**Chapter 13 Exercise 3**

Define a class Arrow, which draws a line with an arrowhead.

First, find a point on line

|  |  |
| --- | --- |
|  | I think this belongs on MathOverflow, but I'll answer since this is your first post. First you calculate the vector from x1y1 to x2y2:  float vx = x2 - x1; float vy = y2 - y1;  Then calculate the length:  float mag = sqrt(vx\*vx + vy\*vy);  Normalize the vector to unit length:  vx /= mag; vy /= mag;  Finally calculate the new vector, which is x2y2 + vxvy \* (mag + distance).  \*px = (int)((float)x1 + vx \* (mag + distance)); \*py = (int)((float)y1 + vy \* (mag + distance));  You can omit some of the calculations multiplying with distance / mag instead. |

<http://stackoverflow.com/questions/1800138/given-a-start-and-end-point-and-a-distance-calculate-a-point-along-a-line>

Then, rotate an end point of arrow

Without getting to heavily into general solutions and maths (as the above posters and lots of articles have covered this already), I could give you an example on how to solve the problem "rotating a point A around point B by C degrees".

Now. First of all, as I described in the previous post, a point that is on the X axis, L distance from origo, is rotated C degrees around origo by

x = L \* cos(C)

y = L \* sin(C)

Similarly, the formula for a perpendicular vector is x = -y | y = x, which means that a point that is on the Y axis (again, L from origo) would be rotated by C using the formula

x = - L \* sin(C)

y = L \* cos(C)

As shown in the above image, the final solution is the sum of the rotations of the projected vectors, so we can derive the formula

x' = x \* cos(C) - y \* sin(C)

y' = y \* cos(C) + x \* sin(C)

... but you knew that already, right? problem is, this formula only rotates around origo. So what we need to do is move the coordinate system we're rotating around to origo, rotate and then move back. This can be done quickly with complex numbers or in general solutions with matrices, but we're gonna stick to vector math on this one to keep it simple.

first step; move the origin point.

x' = A.x - B.x

y' = A.y - B.y

second step, perform rotation

x'' = x' \* cos(C) - y' \* sin(C) = (A.x-B.x) \* cos(C) - (A.y-B.y) \* sin(C)

y'' = y' \* cos(C) + x' \* sin(C) = (A.y-B.y) \* cos(C) + (A.x-B.x) \* sin(C)

third and final step, move back the coordinate frame

x''' = x'' + B.x = (A.x-B.x) \* cos(C) - (A.y-B.y) \* sin(C) + B.x

y''' = y'' + B.y = (A.y-B.y) \* cos(C) + (A.x-B.x) \* sin(C) + B.y

And presto! we have our rotation formula. I'll give it to you without all those >calculations:

Rotating a point A around point B by angle C

A.x' = (A.x-B.x) \* cos(C) - (A.y-B.y) \* sin(C) + B.x

A.y' = (A.y-B.y) \* cos(C) + (A.x-B.x) \* sin(C) + B.y

If you've been following me here (and I'm a pretty lousy teacher, so sorry if you haven't), you can se that the ordering in which you perform these operations is very important. Try to mix step 3 and 1 and see the difference in the formulae you get.

Good luck and all!

http://stackoverflow.com/questions/3837266/finding-the-points-of-a-triangle-after-rotation