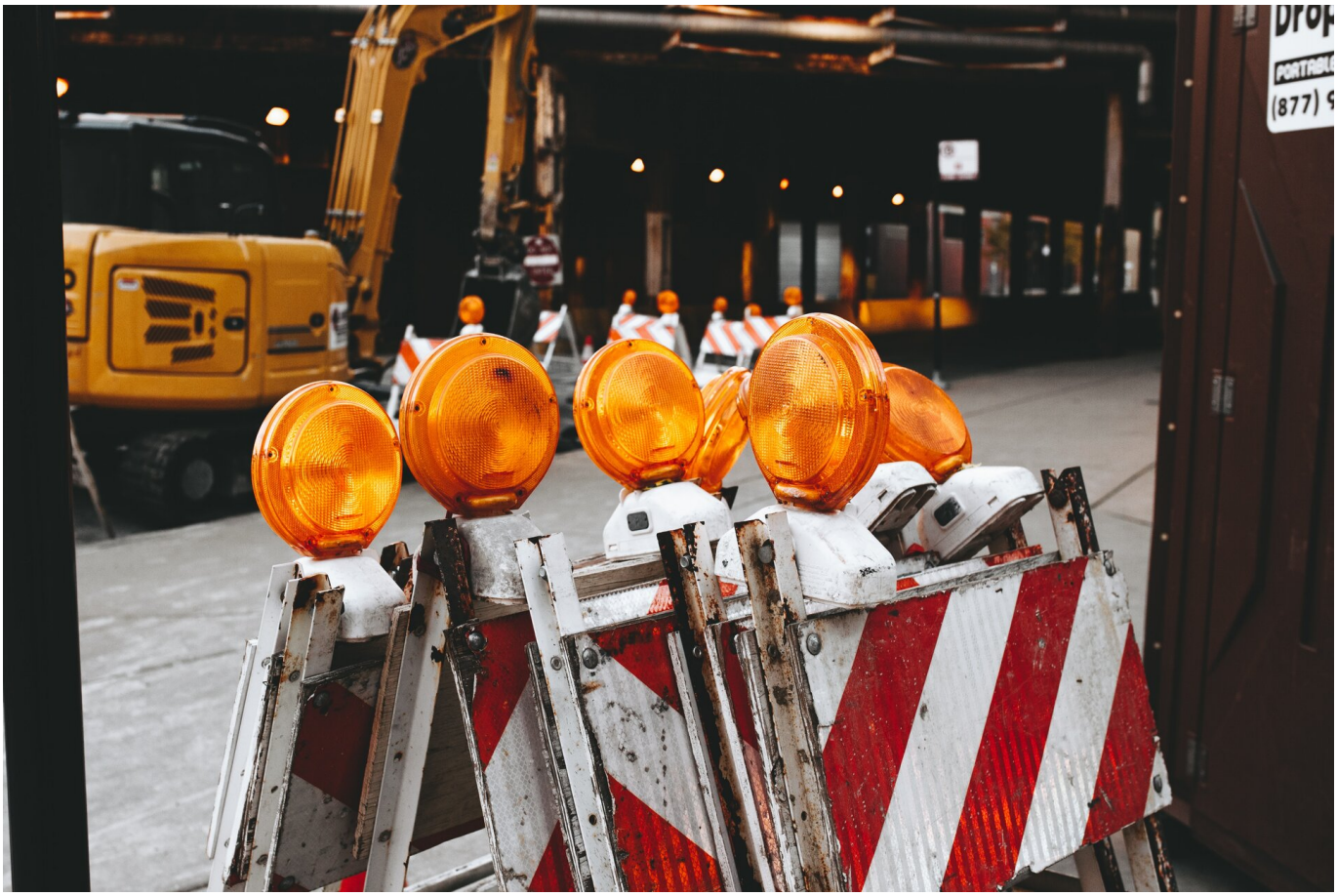
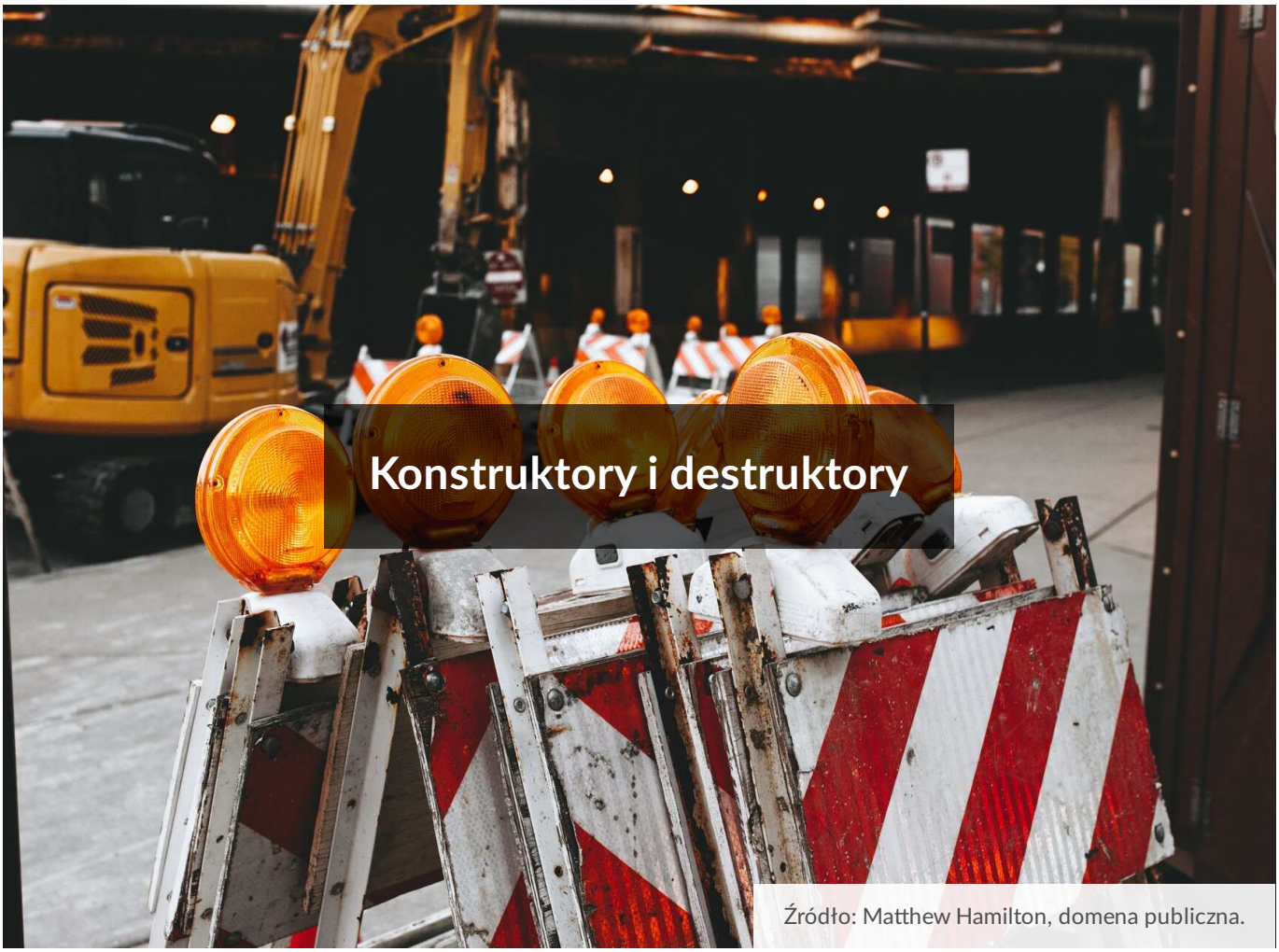


Konstruktory i destruktory



Źródło: Matthew Hamilton, domena publiczna.

W poprzednich e-materiałach poznaliśmy podstawy programowania obiektowego – wiemy już, co to jest klasa, znamy także pojęcie dziedziczenia. W tym e-materiale skupimy się na kolejnych pojęciach: dowiesz się, czym są konstruktory oraz destruktory.

Jeśli chcesz przypomnieć sobie podstawy programowania obiektowego, znajdziesz je w e-materiale:

- Wstęp do programowania obiektowego

Implementacja w poszczególnych językach programowania została przedstawiona w e-materiałach:

- Konstruktory i destruktory w języku C++,
- Konstruktory i destruktory w języku Java,
- Konstruktory i destruktory w języku Python.

Twoje cele

- Wyjaśnisz, czym są konstruktory i destruktory oraz jakie funkcje pełnią w programowaniu obiektowym.
- Wymienisz typy konstruktorów i scharakteryzujesz je.

- Wykonasz kilka ćwiczeń sprawdzających wiedzę z podstaw programowania obiektowego.

Przeczytaj

Konstruktory

Konstruktor to specjalna funkcja (metoda klasy), wywoływana, gdy tworzona jest instancja danej klasy. Wewnątrz konstruktora zdefiniowane są instrukcje, które mają zostać wykonane przy utworzeniu obiektu klasy.

W każdym języku programowania (o ile dany język wspiera programowanie obiektowe) konstruktory deklarowane są na różne sposoby. Najczęściej jednak konstruktor przyjmuje nazwę klasy, w której się znajduje.

Konstruktor zwykły

Przeanalizujmy przykład konstruktora zwykłego.

```
1 klasa Uczen
2     imie: tekst
3     nazwisko: tekst
4     wiek: liczba całkowita
5 
6     funkcja Uczen(imieUcznia, nazwiskoUcznia, wiekUcznia)
7         imie ← imieUcznia
8         nazwisko ← nazwiskoUcznia
9         wiek ← wiekUcznia
```

Zdefiniowana w 6. linii funkcja o nazwie `Uczen()` to analizowany przez nas konstruktor. Oznacza to, że funkcja ta zostanie wywołana podczas tworzenia obiektu klasy `Uczen`. Należy wtedy podać trzy parametry, które nadadzą wartości atrybutom klasy `Uczen`: `imie`, `nazwisko`, `wiek`.

W obrębie danej klasy może być zdefiniowanych wiele konstruktorów. Jeżeli chcemy stworzyć więcej niż jeden konstruktor, to każdy z nich musi różnić się typem, liczbą lub kolejnością parametrów.

W zależności od języka programowania wyróżnia się specjalne typy konstruktorów. Omówimy tutaj niektóre z nich.

Konstruktor domyślny

To rodzaj konstruktora, który **nie ma zdefiniowanych parametrów** (bądź mają one ustawione wartości domyślne). Podczas tworzenia obiektu klasy za pomocą konstruktora domyślnego, nie są podawane żadne argumenty. W przypadku gdy dla danej klasy nie zdefiniowano żadnego konstruktora, zostanie on wygenerowany automatycznie, zgodnie ze specyfikacją danego języka programowania.

Oto przykłady tego rodzaju konstruktora:

```
klasa Uczen
    imie: tekst
    nazwisko: tekst
    wiek: liczba całkowita

    funkcja Uczen()
        imie ← "Imię"
        nazwisko ← "Nazwisko"
        wiek ← 22

    funkcja Uczen(imieUcznia ← "Imię", nazwiskoUcznia ← "Nazwisko
        imie ← imieUcznia
        nazwisko ← nazwiskoUcznia
        wiek ← wiekUcznia
```

Drugi z konstruktorów przedstawionych w pseudokodzie pełni funkcję zarówno konstruktora domyślnego, jak i zwykłego.

Konstruktor kopiujący

Taki konstruktor przyjmuje tylko jeden argument - **referencję do obiektu tej samej klasy**. Następnie ma on za zadanie skopiowanie wartości pól z obiektu podanego przez referencję do nowo tworzonej instancji klasy.

```
klasa Uczen
    imie: tekst
    nazwisko: tekst
    wiek: liczba całkowita

    funkcja Uczen(uczen)
        imie ← uczen.imie
        nazwisko ← uczen.nazwisko
```

```
        wiek ← uczen.wiek
```

Przedstawiony przykład pokazuje praktyczne wykorzystanie konstruktora kopiującego. Utworzono obiekt klasy `Uczen` oraz – poprzez konstruktor zwykły – zostały podane wartości parametrów: `"Jan"`, `"Kowalski"`, `12`. Następnie utworzono drugi obiekt (również klasy `Uczen`: `uczen2`) za pomocą konstruktora kopiującego.

W obiekcie `uczen2` wartości pól `imie`, `nazwisko`, `wiek` będą takie same, jak w obiekcie `uczen`, czyli odpowiednio: `"Jan"`, `"Kowalski"`, `12`.

```
klasa Uczen
    imie: tekst
    nazwisko: tekst
    wiek: liczba całkowita

    funkcja Uczen(imieUcznia, nazwiskoUcznia, wiekUcznia)
        imie ← imieUcznia
        nazwisko ← nazwiskoUcznia
        wiek ← wiekUcznia

    funkcja Uczen(uczen)
        imie ← uczen.imie
        nazwisko ← uczen.nazwisko
        wiek ← uczen.wiek

Uczen uczen ← nowy Uczen("Jan", "Kowalski", 12)
Uczen uczen2 ← nowy Uczen(uczen)
```

Konstruktor teleskopowy

Szczególnym rodzajem konstruktora jest **konstruktor teleskopowy.** Zakłada on w rzeczywistości utworzenie kilku konstruktorów. Spójrzmy na przykład:

```
klasa Osoba
    imie: tekst
    drugieImie: tekst
    nazwisko: tekst
    nazwiskoRodowe: tekst
    wiek: liczba całkowita

```

```
    funkcja Osoba(imieOsoby, drugieImieOsoby, nazwiskoOsoby, nazw
        imie ← imieOsoby
        drugieImie ← drugieImieOsoby
        nazwisko ← nazwiskoOsoby
        nazwiskoRodowe ← nazwiskoRodoweOsoby
        wiek ← wiekOsoby
```

Przedstawiony konstruktor klasy `Osoba` umożliwia utworzenie i nadanie wartości atrybutom obiektu tej klasy. Gdybyśmy skorzystali z tego konstruktora, musielibyśmy podać wszystkie wartości atrybutów. Co jednak w przypadku, gdy dana osoba nie będzie miała drugiego imienia?

Konstruktor teleskopowy okazuje się bardzo przydatny, gdy w obrębie klasy są zdefiniowane atrybuty opcjonalne, takie jak np. drugie imię.

Zobaczmy, jak wygląda jego budowa:

```
klasa Osoba
    imie: tekst
    drugieImie: tekst
    nazwisko: tekst
    nazwiskoRodowe: tekst
    wiek: liczba całkowita

    funkcja Osoba(imieOsoby, nazwiskoOsoby, nazwiskoRodoweOsoby,
        imie ← imieOsoby
        nazwisko ← nazwiskoOsoby
        nazwiskoRodowe ← nazwiskoRodoweOsoby
        wiek ← wiekOsoby


    funkcja Osoba(imieOsoby, drugieImieOsoby, nazwiskoOsoby, nazw
        Osoba(imieOsoby, nazwiskoOsoby, nazwiskoRodoweOsoby, wiek
        drugieImie ← drugieImieOsoby
```

W tym przykładzie pierwszy konstruktor nadaje wartości atrybutom, które nie są opcjonalne – zakładamy, że każda osoba musi mieć imię, nazwisko, nazwisko rodowe i wiek. Drugi konstruktor zawiera w sobie wywołanie pierwszego konstruktora (dla atrybutów `imie`, `nazwisko`, `nazwiskoRodowe` oraz `wiek`), a także oprócz tego nadaje wartość drugiego imienia.

Destruktory

Destruktor, jak sugeruje nazwa, jest **przeciwieństwem konstruktora**. Zostaje on najczęściej wywołany automatycznie, tuż przed **usunięciem** danego obiektu klasy. Ma na celu przygotowanie do fizycznego zwolnienia zajmowanej przez obiekt pamięci. W destruktorze można zawrzeć m.in. instrukcję zamknięcia połączenia z plikiem lub wyrejestrowanie się z innych obiektów. Destruktor nie przyjmuje żadnych argumentów.

Co istotne – podczas gdy klasa może posiadać wiele konstruktorów, destruktor może być **tylko jeden.**

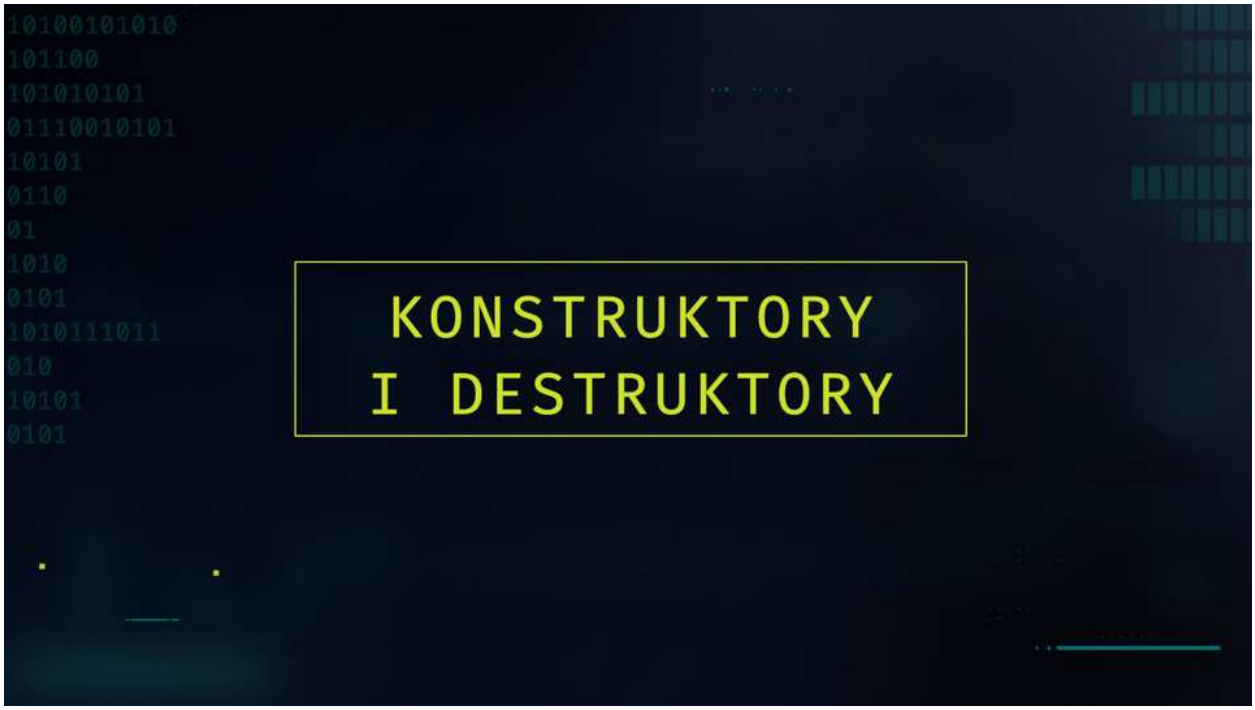
Słownik

> **instancja klasy**
>
> niezależny obiekt danej klasy, zajmujący określone miejsce w pamięci
> **argument**
>
> konkretna wartość lub zmienna przekazywana do funkcji lub metody przy jej wywołaniu
> **parametr**
>
> definicja zmiennej zawarta w deklaracji metody lub funkcji; deklarujemy ją w nawiasach, np. `funkcja(liczba parametr1, tekst parametr2)`
> **referencja**
>
> zmienna zawierająca informacje o położeniu w pamięci innej zmiennej lub obiektu

Animacja

> **Polecenie 1**
>
> Zapoznaj się z animacją, w której przedstawiono zastosowanie konstruktorów i destruktorów w programie. Następnie wykonaj ćwiczenie 1.

> **Uwaga!**
>
> W filmie dla uproszczenia pseudokodu dodano typy danych. Wprowadzone modyfikacje pozwalają na łatwiejsze przeniesienie kodu na języki typowane (C++, Java). W przypadku języka Python typy pól pomijamy.
>
> Zastosowane typy danych:
>
> - `string` – łańcuch znaków;
> - `int` – liczba całkowita;
> - `zmiennaLogiczna` – typ logiczny.



Film dostępny pod adresem /preview/resource/R9xGTsrGvradZ

Film dotyczy konstruktorów i destruktorów.

Pseudokod obiekt zwierzęcia zaprezentowany w filmie.

Plik o rozmiarze 748.00 B w języku polskim

Pseudokod zarządzający grą zaprezentowany w filmie.

Plik o rozmiarze 228.00 B w języku polskim

Ćwiczenie 1

Przygotuj słownik najważniejszych pojęć pojawiających się w animacji.

Sprawdź się

Pokaż ćwiczenia:

Ćwiczenie 1

Zaznacz wszystkie poprawne odpowiedzi. Wskaż wybrane typy konstruktorów.

- [ ] domyślny
- [ ] prosty
- [ ] kopiujący
- [ ] teleskopowy

Ćwiczenie 2

Wskaż poprawną odpowiedź.
Czy w klasie może być zadeklarowanych kilka konstruktorów?

- ( ) nie
- ( ) tak

Ćwiczenie 3

Uzupełnij zdanie właściwymi wyrazami.

Konstruktor to specjalna [ ], metoda klasy, [ ], podczas gdy tworzona jest [ ] danej [ ].

start | funkcji | metody | funkcja | wywoływana | instancja | klasy

Ćwiczenie 4

Wskaż, ile razy może zostać zadeklarowany destruktor w klasie.

- ◯ dwa
- ◯ nieskończenie wiele
- ◯ tyle, ile jest konstruktorów w danej klasie
- ◯ jeden

Ćwiczenie 5

Połącz każdy typ konstruktora z właściwym opisem.

konstruktor teleskopowy	Ma za zadanie skopiowanie wartości pól z obiektu podanego przez referencje do nowo tworzonej instancji klasy.
konstruktor domyślny	Jest przydatny, gdy w obrębie klasy są zdefiniowane pola opcjonalne.
konstruktor kopiujący	Nie ma zdefiniowanych parametrów.

Ćwiczenie 6

Uzupełnij zdanie właściwymi wyrazami. Do wyboru są elementy, którymi można uzupełnić zdanie.

Destruktor jest [ ] [ ]. Zostanie on [ ] tuż przed [ ] danego [ ] klasy.

utworzeniem | typem | konstruktora | rodzajem | parametru | usunięciem | przeciwieństwem | wywołany | zadeklarowany | obiektu

Ćwiczenie 7

Pogrupuj informacje dotyczące konstruktorów i destruktorów.

konstruktor

destruktor

- w klasie może być tylko jeden
- w klasie może być ich kilka
- nigdy nie przyjmuje żadnych argumentów
- może przyjmować argumenty
- wywołanie nastąpi przy inicjalizacji obiektu danej klasy
- wywoływany przed usunięciem obiektu danej klasy

Ćwiczenie 8

Wskaż przykładowe zadanie destruktora.

- ◯ otwarcie połączenia z plikiem
- ◯ zamknięcie połączenia z plikiem
- ◯ inicjalizacja pól w obiekcie

Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Konstruktory i destruktory

Grupa docelowa:

Szkoła ponadpodstawowa, liceum ogólnokształcące, technikum, zakres rozszerzony

Podstawa programowa:

Cele kształcenia – wymagania ogólne

> II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera oraz innych urządzeń cyfrowych: układanie i programowanie algorytmów, organizowanie, wyszukiwanie i udostępnianie informacji, posługiwanie się aplikacjami komputerowymi.

Treści nauczania – wymagania szczegółowe

> II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych.
>
> > Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:
> >
> > > 1) projektuje i tworzy rozbudowane programy w procesie rozwiązywania problemów, wykorzystuje w programach dobrane do algorytmów struktury danych, w tym struktury dynamiczne i korzysta z dostępnych bibliotek dla tych struktur;
> > >
> > > 2) stosuje zasady programowania strukturalnego i obiektowego w rozwiązywaniu problemów;

Kształtowane kompetencje kluczowe:

- kompetencje cyfrowe;
- kompetencje osobiste, społeczne i w zakresie umiejętności uczenia się;
- kompetencje matematyczne oraz kompetencje w zakresie nauk przyrodniczych, technologii i inżynierii.

Cele operacyjne (językiem ucznia):

- Wyjaśnisz, czym są konstruktory i destruktory oraz jakie funkcje pełnią w programowaniu obiektowym.

- Wymienisz typy konstruktorów i scharakteryzujesz je.
- Wykonasz kilka ćwiczeń sprawdzających wiedzę z podstaw programowania obiektowego.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

- dyskusja;
- rozmowa nauczająca z wykorzystaniem multimedium i ćwiczeń interaktywnych;
- ćwiczenia praktyczne.

Formy pracy:

- praca indywidualna;
- praca w parach;
- praca w grupach;
- praca całego zespołu klasowego.

Środki dydaktyczne:

- komputery z głośnikami, słuchawkami i dostępem do internetu;
- zasoby multimedialne zawarte w e-materiale;
- tablica interaktywna/tablica, pisak/kreda.

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Konstruktory i destruktory". Nauczyciel prosi uczniów o zapoznanie się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Nauczyciel wyświetla temat i cele zajęć zawarte w sekcji „Wprowadzenie". Prosi uczniów, by na podstawie wiadomości zdobytych przed lekcją zaproponowali kryteria sukcesu.
2. **Rozpoznanie wiedzy uczniów.** Nauczyciel prosi wybranego ucznia lub uczniów o przedstawienie sytuacji problemowej związanej z tematem lekcji.

Faza realizacyjna:

1. **Praca z tekstem.** Jeżeli przygotowanie uczniów do lekcji jest niewystarczające, nauczyciel prosi o indywidualne zapoznanie się z treścią zawartą w sekcji „Przeczytaj". Każdy uczestnik zajęć podczas cichego czytania wynotowuje najważniejsze kwestie poruszane w tekście.
2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Animacja". Uczniowie wspólnie zapoznają się treścią filmu, w którym za pomocą pseudokodu przedstawiono problem programowania obiektowego. Zapisują ewentualne problemy i pytania, po czym następuje dyskusja, w trakcie której nauczyciel wyjaśnia niezrozumiałe treści.
3. **Ćwiczenie umiejętności.** Uczniowie wykonują indywidualnie ćwiczenia nr 1-6 z sekcji „Sprawdź się", a następnie porównują swoje odpowiedzi z kolegą lub koleżanką.

Faza podsumowująca:

1. Nauczyciel ponownie wyświetla na tablicy temat i cele lekcji zawarte w sekcji „Wprowadzenie". W kontekście ich realizacji następuje omówienie ewentualnych problemów z rozwiązaniem ćwiczeń z sekcji „Sprawdź się".
2. Na koniec zajęć nauczyciel prosi uczniów o rozwinięcie zdania: „Na dzisiejszych zajęciach nauczyłam/łem się jak…".

Praca domowa:

1. Uczniowie wykonują ćwiczenia 7-8 z sekcji „Sprawdź się".

Wskazówki metodyczne:

- Treści w sekcji „Animacja" można wykorzystać na lekcji jako podsumowanie i utrwalenie wiedzy uczniów.