

**Centro Universitario de Ciencias
Exactas e Ingenierías**

Universidad de Guadalajara

REPORTE 06

Productor - Consumidor

Alumnos:

Carbajal Armenta Yessenia Paola
Sánchez Lozano Jonathan

Códigos:

220286482
215768126

Profesora:

Becerra Velázquez Violeta del Rocío

Materia:

Seminario de Soluciones de
Problemas de Sistemas Operativos

Departamento:

Ciencias Computacionales

Carrera:

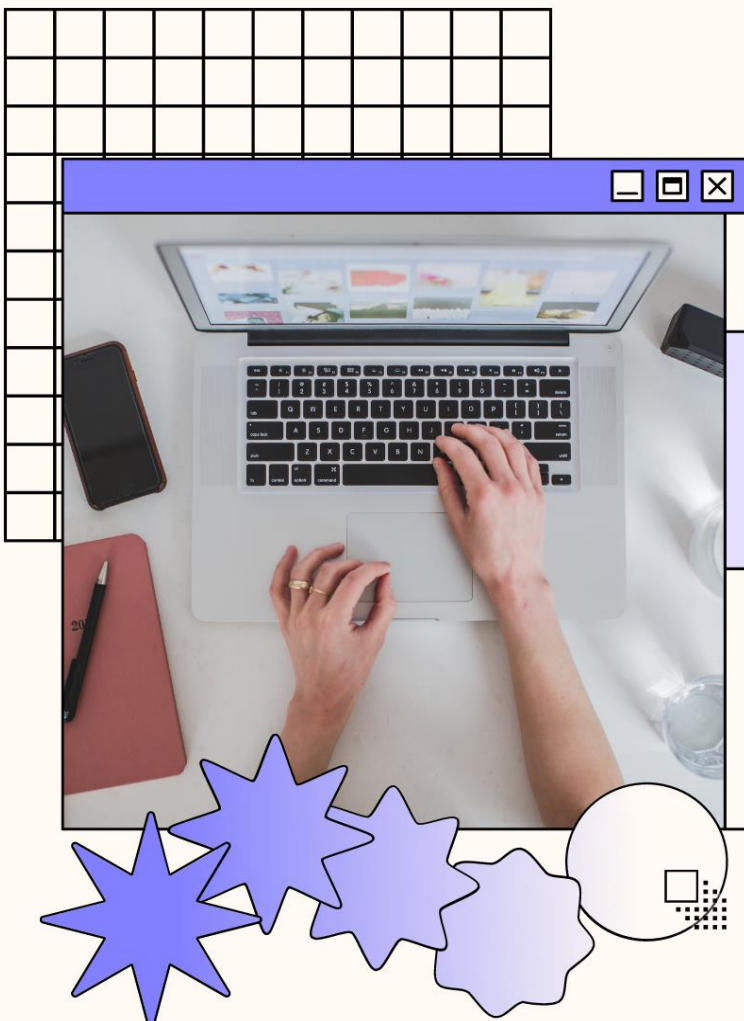
Ingeniería en Computación

NRC:

103844

Sección:

D01



Actividad No. 12

Productor - Consumidor

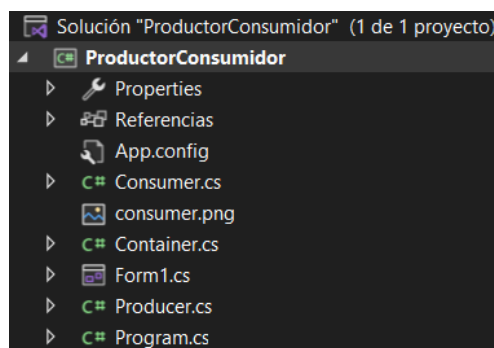
El programa describe dos procesos, productor y consumidor, ambos comparten un buffer de tamaño finito. La tarea del productor es generar un producto, almacenarlo y comenzar nuevamente; mientras que el consumidor toma productos uno a uno. El problema consiste en que el productor no añada más productos que la capacidad del buffer y que el consumidor no intente tomar un producto si el buffer está vacío.

Objetivo

En esta actividad, simularemos el proceso del sistema productor-consumidor. Este sistema ayuda a tener un control muy rígido sobre los objetos que son ingresados a la memoria y los que son trabajados para salir de ella. En esta ocasión, intentaremos simular esa dinámica con un programa sencillo.

Desarrollo

Para este programa se decidió realizar un nuevo entorno de desarrollo en Visual Studio usando el lenguaje de programación de C#. En el proceso se crearon los siguientes archivos:



Al igual que en el programa pasado, seguimos empleando C++ y el framework de Qt ya que, pese a las complicaciones presentadas en las actividades previas, consideramos que hemos aprendido a utilizarlos de mejor manera, además, llegados a este punto, pese a que cada programa puede ser abordado desde el principio con algún otro lenguaje y/o framework sin problemas.

Para esta práctica en específico, se le añadieron las funciones necesarias para hacer posible el uso de las teclas requeridas para esta práctica.

Para validar las nuevas pulsaciones de las teclas, se añadieron las opciones de 'N' como nuevo proceso y 'B' como tabla de procesos al sistema de reconocimiento de tecleo:

Program.cs

En este archivo contamos con lo que sería la inicialización de nuestra aplicación, donde se ejecuta haciendo llamada al form.

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Threading.Tasks;
5  using System.Windows.Forms;
6
7  namespace ProductorConsumidor
8  {
9      static class Program
10     {
11         /// <summary>
12         /// Punto de entrada principal para la aplicación.
13         /// </summary>
14         [STAThread]
15         static void Main()
16         {
17             Application.EnableVisualStyles();
18             Application.SetCompatibleTextRenderingDefault(false);
19             Application.Run(new Form1());
20         }
21     }
22 }
```

Consumer.cs

En esta sección se encuentran los setters y getters del consumidor.

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace ProductorConsumidor
8  {
9      6 referencias
10     class Consumer
11     {
12         private int state;
13         private int currentPos;
14
15         2 referencias
16         public Consumer()
17         {
18             state = 0;
19             currentPos = 0;
20
21         5 referencias
22         public void setState(int value)
23         {
24             state = value;
25
26         5 referencias
27         public int getState()
28         {
29             return state;
30
31         2 referencias
32         public void setCurrentPos(int value)
33         {
34             currentPos = value;
35
36         public int getCurrentPos()
37         {
38             return currentPos;
39
40         1 referencia
41         public bool Consume()
42         {
43             return false;
44         }
45     }
```

Producer.cs

En esta sección se encuentran los setters y getters del productor.

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace ProductorConsumidor
8  {
9      class Producer
10     {
11         private int state;
12         private int currentPos;
13
14         public Producer()
15         {
16             state = 0;
17             currentPos = 0;
18         }
19
20         public void setState(int value)
21         {
22             state = value;
23         }
24
25         public int getState()
26         {
27             return state;
28         }
29
30         public void setCurrentPos(int value)
31         {
32             currentPos = value;
33         }
34
35         public int getCurrentPos()
36         {
37             return currentPos;
38         }
39
40         public bool Produce()
41         {
42             return true;
43         }
44     }
45 }
```

Container.cs

En esta sección contamos con la declaración de las variables necesarias para el programa, tales como el tamaño del contenedor, el tiempo dormido, el tiempo trabajando, el intentando, el turno del consumidor y el turno del productor, la inicialización del buffer, algunos setters y getters, la función que decide aleatoriamente la cantidad a producir y consumir así como el turno aleatorio del productor o del consumidor y la impresión de las etiquetas de quien está en proceso y en que estado del mismo se encuentra.

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using System.Windows.Forms;
7
8  namespace ProductorConsumidor
9  {
10     30 referencias
11     static class Constants
12     {
13         public const int CONTAINER_SIZE = 25;
14         public const int SLEEPING = 0;
15         public const int WORKING = 1;
16         public const int TRYING = 2;
17         public const int PRODUCER_TURN = 1;
18         public const int CONSUMER_TURN = 2;
19     }
20     3 referencias
21     class Container
22     {
23         private Producer producer;
24         private Consumer consumer;
25         private int currentTurn;
26         private Random turnRandom;
27         private Random amount;
28         private bool[] buffer;
29
30         1 referencia
31         public Container()
32         {
33             buffer = new bool[Constants.CONTAINER_SIZE];
34             producer = new Producer();
35             consumer = new Consumer();
36             currentTurn = 0;
37             turnRandom = new Random();
38             amount = new Random();
39             for (int i = 0; i < buffer.Length; ++i)
40             {
41                 buffer[i] = false;
42             }
43         }
44     }
45 }
```

```

42 public bool setAction(int pos, bool action)
43 {
44     if (action == buffer[pos])
45     {
46         if (action)
47         {
48             producer.setState(Constants.TRYING);
49         }
50         else
51         {
52             consumer.setState(Constants.TRYING);
53         }
54         return false;
55     }
56     buffer[pos] = action;
57     return true;
58 }
59
60 2 referencias
61 public Consumer GetConsumer()
62 {
63     return consumer;
64 }
65
66 2 referencias
67 public Producer GetProducer()
68 {
69     return producer;
70 }
71
72 1 referencia
73 public int GetCurrentTurn()
74 {
75     return currentTurn;
76 }
77
78 0 referencias
79 public bool At(int value)
80 {
81     return buffer[value];
82 }

```

```

88 public int ProductsCount()
89 {
90     int productsCount = 0;
91
92     for (int i = 0; i < Constants.CONTAINER_SIZE; ++i)
93     {
94         if (buffer[i])
95         {
96             productsCount++;
97         }
98     }
99     return productsCount;
100 }
101
102 1 referencia
103 public int NextMoves()
104 {
105     int p, a;
106     p = turnRandom.Next() % 100;
107     a = amount.Next(2, 5);
108     Console.WriteLine("Cantidad siguiente: " + a.ToString());
109
110     if (p % 2 == 0)
111     {
112         currentTurn = Constants.PRODUCER_TURN;
113     }
114     else
115     {
116         currentTurn = Constants.CONSUMER_TURN;
117     }
118
119     if (currentTurn == Constants.PRODUCER_TURN)
120     {
121         if (ProductsCount() != Constants.CONTAINER_SIZE)
122         {
123             producer.setState(Constants.WORKING);
124             consumer.setState(Constants.SLEEPING);
125         }
126         else
127         {
128             producer.setState(Constants.TRYING);
129             consumer.setState(Constants.SLEEPING);
130         }
131     }
132 }

```

```

122     }
123     else //CONSUMER TURN
124     {
125         if (ProductsCount() > 0)
126         {
127             consumer.setState(Constants.WORKING);
128             producer.setState(Constants.SLEEPING);
129         }
130         else
131         {
132             consumer.setState(Constants.TRYING);
133             producer.setState(Constants.SLEEPING);
134         }
135     }
136     return a;
137 }
138 }
139 }
140 }
141 }

```

Form1.cs

En esta sección se cuenta con todas las funciones del form, aquí tenemos las etiquetas de los procesos, donde se imprime tanto la imagen del producto (en este caso una dona), así como el color de las etiquetas, las cuales dependen si son del productor, consumidor y en qué estado se encuentre.

```

11 namespace ProductorConsumidor
12 {
13     4 referencias
14     public partial class Form1 : Form
15     {
16         private Container container;
17         1 referencia
18         public Form1()
19         {
20             container = new Container();
21             InitializeComponent();
22         }
23
24         3 referencias
25         public void ChangeStateLabel(int c, int p)
26         {
27             switch (p)
28             {
29                 case Constants.WORKING:
30                     producerState.Text = "Trabajando";
31                     producerState.BackColor = Color.ForestGreen;
32                     producerState.Refresh();
33                     label1.Text = "Productor";
34                     label1.BackColor = Color.ForestGreen;
35                     label1.Refresh();
36                     break;
37                 case Constants.TRYING:
38                     producerState.Text = "Intentando";
39                     producerState.BackColor = Color.Gold;
40                     producerState.Refresh();
41                     label1.Text = "Productor";
42                     label1.BackColor = Color.Gold;
43                     label1.Refresh();
44                     break;
45                 default:
46                     producerState.Text = "Durmiendo";
47                     producerState.BackColor = Color.Red;
48                     producerState.Refresh();
49                     label1.Text = "Productor";
50                     label1.BackColor = Color.Red;
51                     label1.Refresh();
52                     break;
53             }
54         }
55     }
56 }

```



```

51         switch (c)
52         {
53             case Constants.WORKING:
54                 consumerState.Text = "Trabajando";
55                 consumerState.BackColor = Color.ForestGreen;
56                 consumerState.Refresh();
57                 label27.Text = "Consumidor";
58                 label27.BackColor = Color.ForestGreen;
59                 label27.Refresh();
60                 break;
61             case Constants.TRYING:
62                 consumerState.Text = "Intentando";
63                 consumerState.BackColor = Color.Gold;
64                 producerState.Refresh();
65                 label27.Text = "Consumidor";
66                 label27.BackColor = Color.Gold;
67                 label27.Refresh();
68                 break;
69             default:
70                 consumerState.Text = "Durmiendo";
71                 consumerState.BackColor = Color.Red;
72                 consumerState.Refresh();
73                 label27.Text = "Consumidor";
74                 label27.BackColor = Color.Red;
75                 label27.Refresh();
76                 break;
77         }
78     }
79 }

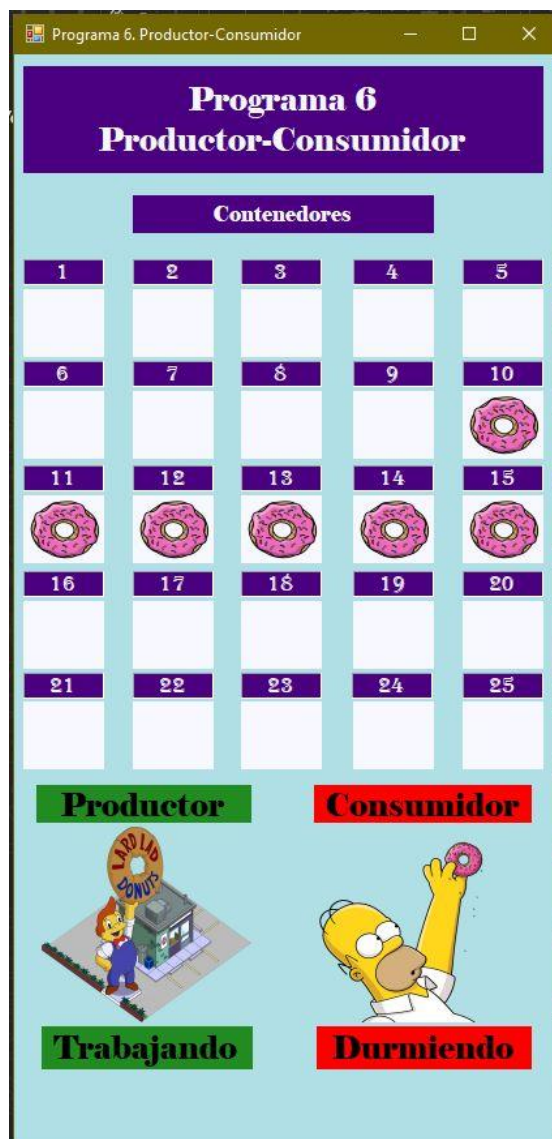
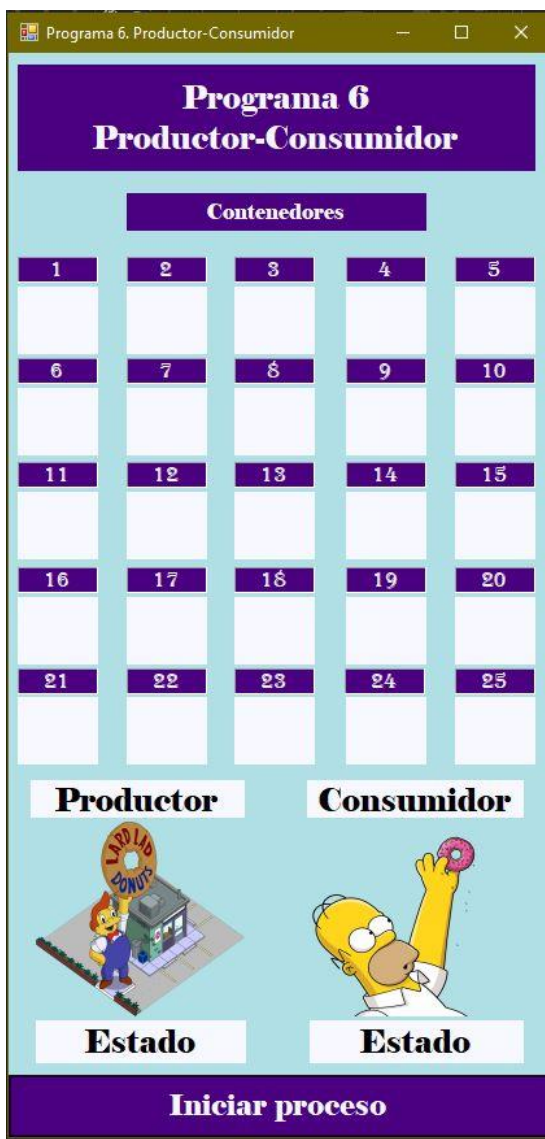
```

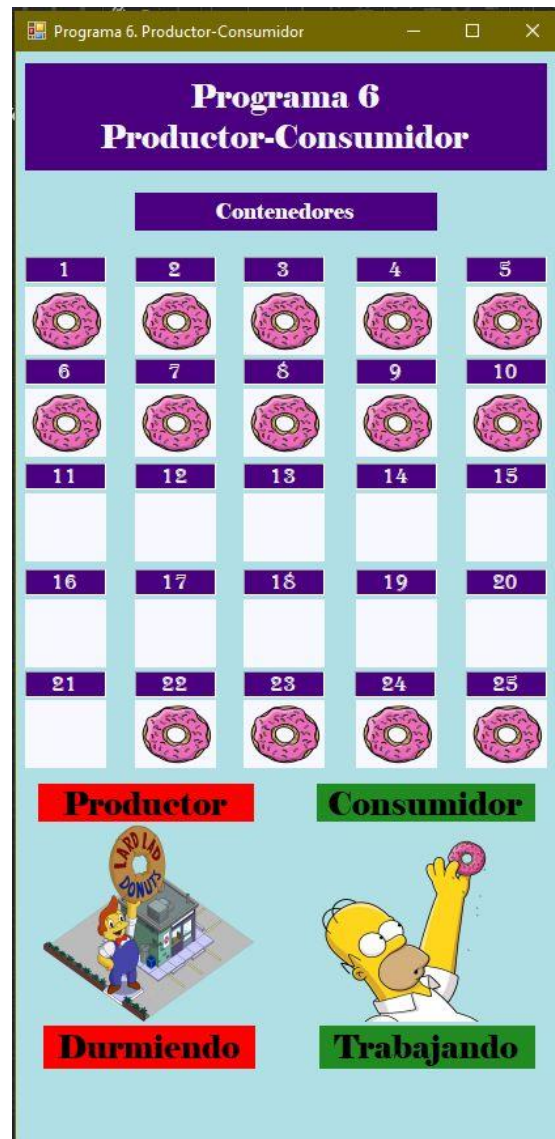
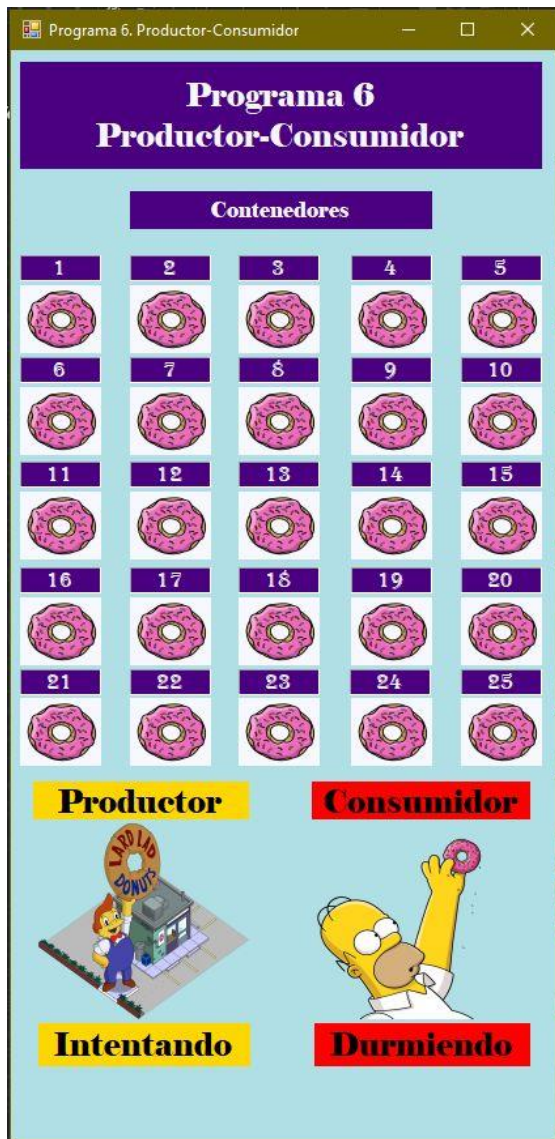
```

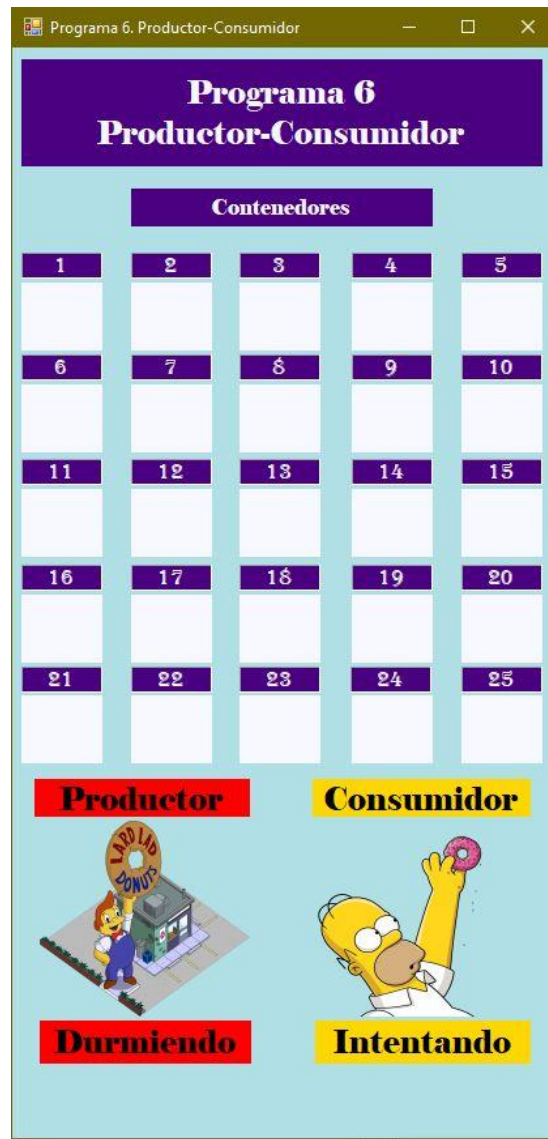
80 public void SetImage(int index, int turn)
81 {
82     Bitmap bmp = new Bitmap("../Rosquilla.png");
83     PictureBox pb = new PictureBox();
84
85     switch (index)
86     {
87         case 0: pb = pictureBox1; break;
88         case 1: pb = pictureBox2; break;
89         case 2: pb = pictureBox3; break;
90         case 3: pb = pictureBox4; break;
91         case 4: pb = pictureBox5; break;
92         case 5: pb = pictureBox6; break;
93         case 6: pb = pictureBox7; break;
94         case 7: pb = pictureBox8; break;
95         case 8: pb = pictureBox9; break;
96         case 9: pb = pictureBox10; break;
97         case 10: pb = pictureBox11; break;
98         case 11: pb = pictureBox12; break;
99         case 12: pb = pictureBox13; break;
100        case 13: pb = pictureBox14; break;
101        case 14: pb = pictureBox15; break;
102        case 15: pb = pictureBox16; break;
103        case 16: pb = pictureBox17; break;
104        case 17: pb = pictureBox18; break;
105        case 18: pb = pictureBox19; break;
106        case 19: pb = pictureBox20; break;
107        case 20: pb = pictureBox21; break;
108        case 21: pb = pictureBox22; break;
109        case 22: pb = pictureBox23; break;
110        case 23: pb = pictureBox24; break;
111        case 24: pb = pictureBox25; break;
112    }
113
114    if (turn == Constants.PRODUCER_TURN)
115    {
116        pb.Image = bmp;
117    }
118    else
119    {
120        if (pb.Image != null)
121        {
122            pb.Image = null;
123        }
124    }
125 }
126 }

```

Programa en ejecución:







Conclusiones


Carbajal Armenta Yessenia Paola:

Esta actividad me pareció interesante ya que fue algo distinto a lo que habíamos estado desarrollando a lo largo del curso, gracias a esto comprendí mejor lo que es este problema y como darle una solución.

Sánchez Lozano Jonathan:

Para esta actividad no fue un gran reto el que tuvimos, pero si fue algo interesante ya que de lo que veníamos haciendo en prácticas pasadas esto fue totalmente diferente, en lo personal me gustó mucho porque fue divertido, aunque sí nos tomó algunos días poder terminarla.

Enlace del código

 <https://drive.google.com/drive/folders/1AOTOol7eGLPfsxuT1yE10u7yGTymSB8Q?usp=sharing>

Enlace del vídeo

 [https://drive.google.com/file/d/1HQRgF_ilkhGtk-hFhv_RR - AU33Eh8WO/view?usp=sharing](https://drive.google.com/file/d/1HQRgF_ilkhGtk-hFhv_RR-AU33Eh8WO/view?usp=sharing)