Minor Assignment 6

ENGINEER 1D04

Dr. Spencer Smith

McMaster University, Fall 2013

Revised: February 18, 2014

Please use AutoMarker (`automarker.mcmaster.ca`) and Avenue to acquire, test, package and submit your assignment. The procedure for submitting assignments is summarized on Avenue, with additional details provided by AutoMarker. **Please frequently back up your work by creating a submission package in AutoMarker**. This will provide a chance to recover your work in the event of an equipment failure.

Background

A *vector* is a mathematical entity that has direction and magnitude. A two-dimensional vector can be represented by a pair $V = (a, b)$ of real numbers where a and b are the x- and y-coordinates of V. Given a real number r and two vectors $V_1 = (a_1, b_1)$ and $V_2 = (a_2, b_2)$, the *magnitude* of V_1 is the real number $|V_1| = \sqrt{(a_1)^2 + (b_1)^2}$, the *scalar multiple* of r and V_1 is the vector $r * V_1 = (r * a_1, r * b_1)$, and the *sum* of V_1 and V_2 is the vector $V_1 + V_2 = (a_1 + a_2, b_1 + b_2)$. If r_1 and r_2 are real numbers, the *linear combination* of V_1, V_2, r_1, and r_2 is the vector $(r_1 * V_1) + (r_2 * V_2)$.

Overview

The purpose of this assignment is to create a class in Python whose instances are two-dimensional vectors. Design, implement, and test a program that satisfies the requirements below. Submit your program on Avenue to Learn.

Design, implement, and test a program that satisfies the requirements below.

****IMPORTANT!!!**:** This assignment will be run through an automated testing program to be graded. Function syntax in your program must be **exactly** as specified, including spelling, capitalization, and the order of function parameters. Failure to precisely follow the requirements below will result in a **significant loss of marks**.

Requirements

1. The program includes a Python class named `Vector`.

2. The `__init__` method for the class has a first formal parameter `self` and two normal parameters numbers a and b as input and creates

two variables to store them. a and b are the x- and y-coordinates, respectively, of the vector represented by the object created when the constructor of the class is called.

3. The class contains accessors for the x- and y-coordinates of the represented vector. They should be named `getX(self)` and `getY(self)` respectively.

4. The class contains a method named `magnitude` that has a first formal parameter `self` and returns as output the magnitude of the vector represented by `self`.

5. The class contains a method named `scalarMultiply` that has a first formal parameter `self` and a normal parameter number s as input and returns as output a `Vector` object that represents the scalar multiple of s and the vector represented by `self`.

6. The class contains a method named `add` that takes first formal parameter `self` and a normal parameter `Vector` object v as input and returns as output a `Vector` object that represents the sum of the vectors represented by `self` and v.

7. The class contains a method named `linearCombo` that takes a first formal parameter `self`, a normal parameter `Vector` object v, and two normal parameter numbers a and b as input and returns as output the linear combination of the vector represented by `self`, and the vector represented by v, the number a, and the number b.

8. The program requires very little besides the function definitions. There is no `main()`.

9. The program does not read anything from standard input or write anything to standard output. That is, the program does not interact with the user who invokes it.

10. The program is written in Python in a module, NOT in the Python Shell. To create a new module in IDLE, go to File $\rightarrow$ New Window. You must save this file with a .py extension. For more information on submitting your program, click the "AutoMarker Instructions" button above.

11. Your name, MacID, student number, and the date are given in comments at the top of your Python (.py) file before your program.

12. Your answers to the design questions and the test plan (see below) are given in comments at the bottom of your Python (.py) file after your program.

13. Your program MUST have valid Python syntax and it must run without errors. Ensure that your program runs properly by running it before you submit.

14. You must sign out with a TA or IAI after you have submitted your lab at the submission station. Failure to do so could result in a zero.

Design Question

1. Given vectors $V_1 = (a_1, b_1)$ and $V_2 = (a_2, b_2)$, the *difference* of V_1 and V_2 is the vector $V_1 - V_2 = (a_1 - a_2, b_1 - b_2)$. Show how the difference of two `Vector` objects can be created using the methods in the class `Vector`.

2. What is the difference between an accessor and a mutator?

3. Is the method `scalarMultiply` a mutator? Why or why not?

Test Plan

Produce a test plan with test cases for each of the methods in your class.

```
Test:  i for method j
Input:  inputs for method j
Expected Output:  expected output for method j
```

EXAMPLE:

```
Test:  1 for method linearCombo, v1.linearCombo(v2, r1, r2)
Input:  v1 = Vector(2,3), v2 = Vector(5,2), r1 = 2, r2 = 3
Expected Output:  Vector(19,12)
```

You should have enough test cases to adequately support the argument that your code is correct. Your test cases should cover as many different classes of input cases as possible, including boundary cases. Your test plan should include case(s) where your expected output is a failure.