CSCI 201L Assignment #1
Spring 2015
2.5% of course grade

C++ to Java: A programming exercise with mathematical vectors

Overview

This assignment will have you warm up your Java skills. Since some things are different in Java than in C++, this assignment will expose you to some of those differences. You should have a relatively easy time with this assignment – all of the core programming concepts are the same as they were with C++.

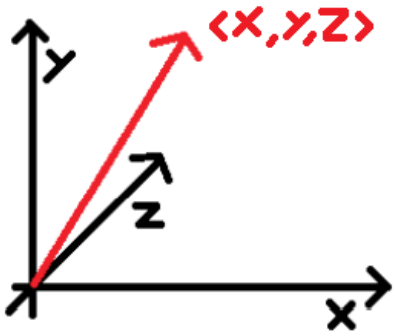
Introduction

First off, don't worry if you are unfamiliar with vectors, all of the math will be provided for you – you will only need to turn the given formulas into code. You will need to create a minimum of three classes: classes that represent something in 3D space, a 3D Point, and a 3D Vector. Once these are created, you will create a simple menu in which a user can manipulate these objects.

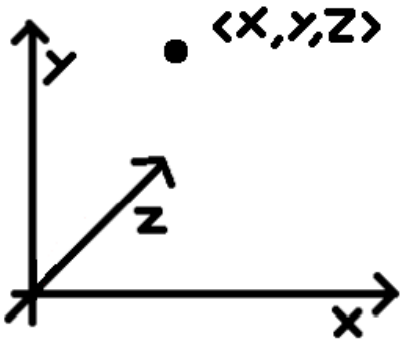
Background

Note: Don't try to memorize the following information. This is simply a reference. Just use the formulas provided (highlighted in yellow) and you'll be fine! Do, however, keep in mind the similarities and differences between the operations.

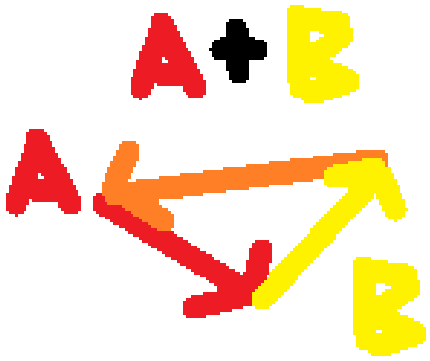
In case you aren't familiar with a Vector in Math, they are a quantity with both a magnitude and direction. We will represent them in a 3D space with three numerical values: <x,y,z>. These numbers indicate the position on the respective axis where the head of the vector lies, with the tail being at the origin: <0,0,0>.



A point in 3D space is simply a point with x,y, and z values.



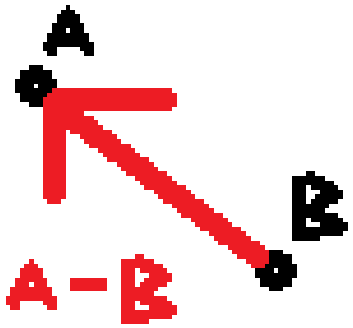
Both Vectors and Points can be represented by three values, <x,y,z> but represent different things.
The addition of vector A and B resulting in Vector C looks like this:
<Cx, Cy, Cz> = <Ax+Bx, Ay+By, Az+Bz>
Or graphically,



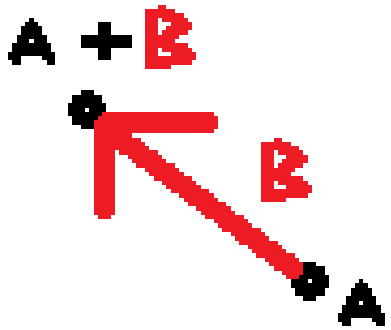
The addition of a point to another point does not make sense. This operation cannot be done.

The subtraction of Vector B from Vector A resulting in Vector C looks like this:
<Cx, Cy, Cz> = <Ax-Bx, Ay-By, Az-Bz>
Or graphically,

The subtraction of Point A from Point B results in a Vector C with the length of the distance from B to A, with the head at A and tail at B:
<Cx,Cy,Cz> = <Ax-Bx, Ay-By, Az-Bz>
Or graphically,

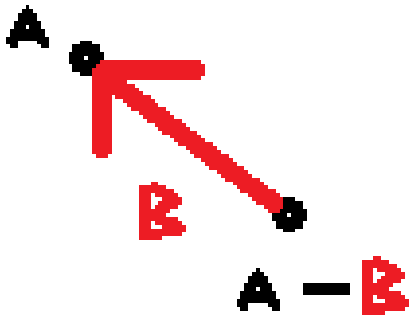


The addition of Vector B to Point A results in a Point C at the head of the Vector B, with the new vector's tail at Point A:
<Cx, Cy, Cz> = <Ax+Bx, Ay+By, Az+Bz>
Or graphically,



The addition of a point to a vector does not make sense. This operation cannot be done.

The subtraction of Vector B from Point A results in a Point C at the head of the Vector A, with the new vector's tail at Point B:
<Cx, Cy, Cz> = <Ax-Bx, Ay-By, Az-Bz>
Or graphically,



The subtraction of a point from a vector does not make sense. This operation cannot be done.

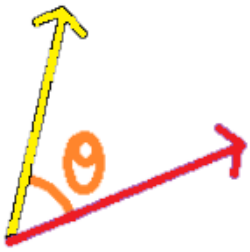
Finding the length of a vector looks rather similar to the distance formula:
L = √ (x^2+y^2+z^2)
This gives the scalar distance from the origin to the head of the vector.

A point has no length, this operation cannot be done on a point.

There are two different ways of multiplying vectors. The dot product (A*B), and the cross product(AxB). Don't worry about the cross product. The dot product results in a scalar value:
(A*B) = Ax*Bx + Ay*By + Az*Bz

We can use the dot product to find the angle between two vectors by taking the arccos of the dot product of the vectors divided by their multiplied length:
Θ = Math.acos((A*B)/(L(A)*L(B)))

Or graphically,



Points cannot have an angle between them or a vector. This operation cannot be done with a point.

Implementation

Step 1
Create a menu like the one below:

```
Object 1 has not been assigned.
Object 2 has not been assigned.
1: Change value of null
2: Change value of null
3: Add the objects.
4: Subtract the objects.
5: Find the angle between the objects.
6: Quit
```

The first two lines should only show “null” when the respective object has not been assigned a value. The user should be able to input a number (1-6) to select an option.

Note: 'null' in options 1 and 2 was not a hard-coded String. This is caused simply by System.out.println(); - when an object is input to this method, it automatically calls the toString(); method on the object. We will later override this method for our objects. For now, our objects don't have value and are null – so they print 'null'.

Step 2
Let the user assign values for the two Objects, no matter a Vector or Point, they will be entered in the format <x,y,z>. If the user deviates from this, and enters in something wrong, simply re ask.

The following are examples of good input:
<4.0,-5.0,-6.293482>
<1,-2,3.>
<1.2>
<.2,-.4>
<-3.4,6.9>

The following are examples of bad input:
3,-4,5
<3,5,7,8,4,3>
(-3,4,5)
<.,.,2>
<3-4,5,3.0>
<>
<3,abc,f5>

Note: <> brackets must be used!

Secondly, after a valid set of <x,y,z> values are entered, the user must specify if the object is a Point or a Vector (since there isn't a way to tell otherwise). Since these are so closely related, you must have a parent class that is abstract from which your Point and Vector classes are inherited. Since Vector is a class that exists in Java in a different context, you will probably need to name your class something different.
Note: Case is not sensitive!

Sample input/output:

```
Object 1 has not been assigned.
Object 2 has not been assigned.
1: Change value of null
2: Change value of null
3: Add the objects.
4: Subtract the objects.
5: Find the angle between the objects.
6: Quit
What do you want to do? 1
Enter values in the following format: <x,y,z>
<3.1,2.0>
Is this a Vector or a Point?
POINT

1: Change value of Point:<3.1,2.0,0.0>
2: Change value of null
3: Add the objects.
4: Subtract the objects.
5: Find the angle between the objects.
6: Quit
What do you want to do? 2
Enter values in the following format: <x,y,z>
<0,1.3,1.10>
Is this a Vector or a Point?
vector

1: Change value of Point:<3.1,2.0,0.0>
2: Change value of Vector:<0.0,1.3,1.1>
3: Add the objects.
4: Subtract the objects.
5: Find the angle between the objects.
6: Quit
What do you want to do?
```

Note: Make sure to override the toString() method in your Point and Vector classes! This way, they will be printed as shown in the example!

Step 3
Now implement the add, subtract, and angle functions. Make sure to only allow the user to use the method if it's valid to do so! A quick reference is below:

Object 1	Operation	Object 2	Result
Vector	Add	Vector	Vector
Point	Add	Point	Invalid
Vector	Subtract	Vector	Vector
Point	Subtract	Point	Vector
Point	Add	Vector	Point
Vector	Add	Point	Invalid
Point	Subtract	Vector	Point
Vector	Subtract	Point	Invalid
Vector	Angle Between	Vector	Angle

Note: No function should be valid when either of the two objects is still null.
Note: Keep in mind that you cannot find an angle between vectors where one or both vectors' length are zero.
Note: Also remember that the angle returned by Math.acos(double val); is in radians. Use Math.toDegrees(double val); to change this.

If the user tries to do something invalid, let them know with a simple message indicating why it was invalid.

Some samples are shown below. Notice that these are different executions of the program, though one execution of the program would re-prompt the user with the menu until they enter 6.

```
…
1: Change value of Point:<0.0,1.0,1.0>
2: Change value of Point:<0.0,0.0,0.0>
3: Add the objects.
4: Subtract the objects.
5: Find the angle between the objects.
6: Quit
What do you want to do? 4
The result is Vector:<0.0,1.0,1.0>
…
…
1: Change value of Vector:<0.0,1.9,3.2>
2: Change value of Point:<0.1,1.2,8.1>
3: Add the objects.
4: Subtract the objects.
5: Find the angle between the objects.
6: Quit
What do you want to do? 3
Cannot add a Point to a Vector!
…
…
1: Change value of Vector:<0.0,-1.0,-1.0>
2: Change value of Vector:<1.0,1.0,0.0>
3: Add the objects.
4: Subtract the objects.
5: Find the angle between the objects.
6: Quit
```

```
What do you want to do? 5
The angle between them is 90.0 degrees
...
```

Grading

This assignment will be graded by using a series of inputs to test your program. We will use a variety of good and bad inputs to test. A general rubric has been provided below. If you lose points anywhere, the category and the point values will be listed on Blackboard. We will release the exact test cases used after the assignment is due so you can test it yourself and check the grade given to you. Partial credit is given.

Category and % of Grade	Criteria
Menu 0.5%	
0.1%	Has general menu.
0.2%	Menu can take in input and act on it.
0.2%	Correct parsing on user input for object creation.
Classes/Polymorphism 1.0%	
0.2%	At least three classes are used.
0.2%	An abstract class is used.
0.2%	Point and Vector class both extend the same abstract class.
0.2%	The objects used in the menu are declared as the abstract class.
0.2%	Text in the menu/results list the correct type when printed.
Functionality 1.0%	
0.75%	The functionality of the menu items return correct output.
0.25%	The program guards against incorrect method usage.