МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «ПОЛОЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ ЕВФРОСИНИИ ПОЛОЦКОЙ»

Факультет информационных технологий

Кафедра технологий программирования

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине «Функциональное программирование»

|  |  |
| --- | --- |
| Выполнил: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.Ю. Страпко |
| Проверил: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_О.Н. Забелендик |

|  |
| --- |
|  |

Введение

Целью данной лабораторной работы является изучение основных концепций функционального программирования на примере языка Haskell. В процессе выполнения работы были рассмотрены ключевые аспекты программирования, такие как рекурсия, сопоставление с образцом, кусочное задание функций и работа со списками. Эти механизмы являются фундаментальными для функциональных языков программирования и позволяют эффективно решать задачи обработки данных.

Особое внимание уделялось реализации рекурсивных функций, которые заменяют циклы в императивных языках программирования. Также была изучена система выравнивания кода в Haskell, которая позволяет обходиться без специальных символов группировки и разделения операторов. Это делает код более читаемым и структурированным.

В рамках лабораторной работы были выполнены практические задания, направленные на закрепление навыков написания функций для обработки списков, чисел и строк. Каждая задача требовала применения теоретических знаний о рекурсии, сопоставлении с образцом и других особенностях языка Haskell.

1 ПРАТИЧЕСКАЯ ЧАСТЬ

В первом задании определить функцию, которая выводит список четных чисел в количестве n-раз в листинге 1.

Листинг 1 – Первое задание

1. evenNaturals :: Int -> [Int]
2. evenNaturals n = take n [2, 4 ..]

Во втором задании требуется определить функцию, которая выводит список пирамидальных чисел n-раз. Функция представлена в листинге 2.

Листинг 2 – Второе задание

1. pyramidalNumbers :: Int -> [Int]
2. pyramidalNumbers n = map (\k -> k \* (k + 1) \* (k + 2) `div` 6)
3. [1..n]

В третьем задании надо было определить функцию, которая складывает два списка, функция представлена в листинге 3.

Листинг 3 – Третье задание

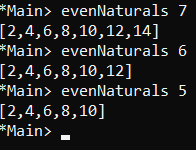
1. addLists :: [Int] -> [Int] -> [Int]
2. addLists [] ys = ys
3. addLists xs [] = xs
4. addLists (x:xs) (y:ys) = (x + y) : addLists xs ys

В третьем задании надо было определить функцию, которая заменяет символ из строки на другой символ, функция представлена в листинге 4.

Листинг 4 – Четвёртое задание

1. addLists :: [Int] -> [Int] -> [Int]
2. addLists [] ys = ys
3. addLists xs [] = xs
4. addLists (x:xs) (y:ys) = (x + y) : addLists xs ys

1. **РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ РАБОТЫ**



**Рисунок 1 – первое задание**

****

**Рисунок 2 – второе задание**

****

**Рисунок 3 – третье задание**

****

**Рисунок 4 – четвертое задание**

## Вывод

В ходе выполнения лабораторной работы были изучены основы функционального программирования на языке Haskell. Особое внимание было уделено работе с интерпретатором Hugs, основным типам данных, использованию кортежей и списков, а также определению пользовательских функций.

Практическое выполнение заданий позволило закрепить теоретические знания о типах данных, арифметических операциях, кортежах, списках и функциях. Были рассмотрены различные способы объявления и вызова функций, применение условных выражений и обработка списков.

Лабораторная работа помогла развить навыки работы с функциональным программированием и познакомиться с ключевыми особенностями языка Haskell. Благодаря строгой типизации и мощной системе работы с данными, Haskell обеспечивает безопасность кода и упрощает выявление ошибок на этапе компиляции.