Vector Calculator

Design Tech I - Interdisciplinary Project
By Seung-Gu Lee

Table of Contents

Skip to #4 to see how to solve the problem manually.

SUMMARY

Welcome to *Vector Calculator*!

This program is a short Java program that guides a user to add up the vectors and then shows the resultant, final direction/angle, total of X and Y components, each X and Y components, number of vectors entered, and/or the average.

- Datas required to run the program: magnitude, direction, angle of multiple vectors
- Datas recommended (not required) to run the program: unit of vector
- Related Subject: Physics Honors (Teacher: Ms. Hidalgo)

SHORT/BRIEF EXPLANATION VERSION

Vector is a quantity having magnitude AND direction. Vectors can be added analytically by dividing the vectors into X and Y components, add all of them, and use Pythagorean theorem and trigonometry to figure out the resultant and the final direction angle.

Step 0. Entering Unit

The program will first prompt the user to enter the unit. This process can be skipped by pressing *Enter/Return* without entering anything. The unit will be assigned to a string and will only be used in displaying process.

Step 1. Separating Vectors into X and Y components

The program will prompt the user to enter magnitude, direction, and angle(not always required). X and Y components will be calculated using the following trigonometry formulas

$Xcomp = magnitude * cos(angle)$, $Y comp = magnitude * sin(angle)$
and they will be assigned to a spot in arrays. The signs(+, -, 0) of X and Y components may be changed depending on the direction. (e.g. If direction is NW(-, +), X component will be changed to negative) Other calculations such as keeping track of the counter will be done too.
The whole Step 1 will be repeated until the user The X and Y components will be added to the total every time step 1 is repeated.

Step 2. Calculating Resultant and Final Direction/Angle
The resultant will be found using the formula (which is slightly modified Pythagorean Theorem) $Resultant = \sqrt{Xcomp^2 + Y comp^2}$. The final angle will be calculated using the formula (slightly modified trigonometry formula) $angle = tan^{-1}(Y comp/Xcomp)$
The angle will be modified so that it becomes an angle *above/below the horizontal axis*.
The final direction will be calculated depending on the sign(+, -, 0) of X and Y components. (e.g. If X component is negative and Y component is negative, the direction is SW/3rd quadrant)

Step 3. Printing Results
The program will print the following: resultant, final direction and angle, total of X and Y components. The program will also calculate/print the average and/or the list of X and Y components if the user requests and then ask the user to whether repeat the entire process or not.

LONG/DETAILED EXPLANATION VERSION

What is a Vector?

Vector is a quantity having magnitude AND direction. Displacement, velocity, acceleration, force, and momentum are examples of vectors.

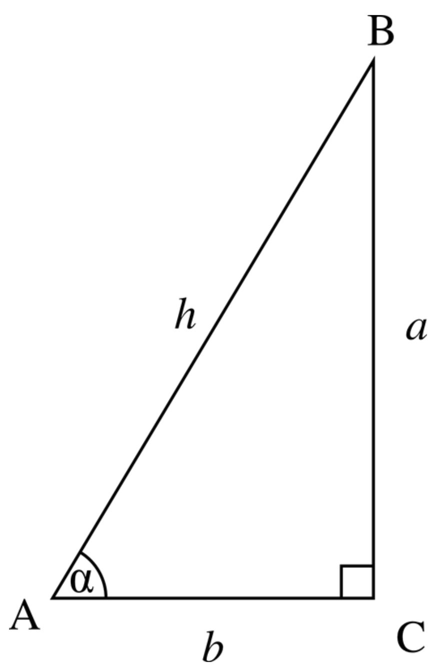
How to Add Vectors

Vectors can be added in two ways: graphically and analytically. Graphically adding is connecting heads and tails of vectors, and connecting the first tail and the last head. The analytical way of resolving a vector into X and Y components, adding the X and Y components, and then using Pythagorean Theorem and Trigonometry to find the resultant and the direction.

This program uses the analytical method to add vectors.
The program will involve in the following steps:
Receive unit → Receive magnitude → Receive direction and angle → Separate(resolve) it to X and Y components → Add new components to total → (Repeat steps above until user has

entered all vectors) → Use Pythagorean Theorem to calculate resultant → Use Trigonometry to calculate the final direction and angle

Basics of Trigonometry
These are the basic trigonometry equations.

<table>
<tr>
<td rowspan="2">
</td>
<td>sin ∠ɑ = a / h
which is same as
sin ∠ɑ * h = a

sin ∠P0 * magnitude
= Y component</td>
</tr>
<tr>
<td>cos ∠ɑ = b / h
which is same as
cos ∠ɑ * h = b

cos ∠P0 * magnitude
= X component</td>
</tr>
</table>

In the case of vectors, the magnitude is the hypotenuse(C), the X component is the adjacent(A), the Y component is the opposite(B), and the ∠P0 is the direction shown by angles.

By using the equations above, A (X component) and B (Y component) can be found with C and ∠P0 (and also sin and cos values of ∠P0, which will be determined using Math methods).

Entering a Unit

Although entering a unit isn't really necessary, it will improve the visibility. Every time before the first vector is entered, the program will ask user the unit and assign it to a String. If the vectors does not have a unit, user can press *enter/return* without entering anything.

Entering a Vector

So the program asks the user the magnitude, which is C or hypotenuse, for each vector. Then, it asks the direction of the vector, whether it is north / south / east / west or northeast (1st quadrant) / northwest (2nd quadrant) / southwest (3rd quadrant) / southeast (4th quadrant). If the direction is NE/NW/SW/SE, the program will also ask the user angle (∠P0).

When the information of the vector(magnitude, direction, angle) is entered, my program divides the vector into X and Y components using the following table of equations. These are the formulas/patterns I found.

	North	South	East	West
	Y+	Y-	X+	X-
X component	0	0	C	-1 * C
Y component	C	-1 * C	0	0

	Northeast	Northwest	Southwest	Southeast
	1st Quadrant	2nd Quadrant	3rd Quadrant	4th Quadrant
X component	sin ∠P0 * C	-1 * sin ∠P0 * C	-1 * sin ∠P0 * C	sin ∠P0 * C
Y component	cos ∠P0 * C	cos ∠P0 * C	-1 * cos ∠P0 * C	-1 * cos ∠P0 * C

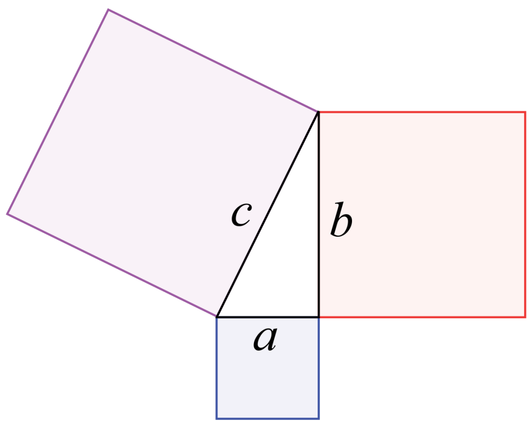
* ∠P0 is an angle below or above the horizontal.

By the way, radians are used for Math.sin and Math.cos methods. But in high school Physics, radians is rarely used and degrees are used instead. So user enters the angle in degrees and my program converts it to radians using Math.toRadians method.

So now that the vector is divided into X and Y components, and they will be each stored into arrays. They will also be added to the total of X and Y components, which are also stored in an array. The counter will also be added by 1 to keep track of the number of vectors and calculate the average.

The program will ask the user if user will enter more vectors. Users can enter up to 100 vectors.
- If user replies *yes*, the entire ‘Entering a Vector’ process will be repeated. This is possible because the entire ‘Entering a Vector’ process is in a do-while loop.
- If user replies *no*, then the loop will be ended and the next part will begin.
- If user replies anything else, an error message will be printed and the user will be prompted to enter either *yes* or *no*, This is possible because the asking part is also a do-while loop.

Calculating Results

	$a^2 + b^2 = c^2$ $c = \sqrt{a^2 + b^2}$
	In the diagram, a is the X component, b is the Y component, and c is the resultant. According to the Pythagorean Theorem, $c = \sqrt{a^2 + b^2}$ so $resultant = \sqrt{Xcomp^2 + Y\,comp^2}$. The program plugs X and Y components to the equation and gets the resultant.

The program will then find the direction using conditional statements. The table below is the pattern of direction and X and Y components. Based on the signs(+, -, 0) of X and Y components, the program assigns the direction to a string.

X	+	-	-	+	+	-	0	0	0
Y	+	+	-	-	0	0	+	-	0
Direction	1st quadrant	2nd quadrant	3rd quadrant	4th quadrant	North	South	East	West	N/A

		X		
		-	0	+
Y	+	NW (2nd quadrant)	N	NE (1st quadrant)
	0	W	N/A	E
	-	SW (3rd quadrant)	S	SE (4th quadrant)

Now the program has to find the angle. The angle can be found using this simple formula $tan^{-1}\,(b\,/\,a) = angle$, which is the same as $tan^{-1}\,(Y\,comp\,/\,Xcomp) = angle$. The program plugs X and Y components to the equation and gets the angle. The sign of the angle will be modified so that it will not be ridiculous(e.g. -60 degrees below horizontal axis—it has to be 60 degrees above horizontal axis according to Ms. Hidalgo).

Displaying Results and Others

After results are calculated, they will be printed like this:

Format	*Example*
The total of your vectors is: Resultant: *[resultant] [unit]* Angle: *[angle]* degrees Direction: *[direction]* [x] *[X component] [unit]* [y] *[Y component] [unit]* You have entered a total of *[counter]* vectors. Please note that the numbers might have up to ±0.01% error due to radian ↔ degree conversion.	The total of your vectors is: Resultant: 10.0 m/s Angle: 53.13010235 degrees Direction: 1st quadrant (Northeast) above the horizontal axis [x] 8.0 m/s [y] 6.0 m/s You have entered a total of 2 vectors. Please note that the numbers might have up to ±0.01% error due to radian ↔ degree conversion.

* Users are recommended to round the numbers up to 2~3 decimals.

Then the program will ask the user following things: (1) if the program will display the average (2) if the program will display all entered vectors separated to X and Y components (3) if the program will repeat the entire process again.

1	If user enters "yes", the program calculates the average using $average = total / counter$ and displays it. If user enters anything else, the program will skip this process.
2	If user enters "yes", the program will display the X and Y components in the following format: #*[counter]* [x] *[X component]* [y] *[Y component]* If user enters anything else, the program will skip this process.
	This is possible because every time the program divides the vector to X and Y components it assigns the X and Y components to set of arrays.
3	If user enters "yes", the program will display the entire process again: a do-while loop is used in the entire program to make this possible. All variables/strings will be resetted. If user enters "no", the program will end the program and display a farewell message. If user enters anything else, the program will prompt the user again.

SOLVING IT MANUALLY

Vector addition can be done both graphically and analytically as mentioned earlier, but this document will focus on the analytical way.

Step 1. Separate the vector into X and Y components, but in absolute value.

- X component = magnitude * cos(angle)
- Y component = magnitude * sin(angle)
- The angle must be in N *degrees above/below horizontal*; if not, convert it.

Step 2. Modify the signs of X and Y components

- If the direction is towards the West, change the X component to negative.
- If the direction is towards the South, change the Y component to negative.

Step 3. Add X and Y components

Step 4. Get resultant

- $resultant = \sqrt{totalY^2 + totalX^2}$

Step 5. Get direction

- Use these tables (either one) to find out the direction.

		X		
		-	0	+
Y	+	NW (2nd quadrant)	N	NE (1st quadrant)
	0	W	N/A	E
	-	SW (3rd quadrant)	S	SE (4th quadrant)

X	+	-	-	+	+	-	0	0	0
Y	+	+	-	-	0	0	+	-	0
Direction	1st quadrant	2nd quadrant	3rd quadrant	4th quadrant	North	South	East	West	N/A

- $angle = tan^{-1}(totalY / totalX)$
- If $angle \geq 90$, subtract the angle by 90
- If $angle < 0$, change to positive

Example 1

50 m/s, 30 degrees North of West 30 m/s, 60 degrees North of East 70 m/s, South

<table>
<tr><td colspan="3">Step 1. Separating vector into X and Y components (but in absolute value)</td></tr>
<tr><td>$Xcomp = 50 * cos(30)$
= 43.30
$Y comp = 50 * sin(30)$
= 25.0</td><td>$Xcomp = 30 * cos(60)$
= 15.0
$Y comp = 30 * sin(60)$
= 25.98</td><td>$Xcomp = 0$ = 0.0
$Y comp = 70$ = 70.0</td></tr>
<tr><td colspan="3">Step 2. Modifying the X and Y components’ signs</td></tr>
<tr><td>North of West: (-, +)
Xcomp = -43.30
Ycomp = 25.0</td><td>North of East: (+, +)
Xcomp = 15.0
Ycomp = 25.98</td><td>South: (0, -)
Xcomp = 0
Ycomp = -70.0</td></tr>
<tr><td>Step 3. Adding components</td><td colspan="2">Total X = -43.30 + 15.0 + 0 = -28.30
Total Y = 25.0 + 25.98 - 70.0 = -19.02</td></tr>
<tr><td>Step 4. Getting Resultant</td><td colspan="2">Step 5. Getting Direction</td></tr>
<tr><td>$Resultant = \sqrt{X^2 + Y^2}$
$= \sqrt{(-28.30)^2 + (-19.02)^2}$
= 34.10</td><td colspan="2">X is (-) and Y is (-) so direction is SW/3rd quadrant
Java $angle = tan^{-1}(Y/X) = tan^{-1}(-19.02/-28.30) =$ 33.90
Modified angle = 33.90</td></tr>
<tr><td>Answer</td><td colspan="2">34.10 m/s; 33.90 degrees, 3rd quadrant (Southeast) below the horizontal axis</td></tr>
<tr><td>Program-calculated Result</td><td colspan="2">The total of your vectors is:
Resultant: 34.09828887356898 m/s
Angle: 33.9021946106759 degrees
Direction: 3rd quadrant (Southwest) below horizontal axis
[x] -28.301270189221935 m/s
[y] -19.019237886466847 m/s
You have entered a total of 3 vectors.
Please note that the numbers might have up to ±0.01% error due to radian ↔ degree conversion.</td></tr>
<tr><td>Compare</td><td colspan="2">What the program calculated is the same as what I did!</td></tr>
</table>

Example 2

7N, 22.5 degrees West of North 15N, 87.3 degrees East of South 16N, 43 degrees North of East

Step 1. Separating vector into X and Y components (but in absolute value)

22.5 degrees West of North = 67.5 degrees North of West	87.3 degrees East of South = 3.7 degrees South of East	$Xcomp = 16 * cos(43)$ = 11.70 $Y\,comp = 16 * sin(43)$ = 10.91
$Xcomp = 7 * cos(67.5)$ = 2.68 $Y\,comp = 7 * sin(67.5)$ = 6.47	$Xcomp = 15 * cos(3.7)$ = 14.97 $Y\,comp = 15 * sin(3.7)$ = 0.97	
Step 2. Modifying the X and Y components' signs		
North of West: (-, +) Xcomp = -2.68 Ycomp = 6.47	South of East: (+, -) Xcomp = 14.97 Ycomp = -0.97	North of East: (+, +) Xcomp = 11.70 Ycomp = 10.91
Step 3. Adding components	Total X = -2.68 + 14.97 + 11.70 = 23.99 Total Y = 6.47 - 0.97 + 10.91 = 16.41	
Step 4. Getting Resultant	Step 5. Getting Direction	
$Resultant = \sqrt{X^2 + Y^2}$ $= \sqrt{(23.99)^2 + (16.41)^2}$ = 29.07	X is (+) and Y is (+) so direction is NE/1st quadrant Java $angle = tan^{-1}(Y/X) = tan^{-1}(16.41/23.99)$ = 34.37 Modified angle = 34.37	
Answer	29.07 N; 34.37 degrees, 1st quadrant (Northeast) above the horizontal axis	
Program-calculated Result	The total of your vectors is: Resultant: 29.067559817105302 N Angle: 34.37356773921072 degrees Direction: 1st quadrant (Northeast) above horizontal axis [x] 23.991609473433783 N [y] 16.411145864784615 N You have entered a total of 3 vectors. Please note that the numbers might have up to ±0.01% error due to radian ⇔ degree conversion.	
Compare	What the program calculated is the same as what I did!	