

Zajęcia 2. Klasy pochodne Informatyk, Lekarz, Nauczyciel

3.1. Wstęp

Obiekty typu **Pracownik** są instancjami ogólnymi (w pewnym sensie abstrakcyjnymi) z niewyszczególnionymi cechami typowymi dla konkretnych pracowników. Opisana w instrukcji rozbudowa projektu umożliwi zdefiniowanie bardziej precyzyjnych obiektów. Będą one reprezentowane przez typy: **Informatyk**, **Lekarz** oraz **Nauczyciel**, pochodne po klasie **Pracownik**, które potencjalnie będzie można dodawać, usuwać, sortować, wyszukiwać w ramach jakiejś kolekcji. Aby udostępnić taką funkcjonalność aplikacji, konieczne będzie poszerzenie projektu o wyżej wymienione klasy. Muszą one zostać umieszczone w projekcie **AplikacjaPracownicy**.

3.2. Dodanie do projektu klasy Informatyk

Klasa o nazwie **Informatyk** ma być klasą pochodną po klasie **Pracownik**. Dla tej klasy należy zdefiniować następujące składowe:

- Prywatne pola:
 - **string** `adresEmail`;
 - **string** `stronaInternetowa`;
- Publiczne właściwości:
 - **AdresEmail** zwracającą i ustawiającą wartość pola `adresEmail`.
 - **StronaInternetowa** zwracającą i ustawiającą wartość pola `stronaInternetowa`.
- Publiczne metody:
 - bezparametrowy konstruktor inicjalizujący pola składowe referencjami **null**. W konstruktorze, na liście inicjalizacyjnej należy wywołać bezargumentowy konstruktor klasy bazowej (słowo kluczowe **base**).
 - konstruktor parametryczny inicjalizujący wszystkie pola składowe na podstawie argumentów:
Informatyk(string imie, string nazwisko, int dzien, string miesiac, int rok, string ulica, string numerDomu, string miasto, string adresEmail, string stronaInternetowa)

W konstruktorze, na liście inicjalizacyjnej należy wywołać konstruktor parametryczny klasy bazowej **Pracownik**; W ciele konstruktora klasy należy zainicjalizować pola składowe obiektu typu **Informatyk**.

- konstruktor kopiujący inicjalizujący wszystkie pola składowe na podstawie argumentu wzorcowego:
Informatyk(Informatyk wzor)
Uwaga: W każdym konstruktorze, na liście inicjalizacyjnej, należy wywołać odpowiedni konstruktor klasy bazowej w celu ustawienia niedostępnych pól klasy bazowej.
- **override Pracownik Clone()**, zwracającą instancję nowostworzonego obiektu typu **Informatyk** na wzór danego (wywołanie konstruktora kopiującego).
- **override string ToString()** zwracającą łańcuch opisujący dane pracownika w formie: imię nazwisko dzień miesiąc rok ulica numer domu miasto adres email strona internetowa.
Wskazówka: najlepiej skorzystać tu z funkcji **String.Format** przesyłając jako argument: **"{0} {1} {2}"**, **base.ToString()**, `adresEmail`, `stronaInternetowa`
- **override string FormatWyjsciowy()** zwracającą łańcuch opisujący dane pracownika w formie:
Imię i nazwisko: imię nazwisko
Data urodzenia: dzień miesiąc rok
Adres zamieszkania: ulica numer domu miasto
Dane dodatkowe: adres email strona internetowa
Definicja tej metody wymaga odpowiedniego sformatowania informacji oraz, ze względu na odniesienie się do danych z klasy bazowej, wywołania metody **FormatWyjsciowy** z klasy **Pracownik**. Implementacja zaprezentowana jest na Listingu 3.1.

```
public override string FormatWyjscowy()
{
    return String.Format("{0}\nDane dodatkowe: {1}, {2}",
        base.FormatWyjscowy(), adresEmail, stronaInternetowa);
}
```

Listing 3.1: Definicja metody formatującej dane informatyka do wymaganej postaci.

- **override void OdczytConsole()**, której zadaniem jest wczytanie z klawiatury wszystkich danych dla informatyka. Obiekt typu **Informatyk** przechowuje również dane zawarte w typie **Pracownik** (odnoszące się do imienia, nazwiska, daty urodzenia i adresu zamieszkania) w związku z tym w definicji metody konieczne będzie wywołanie metody **OdczytConsole** z klasy bazowej. Metoda **OdczytConsole** zdefiniowana dla klasy **Informatyk** przesłania tę samą metodę w klasie **Pracownik** co umożliwia zastosowanie polimorfizmu.
- **override void ZapisConsole()**, której zadaniem jest wypisanie na ekran wszystkich danych informatyka. W przypadku definicji tej metody ma miejsce taka sama sytuacja jak podczas odczytu danych z konsoli: w trakcie wypisania na ekran konieczne jest uwzględnienie informacji o danych podstawowych informatyka (wywołanie metody **OdczytConsole** z klasy **Pracownik**.)

3.3. Dodanie do projektu klasy **Lekarz**

Klasa o nazwie **Lekarz** ma być klasą pochodną po klasie **Pracownik**. Dla tej klasy należy zdefiniować następujące składowe:

- Prywatne pola:
 - **string specjalizacja;**
 - **string tytul;**
- Publiczne właściwości:
 - **Specjalizacja** zwracającą i ustawiającą wartość pola **specjalizacja**.
 - **Tytul** zwracającą i ustawiającą wartość pola **tytul**.
- Publiczne metody:
 - bezparametrowy konstruktor inicjalizujący pola składowe zerami. W konstruktorze, na liście inicjalizacyjnej należy wywołać bezargumentowy konstruktor klasy bazowej.
 - konstruktor parametryczny inicjalizujący wszystkie pola składowe na podstawie argumentów.
Wskazówka: patrz konstruktor parametryczny dla klasy **Informatyk**.
 - konstruktor kopiujący inicjalizujący wszystkie pola składowe na podstawie argumentu wzorcowego typu **Lekarz**. Wskazówka: patrz konstruktor kopiujący dla klasy **Informatyk**.
Uwaga: W każdym konstruktorze, na liście inicjalizacyjnej, należy wywołać odpowiedni konstruktor klasy bazowej w celu ustawienia niedostępnych pól klasy bazowej.
 - **override Pracownik Clone()**, zwracającą instancję nowostworzonego obiektu typu **Lekarz** na wzór danego (wywołanie konstruktora kopiującego).
 - **override string ToString()** zwracającą łańcuch opisujący dane pracownika w formie: imię nazwisko dzień miesiąc rok ulica numer-domu miasto specjalizacja tytuł.
Wskazówka: patrz metoda **ToString** dla klasy **Informatyk**.
 - **override string FormatWyjscowy()** zwracającą łańcuch opisujący dane pracownika w formie:
Imię i nazwisko: imię nazwisko
Data urodzenia: dzień miesiąc rok
Adres zamieszkania: ulica numer domu miasto
Dane dodatkowe: specjalizacja tytuł
 - **override void OdczytConsole()**, której zadaniem jest wczytanie z klawiatury wszystkich danych dla lekarza. Uwaga: W przypadku pobrania z klawiatury informacji o lekarzu, należy uwzględnić jego dane podstawowe (podpunkt 3.3).

- **override void ZapisConsole()**, której zadaniem jest wypisanie na ekran wszystkich danych lekarza. Uwaga: W przypadku wyświetlenia informacji o lekarzu, należy uwzględnić jego dane podstawowe (podpunkt 3.3).

3.4. Dodanie do projektu klasy **Nauczyciel**

Klasa o nazwie **Nauczyciel** ma być klasą pochodną po klasie **Pracownik**. Dla tej klasy należy zdefiniować następujące składowe:

- Prywatne pola:
 - **string przedmiot;**
 - **string tytul;**
- Publiczne właściwości:
 - **Przedmiot** zwracającą i ustawiającą wartość pola **przedmiot**.
 - **Tytul** zwracającą i ustawiającą wartość pola **tytul**.
- Publiczne metody:
 - bezparametrowy konstruktor inicjalizujący pola składowe zerami. W konstruktorze, na liście inicjalizacyjnej należy wywołać bezargumentowy konstruktor klasy bazowej.
 - konstruktor parametryczny inicjalizujący wszystkie pola składowe na podstawie argumentów.
Wskazówka: patrz konstruktor parametryczny dla klasy **Informatyk**.
 - konstruktor kopiujący inicjalizujący wszystkie pola składowe na podstawie argumentu wzorcowego typu **Nauczyciel**. Wskazówka: patrz konstruktor kopiujący dla klasy **Informatyk**.
Uwaga: W każdym konstruktorze, na liście inicjalizacyjnej, należy wywołać odpowiedni konstruktor klasy bazowej w celu ustawienia niedostępnych pól klasy bazowej.
 - **override Pracownik Clone()**, zwracającą instancję nowostworzonego obiektu typu **Nauczyciel** na wzór danego (wywołanie konstruktora kopiującego).
 - **override string ToString()** zwracającą łańcuch opisujący dane pracownika w formie: imię nazwisko dzień miesiąc rok ulica numer domu miasto przedmiot tytuł.
Wskazówka: patrz metoda **ToString** dla klasy **Informatyk**.
 - **override string FormatWyjsciowy()** zwracającą łańcuch opisujący dane pracownika w formie:
Imię i nazwisko: imię nazwisko
Data urodzenia: dzień miesiąc rok
Adres zamieszkania: ulica numer domu miasto
Dane dodatkowe: przedmiot tytuł
 - **override void OdczytConsole()**, której zadaniem jest wczytanie z klawiatury wszystkich danych dla nauczyciela. Uwaga: W przypadku pobrania z klawiatury informacji o nauczycielu, należy uwzględnić jego dane podstawowe (podpunkt 3.3, 3.4).
 - **override void ZapisConsole()**, której zadaniem jest wypisanie na ekran wszystkich danych nauczyciela. Uwaga: W przypadku wyświetlenia informacji o nauczycielu, należy uwzględnić jego dane podstawowe (podpunkt 3.3, 3.4).

3.5. Metoda Main

W metodzie głównej **Main** klasy **Program** należy zdefiniować obiekt klas **Informatyk**, **Lekarz** i **Nauczyciel** i oraz wywołać wybrane metody na rzecz tych obiektów.