Text

Description automatically generated with medium confidence

wzorce projektowe

Sprawozdanie

Zadanie 8 – Chain of responsibility

Michał Kuźniecow

Informatyka, programowanie

Grupa 33\_Inf\_P\_NW\_6

Rok akademicki 2023/24

# Cel

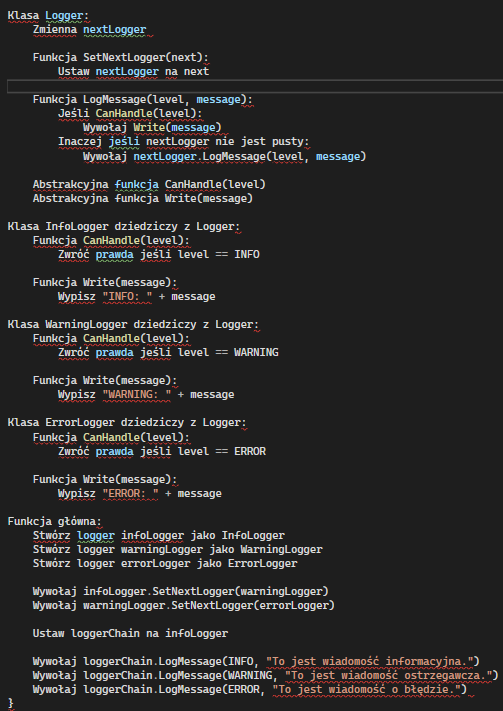
Dokument ten powstał w celu sporządzenia sprawozdania z wykonywania ćwiczenia numer 8 – Chain of responsibility z przedmiotu Wzorce Projektowe.

W ramach ćwiczenia zgodnie z zadaniem wymagane było rozważenie stworzenie systemu logowania dla aplikacji. Należało zaimplementować wzorzec łańcucha odpowiedzialności, aby utworzyć łańcuch logowania. Każde logowanie w łańcuchu powinno obsługiwać komunikaty np. INFO, OSTRZEŻENIE, BŁĄD. Jeśli logowanie nie jest w stanie obsłużyć danego komunikatu, powinno przekazać ten komunikat następnemu logowaniu w łańcuchu .Z uwagi na założenia zadania do rozwiązania zastosowany został wzorzec projektowy łańcucha odpowiedzialności.

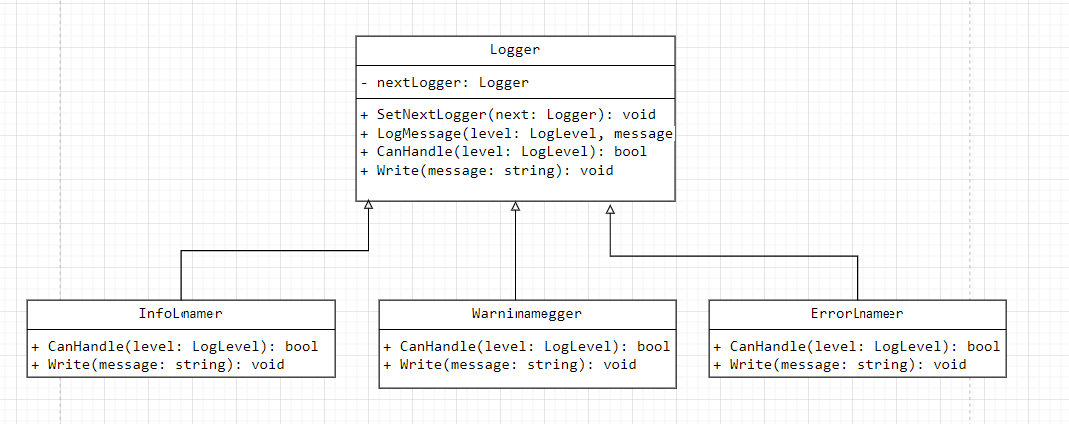
# Opis rozwiązania

W tej części proszę opisać krótko propozycję rozwiązania zadania:

**PSEUDOKOD:**

****

**PROPOZYCJA DIAGRAMU KLAS:**

****

**OPIS ZASTOSOWANEGO ROZWIĄZANIA:**

**Handler (Logger)** Klasa Logger definiuje interfejs do ustawiania następnego loggera w łańcuchu (SetNextLogger) oraz metodę do logowania wiadomości (LogMessage).

LogMessage sprawdza, czy dany logger może obsłużyć poziom komunikatu (CanHandle). Jeśli tak, wywołuje metodę Write. W przeciwnym razie przekazuje komunikat do następnego loggera w łańcuchu.

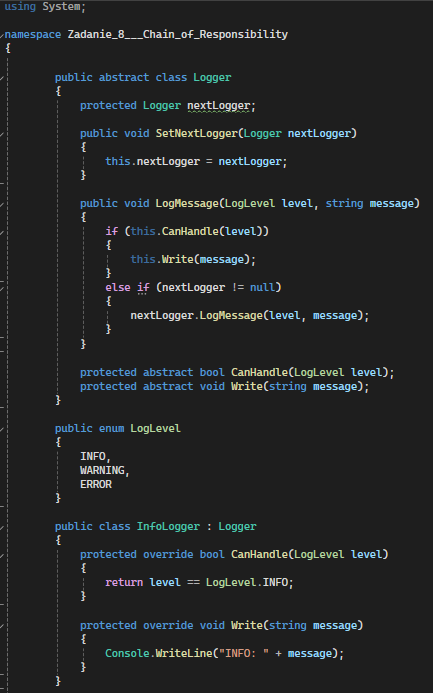
**Konkretne Loggery** (InfoLogger, WarningLogger, ErrorLogger): Każdy konkretny logger dziedziczy z Logger i implementuje metody CanHandle oraz Write, aby obsługiwać odpowiednie poziomy logowania.

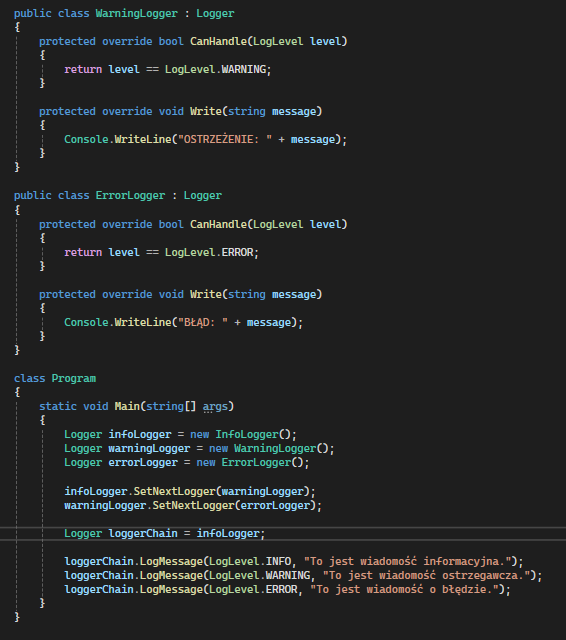
**Konfiguracja Łańcucha**: W funkcji głównej tworzone są instancje konkretnych loggerów. Logger-y są łączone w łańcuch za pomocą SetNextLogger. Przykładowe komunikaty są wysyłane do pierwszego loggera w łańcuchu, który przekazuje je dalej w zależności od ich poziomu.

To rozwiązanie pozwala na elastyczne dodawanie nowych poziomów logowania i modyfikację łańcucha bez zmiany istniejących loggerów.

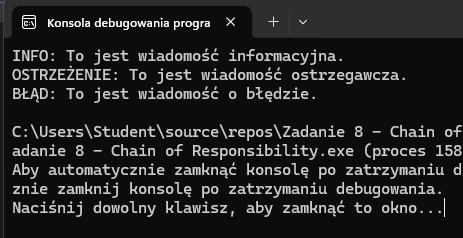
# Implementacja

W tej części proszę zamieścić listing kodu.





**WYNIK DZIAŁANIA PROGRAMU:**



# Podsumowanie

Wzorzec projektowy łańcuch odpowiedzialności został zastosowany w tym zadaniu z uwagi na treść jego polecenia. Implementacja wyżej wymienionego wzorca projektowego przebiegła pomyślnie, jak widać na uzyskanym wyniku działania programu logowania zostały poprawnie przyporządkowane do wybranych poziomów.

W mojej opinii to ten wzorzec projektowy sprawdza się najlepiej ponieważ każdy typ zapytania jest obsługiwany przez oddzielną klasę (handler). Dzięki temu odpowiedzialność za obsługę różnych logowań jest rozdzielona między różne klasy, co upraszcza kod i ułatwia jego zrozumienie oraz utrzymanie. Dodawanie nowych typów zapytań jest proste i nie wymaga modyfikacji istniejących klas. Wystarczy utworzyć nową klasę obsługującą (handler) i włączyć ją do łańcucha. Dzięki temu system jest łatwo skalowalny.

# Lista załączników

https://github.com/KapitanSikMik/Zad8ChainOfResponsibility