która przyjmuje liczbę naturalną n i zwraca jej silnię obliczoną w sposób iteracyjny, czyli za pomocą pętli.

Uwaga: Python nie pozwala wyświetlać liczb całkowitych mających więcej, niż 4300 cyfr. Limit ten można wyłączyć wykonując funkcję `sys.set_int_max_str_digits(0)`, która wymaga zaimportowania biblioteki `sys`.

Zadanie 6. Szeregi

Funkcje $\exp(x)$ i $\cos(x)$ można rozwinąć w szeregi postaci:

$$\exp(x) = \sum_{k=0}^{\infty} \frac{x^k}{k!},$$

$$\cos(x) = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k}}{(2k)!}.$$

Napisz funkcję `szereg_exp`, która przyjmuje liczbę rzeczywistą x i liczbę naturalną n i oblicza przybliżenie funkcji `exp`:

$$\text{szereg_exp}(x, n) = \sum_{k=0}^n \frac{x^k}{k!}.$$

Podobnie napisz funkcję `szereg_cos`, która przyjmuje liczbę rzeczywistą x i liczbę naturalną n i oblicza przybliżenie funkcji `cos`:

$$\text{szereg_cos}(x, n) = \sum_{k=0}^n (-1)^k \frac{x^{2k}}{(2k)!}.$$

Do obliczania silni możesz wykorzystać funkcje z zadania 5 lub funkcję `math.factorial` z biblioteki `math`. Dokładność przybliżeń dla różnych n możesz sprawdzić na przykład porównując ich wyniki z wartościami funkcji `math.exp` i `math.cos`.

Zadanie 7. Przybliżenie liczby π

Liczbę π można obliczyć na przykład ze wzoru Machin’a:

$$\frac{\pi}{4} = 4 \cdot \arctan\left(\frac{1}{5}\right) - \arctan\left(\frac{1}{239}\right).$$

Dla $|x| \leq 1$ funkcja $\arctan(x)$ rozwija się w szereg:

$$\arctan(x) = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{2k+1} = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \dots$$

Korzystając z powyższego rozwinięcia, napisz program, który przyjmuje liczbę naturalną n i zwraca oszacowanie liczby π , z rozwinięciem obciętych do n -tego wyrazu.

Zadanie 8. Ciąg Fibonacciego

Ciąg Fibonacciego definiuje się wzorem

$$F_n = \begin{cases} 1 & \text{dla } n = 1 \text{ lub } n = 2, \\ F_{n-1} + F_{n-2} & \text{dla } n > 2. \end{cases}$$

Kolejne wyrazy ciągu Fibonacciego to

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, \dots$$

Napisz funkcję `rfib`, która przyjmuje liczbę naturalną n i zwraca n -ty wyraz ciągu Fibonacciego, obliczony algorytmem rekurencyjnym.

Napisz funkcję `ifib`, która przyjmuje liczbę naturalną n i zwraca n -ty wyraz ciągu Fibonacciego, obliczony algorytmem iteracyjnym.

Porównaj czasy wykonania `ifib(n)` i `rfib(n)` dla różnych wartości n .