МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение
высшего образования «Санкт-Петербургский политехнический
университет Петра Великого»

Институт компьютерных наук и кибербезопасности

Высшая школа технологий искусственного интеллекта

Направление: 02.03.01 «Математика и компьютерные науки»

Отчет по лабораторной работе №4
по дисциплине
«Функциональное программирование»
Вариант 20

Студент,
группы 5130201/20102 _____ Тищенко А. А.

Преподаватель,
к. т. н., доц. _____ Моторин Д. Е.

«_____»_____ 2024г.

Санкт-Петербург, 2024

Содержание

1 Постановка задачи

Для выполнения лабораторной работы необходимо было сделать следующее. Создать проект в `stack`. Функции записать в библиотеку `Lib.hs` и ограничить доступ к вспомогательным функциям. Тесты записать в `Spec.hs`.

Функция 1: Напишите функцию проверки равенства по модулю `isCongruent :: Int -> Int -> Int -> Bool`, которая проверяет, равны ли два числа по модулю третьего числа. Используя `QuickCheck`, проверьте следующие свойства:

1. Если два числа равны по модулю, то их разность делится на модуль: `isCongruent a b m == (modulus (a - b) m == 0)`.
2. Равенство по модулю является симметричным: `isCongruent a b m == isCongruent b a m`.
3. Если одно число равно другому, то они равны по любому модулю: если `a == b`, то `isCongruent a b m == True`.

Функция 2: Напишите функцию `filterByPredicate :: (a -> Bool) -> [a] -> [a]`, которая фильтрует элементы списка по заданному предикату. Используя `QuickCheck`, проверьте следующие свойства:

1. Все элементы результата удовлетворяют предикату: для каждого элемента `x` из результата должно выполняться условие: `predicate x == True`.
2. Длина результата не превышает длину исходного списка: `length (filterByPredicate predicate xs) <= length xs`.
3. Если предикат всегда возвращает `True`, то результат совпадает с исходным списком: если `predicate x == True` для всех `x`, то `filterByPredicate predicate xs == xs`.

2 Особенности реализации

2.1 Функция isCongruent

Код функции для проверки числовой конгруэнтности представлен в листинге 1. Функция `isCongruent` принимает три числа: `a`, `b` и `d`. Она возвращает `True`, если остатки от деления `a` и `b` на `d` равны, и `False` в противном случае.

Листинг 1. Функция для проверки числовой конгруэнтности.

```
1 isCongruent :: Int -> Int -> Int -> Bool
2 isCongruent a b d = a `mod` d == b `mod` d
```

2.2 Функция filterByPredicate

Код функции для фильтрации элементов списка на основе предиката представлен в листинге 2. Функция `filterByPredicate` принимает предикат (`predicate`) и список (`[a]`). Она возвращает новый список, содержащий только те элементы исходного списка, для которых предикат возвращает `True`.

Листинг 2. Функция для фильтрации списка по предикату.

```
1 filterByPredicate :: (a -> Bool) -> [a] -> [a]
2 filterByPredicate _ [] = []
3 filterByPredicate predicate (x:xs)
4     | predicate x = x : filteredTail
5     | otherwise = filteredTail
6     where
7         filteredTail = filterByPredicate predicate xs
```

2.3 Тесты для функции isCongruent

Для тестирования функции `isCongruent` использовались свойства, проверяемые с использованием библиотеки QuickCheck. Ниже приведены описания тестов, представленных в листинге 3.

- `propCongruentDifference`: Проверяет, что два числа конгруэнтны по модулю `m`, если и только если разность этих чисел делится на `m` без остатка.
- `propCongruentSymmetric`: Проверяет симметричность конгруэнтности, то есть, если `a` конгруэнтно `b`, то `b` конгруэнтно `a`.
- `propCongruentEqualNumbers`: Проверяет, что любое число `a` всегда конгруэнтно самому себе по любому ненулевому модулю.

Листинг 3. Тесты для функции `isCongruent` с использованием QuickCheck.

```
1 propCongruentDifference :: Int -> Int -> Int -> Property
2 propCongruentDifference a b m =
```

```
3   m /= 0 ==> isCongruent a b m == ((a - b) `mod` m == 0)
4
5 propCongruentSymmetric :: Int -> Int -> Int -> Property
6 propCongruentSymmetric a b m =
7   m /= 0 ==> isCongruent a b m == isCongruent b a m
8
9 propCongruentEqualNumbers :: Int -> Int -> Property
10 propCongruentEqualNumbers a m =
11   m /= 0 ==> isCongruent a a m == True
```

2.4 Тесты для функции filterByPredicate

Для тестирования функции `filterByPredicate` использовались свойства, проверяемые с использованием библиотеки QuickCheck. Описание тестов приведено в листинге 4.

- `propFilterByPredicateSatisfiesPredicate`: Проверяет, что все элементы результирующего списка удовлетворяют переданному предикату.
- `propFilterByPredicateLength`: Проверяет, что длина результирующего списка не превышает длину исходного списка.
- `propFilterByPredicateAlwaysTrue`: Проверяет, что если предикат всегда возвращает `True`, результирующий список совпадает с исходным.

Листинг 4. Тесты для функции `filterByPredicate` с использованием QuickCheck.

```
1  propFilterByPredicateSatisfiesPredicate :: Fun Int Bool -> [Int] -> Bool
2  propFilterByPredicateSatisfiesPredicate (Fun _ predicate) xs =
3    all predicate (filterByPredicate predicate xs)
4
5  propFilterByPredicateLength :: Fun Int Bool -> [Int] -> Bool
6  propFilterByPredicateLength (Fun _ predicate) xs =
7    length (filterByPredicate predicate xs) <= length xs
8
9  propFilterByPredicateAlwaysTrue :: [Int] -> Bool
10 propFilterByPredicateAlwaysTrue xs =
11   filterByPredicate (\_ -> True) xs == xs
```

2.5 Запуск тестов

Для выполнения всех тестов в проекте создан файл `Spec.hs`, в котором определена функция `main`. Она последовательно запускает все тесты, используя библиотеку `QuickCheck`. Код функции `main` приведён в листинге 5.

Функция `quickCheck` используется для автоматического тестирования свойств. Она генерирует случайные входные данные, проверяет выполнение свойства на каждом из них и сообщает о результатах. В случае провала теста `quickCheck` предоставляет пример данных, на которых свойство не выполняется.

Для сборки проекта и выполнения всех тестов достаточно выполнить команду `stack test`.

Листинг 5. Функция `main` для запуска тестов.

```
1  main :: IO ()
2  main = do
3    quickCheck propCongruentDifference
4    quickCheck propCongruentSymmetric
5    quickCheck propCongruentEqualNumbers
6
7    quickCheck propFilterByPredicateSatisfiesPredicate
8    quickCheck propFilterByPredicateLength
9    quickCheck propFilterByPredicateAlwaysTrue
```

3 Результаты работы программы

```
Примеры работы `isCongruent`
isCongruent 10 12 2: True
isCongruent 10 11 2: False

Пример работы `filterByPredicate`
filterByPredicate (>5) [1, 6, 3, 7, 2]: [6,7]
```

Рис. 1. Результаты работы программы.

Сама программа лишь выводит примеры работы функций `isCongruent` и `filterByPredicate`. Результаты её запуска представлены на Рис. 1.

```
lab4> test (suite: lab4-test)

+++ OK, passed 100 tests; 13 discarded.
+++ OK, passed 100 tests; 14 discarded.
+++ OK, passed 100 tests; 12 discarded.
+++ OK, passed 100 tests.
+++ OK, passed 100 tests.
+++ OK, passed 100 tests.



lab4> Test suite lab4-test passed
```

Рис. 2. Результаты запуска тестов.

Результаты запуска тестов с помощью команды `stack test` представлены на Рис. 2.

Заключение

В данной работе был создан проект с использованием `stack`, включающий разработку и тестирование двух функций: проверки равенства по модулю и фильтрации списка по предикату. Для каждой функции были написаны три теста. Тесты запускаются на сгенерированных данных с помощью `QuickCheck`. Они подтверждают корректность реализации этих функций. В ходе выполнения лабораторной работы были получены базовые знания об автоматическом тестировании программ, написанных на языке Haskell.

Список литературы

[1] Hackage – QuickCheck: Automatic testing of Haskell programs, URL: https://hackage.haskell.org/package/QuickCheck, Дата обращения: 19.11.2024