Wariant 501

Wykonaj poniższe czynności:

- Stwórz pakiet `com.holidays.akt` i umieść w oddzielnych plikach poniższe klasy.
- stwórz klasę `Trip` z dwoma prywatnymi polami `destination` (typu `String`) oraz `price` (typu double).
- Zaimplementuj konstruktor, który przyjmuje dwa argumenty odpowiadające polom klasy. Nazwy parametrów muszą być takie same jak nazwy pól.
- Dodaj metody dostępowe (gettery i settery).
- Zapewnij walidację, tak aby cena nie była ujemna. W konstruktorze w przypadku błędu przyjmij cenę domyślną `10.12`. W setterze nie ustawiaj wcale błędnej wartości, w przypadku błędnej wartości metoda ma nic nie robić.
- Stwórz klasę `TestTrip` z metodą `main`. Wywołaj tu każdą z metod zdefiniowaną wyżej co najmniej jeden raz.

Zadanie należy umieścić we własnym repozytorium na Githubie.

Wariant 502

Wykonaj poniższe czynności:

- Utwórz pakiet `pl.travel.akt` i umieść w nim w oddzielnych plikach poniższe klasy.
- Stwórz klasę `Vacation` z dwoma prywatnymi polami: `location` (typu `String`) oraz `cost` (typu `double`).
- Zaimplementuj konstruktor, który przyjmuje dwa argumenty odpowiadające polom klasy, przy czym parametry konstruktora mają identyczne nazwy jak pola klasy.
- Dodaj publiczne metody dostępowe (gettery i settery) dla obu pól.
- Wprowadź walidację dla pola `cost`, aby zapobiec przypisaniu wartości ujemnych. W przypadku próby ustawienia ujemnej wartości w konstruktorze, ustaw wartość `15.50` jako domyślną. Setter dla `cost` powinien ignorować ustawienie wartości, jeśli jest ujemna.
- Stwórz klasę `TestVacation` z metodą `main`. Wywołaj tu każdą z metod zdefiniowaną wyżej co najmniej jeden raz.

Zadanie należy umieścić we własnym repozytorium na Githubie.

Wariant 503

Wykonaj poniższe czynności:

- Utwórz pakiet `com.adventure.akt` i umieść w nim w oddzielnych plikach poniższe klasy.
- Zaprojektuj klasę `Excursion` z prywatnymi polami: `spot` (typu `String`) oraz `budget` (typu `double`).
- Zaimplementuj konstruktor, który przyjmuje dwa argumenty zgodne z polami klasy, zachowując jednakowość nazw pól i parametrów konstruktora.
- Dodaj publiczne metody dostępowe (gettery i settery) dla obu pól klasy.
- Dodaj mechanizm walidacji dla pola `budget`, aby nie można było przypisać wartości mniejszej niż zero. W przypadku podania ujemnej wartości w konstruktorze, niech konstruktor ustawi `budget` na wartość `20.00`. Setter dla `budget` powinien odrzucać negatywne wartości i nie zmieniać stanu obiektu.
- Stwórz klasę `TestExcursion` z metodą `main`. Wywołaj tu każdą z metod zdefiniowaną wyżej co najmniej jeden raz.

Zadanie należy umieścić we własnym repozytorium na Githubie.