

Wymagania edukacyjne

Podstawy programowania (Python + Tkinter)

Zawód: technik programista (technikum 5-letnie)

Kwalifikacja: INF.04. Projektowanie, programowanie i testowanie aplikacji

Przedmioty: Podstawy programowania

Klasa: II

Źródła: *Program nauczania wydawnictwa Helion — technik programista* (dział 3.2. Podstawy programowania, dział 3.8. Pracownia podstaw programowania), *Podstawa programowa kształcenia w zawodach* (Dziennik Ustaw, poz. 991) — efekty kształcenia i kryteria weryfikacji ujęte w programie nauczania jako część jednostki INF.04.3. Projektowanie oprogramowania, dotycząca środowisk programistycznych, frameworków i programowania aplikacji desktopowych.

Zasady konstrukcji wymagań

1. Przedmiot **Podstawy programowania** (teoria) jest oceniany wyłącznie za **wiedzę i rozumienie** — uczeń wie, zna, rozróżnia, wyjaśnia, charakteryzuje, porównuje, analizuje na podstawie opisu, zrzutu ekranu lub zapisu kodu (bez uruchamiania).

2. **Zasada kumulatywności ocen.**

Uzyskanie oceny wyższej wymaga opanowania wiadomości i umiejętności przypisanych do wszystkich ocen niższych. Np. na ocenę **dobrą (4)** uczeń musi dodatkowo spełniać wymagania na oceny **dopuszczającą (2)** i **dostateczną (3)**. Ocena **celująca (6)** oznacza pełne (100%), bezbłędne i w pełni samodzielne opanowanie wymagań przypisanych do ocen 2–5 w danym dziale.

3. **Adaptacja do języka Python i biblioteki Tkinter.**

Program nauczania i podstawa programowa formułują efekty kształcenia niezależnie od konkretnego języka i frameworka (przykłady: C++, C#, Visual Basic, Java, Python; WPF, Qt; XAML). Poniższe wymagania dostosowano do Pythona z biblioteką **Tkinter**:

- *środowisko programistyczne* (np. *Visual Studio*, *CodeBlocks*) → środowiska dla Pythona, np. **PyCharm**, **Visual Studio Code**;
- *framework aplikacji desktopowych* (np. *WPF*, *Qt*) → **Tkinter** — biblioteka wchodząca w skład standardowej biblioteki Pythona, oparta na Tk/Tcl; dla porównania teoretycznego wspomniane są także inne biblioteki GUI dla Pythona (np. *PyQt*, *wxPython*);
- *język do projektowania interfejsu użytkownika* (np. *XAML*) → w Tkinter interfejs definiuje się bezpośrednio w kodzie Pythona, za pomocą menedżerów geometrii **pack**, **grid**, **place**, bez oddzielnego języka znaczników;

- *zdarzenia myszy i klawiatury* → mechanizm wiązania zdarzeń w Tkinter (bind, parametr command widżetu) z funkcją obsługi zdarzenia.
4. Godziny zajęć wg planu nauczania: **Podstawy programowania** — klasa II: 2 godz./tydz. (60 godz.).
 5. **Ocena niedostateczna.**

We wszystkich działach ocenę niedostateczną otrzymuje uczeń, który nie spełnia wymagań na ocenę dopuszczającą — nie posiada wiedzy/umiejętności koniecznych do kontynuowania nauki nawet przy pomocy nauczyciela, nie podejmuje samodzielnej pracy lub oddaje prace niesamodzielne.

KLASA II

Podstawy programowania

Dział 1. Środowiska programistyczne i narzędzia RAD dla aplikacji desktopowych

Zakres: pojęcie IDE i RAD (Rapid Application Development), przykładowe środowiska dla Pythona (PyCharm, Visual Studio Code), kryteria doboru środowiska programistycznego, narzędzia wspomagające tworzenie aplikacji desktopowych.

Ocena	Uczeń wie / rozumie / potrafi wyjaśnić
Dopuszczająca (2)	Wymienia przykłady środowisk IDE dla Pythona (np. PyCharm); wskazuje na podstawie opisu lub zrzutu ekranu, gdzie w środowisku znajduje się edytor kodu i przycisk uruchamiania programu.
Dostateczna (3)	Wyjaśnia pojęcia IDE oraz RAD; charakteryzuje podstawowe funkcje środowiska programistycznego (edycja kodu, uruchamianie, sygnalizowanie błędów); wymienia narzędzia wykorzystywane w procesie tworzenia aplikacji desktopowych.
Dobra (4)	Charakteryzuje różnice funkcjonalne między wybranymi środowiskami dla Pythona (np. wsparcie dla podpowiedzi składni, debugera, podglądu zmiennych); wyjaśnia kryteria doboru środowiska programistycznego do rodzaju zadania.
Bardzo dobra (5)	Uzasadnia wybór konkretnego środowiska i narzędzi wspomagających dla opisanego zadania tworzenia aplikacji okienkowej; analizuje, na podstawie opisu, zalety i ograniczenia wskazanego środowiska.
Celująca (6)	W pełni, bezbłędnie i samodzielnie opanował materiał działu: wyczerpująco charakteryzuje i porównuje dostępne środowiska programistyczne dla Pythona pod względem wszystkich omawianych kryteriów i trafnie uzasadnia swoje oceny.

Dział 2. Biblioteka Tkinter jako framework do tworzenia aplikacji okienkowych

Zakres: pojęcie frameworka/biblioteki graficznej, charakterystyka Tkinter (część biblioteki standardowej Pythona, oparcie na Tk/Tcl), rola pętli zdarzeń (mainloop), przykłady innych bibliotek GUI dla Pythona.

Ocena	Uczeń wie / rozumie / potrafi wyjaśnić
Dopuszczająca (2)	Wskazuje nazwę biblioteki Tkinter jako standardowego narzędzia do tworzenia interfejsu graficznego w Pythonie; rozpoznaje instrukcję importu modułu tkinter w gotowym kodzie.
Dostateczna (3)	Wyjaśnia pojęcie frameworka/biblioteki graficznej; charakteryzuje bibliotekę Tkinter jako element biblioteki standardowej Pythona; wyjaśnia, czym jest pętla zdarzeń (mainloop) w aplikacji okienkowej.
Dobra (4)	Wyjaśnia różnicę między działaniem aplikacji konsolowej a okienkowej (model zdarzeniowy); podaje przykłady zastosowania Tkinter oraz innych bibliotek GUI dla Pythona (np. PyQt, wxPython).
Bardzo dobra (5)	Wyjaśnia architekturę aplikacji Tkinter (obiekt głównego okna, hierarchia kontenerów i widżetów, pętla zdarzeń) i jej wpływ na sposób organizacji kodu; porównuje możliwości Tkinter z innymi wymienionymi bibliotekami GUI.
Celująca (6)	W pełni, bezbłędnie i samodzielnie opanował materiał działu: wyczerpująco wyjaśnia architekturę i zasadę działania Tkinter oraz trafnie porównuje ją z innymi bibliotekami GUI pod względem zastosowań omawianych na zajęciach.

Dział 3. Elementy interfejsu użytkownika i obsługa zdarzeń

Zakres: rodzaje okien i dialogów (modalne, niemodalne), typowe widżety Tkinter (Label, Button, Entry, Frame i in.), menedżery geometrii (pack, grid, place), pojęcie zdarzenia i procedury obsługi zdarzenia, zdarzenia myszy i klawiatury, zasady projektowania menu aplikacji.

Ocena	Uczeń wie / rozumie / potrafi wyjaśnić
Dopuszczająca (2)	Rozróżnia podstawowe elementy interfejsu użytkownika (okno, przycisk, pole tekstowe, etykieta) na podstawie zrzutu ekranu; wskazuje przykład zdarzenia (np. kliknięcie przycisku).
Dostateczna (3)	Charakteryzuje przeznaczenie typowych widżetów Tkinter (Label, Button, Entry, Frame); rozróżnia okno główne, dialog modalny i niemodalny; wyjaśnia pojęcie zdarzenia i procedury obsługi zdarzenia (funkcji wywoływanej po zajściu zdarzenia).

Dobra (4)	Wyjaśnia zasadę działania menedżerów geometrii pack, grid i place oraz różnice między nimi; wyjaśnia sposób powiązania zdarzenia myszy lub klawiatury z funkcją obsługi; charakteryzuje zasady projektowania czytelnego menu aplikacji.
Bardzo dobra (5)	Analizuje, na podstawie opisu lub zrzutu ekranu, poprawność doboru menedżera geometrii i układu widżetów dla wskazanego okna aplikacji; uzasadnia wybór rodzaju okna (modalne/niemodalne) dla opisanego scenariusza; ocenia czytelność i ergonomię opisanego interfejsu.
Celująca (6)	W pełni, bezbłędnie i samodzielnie opanował materiał działu: analizuje najbardziej złożone, wieloelementowe przykłady interfejsu opisane w zadaniu i trafnie ocenia poprawność zastosowanych menedżerów geometrii, doboru widżetów oraz obsługi zdarzeń.