Proyecto final: implementación máquina de estado finito

VIRTUAL LOVE

Colaboradores:

CORDERO PRECIADO EDGAR ADRIAN

LANDEROS MORALES JAIRO MAURICIO

LÓPEZ VALENZUELA AKARY BEATRIZ

TEORÍA DE LA COMPUTACIÓN

Asesor:

MTRO. ABELARDO GÓMEZ ANDRADE

GUADALAJARA, JALISCO, MÉXICO. 05 DE DICIEMBRE DE 2020

ÍNDICE

INTRODUCCIÓN

Planteamiento del problema

Si tuviéramos que nombrar todas las necesidades básicas para los seres humanos, aparte de comer, beber y dormir, no dudaríamos en mencionar la necesidad de relacionarse con otras personas. El ser humano es social por naturaleza. Las relaciones interpersonales son imprescindibles en la vida de las personas y socializar es una actividad tan arraigada que ni siquiera nos damos cuenta cuando lo hacemos. Fueron estas ideas las que nos inspiraron a incursionar en el área de las relaciones interpersonales y a crear un proyecto como Virtual Love.

En Virtual Love se pretende explorar el proceso y los retos que conlleva relacionarse con una persona al mismo tiempo que proporcionar una interactividad amigable y divertida para los usuarios. Aunque mencionamos que las relaciones son básicas en nuestras vidas cotidianas es cierto que para una gran cantidad de la población socializar presenta un verdadero reto debido a temas de personalidad. Es aquí donde podemos darnos cuenta de que en una relación podemos apreciar características de las personalidades de los involucrados en la interacción. Es por eso por lo que también se pensó en las ventajas que otorgaría un simulador como este para asistir en estudios enfocados en temas de personalidad y socialización.

En este equipo concordamos que la experiencia del usuario es lo más importante, especialmente cuando se trata de temas de estudio y en general para las audiencias que solo buscan pasar un buen rato. Las interfaces de texto suelen ser bastante limitadas, son propensas a errores frecuentes y pueden llegar incluso a asustar a los usuarios. Especialmente en programas que buscan facilitar procesos de estudio o entretener el aspecto visual es de suma importancia por lo que no se puede prescindir de una interfaz gráfica amigable y llamativa.

Objetivo

Con Virtual Love simularemos la relación entre dos personas (una el usuario y otra el programa) donde se tomarán en cuenta aspectos como estados emocionales y respuestas físicas inmediatas. Para ello utilizaremos la potencia de una máquina de estados, la cual se ajusta perfectamente al modelo de simulación como lo hemos planteado.

Virtual Love busca recrear el proceso de interacción entre dos individuos, de la manera más fiel posible, para apoyar de manera importante en equipos de estudio que se dediquen a la investigación o apoyo a personas con problemas para socializar. De igual manera se busca generar una experiencia divertida en personas que no tengan ningún fin educativo y que prefieran pasar un buen rato observando las reacciones del programa.

Debido a la magnitud del reto que representa simular una relación se optó por utilizar un modelo que no solo nos permitiera diseñar la simulación si no que nos proporcionara herramientas de implementación. Para lograr esto se hace uso de una máquina de estado con la que se puede diseñar rápidamente la simulación y con la que podemos implementar un software bastante optimizado.

El aspecto visual es de suma importancia y algo que no podemos minimizar. Para el apartado gráfico se busca otorgar una experiencia amigable y sobre todo cómoda para los usuarios que pretenden pasar mucho tiempo en ella. Concordamos que la mejor forma de lograr esto es mediante una interfaz minimalista e intuitiva por lo que decidimos diseñar una interfaz de usuario gráfica haciendo uso de swing en java. Esto conlleva realizar la implementación total de Virtual Love utilizando dicho lenguaje de programación.

De igual manera se hace uso de imágenes creadas por nosotros mismos las cuales ayudarán a dotar al programa de personalidad y harán visible los resultados de la simulación por lo que solo será necesario observar las imágenes para darnos cuenta de los que está sucediendo.

Justificación

Las simulaciones dentro de las ciencias computacionales han llamado la atención de los profesionales por mucho tiempo. El hecho de recrear un proceso o comportamiento no solo causa fascinación si no que también es de mucha ayuda tanto para la industria como para los campos de la educación y hasta la industria del entretenimiento.

Las máquinas de estado son modelos de una potencia altísima que facilitan el proceso de diseño de una simulación y que además son fáciles de programar, por lo que se requiere de códigos no muy largos ni complejos para simular un proceso con detalle considerable. Virtual Love hace uso de la potencia de una máquina de estados para hacer una simulación sencilla, cualidad que podremos apreciar en los apartados siguientes.

Las capacidades de una máquina de estados aunadas a la posibilidad de comunicarse con el usuario de una manera amigable y divertida hablan de lo incluyente que pude ser esta simulación para cualquier tipo de público: los quieren consumir un producto de entretenimiento y los que desean un asistente técnico o e estudio. Las múltiples herramientas de diseño con las que contamos permiten que se puedan crear experiencias que resulten familiares al usuario y que podamos comunicarnos en un lenguaje más universal.

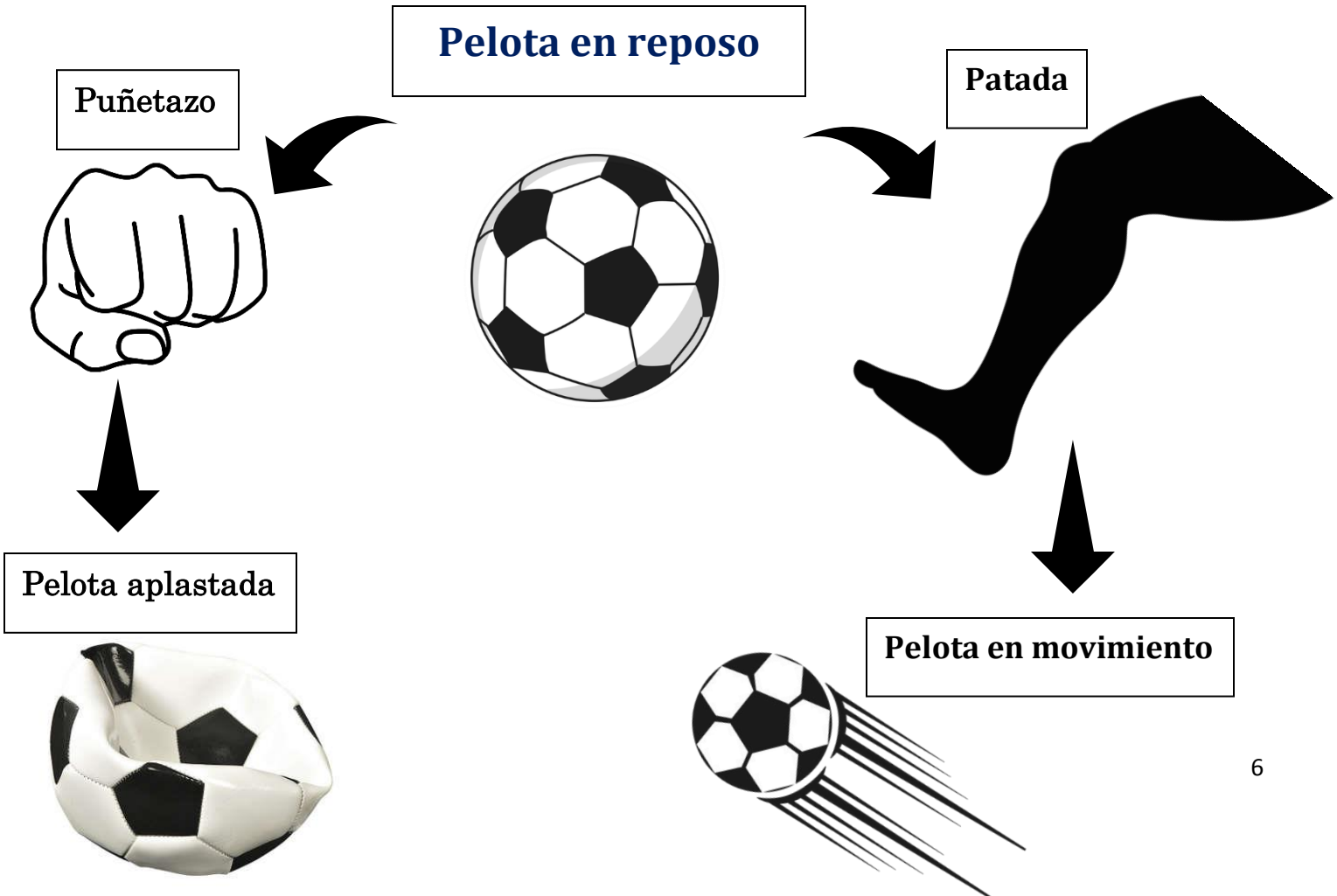
MARCO TEÓRICO

Máquinas de estado

Una máquina de estados no es una máquina física. Es una modelización conceptual, generalmente en forma de diagrama, de un problema.

En este problema el sujeto de interés se encuentra en una situación inicial a la espera de recibir un estímulo del mundo exterior para reaccionar a él. Por ejemplo, si tenemos una pelota de playa posada sobre el suelo, que será su posición inicial, podemos darle una patada, con lo que se moverá en una dirección, o podemos darle un puñetazo hacia abajo, con lo que se aplastará. Si queremos plasmar en un papel este problema, de forma esquematizada y concisa, podemos recurrir a múltiples formas, pero la gran mayoría tendrán las siguientes partes: un objeto de estudio, en este caso la pelota; un estado inicial, parada en el suelo; unas entradas que serán la patada y el puñetazo y unas acciones que serán avanzar y aplastarse.

Podríamos dibujarlo así:



Una máquina de estados, lo que hace es sentar unas bases comunes para la modelización de este tipo de problemas, dado que estos pueden crecer en complejidad y llegar a hacerse muy difíciles de representar.

Lo primero que se hizo fue definir algunos términos.

Estados: se definen así las posiciones o acciones por las que pasa el objeto de estudio. Dentro de los estados destaca el denominado estado inicial o de espera, que representa la posición de partida del objeto en la que se encuentra esperando un evento o entrada para pasar al siguiente estado. En general, el resto de los estados puede partir/llegar de forma directa a esta posición o forman un anillo con principio y fin en él. Cuando nos movemos entre estados solemos decir que pasamos al siguiente estado o que venimos del estado anterior.

Entradas: Son las interacciones del medio con el objeto, que pueden alterarlo y hacer que cambie de estado. En nuestro caso serían la patada y el puñetazo.

Salidas: Son las acciones que realiza el objeto hacia el exterior. En el caso de la pelota de playa no habría ninguna a no ser que tirara algo al salir despedida por la patada o que explotara por la presión expulsando el aire.

Eventos: Son las acciones, ya sean internas o externas que hacen que el objeto cambie de estado. Serían la patada, el puñetazo como externos y la expansión del aire que lo vuelve a llenar y el rozamiento que lo para los internos.

Transiciones: Son los “caminos” por los que el objeto cambia de un estado a otro. Acotan los cambios y comprueban las condiciones.

Por lo tanto, en una máquina de estados, tenemos un objeto que puede estar en x posiciones y que, debido a la acción de los eventos, ya sean internos o externos, se desplaza entre ellos siguiendo las transiciones permitidas mientras puede o no emitir señales al exterior

La importancia de estos aparentemente sencillos esquemas ha traspasado su entorno inicial y ahora se utilizan versiones adaptadas en un gran número de áreas, y no solo en el ámbito de la tecnología.

Los pasos para generar una de estas máquinas son los mismos. Primero se hace una conceptualización del problema definiendo los estados, las entradas, etc. A continuación, se rellena una tabla de estados/entradas donde se dan las pautas para realizar el esquema final.

Con el ejemplo de la pelota quedaría así:

s0: Estado inicial (pelota parada)

s1: Estado 2 (pelota aplastada)

s2: Estado 3 (pelota en movimiento)

i0: Entrada 1 (puñetazo)

i1: Entrada 2 (patada)

o0: Salida del estado 2

o1: Salida del estado 3

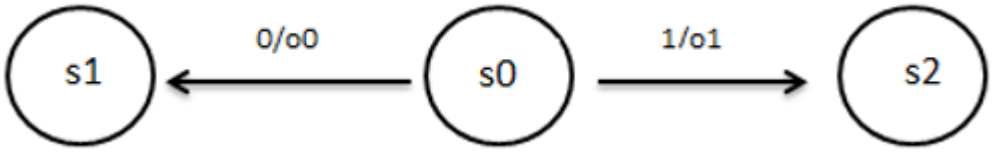
Máquina de Moore:

Estado\entrada	i0 = 0	i1 = 1
s0	s1/x	s2/x

Máquina de Mealv:

Estado\entrada	i0= 0	i1 = 1
s0	s1	s2

o0	o1
x	x



Cuando el sistema es muy complejo, podemos realizar simplificaciones mediante álgebra de Boole a las ecuaciones matemáticas extraídas de las tablas de estado/entrada. Con este procedimiento seremos capaces de construir un gráfico que podrá usarse para implementar muchos proyectos, tanto a nivel de robótica/automática, como a nivel de programación (mediante bucles y condicionales), etc.

Virtual Love como máquina de estado

Una máquina de estados nos permite no solo diseñar la simulación de una relación, sino que también nos proporciona un método de implementación. La forma en la que se estructura el código para trabajar como máquina de estado es mediante tablas. Debido a la naturaleza de las máquinas de estado requerimos trabajar con dos tablas: una que contenga la información del estado siguiente según un estado actual y una entrada y la segunda tabla contiene las salidas según un estado actual y una entrada. Las funciones para acceder a la información en las tablas es la siguiente:

$$f(estado\ actual, símbolo\ de\ salida) = Símbolo\ de\ salida$$
$$g(estado\ actual, símbolo\ de\ salida) = Estado\ siguiente$$

El contenido de la tabla *f* y *g* se determina dese el diseño y se almacenará en una estructura de datos bidimensional (puede ser una matriz) durante la creación del programa. En nuestro código decidimos utilizar una matriz a cuya información accedimos con los operadores de acceso [] que proporciona el lenguaje java. A continuación, se muestran la simbología para las entradas, salidas y estados y las dos tablas características de las máquinas de estado.

INPUTS	OUTPUTS
0. Elogiar	0- Sonríe
1. Insultar	1- Se sonroja
2. Regalar	2- Llanto
3. Mentir	3- Golpea
4. Besar	4- Amenaza
5. Bromear	5- Ríe
6. Escuchar	6- Desaprueba
7. Ignorar	7- Indiferencia
8. Insinuar	8- Agradece
9. Salir con amigos	9- Enfurecer
10. --	10- Sorprende
11. --	11- Celebra
12. --	12- Aprueba
13. --	13- Se enamora

ESTADOS
0. Indiferente
1. Sonrojado
2. Enojado
3. Agradecido
4. Feliz
5. Triste
6. Furioso
7. Cachondo
8. Desconcertado
9. Muy feliz
10. Enamorado
11. Desconsolado

Tabla de estado siguiente

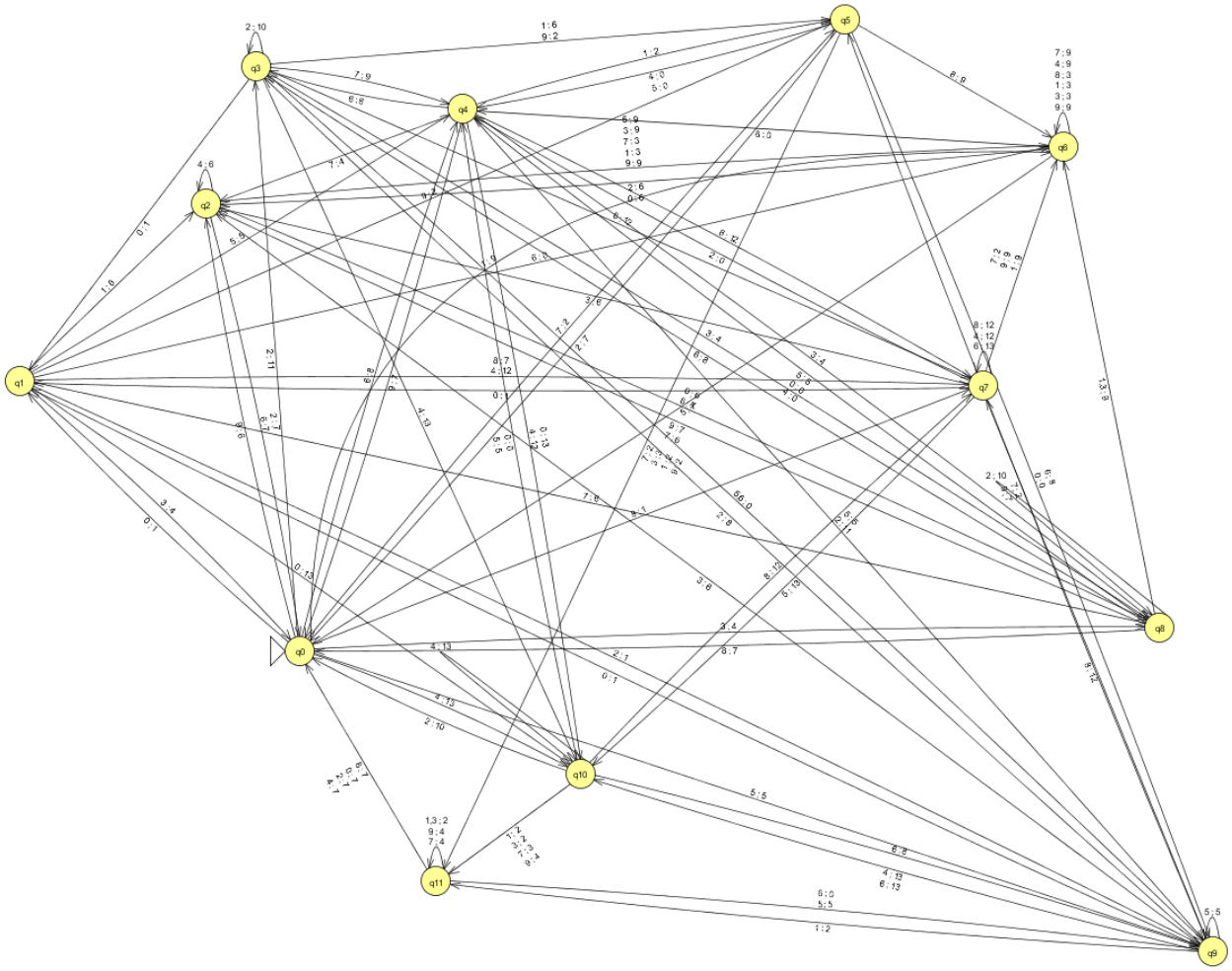
S\I	0	1	2	3	4	5	6	7	8	9
0	1	6	3	8	10	9	4	5	7	2
1	10	2	9	0	7	4	6	8	7	5
2	8	6	0	6	2	6	0	6	8	6
3	1	5	3	8	10	9	9	4	7	5
4	10	5	9	8	10	9	3	2	7	0
5	9	11	0	11	4	4	9	11	6	11
6	2	6	2	6	6	0	4	6	6	6
7	1	6	4	2	7	10	7	6	7	6
8	4	6	8	6	4	4	3	2	0	2
9	1	11	3	2	10	9	10	5	7	5
10	4	11	0	11	10	4	9	11	7	11
11	0	11	0	11	0	9	9	11	0	11

Tabla de salidas

S\I	0	1	2	3	4	5	6	7	8	9
0	1	9	11	4	13	5	8	2	1	6
1	13	6	1	4	12	5	0	6	7	2
2	6	3	7	9	6	9	7	3	7	9
3	1	6	10	4	13	0	0	9	12	2
4	13	2	11	4	12	5	8	4	12	7
5	0	2	7	3	0	0	8	2	9	2
6	6	3	6	3	9	7	0	9	3	9
7	1	9	0	6	12	13	13	2	12	9
8	0	9	10	9	0	5	8	6	7	7
9	1	2	8	6	13	5	13	2	12	7
10	0	2	10	2	13	5	8	3	12	4
11	7	2	7	2	7	5	0	4	7	3

Diagrama de transiciones

El de diagrama de transiciones fue elaborado en JFLAP



METODOLOGÍA

Implementación máquina de estado

El principal aspecto que tendrá el proyecto y será la primera interacción que tendrá el usuario, es que se desplegará una interfaz gráfica con la cual el usuario tendrá la posibilidad de interactuar con el proyecto.

La interfaz gráfica contará con 9 botones con los cuales el usuario podrá interactuar presionándolos con el click izquierdo (cabe aclarar que cada uno de los botones de la interfaz, representará una acción o entrada de la máquina).

Para poder entender el funcionamiento del código, es tan sencillo como el entender el funcionamiento de matriz o un “array bidimensional”. En este código nosotros tendremos dos matrices en las cuales en una de ellas obtendremos una salida, la cual será evaluada por el programa para poder ejecutar una acción y una salida que va a poner el estado siguiente (será explicado posteriormente). Entonces tenemos las dos matrices donde una controlará la acción (imagen que representa un estado de ánimo/sentimiento) y otra matriz que controla las transiciones:

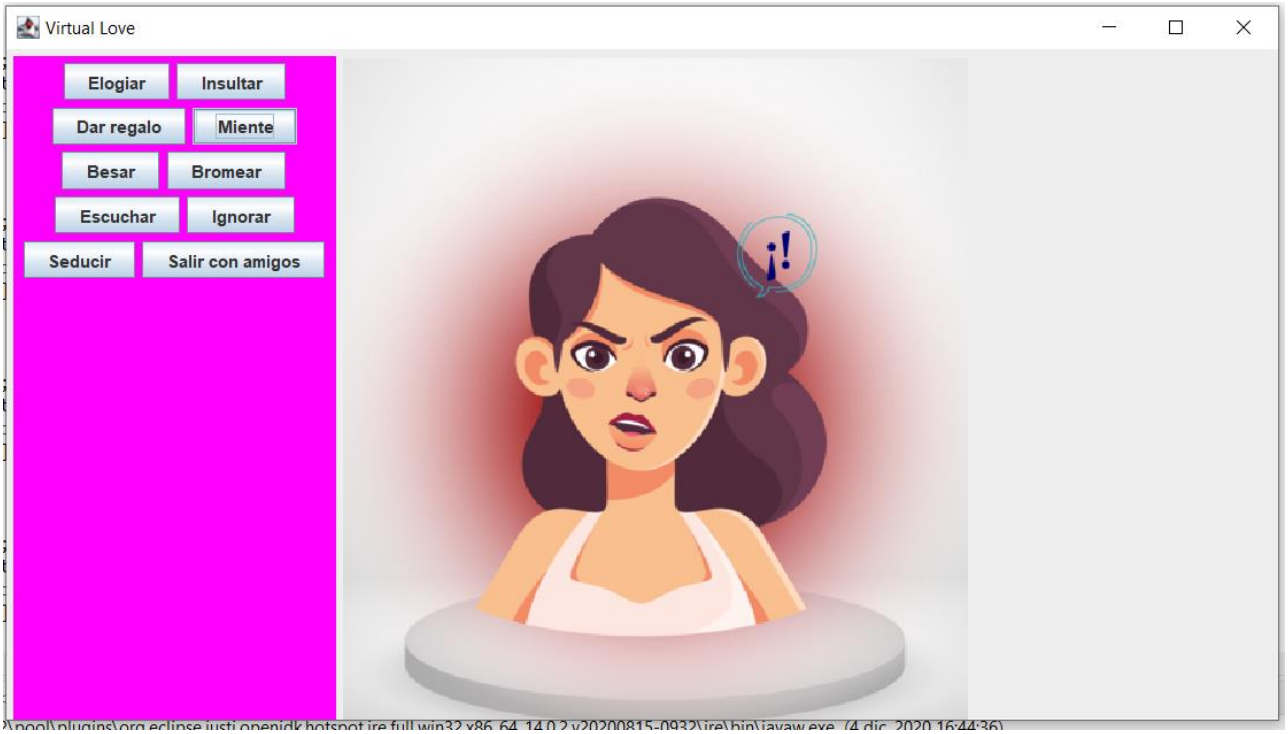
```
int [][] nextState = {
                {1,6,3,8,10,9,4,5,7,2},
                {10,2,9,0,7,4,6,8,7,5},
                {8,6,0,6,2,6,0,6,8,6},
                {1,5,3,8,10,9,9,4,7,5},
                {10,5,9,8,10,9,3,2,7,0},
                {9,11,0,11,4,4,9,11,6,11},
                {2,6,2,6,6,0,4,6,6,6},
                {1,6,4,2,7,10,7,6,7,6},
                {4,6,8,6,4,4,3,2,0,2},
                {1,11,3,2,10,9,10,5,7,5},
                {4,11,0,11,10,4,9,11,7,11},
                {0,11,0,11,0,9,9,11,0,11}};

int [][] outPuts = {
                {1,9,11,4,13,5,8,2,1,6},
                {13,6,1,4,12,5,0,6,7,2},
                {6,3,7,9,6,9,7,3,7,9},
                {1,6,10,4,13,0,0,9,12,2},
                {13,2,11,4,12,5,8,4,12,7},
                {0,2,7,3,0,0,8,2,9,2},
                {6,3,6,3,9,7,0,9,3,9},
                {1,9,0,6,12,13,13,2,12,9},
                {0,9,10,9,0,5,8,6,7,7},
                {1,2,8,6,13,5,13,2,12,7},
                {0,2,10,2,13,5,8,3,12,4},
                {7,2,7,2,7,5,0,4,7,3}};
```

Para poder acceder a cada dato de una matriz, deberemos de escribir la dirección del dato, donde deberemos de ingresar en los corchetes el dato del estado y de la acción o entrada. El primer dato que deberemos de ingresar en el primer corchete es el dato del estado “q” y después la entrada “in”.

Por ejemplo, si nosotros al inicio del programa (el estado “q” se inicia en 0) y presionamos en botón de “mentir”, el programa sabe que al presionar el botón de “mentir” será una entrada de valor 3, por lo que la matriz de salidas arrojará el valor correspondiente que tiene en la matriz ejecutando la acción

```
int [q=0][in=3] outPuts = {
                                {1,9,11,4,13,5,8,2,1,6},
                                {13,6,1,4,12,5,0,6,7,2},
                                {6,3,7,9,6,9,7,3,7,9},
                                {1,6,10,4,13,0,0,9,12,2},
                                {13,2,11,4,12,5,8,4,12,7},
                                {0,2,7,3,0,0,8,2,9,2},
                                {6,3,6,3,9,7,0,9,3,9},
                                {1,9,0,6,12,13,13,2,12,9},
                                {0,9,10,9,0,5,8,6,7,7},
                                {1,2,8,6,13,5,13,2,12,7},
                                {0,2,10,2,13,5,8,3,12,4},
                                {7,2,7,2,7,5,0,4,7,3}
            };
```

Acción mostrada: “AMENAZA”.



Después de haber ejecutado la acción, se actualiza el estado q con la matriz de “nextState” realizando así la transición y seguir con la máquina.

```
int [q=0][in=3] nextState = {
                {1,6,3,8,10,9,4,5,7,2},
                {10,2,9,0,7,4,6,8,7,5},
                {8,6,0,6,2,6,0,6,8,6},
                {1,5,3,8,10,9,9,4,7,5},
                {10,5,9,8,10,9,3,2,7,0},
                {9,11,0,11,4,4,9,11,6,11},
                {2,6,2,6,6,0,4,6,6,6},
                {1,6,4,2,7,10,7,6,7,6},
                {4,6,8,6,4,4,3,2,0,2},
                {1,11,3,2,10,9,10,5,7,5},
                {4,11,0,11,10,4,9,11,7,11},
                {0,11,0,11,0,9,9,11,0,11}};
```

Por lo que para la próxima evaluación q tendrá un valor de 8 y después seguirá con las salidas y las transiciones que corresponderá.

Interfaz gráfica

La principal razón por la cual se eligió java como lenguaje de programación para realizar Virtual Love fue la facilidad con la que se puede diseñar una interfaz gráfica. Para la creación de la GUI se utilizaron dos herramientas de desarrollo de java: swing y awt.

La forma en la que estas herramientas operan es muy simple. Solamente es necesario crear una ventana mediante la clase *JFRame* que pertenece a swing. Una vez creada la ventana continuaremos creando objetos que finalmente añadiremos a la ventana mediante la operación *add*. A continuación, se aprecia el código necesario para crear una ventana.

```
frame = new JFrame();
frame.setSize(width, height);
frame.setResizable(true);
frame.setTitle("Virtual Love");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setLocationRelativeTo(null);
      frame.setLayout(new FlowLayout(FlowLayout.LEFT));
```

Una vez creada la ventana se creó un panel color magenta donde se distribuirían todos los botones con los que se interactuaría con el usuario. Un panel es básicamente un rectángulo independiente de la venta donde, al igual que la ventana, se le agregaran los elementos que se quieran dibujar. Al final si será necesario agregar el panel, con todo y sus elementos previamente añadidos a la ventana principal. A continuación, podemos ver el código necesario para crear el panel mencionado.

```
panel = new JPanel();
panel.setPreferredSize(new Dimension(width/4, height));
panel.setBackground(Color.MAGENTA);
panel.setLayout(new FlowLayout());
```

Los botones interactivos se agregan individualmente (como objetos diferentes) al panel. En el código anterior podemos observar una clase llamada FlowLayout, esta se encargará de dibujar todos los botones que se agreguen en fila y los distribuirá simétricamente dentro del espacio al que se agregaron, en este caso dentro del panel. A continuación, se puede ver la creación de un nuevo botón.

```
JButton buttonFlirt = new JButton("Elogiar");
```

El apartado visual consiste en mostrar diferentes imágenes en pantalla, por lo que es necesario cargar los archivos que contienen en estas imágenes. Para ello se utiliza la clase ImageIcon para leer el archivo y cada imagen se añade a la ventana en forma de una capa más y se pueden hacer visibles o invisibles. El siguiente código muestra como cargar una imagen de los archivos del proyecto.

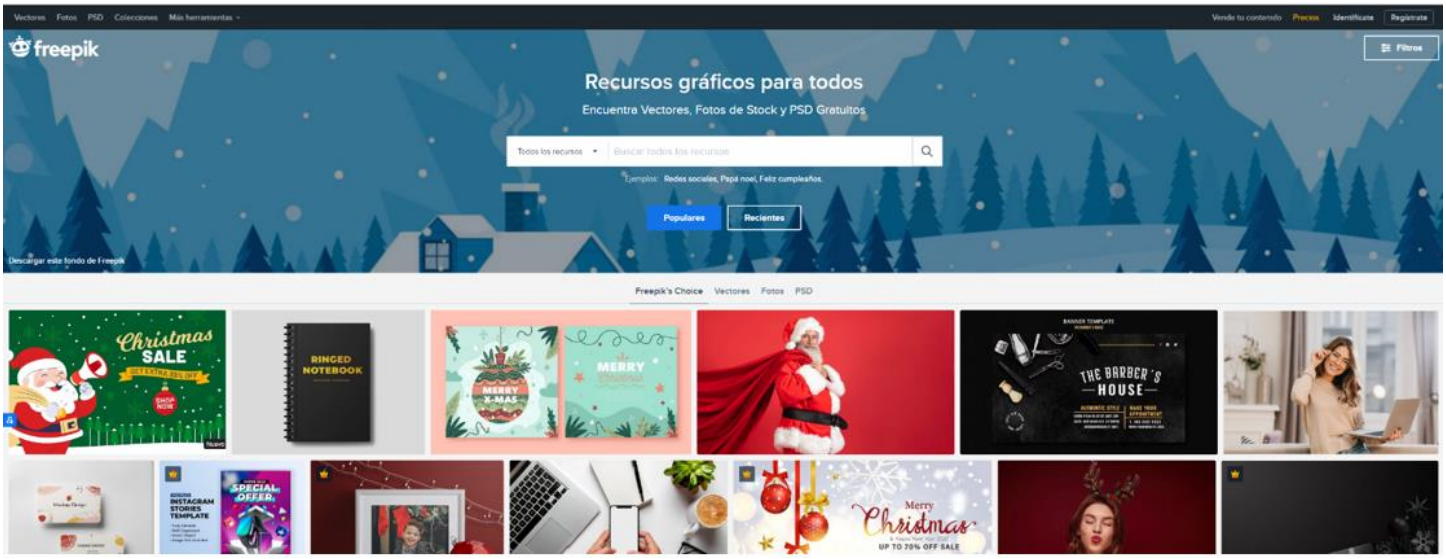
```
public JLabel load(String path) {
	image = new ImageIcon(path);
	label = new JLabel();

	label.setIcon(image);
	label.setVerticalAlignment(JLabel.CENTER);
	label.setHorizontalAlignment(JLabel.CENTER);
	label.setVisible(false);

	return label;
	}
```

Por último, la clase JButton dibuja un botón en la pantalla y genera una animación cada que se hace click sobre él, pero el nivel de interactividad cesa ahí. Para lograr que el presionar un botón realice una acción se requiere de implementar un *Action Listener* o el cual funciona como un sistemas de interrupciones en un procesador. El programa está “atento” a detectar cada vez que uno de los botones es presionado y ejecutar una acción correspondiente. A continuación, podemos la implementación en código.

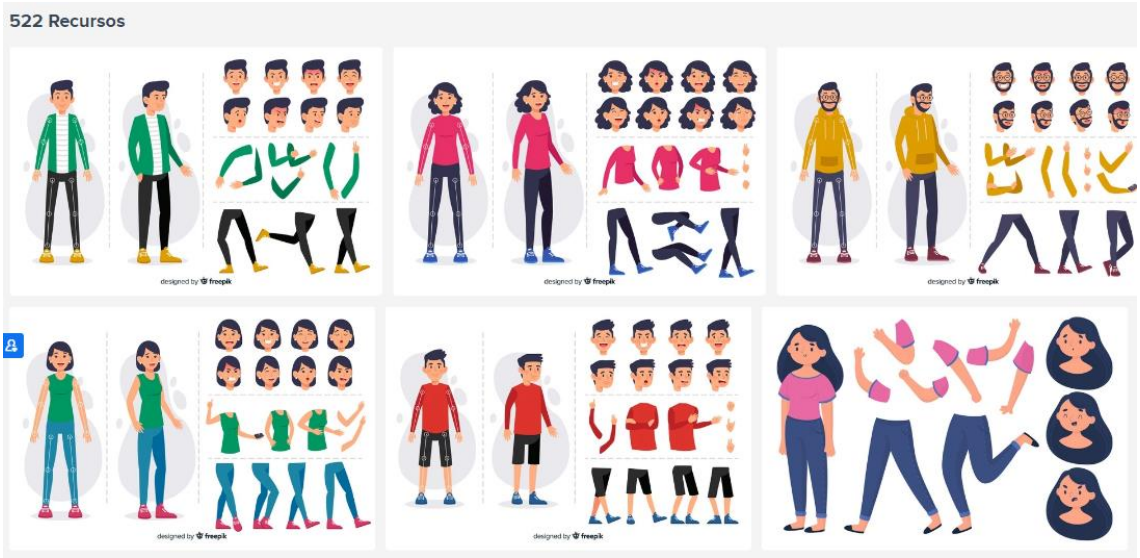
```
buttonFlirt.addActionListener(
	new ActionListener() {
		public void actionPerformed(ActionEvent e) {
			if(e.getSource() == buttonFlirt) {
				input = 0;
			}
		}
	}
		);
```

Proceso de diseño artístico

Primero acude a la página https://www.freepik.es/ donde te encontraras con esta interfaz:



Seleccionaras en la casilla de búsqueda y en ella buscaras “Personajes para acciones” y darás Click en la lupa: Los resultados de la búsqueda mostrarán una cierta cantidad de resultados donde encontraras personajes distintos y podrás elegir alguno:



Una vez te hayas decidido por alguno procederemos a descargarlo.

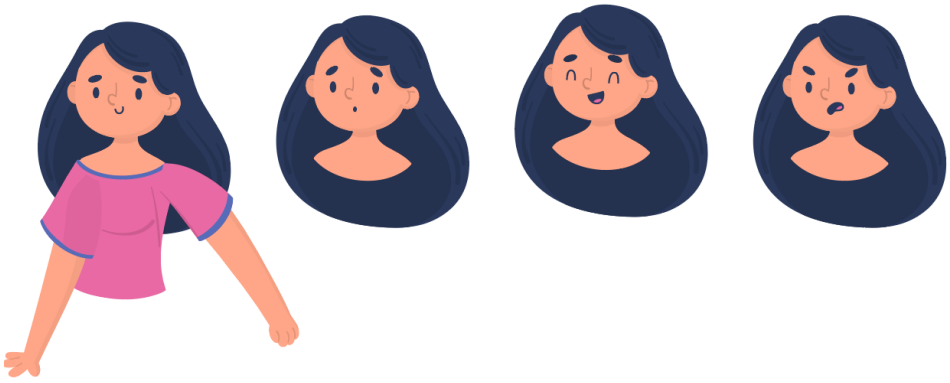
Una vez descargado abrirás el archivo con algún editor de imágenes en este caso utilizaremos Adobe Illustrator:.

3528087	Archivo JPG	414 KB	No	475 KB	13%	23/02/2020 09:56 p. m.
3528088	Adobe Illustrator Artwork ...	534 KB	No	555 KB	4%	23/02/2020 09:56 p. m.
3528089	Encapsulated PostScript	625 KB	No	1,809 KB	66%	23/02/2020 09:56 p. m.
License free	Documento de texto	1 KB	No	2 KB	51%	24/02/2020 01:44 p. m.
License premium	Documento de texto	1 KB	No	2 KB	51%	24/02/2020 01:44 p. m.

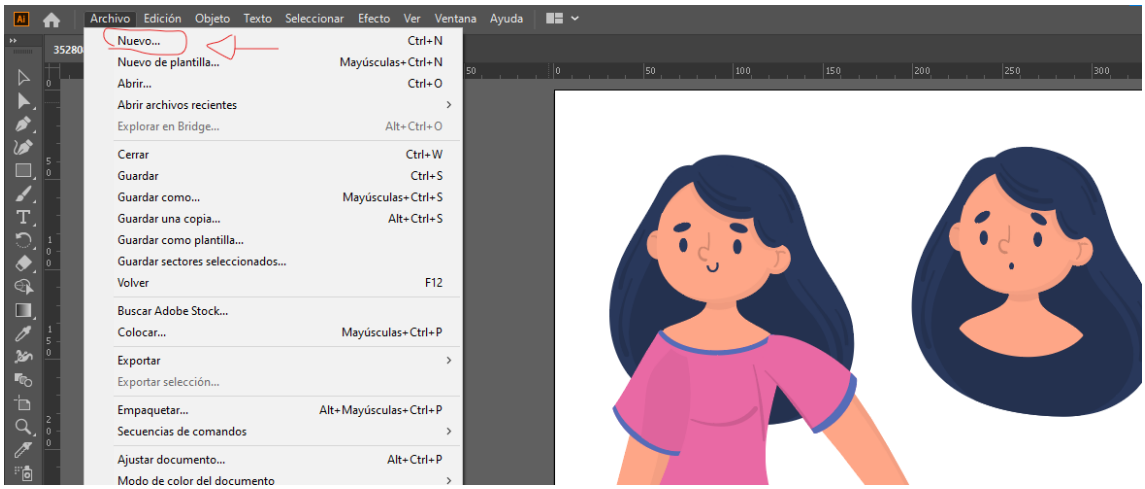
Una vez aquí solo nos concentraremos en el rostro así que podemos suprimir todo el cuerpo menos el rostro.

Para eliminar solo el cuerpo tendremos que dar por lo menos 2 Clicks y seguido de eso en suprimir en la parte del cuerpo que deseemos eliminar.

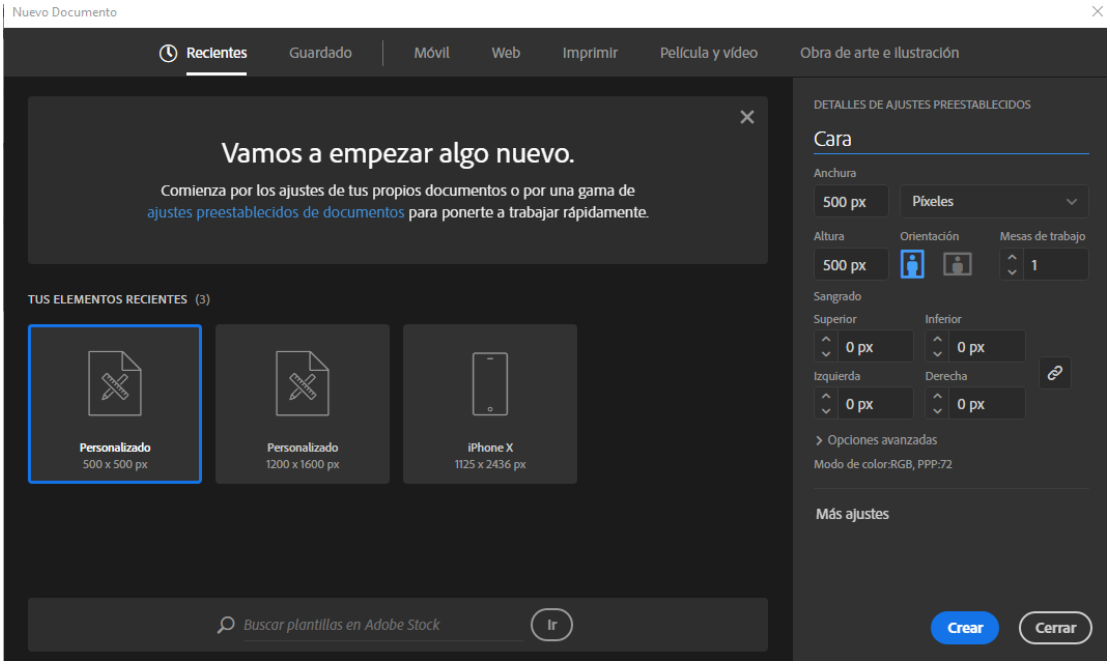
Podemos jugar con los brazos y mover las piezas que queremos poner.



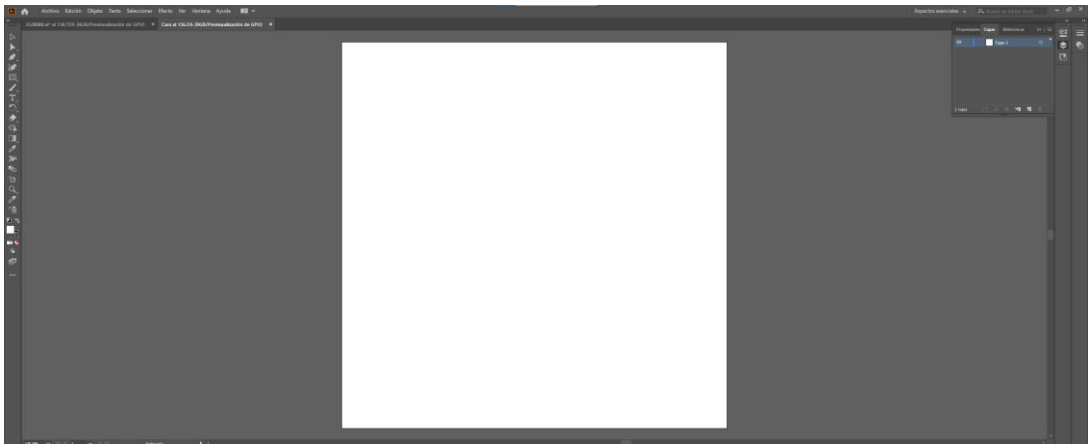
Una vez tengamos las piezas que ocuparemos. Crearemos un archivo nuevo:



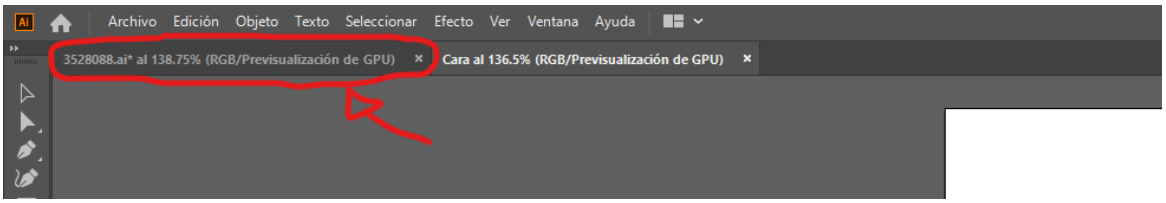
Ese archivo tendrá las siguientes medidas y ajustes, también podrás poner el nombre que te guste. Una vez puesto los datos darás en crear



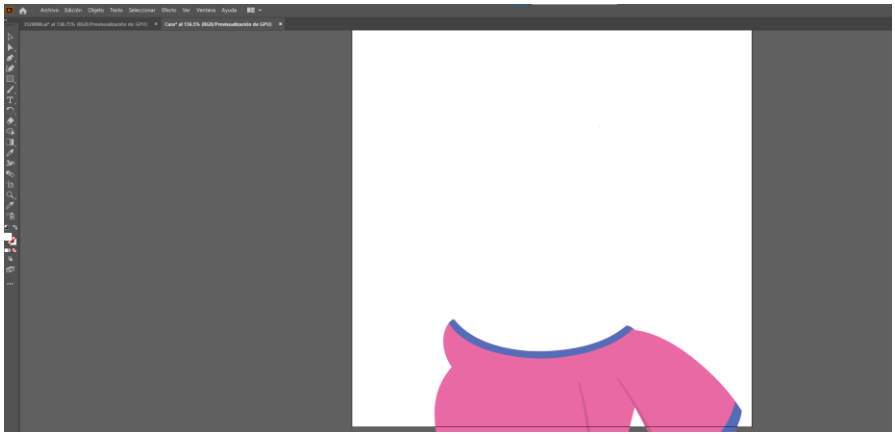
Se te mostrara una mesa como esta:



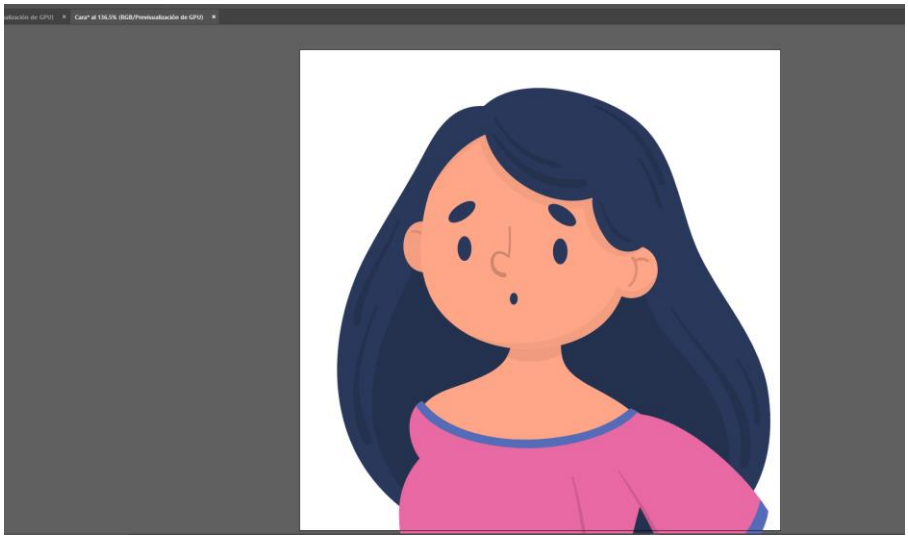
En la cual vamos a copiar y pegar lo que queremos crear. En este caso lo que tenemos en el otro archivo:



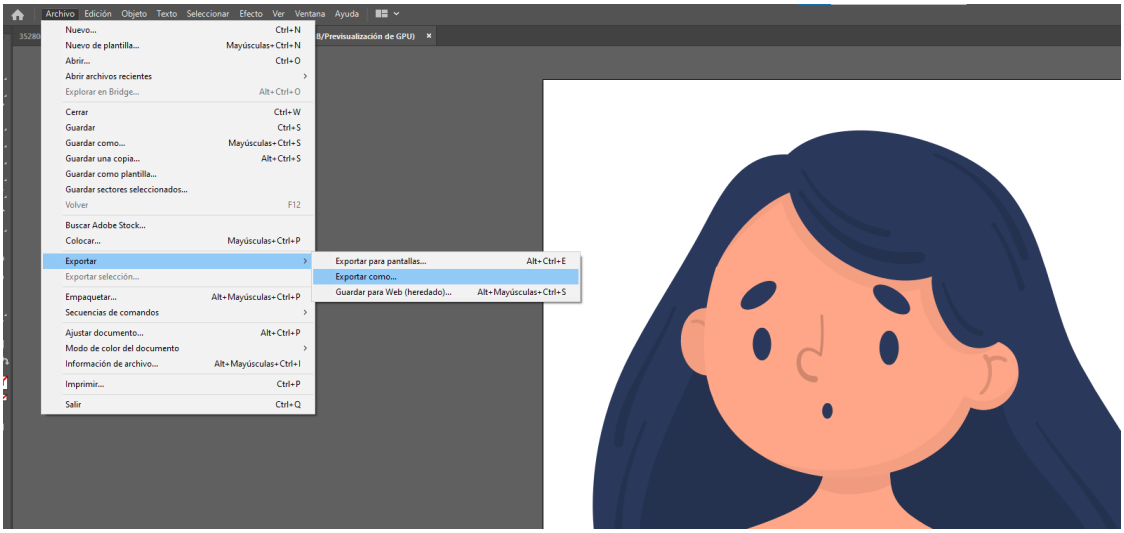
Vas a copiar lo que es cuerpo del personaje sin la cabeza.



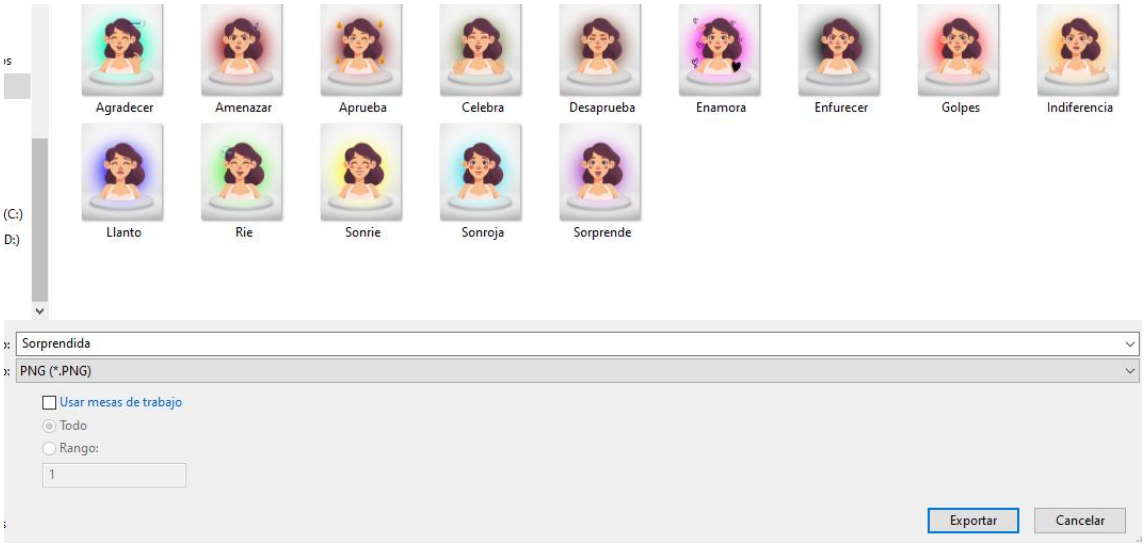
Después copiaras y pegaras el rostro que desees.



Para generar las imágenes y finalizar. Seleccionaras en archivo la opción exportar y después en exportar como:



Se abrirá esta ventana en la que podrás poner el nombre que desees y selecciona la opción PNG para guardarla como imagen y das en exportar.



PRUEBAS Y RESULTADOS

Debido a la gran cantidad de resultados que puede otorgar la máquina de estados de nuestro programa solo mostraremos 4 recorridos de prueba que puedan demostrar el funcionamiento de Virtual Love. Se mostrarán los resultados de la aplicación con la que interactúa el usuario y el recorrido de simulación generado en JFLAP para que se puedan apreciar mejor los resultados. De igual manera se pueden encontrar las tablas y diagrama de transiciones aquí.

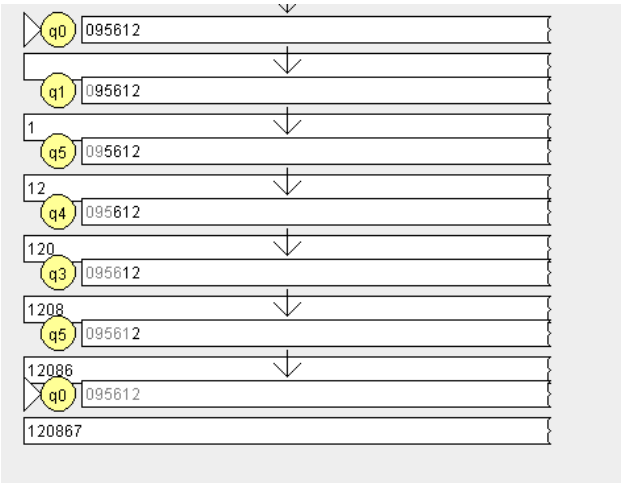
EJEMPLO 1

SUCESION:

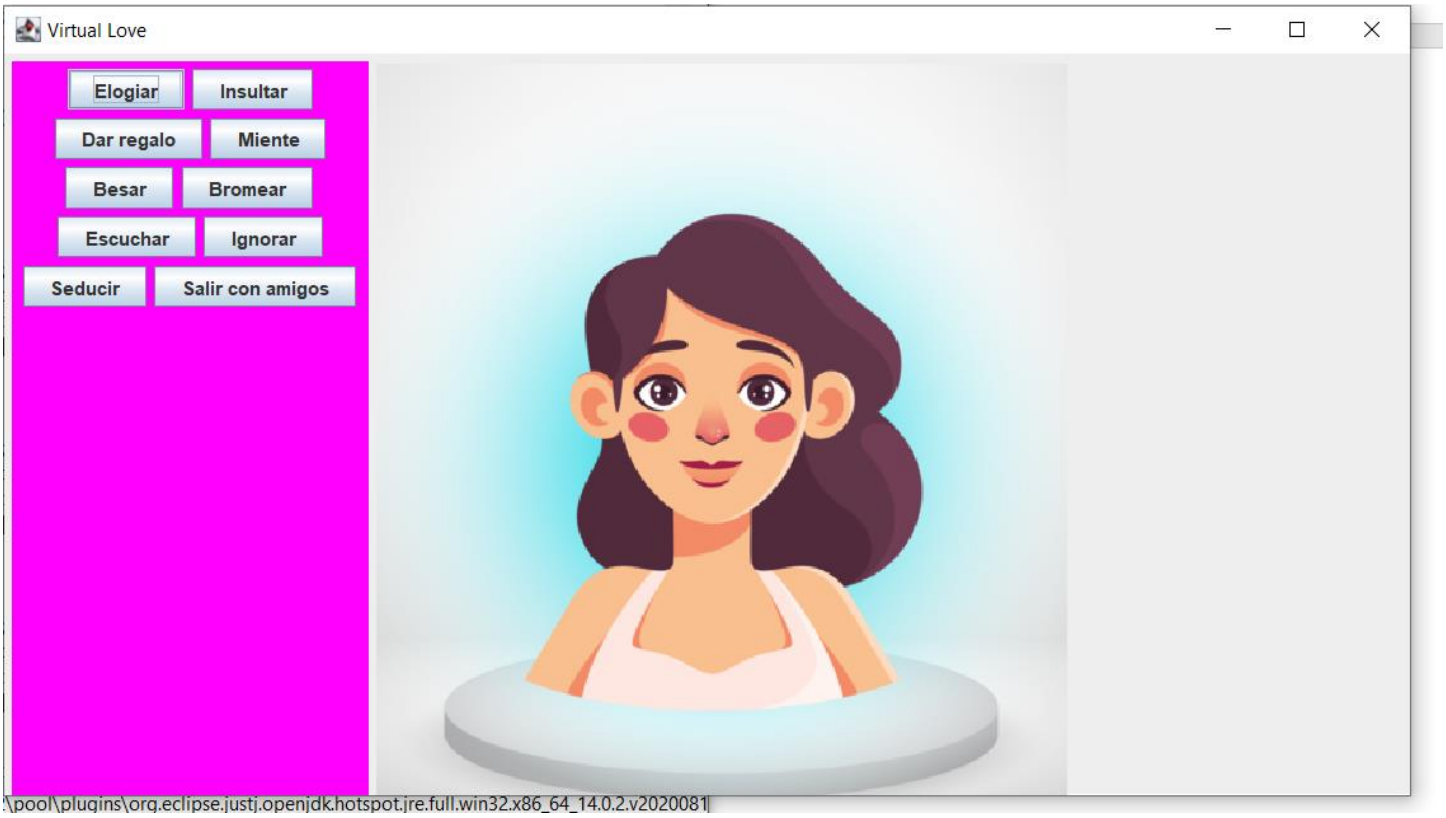
Entradas: *0,9,5,6,1,2*

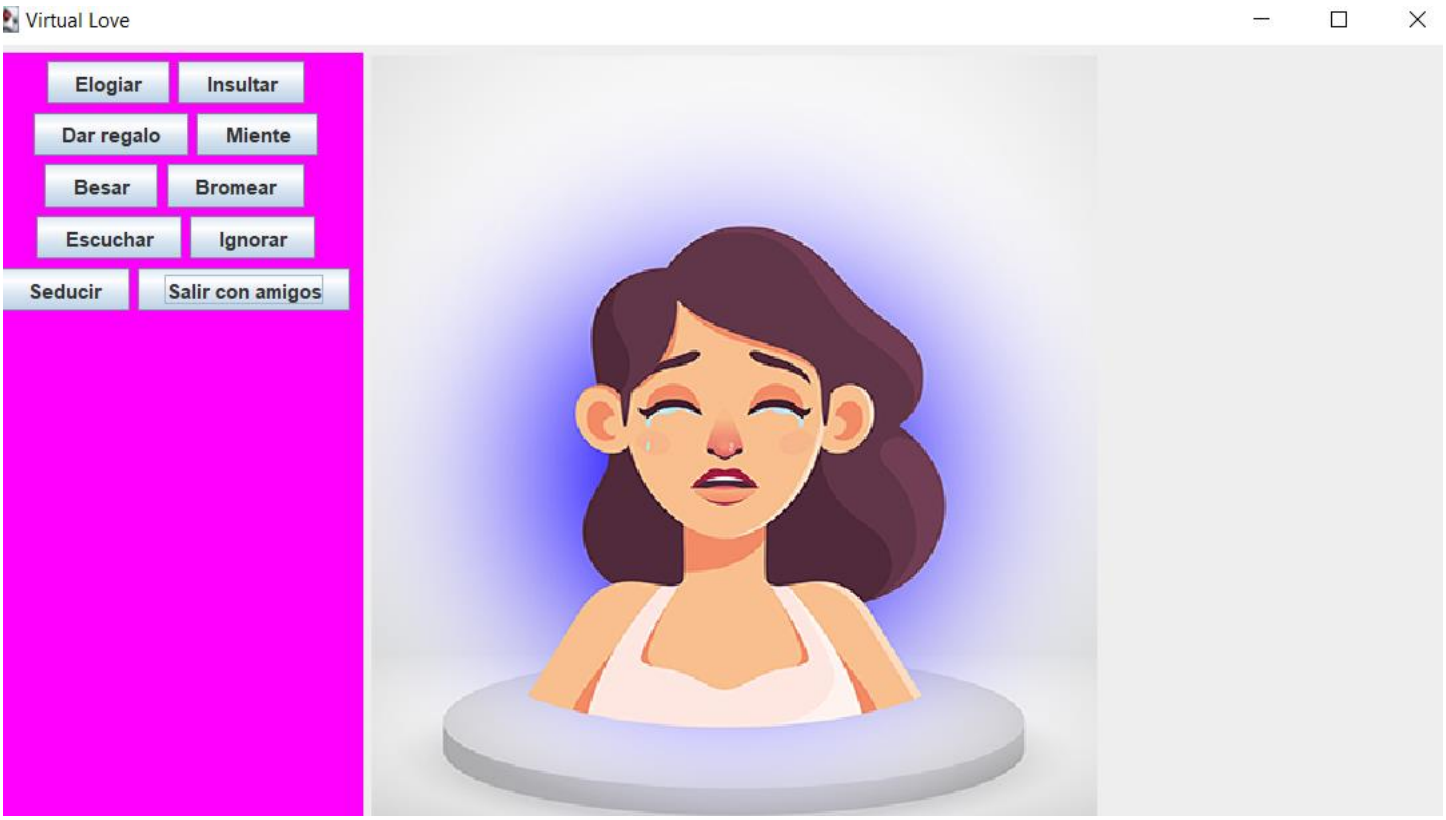
Salidas: *1,2,0,8,6,7*

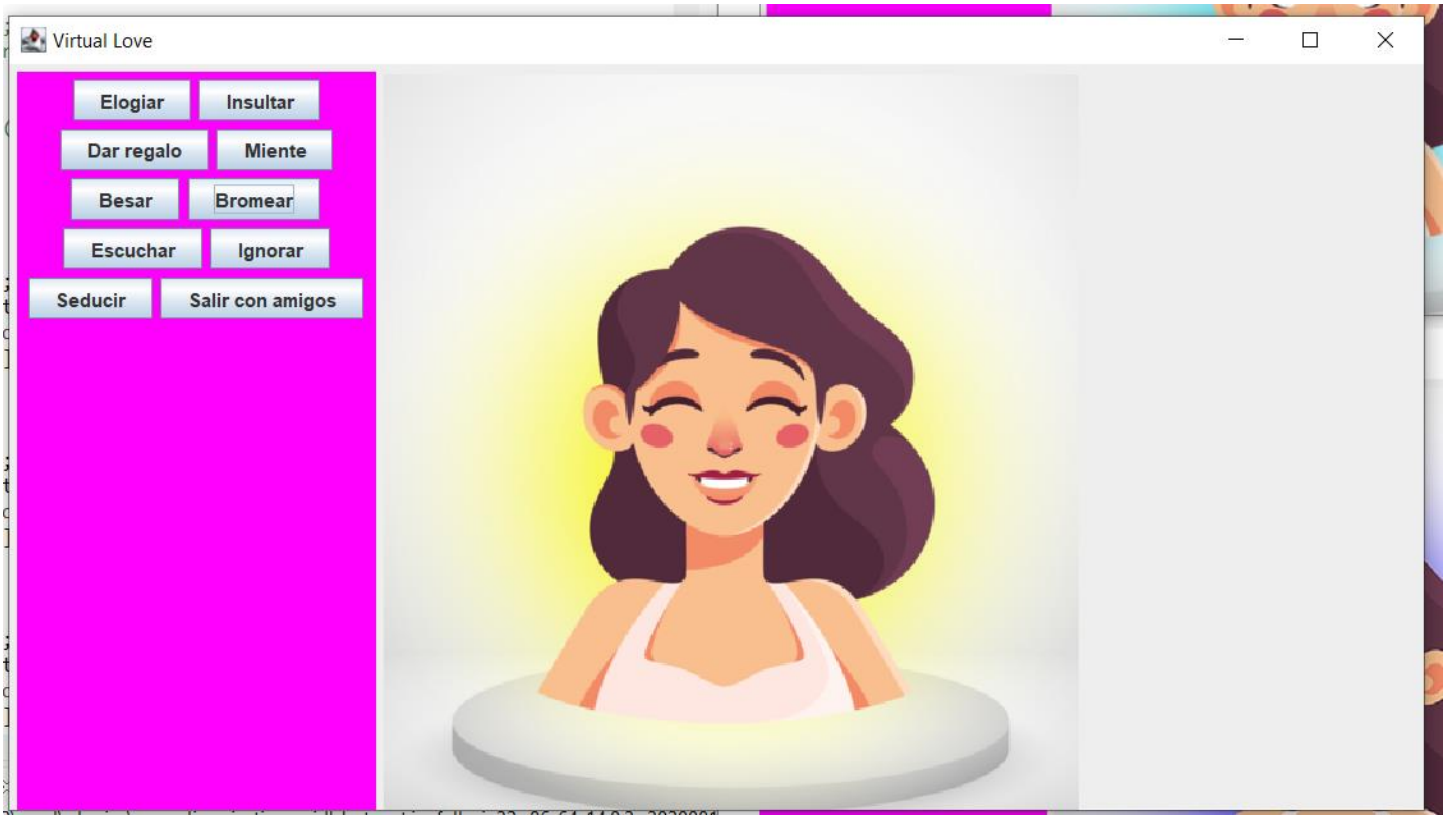
Transiciones:

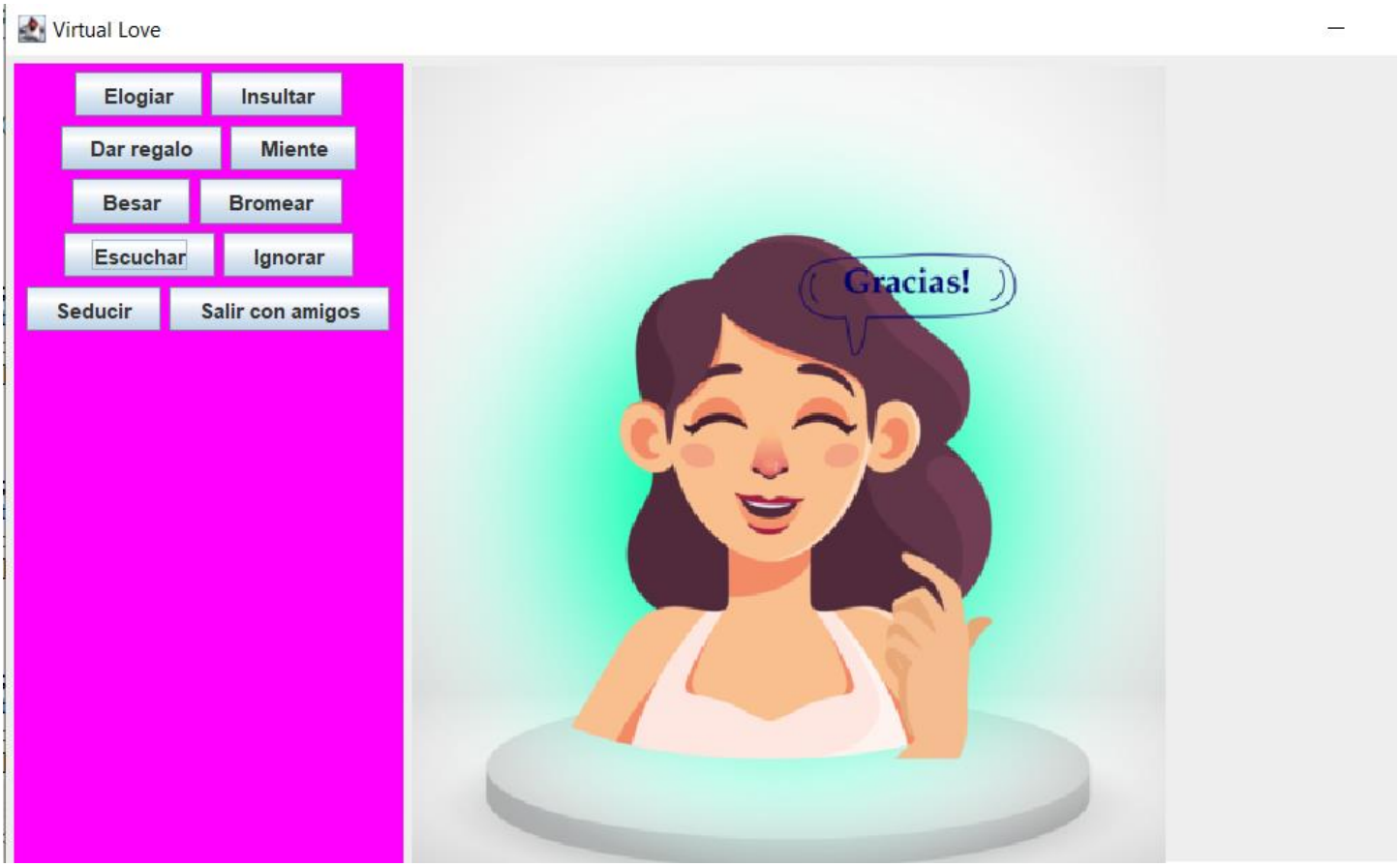


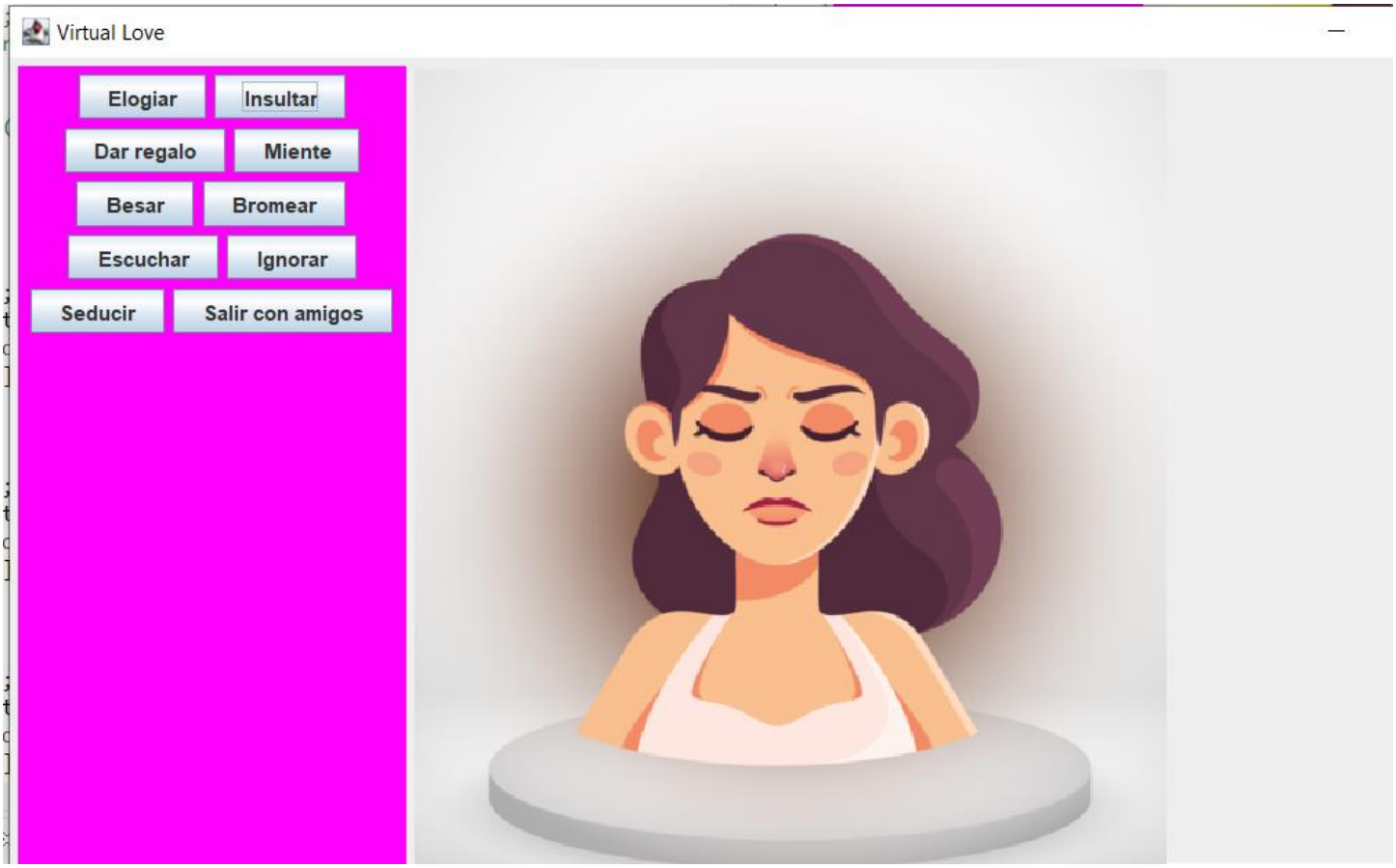
Display:

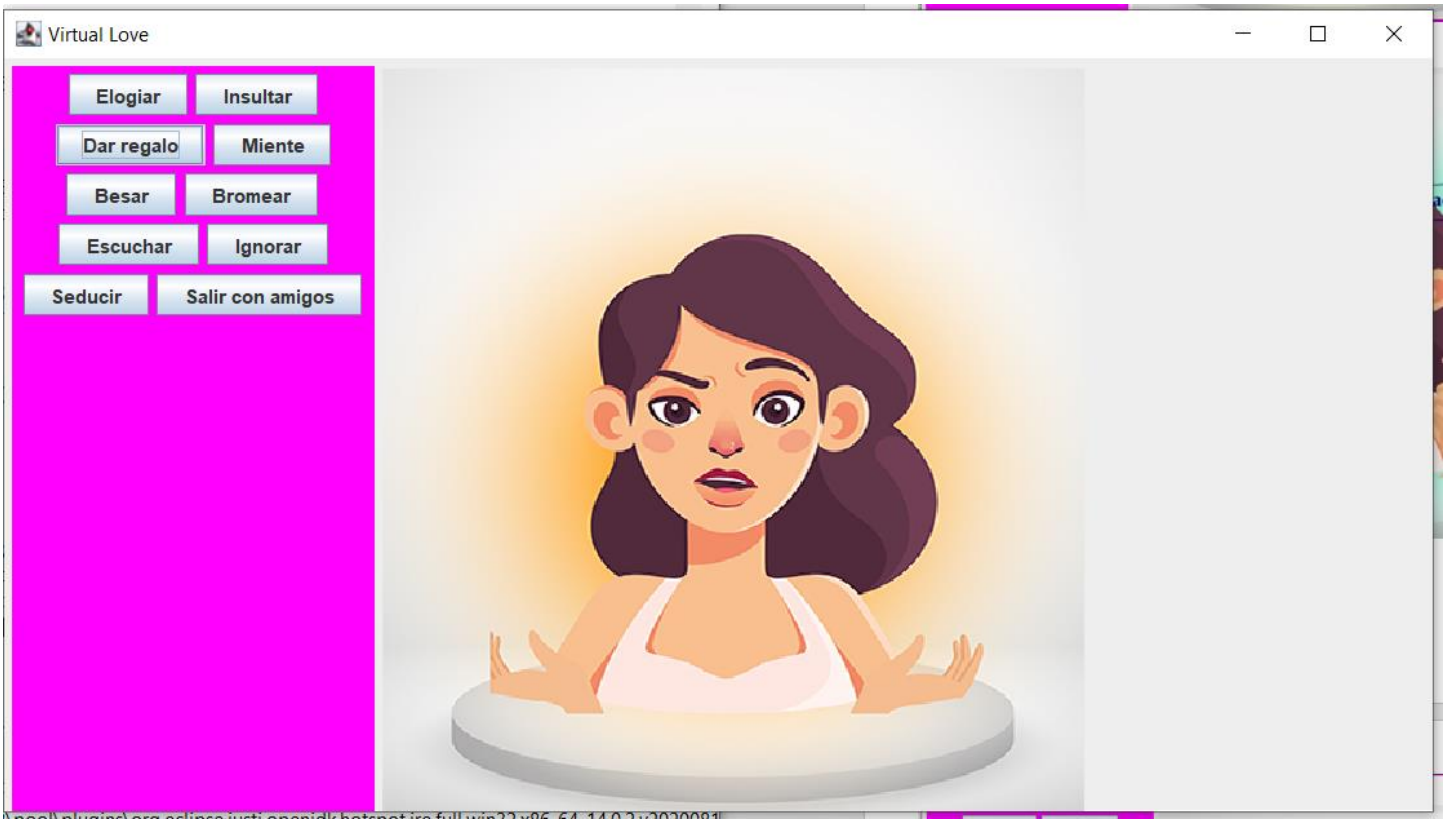

Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos
\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_14.0.2.v2020081


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos
Gracias!


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos

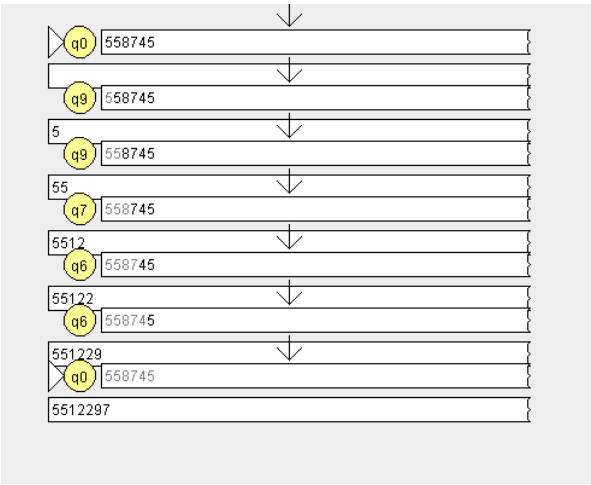
EJEMPLO 2

SUCESION:

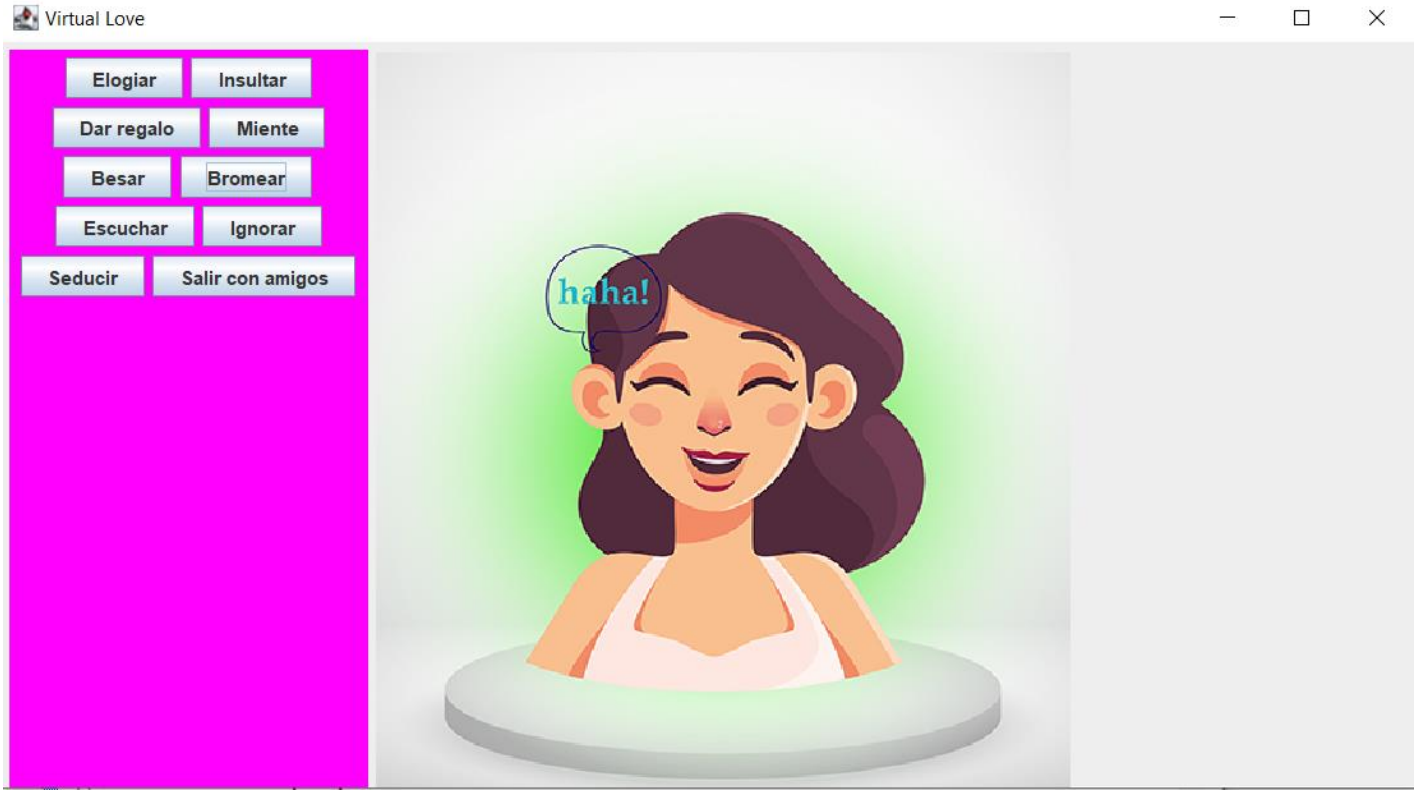
Entradas: 5,5,8,7,4,5

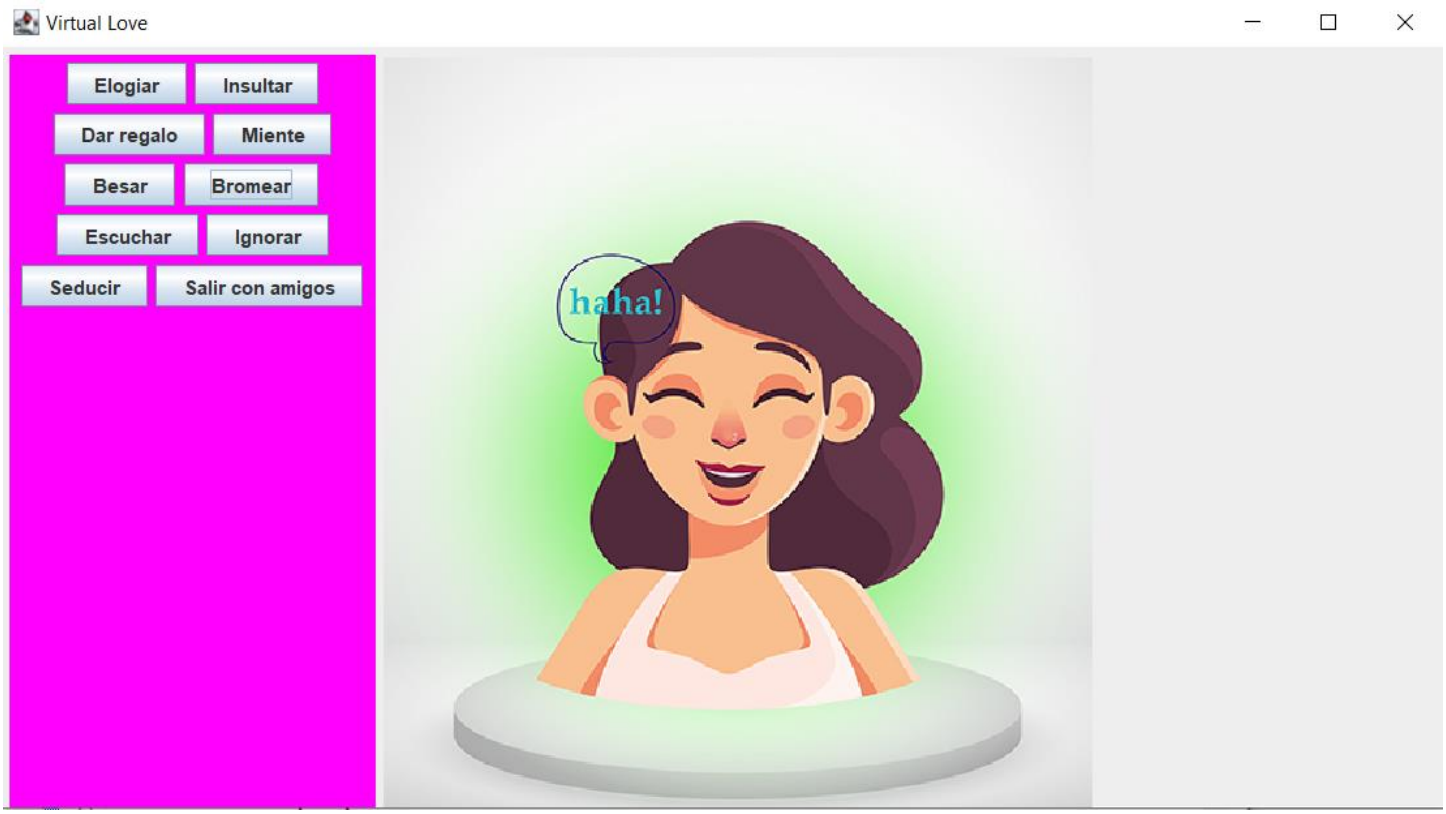
Salidas*: 5,5,12,2,9,7*

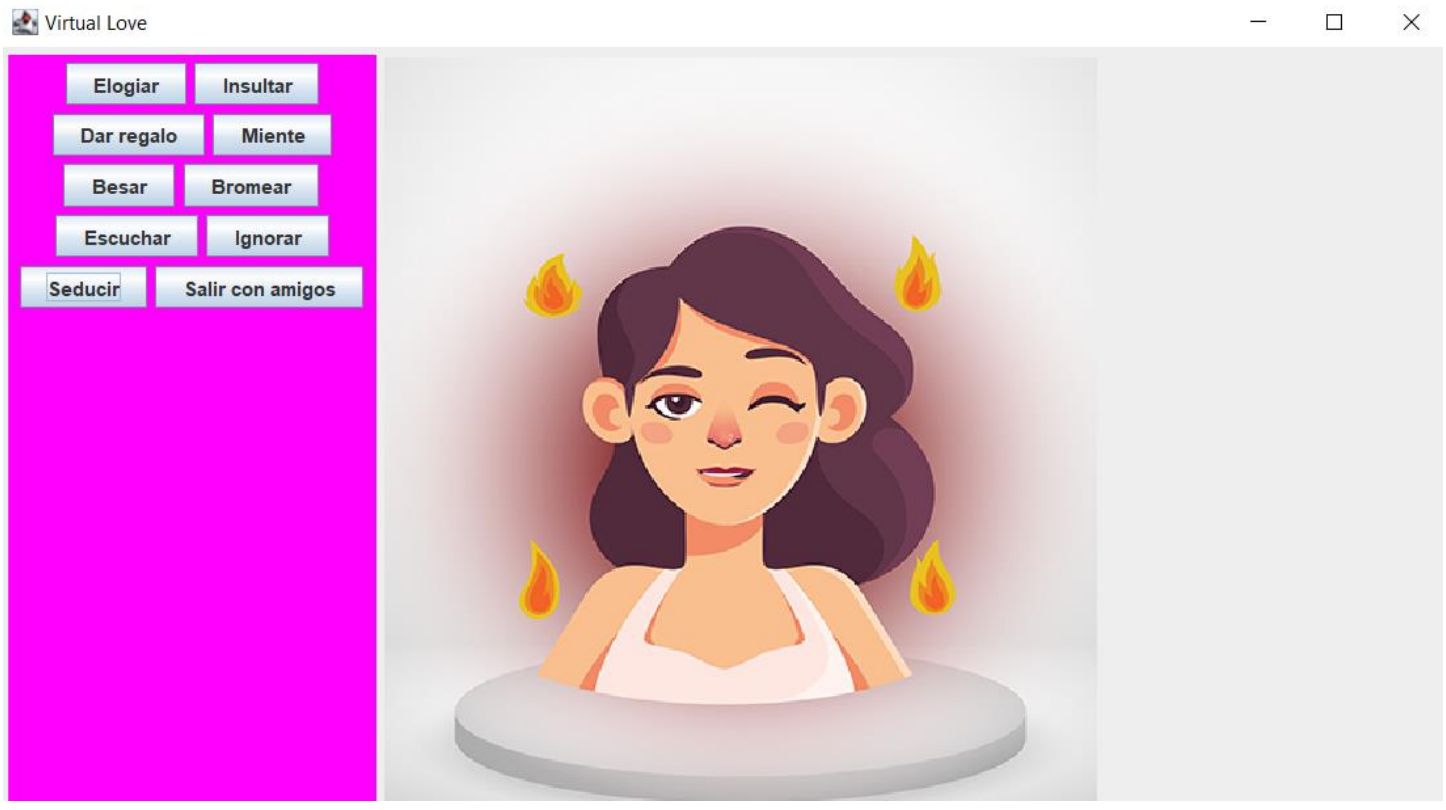
Transiciones:



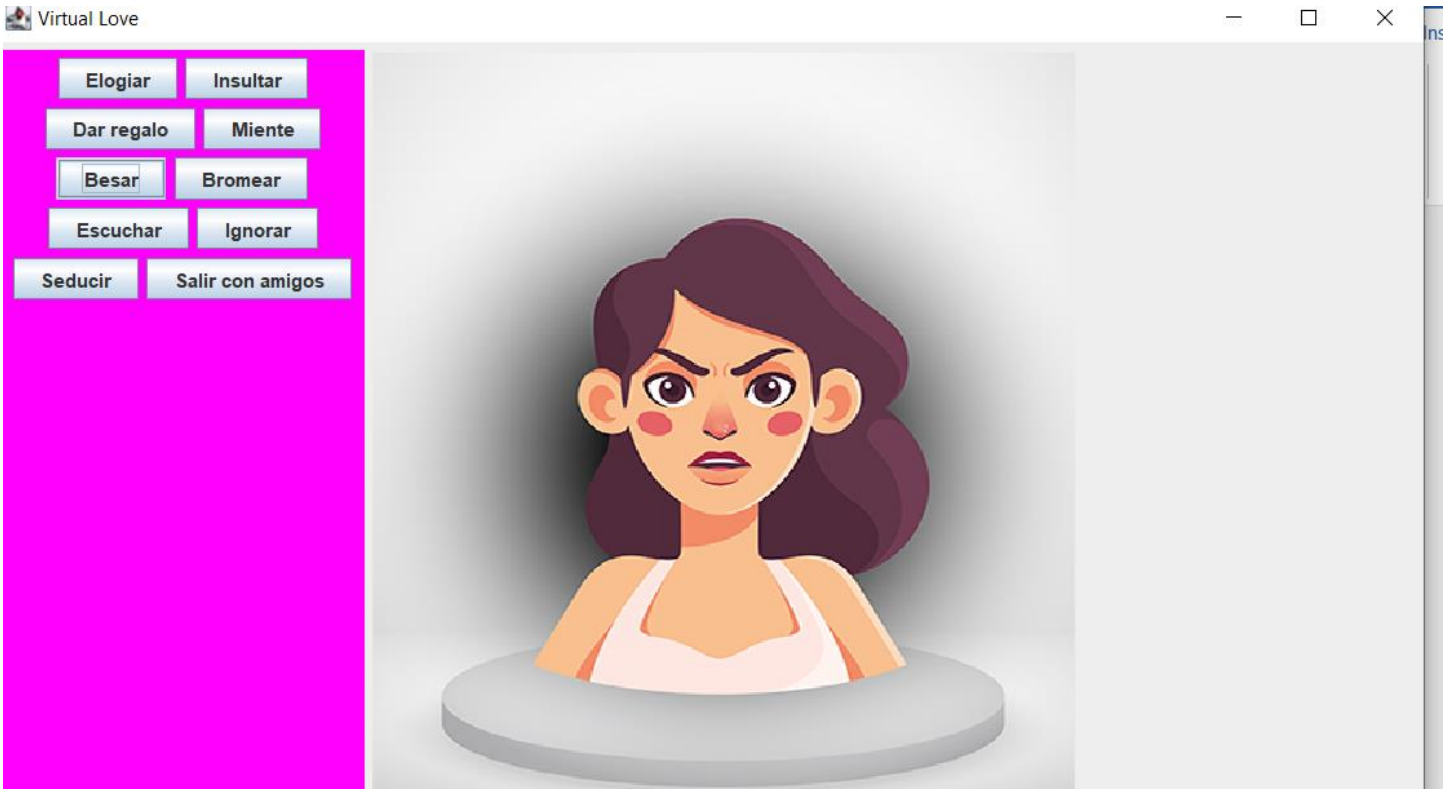
Display:

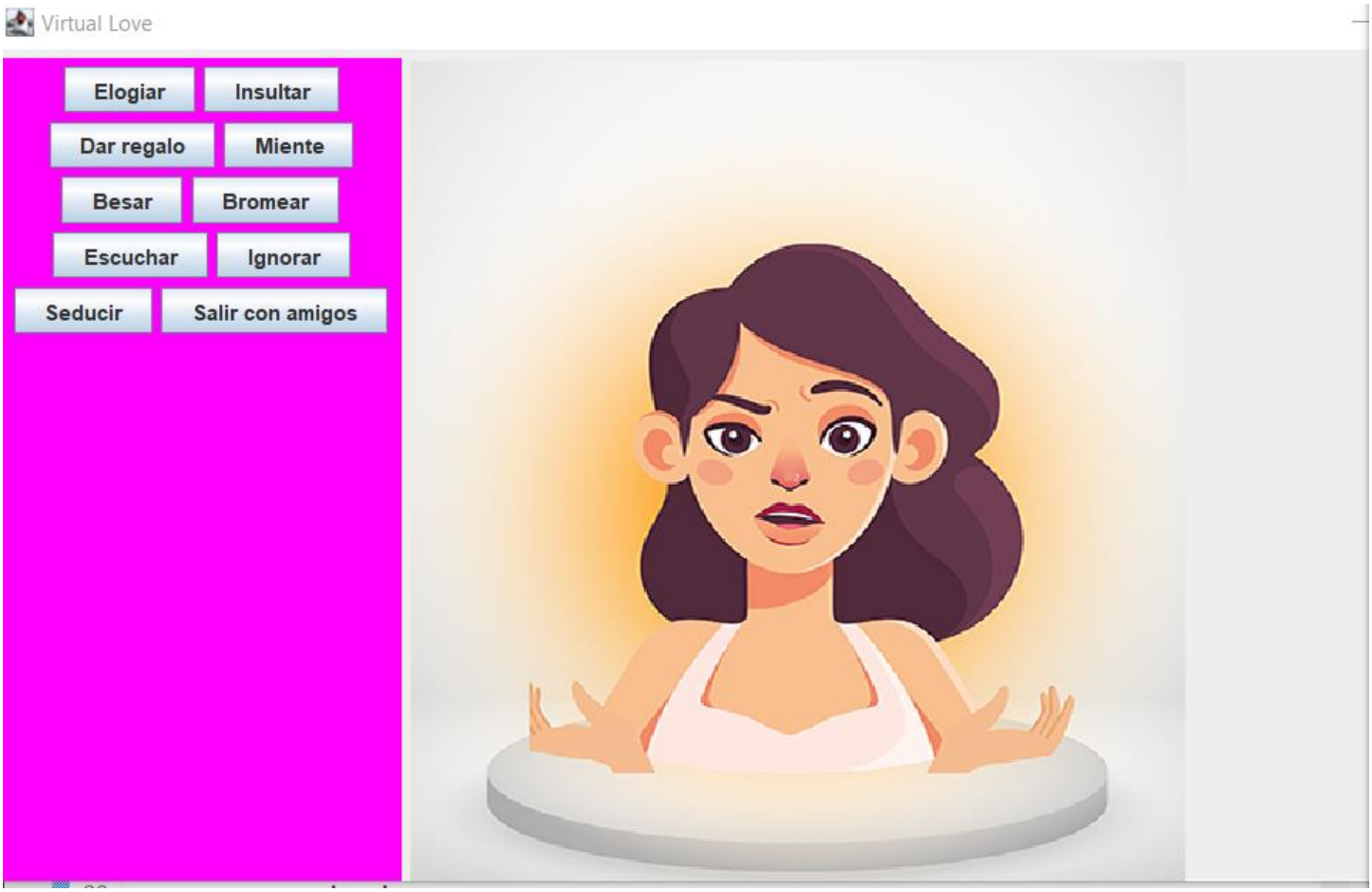



Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos
haha!


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos

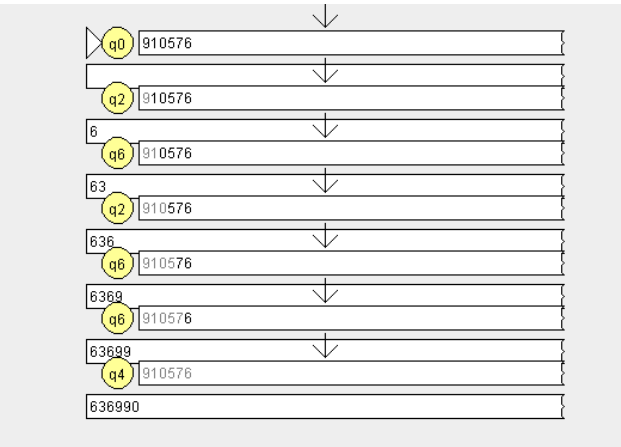
EJEMPLO 3

SUCESION:

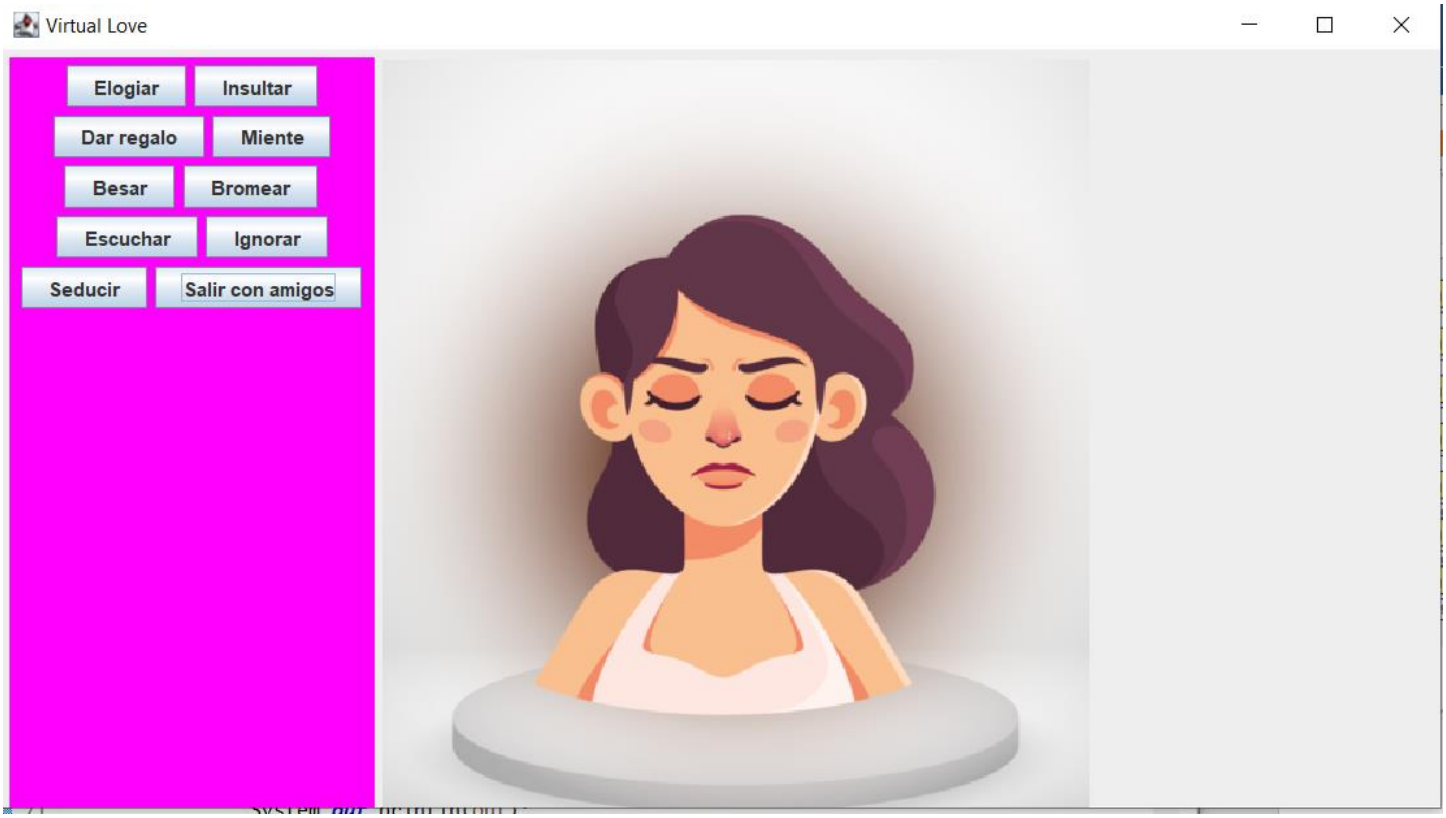
Entradas: *9,1,0,5,7,6*

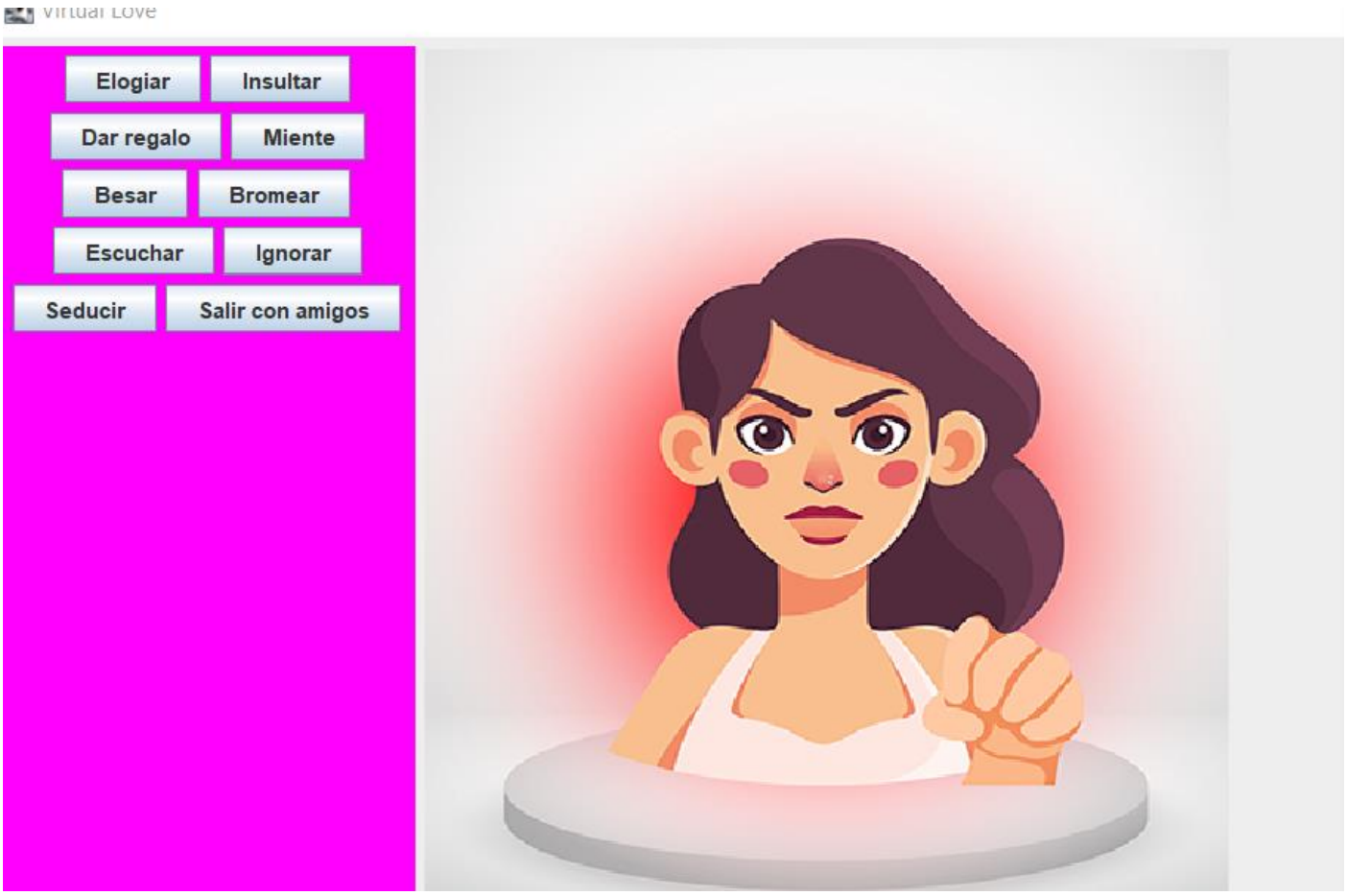
Salidas: *6,3,6,9,9,0*

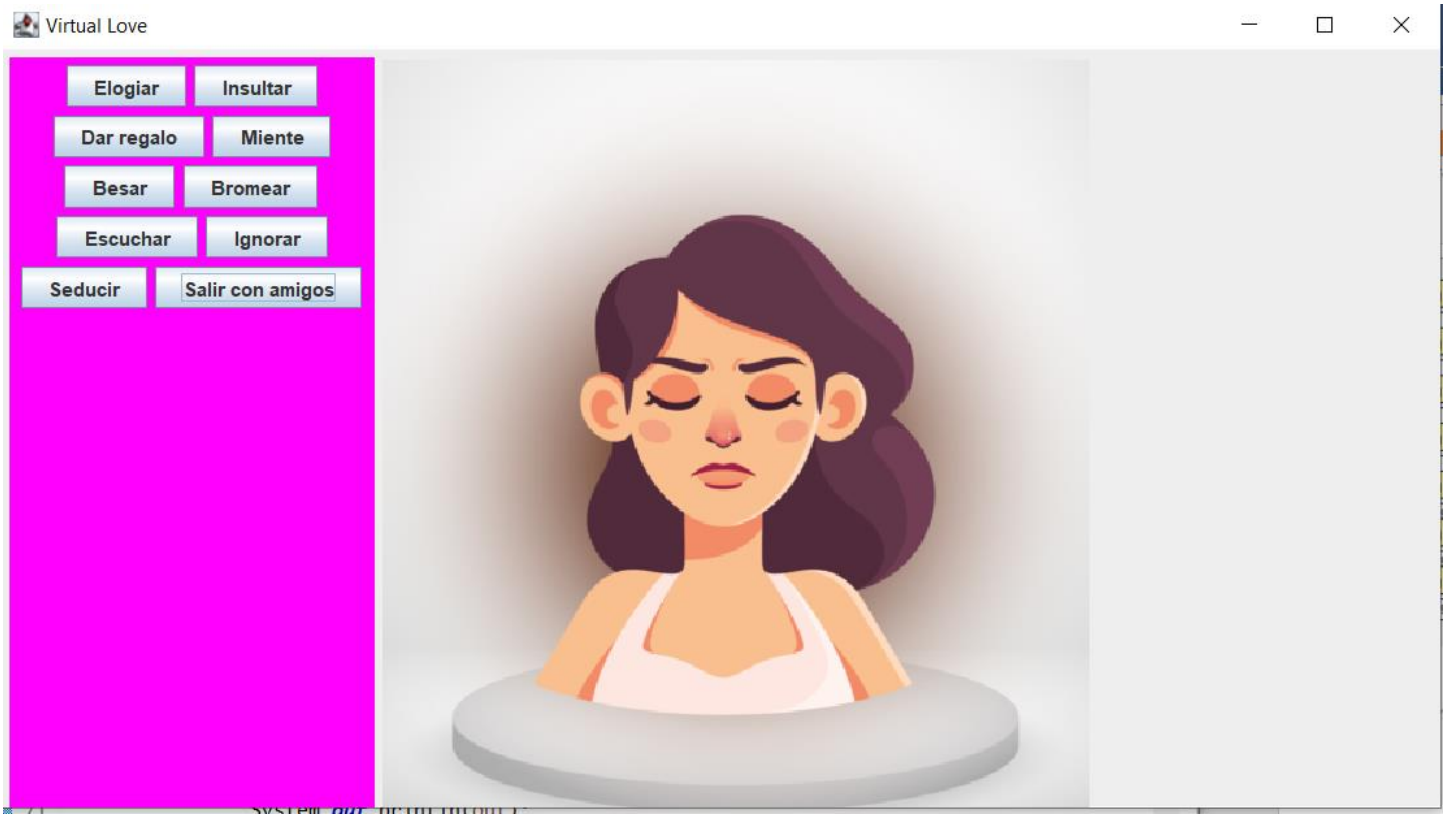
Transiciones:

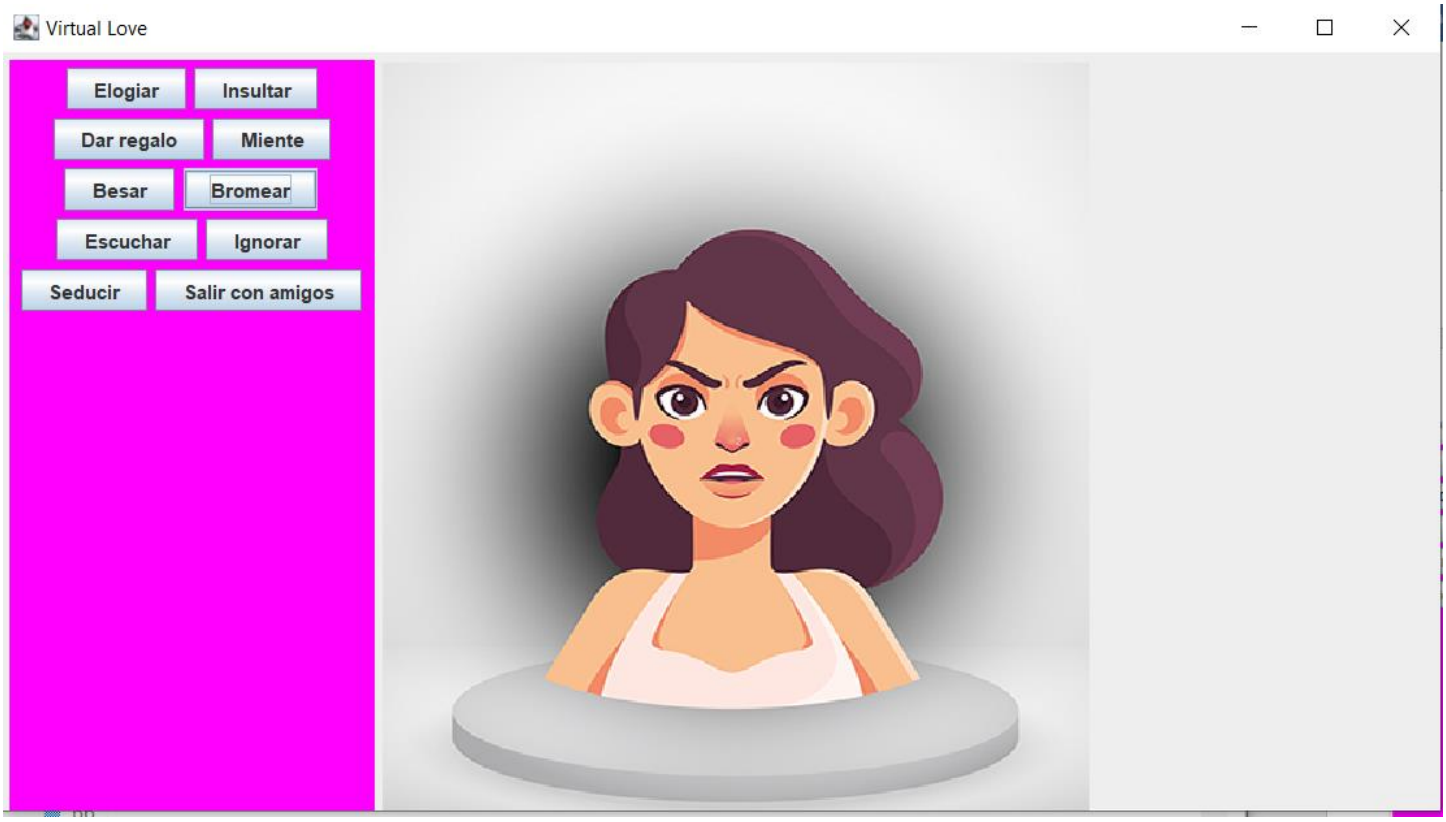


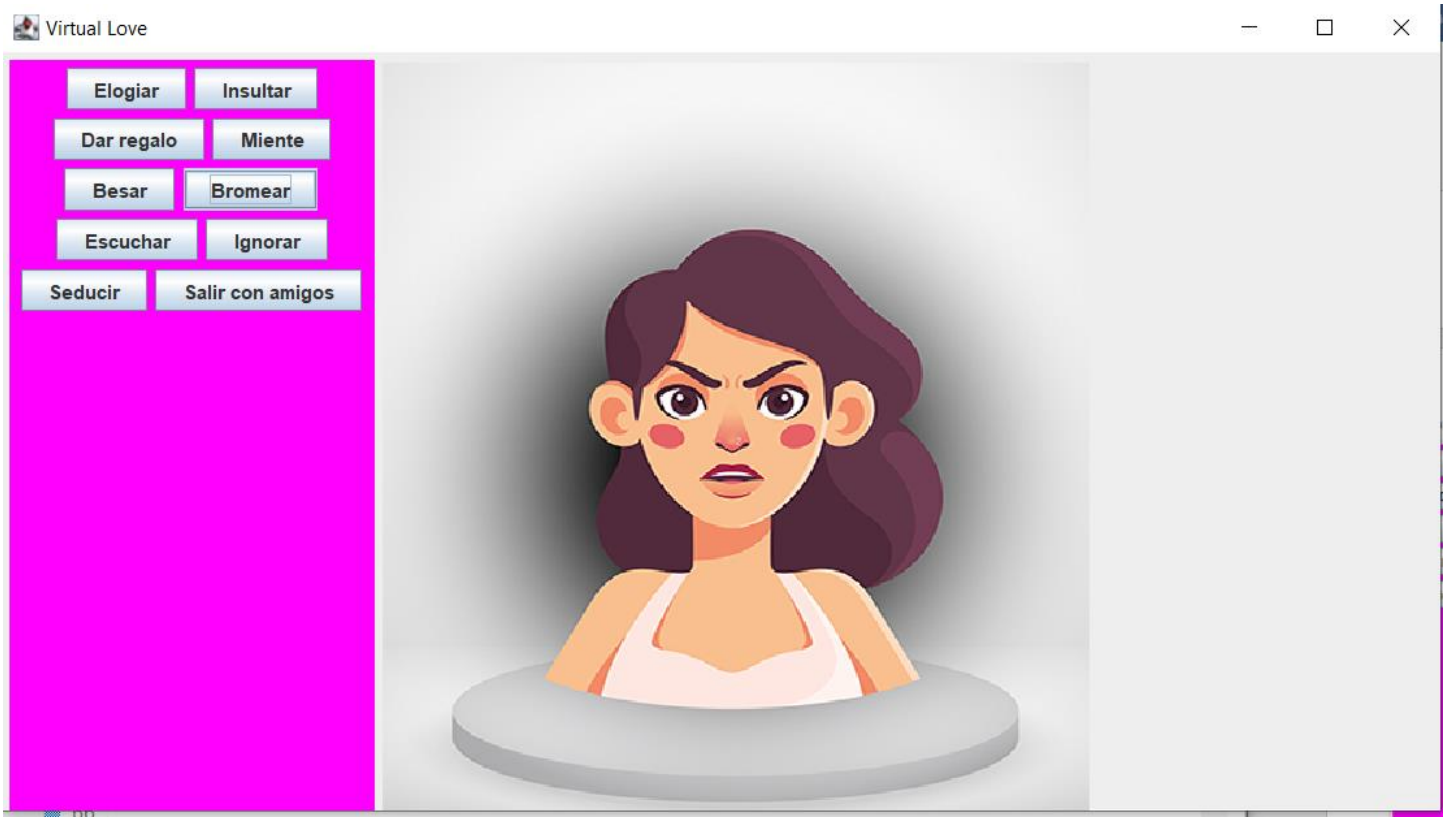
Display:

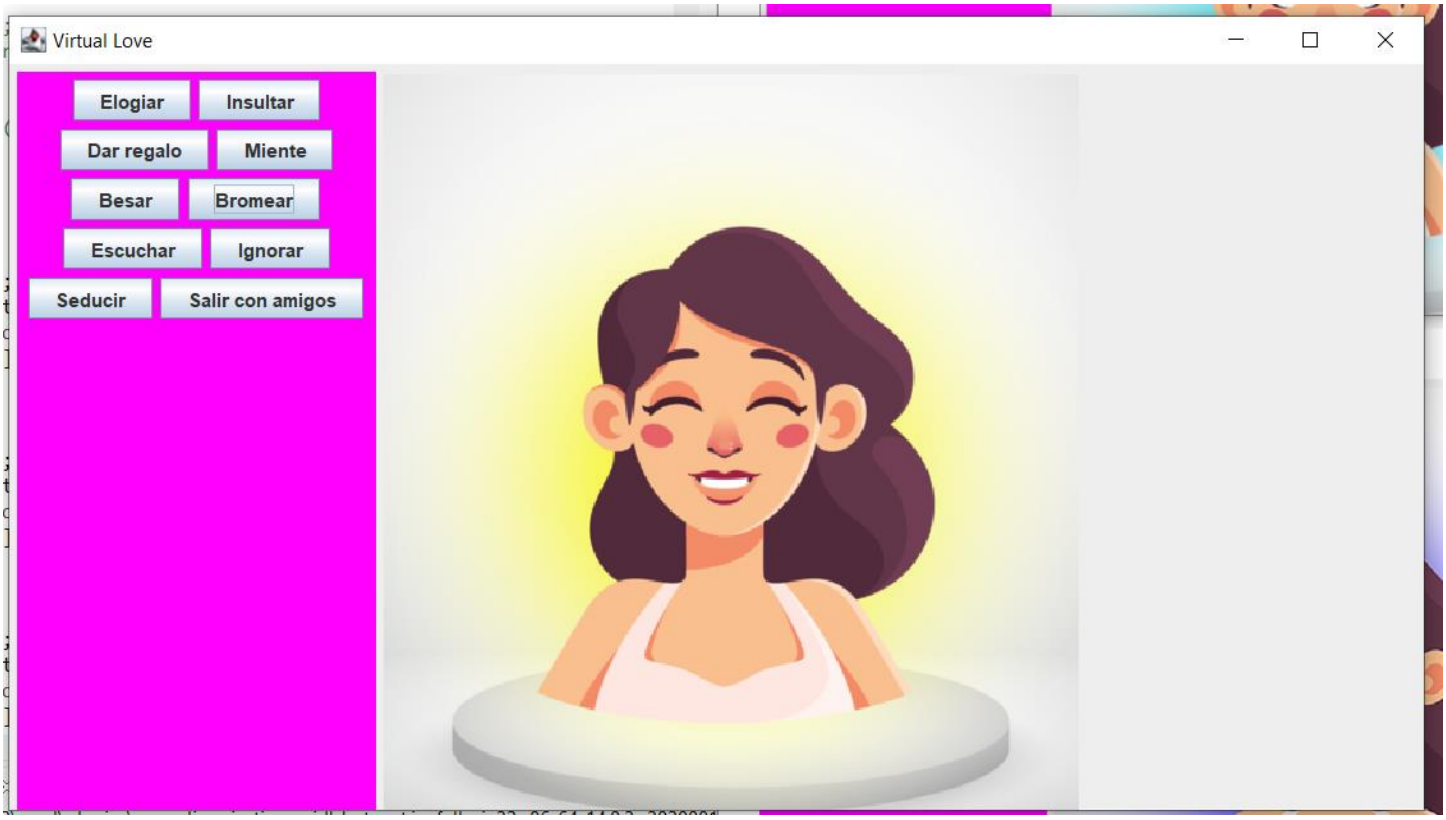



Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos

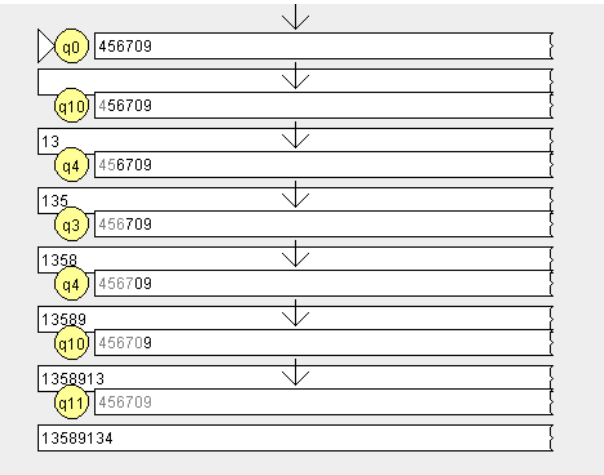
EJEMPLO 4

SUCESION:

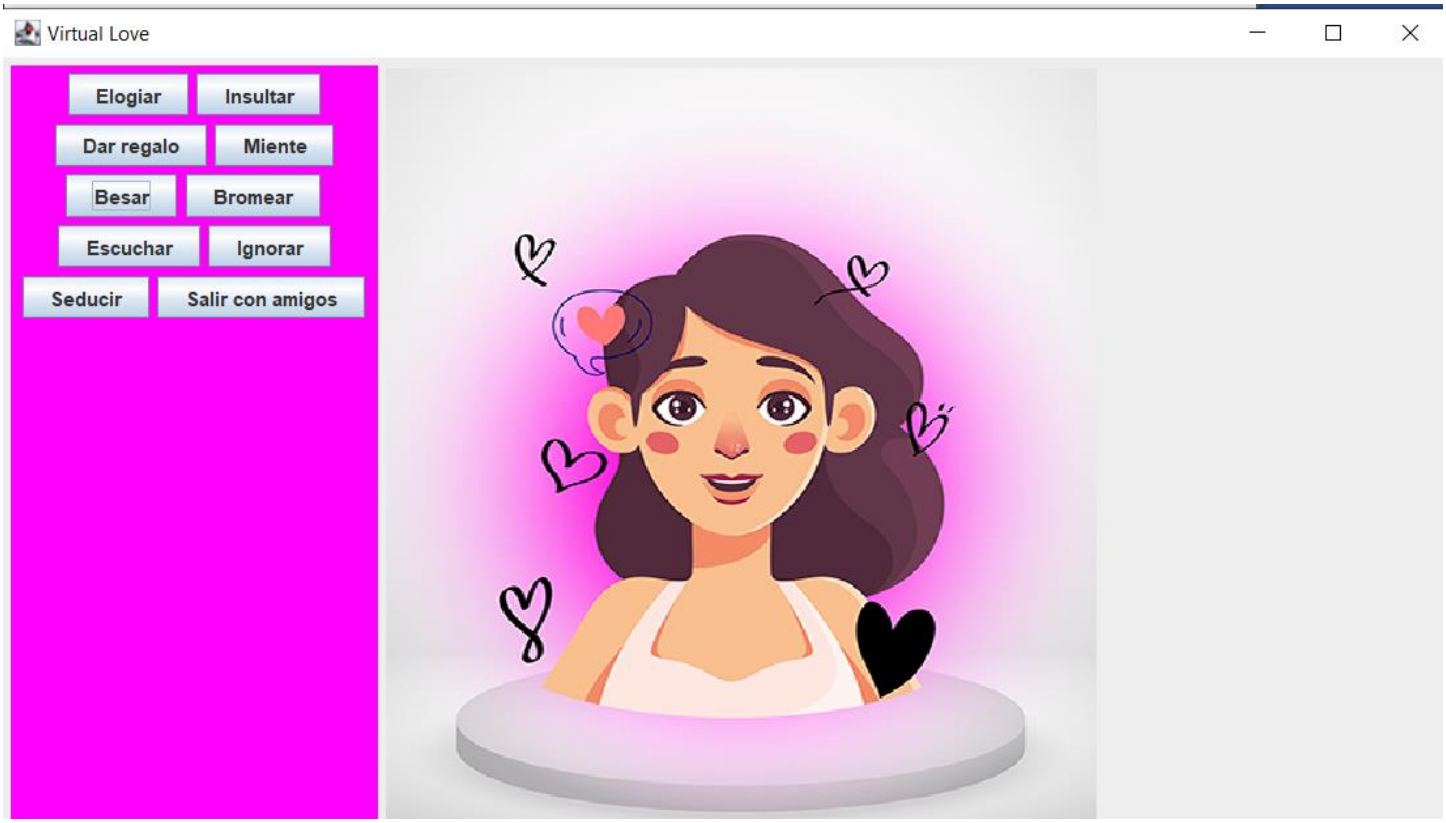
Entradas: 4,5,6,7,0,9

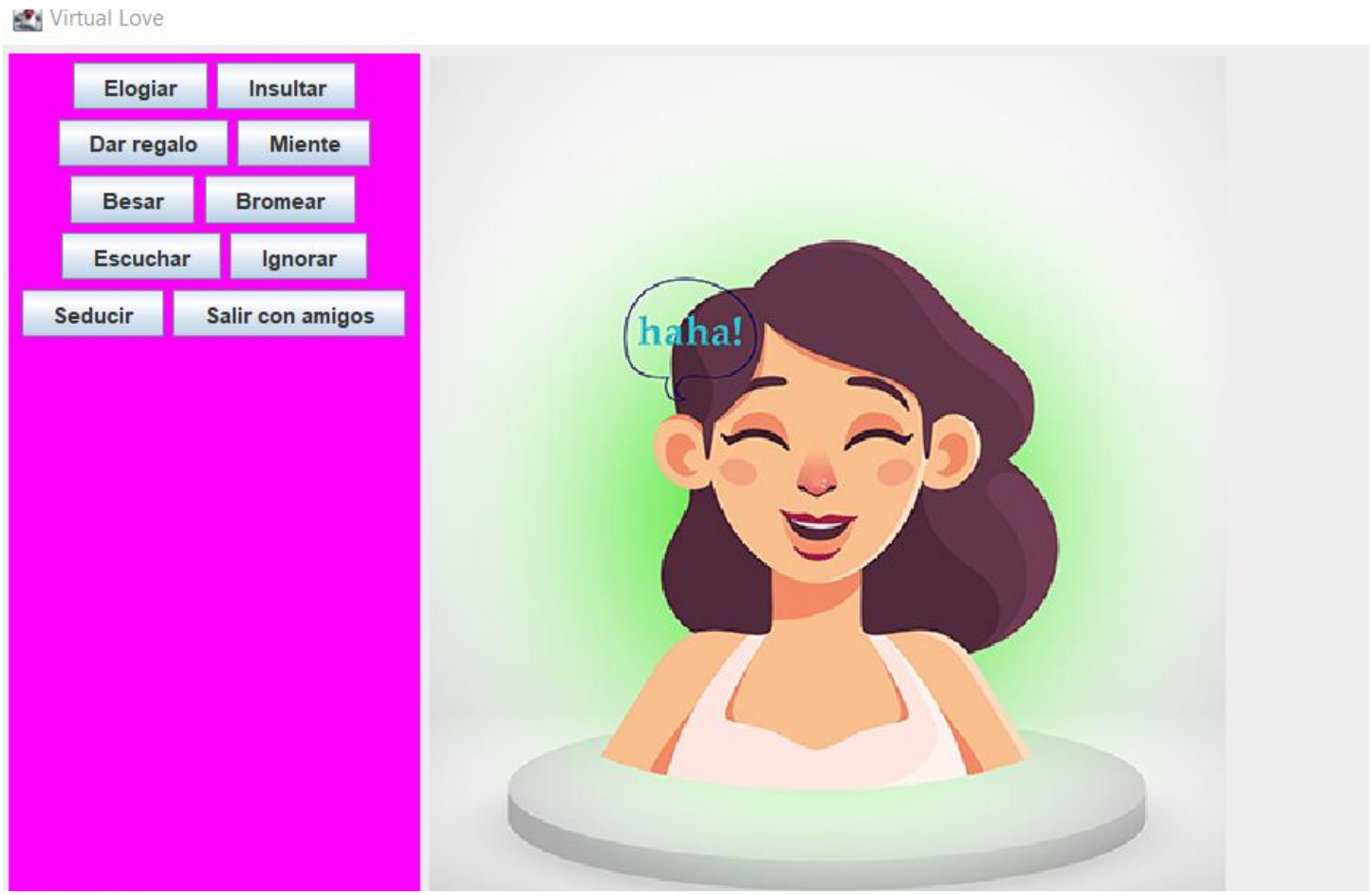
Salidas: 13,5,8,9,13,4

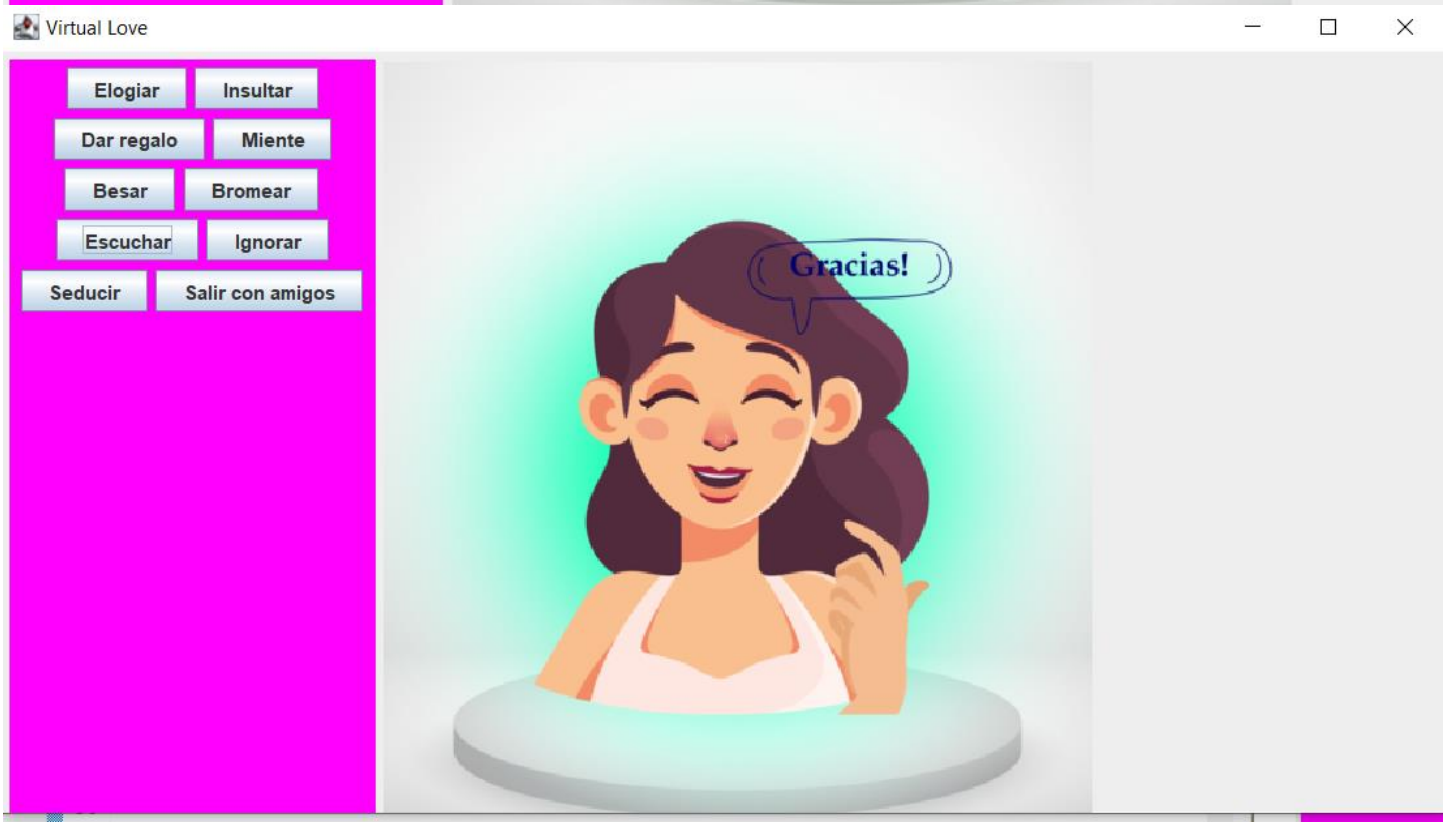
Transiciones:



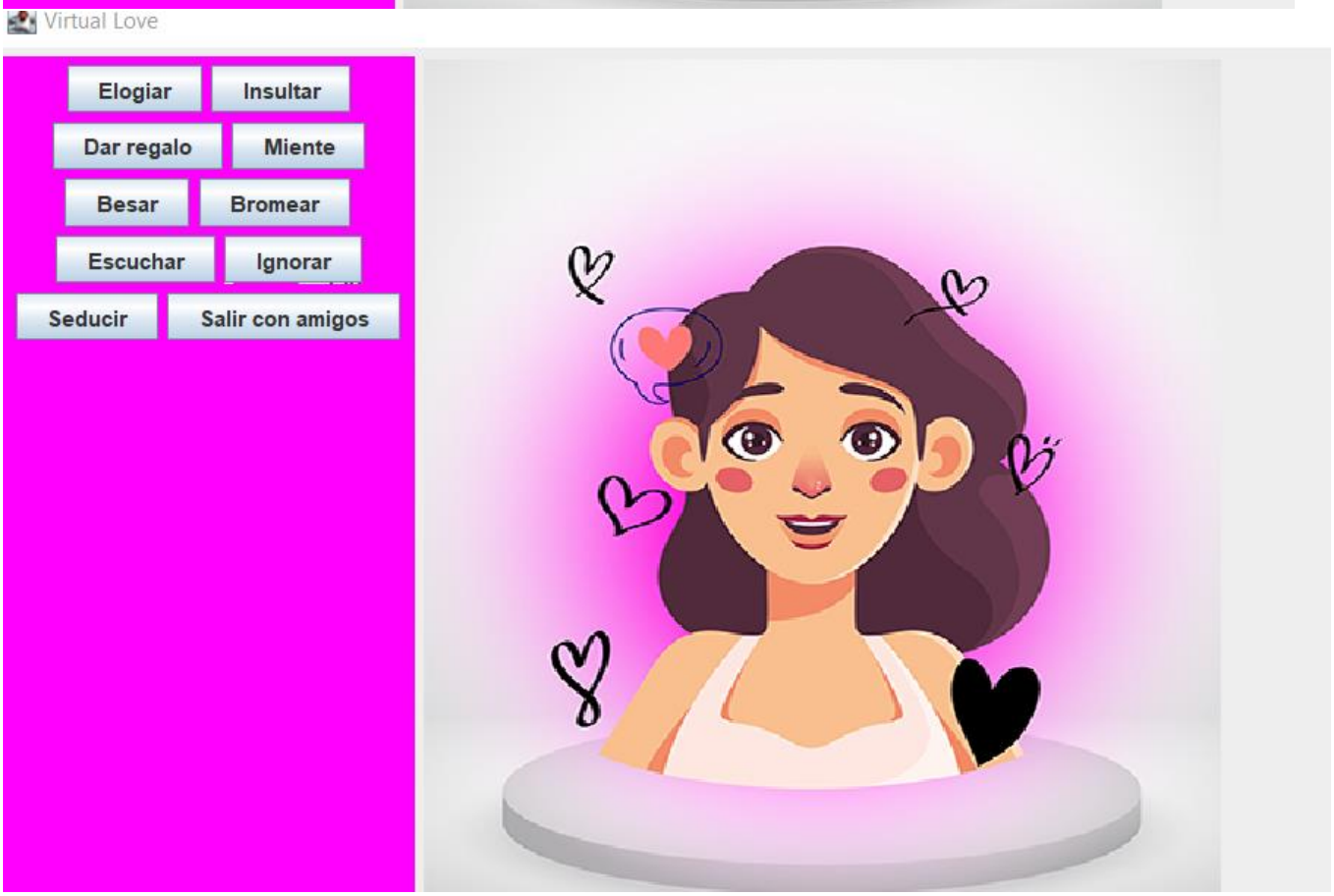
Display:

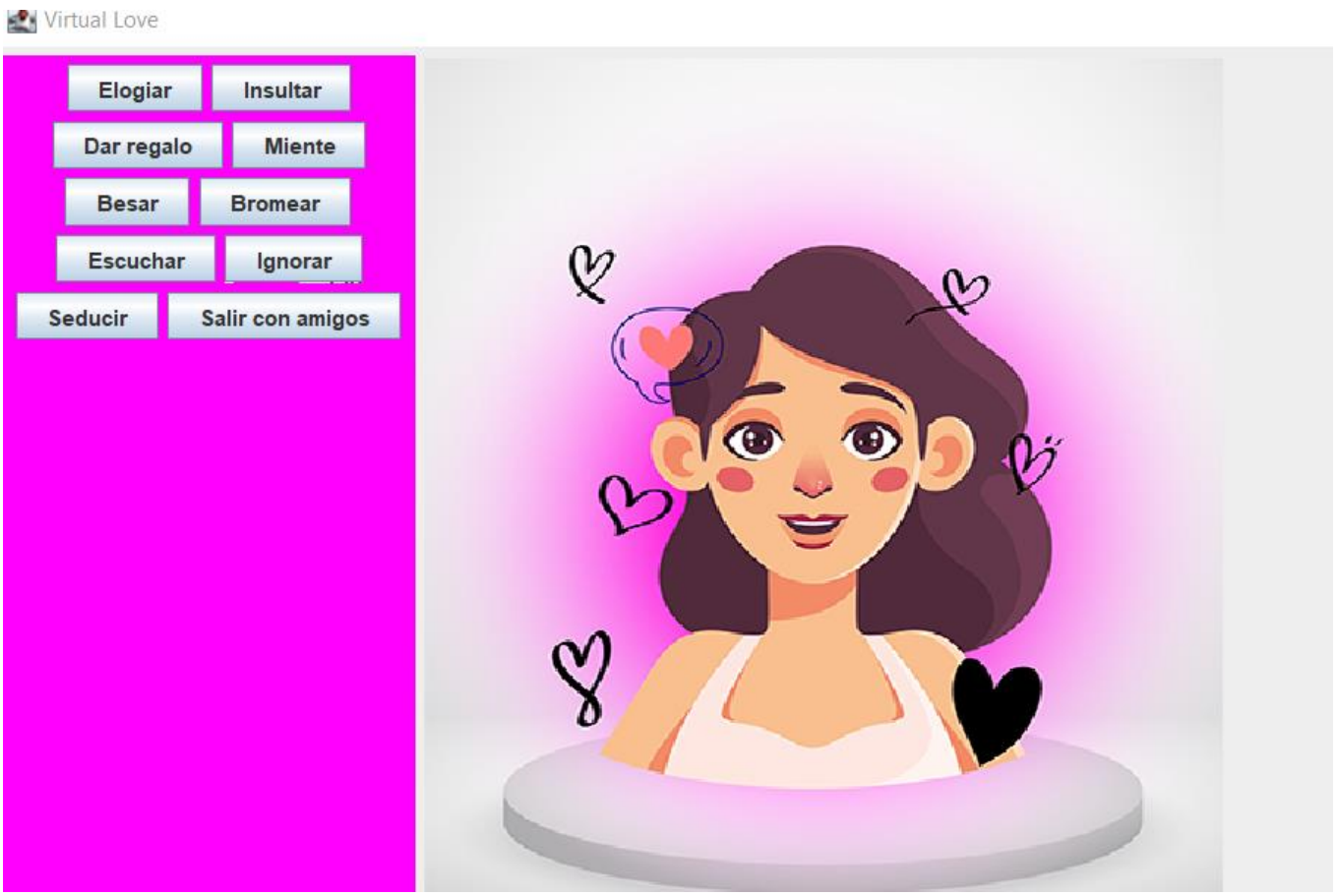


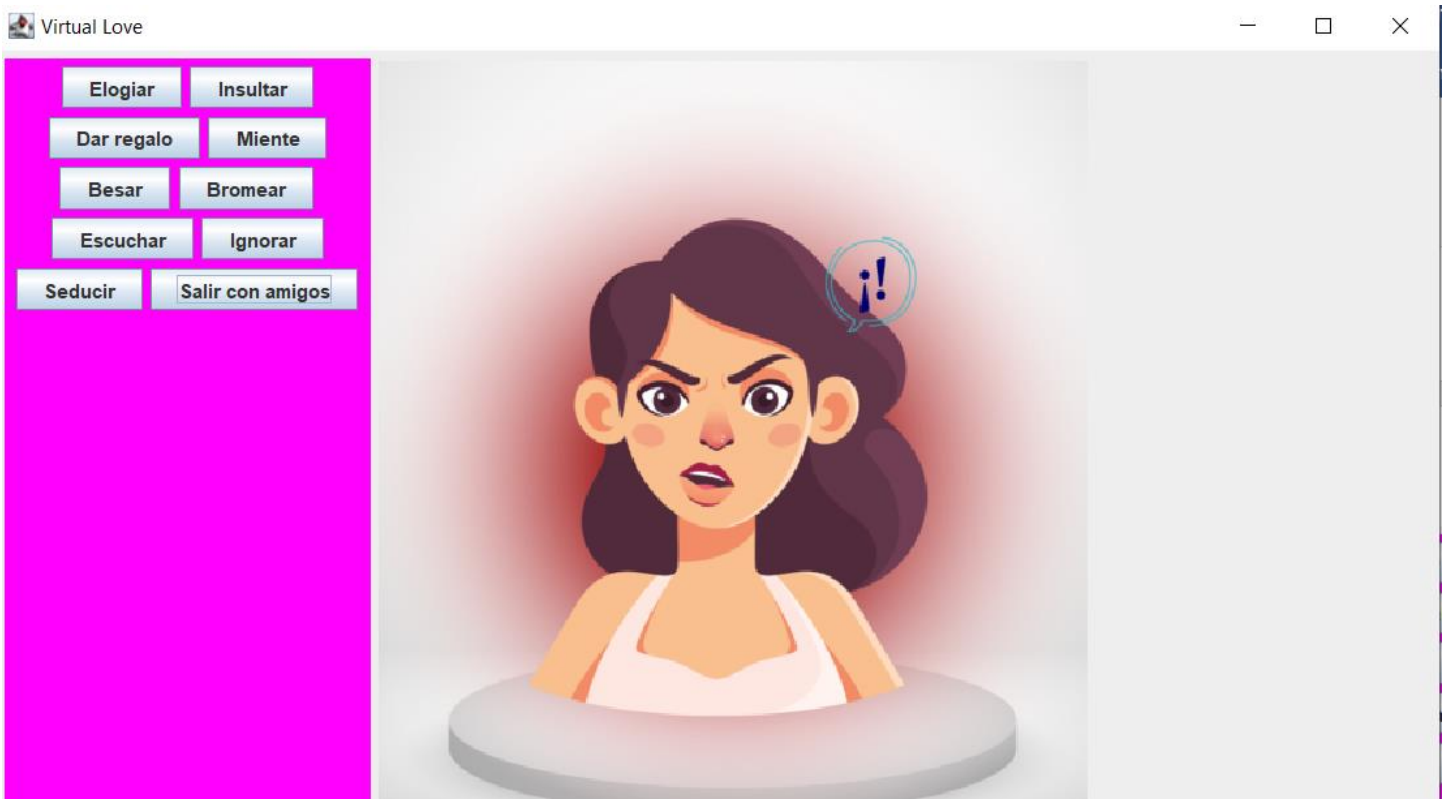

Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos
haha!


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos
Gracias!


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos


Virtual Love
Elogiar
Insultar
Dar regalo
Miente
Besar
Bromear
Escuchar
Ignorar
Seducir
Salir con amigos
¡!

CONCLUSIONES

A lo largo de este reporte hemos hecho énfasis en repetidas ocasiones sobre el poder de las máquinas de estado y como muchos de los programas computacionales más complejos aprovechan este poder. Durante la etapa de programación de Virtual Love decidimos experimentar un poco y averiguar como resultaría nuestro código si no utilizáramos el modelo de tablas de la máquina de estados: el resultado era en comparación mucho más complejo que el resultante. Se requería una mayor cantidad de condiciones que revisar y acciones que ejecutar lo que se traducía en una cantidad mayor de trabajo para nuestro equipo, un código de difícil lectura y poco mantenibilidad. Por el contrario, la implementación de matrices nos otorgó un código que vuelve en ventajas todas las contras antes mencionadas.

Nos parece importante recalcar que Virtual Love cuenta con mucha capacidad de mejora. Al tratarse de nuestra primer experiencia con java no fuimos capaces de exprimir todas las posibilidades y ventajes que un lenguaje tan versátil como java ofrece. La versión final cuenta con algunos problemas relacionados con algunas funciones utilizadas que pertenecen a los kits de desarrollo de la interfaz gráfica y que tienen que ver con el manejo de hilos. Para solucionar este problema acudimos a una solución bastante arcaica y dummy: forzamos una iteración de tiempo considerable para que le diera tiempo de trabajar a la interfaz, esto puede provocar resultados no deseados en algunos equipos de cómputo. Teniendo en cuenta el objetivo del Virtual Love podemos decir que esto es lo único que podría mejorar.

Virtual Love cumplió con ser una simulación minimalista de una relación amorosa al mismo tiempo que se presentó de forma amigable y divertida a los usuarios más allá de los problemas internos y sus soluciones. Como ya mencionamos, el poder de la implementación como máquina de estado permitió a Virtual Love mantener el nivel de complejidad a un costo reducido.

REFERENCIAS

Casasnovas M., (julio 2014) Máquinas de Estado Finito https://www.profesores.frc.utn.edu.ar/electronica/tecnicasdigitalesi/pub/file/AportesDelCudar/Maquinas%20de%20Estado%20MC%20V5.pdf

Hermida, R., del Corral, A. M., Pastor, E., Sánchez, F. (2008) Fundamentos de Computadores. Málaga: SÍNTESIS.