**Доклад**

**по**

**Функционално програмиране**

**Тема: Абстракции**

1. **Абстракции чрез функции**
2. **Изчисления върху списъци**
3. **Източници**

Изготвил: Ваня Ванева 12а

1. **Абстракции чрез функции**

В функционалното програмиране, функциите се използват като основен инструмент за абстракция. Те могат да бъдат представени като първокласни граждани, което означава, че могат да бъдат предавани като аргументи на други функции, върнати като резултат от други функции и съхранявани в променливи.

Примерен код:

**public class FunctionalAbstraction**

**{**

**// Функция, която удвоява дадено число**

**public static int Double(int x)**

**{**

**return x \* 2;**

**}**

**// Функция, която прилага дадена функция върху всеки елемент на масив**

**public static int[] ApplyFunction(int[] array, Func<int, int> function)**

**{**

**int[] result = new int[array.Length];**

**for (int i = 0; i < array.Length; i++)**

**{**

**result[i] = function(array[i]);**

**}**

**return result;**

**}**

**}**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**int[] numbers = { 1, 2, 3, 4, 5 };**

**// Прилагане на абстракция чрез функции**

**int[] doubledNumbers = FunctionalAbstraction.ApplyFunction(numbers, FunctionalAbstraction.Double);**

**Console.WriteLine("Doubled numbers:");**

**foreach (int num in doubledNumbers)**

**{**

**Console.WriteLine(num);**

**}**

**}**

**}**

1. **Изчисления върху списъци**

В функционалното програмиране списъците са често използвана структура за съхранение на данни. Изчисленията върху списъци са основен начин за манипулиране и обработка на данни.

Примерен код:

**using System.Collections.Generic;**

**using System.Linq;**

**public class ListCalculations**

**{**

**// Функция, която събира всички числа в даден списък**

**public static int Sum(List<int> numbers)**

**{**

**Func<List<int>, int> sumFunc = (nums) => nums.Sum();**

**return sumFunc(numbers);**

**}**

**// Функция, която намира средното аритметично на числата в даден списък**

**public static double Average(List<int> numbers)**

**{**

**Func<List<int>, double> averageFunc = (nums) => nums.Average();**

**return averageFunc(numbers);**

**}**

**}**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**List<int> numbers = new List<int> { 1, 2, 3, 4, 5 };**

**// Изчисления върху списъци**

**int sum = ListCalculations.Sum(numbers);**

**double average = ListCalculations.Average(numbers);**

**Console.WriteLine("Sum: " + sum);**

**Console.WriteLine("Average: " + average);**

**}**

**}**

1. **Източници**

<https://elhacker.info/manuales/Lenguajes%20de%20Programacion/Haskell/Thinking%20Functionally%20With%20Haskell.pdf>

<https://www.tutorchase.com/answers/a-level/computer-science/how-are-lists-implemented-and-used-in-functional-programming-languages>

<https://www.sciencedirect.com/topics/computer-science/functional-programming>

<https://en.wikipedia.org/wiki/First-class_function>

<https://medium.com/@rabailzaheer/first-class-and-higher-order-functions-86d14e40c688>

<https://en.wikipedia.org/wiki/Functional_programming>