Proyecto Final

Manual Técnico

Johan Daniel Cortes García

20181020068

Cristhian Mauricio Yara Pardo

20181020081

Universidad Distrital Francisco José de Caldas

Facultad de Ingeniería

Ingeniería de Sistemas

Bogotá

2018

Proyecto Final

Manual Técnico

Johan Daniel Cortes García

20181020068

Cristhian Mauricio Yara Pardo

20181020081

Profesor:

José David Álvarez Plata

Universidad Distrital Francisco José de Caldas

Facultad de Ingeniería

Ingeniería de Sistemas

Bogotá

2018

Tabla de contenido

INTRODUCCIÓN

El presente Manual Técnico tiene como fin describir y explicar detalladamente el desarrollo del juego del Gato (Triqui) que corresponde al proyecto final de Programación Básica, el cual se llevó a cabo en el lenguaje de programación c/c++, pseudocódigo y DFD. En donde se implementó lo aprendido en clase a lo largo del semestre utilizando estructuras de control, tipos de datos, bibliotecas, funciones (predefinas o creadas por nosotros) y finalmente comentarios que facilitaran la comprensión del programa.

Cabe mencionar que el presente no pretende ser un curso de aprendizaje de cada una de las herramientas empleadas en la solución del problema, sino documentar y sustentar la aplicación, desarrollo y lógica que realizamos de este. Si se desea obtener un mayor detalle acerca de cada una de las herramientas utilizadas, su forma de operación y aplicación, se recomienda consultar los respectivos manuales de cada una de estas.

Objetivo General

Proporcionar al lector una guía que facilite la comprensión del desarrollo y solución del programa del juego del Triqui.

Objetivos Específicos

- Implementar correctamente las herramientas de programación aprendidas a lo largo del semestre.
- Complementar dichas herramientas que permitieran llevar a cabo la solución del problema.

Desarrollo del problema

Análisis del problema

El juego del Triqui, se juega entre dos personas (en este caso usuario vs pc) y consiste en que alternadamente cada uno de los jugadores hace una jugada en alguna de las casillas disponibles, intentando ser el primero en completar con sus fichas una línea de tres casillas.

El objetivo consiste en construir un programa que implemente en lenguaje C/C++ el juego del Gato de tal forma que uno de los jugadores sea una persona, la que ejecuta el programa, y que el otro jugador sea el computador.

Se dispone de 9 posiciones ordenadas de forma matricial como lo indica la figura:

1	2	3
4	5	6
7	8	9

El juego comienza cuando los jugadores lanzan un dado para determinar quien juega primero. El jugador que obtenga el valor más alto debe comenzar.

Al comienzo del juego las 9 posiciones están vacías. El jugador que inicia el juego selecciona una casilla vacía y la marca. A continuación, el segundo jugador selecciona una casilla vacía y la marca. En seguida juega nuevamente el primer jugador, marcando alguna casilla vacía. Esto continua hasta que alguno de los dos jugadores marca 3 casillas que estén en línea (puede ser diagonal), si ninguno de los jugadores logra este objetivo y no hay más casillas vacías, el juego se declara en empate.

Reglas del juego C/C++

- El juego es usuario vs la computadora.
- Deberás seguir las instrucciones tal y como las plantea el programa.
- Para llenar una casilla deberás oprimir el número de la respectiva casilla, para saber que casilla es cuenta desde arriba a la izquierda [1], hacia la derecha, bajando a la siguiente columna leyendo de izquierda a derecha, hasta llegar a la última posición, la cual es abajo a la derecha [9].
- El símbolo del usuario es 'X' y el de la computadora es 'O'.

Desarrollo del juego en pseudocodigo

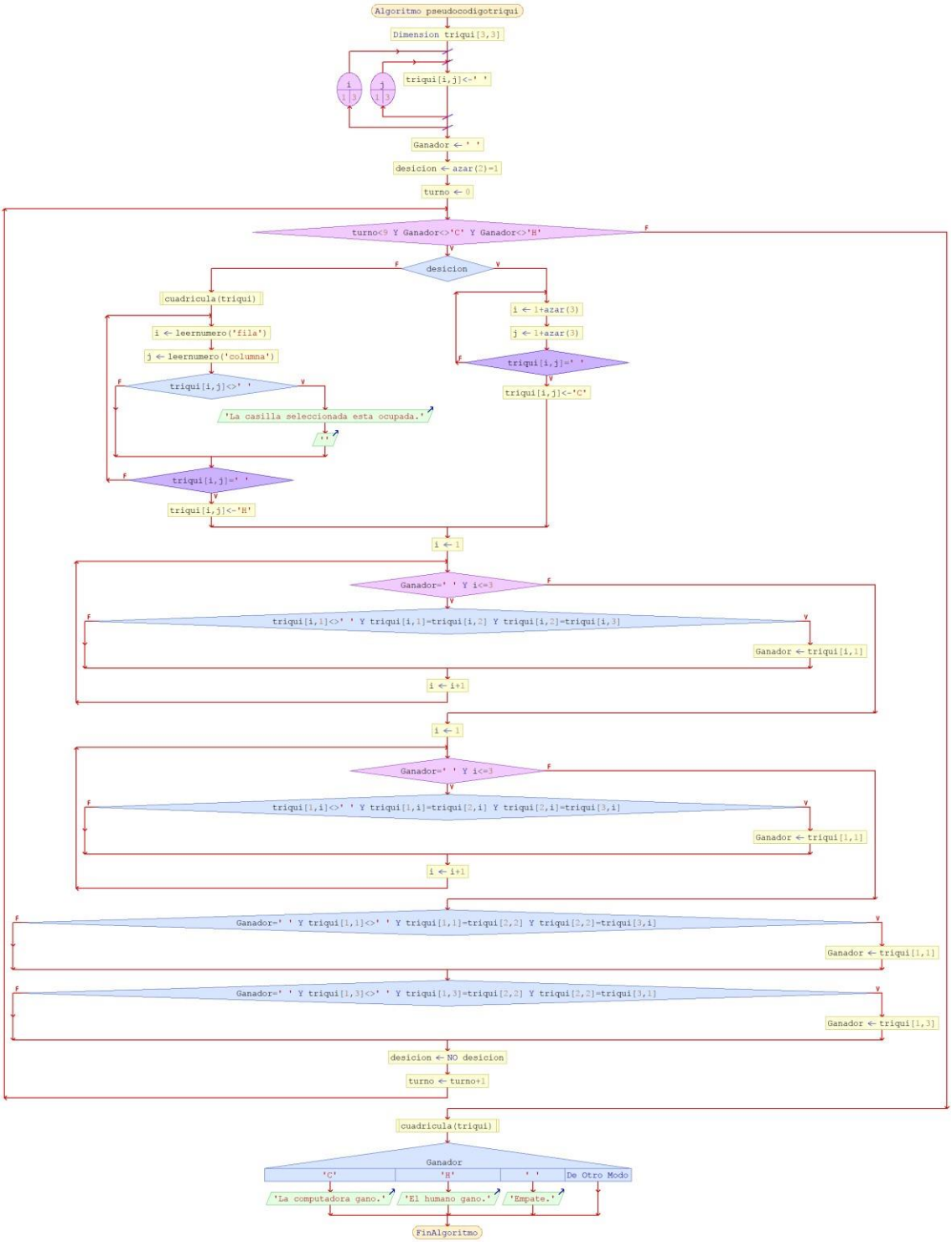
Se tomó este código como base para posteriormente realizar el código en c++.

```
Subproceso cuadricula(triqui)

	Borrar  Pantalla;
	Escribir "    TRIQUI-UD 2018  ";
	Escribir " ";
	Escribir " Ejemplo posiciones: ";
	Escribir " ";
	Escribir "      ----------------";
	Escribir "      | 11 | 12 | 13 |";
	Escribir "      ----------------";
	Escribir "      | 21 | 22 | 23 |";
	Escribir "      ----------------";
	Escribir "      | 31 | 32 | 33 |";
	Escribir "      ----------------";
	Escribir "";
	Escribir "      -------------";
	Para i<- 1 hasta 3 con paso 1 Hacer
		si i<> 1 Entonces
			Escribir "      -------------";
		FinSi
		Escribir  Sin saltar "      |";
		Para j<- 1 Hasta 3 con paso 1 Hacer
			Escribir Sin saltar " ", triqui[i,j]," |";
		FinPara
		Escribir "";
	FinPara
	Escribir "      -------------";
	Escribir "";
	FinSubProceso


Subproceso numero <- leernumero(variable)
	Repetir
		Escribir sin saltar "Seleccion el numero de ", variable, ":";
		leer numero;
		si numero<1 o numero>3 Entonces
			Escribir "Numero no valido";
		FinSi
	Hasta Que numero=1 o numero=2 o numero=3;
FinSubProceso

Proceso  pseudocodigotriqui
	Dimension triqui[3,3];
	para i<- 1 hasta 3 con paso 1 Hacer
		para j<- 1 hasta 3 con paso 1 Hacer
			triqui[i,j] <- ' ';
		FinPara
	FinPara
	Ganador <- ' ';
	desicion <- azar(2)=1
	turno <- 0;
	Mientras turno < 9 y Ganador <>'O' y Ganador <> 'X' Hacer
		si desicion Entonces
			Repetir
				i <- 1+azar(3);
				j <- 1+azar(3);
			Hasta que triqui[i,j] = ' ';
			triqui[i,j] <- 'O';
		Sino
			cuadricula(triqui);
			Repetir
				i <- leernumero("fila");
				j <- leernumero("columna");
```

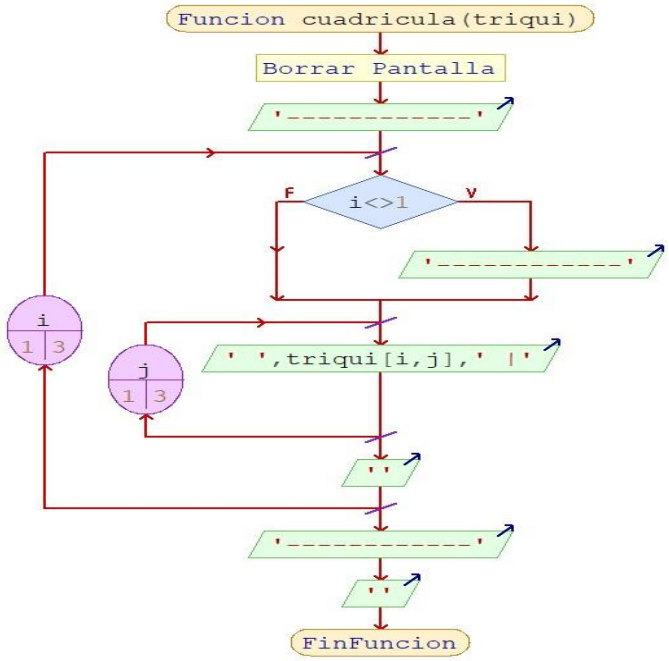
```
                                    si triqui[i,j] <> ' ' Entonces
                                        Escribir "La casilla seleccionada esta ocupada.";
                                        Escribir "";
                                    FinSi
                                Hasta que triqui[i,j] = ' ';
                                triqui[i,j] <- 'X';
                        FinSi
                        i <- 1;
                        mientras Ganador = ' ' Y i <= 3 Hacer
                                si triqui[i,1] <> ' ' y triqui[i,1] = triqui[i,2] y triqui[i,2] = triqui[i,3] Entonces
                                    Ganador <- triqui [i,1]
                                Finsi
                                i <- i + 1;
                        FinMientras
                        i <- 1;
                        mientras Ganador = ' ' Y i <= 3 Hacer
                                si triqui[1,i] <> ' ' y triqui[1,i] = triqui[2,i] y triqui[2,i] = triqui[3,i] Entonces
                                    Ganador <- triqui [1,1];
                                FinSi
                                i <- i + 1;
                        FinMientras
                        Si Ganador = ' ' y triqui[1,1] <> ' ' y triqui[1,1] = triqui[2,2] y triqui[2,2] = triqui[3,3] Entonces
                                Ganador <- triqui[1,1];
                        FinSi
                        si Ganador = ' ' y triqui[1,3] <> ' ' y triqui[1,3] = triqui[2,2] y triqui[2,2] = triqui[3,1] Entonces
                                Ganador <- triqui[1,3];
                        FinSi
                        desicion <- No desicion;
                        turno <- turno + 1;
                FinMientras


                cuadricula(triqui);
                segun Ganador Hacer
                        'O':
                                Escribir "La computadora gano.";
                        'X':
                                Escribir "El Usuario gano.";
                        ' ':
                                Escribir "Empate.";
                FinSegun
FinProceso
```

Diagramas de Flujo

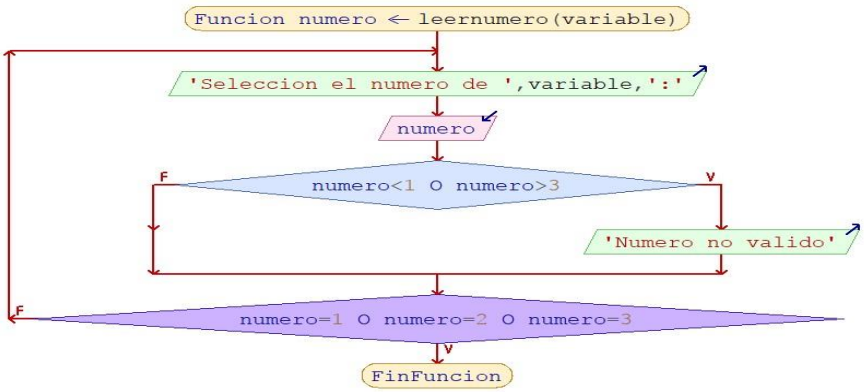
El pseudocodigo tiene como proceso principal del triqui:



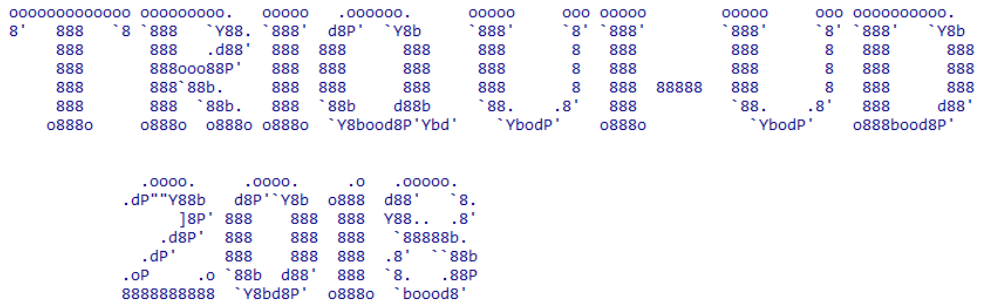
El pseudocodigo tiene dos subprocesos el primero muestra la cuadricula del triqui:



El segundo subproceso se encarga de que el usuario escoja la posición del triqui con el respectivo número de filas y columnas, más adelante se podrá observar cómo se logró optimizar este proceso al realizar el programa en c++ ya que en este último solo será necesario seleccionar el número de la posición del triqui para marcarla.



Desarrollo del juego en c++



Basándonos en el pseudocodigo y en los diagramas de flujo mostrados anteriormente realizamos el siguiente código en c++:

Librerías utilizadas

1. **Iostream,** es un componente de la biblioteca estándar (STL) del lenguaje de programación C++ que es utilizado para operaciones de entrada/salida. Su nombre es un acrónimo de Input/Output Stream. El flujo de entrada y salida de datos en C++ (y su predecesor C) no se encuentra definida dentro de la sintaxis básica y se provee por medio de librerías de funciones especializadas como iostream. iostream define los siguientes objetos:
   - cin : Flujo de entrada
   - cout : Flujo de salida
   - cerr : Flujo de error no almacenado.
   - clog : Flujo de error almacenado.

   Todos los objetos derivados de iostream hacen parte del espacio de nombres std.

2. **stdlib.h,** (std-lib: standard library o biblioteca estándar). Es el archivo de cabecera de la biblioteca estándar de propósito general del lenguaje de programación C. Contiene los prototipos de funciones de C para gestión de memoria dinámica, control de procesos y otras. Es compatible con C++ donde se conoce como cstdlib.

Generación de números pseudo-aleatorios	
`rand`	Genera un número pseudo-aleatorio
`srand`	Establece la semilla para el generador de números pseudo-aleatorios

Control de procesos	
`abort`	terminar ejecución anormalmente
`atexit`	registrar una función callback para la salida del programa
`exit (operating system)`	terminar ejecución del programa
`getenv`	recuperar una variable de entorno
`system (C Standard Library)`	ejecutar un comando externo

3. **time.h,** relacionado con formato de hora y fecha es un archivo de cabecera de la biblioteca estándar del lenguaje de programación C que contiene funciones para manipular y formatear la fecha y hora del sistema.

4. **stdio.h,** que significa "standard input-output header" (cabecera estándar E/S), es el archivo de cabecera que contiene las definiciones de las macros, las constantes, las declaraciones de funciones de la biblioteca estándar del lenguaje de programación C para hacer operaciones, estándar, de entrada y salida, así como la definición de tipos necesarias para dichas operaciones. Por motivos de compatibilidad, el lenguaje de programación C++ (derivado de C) también tiene su propia implementación de estas funciones, que son declaradas con el archivo de cabecera cstdio. Las funciones declaradas en stdio.h son sumamente populares.

5. **String,** es un archivo de la Biblioteca estándar del lenguaje de programación C que contiene la definición de macros, constantes, funciones y tipos y algunas operaciones de manipulación de memoria.

   Las funciones declaradas en string.h se han hecho muy populares, por lo que están garantizadas para cualquier plataforma que soporte C. Sin embargo, existen algunos problemas de seguridad con estas funciones, como el desbordamiento de buffer (buffer overflow), que hacen que algunos programadores prefieran opciones más seguras frente a la portabilidad que estas funciones ofrecen. Además, las funciones para

cadenas de caracteres sólo trabajan con conjuntos de caracteres ASCII o extensiones ASCII compatibles.

6. **vector,** nos permiten añadir elementos de forma ilimitada y quitarlos, a estructura de la clase vector está pensada para operar con arreglos unidimensionales de datos, los elementos de un vector pueden ser manipulados de la misma manera en que se hace con las estructuras de arreglos (arrays) tradicionales en C, C++; es decir, los componentes de un vector pueden ser referenciados a través de un índice numérico, de la misma manera que en un arreglo cualquiera. Por ejemplo, si A es un objeto de vector, entonces la instrucción: A[0]; se refiere al componente 0 (primer elemento) de A. El resultado de todo esto es que usted puede navegar o iterar a través de los componentes de una lista haciendo uso de índices, o si lo prefiere a través de punteros iteradores. Si usted desea ver una lista completa de los métodos asociados a la clase vector siga éste enlace (Tabla de métodos), pero recuerde que no todos ellos serán cubiertos aquí.

Nombre	Descripción	queue	priority_queue
empty	cierto (true) si la cola está vacia	Si	Si
pop	borra el elemento del frente de la cola	Si	Si
push	agrega un elemento al frente de la cola	Si	Si
size	regresa el número de elementos en la cola	Si	Si
front	regresa una referencia al primer elemento en la cola	Si	No
back	regresa una referencia al último elemento en la cola	Si	No
top	regresa una referencia al primer elemento en la cola	No	Si

Funciones

Para comenzar se definieron las siguientes funciones:

- **int cti (string cl)**, *Esta función va a permitir convertir de cadena a int*.
- **void mostrar (string Triqui [3][3])**, *Mostrará en pantalla el juego del triqui actual***.**
- **bool pertenece (vector<int> v,int x),** *Verifica si el numero x ingresado pertenece al vector v***.**
- **void resetear(string Triqui[3][3]),** *Resetea el juego para volver a jugar*
- **int eleccion_computador(string Triqui[3][3],int casillas_ocupadas),** *la computadora analiza el juego y toma una elección.*
- **void marcar (string Triqui[3][3],int posicion,string X)**, *marcar una posicion del triqui con la variable X**.***
- **string ganador(string Triqui[3][3]),** *verifica quien es el ganador en la partida*

Después de definir las funciones que usaremos a lo largo del programa se definen las variables para el lanzamiento del dado, para llenar el triqui y para mostrar en pantalla las reglas del juego:

- string Triqui [3][3];
- int Dadoh=0,Dadoc=0,first=0,second=0;
- string desicion;
- string pdado;
- string Desicionreglas;
- int computador,humano;
- string humano2;
- int contadoreglas=1;
- srand(time(NULL)); Genera números aleatorios

Una vez el usuario ya conoce las reglas del juego los jugadores lanzan un dado para determinar quien juega primero. El jugador que obtenga el valor más alto debe comenzar.

Una vez se determina quien inicia a jugar se comienza el juego.

Una vez terminado el juego se reinicia la partida en donde el humano decide si quiere jugar nuevamente o no.

Conclusiones

- Con el fin de que en el juego el computador fuese inteligente al momento de escoger sus jugadas para ganar o bloquear la siguiente partida del humano, se hallaron todas las posibilidades que tenía el computador para ganar o bloquear en cada fila y columna, lo cual generó que en cierta medida el computador estuviese al nivel del humano.
- Se definieron claramente las reglas del juego dado caso que el usuario las desconociera.
- Con el fin de mejorar la comprensión de las herramientas de programación utilizadas en la solución del problema, entre ellas: estructuras de control, tipos de datos, bibliotecas, funciones, etc. Se agregaron comentarios que facilitaron dicho proceso.
- Se optimizo el programa de pseudocodigo al realizarlo en c++ al momento de escoger la posición del triqui que se deseaba marcar, reduciendo el tiempo entre las partidas de los jugadores.
- Se logró llevar a cabo el propósito principal que consistía en implementar las herramientas de programación aprendidas en clase a lo largo del semestre integrándolas correctamente en la solución del problema.

Bibliografía

[1]ri.ufg.edu.sv/jspui/bitstream/11592/7034/11/338.4791-A473d
[2]ri.ufg.edu.sv/jspui/bitstream/11592/7034/11/338.4791-A473d-Anexo 7.pdfAnexo%207.pdf
[3]www.programarya.com/Cursos/C++/Estructuras-de-Datos
[4]www.programarya.com/Cursos/C++/Funciones
[5]www.cplusplus.com/reference/cstdlib/
[6]es.wikibooks.org/w/index.php?search=cstdlib.H&title=Especial:Buscar&go=Ir&searchToken=7cy3zncvksowt8rmlq66itucj
[7]es.wikibooks.org/w/index.php?search=cstdlib.H&title=Especial%3ABuscar&go=Ir&wprov=acrw1_-1
[8]es.wikibooks.org/wiki/Programaci%C3%B3n_en_C%2B%2B/Librer%C3%ADa_Est%C3%A1ndar_de_Plantillas/Colas?wprov=srpw1_0
[9]es.wikibooks.org/w/index.php?search=cstdlib&title=Especial:Buscar&go=Ir
[10]es.wikibooks.org/wiki/Programaci%C3%B3n_en_C%2B%2B/Biblioteca_Est%C3%A1ndar_de_Plantillas/Vectores
[11]www.cplusplus.com/reference/vector/vector/
[12]es.wikipedia.org/wiki/Stdlib.h