Wariant 161

- Utwórz pakiet `math.sqrt` dla operacji obliczania pierwiastka kwadratowego.
- Zdefiniuj interfejs `SquareRootCalculator` z dwoma metodami abstrakcyjnymi: `calculateSqrt(int number)` zwracającą `double` i `isNonNegative(int number)` zwracającą `boolean`.
- Stwórz klasę `SimpleSquareRootCalculator`, która implementuje `SquareRootCalculator`. W `calculateSqrt` oblicz pierwiastek kwadratowy liczby, a w `isNonNegative` sprawdź, czy liczba nie jest ujemna.
- W klasie `TestSquareRootCalculator` utwórz 2 obiekty i przetestuj metody.

Zadanie należy umieścić we własnym repozytorium na Githubie.

Wariant 162

- Utwórz pakiet `math.logarithm` dla operacji logarytmicznych.
- Zdefiniuj interfejs `LogarithmCalculator` z dwoma metodami abstrakcyjnymi: `calculateLog(int number)` zwracającą `double` i `isGreaterThanOne(int number)` zwracającą `boolean`.
- Stwórz klasę `SimpleLogarithmCalculator`, która implementuje `LogarithmCalculator`. W `calculateLog` oblicz logarytm naturalny podanej liczby, a w `isGreaterThanOne` sprawdź, czy liczba jest większa od jednego.
- W klasie `TestLogarithmCalculator` utwórz 2 obiekty i przetestuj metody.

Zadanie należy umieścić we własnym repozytorium na Githubie.

Wariant 163

- Utwórz pakiet `math.trigonometry` dla operacji trygonometrycznych.
- Zdefiniuj interfejs `TrigonometryCalculator` z dwoma metodami abstrakcyjnymi: `calculateSin(int angle)` zwracającą `double` i `isAcuteAngle(int angle)` zwracającą `boolean`.
- Stwórz klasę `SimpleTrigonometryCalculator`, która implementuje `TrigonometryCalculator`. W `calculateSin` oblicz sinus kąta (wyrażonego w stopniach), a w `isAcuteAngle` sprawdź, czy kąt jest ostrym (mniejszym niż 90 stopni). Zapewnij obsługę przypadków, gdy argumentem jest wartość null.
- W klasie `TestTrigonometryCalculator` utwórz 2 obiekty i przetestuj metody.

Zadanie należy umieścić we własnym repozytorium na Githubie.

Wariant 164

- Utwórz pakiet `task.status` dla zarządzania statusami zadań.
- Zdefiniuj interfejs `StatusValidator` z dwoma metodami abstrakcyjnymi: `isStatusValid(String status)` zwracającą `boolean` i `isStatusCompleted(String status)` zwracającą `boolean`.
- Stwórz klasę `SimpleStatusValidator`, która implementuje `StatusValidator`. W `isStatusValid` sprawdź, czy status jest jednym z: "New", "In Progress", "Completed", a w `isStatusCompleted` sprawdź, czy status to "Completed". Zapewnij obsługę sytuacji, gdy argumenty są nullami.
- W klasie `TestStatusValidator` utwórz 2 obiekty i przetestuj metody.

Zadanie należy umieścić we własnym repozytorium na Githubie.