```
////////////////////////////////////////////////////////////////////////////
//                                                                        //
//   The following program is the source code for "THE GAME OF LIFE" a game  //
//            which uses a few rules to create beautiful designs.         //
//                                                                        //
//                         WRITTEN BY:                                    //
//                                Alain Dadaian                           //
//                                                                        //
////////////////////////////////////////////////////////////////////////////

class Board
{
   private int rmax, cmax;
   private int [][] board;

   //-----------------------------------------------------------------
   //  The Board constructor takes in two arguements which are the
   //  the limits of the game and fills an array of correct size with
   //  zeros.
   //-----------------------------------------------------------------
   public Board (int rmax, int cmax)
   {
      this.rmax = rmax;
      this.cmax = cmax;

      board = new int[rmax][cmax];

      for (int j = 0; j < cmax; j++)
         for (int i = 0; i < rmax; i++)
            board[j][i] = 0;

   }

   //-----------------------------------------------------------------
   //  Prints the game using 1's and 0's.
   //-----------------------------------------------------------------
   public void print()
   {
      for (int j = 0; j < cmax; j++)
      {
         for (int i = 0; i < rmax; i++)
            System.out.print(board[j][i]);

         System.out.println();
      }
   }

   //-----------------------------------------------------------------
   //  Prints the game but uses characters instead of 1's and 0's.
   //-----------------------------------------------------------------
   public void print (char char0, char char1)
   {
      for (int j = 0; j < cmax; j++)
      {
         for (int i = 0; i < rmax; i++)
         {
            if (board[j][i] == 0)
               System.out.print(char0);

            if (board[j][i] == 1)
               System.out.print(char1);
         }

         System.out.println();
      }
   }

   //-----------------------------------------------------------------
   //  Checks to see if the piece at a particular place is alive or 1.
   //-----------------------------------------------------------------
   public boolean isAlive(int i , int j)
   {
      if (i >= 0 && j >= 0 && i <= rmax - 1 && j <= cmax - 1 && board[j][i] == 1)
         return true;
      else
         return false;
   }

   //-----------------------------------------------------------------
```

```
   //  Returns the number of pieces alive around a certain piece.
   //-----------------------------------------------------------------
   public int livingNeighbors (int i, int j)
   {
      int result = 0;

      if (isAlive(i-1, j-1)) result++;
      if (isAlive(i, j-1)) result++;
      if (isAlive(i+1, j-1)) result++;
      if (isAlive(i-1, j)) result++;
      if (isAlive(i+1, j)) result++;
      if (isAlive(i-1, j+1)) result++;
      if (isAlive(i, j+1)) result++;
      if (isAlive(i+1, j+1)) result++;

      return result;
   }

   //-----------------------------------------------------------------
   //  Computes the next generation Board according to the rules of
   //  the Game of Life.
   //-----------------------------------------------------------------
   public Board next()
   {
      Board nextGenBoard = new Board(rmax, cmax);

      for (int j = 1; j < cmax - 1; j++)
         for (int i = 1; i < rmax - 1; i++)
         {
            if (board[j][i] == 1 && (livingNeighbors(i, j) == 2 || livingNeighbors(i, j) == 3))
               nextGenBoard.set(i, j);
            else
               nextGenBoard.reset(i, j);

            if (board[j][i] == 0 && livingNeighbors(i, j) == 3)
               nextGenBoard.set(i, j);
         }

      return nextGenBoard;
   }

   //-----------------------------------------------------------------
   //  Sets the value of the pixel at position (r, c) to be 1 or
   //  alive.
   //-----------------------------------------------------------------
   public void set (int r, int c)
   {
      if (r == 0 || r == rmax || c == 0 || c == cmax)
         board[c][r] = 0;
      else
         board[c][r] = 1;
   }

   //-----------------------------------------------------------------
   //  Sets the value of the pixel at position (r, c) to be 0 or
   //  dead.
   //-----------------------------------------------------------------
   public void reset (int r, int c)
   {
      board[c][r] = 0;
   }
   //-----------------------------------------------------------------

}// end of class Board

////////////////////////////////////////////////////////////////////////////

class Hw11
{
   public static void main (String[] args)
   {
      Board b = new Board(50, 50);

      for (int j = 21; j <= 30; j++)
         for (int i = 21; i <= 30; i++)
           b.set(i, j);

      System.out.println("\nGeneration 0:");
      b.print('.', 'X');
```

```
      System.out.println("\n");

      for (int i = 1; i <= 20; i++)
      {
         b = b.next();

         if (i == 1 || i == 2 || i == 3 || i == 10 || i == 20)
         {
            System.out.println("Generation " + i + ":");
            b.print('.', 'X');
            System.out.println("\n");
         }
      }
   }
}

////////////////////////////////////////////////////////////////////////////////
```