Ejercicios Clase 7. Arreglos y matrices

1. Aplicar un matriz de transformación cuyos elementos dependan de la posición del mouse, a una composición formada por triángulos. Tanto la composición original como la composición "transformada" por la matriz deben ser dibujadas en la pantalla.

3.* Utilizar arreglos unidimensionales para crear una composición donde un gran número de elementos se desplazan por la pantalla, utilizando colores y efectos de transparecia para hacer la visualización interesante. Partir de este ejemplo de código:

```
// Ejemplo de sistema de particulas.

// Estos dos arreglos se usan para guardar las coordenadas
// x e y de las particulas.
float[] x;
float[] y;

// Estos dos arreglos se usan para guardar las velocidades
// vx y vy de las particulas.
float[] vx;
float[] vy;

void setup()
{
   size(400, 400);

   // Se crean loa arreglos para guardar 100 particulas:
   x = new float[100];
   y = new float[100];

   vx = new float[100];
   vy = new float[100];

   // Se asignan posiciones iniciales y velocidades aleatorias.
   for (int i = 0; i < 100; i++)
   {
      x[i] = random(0, width);
      y[i] = random(0, height);

      vx[i] = random(-1, 1);
      vy[i] = random(-1, 1);
   }
}

void draw()
{
```

```
  background(0);

  // En este bucle se actualizan las posiciones de cada particula
  // dependiendo de su velocidad.
  for (int i = 0; i < 100; i++)
  {
    // Se actualiza la coordenada x.
    x[i] = x[i] + vx[i];
    if (width < x[i])
    {
      // Si la particula se sale de la pantalla por la
      // derecha vuelve a entrar por la izquierda.
      x[i] = 0;
    }
    if (x[i] < 0)
    {
      // Si la particula se sale de la pantalla por la
      // izquierda vuelve a entrar por la derecha.
      x[i] = width;
    }

    y[i] = y[i] + vy[i];
    if (height < y[i])
    {
      // Si la particula se salen de la pantalla por la
      // abajo vuelve a entrar por la arriba.
      y[i] = 0;
    }
    if (y[i] < 0)
    {
      // Si la particula se salen de la pantalla por la
      // arriba vuelve a entrar por la abajo.
      y[i] = height;
    }

    // Se dibuja la particula.
    ellipse(x[i], y[i], 10, 10);
  }
}
```

4.* Crear una animación resultante de aplicar distintas matrices de rotación a distintos elementos dibujados en la pantalla. Los ángulos de rotación deben variar de manera autónoma.