Text

Description automatically generated with medium confidence

wzorce projektowe

Sprawozdanie

Zadanie x

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 7 – Chain of responsibility z przedmiotu Wzorce Projektowe.

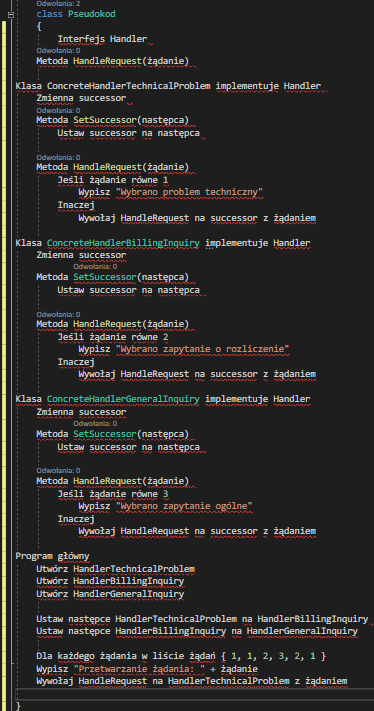
W ramach ćwiczenia zgodnie z zadaniem wymagane było rozważenie system zarządzania biletami obsługi klienta (tzw. tickets). Należało zaimplementować wzorzec łańcucha odpowiedzialności, aby obsługiwać różne typy zgłoszeń do pomocy technicznej, takie jak problemy techniczne, zapytania dotyczące rozliczeń i zapytania ogólne. Każdy handler w łańcuchu powinien być odpowiedzialny za obsługę określonego rodzaju biletu. Należało również stworzyć kod klienta, który generuje różne bilety pomocy technicznej i przekazuje je przez łańcuch do obsługi.

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:**

Interfejs IHandler:

* Definiuje metodę HandleRequest, która przyjmuje parametr typu int reprezentujący zapytanie.

Klasy konkretnych obsługujących:

* ConcreteHandlerTechnicalProblem: Obsługuje zapytania o wartościach równych 1 (problemy techniczne).
* ConcreteHandlerBillingInquiry: Obsługuje zapytania o wartościach równych 2 (zapytania o rozliczenia).
* ConcreteHandlerGeneralInquiry: Obsługuje zapytania o wartościach równych 3 (zapytania ogólne).

Każda z tych klas implementuje interfejs IHandler i posiada referencję do następnego obsługującego (następcy), której używa, jeśli nie jest w stanie obsłużyć zapytania.

Metoda SetSuccessor:

* Ustawia następnika dla każdej klasy obsługującej, tworząc łańcuch odpowiedzialności.

Metoda HandleRequest:

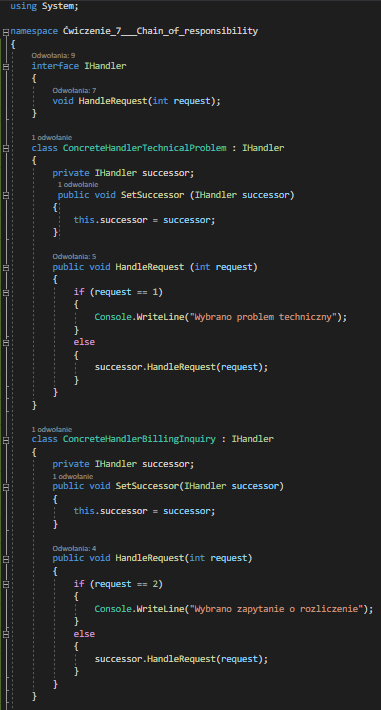
* Sprawdza wartość zapytania. Jeśli wartość jest obsługiwana przez konkretną klasę, wykonuje odpowiednią akcję (wypisuje wiadomość na konsolę). Jeśli nie, przekazuje zapytanie do następcy w łańcuchu.

Klasa Program i metoda Main:

* Tworzy instancje obsługujących i ustawia ich następców, tworząc łańcuch odpowiedzialności.
* Definiuje listę zapytań do przetworzenia.Iteruje przez każde zapytanie, rozpoczynając przetwarzanie od pierwszego obsługującego w łańcuchu (ConcreteHandlerTechnicalProblem).

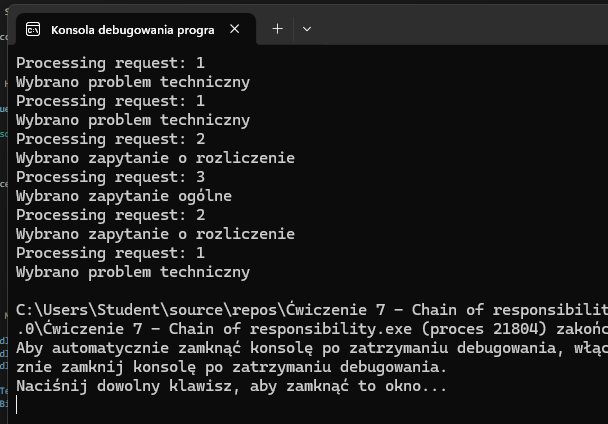
# Implementacja

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 tickety zostały poprawnie przyporządkowane do wybranych zapytań.

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 zapytań 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/Zad7ChainOfResponsibility>