Computing selected topics

- Prácticas segundo parcial (Hormiga de Langton)-

Grupo 3CM8

Vargas Romero Erick Efraín
Prof. Juárez Martínez Genaro

Instituto Politécnico Nacional
Escuela Superior de Cómputo
Juan de Dios Bátiz, nueva industrial Vallejo
07738 ciudad de México

Contents

Chapter 1

Hormiga de Langton

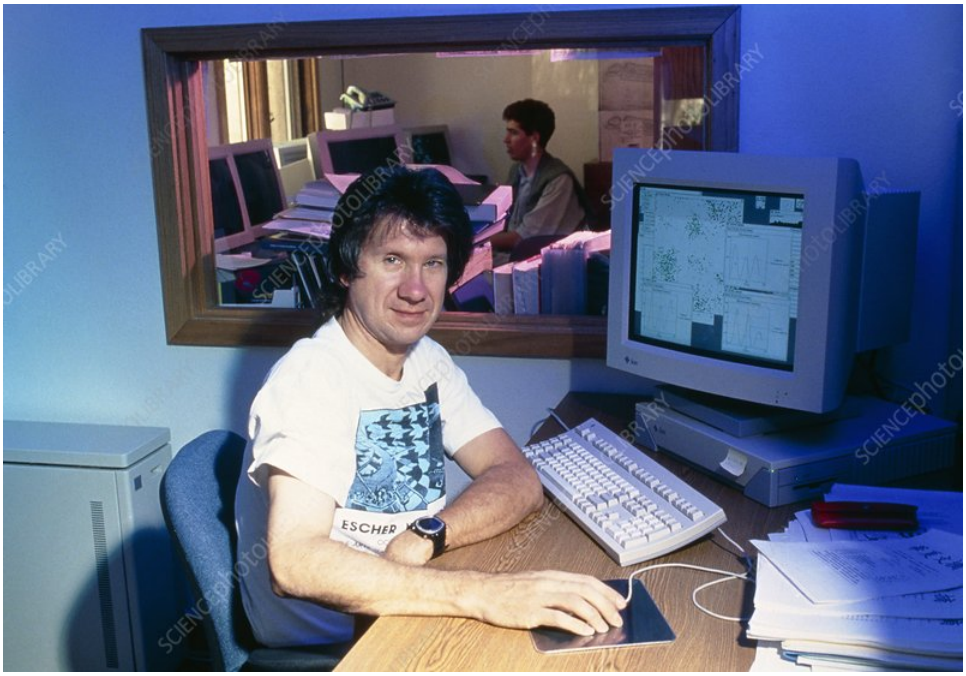
1.1 Introducción

La hormiga de Langton originalmente es una máquina de Turing de dos dimensiones, con un conjunto de reglas bastante simple, pero increíblemente es posible describir comportamientos complejos.

El inventor fue Chris Langton en 1986 y la universalidad fue demostrada en el año 2000

Es necesario mencionar que también hay una versión más de la hormiga de Langton la cual es descrita con un autómata celular, donde cada célula del espacio se pinta de negro o blanco y la hormiga se pinta de ocho colores diferentes dependiendo del color de la célula sobre la cual esté la hormiga y la dirección en que está mirando.

Figure 1.1: Christopher Langton



1.2 Comportamiento

Este autómata celular tiene solo tres reglas, las cuales son bastante simples, pero con tan solo estas reglas se pueden describir comportamientos complejos. Inicialmente el espacio es de un solo color, para nuestro caso es color negro y se presentan los siguientes comportamientos:

1. Simplicidad: Durante los primeros centenares de pasos la hormiga crea patrones muy sencillos y de forma general son simétricos.
2. Caos: Después de algunos cientos de pasos aparece un patrón grande y además irregular.
3. Orden emergente: La hormiga a partir de los diez mil pasos empieza a crear una avenida, el cual es un patrón de ciento cuatro pasos que se repite indefinidamente.

1.3 Reglas

Las reglas utilizadas en la hormiga de Langton son bastante sencillas y se mencionan a continuación:

- Si la hormiga está sobre una célula de color blanco, cambia el color de esta y la hormiga gira noventa grados a la derecha

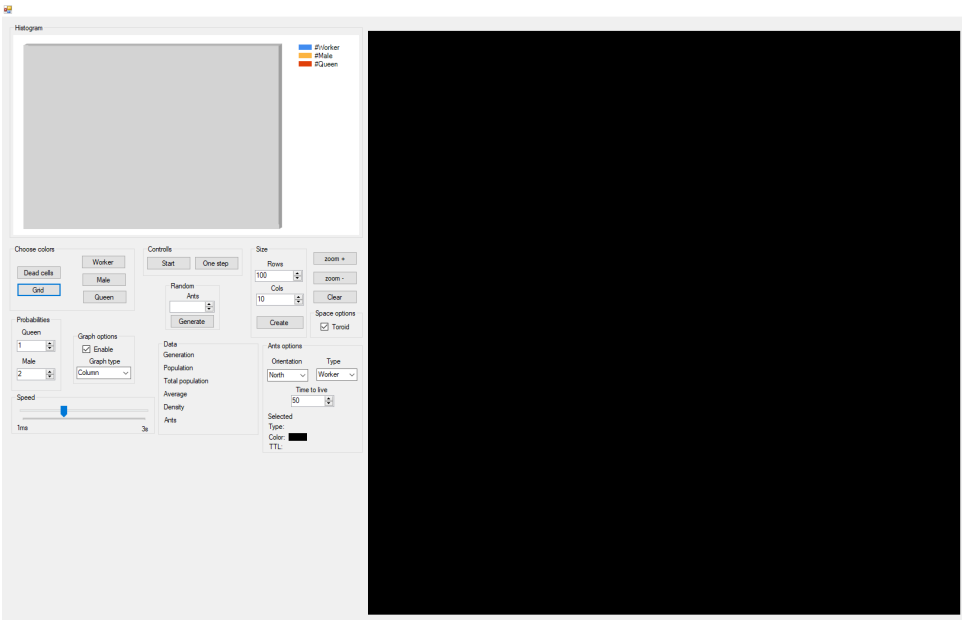
- Si la hormiga está sobre una célula de color negro, cambia el color de esta y la hormiga gira noventa grados a la izquierda

1.4 Práctica

1.4.1 Descripción y pruebas

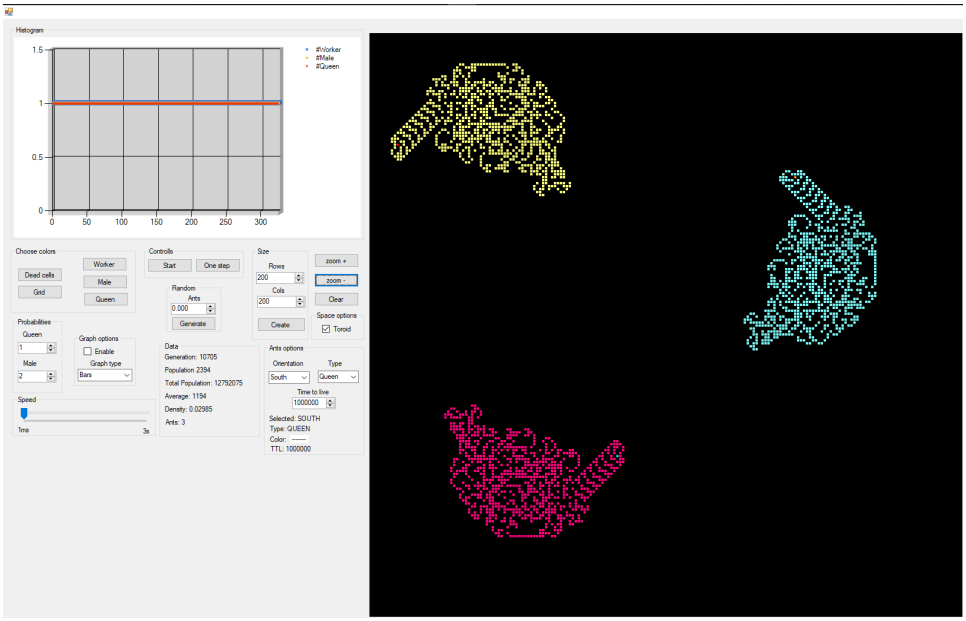
En palabras simples, para esta práctica se ha realizado la implementación de la hormiga de Langton, a continuación se muestra la interfaz de este autómata de forma general.

Figure 1.2: Interfaz



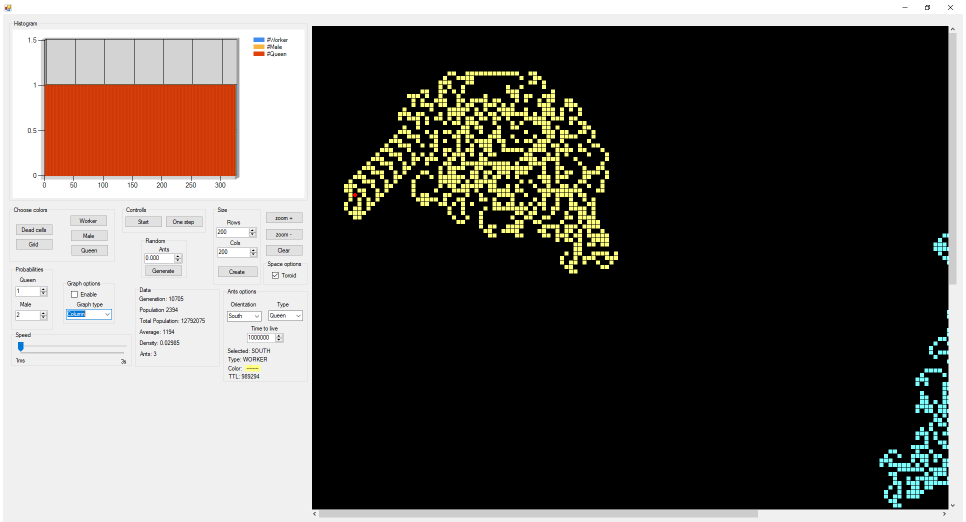
Para este autómata celular se han aplicado las reglas de la hormiga de Langton original, las cuales ya han sido descritas anteriormente, además de mostrar el comportamiento del autómata de forma gráfica También se puede observar la cantidad de células vivas que hay en el espacio en un histograma y el usuario tiene la libertad de elegir si desea mostrar un gráfico de columnas, barras o bien puntos. Esto se muestra en la figura siguiente

Figure 1.3: Prueba funcionamiento



Si se da un click derecho sobre alguna hormiga, las cuales se han representado con círculos en lugar de cuadrados, para ser mas fácil de distinguir, es posible elegir el color que tendrá el "rastro" que deje la hormiga. También tenemos un conjunto de elementos en la parte inferior, la cual está agrupada con la leyenda "Ants options", esta sección permite elegir la orientación que tendrá la hormiga que será añadida, el tipo de hormiga, su tiempo de vida y podrá observar la información de la hormiga sobre la cual está el mouse actualmente.

Figure 1.4: Información de las hormigas

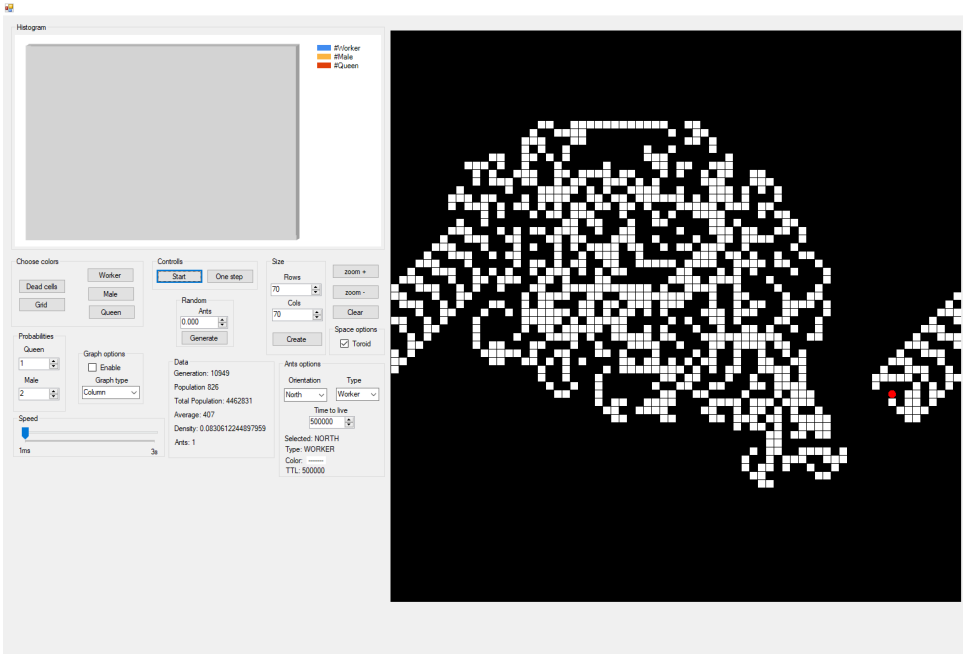


En este autómata el usuario tiene la libertad de elegir entre dos tipos de frontera:

- Frontera periódica: Las células que están en los límites o en la frontera interaccionan con sus vecinos inmediatos y con las células que están en el extremo opuesto del arreglo, como si el plano estuviese doblado a manera de cilindro.
- Frontera cerrada: No hay células más allá de los límites del espacio.

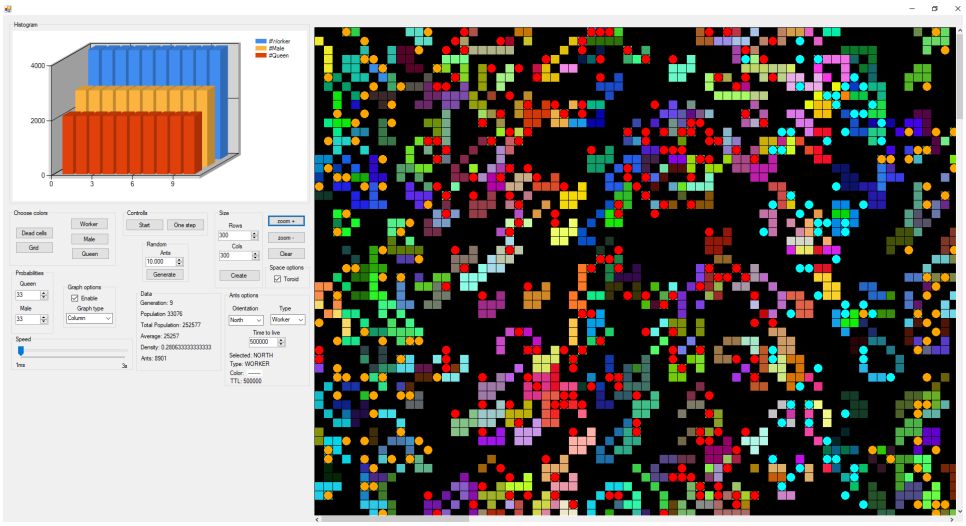
Esto se puede cambiar simplemente dando click en el check box que se encuentra en la sección con la leyenda "Space options" tal y como se encuentra en la siguiente imagen. Además podemos ver que en efecto el espacio mostrado es de frontera periódica, ya que las células vivas de color blanco ya están del extremo opuesto.

Figure 1.5: Cambio de frontera



También el usuario tiene la posibilidad de generar aleatoriamente hormigas en el espacio, pero no solo eso, es posible elegir la probabilidad con la que salgan hormigas macho y las hormigas reinas, como es evidente la diferencia entre el total, es decir cien por ciento y la suma de hormigas macho y hormigas reinas nos dará el porcentaje que se le asignará a las hormigas obreras. Esto se puede realizar en la sección con la leyenda "Probabilities". También es necesario el mencionar que el color del "rastro" que dejan las hormigas son diferentes. Lo antes mencionado se puede observar en la figura siguiente.

Figure 1.6: Aleatoriedad



Además de lo ya mencionado, las hormigas reinas podemos decir que son un caso bastante peculiar ya que se han añadido las siguientes funciones:

- Si una hormiga reina se encuentra en la misma posición que una hormiga macho, nace una nueva hormiga. La nueva hormiga puede ser una hormiga obrera, una hormiga macho o bien otra reina, esto según las probabilidades que se hayan establecido en el programa.

- Si una hormiga reina se encuentra en la misma posición que otra hormiga macho, una de las dos hormigas muere.

También es necesario mencionar que las hormigas tienen un tiempo de vida definido, para el caso de este programa el parámetro TTL es el que nos ayuda a definir esto. Si TTL llega a el valor de cero, la hormiga en cuestión, muere.

1.4.2 Código

A continuación se muestra el código generado para la implementación de este autómata celular. Primeramente tenemos el código auto generado por el entorno de desarrollo integrado (IDE) el cual fue Visual Studio, además de que cabe mencionar que el lenguaje utilizado fue C#. El código mostrado a continuación es de la interfaz gráfica.

```
1 namespace Langton_s_ant
2 {
3     partial class Form1
4     {
5         /// <summary>
6         /// Variable del ñdiseador necesaria.
7         /// </summary>
```

```
8           private System.ComponentModel.IContainer components = null;
9
10          /// <summary>
11          /// Limpiar los recursos que se éestn usando.
12          /// </summary>
13          /// <param name="disposing">true si los recursos administrados
                se deben desechar; false en caso contrario.</param>
14          protected override void Dispose(bool disposing)
15          {
16              if (disposing && (components != null))
17              {
18                  components.Dispose();
19              }
20              base.Dispose(disposing);
21          }
22
23          #region óCdigo generado por el ñDiseador de Windows Forms
24
25          /// <summary>
26          /// éMtodo necesario para admitir el ñDiseador. No se puede
                modificar
27          /// el contenido de este émtodo con el editor de ócdigo.
28          /// </summary>
29          private void InitializeComponent()
30          {
31              this.components = new System.ComponentModel.Container();
32              System.Windows.Forms.DataVisualization.Charting.ChartArea
                    chartArea1 = new System.Windows.Forms.DataVisualization.
                    Charting.ChartArea();
33              System.Windows.Forms.DataVisualization.Charting.Legend
                    legend1 = new System.Windows.Forms.DataVisualization.
                    Charting.Legend();
34              System.Windows.Forms.DataVisualization.Charting.Series
                    series1 = new System.Windows.Forms.DataVisualization.
                    Charting.Series();
35              System.Windows.Forms.DataVisualization.Charting.Series
                    series2 = new System.Windows.Forms.DataVisualization.
                    Charting.Series();
36              System.Windows.Forms.DataVisualization.Charting.Series
                    series3 = new System.Windows.Forms.DataVisualization.
                    Charting.Series();
37              this.PBAutomataSimulator = new System.Windows.Forms.
                    PictureBox();
38              this.CHHistogram = new System.Windows.Forms.
                    DataVisualization.Charting.Chart();
39              this.groupBox1 = new System.Windows.Forms.GroupBox();
40              this.BTNStart = new System.Windows.Forms.Button();
41              this.TXTGeneration = new System.Windows.Forms.Label();
42              this.TXTPopulation = new System.Windows.Forms.Label();
43              this.BTNStep = new System.Windows.Forms.Button();
44              this.TBSpeed = new System.Windows.Forms.TrackBar();
45              this.TimerSimulation = new System.Windows.Forms.Timer(this.
                    components);
```

```
46              this.flowLayoutPanel1 = new System.Windows.Forms.
                    FlowLayoutPanel();
47              this.groupBox3 = new System.Windows.Forms.GroupBox();
48              this.label6 = new System.Windows.Forms.Label();
49              this.label1 = new System.Windows.Forms.Label();
50              this.BTNZoomP = new System.Windows.Forms.Button();
51              this.BTNZoomM = new System.Windows.Forms.Button();
52              this.groupBox4 = new System.Windows.Forms.GroupBox();
53              this.groupBox5 = new System.Windows.Forms.GroupBox();
54              this.button1 = new System.Windows.Forms.Button();
55              this.numericOnes = new System.Windows.Forms.NumericUpDown();
56              this.Ants = new System.Windows.Forms.Label();
57              this.groupBox6 = new System.Windows.Forms.GroupBox();
58              this.BTNColorGrid = new System.Windows.Forms.Button();
59              this.BTNColorWorker = new System.Windows.Forms.Button();
60              this.BTNColorDead = new System.Windows.Forms.Button();
61              this.BTNClear = new System.Windows.Forms.Button();
62              this.groupBox7 = new System.Windows.Forms.GroupBox();
63              this.BTNCreateMatrix = new System.Windows.Forms.Button();
64              this.numericCols = new System.Windows.Forms.NumericUpDown();
65              this.numericRows = new System.Windows.Forms.NumericUpDown();
66              this.label11 = new System.Windows.Forms.Label();
67              this.label8 = new System.Windows.Forms.Label();
68              this.label12 = new System.Windows.Forms.Label();
69              this.label13 = new System.Windows.Forms.Label();
70              this.label14 = new System.Windows.Forms.Label();
71              this.groupBox9 = new System.Windows.Forms.GroupBox();
72              this.CheckGraphEnabled = new System.Windows.Forms.CheckBox()
                    ;
73              this.groupBox10 = new System.Windows.Forms.GroupBox();
74              this.label3 = new System.Windows.Forms.Label();
75              this.groupBox2 = new System.Windows.Forms.GroupBox();
76              this.CBToroid = new System.Windows.Forms.CheckBox();
77              this.groupBox11 = new System.Windows.Forms.GroupBox();
78              this.label17 = new System.Windows.Forms.Label();
79              this.label16 = new System.Windows.Forms.Label();
80              this.NumericTTL = new System.Windows.Forms.NumericUpDown();
81              this.label15 = new System.Windows.Forms.Label();
82              this.label10 = new System.Windows.Forms.Label();
83              this.Orientation = new System.Windows.Forms.Label();
84              this.ComboOrientation = new System.Windows.Forms.ComboBox();
85              this.ComboType = new System.Windows.Forms.ComboBox();
86              this.label2 = new System.Windows.Forms.Label();
87              this.label5 = new System.Windows.Forms.Label();
88              this.colorDialog = new System.Windows.Forms.ColorDialog();
89              this.groupBox12 = new System.Windows.Forms.GroupBox();
90              this.label9 = new System.Windows.Forms.Label();
91              this.label7 = new System.Windows.Forms.Label();
92              this.NumericMale = new System.Windows.Forms.NumericUpDown();
93              this.NumericQueen = new System.Windows.Forms.NumericUpDown()
                    ;
94              this.label18 = new System.Windows.Forms.Label();
95              this.BTNColorMale = new System.Windows.Forms.Button();
```

```
96              this.BTNColorQueen = new System.Windows.Forms.Button();
97              this.ComboTypeGraph = new System.Windows.Forms.ComboBox();
98              this.label4 = new System.Windows.Forms.Label();
99              ((System.ComponentModel.ISupportInitialize)(this.
                    PBAutomataSimulator)).BeginInit();
100             ((System.ComponentModel.ISupportInitialize)(this.CHHistogram
                    )).BeginInit();
101             this.groupBox1.SuspendLayout();
102             ((System.ComponentModel.ISupportInitialize)(this.TBSpeed)).
                    BeginInit();
103             this.flowLayoutPanel1.SuspendLayout();
104             this.groupBox3.SuspendLayout();
105             this.groupBox4.SuspendLayout();
106             this.groupBox5.SuspendLayout();
107             ((System.ComponentModel.ISupportInitialize)(this.numericOnes
                    )).BeginInit();
108             this.groupBox6.SuspendLayout();
109             this.groupBox7.SuspendLayout();
110             ((System.ComponentModel.ISupportInitialize)(this.numericCols
                    )).BeginInit();
111             ((System.ComponentModel.ISupportInitialize)(this.numericRows
                    )).BeginInit();
112             this.groupBox9.SuspendLayout();
113             this.groupBox10.SuspendLayout();
114             this.groupBox2.SuspendLayout();
115             this.groupBox11.SuspendLayout();
116             ((System.ComponentModel.ISupportInitialize)(this.NumericTTL)
                    ).BeginInit();
117             this.groupBox12.SuspendLayout();
118             ((System.ComponentModel.ISupportInitialize)(this.NumericMale
                    )).BeginInit();
119             ((System.ComponentModel.ISupportInitialize)(this.
                    NumericQueen)).BeginInit();
120             this.SuspendLayout();
121             //
122             // PBAutomataSimulator
123             //
124             this.PBAutomataSimulator.BackColor = System.Drawing.
                    SystemColors.ActiveCaptionText;
125             this.PBAutomataSimulator.Location = new System.Drawing.Point
                    (3, 3);
126             this.PBAutomataSimulator.Name = "PBAutomataSimulator";
127             this.PBAutomataSimulator.Size = new System.Drawing.Size(545,
                     505);
128             this.PBAutomataSimulator.SizeMode = System.Windows.Forms.
                    PictureBoxSizeMode.AutoSize;
129             this.PBAutomataSimulator.TabIndex = 1;
130             this.PBAutomataSimulator.TabStop = false;
131             this.PBAutomataSimulator.Click += new System.EventHandler(
                    this.PBAutomataSimulator_Click);
132             this.PBAutomataSimulator.Paint += new System.Windows.Forms.
                    PaintEventHandler(this.PBAutomataSimulator_Paint);
133             this.PBAutomataSimulator.MouseMove += new System.Windows.
```

```
            Forms.MouseEventHandler(this.
            PBAutomataSimulator_MouseMove);
134         //
135         // CHHistogram
136         //
137         chartArea1.AlignmentOrientation = ((System.Windows.Forms.
            DataVisualization.Charting.AreaAlignmentOrientations)((
            System.Windows.Forms.DataVisualization.Charting.
            AreaAlignmentOrientations.Vertical | System.Windows.
            Forms.DataVisualization.Charting.
            AreaAlignmentOrientations.Horizontal)));
138         chartArea1.Area3DStyle.Enable3D = true;
139         chartArea1.Name = "ChartArea1";
140         this.CHHistogram.ChartAreas.Add(chartArea1);
141         legend1.Name = "Legend1";
142         this.CHHistogram.Legends.Add(legend1);
143         this.CHHistogram.Location = new System.Drawing.Point(6, 19);
144         this.CHHistogram.Name = "CHHistogram";
145         series1.ChartArea = "ChartArea1";
146         series1.Legend = "Legend1";
147         series1.Name = "#Worker";
148         series1.YValuesPerPoint = 2;
149         series2.ChartArea = "ChartArea1";
150         series2.Legend = "Legend1";
151         series2.Name = "#Male";
152         series2.YValuesPerPoint = 2;
153         series3.ChartArea = "ChartArea1";
154         series3.Legend = "Legend1";
155         series3.Name = "#Queen";
156         series3.YValuesPerPoint = 6;
157         this.CHHistogram.Series.Add(series1);
158         this.CHHistogram.Series.Add(series2);
159         this.CHHistogram.Series.Add(series3);
160         this.CHHistogram.Size = new System.Drawing.Size(584, 337);
161         this.CHHistogram.TabIndex = 2;
162         this.CHHistogram.Text = "chart1";
163         //
164         // groupBox1
165         //
166         this.groupBox1.Controls.Add(this.CHHistogram);
167         this.groupBox1.Location = new System.Drawing.Point(13, 12);
168         this.groupBox1.Name = "groupBox1";
169         this.groupBox1.Size = new System.Drawing.Size(596, 362);
170         this.groupBox1.TabIndex = 3;
171         this.groupBox1.TabStop = false;
172         this.groupBox1.Text = "Histogram";
173         //
174         // BTNStart
175         //
176         this.BTNStart.Location = new System.Drawing.Point(6, 19);
177         this.BTNStart.Name = "BTNStart";
178         this.BTNStart.Size = new System.Drawing.Size(75, 23);
179         this.BTNStart.TabIndex = 4;
```

```
180             this.BTNStart.Text = "Start";
181             this.BTNStart.UseVisualStyleBackColor = true;
182             this.BTNStart.Click += new System.EventHandler(this.
                    BTNStart_Click);
183             //
184             // TXTGeneration
185             //
186             this.TXTGeneration.AutoSize = true;
187             this.TXTGeneration.BackColor = System.Drawing.Color.
                    Transparent;
188             this.TXTGeneration.ForeColor = System.Drawing.Color.Black;
189             this.TXTGeneration.Location = new System.Drawing.Point(6,
                    18);
190             this.TXTGeneration.Name = "TXTGeneration";
191             this.TXTGeneration.Size = new System.Drawing.Size(62, 13);
192             this.TXTGeneration.TabIndex = 7;
193             this.TXTGeneration.Text = "Generation ";
194             //
195             // TXTPopulation
196             //
197             this.TXTPopulation.AutoSize = true;
198             this.TXTPopulation.Location = new System.Drawing.Point(6,
                    40);
199             this.TXTPopulation.Name = "TXTPopulation";
200             this.TXTPopulation.Size = new System.Drawing.Size(57, 13);
201             this.TXTPopulation.TabIndex = 8;
202             this.TXTPopulation.Text = "Population";
203             //
204             // BTNStep
205             //
206             this.BTNStep.Location = new System.Drawing.Point(87, 19);
207             this.BTNStep.Name = "BTNStep";
208             this.BTNStep.Size = new System.Drawing.Size(75, 23);
209             this.BTNStep.TabIndex = 9;
210             this.BTNStep.Text = "One step";
211             this.BTNStep.UseVisualStyleBackColor = true;
212             this.BTNStep.Click += new System.EventHandler(this.
                    BTNStep_Click);
213             //
214             // TBSpeed
215             //
216             this.TBSpeed.Location = new System.Drawing.Point(6, 19);
217             this.TBSpeed.Maximum = 3000;
218             this.TBSpeed.Minimum = 1;
219             this.TBSpeed.Name = "TBSpeed";
220             this.TBSpeed.Size = new System.Drawing.Size(233, 45);
221             this.TBSpeed.TabIndex = 10;
222             this.TBSpeed.Value = 1000;
223             this.TBSpeed.ValueChanged += new System.EventHandler(this.
                    trackBar1_ValueChanged);
224             //
225             // TimerSimulation
226             //
```

```
227             this.TimerSimulation.Tick += new System.EventHandler(this.
                    TimerSimulation_Tick);
228             //
229             // flowLayoutPanel1
230             //
231             this.flowLayoutPanel1.Anchor = ((System.Windows.Forms.
                    AnchorStyles)((((System.Windows.Forms.AnchorStyles.Top |
                     System.Windows.Forms.AnchorStyles.Bottom)
232             | System.Windows.Forms.AnchorStyles.Left)
233             | System.Windows.Forms.AnchorStyles.Right)));
234             this.flowLayoutPanel1.AutoScroll = true;
235             this.flowLayoutPanel1.AutoSizeMode = System.Windows.Forms.
                    AutoSizeMode.GrowAndShrink;
236             this.flowLayoutPanel1.Controls.Add(this.PBAutomataSimulator)
                    ;
237             this.flowLayoutPanel1.Location = new System.Drawing.Point
                    (615, 21);
238             this.flowLayoutPanel1.MinimumSize = new System.Drawing.Size
                    (639, 600);
239             this.flowLayoutPanel1.Name = "flowLayoutPanel1";
240             this.flowLayoutPanel1.Size = new System.Drawing.Size(639,
                    710);
241             this.flowLayoutPanel1.TabIndex = 11;
242             //
243             // groupBox3
244             //
245             this.groupBox3.Controls.Add(this.label6);
246             this.groupBox3.Controls.Add(this.label1);
247             this.groupBox3.Controls.Add(this.TBSpeed);
248             this.groupBox3.Location = new System.Drawing.Point(16, 635);
249             this.groupBox3.Name = "groupBox3";
250             this.groupBox3.Size = new System.Drawing.Size(242, 69);
251             this.groupBox3.TabIndex = 12;
252             this.groupBox3.TabStop = false;
253             this.groupBox3.Text = "Speed";
254             //
255             // label6
256             //
257             this.label6.AutoSize = true;
258             this.label6.Location = new System.Drawing.Point(217, 53);
259             this.label6.Name = "label6";
260             this.label6.Size = new System.Drawing.Size(18, 13);
261             this.label6.TabIndex = 12;
262             this.label6.Text = "3s";
263             //
264             // label1
265             //
266             this.label1.AutoSize = true;
267             this.label1.Location = new System.Drawing.Point(6, 51);
268             this.label1.Name = "label1";
269             this.label1.Size = new System.Drawing.Size(26, 13);
270             this.label1.TabIndex = 11;
271             this.label1.Text = "1ms";
```

```
272             //
273             // BTNZoomP
274             //
275             this.BTNZoomP.Location = new System.Drawing.Point(525, 396);
276             this.BTNZoomP.Name = "BTNZoomP";
277             this.BTNZoomP.Size = new System.Drawing.Size(75, 23);
278             this.BTNZoomP.TabIndex = 13;
279             this.BTNZoomP.Text = "zoom +";
280             this.BTNZoomP.UseVisualStyleBackColor = true;
281             this.BTNZoomP.Click += new System.EventHandler(this.
                    BTNZoomP_Click);
282             //
283             // BTNZoomM
284             //
285             this.BTNZoomM.Location = new System.Drawing.Point(525, 429);
286             this.BTNZoomM.Name = "BTNZoomM";
287             this.BTNZoomM.Size = new System.Drawing.Size(75, 23);
288             this.BTNZoomM.TabIndex = 14;
289             this.BTNZoomM.Text = "zoom -";
290             this.BTNZoomM.UseVisualStyleBackColor = true;
291             this.BTNZoomM.Click += new System.EventHandler(this.
                    BTNZoomM_Click);
292             //
293             // groupBox4
294             //
295             this.groupBox4.Controls.Add(this.BTNStart);
296             this.groupBox4.Controls.Add(this.BTNStep);
297             this.groupBox4.Location = new System.Drawing.Point(238, 385)
                    ;
298             this.groupBox4.Name = "groupBox4";
299             this.groupBox4.Size = new System.Drawing.Size(168, 53);
300             this.groupBox4.TabIndex = 15;
301             this.groupBox4.TabStop = false;
302             this.groupBox4.Text = "Controlls";
303             //
304             // groupBox5
305             //
306             this.groupBox5.Controls.Add(this.button1);
307             this.groupBox5.Controls.Add(this.numericOnes);
308             this.groupBox5.Controls.Add(this.Ants);
309             this.groupBox5.Location = new System.Drawing.Point(276, 447)
                    ;
310             this.groupBox5.Name = "groupBox5";
311             this.groupBox5.Size = new System.Drawing.Size(92, 82);
312             this.groupBox5.TabIndex = 16;
313             this.groupBox5.TabStop = false;
314             this.groupBox5.Text = "Random";
315             //
316             // button1
317             //
318             this.button1.Location = new System.Drawing.Point(8, 55);
319             this.button1.Name = "button1";
320             this.button1.Size = new System.Drawing.Size(75, 23);
```

```
321             this.button1.TabIndex = 4;
322             this.button1.Text = "Generate";
323             this.button1.UseVisualStyleBackColor = true;
324             this.button1.Click += new System.EventHandler(this.
                    button1_Click);
325             //
326             // numericOnes
327             //
328             this.numericOnes.DecimalPlaces = 3;
329             this.numericOnes.Location = new System.Drawing.Point(7, 32);
330             this.numericOnes.Maximum = new decimal(new int[] {
331             10,
332             0,
333             0,
334             0});
335             this.numericOnes.Name = "numericOnes";
336             this.numericOnes.Size = new System.Drawing.Size(76, 20);
337             this.numericOnes.TabIndex = 2;
338             //
339             // Ants
340             //
341             this.Ants.AutoSize = true;
342             this.Ants.Location = new System.Drawing.Point(33, 17);
343             this.Ants.Name = "Ants";
344             this.Ants.Size = new System.Drawing.Size(28, 13);
345             this.Ants.TabIndex = 0;
346             this.Ants.Text = "Ants";
347             //
348             // groupBox6
349             //
350             this.groupBox6.Controls.Add(this.BTNColorQueen);
351             this.groupBox6.Controls.Add(this.BTNColorMale);
352             this.groupBox6.Controls.Add(this.BTNColorGrid);
353             this.groupBox6.Controls.Add(this.BTNColorWorker);
354             this.groupBox6.Controls.Add(this.BTNColorDead);
355             this.groupBox6.Location = new System.Drawing.Point(13, 385);
356             this.groupBox6.Name = "groupBox6";
357             this.groupBox6.Size = new System.Drawing.Size(211, 112);
358             this.groupBox6.TabIndex = 17;
359             this.groupBox6.TabStop = false;
360             this.groupBox6.Text = "Choose colors";
361             //
362             // BTNColorGrid
363             //
364             this.BTNColorGrid.Location = new System.Drawing.Point(12,
                    64);
365             this.BTNColorGrid.Name = "BTNColorGrid";
366             this.BTNColorGrid.Size = new System.Drawing.Size(75, 23);
367             this.BTNColorGrid.TabIndex = 10;
368             this.BTNColorGrid.Text = "Grid";
369             this.BTNColorGrid.UseVisualStyleBackColor = true;
370             this.BTNColorGrid.Click += new System.EventHandler(this.
                    BTNColorGrid_Click);
```

```
371             //
372             // BTNColorWorker
373             //
374             this.BTNColorWorker.Location = new System.Drawing.Point(121,
                    17);
375             this.BTNColorWorker.Name = "BTNColorWorker";
376             this.BTNColorWorker.Size = new System.Drawing.Size(75, 23);
377             this.BTNColorWorker.TabIndex = 9;
378             this.BTNColorWorker.Text = "Worker";
379             this.BTNColorWorker.UseVisualStyleBackColor = true;
380             this.BTNColorWorker.Click += new System.EventHandler(this.
                    BTNColorAnt_Click);
381             //
382             // BTNColorDead
383             //
384             this.BTNColorDead.Location = new System.Drawing.Point(12,
                    35);
385             this.BTNColorDead.Name = "BTNColorDead";
386             this.BTNColorDead.Size = new System.Drawing.Size(75, 23);
387             this.BTNColorDead.TabIndex = 8;
388             this.BTNColorDead.Text = "Dead cells";
389             this.BTNColorDead.UseVisualStyleBackColor = true;
390             this.BTNColorDead.Click += new System.EventHandler(this.
                    BTNColorDead_Click);
391             //
392             // BTNClear
393             //
394             this.BTNClear.Location = new System.Drawing.Point(525, 461);
395             this.BTNClear.Name = "BTNClear";
396             this.BTNClear.Size = new System.Drawing.Size(75, 23);
397             this.BTNClear.TabIndex = 19;
398             this.BTNClear.Text = "Clear";
399             this.BTNClear.UseVisualStyleBackColor = true;
400             this.BTNClear.Click += new System.EventHandler(this.
                    BTNClear_Click);
401             //
402             // groupBox7
403             //
404             this.groupBox7.Controls.Add(this.BTNCreateMatrix);
405             this.groupBox7.Controls.Add(this.numericCols);
406             this.groupBox7.Controls.Add(this.numericRows);
407             this.groupBox7.Controls.Add(this.label11);
408             this.groupBox7.Controls.Add(this.label8);
409             this.groupBox7.Location = new System.Drawing.Point(420, 385)
                    ;
410             this.groupBox7.Name = "groupBox7";
411             this.groupBox7.Size = new System.Drawing.Size(95, 157);
412             this.groupBox7.TabIndex = 20;
413             this.groupBox7.TabStop = false;
414             this.groupBox7.Text = "Size";
415             //
416             // BTNCreateMatrix
417             //
```

```
418             this.BTNCreateMatrix.Location = new System.Drawing.Point(9,
                    119);
419             this.BTNCreateMatrix.Name = "BTNCreateMatrix";
420             this.BTNCreateMatrix.Size = new System.Drawing.Size(81, 23);
421             this.BTNCreateMatrix.TabIndex = 21;
422             this.BTNCreateMatrix.Text = "Create";
423             this.BTNCreateMatrix.UseVisualStyleBackColor = true;
424             this.BTNCreateMatrix.Click += new System.EventHandler(this.
                    BTNCreateMatrix_Click);
425             //
426             // numericCols
427             //
428             this.numericCols.Location = new System.Drawing.Point(9, 82);
429             this.numericCols.Maximum = new decimal(new int[] {
430             1000,
431             0,
432             0,
433             0});
434             this.numericCols.Minimum = new decimal(new int[] {
435             10,
436             0,
437             0,
438             0});
439             this.numericCols.Name = "numericCols";
440             this.numericCols.Size = new System.Drawing.Size(81, 20);
441             this.numericCols.TabIndex = 3;
442             this.numericCols.Value = new decimal(new int[] {
443             10,
444             0,
445             0,
446             0});
447             //
448             // numericRows
449             //
450             this.numericRows.Location = new System.Drawing.Point(7, 42);
451             this.numericRows.Maximum = new decimal(new int[] {
452             1000,
453             0,
454             0,
455             0});
456             this.numericRows.Minimum = new decimal(new int[] {
457             10,
458             0,
459             0,
460             0});
461             this.numericRows.Name = "numericRows";
462             this.numericRows.Size = new System.Drawing.Size(81, 20);
463             this.numericRows.TabIndex = 2;
464             this.numericRows.Value = new decimal(new int[] {
465             100,
466             0,
467             0,
468             0});
```

```
469             //
470             // label11
471             //
472             this.label11.AutoSize = true;
473             this.label11.Location = new System.Drawing.Point(32, 66);
474             this.label11.Name = "label11";
475             this.label11.Size = new System.Drawing.Size(27, 13);
476             this.label11.TabIndex = 1;
477             this.label11.Text = "Cols";
478             //
479             // label8
480             //
481             this.label8.AutoSize = true;
482             this.label8.Location = new System.Drawing.Point(25, 25);
483             this.label8.Name = "label8";
484             this.label8.Size = new System.Drawing.Size(34, 13);
485             this.label8.TabIndex = 0;
486             this.label8.Text = "Rows";
487             //
488             // label12
489             //
490             this.label12.AutoSize = true;
491             this.label12.Location = new System.Drawing.Point(6, 62);
492             this.label12.Name = "label12";
493             this.label12.Size = new System.Drawing.Size(83, 13);
494             this.label12.TabIndex = 22;
495             this.label12.Text = "Total population";
496             //
497             // label13
498             //
499             this.label13.AutoSize = true;
500             this.label13.Location = new System.Drawing.Point(6, 84);
501             this.label13.Name = "label13";
502             this.label13.Size = new System.Drawing.Size(47, 13);
503             this.label13.TabIndex = 23;
504             this.label13.Text = "Average";
505             //
506             // label14
507             //
508             this.label14.AutoSize = true;
509             this.label14.Location = new System.Drawing.Point(6, 106);
510             this.label14.Name = "label14";
511             this.label14.Size = new System.Drawing.Size(42, 13);
512             this.label14.TabIndex = 24;
513             this.label14.Text = "Density";
514             //
515             // groupBox9
516             //
517             this.groupBox9.Controls.Add(this.label4);
518             this.groupBox9.Controls.Add(this.ComboTypeGraph);
519             this.groupBox9.Controls.Add(this.CheckGraphEnabled);
520             this.groupBox9.Location = new System.Drawing.Point(120, 532)
                ;
```

```
521             this.groupBox9.Name = "groupBox9";
522             this.groupBox9.Size = new System.Drawing.Size(104, 88);
523             this.groupBox9.TabIndex = 25;
524             this.groupBox9.TabStop = false;
525             this.groupBox9.Text = "Graph options";
526             //
527             // CheckGraphEnabled
528             //
529             this.CheckGraphEnabled.AutoSize = true;
530             this.CheckGraphEnabled.Checked = true;
531             this.CheckGraphEnabled.CheckState = System.Windows.Forms.
                    CheckState.Checked;
532             this.CheckGraphEnabled.Location = new System.Drawing.Point
                    (16, 21);
533             this.CheckGraphEnabled.Name = "CheckGraphEnabled";
534             this.CheckGraphEnabled.Size = new System.Drawing.Size(59,
                    17);
535             this.CheckGraphEnabled.TabIndex = 0;
536             this.CheckGraphEnabled.Text = "Enable";
537             this.CheckGraphEnabled.UseVisualStyleBackColor = true;
538             //
539             // groupBox10
540             //
541             this.groupBox10.Controls.Add(this.label3);
542             this.groupBox10.Controls.Add(this.TXTGeneration);
543             this.groupBox10.Controls.Add(this.TXTPopulation);
544             this.groupBox10.Controls.Add(this.label14);
545             this.groupBox10.Controls.Add(this.label12);
546             this.groupBox10.Controls.Add(this.label13);
547             this.groupBox10.Location = new System.Drawing.Point(264,
                    545);
548             this.groupBox10.Name = "groupBox10";
549             this.groupBox10.Size = new System.Drawing.Size(169, 161);
550             this.groupBox10.TabIndex = 26;
551             this.groupBox10.TabStop = false;
552             this.groupBox10.Text = "Data";
553             //
554             // label3
555             //
556             this.label3.AutoSize = true;
557             this.label3.Location = new System.Drawing.Point(7, 128);
558             this.label3.Name = "label3";
559             this.label3.Size = new System.Drawing.Size(28, 13);
560             this.label3.TabIndex = 25;
561             this.label3.Text = "Ants";
562             //
563             // groupBox2
564             //
565             this.groupBox2.Controls.Add(this.CBToroid);
566             this.groupBox2.Location = new System.Drawing.Point(520, 493)
                    ;
567             this.groupBox2.Name = "groupBox2";
568             this.groupBox2.Size = new System.Drawing.Size(89, 49);
```

```
569             this.groupBox2.TabIndex = 27;
570             this.groupBox2.TabStop = false;
571             this.groupBox2.Text = "Space options";
572             //
573             // CBToroid
574             //
575             this.CBToroid.AutoSize = true;
576             this.CBToroid.Checked = true;
577             this.CBToroid.CheckState = System.Windows.Forms.CheckState.
                    Checked;
578             this.CBToroid.Location = new System.Drawing.Point(18, 22);
579             this.CBToroid.Name = "CBToroid";
580             this.CBToroid.Size = new System.Drawing.Size(56, 17);
581             this.CBToroid.TabIndex = 0;
582             this.CBToroid.Text = "Toroid";
583             this.CBToroid.UseVisualStyleBackColor = true;
584             //
585             // groupBox11
586             //
587             this.groupBox11.Controls.Add(this.label18);
588             this.groupBox11.Controls.Add(this.label17);
589             this.groupBox11.Controls.Add(this.label16);
590             this.groupBox11.Controls.Add(this.NumericTTL);
591             this.groupBox11.Controls.Add(this.label15);
592             this.groupBox11.Controls.Add(this.label10);
593             this.groupBox11.Controls.Add(this.Orientation);
594             this.groupBox11.Controls.Add(this.ComboOrientation);
595             this.groupBox11.Controls.Add(this.ComboType);
596             this.groupBox11.Controls.Add(this.label2);
597             this.groupBox11.Controls.Add(this.label5);
598             this.groupBox11.Location = new System.Drawing.Point(440,
                    548);
599             this.groupBox11.Name = "groupBox11";
600             this.groupBox11.Size = new System.Drawing.Size(169, 188);
601             this.groupBox11.TabIndex = 28;
602             this.groupBox11.TabStop = false;
603             this.groupBox11.Text = "Ants options";
604             //
605             // label17
606             //
607             this.label17.AutoSize = true;
608             this.label17.Location = new System.Drawing.Point(8, 170);
609             this.label17.Name = "label17";
610             this.label17.Size = new System.Drawing.Size(30, 13);
611             this.label17.TabIndex = 13;
612             this.label17.Text = "TTL:";
613             //
614             // label16
615             //
616             this.label16.AutoSize = true;
617             this.label16.Location = new System.Drawing.Point(8, 154);
618             this.label16.Name = "label16";
619             this.label16.Size = new System.Drawing.Size(34, 13);
```

```
620             this.label16.TabIndex = 12;
621             this.label16.Text = "Color:";
622             //
623             // NumericTTL
624             //
625             this.NumericTTL.Location = new System.Drawing.Point(48, 90);
626             this.NumericTTL.Maximum = new decimal(new int[] {
627             1000000,
628             0,
629             0,
630             0});
631             this.NumericTTL.Minimum = new decimal(new int[] {
632             40,
633             0,
634             0,
635             0});
636             this.NumericTTL.Name = "NumericTTL";
637             this.NumericTTL.Size = new System.Drawing.Size(73, 20);
638             this.NumericTTL.TabIndex = 0;
639             this.NumericTTL.Value = new decimal(new int[] {
640             50,
641             0,
642             0,
643             0});
644             //
645             // label15
646             //
647             this.label15.AutoSize = true;
648             this.label15.Location = new System.Drawing.Point(53, 74);
649             this.label15.Name = "label15";
650             this.label15.Size = new System.Drawing.Size(61, 13);
651             this.label15.TabIndex = 11;
652             this.label15.Text = "Time to live";
653             //
654             // label10
655             //
656             this.label10.AutoSize = true;
657             this.label10.Location = new System.Drawing.Point(105, 26);
658             this.label10.Name = "label10";
659             this.label10.Size = new System.Drawing.Size(31, 13);
660             this.label10.TabIndex = 10;
661             this.label10.Text = "Type";
662             //
663             // Orientation
664             //
665             this.Orientation.AutoSize = true;
666             this.Orientation.Location = new System.Drawing.Point(13, 26)
                    ;
667             this.Orientation.Name = "Orientation";
668             this.Orientation.Size = new System.Drawing.Size(58, 13);
669             this.Orientation.TabIndex = 9;
670             this.Orientation.Text = "Orientation";
671             //
```

```
672             // ComboOrientation
673             //
674             this.ComboOrientation.AutoCompleteCustomSource.AddRange(new
                    string[] {
675             "North",
676             "East",
677             "South",
678             "West"});
679             this.ComboOrientation.FormattingEnabled = true;
680             this.ComboOrientation.Items.AddRange(new object[] {
681             "North",
682             "East",
683             "South",
684             "West"});
685             this.ComboOrientation.Location = new System.Drawing.Point(8,
                     46);
686             this.ComboOrientation.Name = "ComboOrientation";
687             this.ComboOrientation.Size = new System.Drawing.Size(69, 21)
                    ;
688             this.ComboOrientation.TabIndex = 8;
689             this.ComboOrientation.Text = "North";
690             //
691             // ComboType
692             //
693             this.ComboType.AutoCompleteCustomSource.AddRange(new string
                    [] {
694             "Worker",
695             "Male",
696             "Queen"});
697             this.ComboType.FormattingEnabled = true;
698             this.ComboType.Items.AddRange(new object[] {
699             "Worker",
700             "Male",
701             "Queen"});
702             this.ComboType.Location = new System.Drawing.Point(91, 45);
703             this.ComboType.Name = "ComboType";
704             this.ComboType.Size = new System.Drawing.Size(69, 21);
705             this.ComboType.TabIndex = 7;
706             this.ComboType.Text = "Worker";
707             //
708             // label2
709             //
710             this.label2.AutoSize = true;
711             this.label2.Location = new System.Drawing.Point(7, 136);
712             this.label2.Name = "label2";
713             this.label2.Size = new System.Drawing.Size(34, 13);
714             this.label2.TabIndex = 6;
715             this.label2.Text = "Type:";
716             //
717             // label5
718             //
719             this.label5.AutoSize = true;
720             this.label5.Location = new System.Drawing.Point(6, 119);
```

```
721             this.label5.Name = "label5";
722             this.label5.Size = new System.Drawing.Size(55, 13);
723             this.label5.TabIndex = 5;
724             this.label5.Text = "Selected: ";
725             //
726             // groupBox12
727             //
728             this.groupBox12.Controls.Add(this.label9);
729             this.groupBox12.Controls.Add(this.label7);
730             this.groupBox12.Controls.Add(this.NumericMale);
731             this.groupBox12.Controls.Add(this.NumericQueen);
732             this.groupBox12.Location = new System.Drawing.Point(16, 504)
                    ;
733             this.groupBox12.Name = "groupBox12";
734             this.groupBox12.Size = new System.Drawing.Size(84, 128);
735             this.groupBox12.TabIndex = 29;
736             this.groupBox12.TabStop = false;
737             this.groupBox12.Text = "Probabilities";
738             //
739             // label9
740             //
741             this.label9.AutoSize = true;
742             this.label9.Location = new System.Drawing.Point(15, 70);
743             this.label9.Name = "label9";
744             this.label9.Size = new System.Drawing.Size(30, 13);
745             this.label9.TabIndex = 3;
746             this.label9.Text = "Male";
747             //
748             // label7
749             //
750             this.label7.AutoSize = true;
751             this.label7.Location = new System.Drawing.Point(15, 21);
752             this.label7.Name = "label7";
753             this.label7.Size = new System.Drawing.Size(39, 13);
754             this.label7.TabIndex = 2;
755             this.label7.Text = "Queen";
756             //
757             // NumericMale
758             //
759             this.NumericMale.Location = new System.Drawing.Point(9, 89);
760             this.NumericMale.Name = "NumericMale";
761             this.NumericMale.Size = new System.Drawing.Size(65, 20);
762             this.NumericMale.TabIndex = 1;
763             this.NumericMale.Value = new decimal(new int[] {
764             2,
765             0,
766             0,
767             0});
768             //
769             // NumericQueen
770             //
771             this.NumericQueen.Location = new System.Drawing.Point(9, 41)
                    ;
```

```
772             this.NumericQueen.Name = "NumericQueen";
773             this.NumericQueen.Size = new System.Drawing.Size(65, 20);
774             this.NumericQueen.TabIndex = 0;
775             this.NumericQueen.Value = new decimal(new int[] {
776             1,
777             0,
778             0,
779             0});
780             //
781             // label18
782             //
783             this.label18.AutoSize = true;
784             this.label18.Location = new System.Drawing.Point(44, 154);
785             this.label18.Name = "label18";
786             this.label18.Size = new System.Drawing.Size(31, 13);
787             this.label18.TabIndex = 14;
788             this.label18.Text = "--------";
789             //
790             // BTNColorMale
791             //
792             this.BTNColorMale.Location = new System.Drawing.Point(122,
                    47);
793             this.BTNColorMale.Name = "BTNColorMale";
794             this.BTNColorMale.Size = new System.Drawing.Size(75, 23);
795             this.BTNColorMale.TabIndex = 11;
796             this.BTNColorMale.Text = "Male";
797             this.BTNColorMale.UseVisualStyleBackColor = true;
798             this.BTNColorMale.Click += new System.EventHandler(this.
                    BTNColorMale_Click);
799             //
800             // BTNColorQueen
801             //
802             this.BTNColorQueen.Location = new System.Drawing.Point(123,
                    76);
803             this.BTNColorQueen.Name = "BTNColorQueen";
804             this.BTNColorQueen.Size = new System.Drawing.Size(75, 23);
805             this.BTNColorQueen.TabIndex = 12;
806             this.BTNColorQueen.Text = "Queen";
807             this.BTNColorQueen.UseVisualStyleBackColor = true;
808             this.BTNColorQueen.Click += new System.EventHandler(this.
                    BTNColorQueen_Click);
809             //
810             // ComboTypeGraph
811             //
812             this.ComboTypeGraph.AutoCompleteCustomSource.AddRange(new
                    string[] {
813             "Bars",
814             "Pie",
815             "Points"});
816             this.ComboTypeGraph.FormattingEnabled = true;
817             this.ComboTypeGraph.Items.AddRange(new object[] {
818             "Column",
819             "Bars",
```

```
820             "Points"});
821             this.ComboTypeGraph.Location = new System.Drawing.Point(6,
                    58);
822             this.ComboTypeGraph.Name = "ComboTypeGraph";
823             this.ComboTypeGraph.Size = new System.Drawing.Size(92, 21);
824             this.ComboTypeGraph.TabIndex = 3;
825             this.ComboTypeGraph.Text = "Column";
826             this.ComboTypeGraph.SelectedIndexChanged += new System.
                    EventHandler(this.ComboTypeGraph_SelectedIndexChanged);
827             //
828             // label4
829             //
830             this.label4.AutoSize = true;
831             this.label4.Location = new System.Drawing.Point(25, 42);
832             this.label4.Name = "label4";
833             this.label4.Size = new System.Drawing.Size(59, 13);
834             this.label4.TabIndex = 4;
835             this.label4.Text = "Graph type";
836             //
837             // Form1
838             //
839             this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F)
                    ;
840             this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font
                    ;
841             this.ClientSize = new System.Drawing.Size(1266, 740);
842             this.Controls.Add(this.groupBox12);
843             this.Controls.Add(this.groupBox11);
844             this.Controls.Add(this.groupBox2);
845             this.Controls.Add(this.groupBox10);
846             this.Controls.Add(this.groupBox9);
847             this.Controls.Add(this.groupBox7);
848             this.Controls.Add(this.BTNClear);
849             this.Controls.Add(this.groupBox6);
850             this.Controls.Add(this.groupBox5);
851             this.Controls.Add(this.groupBox4);
852             this.Controls.Add(this.BTNZoomM);
853             this.Controls.Add(this.BTNZoomP);
854             this.Controls.Add(this.groupBox3);
855             this.Controls.Add(this.groupBox1);
856             this.Controls.Add(this.flowLayoutPanel1);
857             this.Name = "Form1";
858             this.Text = " ";
859             ((System.ComponentModel.ISupportInitialize)(this.
                    PBAutomataSimulator)).EndInit();
860             ((System.ComponentModel.ISupportInitialize)(this.CHHistogram
                    )).EndInit();
861             this.groupBox1.ResumeLayout(false);
862             ((System.ComponentModel.ISupportInitialize)(this.TBSpeed)).
                    EndInit();
863             this.flowLayoutPanel1.ResumeLayout(false);
864             this.flowLayoutPanel1.PerformLayout();
865             this.groupBox3.ResumeLayout(false);
```

```
866             this.groupBox3.PerformLayout();
867             this.groupBox4.ResumeLayout(false);
868             this.groupBox5.ResumeLayout(false);
869             this.groupBox5.PerformLayout();
870             ((System.ComponentModel.ISupportInitialize)(this.numericOnes
                    )).EndInit();
871             this.groupBox6.ResumeLayout(false);
872             this.groupBox7.ResumeLayout(false);
873             this.groupBox7.PerformLayout();
874             ((System.ComponentModel.ISupportInitialize)(this.numericCols
                    )).EndInit();
875             ((System.ComponentModel.ISupportInitialize)(this.numericRows
                    )).EndInit();
876             this.groupBox9.ResumeLayout(false);
877             this.groupBox9.PerformLayout();
878             this.groupBox10.ResumeLayout(false);
879             this.groupBox10.PerformLayout();
880             this.groupBox2.ResumeLayout(false);
881             this.groupBox2.PerformLayout();
882             this.groupBox11.ResumeLayout(false);
883             this.groupBox11.PerformLayout();
884             ((System.ComponentModel.ISupportInitialize)(this.NumericTTL)
                    ).EndInit();
885             this.groupBox12.ResumeLayout(false);
886             this.groupBox12.PerformLayout();
887             ((System.ComponentModel.ISupportInitialize)(this.NumericMale
                    )).EndInit();
888             ((System.ComponentModel.ISupportInitialize)(this.
                    NumericQueen)).EndInit();
889             this.ResumeLayout(false);
890
891         }
892
893         #endregion
894
895         private System.Windows.Forms.PictureBox PBAutomataSimulator;
896         private System.Windows.Forms.DataVisualization.Charting.Chart
                CHHistogram;
897         private System.Windows.Forms.GroupBox groupBox1;
898         private System.Windows.Forms.Button BTNStart;
899         private System.Windows.Forms.Label TXTGeneration;
900         private System.Windows.Forms.Label TXTPopulation;
901         private System.Windows.Forms.Button BTNStep;
902         private System.Windows.Forms.TrackBar TBSpeed;
903         private System.Windows.Forms.Timer TimerSimulation;
904         private System.Windows.Forms.GroupBox groupBox3;
905         private System.Windows.Forms.Label label6;
906         private System.Windows.Forms.Label label1;
907         private System.Windows.Forms.Button BTNZoomP;
908         private System.Windows.Forms.Button BTNZoomM;
909         private System.Windows.Forms.GroupBox groupBox4;
910         private System.Windows.Forms.GroupBox groupBox5;
911         private System.Windows.Forms.Button button1;
```

```
912         private System.Windows.Forms.NumericUpDown numericOnes;
913         private System.Windows.Forms.Label Ants;
914         private System.Windows.Forms.GroupBox groupBox6;
915         private System.Windows.Forms.Button BTNClear;
916         private System.Windows.Forms.GroupBox groupBox7;
917         private System.Windows.Forms.NumericUpDown numericCols;
918         private System.Windows.Forms.NumericUpDown numericRows;
919         private System.Windows.Forms.Label label11;
920         private System.Windows.Forms.Label label8;
921         private System.Windows.Forms.Button BTNCreateMatrix;
922         private System.Windows.Forms.Label label12;
923         private System.Windows.Forms.Label label13;
924         private System.Windows.Forms.Label label14;
925         private System.Windows.Forms.GroupBox groupBox9;
926         private System.Windows.Forms.CheckBox CheckGraphEnabled;
927         private System.Windows.Forms.GroupBox groupBox10;
928         public System.Windows.Forms.FlowLayoutPanel flowLayoutPanel1;
929         private System.Windows.Forms.GroupBox groupBox2;
930         private System.Windows.Forms.CheckBox CBToroid;
931         private System.Windows.Forms.Label label3;
932         private System.Windows.Forms.GroupBox groupBox11;
933         private System.Windows.Forms.Label label5;
934         private System.Windows.Forms.ColorDialog colorDialog;
935         private System.Windows.Forms.Button BTNColorGrid;
936         private System.Windows.Forms.Button BTNColorWorker;
937         private System.Windows.Forms.Button BTNColorDead;
938         private System.Windows.Forms.Label label2;
939         private System.Windows.Forms.GroupBox groupBox12;
940         private System.Windows.Forms.Label label9;
941         private System.Windows.Forms.Label label7;
942         private System.Windows.Forms.NumericUpDown NumericMale;
943         private System.Windows.Forms.NumericUpDown NumericQueen;
944         private System.Windows.Forms.NumericUpDown NumericTTL;
945         private System.Windows.Forms.Label label15;
946         private System.Windows.Forms.Label label10;
947         private System.Windows.Forms.Label Orientation;
948         private System.Windows.Forms.ComboBox ComboOrientation;
949         private System.Windows.Forms.ComboBox ComboType;
950         private System.Windows.Forms.Label label17;
951         private System.Windows.Forms.Label label16;
952         private System.Windows.Forms.Label label18;
953         private System.Windows.Forms.Button BTNColorQueen;
954         private System.Windows.Forms.Button BTNColorMale;
955         private System.Windows.Forms.Label label4;
956         private System.Windows.Forms.ComboBox ComboTypeGraph;
957     }
958 }
```

A continuación se muestra el código utilizado para nuestro objeto hormiga.

```
1 using System;
2 using System.Collections.Generic;
3 using System.Drawing;
4 using System.Linq;
```

```
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace Langton_s_ant
9  {
10     public class ANT
11     {
12         //ANT STATES
13
14
15         //Time to live
16         private int TTL = 50;
17
18         //CELLS STATES
19         private const uint DEAD = 0;
20
21         //Ant data
22         private int x, y;
23         private int orientation;
24         private int type;
25         private Color color = Color.White;
26         /// <summary>
27         /// Constructor
28         /// </summary>
29         /// <param name="x_i">X position</param>
30         /// <param name="y_i">Y position</param>
31         /// <param name="orientation_i">Orientation</param>
32         public ANT(int x_i, int y_i, int orientation_i, int type_i, int
               TTL_i)
33         {
34             x = x_i;
35             y = y_i;
36             orientation = orientation_i;
37             type = type_i;
38             TTL = (TTL_i == 0)?50:TTL_i;
39         }
40
41         public int getType() {
42             return type;
43         }
44
45
46         public int getTTL() {
47             return TTL;
48         }
49
50         public void setTTL() {
51             Random rand = new Random();
52             TTL = rand.Next(45, 60);
53         }
54         /// <summary>
55         /// Rotates the ant to left
56         /// </summary>
```

```
57          public void rotateLeft()
58          {
59              orientation = (++orientation > Helper.EAST) ? Helper.NORTH :
                    orientation;
60          }
61
62          /// <summary>
63          /// Rotates the ant to right
64          /// </summary>
65          public void rotateRight()
66          {
67              orientation = (--orientation < Helper.NORTH) ? Helper.EAST :
                    orientation;
68          }
69
70          public int getX()
71          {
72              return x;
73          }
74
75          public int getY()
76          {
77              return y;
78          }
79
80          public int getOrientation()
81          {
82              return orientation;
83          }
84
85          public void setPosition(int n_position)
86          {
87              orientation = n_position;
88          }
89          public void setCoords(int n_x, int n_y)
90          {
91              x = n_x;
92              y = n_y;
93          }
94
95          public void setColor(Color n_color)
96          {
97              color = n_color;
98          }
99
100         public Color getColor()
101         {
102             return color;
103         }
104
105         /// <summary>
106         /// Calculates the next generation state of the ant
107         /// </summary>
```

```
108         /// <param name="p_matrix">The space</param>
109         /// <param name="is_toroid">Enable a closed space</param>
110         /// <returns></returns>
111         public uint[,] nextGeneration(uint[,] p_matrix, bool is_toroid)
112         {
113             //Getting the max X and Y size of the space
114             int max_x = p_matrix.GetLength(0);
115             int max_y = p_matrix.GetLength(1);
116             uint current_cell = p_matrix[x, y];
117             TTL--;
118             //We need to know the value of the actual cell
119             switch (current_cell)
120             {
121                 //The cell is dead
122                 case DEAD:
123                     //Set an integer value (uint equivalent of a colour
                            object)
124                     p_matrix[x, y] = new ColorHandler().fromColorToInt(
                            getColor());
125                     //The ant rotates to the right
126                     rotateRight();
127                     break;
128                 //Other case
129                 default:
130                     //Change the state of our current cell to dead
131                     p_matrix[x, y] = DEAD;
132                     //The ant rotates to the left
133                     rotateLeft();
134                     break;
135 
136             }
137             move(max_x - 1, max_y - 1, is_toroid);
138             return p_matrix;
139         }
140 
141         /// <summary>
142         /// Changes the X and Y position of our ant.
143         /// </summary>
144         /// <param name="max_x">Max X size of the space</param>
145         /// <param name="max_y">Max Y size of the space</param>
146         /// <param name="is_toroid">Enable a closed space</param>
147         public void move(int max_x, int max_y, bool is_toroid)
148         {
149             switch (orientation)
150             {
151 
152                 case Helper.NORTH:
153                     //We suppose that the space is a toroid
154                     y = ((y) > 0) ? y-1 : max_y;
155                     //The space is a toroid?
156                     y = (is_toroid) ? y : 0;
157                     break;
158 
```

```
159                 case Helper.WEST:
160                     //We suppose that the space is a toroid
161                     x = ((x) > 0) ? x-1 : max_x;
162                     //The space is a toroid?
163                     x = (is_toroid) ? x : 0;
164                     break;
165 
166                 case Helper.SOUTH:
167                     //We suppouse that space is a toroid
168                     y = ((y) < max_y) ? y+1 : 0;
169                     //The space is a toroid?
170                     y = (is_toroid) ? y : max_y;
171                     break;
172 
173                 case Helper.EAST:
174                     //We suppouse that space is a toroid
175                     x = ((x) < max_x) ? x+1 : 0;
176                     //The space is a toroid?
177                     x = (is_toroid) ? x : max_x;
178                     break;
179             }
180         }
181     }
182 }
```

También tenemos otra clase más llamada ColorHandler, la cual nos permite generar colores únicos, los cuales usamos para el "rastro" que dejan las hormigas.

```
1  using System;
2  using System.Collections.Generic;
3  using System.Drawing;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7 
8  namespace Langton_s_ant
9  {
10     public class ColorHandler
11     {
12         private int g_color_index = 1000;
13         private readonly int g_offset_increment;
14 
15         public ColorHandler() {
16             g_offset_increment = 20000;
17         }
18 
19         public ColorHandler(uint offset_increment) {
20             g_offset_increment = (int)offset_increment;
21         }
22         public uint fromColorToInt(Color color)
23         {
24             return (uint)((color.A << 24) | (color.R << 16) | (color.G
                   << 8) | (color.B << 0));
25         }
```

```
26
27          public Color fromIntToColor(uint argb)
28          {
29              byte[] bytes = BitConverter.GetBytes(argb);
30              return Color.FromArgb(bytes[2], bytes[1], bytes[0]);
31          }
32
33          public Color randomColor() {
34              return fromIntToColor( (uint)(g_color_index +=
                    g_offset_increment));
35          }
36      }
37  }
```

Esta clase nos permite realizar algunas acciones que nos ayudan a manipular interfaz además de contener algunas constantes que utilizaremos en todo el funcionamiento del programa.

```
1   using System;
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Text;
5   using System.Threading.Tasks;
6
7   namespace Langton_s_ant
8   {
9       public static class Helper
10      {
11          //Ants types
12          public const int QUEEN = 1;
13          public const int MALE = 2;
14          public const int WORKER = 3;
15          //Ants orientation
16          public const int NORTH = 0;
17          public const int WEST = 1;
18          public const int SOUTH = 2;
19          public const int EAST = 3;
20
21          /// <summary>
22          /// Generate a random type of ant
23          /// The probability of get a Queen it's 1%
24          /// The probability of get a Male it's 2%
25          /// The probability of get a Worker it's 97%
26          /// </summary>
27          /// <returns></returns>
28          public static int generateType(int queen_prob, int male_prob) {
29              Random random = new Random();
30              int rand_val = random.Next(0,100);
31              if ((queen_prob + male_prob) >= 100) {
32                  queen_prob = 1;
33                  male_prob = 2;
34              }
35              male_prob += queen_prob;
```

```
36              if (rand_val >= 0 && rand_val <= queen_prob)
37                  return QUEEN;
38              else if (rand_val > queen_prob && rand_val <= (male_prob))
39                  return MALE;
40              else
41                  return WORKER;
42          }
43
44          /// <summary>
45          /// Generate a random orientation of an ant
46          /// </summary>
47          /// <returns></returns>
48          public static int generateOrientation() {
49              Random random = new Random();
50              int rand_val = random.Next(0, 100);
51
52              if (rand_val >= 0 && rand_val <= 25)
53                  return NORTH;
54              else if (rand_val > 25 && rand_val <= 50)
55                  return WEST;
56              else if (rand_val > 50 && rand_val <= 75)
57                  return SOUTH;
58              else
59                  return EAST;
60          }
61
62          public static int getOrientationValue(string str) {
63              if (str == "North")
64                  return NORTH;
65              else if (str == "West")
66                  return WEST;
67              else if (str == "South")
68                  return SOUTH;
69              else
70                  return EAST;
71          }
72
73          public static int getTypeValue(string str) {
74              if (str == "Worker")
75                  return WORKER;
76              else if (str == "Male")
77                  return MALE;
78              else
79                  return QUEEN;
80          }
81
82          /// <summary>
83          /// Evaluates the orientation of an ant and returns
84          /// a string with the orientation
85          /// </summary>
86          /// <param name="orientation">Orientation code</param>
87          /// <returns>Text of the orientation</returns>
88          public static string textOrientation(int orientation) {
```

```
89              if (orientation == NORTH)
90                  return "Selected: NORTH";
91              else if (orientation == WEST)
92                  return "Selected: WEST";
93              else if (orientation == SOUTH)
94                  return "Selected: SOUTH";
95              else
96                  return "Selected: EAST";
97          }
98          /// <summary>
99          /// Evaluates the type of an ant and returns
100         /// a string with the type
101         /// </summary>
102         /// <param name="type">Type code</param>
103         /// <returns>Text of the type</returns>
104         public static string textType(int type) {
105             if (type == QUEEN)
106                 return "Type: QUEEN";
107             else if (type == MALE)
108                 return "Type: MALE";
109             else
110                 return "Type: WORKER";
111         }
112     }
113 }
```

Finalmente tenemos el código que hace que funcione nuestro autómata celular.

```
1  using System;
2  using System.Drawing;
3  using System.Windows.Forms;
4  using System.IO;
5  using System.Collections;
6  using System.Collections.Generic;
7
8  namespace Langton_s_ant
9  {
10
11     /// <summary>
12     /// To make this automata we need to know the behaviour of the
           Lanton's ant:
13     /// 1. If the ant is in a white cell, the ant changes the color of
           the cell, rotates 90 and
14     /// move it one cell
15     /// 2. If the ant is int a black cell, the ant changes the color of
           the cell, rotates 90 and
16     /// move it one cell
17     /// </summary>
18     public partial class Form1 : Form
19     {
20
21         /*************************************
22          *          GLOBAL VARIABLES         *
23          *************************************/
```

```
24
25          //Space
26          private uint[,] matrix;
27
28          //Area of each cell
29          private int cellArea = 10;
30          //Total of alive accumulated cells
31          private long acumOnes = 0;
32          //Generation
33          private int generation = 1;
34          //Alive cells
35          private int total_cells = 0;
36
37          //Ant and cell colors
38          private SolidBrush alive = new SolidBrush(Color.White);
39          private SolidBrush dead = new SolidBrush(Color.Black);
40          private SolidBrush ant_worker = new SolidBrush(Color.Red);
41          private SolidBrush ant_male = new SolidBrush(Color.Orange);
42          private SolidBrush ant_queen = new SolidBrush(Color.Cyan);
43          //Grid color
44          private Pen grid = Pens.Gray;
45
46          //Initial position ant
47          private int put_ant = Helper.NORTH;
48          private string message_orientation = "Selected ";
49          private string message_type = "Type: ";
50          private uint message_color;
51          private string message_TTL = "TTL: ";
52          //Our ants
53          List<ANT> ants = new List<ANT>();
54          //CELLS STATES
55          private const int ALIVE = 1;
56          private const int DEAD = 0;
57
58          //Color of our cells and the grid
59          ColorHandler colors = new ColorHandler();
60          /// <summary>
61          /// Constructor
62          /// </summary>
63          public Form1()
64          {
65              InitializeComponent();
66              createMatrix(100, 100);
67              scrollBox();
68          }
69
70          /// <summary>
71          /// This method paints the matrix in the Paint Box
72          /// </summary>
73          /// <param name="sender"></param>
74          /// <param name="e"></param>
75          private void PBAutomataSimulator_Paint(object sender,
                PaintEventArgs e)
```

```
76          {
77              List<ANT> aux_ant = new List<ANT>();
78              int x_size = matrix.GetLength(0);
79              int y_size = matrix.GetLength(1);
80              total_cells = x_size * y_size;
81
82              Graphics graphics = e.Graphics;
83
84              for (int x = 0; x < x_size; x++)
85              {
86
87                  for (int y = 0; y < y_size; y++)
88                  {
89
90                      uint current_cell = matrix[x, y];
91                      if (current_cell != DEAD)
92                      {
93                          SolidBrush aliveCellColor = new SolidBrush(
                                colors.fromIntToColor(current_cell));
94                          graphics.FillRectangle(aliveCellColor, x *
                                cellArea, y * cellArea, cellArea, cellArea);
95                      }
96                      else
97                      {
98                          graphics.FillRectangle(dead, x * cellArea, y *
                                cellArea, cellArea, cellArea);
99                      }
100                 }
101             }
102
103             for (int i = 0; i < ants.Count; i++)
104             {
105                 ANT ant_obj = ants[i];
106                 if (ant_obj.getType() == Helper.WORKER)
107                     graphics.FillEllipse(ant_worker, ant_obj.getX() *
                            cellArea, ant_obj.getY() * cellArea, cellArea,
                            cellArea);
108                 else if (ant_obj.getType() == Helper.MALE)
109                     graphics.FillEllipse(ant_male, ant_obj.getX() *
                            cellArea, ant_obj.getY() * cellArea, cellArea,
                            cellArea);
110                 else
111                 {
112                     graphics.FillEllipse(ant_queen, ant_obj.getX() *
                            cellArea, ant_obj.getY() * cellArea, cellArea,
                            cellArea);
113
114                 }
115                 graphics.DrawEllipse(new Pen(Color.Black), ant_obj.getX
                        () * cellArea, ant_obj.getY() * cellArea, cellArea,
                        cellArea);
116                 try
117                 {
```

```
118                     if (ant_obj.getTTL() > 0)
119                         aux_ant.Add(ant_obj);
120                 }
121                 catch (Exception) { }
122             }
123             ants = new List<ANT>(aux_ant);
124             for (int y = 0; y < y_size; y++)
125                 graphics.DrawLine(grid, 0, y * cellArea, total_cells *
                        cellArea, y * cellArea);
126 
127             for (int x = 0; x < x_size; x++)
128                 graphics.DrawLine(grid, x * cellArea, 0, x * cellArea,
                        total_cells * cellArea);
129 
130         }
131         /// <summary>
132         /// This function creates a matrix of ints which size it's n x m
133         /// </summary>
134         /// <param name="rows"></param>
135         /// <param name="cols"></param>
136         private void createMatrix(int rows, int cols)
137         {
138             matrix = new uint[rows, cols];
139             scrollBox();
140         }
141 
142         /// <summary>
143         /// This method calls nextGeneration method and
144         /// updates the GUI and the count of our alive cells
145         /// </summary>
146         private void step()
147         {
148             List<ANT> aux = new List<ANT>(ants);
149             for (int i = 0; i < ants.Count; i++)
150             {
151                 ANT ant_obj = ants[i];
152                 if (ant_obj.getType() == Helper.QUEEN)
153                 {
154                     aux = QueenActions(i);
155                 }
156                 matrix = ant_obj.nextGeneration(matrix, CBToroid.Checked
                        );
157                 PBAutomataSimulator.Invalidate();
158             }
159             ants = new List<ANT>(aux);
160             updateTextGeneration();
161             countOnes();
162             if (ants.Count == 0)
163                 PBAutomataSimulator.Invalidate();
164         }
165 
166         private List<ANT> QueenActions(int queen_index)
167         {
```

```
168             List<ANT> new_ants = new List<ANT>();
169             ANT first_queen = ants[queen_index];
170             int queen_fx = first_queen.getX();
171             int queen_fy = first_queen.getY();
172             bool flag = true;
173             for (int i = 0; i < ants.Count; i++)
174             {
175                 flag = true;
176                 ANT current_ant = ants[i];
177                 if ((i != queen_index) && (queen_fx == current_ant.getX
                        () && queen_fy == current_ant.getY()))
178                 {
179                     switch (current_ant.getType())
180                     {
181                         case Helper.QUEEN:
182                             if (new Random().Next(0, 100) <= 50)
183                                 new_ants.Add(current_ant);
184                             else
185                                 new_ants.Add(ants[queen_index]);
186                             flag = false;
187                             break;
188 
189                         case Helper.MALE:
190                             new_ants.Add(new ANT(queen_fx, queen_fy,
                                    Helper.generateOrientation(),
191                                 Helper.generateType((int)NumericQueen.
                                        Value,
192                                 (int)NumericMale.Value),
193                                 (int)NumericTTL.Value));
194                             flag = false;
195                             break;
196                     }
197 
198                 }
199                 if (flag)
200                     new_ants.Add(current_ant);
201             }
202             return new_ants;
203         }
204 
205         /// <summary>
206         /// Here we just change the text that show us
207         /// the number of generations
208         /// </summary>
209         private void updateTextGeneration()
210         {
211             TXTGeneration.Text = "Generation: " + generation++;
212         }
213 
214         /// <summary>
215         /// This method make a rezise of the Paint Box and flow layout
                panel
216         /// it makes possible make zoom and the movement into the GUI
```

```
217         /// </summary>
218         private void scrollBox()
219         {
220             PBAutomataSimulator.Size = new Size((matrix.GetLength(0)) *
                    cellArea, (matrix.GetLength(1)) * cellArea);
221             PBAutomataSimulator.SizeMode = PictureBoxSizeMode.AutoSize;
222             flowLayoutPanel1.AutoScroll = true;
223             flowLayoutPanel1.Controls.Add(PBAutomataSimulator);
224         }
225 
226         /// <summary>
227         /// As you can imagine here we just get the number of ones
228         /// in our matrix (alive cells)
229         /// </summary>
230         private void countOnes()
231         {
232             int ones = 0;
233             int work = 0, male = 0, queen = 0;
234             for (int x = 0; x < matrix.GetLength(0); x++)
235             {
236 
237                 for (int y = 0; y < matrix.GetLength(1); y++)
238                 {
239                     if (matrix[x, y] != DEAD)
240                         ones++;
241                 }
242             }
243 
244             foreach (ANT ant_obj in ants) {
245                 if (ant_obj.getType() == Helper.WORKER)
246                     work++;
247                 else if (ant_obj.getType() == Helper.MALE)
248                     male++;
249                 else
250                     queen++;
251             }
252             if (CheckGraphEnabled.Checked)
253             {
254                 CHHistogram.Series["#Worker"].Points.AddY(work);
255                 CHHistogram.Series["#Male"].Points.AddY(male);
256                 CHHistogram.Series["#Queen"].Points.AddY(queen);
257             }
258 
259             TXTPopulation.Text = "Population " + ones;
260             acumOnes += ones;
261             double val = acumOnes / generation;
262             label12.Text = "Total Population: " + acumOnes;
263             label13.Text = "Average: " + (val);
264             label14.Text = "Density: " + (val / (matrix.GetLength(0) *
                    matrix.GetLength(1)));
265             label3.Text = "Ants: " + ants.Count;
266         }
267 
```

```
268         /*****************************************************
269          *                        Events                     *
270          *****************************************************/
271
272         private void BTNStep_Click(object sender, EventArgs e)
273         {
274             PBAutomataSimulator.Invalidate();
275             step();
276         }
277
278         private Color getColor()
279         {
280             DialogResult result = colorDialog.ShowDialog();
281             return colorDialog.Color;
282         }
283         private void BTNStart_Click(object sender, EventArgs e)
284         {
285             if (BTNStart.Text == "Start")
286             {
287                 TimerSimulation.Start();
288                 BTNStart.Text = "Stop";
289             }
290             else
291             {
292                 TimerSimulation.Stop();
293                 BTNStart.Text = "Start";
294             }
295         }
296
297         private void trackBar1_ValueChanged(object sender, EventArgs e)
298         {
299             TimerSimulation.Interval = TBSpeed.Value;
300         }
301
302         private void TimerSimulation_Tick(object sender, EventArgs e)
303         {
304             step();
305         }
306
307         private void BTNZoomP_Click(object sender, EventArgs e)
308         {
309             if (cellArea < 50)
310             {
311                 cellArea++;
312                 PBAutomataSimulator.Invalidate();
313                 scrollBox();
314             }
315         }
316
317         private void BTNZoomM_Click(object sender, EventArgs e)
318         {
319             if (cellArea > 1)
320             {
```

```
321                 cellArea--;
322                 PBAutomataSimulator.Invalidate();
323                 scrollBox();
324             }
325         }
326 
327         private void button1_Click(object sender, EventArgs e)
328         {
329             int pos = 0;
330             Random ra = new Random();
331             for (int x = 0; x < matrix.GetLength(0); x++)
332             {
333 
334                 for (int y = 0; y < matrix.GetLength(1); y++)
335                 {
336                     //Probability of put an ant
337                     int integer_rand = ra.Next(0, 99);
338                     double rand = (ra.NextDouble()) + integer_rand;
339                     if (rand < float.Parse(numericOnes.Text))
340                     {
341                         ANT ant_obj = new ANT(x, y, Helper.
                                generateOrientation(),
342                             Helper.generateType((int)NumericQueen.Value,
343                             (int)NumericMale.Value),
344                             (int)NumericTTL.Value);
345                         ant_obj.setColor(colors.randomColor());
346                         ants.Add(ant_obj);
347                     }
348                     else matrix[x, y] = DEAD;
349                 }
350             }
351             PBAutomataSimulator.Invalidate();
352         }
353 
354         private void BTNClear_Click(object sender, EventArgs e)
355         {
356             CHHistogram.Series["#Worker"].Points.Clear();
357             CHHistogram.Series["#Male"].Points.Clear();
358             CHHistogram.Series["#Queen"].Points.Clear();
359             for (int x = 0; x < matrix.GetLength(0); x++)
360             {
361 
362                 for (int y = 0; y < matrix.GetLength(1); y++)
363                 {
364 
365                     matrix[x, y] = DEAD;
366                 }
367             }
368             ants.Clear();
369             PBAutomataSimulator.Invalidate();
370             acumOnes = generation = 0;
371 
372         }
```

```
373
374         private void BTNCreateMatrix_Click(object sender, EventArgs e)
375         {
376             int rows = (numericRows.Value == 0) ? 100 : (int)numericRows
                .Value;
377             int cols = (numericCols.Value == 0) ? 100 : (int)numericCols
                .Value;
378             createMatrix(rows, cols);
379         }
380
381
382         private void PBAutomataSimulator_MouseMove(object sender,
            MouseEventArgs e)
383         {
384             int x = e.X / cellArea;
385             int y = e.Y / cellArea;
386
387
388             foreach (ANT ant_obj in ants)
389             {
390
391                 if (ant_obj.getX() == x && ant_obj.getY() == y)
392                 {
393                     message_orientation = Helper.textOrientation(ant_obj
                        .getOrientation());
394                     message_type = Helper.textType(ant_obj.getType());
395                     message_TTL = "TTL: " + ant_obj.getTTL() + "";
396                     message_color = colors.fromColorToInt(ant_obj.
                        getColor());
397                 }
398             }
399             label5.Text = message_orientation;
400             label2.Text = message_type;
401             label17.Text = message_TTL;
402             label18.BackColor = colors.fromIntToColor(message_color);
403         }
404
405         private void BTNColorDead_Click(object sender, EventArgs e)
406         {
407             dead = new SolidBrush(getColor());
408             PBAutomataSimulator.Invalidate();
409         }
410
411         private void BTNColorAnt_Click(object sender, EventArgs e)
412         {
413             ant_worker = new SolidBrush(getColor());
414             PBAutomataSimulator.Invalidate();
415         }
416
417         private void BTNColorMale_Click(object sender, EventArgs e)
418         {
419             ant_male = new SolidBrush(getColor());
420             PBAutomataSimulator.Invalidate();
```

```
421         }
422
423         private void BTNColorQueen_Click(object sender, EventArgs e)
424         {
425             ant_queen = new SolidBrush(getColor());
426             PBAutomataSimulator.Invalidate();
427         }
428
429         private void BTNColorGrid_Click(object sender, EventArgs e)
430         {
431             grid = new Pen(getColor());
432             PBAutomataSimulator.Invalidate();
433         }
434
435
436         private void PBAutomataSimulator_Click(object sender, EventArgs
                e)
437         {
438             MouseEventArgs me = (MouseEventArgs)e;
439             int x = (me.X / cellArea);
440             int y = (me.Y / cellArea);
441             if (me.Button == System.Windows.Forms.MouseButtons.Right)
442             {
443                 foreach (ANT ant_obj in ants)
444                 {
445                     if (ant_obj.getX() == x && ant_obj.getY() == y)
446                     {
447                         ant_obj.setColor(getColor());
448                         break;
449                     }
450                 }
451             }
452             else
453             {
454                 bool can_put_ant = true;
455                 int i = 0, j = 0;
456                 Random rnd = new Random();
457                 for (i = 0; i < ants.Count; i++)
458                 {
459                     ANT ant_obj = ants[i];
460                     if (ant_obj.getX() == x && ant_obj.getY() == y)
461                     {
462                         can_put_ant = false;
463                         j = i;
464                         break;
465                     }
466                 }
467
468                 //Add a new ant
469                 if (can_put_ant)
470                     ants.Add(new ANT(x, y, Helper.getOrientationValue(
                            ComboOrientation.Text),
471                         Helper.getTypeValue(ComboType.Text), (int)
```

```
                                    NumericTTL.Value));
472                 else
473                 {
474                     ants.RemoveAt(j);
475                 }
476             }
477             PBAutomataSimulator.Invalidate();
478         }
479
480         private void ComboTypeGraph_SelectedIndexChanged(object sender,
                EventArgs e)
481         {
482             if (ComboTypeGraph.Text == "Column")
483             {
484                 CHHistogram.Series[0].ChartType = System.Windows.Forms.
                        DataVisualization.Charting.SeriesChartType.Column;
485                 CHHistogram.Series[1].ChartType = System.Windows.Forms.
                        DataVisualization.Charting.SeriesChartType.Column;
486                 CHHistogram.Series[2].ChartType = System.Windows.Forms.
                        DataVisualization.Charting.SeriesChartType.Column;
487             }
488             else if (ComboTypeGraph.Text == "Bar")
489             {
490                 CHHistogram.Series[0].ChartType = System.Windows.Forms.
                        DataVisualization.Charting.SeriesChartType.Line;
491                 CHHistogram.Series[1].ChartType = System.Windows.Forms.
                        DataVisualization.Charting.SeriesChartType.Line;
492                 CHHistogram.Series[2].ChartType = System.Windows.Forms.
                        DataVisualization.Charting.SeriesChartType.Line;
493             }
494             else
495             {
496                 CHHistogram.Series[0].ChartType = System.Windows.Forms.
                        DataVisualization.Charting.SeriesChartType.Point;
497                 CHHistogram.Series[1].ChartType = System.Windows.Forms.
                        DataVisualization.Charting.SeriesChartType.Point;
498                 CHHistogram.Series[2].ChartType = System.Windows.Forms.
                        DataVisualization.Charting.SeriesChartType.Point;
499             }
500         }
501     }
502 }
```