COMPSCI 1JC3

Introduction to Computational Thinking

Fall 2017

Assignment 2

Dr. William M. Farmer

McMaster University

Revised: October 2, 2017

The purpose of Assignment 2 is to write a module in Haskell that implements a 3-dimensional vector space. The requirements for Assignment 2 and for Assignment 2 Extra Credit are given below. You are required to do Assignment 2, but Assignment 2 Extra Credit is optional. Please submit Assignment 2 as a single `.hs` file to the Assignment 2 folder on Avenue under Assessments/Assignments. If you choose to do Assignment 2 Extra Credit for extra marks, please submit it also as a single `.hs` file to the Assignment 2 Extra Credit folder on Avenue in the same place. Both Assignment 2 and Assignment 2 Extra Credit are due **October 20, 2017 before midnight.** Assignment 2 is worth 3% of your final grade, while Assignment 2 Extra Credit is worth 2 extra percentage points.

Although you are allowed to receive help from the instructional staff and other students, your submitted program must be your own work. Copying will be treated as academic dishonesty!

1 Background

A *vector* is a mathematical entity that has direction and magnitude. A vector can be identified with a point in Euclidean space. A point in 3-dimensional Euclidean space can be represented with *Cartesian coordinates* as a triple (3-tuple) $V = (a, b, c)$ of real numbers where a is the x-coordinate, b is the y-coordinate, and c is the z-coordinate of the point, respectively. (A point in 3-dimensional Euclidean space could also be represented in other ways such as with *polar coordinates*.)

Suppose $V = (a, b, c)$ and $V' = (a', b', c')$ are two vectors represented by points in 3-dimensional Euclidean space. V is the *zero vector* if $a = b = c = 0$. The *scalar product* of a real number r and V is the vector $(r * a, r * b, r * c)$. The *magnitude* of V is the real number $\sqrt{a^2 + b^2 + c^2}$. The *sum* of V and V' is the vector $(a + a', b + b', c + c')$. The *difference* of V and V' is the sum of V and the scalar multiple of -1 and V'. The *distance* between V and V' is the magnitude of the difference of V and V'.

2 Assignment 2

The purpose of this assignment is to create a Haskell module for the vector space of 3-dimensional vectors whose coordinates are of type `Double`.

2.1 Requirements

1. The name of your Haskell file is `Assign_2_`*`YourMacID`*`.hs` where *YourMacID* is your actual MacID.

2. Your name, MacID, the date, and "Assignment 2" are given in comments at the top of your file.

3. The first uncommented line of the file should be

```
module VectorSpace where
```

4. The file contains the type definition

```
type Vector = (Double,Double,Double)
```

5. The file includes a constant named `vecZero` of type `Vector` that implements the zero vector constant.

6. The file includes a function named `vecScalarProd` of type `Double -> Vector -> Vector` that implements the scalar product function.

7. The file includes a function named `vecSum` of type `Vector -> Vector -> Vector` that implements the sum function.

8. The file includes a function named `vecMagnitude` of type `Vector -> Double` that implements the magnitude function.

9. The file includes a function named `vecF` of type `Vector -> [Vector] -> [Double]` such that `vecF x y` equals a list whose ith entry is the distance between `x` and the ith entry of the list `y`.

10. Your file can be imported into GHCi and all of your functions perform correctly.

2.2 Testing

Include in your file a test plan for the functions `vecScalarProd`, `vecSum`, `vecMagnitude`, and `vecF`. The test plan must include at least three test cases for each function. Each test case should have following form:

`Function:` Name of the function being tested.
`Test Case Number:` The number of the test case.
`Input:` Inputs for function.
`Expected Output:` Expected output for the function.
`Actual Output:` Actual output for the function.

The test plan should be at the bottom of your file in a comment region beginning with a {- line and ending with a -} line.

3 Assignment 2 Extra Credit

The purpose of this assignment is to create a Haskell module for vector spaces of n-dimensional vectors (where $n \geq 1$) whose coordinates are of a Floating type.

3.1 Requirements

1. The name of your Haskell file is `Assign_2_ExtraCredit_`*YourMacID*`.hs` where *YourMacID* is your actual MacID.

2. Your name, MacID, the date, and "Assignment 2 Extra Credit" are given in comments at the top of your file.

3. The first uncommented line of the file should be

```
module VectorSpace where
```

4. The file contains the following type definitions and type class definition:

```
newtype Vector2 a = Vector2 (a,a)
    deriving (Show,Eq)
newtype Vector3 a = Vector3 (a,a,a)
    deriving (Show,Eq)
newtype Vector4 a = Vector4 (a,a,a,a)
    deriving (Show,Eq)

class VectorSpace v where
    vecZero         :: (Num a) => v a
    vecSum          :: (Num a) => v a -> v a -> v a
    vecScalarProd   :: (Num a) => a -> v a -> v a
    vecMagnitude    :: (Floating a) => v a -> a
```

5. The file includes instance statements that define `Vector2`, `Vector3`, and `Vector4` to be instances of the type class `VectorSpace`.

6. The file includes a function named `vecF` of type `(Floating a,VectorSpace v) => v a -> [v a] -> [a]` such that `vecF x y` equals a list whose ith entry is the distance between `x` and the ith entry of the list `y`. Use the functions defined in the type class `Vector` to define `vecF`.

7. Your file successfully loads into GHCi and all of your functions perform correctly.

3.2 Testing

Include in your file a test plan (as described above) for the functions `vecScalarProd`, `vecSum`, `vecMagnitude`, and `vecF`. The test plan must include at least three test cases for each function.