PROGRAMMING ASSIGNMENT #6 - REVIEW

Program.cs

```
using System;
using System.Threading;

namespace LWTech.ChipAnderson.GameOfLife
{
   class Program
   {
      static Random rng = new Random();

      const int ALIVE = 1;
      const int DEAD = 0;

      const int maxX = 120;
      const int maxY = 40;

      const int randomFillPercent = 20;
      const int randomCosmicRayChance = 1000;

      static void Main(string[] args)
      {
         int sleepTime = 50;
         bool randomRays = false;

         Console.WriteLine("Conway's Game of Life:");
         Console.WriteLine("======================");

         int numGeneration = 0;
         int[,] oldGrid = new int[maxX, maxY];
         int[,] newGrid = new int[maxX, maxY];

         FillGridRandom(newGrid, randomFillPercent);

         bool quit = false;
         while (!quit)
```

```
{
    numGeneration++;

    Console.WriteLine("Generation #" + numGeneration + (randomRays ? " w/Cosmic Rays!" : ""));
    DisplayGrid(newGrid);
    oldGrid = (int[,])newGrid.Clone();

    // Apply the rules of Life using "static" borders and no grid wrapping
    for (int j = 1; j < maxY - 1; j++)
    {
        for (int i = 1; i < maxX - 1; i++)
        {
            int numNeighbors = oldGrid[i - 1, j - 1] + oldGrid[i, j - 1] + oldGrid[i + 1, j - 1]
                + oldGrid[i - 1, j] + oldGrid[i + 1, j]
                + oldGrid[i - 1, j + 1] + oldGrid[i, j + 1] + oldGrid[i + 1, j + 1];

            if (oldGrid[i, j] == 1)
                newGrid[i, j] = (numNeighbors == 2 || numNeighbors == 3) ? ALIVE : DEAD;
            else
                newGrid[i, j] = (numNeighbors == 3) ? ALIVE : DEAD;
        }
    }

    if (randomRays && rng.Next(randomCosmicRayChance) < 80)
        newGrid[rng.Next(maxX), rng.Next(maxY)] = ALIVE;

    Thread.Sleep(sleepTime);

    if (Console.KeyAvailable) {
        ConsoleKey key = Console.ReadKey(true).Key;
        switch (key)
        {
            case ConsoleKey.D5:
                AddRPentomino(newGrid);
                break;
            case ConsoleKey.F:
                FillGrid(newGrid);
                break;
            case ConsoleKey.R:
                FillGridRandom(newGrid, randomFillPercent);
                break;
            case ConsoleKey.Y:
```

```
                randomRays = !randomRays;
                break;
            case ConsoleKey.Q:
                quit = true;
                break;
            default:
                break;
        }

    }

}

private static void FillGridRandom(int[,] grid, int randomPercent)
{
    for (int y = 0; y < maxY; y++)
        for (int x = 0; x < maxX; x++)
            grid[x, y] = (rng.Next(100) < randomPercent) ? ALIVE : DEAD;
}

private static void FillGrid(int[,] grid)
{
    for (int y = 0; y < maxY; y++)
        for (int x = 0; x < maxX; x++)
            grid[x, y] = ALIVE;
}

private static void AddRPentomino(int[,] grid)
{
    int x, y;
    for (y = 0; y < maxY; y++)
        for (x = 0; x < maxX; x++)
            grid[x, y] = DEAD;

    x = maxX / 2;
    y = maxY / 2;

    grid[x, y - 1] = ALIVE;
    grid[x + 1, y - 1] = ALIVE;
    grid[x - 1, y] = ALIVE;
```

```
            grid[x, y] = ALIVE;
            grid[x, y + 1] = ALIVE;

        }

        private static void DisplayGrid(int[,] grid)
        {
            string line;
            for (int y = 0; y < maxY; y++)
            {
                line = "";
                for (int x = 0; x < maxX; x++)
                {
                    line += grid[x, y] == ALIVE ? "*" : " ";
                }
                Console.WriteLine(line);
            }
        }

    }
}
```