CMSC425 Ray-Circle intersection problem
Notes: Taking a problem from math to Unity

Problem:
Given a point **p**, a vector **v**, and a circle defined by a center **c** and a radius **r**. All in 2D. Give a Boolean answer to the question:

Does the ray defined by **p** and **v** intersect the circle defined by **c** and **r**? The ray is the half line defined by r(t) = p + tv with t in [0,inf].

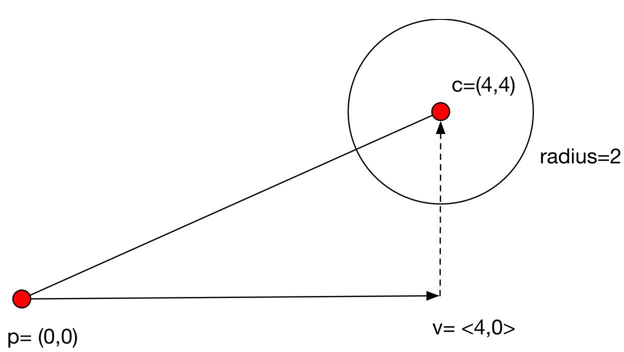
Notes:
This problem was assigned as a homework in Spring 2019 and the solutions to this will be posted for you. This handout is to suggest a full cycle in solving this problem, from problem to solution to code to testing to final code.

Step 1) Solve the problem in mathematical form, here by decomposing the vector u, from the ray origin point p to the circle center point c, into two orthogonal vectors based on v.

From class:	Octave-online code
a) Normalize ***v*** to get unit vector ***vn***	**vn = v / norm(v)**
b) Compute vector ***u*** = **c** – **p**.	**u** = **c** – **p**
c) Project u onto v with ***u1* = (*u*•*v*)*v***	**u1 = dot(u,v)*v**
d) Compute orthogonal ***u2* = *u* – *u1***	**u2 = u – u1**
e) Find distance point to ray with **d =|u2|**	**d = norm(u2)**
f) Test if d > radius	

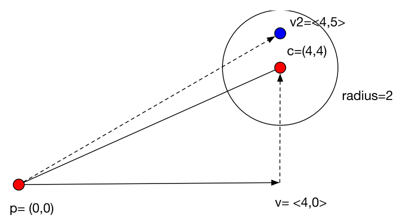
Step 2) Quickly prototype your solution in Octave-online or other Matlab-like program. This is a lightweight way to test your solution where it's easy to see intermediate values.

Step 3) Test your solution on a sequence of cases. Start with an obvious case where you know the answer right away. Here the ray is along the x-axis and the circle 4 units above, so we know that the intersection should fail.



Case 1: p=(0,0), c=(4,4), v=<4,0>, radius = 2.
a) Normalize v to get vn = v/4 = <1,0>
b) Compute u = c – p = <4,4>
c) Project u onto v with u1 = dot(u,v) * vn = <4,0>
d) Compute u2 = u – u1 = <4,4> - <4,0> = <0,4>
e) Find the distance from c to the ray as magnitude |u1| = 4
f) Since 4 > radius 2, no intersection.

Then add additional cases also with known answers, like on the left where the cases with v2 or with v=c-p should succeed. What will u2 be if v goes directly through c?



Case 2: p=(0,0), c=(4,4), v2=<4,5>, radius = 2.
a) Normalize v to get vn2 = v2/6.4031 = < 0.62470,0.78087>
b) Compute u = c – p = <4,4>
c) Project u onto v with u1 = dot(u,v) * vn2 = < 3.5122,4.3902>
d) Compute u2 = u – u1 = <4,4> - < 3.5122,4.3902>= < 0.48780,-0.39024>
e) Find the distance from c to the ray as magnitude |u1| = 0.62470
f) Since 0.62470 < radius 2, intersection.

Step 4) Convert Octave code into C# code for Unity and repeat the tests.

```
Vector3 P; //
Vector3 v; // ray vector
Vector3 C; //
float r;   // Radius

v = v.normalize;
Vector3 u  = C - P;
Vector3 u1 = Vector3.dot(u,v) * v;
Vector3 u2 = u - u1;
float d    = u.magnitude;
if (d < r) then return true; else return false;
```