Wariant 151

- Utwórz pakiet `math.functions` dla przeprowadzenia kolejnych działań.
- Zdefiniuj interfejs `Calculator` z dwoma metodami abstrakcyjnymi: `calculate(int number)` zwracającą `int` i `isPositive(int number)` zwracającą `boolean`.
- Stwórz klasę `SimpleCalculator`, która implementuje `Calculator`. W metodzie `calculate` zwróć kwadrat podanej liczby, a w metodzie `isPositive` sprawdź, czy liczba jest dodatnia. Zapewnij obsługę przypadków, gdy argumentem jest wartość null (o ile to potrzebne).
- W klasie `TestCalculator` utwórz 2 obiekty i przetestuj metody.

Zadanie należy umieścić we własnym repozytorium na Githubie.

Wariant 152

- Utwórz pakiet `user.management` gdzie będą realizowane kolejne zadania.
- Zdefiniuj interfejs `UserValidator` z dwoma metodami abstrakcyjnymi: `validateUser(String userId)` zwracającą `boolean` i `isAdult(int age)` zwracającą `boolean`.
- Stwórz klasę `SimpleUserValidator`, która implementuje `UserValidator`. W metodzie `validateUser` sprawdź, czy identyfikator użytkownikajest trzycyfrową liczbą naturalną, a w metodzie `isAdult` sprawdź, czy wiek użytkownika pozwala na zakwalifikowanie go jako dorosłego. Zapewnij obsługę sytuacji, gdy argumenty są nullami (o ile to potrzebne).
- W klasie `TestUserValidator` utwórz 2 obiekty i przetestuj metody.

Zadanie należy umieścić we własnym repozytorium na Githubie.

Wariant 153

- Utwórz pakiet `product.validation` dla przeprowadzenia kolejnych działań.
- Zdefiniuj interfejs `ProductValidator` z dwoma metodami abstrakcyjnymi: `isValidProduct(String productId)` zwracającą `boolean` i `isInStock(int quantity)` zwracającą `boolean`.
- Stwórz klasę `SimpleProductValidator`, która implementuje `ProductValidator`. W metodzie `isValidProduct` sprawdź, czy identyfikator produktu jest sześciocyfrową liczbą, a w metodzie `isInStock` sprawdź, czy ilość produktu na stanie jest większa niż 0. Zapewnij obsługę sytuacji, gdy argumenty są nullami (o ile to potrzebne).
- W klasie `TestProductValidator` utwórz 2 obiekty i przetestuj metody.

Zadanie należy umieścić we własnym repozytorium na Githubie.

Wariant 154

- Utwórz pakiet `task.management` gdzie będą realizowane kolejne zadania.
- Zdefiniuj interfejs `TaskValidator` z dwoma metodami abstrakcyjnymi: `isValidTask(String taskId)` zwracającą `boolean` i `isPriorityHigh(int priority)` zwracającą `boolean`.
- Stwórz klasę `SimpleTaskValidator`, która implementuje `TaskValidator`. W metodzie `isValidTask` sprawdź, czy identyfikator zadania jest pięciocyfrową liczbą, a w metodzie `isPriorityHigh` sprawdź, czy priorytet zadania jest większy niż 5. Zapewnij obsługę sytuacji, gdy argumenty są nullami (o ile to potrzebne).
- W klasie `TestTaskValidator` utwórz 2 obiekty i przetestuj metody.

Zadanie należy umieścić we własnym repozytorium na Githubie.