

Programowanie 1R – Ćwiczenia 7

Klasy (metody specjalne/magiczne)

Mikołaj Marcinkowski

21 kwietnia 2026

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

Zadania 2 i 3 wzorowanie na Serii 7 zadań autorstwa Agnieszki Makulskiej (https://github.com/AgnieszkaMakulska/Programowanie_IR_2026/blob/main/%C4%87w_7/PIR_seria_7.pdf)

Zadania 1 i 4 wzorowane na Serii 7 zadań autorstwa Bartłomieja Zglinickiego (<https://www.fuw.edu.pl/~bzglinicki/teaching/plr/zadania/seria-7/>), udostępnionej na licencji CC BY 4.0.

Zadanie 1. Relatywistyczne prędkości

Niech $\beta = v/c$ będzie prędkością wyrażoną w jednostkach prędkości światła c . Zgodnie z relatywistycznym prawem składania prędkości, w przypadku jednowymiarowym, jeśli dla nas ciało 1 porusza się z prędkością β_1 , a w układzie odniesienia ciała 1 ciało 2 porusza się z prędkością β_2 , to my zaobserwujemy je poruszające się z prędkością:

$$\beta = \frac{\beta_1 + \beta_2}{1 + \beta_1 \beta_2}. \quad (1)$$

Napisz klasę `RelatPredkosc` reprezentującą prędkość β . Klasa ta powinna zawierać:

- konstruktor, który przyjmuje wartość β ,
- metodę `gamma`, która zwraca wartość czynnika Lorentza γ :

$$\gamma = \frac{1}{\sqrt{1 - \beta^2}} \quad (2)$$

- operator dodawania `+` (metoda `__add__`), który działa zgodnie z prawem składania prędkości (1),

- operator `+=` (metoda `__iadd__`), analogicznie do powyższego,
- metodę `__str__`, formatującą opis obiektu klasy wyświetlany, gdy go `printujemy`.

Przetestuj działanie tej klasy, tworząc i dodając do siebie prędkości.

Zadanie 2. Wektor 2D

Napisz klasę `Vector2D` reprezentującą wektor w dwuwymiarowej przestrzeni rzeczywistej. Klasa ta powinna zawierać:

- konstruktor przyjmujący współrzędne x i y wektora,
- operator dodawania `+`, który przyjmuje inny obiekt klasy `Vector2D` i zwraca wektor będący sumą obu wektorów,
- operator mnożenia `*` (metoda `__mul__`), który powinien obsługiwać dwa przypadki (możesz wykorzystać funkcję `isinstance()` do sprawdzenia typu drugiego argumentu):
 - jeśli drugim argumentem jest liczba, pomnóż wektor przez nią,
 - jeśli drugim argumentem jest obiekt klasy `Vector2D`, zwróć iloczyn skalarny między nimi.
- metodę `__rmul__`, obsługującą mnożenie przez liczbę z lewej,
- metodę `__str__`, formatującą opis obiektu klasy, np. $[x, y]$.

Zadanie 3. Masa na sprężynie

Napisz klasę `MasaPunktowa` reprezentującą punkt materialny posiadający masę. Klasa ta powinna zawierać:

- pole `masa`, przechowujące masę punktu,
- pola `polozenie` i `predkosc`, przechowujące aktualne położenie $\vec{r}$ i prędkość punktu $\vec{v}$ jako wektory `Vector2D` z zadania 2.
- konstruktor przyjmujący masę, początkowe położenie i prędkość,
- pole `trajektoria`, początkowo pustą listę, do której dodawane będą kolejne położenia punktu,

- metodę **krok**, symulującą przemieszczenie punktu pod wpływem czasu. Metoda ta powinna przyjmować dwa argumenty: wektor **F** klasy **Vector2D** – siła działająca w tej chwili na ciało, oraz długość kroku czasowego **dt**. Metoda powinna wykonywać następujące operacje, używając operatorów zdefiniowanych w klasie **Vector2D**:

1. dodać obecne położenie do listy **trajektoria**,
2. zaktualizować prędkość metodą Eulera: $\vec{v} = \vec{v} + (\vec{F}/m) \cdot dt$,
3. zaktualizować położenie metodą Eulera: $\vec{r} = \vec{r} + \vec{v} \cdot dt$,

Wykorzystując obiekt klasy **MasaPunktowa** zasymuluj ciało o masie $m = 1$ zaczepione na końcu sprężyny działającej na nie z siłą $\vec{F} = -k\vec{r}$, gdzie $k = 10$. Narysuj wykresy $x(t)$ i $y(t)$, zaczynając od wychylenia z położenia równowagi.

Zadanie 4. Liczby wymierne

Napisz klasę **LiczbaWymierna** reprezentującą liczbę wymierną. Klasa powinna przechowywać liczby całkowite p i q , których iloraz $\frac{p}{q}$ jest równy reprezentowanej liczbie wymiernej, przy czym p i q powinny być względnie pierwsze oraz $q > 0$. Klasa ta powinna zawierać:

- konstruktor, który można wywołać bez argumentów lub z jednym albo dwoma argumentami całkowitymi; w pierwszym przypadku tworzony obiekt powinien być inicjalizowany wartością 0, w drugim – zadaną liczbą całkowitą, zaś w trzecim – ilorazem argumentów konstruktora; konstruktor wywołany z dwoma argumentami powinien działać poprawnie również wtedy, gdy wartości jego argumentów nie są względnie pierwsze lub gdy drugi argument jest ujemny,
- metody **licznik** i **mianownik**, zwracające odpowiednio licznik p i mianownik q ,
- operator konwersji do typu **float** (metoda **__float__**),
- operator zmiany znaku – (metoda **__neg__**),
- operator porównania **<** (metoda **__lt__**),
- operatory dodawania **+**, odejmowania **-** (metoda **__sub__**), mnożenia ***** i dzielenia **/** (metoda **__truediv__**),
- operatory **+=**, **-=**, ***=**, **/=**,
- metodę **__str__**, formatującą opis obiektu klasy jako p/q .

Przetestuj działanie obiektów tej klasy.