Paweł Kiełbasa, Wojciech Kosztyła

Sprawozdanie PO – Testy

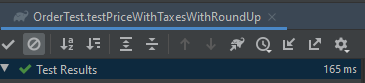
1. Zmiana wartości podatku na 23% - Zmiana wartości w klasie Order

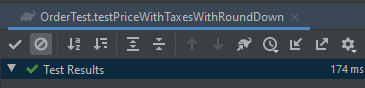


Na potrzebę sprawdzenia zaokrąglania kwoty produktu w testach zdodano przypadki w sytacjach brzegowych



Rezultat

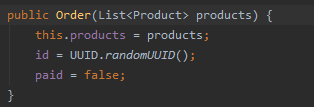




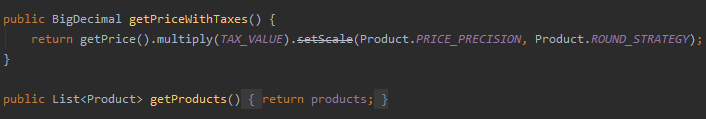
1. Dodanie możliwości posiadania wielu produktów w jednym zamówieniu. W celu zaimplementowania tej funkcjonalności został zmieniony typ pola products w klasie Order

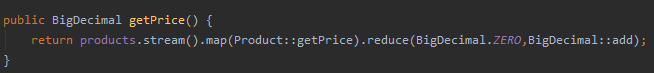


Zmiana w kostruktorze

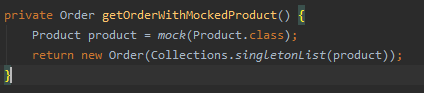


W związku ze zmianą typu pola product musiały zostać zmienione metody getProducts i getPrice





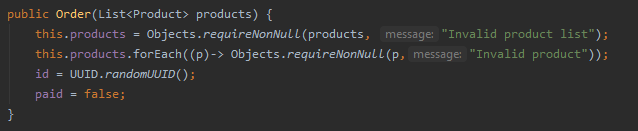
W związku z powyższymi zmianami wszystkie testy zawierające konstrunktor klasy Order przyjmujący jako argument Product musiały zostać zmienione. Jeden z przykładów poniżej



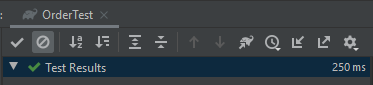
Oprócz tego dodano test sprawdzający cenę zamówienia z większą ilością produktów



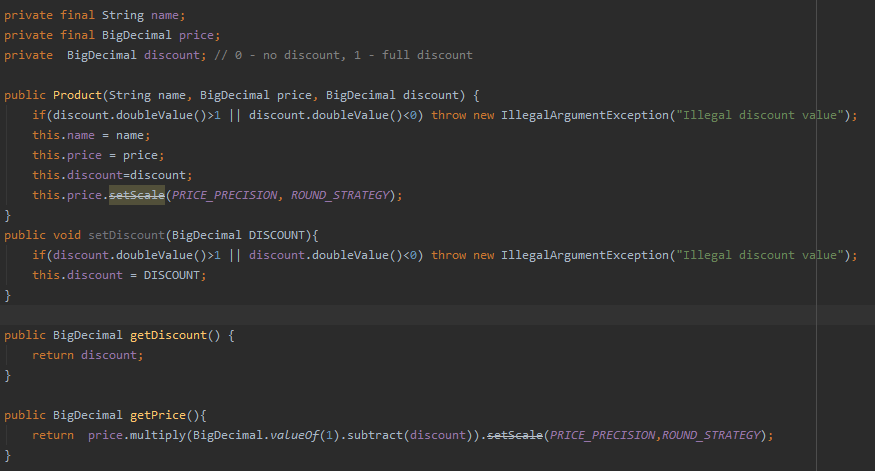
Aby zapobiec próbie wstawienia pustej listy produktów do zamówienia należy zmianić kostruktor klasy Order



Wyniki testów



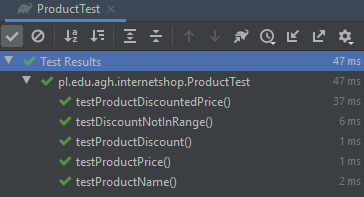
1. Dodanie rabatu na całe zamówienie i pojedyncze produkty. Wyamga to dodania nowego pola w klasie Product oraz getterów i seterów



W związku z dodaniem nowej funkcjonalności stworzyłem dodatkowe testy na sprawdzenie czy dodatkowe metody działają



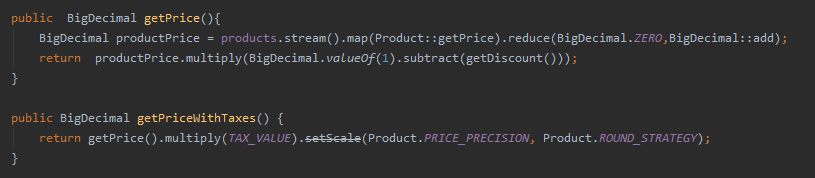
Otrzymano następujące wyniki testów



Dodanie możliwości posiadania zniżki w klasie Order (wymaga to zmiany w konstruktorze oraz dodania gettera i setera dla danego pola).



W związku z powyższymi modyfikacjami metody getPrice i getPriceWithTaxes musiały zostać zmienione

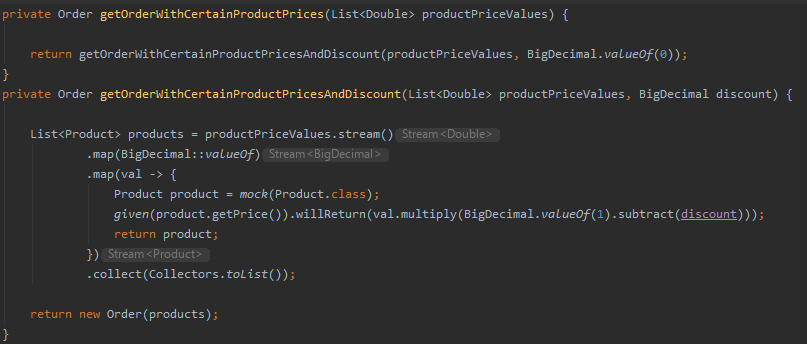


Do klasy OrderTest dodałem następujące testy aby sprawdzić działanie nowych funkcji. Wymagało to dodania metody dodawania zamówienia z określonymi kwotami produktów oraz zniżką ( na potrzeby testów założyłem że zniżka dla każdego produktu jest taka sama

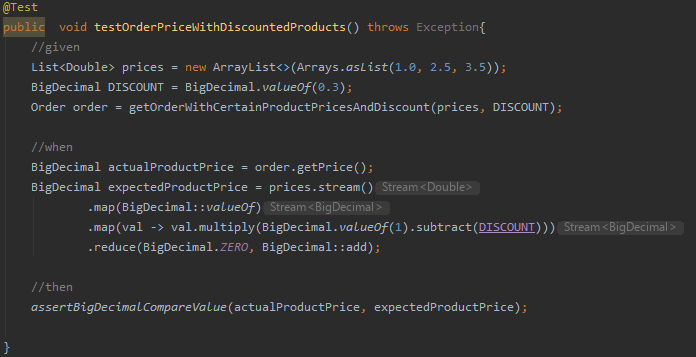
Test gettera i setera dla zniżki

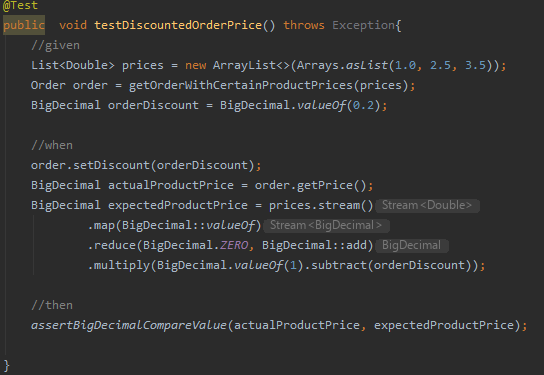


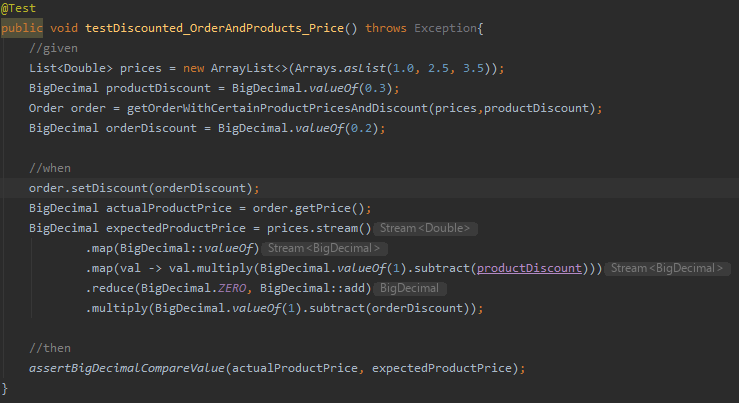
Metody tworzące nowe zamówienie na podstawie danych cen i wartości zniżki



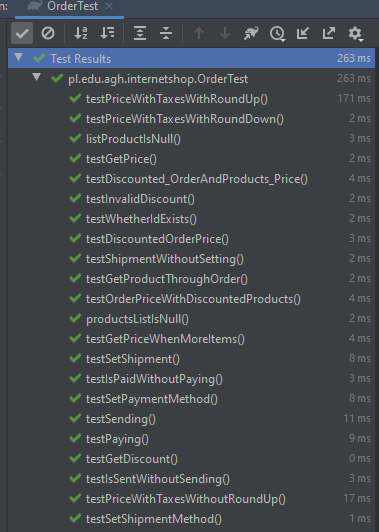
Testy sprawdzające poprawność naliczania zniżek

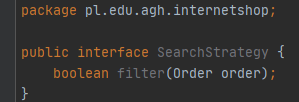
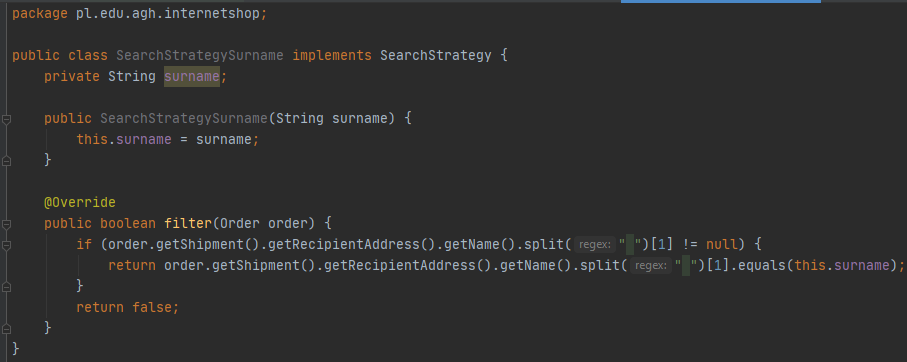
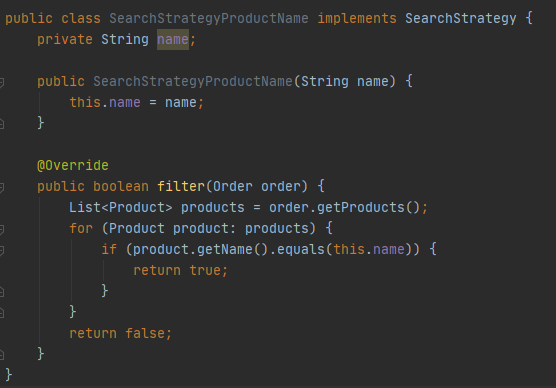
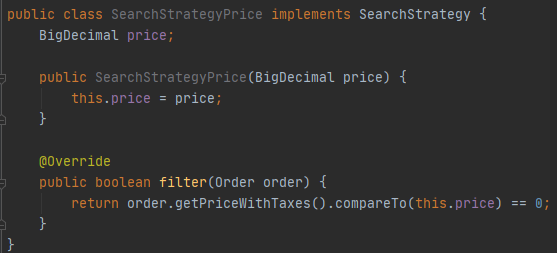
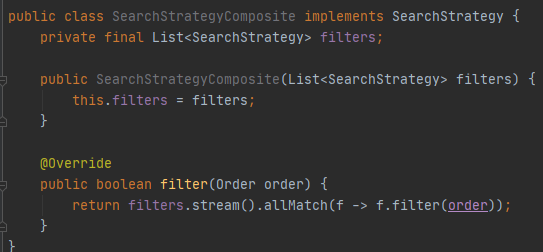
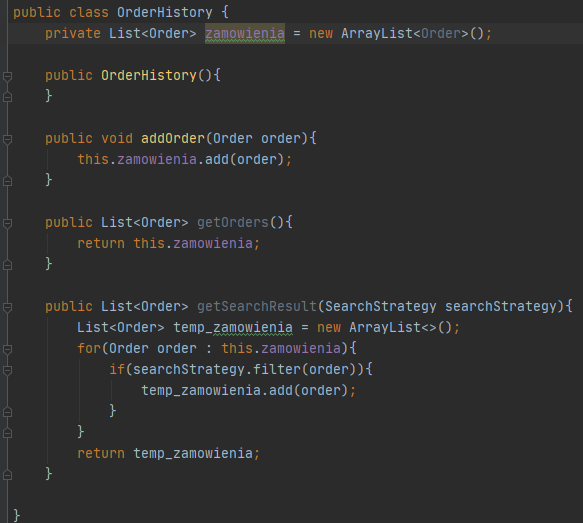
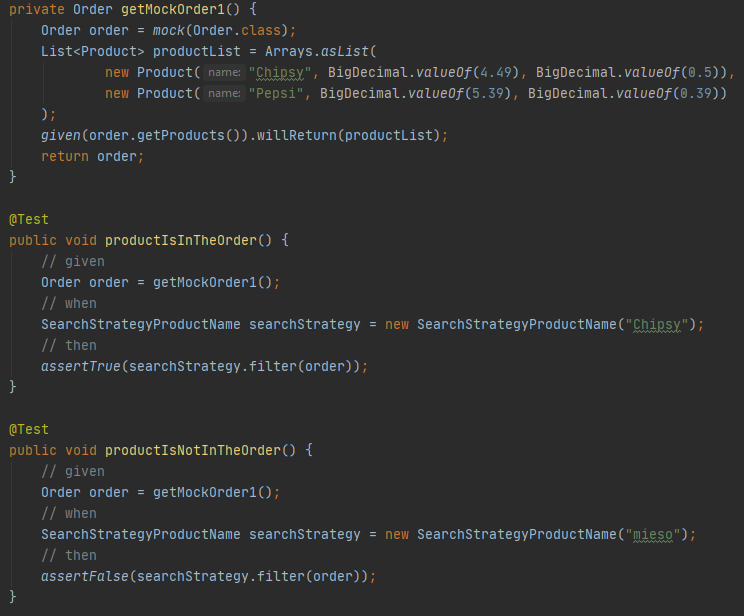
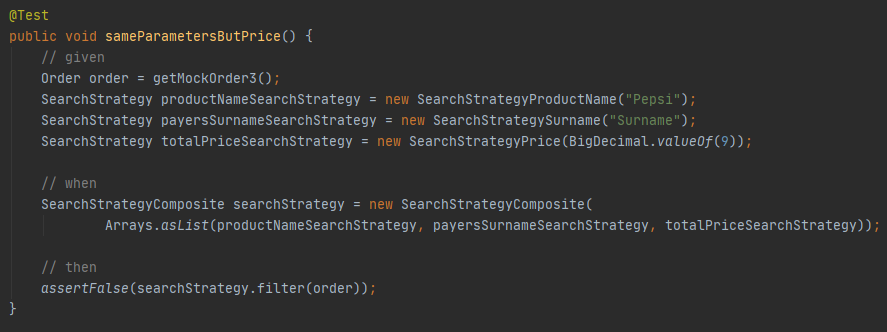
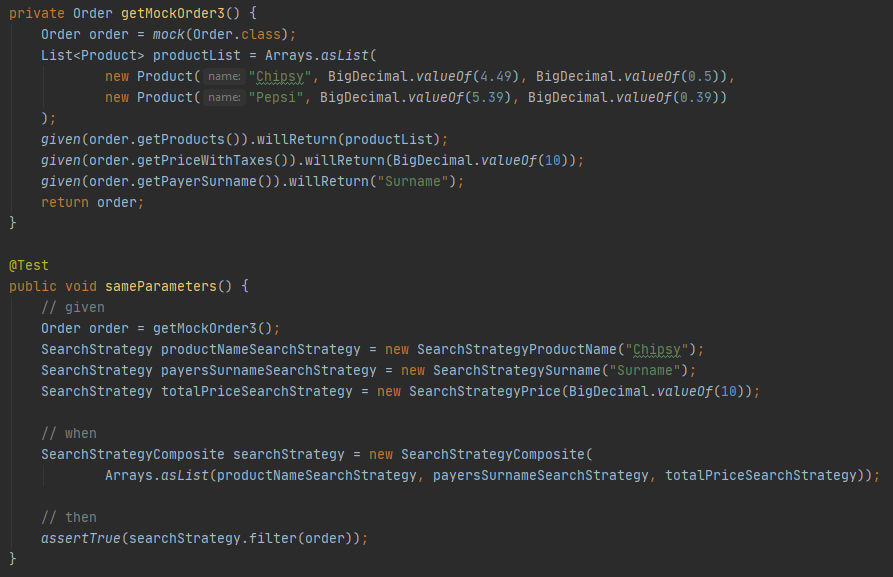
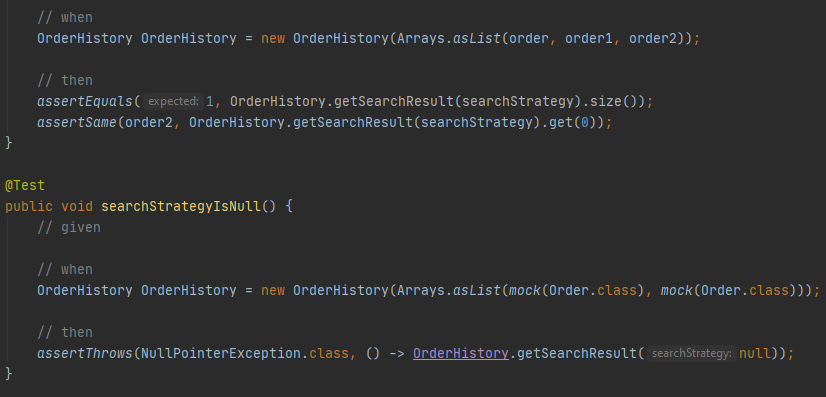
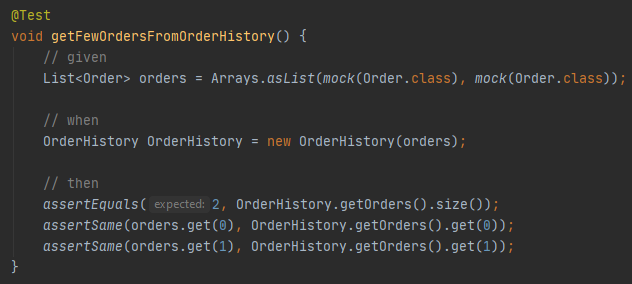






Otrzymano następujące wyniki testów



1. Umożliwienie przechowywania historii zamówień z wyszukiwaniem po: nazwie produktu, kwocie zamówienia, nazwisku zamawiającego. Wyszukać można przy użyciu jednego lub wielu kryteriów.  
   (już tworzyliśmy filtry i na tamtych filtrach się bazowaliśmy)
   1. Stworzyłem interfejs filtrów  
      
   2. Dodałem klasy obsługujące filtrowanie  
        
        
        
        
      
   3. Aby wykorzystać dowolną kombinację filtrów stworzyłem kompozyt SearchStrategyComposite.  
      
   4. Aby przechowywać wszystkie zamówienia dodałem klasę OrderHistory  
      
   5. Dodałem klasę OrderHistoryTest, aby testować wszystkie dodane klasy  
      
      1. productIsInTheOrder – sprawdza, czy filtr z ProductName działa
      2. productIsNotInTheOrder – analogicznie, czy filtr działa i „mieso” nie należy do Orderu
   6. 
      1. sameSurname – sprawdza działanie filtru „surname”, czy funkcje zwrócą znaleziony wynik
      2. notTheSameSurname – analogicznie do tego wyżej
      3. surnameIsNull – sprawdza działanie programu na wypadek podania do filtru wartość null
   7. 
      1. samePrice – sprawdza, czy filtr z „price” zadziała i pokaże właśnie dodany Order
      2. notTheSamePrice – na odwrót – sprawdza czy filtr działa i czy przy podanym błędnym polu coś zwróci
   8. 
      1. sameParameters – test wspólnego działania wszystkich filtrów
      2. sameParametersBut \* - testy każdej kombinacji dwóch filtrów
   9. 
      1. getFewOrdersFromOrdersHistory – sprawdza poprawne działanie historii Order’ów
      2. getSearchResultsWith\* - sprawdzają zachowanie programu, gdy muszą obsłużyć więcej niż jeden filtr każdego typu
      3. getCompositeSearchResults – sprawdza działanie getSearchResults przy filtrach na wszystkich polach
      4. searchStrategyIsNull – sprawdza zachowanie programu w przypadku podania nulowego / nie podania sposobu filtracji
2. Wyniki testów:  
   