Programowanie komputerów.
Ćwiczenia 7 i 8.

TYPY I METODY GENERYCZNE, DELEGATY, WYRAŻENIA LAMBDA

Przed przystąpieniem do zadań należy przeanalizować treść wykładu na temat typów generycznych, delegatów i wyrażeń lambda oraz z rozdziałem 4. podręcznika *„Programowanie komputerów w zadaniach. Język C#”*.

Zadanie 1.

Napisz program z klasą generyczną **Tablica<T>** służącą do przechowywania danych w tablicy jednowymiarowej o dynamicznym rozmiarze. Klasa powinna udostępnić metodę odczytującą wskazany element z tablicy (na podstawie indeksu) oraz metodę wpisującą element do tablicy. Przykładowe wywołanie metody wpisującej: `tab.Wpisz(0, new Osoba("Jan", "Kowalski", 34));`, gdzie **tab** to obiekt zdefiniowanej klasy generycznej, pierwszy argument metody **Wpisz** to indeks w tablicy, a drugi to obiekt typu **T**. W przypadku wpisania indeksu większego niż na to pozwala dotychczasowy rozmiar tablicy należy zwiększyć jej rozmiar (można wykorzystać w tym celu metodę `Resize<T>` z klasy **Array**). Przetestuj stworzoną klasę dla danych dwóch różnych typów.

A następnie dodaj metodę wyświetlającą na konsoli wykaz obiektów tablicy z uwzględnieniem preferencji użytkownika klasy dotyczących formatowania pojedynczego wiersza opisującego cały obiekt. Wykorzystaj w tym celu delegat.

Zadanie 2.

a)

Napisz program z metodą generyczną służącą do tasowania tablicy obiektów. Tasowanie (mieszanie) jest operacją odwrotną względem sortowania, polega na losowej zamianie miejscami elementów danej kolekcji. Przetestuj metodę na dwóch tablicach, jednej typu `int`, a drugiej typu `Karta`. Zdefiniuj klasę **Karta** zawierającą dwa pola o odpowiednich typach: `kolorKarty` (trefl, pik, karo, kier) oraz `nazwaKarty` (np. walet, dama, król, as, dwójka, trójka, czwórka, itd.).

*W zadaniu możesz wykorzystać prosty algorytm do tasowania znany pod nazwą **Fisher-Yates shuffle**.*

b)

Zmodyfikuj program z poprzedniego zadania poprzez wywołanie metody `Array.ForEach(T[], Action<T>)` w celu wyświetlenia zawartości tablicy przed tasowaniem i po tasowaniu. Metoda ta dla podanej tablicy typu `T` wykonuje akcję zdefiniowaną poprzez delegata `Action<T>`. Do definicji akcji użyj **wyrażenia lambda**.

Zadanie 3.

Napisz program z metodą generyczną służącą do sortowania tablicy jednowymiarowej mającej dane wbudowanego typu liczbowego (np. `int`, `double`). Sortowanie wykonaj samodzielnie, stosując algorytm sortowania bąbelkowego

Zadanie 4.

Napisz program z metodą generyczną służącą do sortowania tablicy obiektów dla dowolnej. Podobnie jak w poprzednim zadaniu, sortowanie wykonaj samodzielnie, stosując algorytm sortowania bąbelkowego. Zdefiniuj delegat wskazujący na metodę służącą do porównywania obiektów (według wybranego pola klasy lub jej właściwości).

Zadanie 5.

Napisz program z klasą generyczną dla typów dziedziczących po klasie abstrakcyjnej **`Osoba`** (wykorzystaj w tym celu stosowne ograniczenie typu podczas definicji klasy generycznej). Program ma służyć do formatowania tekstu z adresami osób, jaki docelowo można by drukować na naklejkach (przyklejanych na kopertach lub paczkach). Klasa **`Osoba`** ma posiadać stosowne pola z danymi potrzebnymi do napisania adresu, takie jak imię, nazwisko, kod pocztowy, miasto, ulica, numer domu, numer mieszkania. Stwórz w programie dwie klasy dziedziczące po klasie **`Osoba`**, np. **`Klient`** i **`Pracownik`**, a następnie przetestuj program wyświetlając sformatowane adresy na ekranie dla kilku różnych obiektów umieszczonych w tablicy.

Zadanie 6.

Napisz program z klasą generyczną dla typów referencyjnych do generowania opisu obiektów (zapisanych w tablicy obiektów) według formatu:
`nazwa właściwości: wartość`

i tak w kolejnych wierszach należy umieścić wykaz wszystkich publicznych właściwości danego typu. Poszczególne obiekty należy oddzielić poziomą linią zbudowaną ze znaku minus. Przetestuj działanie zdefiniowanej klasy generycznej dla dowolnej klasy.

Wskazówki: Do pobrania informacji o publicznych właściwościach danego typu `T` (wystarczy dostęp publiczny do akcesora **`get`**) wykorzystaj mechanizm refleksji. **Mechanizm refleksji** pozwala zarządzać kodem tak, jakby był danymi. Jednym z ważniejszych zastosowań tego mechanizmu jest pobranie informacji o typie. W programie użyj tego mechanizmu do pobrania informacji o właściwościach danego typu `T`. Zdefiniuj w tym celu tablicę typu **`PropertyInfo`** z przestrzeni nazw **`System.Reflection`** i wykorzystaj metodę **`GetProperties`** z klasy **`Type`**.

Przykład działania programu z użyciem zdefiniowanej klasy generycznej dla tablicy obiektów klasy Towar z dwoma elementami.