CSS385: Assignments 2 and 3 Question Sheet
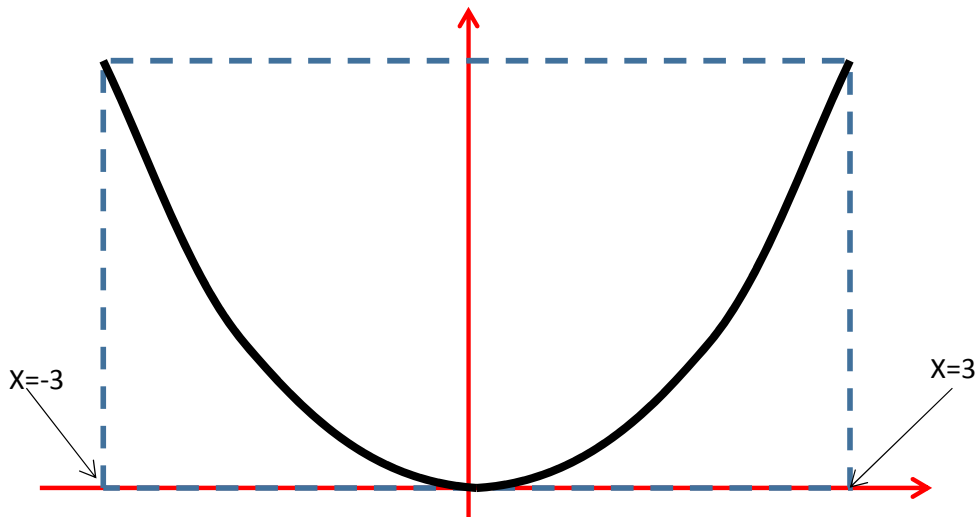
Assignment 2

1. (**12pt**) If you want your patrol to travels along the function $y = 4x^2$ in the range of $-3 \leq x \leq 3$. In other words, you want to fit the following plot:

into your game window and have the patrol follow the parabola. Assuming the world origin (0,0) is located at the lower-left corner of the game window, the following function plots the patrol path as a sequence of circles:

```
private void PlotCurve()
{
    for (int x = 0; x < World.WorldDimension.X; x++)
    {
        float y = GetYValue(x);
        XNACS1Circle c = new XNACS1Circle(new Vector2(x, y), 0.2f);
        mPlotSet.AddToSet(c);
    }
}
```

Where *GetYValue()* is defined as

```
private float GetYValue(float x)
{
    compute the value for useX

    compute the value for yScale

    return yScale * 4 * (useX * useX);
}
```

Now, consider the *GetYValue(float x)* function as given above, and the fact that the *GetYValue()* is only called from the *PlotCurve()* function:

a.   (**4pts**) Minimum values:
  i.   (**1pt**) What is the smallest value that will be passed into the *GetYValue()* function (smallest number for x, the parameter of the *GetYValue()* function)?
  ii. (**2pt**) In this case, what should be the computed value for **useX** ?
  iii. (**1pt**) In this case, what should be the return value from *GetYValue()*?

b.   (**4pts**) Maximum values:
  i.   (**1pt**) What is the largest value that will be passed into the *GetYValue()* function (largest number for x, the parameter of the *GetYValue()* function)?
  ii. (**2pts**) In this case, what should be the computed value for **useX**?
  iii. (**1pt**) In this case, what should be the return value from *GetYValue()*?

c.   (**2pt**) Consider your answer to **a** and **b**, derive the expression for computing **_yScale_**.

d.   (**2pt**) Now derive the expression for computing **_useX_**.

Assignment 3

1. **(5pt)** Give the following code that implements home-in chasing:

```
91    private void ComputeNewDirection(Vector2 toTarget)
92    {
93        double cosTheta = Vector2.Dot(toTarget, FrontDirection);
94        float theta = MathHelper.ToDegrees((float)Math.Acos(cosTheta));
95        if (theta > 0.001f)
96        {
97            Vector3 frontDir3 = new Vector3(FrontDirection, 0f);
98            Vector3 toTarget3 = new Vector3(toTarget, 0f);
99            Vector3 zDir = Vector3.Cross(frontDir3, toTarget3);
100           RotateAngle += Math.Sign(zDir.Z) * 0.15f * theta;
101           VelocityDirection = FrontDirection;
102       }
103   }
```

Please identify by indicating the exact line number(s) where we are performing the following tasks. In cases where the specified task is not supported by the given code, please say "***Not Supported***". For example, for the task:

Compute the color of the chaser. **Answer: *Not Supported***

a. **(1pt)** Compute the vector from the chaser to the target.
b. **(1pt)** Compute the distance between the chaser and the target.
c. **(1pt)** Compute the angle (in degrees) between the chaser's front direction and the direction towards the target.
d. **(1pt)** Check to ensure the chaser does not overlap the target.
e. **(1pt)** Changing the front direction of the chaser.

2. **(4pt)** Continue with the above question:
   a. **(2pt)** Does the given method belongs to the chaser-class or the target-class?
   b. **(2pt)** At what rate does the given method turns?

3. **(3pt)** Our hero has the following states:
    - Run
    - Walk
    - Attack
    - Stand
    - Sit

We want to be able to transition the hero into each of the states, and the hero should never get *stuck* in any of the states (the hero can always transit out of a state), **not** counting remaining in the same state as a transition, and note that a transition from state-A to state-B is **different** from the transition from state-B to state-A:

   a. **(1pt)** How many distinct state transitions can we have in <u>minimum</u>?

   b. **(2pt)** How many distinct state transitions can we have in <u>maximum</u>?

4. **(2pt)** Given the following vectors:
    - $\vec{V_1} = \begin{bmatrix} 10 \\ 4 \end{bmatrix}$

    - $\vec{V_2} = \begin{bmatrix} -2 \\ 5 \end{bmatrix}$

   a. **(1pt)** What is the value of $\vec{V_1} \cdot \vec{V_2}$?

   b. **(1pt)** From **a**, what do we know about these two vectors?