LABORATORIUM 6

TREŚCI KSZTAŁCENIA:

- F# - PRACA Z DANYMI TEKSTOWYMI

Spis treści

F# - praca z danymi tekstowymi

W F# praca z danymi tekstowymi jest dość wygodna dzięki funkcjom i operatorom wbudowanym oraz funkcjom dostępnym w przestrzeni nazw System. Oto przegląd różnych podejść i narzędzi, które można wykorzystać do pracy z danymi tekstowymi:

Podstawowe operacje na stringach

F# obsługuje ciągi znaków (string) w sposób podobny do innych języków .NET, takich jak C#:

1. Tworzenie i wyświetlanie tekstu:

```
let text = "Hello, World!"
printfn "%s" text
```

2. Konkatenacja ciągów:

```
let firstName = "John"
let lastName = "Doe"
let fullName = firstName + " " + lastName
```

3. Interpolacja ciągów (od F# 5.0):

```
let firstName = "John"
let lastName = "Doe"
let fullName = firstName + " " + lastName

let age = 30
let message = $"Witaj, {firstName} {lastName}. Masz {age} lat."
```

Podział i łączenie ciągów

F# pozwala na łatwe dzielenie i łączenie ciągów przy użyciu metod z System.String.

```
// Podział ciągu na tablicę:
let sentence = "F# jest świetnym językiem."
let words = sentence.Split(' ')  // ["F#", "jest", "świetnym", "językiem."]

//Łączenie tablicy w ciąg:
open System
let wordsArray = [| "Hello"; "World"; "from"; "F#" |]
let combined = String.Join(" ", wordsArray)  // "Hello World from F#"
```

Funkcje manipulujące tekstem

```
// Zmiana wielkości liter:
let lower = "HELLO".ToLower()  // "hello"
let upper = "world".ToUpper()  // "WORLD"

//Przycinanie białych znaków:
let trimmed = "   F#   ".Trim()  // "F#"
```

```
//Zamiana fragmentu tekstu:
let replaced = "Hello F#".Replace("F#", "World")  // "Hello World"
```

Regularne wyrażenia

Do pracy z bardziej złożonymi operacjami na tekstach można używać regularnych wyrażeń za pomocą klasy System.Text.RegularExpressions.Regex.

Wyszukiwanie wzorca:

```
open System.Text.RegularExpressions

let pattern = @"\d+"  // Wyszukuje liczby
let input = "F# 5.0 jest szybszy od F# 4.7"
let matches = Regex.Matches(input, pattern)

for m in matches do
    printfn "Znaleziono: %s" m.Value
```

Praca z funkcjami pomocniczymi F#

F# oferuje zestaw funkcji String oraz operatorów, które ułatwiają pracę z tekstami:

```
open System
//Filtrowanie i mapowanie znaków:
let onlyDigits = "F#123".ToCharArray() |> Array.filter Char.IsDigit |> String
// Sprawdzanie warunków:
let isNumeric = "12345".ToCharArray() |> Array.forall Char.IsDigit  // true
```

Wielowierszowe ciągi znaków

W F# możesz tworzyć ciągi wielowierszowe przy użyciu potrójnych cudzysłowów """:

```
let multiLineString = """
To jest wielowierszowy
ciąg znaków w F#.
"""
```

Wczytywanie danych tekstowych z plików

Do wczytywania danych z plików tekstowych można używać funkcji System.IO:

```
open System.IO

let lines = File.ReadAllLines("path/to/file.txt")
lines |> Array.iter (printfn "%s")
```

Przetwarzanie tekstu przy użyciu funkcji wyższego rzędu

Można używać funkcji takich jak List.map, List.filter itp. do przetwarzania kolekcji tekstów:

```
open System

let texts = ["abc"; "def"; "ghi"]
```

```
let upperTexts = texts |> List.map (fun text -> text.ToUpper())  // ["ABC";
"DEF"; "GHI"]

// Wyświetlenie przekształconych tekstów
upperTexts |> List.iter (printfn "%s")
```

Zadania do samodzielnego rozwiązania

Rozwiązania w postaci niezbędnych plików źródłowych należy przesłać do utworzonego zadania na platformie e-learningowej zgodnie ze zdefiniowanymi instrukcjami oraz w nieprzekraczalnym wyznaczonym terminie.

Zadanie 1. Liczba słów i znaków

Napisz program, który pobiera tekst od użytkownika, a następnie oblicza i wyświetla:

- Liczbę słów w podanym tekście.
- Liczbę znaków (bez spacji).

Zadanie 2. Sprawdzenie palindromu

Stwórz funkcję, która sprawdza, czy podany ciąg znaków jest palindromem (czytany od przodu i od tyłu jest taki sam). Program powinien pobierać tekst od użytkownika i wyświetlać odpowiedni komunikat.

Zadanie 3: Usuwanie duplikatów

Napisz funkcję, która przyjmuje listę ciągów (np. słów) i zwraca nową listę, w której usunięte są duplikaty. Użytkownik powinien mieć możliwość wprowadzenia słów oddzielonych spacjami, a program powinien wyświetlić listę unikalnych słów.

Zadanie 4: Zmiana formatu tekstu

Stwórz program, który przyjmuje tekst w formacie "imię; nazwisko; wiek" i przekształca go na format "Nazwisko, Imię (wiek lat)". Użytkownik powinien móc wprowadzić dowolną liczbę takich wpisów, a program powinien je przetworzyć i wyświetlić w nowym formacie.

Zadanie 5: Znajdowanie najdłuższego słowa

Stwórz funkcję, która przyjmuje tekst od użytkownika i zwraca najdłuższe słowo. Program powinien także wyświetlać długość najdłuższego słowa.

Zadanie 6. Wyszukiwanie i zamiana

Stwórz program, który umożliwia użytkownikowi wyszukiwanie określonego słowa w tekście i zamianę go na inne. Program powinien wyświetlić zmodyfikowany tekst.