

```
1 using System;
2 using System.Collections.Generic;
3 using System.Drawing;
4 using System.Windows.Forms;
5
6 namespace Assign_1
7 {
8     /** Matthew Alunni
9      * 5865647
10     * COSC 3P71
11     * Assignment 1 */
12
13     public partial class Form1 : Form
14     {
15
16         List<Solution> solutions = new List<Solution>(); //list of solutions
17         int current = -1; //which solution is being displayed
18
19         public Form1()
20         {
21             InitializeComponent();
22         }
23
24         /** this method finds a solution and prints it to the board on click**/
25         private void btnDrawBoard_Click(object sender, EventArgs e)
26         {
27             FindSolutions();
28
29             if (solutions.Count > 0)
30             {
31                 current = 0;
32                 PrintSolution();
33             }
34         }
35
36         /** this method finds solutions, if the user inputs a valid number**/
37         private void FindSolutions()
38         {
39             solutions = new List<Solution>();
40
41             current = 0;
42             var pos = new Position(0, Int32.MinValue, null);
43
44             if (numericNumberOfQueens.Value >= 2 && numericNumberOfQueens.Value < 4)
45             {
46                 System.Windows.Forms.MessageBox.Show("Please enter a valid
47                 number.");
48             }
49             else
50             {
51                 pos.FindSolution(solutions, Convert.ToInt32
```

```
(numericNumberOfQueens.Value), 0);  
51     }  
52 }  
53  
54  
55 /** this method is used for the bottom scroller to print solutions**/  
56 public void PrintSolution()  
57 {  
58     if (current == 0)  
59     {  
60         buttonFirst.Enabled = false;  
61         buttonPrevious.Enabled = false;  
62     }  
63     else  
64     {  
65         buttonFirst.Enabled = true;  
66         buttonPrevious.Enabled = true;  
67     }  
68  
69     if (current == solutions.Count - 1)  
70     {  
71         buttonNext.Enabled = false;  
72         buttonLast.Enabled = false;  
73     }  
74     else  
75     {  
76         buttonNext.Enabled = true;  
77         buttonLast.Enabled = true;  
78     }  
79  
80  
81     labelCost.Text = string.Format("Heuristic cost {0}", solutions  
[current].Cost);  
82     labelResults.Text = string.Format("Solution {0} of {1}", current + 1,   
solutions.Count);  
83  
84     Position Node = solutions[current].Position;  
85  
86     panel1.Visible = false;  
87  
88     PictureBox[,] theBoard = createBoard(Convert.ToInt32  
(numericNumberOfQueens.Value));  
89  
90     while (Node.Row >= 0)  
91     {  
92         theBoard[Node.Row, Node.Line - 1].Image =  
Assign_1.Properties.Resources.crown;  
93         theBoard[Node.Row, Node.Line - 1].SizeMode =  
PictureBoxSizeMode.StretchImage;  
94         Node = Node.Parent;  
95     }  
96     panel1.Visible = true;
```

```
97     }
98
99     /** this method sets up the board**/
100    public PictureBox[,] createBoard(int size)
101    {
102        panel1.Controls.Clear();
103
104        PictureBox[,] board = new PictureBox[size, size];
105        int w = 0, h = 0;
106        w = panel1.Width;
107        h = panel1.Height;
108        int horizontal = (int)((double)w / (double)size);
109        int vertical = (int)((double)h / (double)size);
110        for (int len = 0; len < size; len++)
111            for (int wid = 0; wid < size; wid++)
112            {
113                board[len, wid] = new PictureBox();
114                board[len, wid].Parent = panel1;
115                board[len, wid].Location = new Point(wid * horizontal + 1, len * vertical + 1);
116                board[len, wid].Size = new Size(horizontal, vertical);
117                if ((len + wid) % 2 == 0)
118                    board[len, wid].BackColor = Color.White;
119                else
120                    board[len, wid].BackColor = Color.Black;
121            }
122
123        return board;
124    }
125
126    private void buttonPrevious_Click(object sender, EventArgs e)
127    {
128        current--;
129        PrintSolution();
130    }
131
132    private void buttonNext_Click(object sender, EventArgs e)
133    {
134        current++;
135        PrintSolution();
136    }
137
138
139    private void buttonFirst_Click(object sender, EventArgs e)
140    {
141        current = 0;
142        PrintSolution();
143    }
144
145    private void buttonLast_Click(object sender, EventArgs e)
146    {
147        current = solutions.Count - 1;
```

```
148         PrintSolution();
149     }
150
151     private void buttonHeuristic_Click(object sender, EventArgs e)
152     {
153         FindSolutions();
154         //solutions = solutions.Sort(
155         solutions.Sort((x, y) => x.Cost.CompareTo(y.Cost));
156
157         if (solutions.Count > 0)
158         {
159             current = 0;
160             PrintSolution();
161         }
162
163     }
164
165     private void label1_Click(object sender, EventArgs e)
166     {
167
168     }
169
170     private void numericNumberOfQueens_ValueChanged(object sender, EventArgs e)
171     {
172
173     }
174
175 }
176
177
178 }
179
```