

Plan 5-dniowy (intensywny): OOP w Pythonie + Arcade — wersja PL

Teoria OOP po polsku • praktyka w Arcade po angielsku (z tłumaczeniem przeglądarki) • ok. 6-8 h dziennie

Zasada: teoria OOP w małych dawkach rano — po polsku. Resztę dnia kodowanie w Arcade — tutoriale są po angielsku, ale tekstowe strony świetnie działają z tłumaczeniem przeglądarki (Chrome/Edge: prawy przycisk → „Przetłumacz na polski”; kod zostaje nietknięty), a filmy na YouTube mają automatyczne polskie napisy. Przed startem: `pip install arcade` (Python 3.9+).

Dzień 1 — Podstawy OOP (PL) + pierwsze okno

Rano (2-3 h): teoria po polsku

- Kacper Sieradziński — „Kurs programowania obiektowego: Podstawy, Metody specjalne, Dziedziczenie w Pythonie” (YouTube, PL): obejrzeć część o podstawach — klasy, obiekty, atrybuty, metody, konstruktor, self. Oglądać z pauzami i przepisywać kod.
<https://www.youtube.com/watch?v=XYmt6foUCFI>
- Kodologia — „Python. Programowanie obiektowe” (PL, interaktywny, kodowanie w przeglądarce, 22 automatycznie sprawdzane zadania): zrobić pierwsze rozdziały jako ćwiczenia.
<https://kodologia.pl/kursy/python-programowanie-obiektowe>

Po południu (4-5 h): praktyka w Arcade

- Real Python — „Arcade: A Primer on the Python Game Framework” (ang., czytać z tłumaczeniem przeglądarki): okno, rysowanie, sprite'y, pętla gry, klawiatura. Przejść całość, potem zmodyfikować grę po swojemu.
<https://realpython.com/arcade-python-game-framework/>
- Alternatywnie/pomocniczo: polska (maszynowo tłumaczona) wersja tego primera — jakość nierówna, ale bywa pomocna:
<https://pl.python-3.com/?p=333>

Dzień 2 — Dziedziczenie (PL) + sprite'y, kolizje, zdarzenia

Rano (1,5-2 h): teoria po polsku

- Sieradziński — część o dziedziczeniu i metodach specjalnych (dunder methods) z tego samego filmu; do tego odpowiednie rozdziały Kodologii (dziedziczenie) jako zadania.

Po południu (5-6 h): praktyka w Arcade

- Wybrane rozdziały kursu autora biblioteki Arcade Academy — Learn Python (ang., dobrze działa z tłumaczeniem przeglądarki): sprite'y, kolizje, poruszanie, dźwięki. Rozdziały wstępne (zmienne, pętle) pominąć.
<https://learn.arcade.academy/>
- Ćwiczenie na koniec dnia: mini-gra „zbieraj monety, unikaj wrogów” z klasą bazową Enemy i dwoma typami przeciwników dziedziczącymi po niej.

Dzień 3 — Platformówka, część 1

- Oficjalny tutorial platformówki (dokumentacja Arcade, ang. + tłumaczenie przeglądarki; zgodny z aktualnym API): gracz, fizyka, grawitacja, skoki, kamera, mapa pozioma. Pierwsza połowa kroków.
https://api.arcade.academy/en/latest/tutorials/platform_tutorial/index.html
- Wideo-wsparcie: PyCon 2019, Paul Craven — „Build Your Own 2D Platformer Game” (YouTube; włączyć automatyczne polskie napisy: ikona napisów → ustawienia → Przetłumacz automatycznie → polski).

<https://www.youtube.com/watch?v=Djtm1DzWSvo>

- Darmowe assety (grafiki, dźwięki): kenney.nl/assets

Dzień 4 — Platformówka, część 2 + szlify OOP (PL)

- Dokończyć tutorial platformówki: wrogowie, punkty, wiele poziomów, ekran końca gry.
- Teoria PL (30–45 min): rozdziały Kodologii o hermetyzacji i property (dekorator @property, getter/setter) — i od razu zastosować w platformówce, np. property „health” z ograniczeniem 0–100.
- Alternatywne ujęcie platformówki (ang., tekst): Real Python — „Build a Platform Game in Python With Arcade” — pisany pod starsze API; przy rozbieżnościach wygrywa oficjalna dokumentacja.

<https://realpython.com/platformer-python-arcade/>

Dzień 5 — Własny projekt

- Cały dzień na grę własnego pomysłu od zera: remake Tetrisa / Snake'a / Flappy Birda albo coś autorskiego. Wymóg projektowy: minimum 3 własne klasy, w tym jedna hierarchia dziedziczenia — tu OOP naprawdę „klika”.
- Kopalnia przykładów (kod źródłowy dziesiątek mini-gier w Arcade): https://api.arcade.academy/en/latest/example_code/index.html
- Na koniec: wrzucić projekt na GitHuba — dobre domknięcie tygodnia.

Uwagi

- *Cała teoria OOP jest po polsku (Sieradziński + Kodologia). Po angielsku zostają tylko tutoriale samej biblioteki Arcade — polskiego kursu Arcade po prostu nie ma; tłumaczenie przeglądarki + automatyczne napisy YouTube w pełni to nadrabiają.*
- *Płatna alternatywa PL dla teorii, gdyby syn wolał zwarty kurs wideo: „Programowanie obiektowe w Python” (strefakursow.pl) albo kurs Karola Kurka „50 zadań praktycznych z programowania obiektowego” (Videopoint/Helion).*
- *Arcade 3.x zmieniło część API względem starszych tutoriali (np. `arcade.start_render()` → `self.clear()`). Przy rozbieżnościach zawsze wygrywa aktualna dokumentacja: api.arcade.academy.*
- *Jeśli tempo okaże się za szybkie, dzień 5 można poświęcić na dokończenie platformówki, a własny projekt przenieść na później.*