Wariant 155

- Utwórz pakiet `math.absolute` dla operacji obliczania wartości bezwzględnej.
- Zdefiniuj interfejs `AbsoluteCalculator` z dwoma metodami abstrakcyjnymi: `calculateAbsolute(int number)` zwracającą `int` i `isNegative(int number)` zwracającą `boolean`.
- Stwórz klasę `SimpleAbsoluteCalculator`, która implementuje `AbsoluteCalculator`.
  W `calculateAbsolute` zwróć wartość bezwzględną liczby, a w `isNegative` sprawdź, czy liczba jest ujemna. Obsłuż przypadki, gdy argument jest wartością null.
- W klasie `TestAbsoluteCalculator` utwórz 2 obiekty i przetestuj metody.

Zadanie należy umieścić we własnym repozytorium na Githubie.

Wariant 156

- Utwórz pakiet `math.operations` dla wykonania operacji arytmetycznych.
- Zdefiniuj interfejs `ArithmeticOperations` z dwoma metodami abstrakcyjnymi: `add(int a, int b)` zwracającą `int` i `isEven(int number)` zwracającą `boolean`.
- Stwórz klasę `BasicArithmetic`, która implementuje `ArithmeticOperations`. W metodzie `add` zwróć sumę dwóch liczb, a w `isEven` sprawdź, czy liczba jest parzysta. Obsłuż przypadki, gdy argumenty są wartościami null (o ile to konieczne).
- W klasie `TestArithmetic` utwórz obiekty i przetestuj metody.

Zadanie należy umieścić we własnym repozytorium na Githubie.

Wariant 157

- Utwórz pakiet `math.factorial` dla obliczeń związanych z silnią.
- Zdefiniuj interfejs `FactorialCalculator` z dwoma metodami abstrakcyjnymi: `calculateFactorial(int number)` zwracającą `int` i `isNonNegative(int number)` zwracającą `boolean`.
- Stwórz klasę `SimpleFactorialCalculator`, która implementuje `FactorialCalculator`.
  W `calculateFactorial` oblicz silnię danej liczby, a w `isNonNegative` sprawdź, czy liczba nie jest ujemna. Zapewnij obsługę przypadków, gdy argument jest wartością null.
- W klasie `TestFactorialCalculator` utwórz 2 obiekty i przetestuj metody.

Zadanie należy umieścić we własnym repozytorium na Githubie.

Wariant 158

- Utwórz pakiet `math.inverse` dla obliczeń odwrotności liczby.
- Zdefiniuj interfejs `InverseCalculator` z dwoma metodami abstrakcyjnymi: `calculateInverse(int number)` zwracającą `double` i `isNonZero(int number)` zwracającą `boolean`.
- Stwórz klasę `SimpleInverseCalculator`, która implementuje `InverseCalculator`.
  W `calculateInverse` oblicz odwrotność liczby (1 podzielone przez liczbę), a w `isNonZero` sprawdź, czy liczba nie jest równa zero. Obsłuż przypadki, gdy argument jest wartością null.
- W klasie `TestInverseCalculator` utwórz 2 obiekty i przetestuj metody.

Zadanie należy umieścić we własnym repozytorium na Githubie.