Podstawy programowania obiektowego

Zadanie

Napisz definicję prostej klasy o nazwie `Osoba`. Niech klasa zawiera następujące pola prywatne:

- `_imie` typu reprezentującego ciąg znaków,
- `_nazwisko` typu reprezentującego ciąg znaków,
- `_dataUrodzenia` typu reprezentującego datę,
- `_plec` typu wyliczeniowego `Plec` mogącego przyjmować wartości ze zbioru: *{niezdefiniowana, mężczyzna, kobieta}*.

Dla każdego z pól pola utwórz publiczne metody dostępowe lub właściwości (Properties). Podczas tworzenia osoby należy sprawdzić, czy imię oraz nazwisko są postaci:

- imię musi rozpoczynać się wielką literą, a następnie może zawierać jedynie małe litery. Przykład poprawnych imion: *Marian*, *Xxxx*, *A*.
- nazwisko, podobnie jak imię, rozpoczyna się wielką literą i składa się następnie z małych liter, lecz może mieć formę k-członową, rozdzieloną myślnikami. Przykłady poprawnych nazwisk: Marian, Xxxx, A, Pawlikowska-Jasnorzewska, Marian-Xxxx-A.

Należy również sprawdzić, czy podana data urodzenia nie jest datą z przyszłości. Utwórz konstruktor domyślny oraz konstruktor parametryczny, który pozwoli tworzyć obiekty tej klasy wraz z podaniem imienia, nazwiska, płci oraz daty urodzenia.

Zaimplementuj mechanizm posiadania przez osoby listę dzieci oraz jednego małżonka. Zdefiniuj w tym celu odpowiednie pola oraz metody: `DodajDziecko`, `UstawMalzonka`, które jako parametr przyjmą referencję do obiektu typu `Osoba`. Podczas ustawienia małżonka należy zweryfikować odmienność płci, wyeliminować możliwość bigamii oraz zadbać o to, by połączenie było dwukierunkowe.

Zaimplementuj metody: `WypiszDzieci` oraz `WypiszDrzewoGenealogiczne`. Działanie tej drugiej wypisuje w sposób estetyczny na ekran imię i nazwisko osoby, imię i nazwisko małżonka oraz listę dzieci, dla których rekursywnie wypisuje to samo – imię i nazwisko potomka, współmałżonka oraz listę jego dzieci.

W funkcji Main oprogramuj funkcjonalność, w ramach której przedstawisz możliwości opracowanej klasy.