EIE

Escuela de
Ingeniería Eléctrica

UNIVERSIDAD DE COSTA RICA

UNIVERSIDAD DE COSTA RICA

FACULTAD DE INGENIERIA

ESCUELA DE INGENIERIA ELÉCTRICA

ESTRUCTURAS ABSTRACTAS DE DATOS Y ALGORITMOS PARA INGENIERÍA

LABORATORIO 1:
DESARROLLO DEL JUEGO CLUEDO EN C++

ESTUDIANTES:
JESÚS ZÚÑIGA MÉNDEZ
DENNIS CHAVARRÍA SOTO

PROFESOR:
RICARDO ROMÁN BRENES; M. SC.

II CICLO 2019

Índice

1. Introduccion. **2**

2. Funcionamiento del programa. **2**
2.1. Repartición y números pseudoaleatorios 2
2.2. Tablero de juego 2

3. Conclusiones. **3**

4. Bibliografía **3**

5. Anexos **4**
5.0.1. formato.h 4
5.0.2. Includes.h 6
5.0.3. jugador.h 7
5.0.4. tablero.h 8
5.0.5. cartas.h 13
5.0.6. tablero.h 16

Índice de figuras

1. Introduccion.

En este laboratorio, el objetivo fue que los estudiantes aplicaran los conocimientos de memoria dinámica, punteros y, en general, el conocimiento previo de los cursos anteriores de programación, con la finalidad de desarrollar el juego de mesa Clue, también conocido como Cluedo, pero en su versión para computadora. El laboratorio implica que los estudiantes apliquen programación orientada a objetos, así como la implementación de las mecánicas básicas del juego de mesa original.

Para desarrollar ambas versiones, se tomó como punto de partida la idea de utilizar una base de datos que contiene toda la información necesaria, de este modo, el programa podría ser más compacto y eficiente, pues, no tendría que desarrollarse una gran cantidad de condicionales ni analizar múltiples patrones, sino, simplemente, dejar que el proceso de automatización se encargue de analizar la información proporcionada.

2. Funcionamiento del programa.

Este programa requiere la implementación de múltiples objetos, básicamente dos clases, una llamada jugador que incluye los atributos: name, para almacenar el nombre del jugador una vez que se instancia; location, para proporcionar un identificador que indique dónde se encuentra en el tablero del juego; active es una variable que se debe utilizar para indicar si se instació la clase; deck, es un arreglo de string que permite almacenar las cartas que el método Sr Vladimir proporciona de forma aleatoria; Además, contiene el método callme, utilizado para darle el nombre a los jugadores.

2.1. Repartición y números pseudoaleatorios

Para la repartición de cartas se utiliza el método Sr. Vladimir, este debe recibir la longitud del arreglo así como un arreglo del cual tomará cartas. Básicamente esta función toma el arreglo del parámetro, y lo copia en otros dos arreglos para así verificar, de uno si las cartas ya fueron asignadas, y de otro las distribuye. Esto es logrado mediante dos ciclos que iteran en las posiciones de los arreglos ingresados y luego en la posición de la baraja del jugador. Este método es fundamental, pues, sin él, no se podría dar inicio de forma automática al juego.

Otra función fundamental del juego es randgen, que se utiliza en múltiples partes del código para generar un número pseudoaleatorio a partir de la fecha y hora del computador. Es utilizada para tirar dados, y cualquier procedimiento que requiera obtener algún número al azar.

2.2. Tablero de juego

Uno de los requisitos básicos del juego es proporcionar la posibilidad del imprimir las posiciones de los jugadores en el tablero, por lo tanto, se utilizó una serie de funciones y variables especiales para imprimir el tablero con colores, mejorando así la presentación del mismo y la facilidad para diferenciar los diferentes lugares por los cuales los jugadores se movilizan. Con ello, se creó la clase Tablero, que cuenta con el método que permite mover las fichas. Esta clase cuenta con atributos tales como las dimensiones del tablero de juego, esto a partir de una matriz, además, esta se debe crear de forma dinámica, reservando así, un espacio de memoria no fijo. El tablero cuenta con un método que permite imprimir el estado actual del juego, esto gracias a dos ciclos que iteran entre columnas y filas, además de identificar si ya se llegó a la posición límite (El borde) del mismo, con el propósito de imprimir un salto de línea. En el respectivo archivo que contiene todo lo referente a tablero.h, se encuentra el constructor y el destructor del objeto. Además, contiene el método que permite asignar las dimensiones del tablero, llamado .ºbtenerDimensiones"que, abre el archivo que contiene el de referencia, para luego recorrerlo y contar el tamaño; de esta forma, establece las dimensiones de forma dinámica y efectiva

3. Conclusiones.

Después de desarrollar la primera version de este programa se pueden obtener las siguientes conclusiones.

- Fue posible implementar un programa en los lenguajes de programación, Python y C++,para determinar el nombre de los codones constituyentes de una cadena de ARN.
- Se logró, de manera satisfactoria, utilizar los conocimientos fundamentales de los cursos anteriores de programación, para desarrollar los algoritmos solicitados para la práctica de laboratorio.
- Los programas fueron desarrollados de la forma propuesta en grupo, mediante el uso de una base de datos, permitiendo, así, reducir la extensión del código.

4. Bibliografía

Referencias

5. Anexos

5.0.1. formato.h

"formato.h"

```
#ifndef FORMATO_H
#define FORMATO_H

#include <iostream>

using namespace std;
// Combinación   Color
// ========================
// 0;00          Reinicio
#define FORMATO_ANSI_COLOR_RESET "\x1b[0m"
// 0;30          Negro
#define FORMATO_ANSI_COLOR_BLACK "\x1b[30m"
// 0;31          Rojo
#define FORMATO_ANSI_COLOR_RED "\x1b[31m"
// 0;32          Verde
#define FORMATO_ANSI_COLOR_GREEN "\x1b[32m"
// 0;33          Marrón
#define FORMATO_ANSI_COLOR_BROWN "\x1b[33m"
// 0;34          Azul
#define FORMATO_ANSI_COLOR_BLUE "\x1b[34m"
// 0;35          Púrpura
#define FORMATO_ANSI_COLOR_PURPLE "\x1b[35m"
// 0;36          Cian
#define FORMATO_ANSI_COLOR_CYAN "\x1b[36m"
// 0;37          Gris claro
#define FORMATO_ANSI_COLOR_LIGTH_GRAY "\x1b[37m"
// 1;30          Gris oscuro
#define FORMATO_ANSI_COLOR_DARK_GRAY "\x1b[01;30m"
// 1;31          Rojo claro
#define FORMATO_ANSI_COLOR_LIGHT_RED "\x1b[01;31m"
// 1;32          Verde claro
#define FORMATO_ANSI_COLOR_LIGHT_GREEN "\x1b[01;32m"
// 1;33          Amarillo
#define FORMATO_ANSI_COLOR_YELLOW "\x1b[01;33m"
// 1;34          Azul claro
#define FORMATO_ANSI_COLOR_LIGHT_BLUE "\x1b[01;34m"
// 1;35          Púrpura claro
#define FORMATO_ANSI_COLOR_LIGHT_PURPLE "\x1b[01;35m"
// 1;36          Cian claro
#define FORMATO_ANSI_COLOR_LIGHT_CYAN "\x1b[01;36m"
// 1;37          Blanco
#define FORMATO_ANSI_COLOR_WHITE "\x1b[01;37m"

// Colores de fondo (background)
// -----------------------------

// Combinación   Color
// ========================
// 40            Negro
```

```
#define FORMATO_BACKGROUND_COLOR_BLACK "\x1b[40m"
// 41            Rojo
#define FORMATO_BACKGROUND_COLOR_RED "\x1b[41m"
// 42            Verde
#define FORMATO_BACKGROUND_COLOR_GREEN "\x1b[42m"
// 43            Marrón
#define FORMATO_BACKGROUND_COLOR_BROWN "\x1b[43m"
// 44            Azul
#define FORMATO_BACKGROUND_COLOR_BLUE "\x1b[44m"
// 45            Púrpura
#define FORMATO_BACKGROUND_COLOR_PURPLE "\x1b[45m"
// 46            Turquesa
#define FORMATO_BACKGROUND_COLOR_TURQUOISE "\x1b[46m"
// 47            Gris
#define FORMATO_BACKGROUND_COLOR_GRAY "\x1b[40m"
// 100            Gris oscuro
#define FORMATO_BACKGROUND_COLOR_DARK_GRAY "\x1b[100m"
// 101            Rojo
#define FORMATO_BACKGROUND_COLOR_LIGTH_RED "\x1b[101m"
// 102            Verde
#define FORMATO_BACKGROUND_COLOR_LIGTH_GREEN "\x1b[102m"
// 103            Marrón
#define FORMATO_BACKGROUND_COLOR_LIGTH_YELLOW "\x1b[103m"
// 104            Azul
#define FORMATO_BACKGROUND_COLOR_LIGTH_BLUE "\x1b[104m"
// 105            Púrpura
#define FORMATO_BACKGROUND_COLOR_LIGTH_PURPLE "\x1b[105m"
// 106            Turquesa
#define FORMATO_BACKGROUND_COLOR_LIGTH_TURQUOISE "\x1b[106m"
// 107            Gris
#define FORMATO_BACKGROUND_COLOR_WHITE "\x1b[107m"

// Efectos de carácter
// ------------------------------

// Combinación   Estilo
// ======================
// 0;4           Subrayado
#define FORMATO_UNDERLINE_EFFECT "\x1b[00;4m"
// 0;5           Titilante (blink)(puede que no funcione)
#define FORMATO_BLINK_EFFECT "\x1b[00;5m"
// 0;1           Negrita
#define FORMATO_BOLD_EFFECT "\x1b[00;1m"
// 0;8           Transparente (?)
#define FORMATO_TRANSPARENT_EFFECT "\x1b[00;8m"


#endif
```

5.0.2. Includes.h

ïncludes.h"

```
#ifndef INCLUDES_H
#define INCLUDES_H
    #include <iostream>
    #include <chrono>
    #include <random>
    #include <fstream>
    #include "./formato.h"
    #include "./tablero.h"
    #include "./jugador.h"
    #include "./cartas.h"
#endif
```

5.0.3. jugador.h

"jugador.h"

```
#ifndef JUGADOR_H
#define JUGADOR_H

    #include "./includes.h"

    using namespace std;

    class Jugador
    {
        public:
            Jugador(string nom){
                nombre = nom;
                for (int i = 0; i < 9; i++){
                    barajaJugador[i] = "";
                }
                estaVivo = 1;
            }
            Jugador(){
            }

            ~Jugador(){
            }

            string barajaJugador[9]; // es la baraja/mano personal para cada jugador
            string nombre;
            int estaVivo;
        private:
    };
#endif
```

5.0.4. tablero.h

"tablero.h"

```
#ifndef TABLERO_H
#define TABLERO_H
    #include "./includes.h"

    using namespace std;

    class Tablero
    {
        public:
            Tablero(string nombre)
            {
                ruta = "./resources/"+nombre;
                obtenerDimensiones();
                //reservamos memoria para la matriz del tablero
                matriz = new char*[filas];
                for (uint i=0 ; i< filas ; i++){
                    matriz[i] = new char[columnas];
                }
                cargarTablero();
                //cout << "constructor string" << endl;
            };

            ~Tablero(){
                delete matriz;
                //cout << "destructor" << endl;
            };

            /**
             * @brief imprimime una matriz
             */
            void imprimirTablero(char jugador){
                for (uint i =0; i < filas; i++){
                    for (uint j = 0; j < columnas; j++){
                        if (*(*(matriz+i)+j) == 13){
                            cout << endl;
                        }else{
                            formato(*(*(matriz+i)+j) , jugador);
                        }
                    }
                }
                cout << endl << endl;
            }
            /**
             * @brief funcion que mueve jugadores por el tablero
             * @param ficha es el jugador
             * @param direccion es un identificador w arriba, a izquierda, s abajo, d dere
             * @return devuelve 1 se se pudo mover, 0 si no pudo, -1 si entro a una habita
             */
            int moverFicha(char ficha, char direccion){
                uint posicionX = 0;
                uint posicionY = 0;
```

```
52                  int respuesta = 0;
53                  uint moverX = 0;
54                  uint moverY = 0;
55                  for (uint i =0; i < filas; i++){
56                      for (uint j = 0; j < columnas; j++){
57                          if (*(*(matriz+i)+j) == ficha){
58                              posicionY = i;
59                              posicionX = j;
60                          }
61                      }
62                  }
63                  //cout << "la posicion de  " << ficha << "es "<< posicionX << " " << posic
64                  moverY = posicionY;
65                  moverX = posicionX;
66                  if ((direccion == 'w') || (direccion == 'W')){
67                      moverY= posicionY -1;
68                  }
69                  if ((direccion == 's') || (direccion == 'S')){
70                      moverY= posicionY +1;
71                  }
72                  if ((direccion == 'a') || (direccion == 'A')){
73                      moverX= posicionX -1;
74                  }
75                  if ((direccion == 'd') || (direccion == 'D')){
76                      moverX= posicionX +1;
77                  }
78                  if ((moverX == 1) && (moverY == 1)){
79                      //cout << "primer if " << endl;
80                      moverX = columnas - 3;
81                      moverY = filas -2;
82                  }else if ((moverX == columnas - 3) && (moverY == filas -2)){
83                      //cout << "segundo if " << endl;
84                      moverX = 1;
85                      moverY = 1;
86                  }else if ((moverX == 1) && (moverY == filas -2)){
87                      //cout << "tercer if " << endl;
88                      moverX = columnas - 3;
89                      moverY = 1;
90                  }else if ((moverX == columnas - 3) && (moverY == 1)){
91                      //cout << "cuerto if " << endl;
92                      moverX = 1;
93                      moverY = filas-2;
94                  }
95                  //cout << "la voy a mover a  " << moverX << " " << moverY << endl;
96                  for (int i =0; i < tamEspaciosPermitidos; i++){
97                      if (*(*(matriz+moverY)+moverX)== espaciosPermitidos[i]){
98                          *(*(matriz+moverY)+moverX) = ficha;
99
100
101                         if ((espaciosPermitidos [i] != ' ') && (espaciosPermitidos[i] != '
102                             respuesta = -1;
103                         }else{
104                             respuesta = 1;
105                         }
```

```
106
107
108
109                     for (int j=0; j < tamJugadores; j++){
110                         if (caracterPosicionJugador[0][j] == ficha){
111                             *(*(matriz+posicionY)+posicionX) = caracterPosicionJugador
112                             caracterPosicionJugador[1][j] = espaciosPermitidos[i];
113                         }
114                     }
115                 }
116             }
117
118             return respuesta;
119         }
120
121
122     private:
123         uint filas = 0;
124         uint columnas = 0;
125         string ruta = "";
126         char ** matriz = NULL;
127         //se puede cambiar este array por uno dinamico para tableros diferentes
128         int tamEspaciosPermitidos = 11;
129         char espaciosPermitidos[11] = {'A','I','H','G','B','F','D','E','C',' ','/'};
130         int tamJugadores = 6;
131         char caracterPosicionJugador[2][6] = {{'1','2','3','4','5','6'},{' ',' ',' ','
132         /**
133          * @brief abre un archivo
134          */
135         ifstream abrirArchivo(){
136             ifstream archivo;
137             archivo.open(ruta.c_str(),ios::in);
138             if (archivo.fail()){
139                 cout << "No se pudo abrir el archivo "  << ruta << endl;
140             exit(1);
141             }
142             return archivo;
143         }
144         /*
145         *@brief Funion que permite obtener el numero de filas y columnas del tablero
146         */
147         void obtenerDimensiones(){ ;
148             ifstream archivo = abrirArchivo();
149             string linea = "";
150             int contador = 0;
151             while (!archivo.eof()){
152                 getline(archivo,linea);
153                 contador++;
154                 if (linea.size() > columnas){
155                     columnas = linea.size();
156                 }
157             }
158             filas = contador;
159             archivo.close();
```

```
160             }
161             /**
162              * @brief funcion que carga el tablero en el arreglo
163              */
164             void cargarTablero (){//char **matriz, uint f, uint c){
165                 ifstream archivo = abrirArchivo();
166                 string linea = "";
167                 char salto = 00;
168                 //primero llenamos la matriz con algun caracter en este caso caracter nulo
169                 for (uint i =0; i < filas; i++){
170                     for (uint j = 0; j < columnas; j++){
171                         *(*(matriz+i)+j) = salto;
172                     }
173                 }
174                 while (!archivo.eof()){
175                     for (uint i =0; i < filas; i++){
176                         getline(archivo,linea);
177                         for (uint j = 0; j < linea.size(); j++){
178                             *(*(matriz+i)+j) = linea[j];
179 
180                         }
181                     }
182 
183                 }
184                 archivo.clear();
185             }
186 
187             void formato (char caracter, char jugador){
188                 if (caracter == jugador){
189                     cout << FORMATO_BLINK_EFFECT << FORMATO_BACKGROUND_COLOR_GREEN << FORMA
190                     cout << FORMATO_ANSI_COLOR_RESET;
191                     caracter = 00;
192                 }
193                 switch (caracter)
194                     {
195                     case 'H':
196                         //ElCataluña
197                         cout << FORMATO_BACKGROUND_COLOR_BLUE << " ";
198                         break;
199                     case 'I':
200                         //ELCatilinux
201                         cout << FORMATO_BACKGROUND_COLOR_BROWN << " ";
202                         break;
203                     case 'A':
204                         //Panters
205                         cout << FORMATO_BACKGROUND_COLOR_LIGTH_BLUE << " ";
206                         break;
207                     case 'G':
208                         //ElRey
209                         cout << FORMATO_BACKGROUND_COLOR_LIGTH_GREEN << " ";
210                         break;
211                     case 'B':
212                         //Masagex
213                         cout << FORMATO_BACKGROUND_COLOR_PURPLE << " ";
```

```
214                         break;
215                     case 'F':
216                         //Pangea
217                         cout << FORMATO_BACKGROUND_COLOR_LIGTH_PURPLE << " ";
218                         break;
219                     case 'E':
220                         //Pangea
221                         cout << FORMATO_BACKGROUND_COLOR_TURQUOISE << " ";
222                         break;
223                     case 'D':
224                         //Pangea
225                         cout << FORMATO_BACKGROUND_COLOR_BLUE << " ";
226                         break;
227                     case 'C':
228                         //Pangea
229                         cout << FORMATO_BACKGROUND_COLOR_BROWN << " ";
230                         break;
231                     case '/':
232                         //Pangea
233                         cout << FORMATO_BACKGROUND_COLOR_WHITE << " ";
234                         break;
235                     case '#':
236                         //Pangea
237                         cout << FORMATO_ANSI_COLOR_RESET << FORMATO_BACKGROUND_COLOR_BLACK
238                         cout << FORMATO_ANSI_COLOR_RESET;
239                         break;
240                     case ' ':
241                         //Pangea
242                         cout << FORMATO_BACKGROUND_COLOR_WHITE << " ";
243                         break;
244                     default:
245                         cout << FORMATO_ANSI_COLOR_BLACK << FORMATO_BACKGROUND_COLOR_RED
246                         cout << FORMATO_ANSI_COLOR_RESET;
247                         break;
248                 }
249             }
250     };
251
252 #endif
```

5.0.5. cartas.h

"tablero.h"

```
#ifndef CARTAS_H
#define CARTAS_H

    #include "./includes.h"

    using namespace std;

    class Cartas
    {
        public:

            Cartas(int ctdJugadores, Jugador jugadores[]){
                numHabitaciones = 9;
                numArmas = 5;
                numJugadores = ctdJugadores;
                habitaciones = new string[numHabitaciones];
                habitaciones[0] = "El Eden";
                habitaciones[1] = "El CataLinux";
                habitaciones[2] = "El Paraiso";
                habitaciones[3] = "Masajex";
                habitaciones[4] = "El Cataluña";
                habitaciones[5] = "El Rey";
                habitaciones[6] = "La Oficina";
                habitaciones[7] = "Venus";
                habitaciones[8] = "La Casa";
                copiahabitaciones = new string[numHabitaciones];
                copiahabitaciones[0] = "El Eden";
                copiahabitaciones[1] = "El CataLinux";
                copiahabitaciones[2] = "El Paraiso";
                copiahabitaciones[3] = "Masajex";
                copiahabitaciones[4] = "El Cataluña";
                copiahabitaciones[5] = "El Rey";
                copiahabitaciones[6] = "La Oficina";
                copiahabitaciones[7] = "Venus";
                copiahabitaciones[8] = "La Casa";
                armas = new string[numArmas];
                armas[0] = "Ak 47 con silenciador";
                armas[1] = "Bazooka";
                armas[2] = "Espada del Rey Arturo";
                armas[3] = "Conversor Religioso";
                armas[4] = "Mjolnir";
                copiaarmas = new string[numArmas];
                copiaarmas[0] = "Ak 47 con silenciador";
                copiaarmas[1] = "Bazooka";
                copiaarmas[2] = "Espada del Rey Arturo";
                copiaarmas[3] = "Conversor Religioso";
                copiaarmas[4] = "Mjolnir";
                nombresJugadores = new string[ctdJugadores];
                for (int i = 0; i < numJugadores; i++){
                    nombresJugadores[i] = jugadores[i].nombre;
                }
```

```
52              copianombresJugadores = new string[ctdJugadores];
53              for (int i = 0; i < numJugadores; i++){
54                  copianombresJugadores[i] = jugadores[i].nombre;
55              }
56              combinacionGanadora = new string[3];
57              combinacionGanadora[0] = sacarCombinacionGanadora(habitaciones, (numHabita
58              combinacionGanadora[1] = sacarCombinacionGanadora(armas, (numArmas-1));
59              combinacionGanadora[2] = sacarCombinacionGanadora(nombresJugadores, (numJu
60          }
61
62          ~Cartas(){
63              // delete habitaciones;
64              // delete armas;
65              // delete nombresJugadores;
66              // delete combinacionGanadora;
67              // delete copiahabitaciones;
68              // delete copiaarmas;
69              // delete copianombresJugadores;
70          }
71
72
73          /**
74           * @brief Funcion que toma un arreglo y devuelve una casilla del mismo al azar
75           * @param habitaciones es el arreglo que contien las habitaciones
76           * @param tamanio es la dimension del arreglo
77           * @return devuelve el dato seleccionado
78           */
79          string sacarCombinacionGanadora (string* arreglo, int tamanio){
80              string seleccion = "";
81              int aleatorio = numRandom(0,tamanio);
82              seleccion = arreglo[aleatorio];
83              arreglo[aleatorio] = "";
84              return seleccion;
85          }
86
87
88          /*@brief random: Método que recibe dos números y establece un intervalo a parti
89           *@param menor: Límite izquierdo del intervalo lo incluye.
90           *@param mayor: Límite derecho del intervalo lo incluye.
91           *@return devuelve un número aleatorio
92           */
93
94          int numRandom(int menor, int mayor){
95              mayor++;
96              //codigo tomado de https://en.cppreference.com/w/cpp/numeric/random/unifor
97              random_device rd;  //Will be used to obtain a seed for the random number e
98              mt19937 gen(rd()); //Standard mersenne_twister_engine seeded with rd()
99              uniform_real_distribution<> dis(menor, mayor);
100             return(dis(gen));
101         }
102
103         /**
104          * @brief Imprime el arreglo solicitado
105          */
```

```
106             void imprimirArreglos(){
107                 cout << "Habitaciones" << endl;
108                 for (int i=0;i< numHabitaciones;i++){
109                     cout << habitaciones[i] << "  ";
110                 }
111                 cout << endl;
112                 cout << "Armas" << endl;
113                 for (int i=0;i< numArmas;i++){
114                     cout << armas[i] << "  ";
115                 }
116                 cout << endl;
117                 cout << "Jugadores" << endl;
118                 for (int i=0;i< numJugadores;i++){
119                     cout << nombresJugadores[i] << "  ";
120                 }
121                 cout << endl;
122                 cout << "Ganadora" << endl;
123                 for (int i=0;i< 3;i++){
124                     cout << combinacionGanadora[i] << "  ";
125                 }
126                 cout << endl;
127             }
128 
129 
130             int numHabitaciones;
131             int numArmas;
132             int numJugadores;
133             string * habitaciones;
134             string * armas;
135             string * combinacionGanadora;
136             string * nombresJugadores;
137             string * copiahabitaciones;
138             string * copiaarmas;
139             string * copianombresJugadores;
140         private:
141     };
142 #endif
```

5.0.6. tablero.h

"tablero.h"

```
#ifndef CARTAS_H
#define CARTAS_H

    #include "./includes.h"

    using namespace std;

    class Cartas
    {
        public:

            Cartas(int ctdJugadores, Jugador jugadores[]){
                numHabitaciones = 9;
                numArmas = 5;
                numJugadores = ctdJugadores;
                habitaciones = new string[numHabitaciones];
                habitaciones[0] = "El Eden";
                habitaciones[1] = "El CataLinux";
                habitaciones[2] = "El Paraiso";
                habitaciones[3] = "Masajex";
                habitaciones[4] = "El Cataluña";
                habitaciones[5] = "El Rey";
                habitaciones[6] = "La Oficina";
                habitaciones[7] = "Venus";
                habitaciones[8] = "La Casa";
                copiahabitaciones = new string[numHabitaciones];
                copiahabitaciones[0] = "El Eden";
                copiahabitaciones[1] = "El CataLinux";
                copiahabitaciones[2] = "El Paraiso";
                copiahabitaciones[3] = "Masajex";
                copiahabitaciones[4] = "El Cataluña";
                copiahabitaciones[5] = "El Rey";
                copiahabitaciones[6] = "La Oficina";
                copiahabitaciones[7] = "Venus";
                copiahabitaciones[8] = "La Casa";
                armas = new string[numArmas];
                armas[0] = "Ak 47 con silenciador";
                armas[1] = "Bazooka";
                armas[2] = "Espada del Rey Arturo";
                armas[3] = "Conversor Religioso";
                armas[4] = "Mjolnir";
                copiaarmas = new string[numArmas];
                copiaarmas[0] = "Ak 47 con silenciador";
                copiaarmas[1] = "Bazooka";
                copiaarmas[2] = "Espada del Rey Arturo";
                copiaarmas[3] = "Conversor Religioso";
                copiaarmas[4] = "Mjolnir";
                nombresJugadores = new string[ctdJugadores];
                for (int i = 0; i < numJugadores; i++){
                    nombresJugadores[i] = jugadores[i].nombre;
                }
```

```
52              copianombresJugadores = new string[ctdJugadores];
53              for (int i = 0; i < numJugadores; i++){
54                  copianombresJugadores[i] = jugadores[i].nombre;
55              }
56              combinacionGanadora = new string[3];
57              combinacionGanadora[0] = sacarCombinacionGanadora(habitaciones, (numHabita
58              combinacionGanadora[1] = sacarCombinacionGanadora(armas, (numArmas-1));
59              combinacionGanadora[2] = sacarCombinacionGanadora(nombresJugadores, (numJu
60          }
61
62          ~Cartas(){
63              // delete habitaciones;
64              // delete armas;
65              // delete nombresJugadores;
66              // delete combinacionGanadora;
67              // delete copiahabitaciones;
68              // delete copiaarmas;
69              // delete copianombresJugadores;
70          }
71
72
73          /**
74           * @brief Funcion que toma un arreglo y devuelve una casilla del mismo al azar
75           * @param habitaciones es el arreglo que contien las habitaciones
76           * @param tamanio es la dimension del arreglo
77           * @return devuelve el dato seleccionado
78           */
79          string sacarCombinacionGanadora (string* arreglo, int tamanio){
80              string seleccion = "";
81              int aleatorio = numRandom(0,tamanio);
82              seleccion = arreglo[aleatorio];
83              arreglo[aleatorio] = "";
84              return seleccion;
85          }
86
87
88          /*@brief random: Método que recibe dos números y establece un intervalo a parti
89           *@param menor: Límite izquierdo del intervalo lo incluye.
90           *@param mayor: Límite derecho del intervalo lo incluye.
91           *@return devuelve un número aleatorio
92           */
93
94          int numRandom(int menor, int mayor){
95              mayor++;
96              //codigo tomado de https://en.cppreference.com/w/cpp/numeric/random/unifor
97              random_device rd;  //Will be used to obtain a seed for the random number e
98              mt19937 gen(rd()); //Standard mersenne_twister_engine seeded with rd()
99              uniform_real_distribution<> dis(menor, mayor);
100             return(dis(gen));
101         }
102
103         /**
104          * @brief Imprime el arreglo solicitado
105          */
```

```
106             void imprimirArreglos(){
107                 cout << "Habitaciones" << endl;
108                 for (int i=0;i< numHabitaciones;i++){
109                     cout << habitaciones[i] << "  ";
110                 }
111                 cout << endl;
112                 cout << "Armas" << endl;
113                 for (int i=0;i< numArmas;i++){
114                     cout << armas[i] << "  ";
115                 }
116                 cout << endl;
117                 cout << "Jugadores" << endl;
118                 for (int i=0;i< numJugadores;i++){
119                     cout << nombresJugadores[i] << "  ";
120                 }
121                 cout << endl;
122                 cout << "Ganadora" << endl;
123                 for (int i=0;i< 3;i++){
124                     cout << combinacionGanadora[i] << "  ";
125                 }
126                 cout << endl;
127             }
128
129
130             int numHabitaciones;
131             int numArmas;
132             int numJugadores;
133             string * habitaciones;
134             string * armas;
135             string * combinacionGanadora;
136             string * nombresJugadores;
137             string * copiahabitaciones;
138             string * copiaarmas;
139             string * copianombresJugadores;
140         private:
141     };
142 #endif
```