|  |  |  |
| --- | --- | --- |
| **CARRERA:**  Ingeniería en Software | **GUÍA**  No. 01 | **TIEMPO ESTIMADO:** |
| **ASIGNATURA:**  Estructura de datos  NRC: 2967 | **FECHA DE ELABORACION:** 27-01-2020  **SEMESTRE**: septiembre 2019 – enero 2020 | |
| **TÍTULO:**  Proyecto Primer Parcial (Sudoku) | **DOCENTE:** Ing. Fernando Solís | |

**OBJETIVO**

Aplicar el conocimiento de la recursividad y back tracking en el desarrollo de un sudoku 9\*9.

**INSTRUCCIONES**

1. Utilice como material principal, las principales herramientas para desarrollo en C++
2. Utilice información consultada en Internet y conocimiento adquirido en clase.

**ACTIVIDADES**

1. **Ubicación de recursos**
2. Formar grupos de máximo 2 personas por computador
3. Realizar el programa en cualquier IDE para C++
4. **Planteamiento del problema**

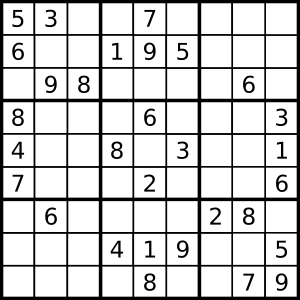
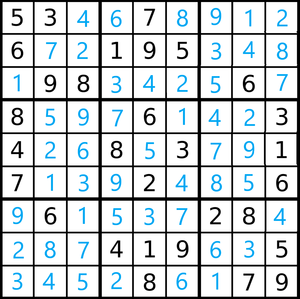
**p.e. Ejercicio No 1**

Desarrollar un programa que aplique los conocimientos aprendidos en el primer parcial, principalmente “back tracking” y recursividad. Utilizar memoria dinámica

1. **Marco Teórico**

El Sudoku es un juego matemático que se inventó a finales de la década de 1970, adquirió popularidad en Japón en la década de 1984 y se dio a conocer en el ámbito internacional en 2005 cuando numerosos periódicos empezaron a publicarlo en su sección de pasatiempos.1​El objetivo del sudoku es rellenar una cuadrícula de 9 × 9 celdas (81 casillas) dividida en

subcuadrículas de 3 × 3 (también llamadas "cajas" o "regiones") con las cifras del 1 al 9 partiendo de algunos números ya dispuestos en algunas de las celdas.



**¿Como resolverlo con back tracking?**

Una llamada inicial: ‘‘‘simplemente se encarga de llamar a la función recursiva. Para darle más utilidad retornará un booleano que indicará si se ha encontrado alguna solución al sudoku.

Una función recursiva: ‘‘‘se encarga del proceso de ir probando números. Incluye un parámetro de entrada/salida que indica si se ha encontrado alguna solución.

El comprobador de validez: ‘‘‘simplemente es una función que retorna un booleano indicando si un valor puede estar en un sitio indicado.

1. **Código del proyecto.**

/\*\*

\***@file** Tabla.cpp

\***@version** 2.0

\***@author** Antoni Toapanta

\*@title Funciones de la Clase Tabla

\***@brief** Implementacion de los metos de la Clase Tabla

\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Module: Tabla.cpp

\* Author: RODRIGO

\* Modified: mi鲣oles, 6 de noviembre de 2019 02:01:52

\* Purpose: Implementation of the class Tabla

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "Tabla.h"

#include "ManejoMemoria.h"

/\*\*

**@brief** Busaca la posisicon donde puede hacer el cambio

**@param** Fila,Columna

**@return** Verdadero si encuentra un 0 donde puede hacer el cambio si no un falso

\*/

bool Tabla**::**buscarLugar**(**int**&** fila**,** int**&** col**)**

**{**

**for** **(**fila **=** 0**;** fila **<** 9**;** fila**++)**

**for** **(**col **=** 0**;** col **<** 9**;** col**++)**

**if** **(\*(\*(**tablero**+**fila**)+**col**)** **==** 0**)**

**return** **true;**

**return** **false;**

**}**

/\*\*

**@brief** Comprueba si cumple las condiciones de Sudoku

**@param** Fila,Columna,num

**@return** Verdadero si cumple si no un falso

\*/

bool Tabla**::**comprobar**(** int fila**,** int col**,** int num**)**

**{**

**return** **!**UtilizadoEnFila**(**fila**,** num**)** **&&!**UtilizadoEnColumna**(**col**,** num**)** **&&!**UtilizadoEnSudoku**(**fila **-** fila **%** 3 **,**col **-** col **%** 3**,** num**)** **&&\*(\*(**tablero**+**fila**)+**col**)==** 0**;**

**}**

/\*\*

**@brief** Muestra el Sudoku y guarda informacion en el .csv

\*/

void Tabla**::**mostrarT**()**

**{**

mostrar**(**tablero**,**9**,**9**);**

**}**

/\*\*

**@brief** Obtiene tablero

\*/

int**\*\*** Tabla**::**getTablero**(**void**)**

**{**

**return** tablero**;**

**}**

/\*\*

**@brief** Camcio de tablero

\*/

void Tabla**::**setTablero**(**int**\*\*** newTablero**)**

**{**

tablero **=** newTablero**;**

**}**

/\*\*

**@brief** Constructor de la Clase

\*/

Tabla**::**Tabla**()**

**{**

**this->**tablero**=**reservar**(**9**,**9**);**

encerar**(this->**tablero**,**9**,**9**);**

**}**

/\*\*

**@brief** Destructor de la clase

\*/

Tabla**::~**Tabla**()**

**{**

liberarMemoria**(this->**tablero**,**9**);**

**}**

bool Tabla**::**Sudoku**()**

**{**

int fila**,** col**;**

**if** **(!**buscarLugar**(**fila**,** col**))**

**return** **true;**

**for** **(**int num **=** 1**;** num **<=** 9**;** num**++)**

**{**

**if** **(**comprobar**(**fila**,** col**,** num**))**

**{**

**\*(\*(**tablero**+**fila**)+**col**)** **=** num**;**

escribirArchivo**(**tablero**,**9**);**

cout**<<**endl**;**

**if** **(**Sudoku**())**

**return** **true;**

**\*(\*(**tablero**+**fila**)+**col**)=** 0**;**

**}**

**}**

**return** **false;**

}

bool Tabla::UtilizadoEnFila(int fila, int num)

{

for (int col = 0; col < 9; col++)

if (\*(\*(tablero+fila)+col)== num)

return true;

return false;

}

bool Tabla::UtilizadoEnColumna(int col, int num)

{

for (int fila = 0; fila < 9; fila++)

if (\*(\*(tablero+fila)+col) == num)

return true;

return false;

}

bool Tabla::UtilizadoEnSudoku(int inicioFila, int inicioColumna, int num)

{

for (int fila = 0; fila< 3; fila++)

for (int col = 0; col < 3; col++)

if (\*(\*(tablero+(fila + inicioFila))+(col + inicioColumna)) == num)

return true;

return false;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Module: Tabla.h

\* Author: RODRIGO

\* Modified: mi鲣oles, 6 de noviembre de 2019 02:01:52

\* Purpose: Declaration of the class Tabla

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#if !defined(\_\_Class\_Diagram\_1\_Tabla\_h)

#define \_\_Class\_Diagram\_1\_Tabla\_h

#include "Operaciones.h"

class Tabla **:** public Operaciones

**{**

public**:**

bool buscarLugar**(**int**&** fila**,** int**&** col**);**

bool comprobar**(**int fila**,** int col**,** int num**);**

void mostrarT**();**

int**\*\*** getTablero**(**void**);**

void setTablero**(**int**\*\*** newTablero**);**

Tabla**();**

**~**Tabla**();**

bool Sudoku**();**

bool UtilizadoEnFila**(**int fila**,** int num**);**

bool UtilizadoEnColumna**(**int col**,** int num**);**

bool UtilizadoEnSudoku**(**int inicioFila**,** int inicioColumna**,** int num**);**

protected**:**

private**:**

int**\*\*** tablero**;**

**};**

#endif

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Module: Operaciones.h

\* Author: RODRIGO

\* Modified: mi鲣oles, 6 de noviembre de 2019 01:54:07

\* Purpose: Declaration of the class Operaciones

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#if !defined(\_\_Class\_Diagram\_1\_Operaciones\_h)

#define \_\_Class\_Diagram\_1\_Operaciones\_h

class Operaciones

**{**

public**:**

/\*\*

**@brief** Comienza a resolver el sudoku usasando Backtraking

**@param** Nada

**@return** Un Verdadero si se puede resolver o un falso si es imposible

\*/

virtual bool Sudoku**()=**0**;**

/\*\*

**@brief** Comprueba si se encuentra el numero en la fila

**@param** fila, numero

**@return** Verdadero sis e encuentra o falso si no

\*/

virtual bool UtilizadoEnFila**(**int fila**,** int num**)=**0**;**

/\*\*

**@brief** Comprueba si se encuentra el numero en la columna

**@param** fila, numero

**@return** Verdadero si se encuentra o falso si no

\*/

virtual bool UtilizadoEnColumna**(**int col**,** int num**)=**0**;**

/\*\*

**@brief** Comprueba si se encuentra el numero en la tanto en fila como en columna

**@param** fila, numero

**@return** Verdadero si se encuentra o falso si no

\*/

virtual bool UtilizadoEnSudoku**(**int inicioFila**,** int inicioColumna**,** int num**)=**0**;**

protected**:**

private**:**

**};**

#endif

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Module: Manejo de memoria.h

\* Author: RODRIGO

\* Modified: mi鲣oles, 20 de octubre de 2019 01:54:07

\* Purpose: Declaration of the class Operaciones

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef MANEJOMEMORIA\_H\_INCLUDED

#define MANEJOMEMORIA\_H\_INCLUDED

#include<iostream>

#include <conio.h>

#include<time.h>

#include <stdio.h>

#include<fstream>

**using** **namespace** std**;**

/\*\*

**@brief** Reservar memoria

**@param** Fila,Columna

**@return** Una reserva de memoria dinamica entera

\*/

int **\*\***reservar**(**int fila**,**int colum**)**

**{**

int **\*\***m**,**j**;**

m**=new** int**\*[**fila**];**//reservando memoria de filas

**for(**int i**=**0**;**i**<**fila**;**i**++)**

**{**

m**[**i**]=new** int**[**colum**];**//reservar columnas

**}**

**return** m**;**

**}**

/\*\*

**@brief** Agregar a mi arreglo dinamico

**@param** \*\*matriz,tamanio

**@return** Nada

\*/

void agregar**(**int **\*\***matriz**,**int tam**)**

**{**

**for(**int i**=**0**;**i**<**tam**;**i**++)**

**{**

**for(**int j**=**0**;**j**<**tam**;**j**++)**

**{**

cout**<<**i**+**1**<<**" "**<<**j**+**1**<<**endl**;**

cin**>>\*(\*(**matriz**+**i**)+**j**);**

**}**

**}**

**}**

/\*\*

**@brief** Muestrai arreglo dinamico

**@param** \*\*matriz,fila,columna

**@return** Nada

\*/

void mostrar**(**int**\*\***matriz**,**int fila**,**int columna**)**

**{**

**for(**int i**=**0**;**i**<**fila**;**i**++)**

**{**

**for(**int j**=**0**;**j**<**columna**;**j**++)**

**{**

**if(\*(\*(**matriz**+**i**)+**j**)!=**0**)**

**{**

cout**<<\*(\*(**matriz**+**i**)+**j**)<<**"\t"**;**

**}**

**else{**

cout**<<**" "**<<**"\t"**;**

**}**

**}**

cout**<<**endl**;**

cout**<<**endl**;**

**}**

**}**

/\*\*

**@brief** Libera memoria

**@param** \*\*matriz,columna

**@return** Nada

\*/

void liberarMemoria**(**int **\*\***matriz**,**int columna**)**

**{**

**for(**int i**=**0**;**i**<**columna**;**i**++)**

**{**

**delete[]** matriz**[**i**];**

**}**

**delete** **[]**matriz**;**

**}**

/\*\*

**@brief** Llena de 0 mi matriz

**@param** \*\*matriz,columna

**@return** Nada

\*/

void encerar**(**int **\*\*** m**,**int fila**,** int columna**)**

**{**

**for(**int i**=**0**;**i**<**fila**;**i**++)**

**{**

**for(**int j**=**0**;**j**<**columna**;**j**++)**

**{**

**\*(\*(**m**+**i**)+**j**)=**0**;**

**}**

**}**

**}**

/\*\*

**@brief** Llena de aleatorioa una arreglo Unidimencional

**@param** \*\*matriz,columna

**@return** Nada

\*/

void llenarAleatorioUni**(**int **\***m**,**int tam**)**

**{**

int aux**;**

//srand(time(NULL));

**for(**int i**=**0**;**i**<**tam**;**i**++)**

**{**

aux**=**1**+**rand**()%**tam**;**

int aux2 **=** 0**;**

**while(**aux2 **<** i**)**

**{**

**if(**aux **!=** **\*(**m**+**aux2**))**

**{**

aux2**++;**

**}**

**else**

**{**

aux **=** **(**1 **+** rand**()** **%** tam**);**

aux2 **=** 0**;**

**}**

**}**

**\*(**m**+**i**)** **=** aux**;**

**}**

**}**

/\*\*

**@brief** Escribe en un archivo csv

**@param** \*\*matriz,columna

**@return** Nada

\*/

void escribirArchivo**(**int **\*\***guardar**,**int tam**)**

**{**

ofstream archivo**;**

archivo**.**open**(**"Proyecto.csv"**,**ios**::**out **|** ios**::**app**);**

**if(**archivo**.**fail**())**

**{**

cout**<<**"No se pudo Abrir"**<<**endl**;**

exit**(**1**);**

**}**

**else{**

**for(**int i**=**0**;**i**<**tam**;**i**++)**

**{**

**for(**int j**=**0**;**j**<**tam**;**j**++)**

**{**

archivo**<<\*(\*(**guardar**+**i**)+**j**)<<**" \t"**;**

**}**

archivo**<<**" \n"**;**

archivo**<<**" \n"**;**

**}**

**}**

archivo**<<**" \n"**;**

archivo**<<**" \n"**;**

archivo**<<**" \n"**;**

archivo**<<**" \n"**;**

archivo**.**close**();**

**}**

#endif // MANEJOMEMORIA\_H\_INCLUDED

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Module: ingreso.h

\* Author: RODRIGO & JHONY

\* Modified: miércoles, 6 de noviembre de 2019 01:54:07

\* Purpose: Validar ingreso de enteros

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <string>

#include <stdlib.h>

**using** **namespace** std**;**

class Ingreso**{**

public**:**

float ingresarFloat**(**char**\*);**

bool validarFloat**(**string**);**

int ingresarInt**(**char**\*);**

bool validarInt**(**string**);**

string ingresarString**(**char**\*);**

bool validarString**(**string**);**

string ingresar10Digitos**(**char**\*);**

bool validar10Digitos**(**string**);**

**};**

float Ingreso**::**ingresarFloat**(**char**\*** msg**){**

string flotante**;**

bool flag**;**

**do{**

**try{**

cout**<<**msg**;**

getline**(**cin**,**flotante**);**

flag **=** validarFloat**(**flotante**);**

**if(**flag**){**

**throw** flotante**;**

**}**

**}catch(**string e**){**

cout<<"Dato invalido "<<e<<endl;

}

}while(flag);

return atof(flotante.c\_str());

}

bool Ingreso::validarFloat(string valor){

bool flag;

for(int i = 0; i<valor.length(); i++){

if(isdigit(valor[i])){

flag = false;

}else{

if(valor[i]== '.'){

flag = false;

}else{

flag = true;

break;

}

}

}

return flag;

}

int Ingreso::ingresarInt(char\* msg){

string numero;

bool flag;

do{

try{

cout<<msg;

getline(cin,numero);

flag = validarInt(numero);

if(flag){

throw numero;

}

}catch(string e){

cout<<"Dato invalido "<<e<<endl;

}

}while(flag);

return atoi(numero.c\_str());

}

bool Ingreso::validarInt(string valor){

bool flag;

for(int i = 0; i<valor.length(); i++){

if(isdigit(valor[i])){

flag = false;

}else{

return true;

}

}

return flag;

}

string Ingreso::ingresarString(char\* msg){

string dato;

bool flag;

do{

try{

cout<<msg;

getline(cin,dato);

flag = validarString(dato);

if(flag){

throw dato;

}

}catch(string e){

cout<<"Dato invalido "<<e<<endl;

}

}while(flag);

return dato;

}

bool Ingreso::validarString(string valor){

bool flag;

for(int i = 0; i<valor.length(); i++){

if(isalpha(valor[i]) || valor[i] == ' '){

flag = false;

}else{

return true;

}

}

return flag;

}

string Ingreso::ingresar10Digitos(char\* msg){

string dato;

bool flag;

do{

try{

cout<<msg;

getline(cin,dato);

flag = validar10Digitos(dato);

if(flag){

throw dato;

}

}catch(string e){

cout<<"Dato invalido "<<e<<endl;

}

}while(flag);

return dato;

}

bool Ingreso::validar10Digitos(string valor){

bool flag;

int cont = 0;

for(int i = 0; i<valor.length(); i++){

if(isdigit(valor[i])){

cont++;

if(cont == 10){

flag = false;

}else{

flag = true;

}

}else{

return true;

}

}

return flag;

}

/\*\*

\***@file** Main

\***@version** 2.0

\***@author** Antoni Toapanta, Jhonny Naranjo

\*@title Main

\***@brief** Mi programa se ejecuta aqui

\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Module: Main.cpp

\* Author: Antoni Toapanta Jhonny Naranjo

\* Modified: mi鲣oles, 6 de noviembre de 2019 01:54:07

\* Purpose: Declaration of the class Operaciones

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include<stdlib.h>

#include "Tabla.cpp"

#include "ingreso.h"

#include <pthread.h>

#include <windows.h>

**using** **namespace** std**;**

void **\***moverMarquesina**(**void **\***marq**);**

void gotoxy**(**short x**,** short y**);**

int main**()**

**{**

pthread\_t thread1**;**

pthread\_t thread2**;**

pthread\_create**(&**thread1**,NULL,**moverMarquesina**,NULL);**

main**:**

cout**<<**endl**;**

cout**<<**endl**;**

Ingreso lee**;**

int opc**;**

cout**<<**"============================================================="**<<**endl**;**

cout**<<**"\tToapanta Antoni-Naranjo Jhony"**<<**endl**;**

cout**<<**"\tEstructura de datos\n\t2019-2020"**<<**endl**;**

cout**<<**"==============================================================\n"**;**

do{

cout<<"Escoja la opcion que desea realizar"<<endl;

cout<<"1. Mostrar solucion sudoku\n2.Mostrar ayuda del programa";

opc = lee.ingresarInt("\n---> ");

}while(opc != 1 && opc != 2 );

if(opc==2)

system("ayuda.chm");

if(opc==1){

Tabla t = Tabla();

if (t.Sudoku() == true)

{

system("cls");

cout<<endl;

cout<<endl;

cout<<endl;

cout<<endl;

cout<<endl;

cout<<endl;

cout<<endl;

t.mostrarT();

cout<<endl;

cout<<endl;

cout<<endl;

cout<<endl;

system("txt2pdf.exe Proyecto.csv SudokuBacktraking.pdf -oao -pfs60 -pps43 -ptc0 -width3000 -height2000");

system("pause");

system("cls");

goto main;

}

else

{

cout << "No solution exists";

}

t.~Tabla();

system("pause");

}

return 0;

}

void gotoxy(short x, short y) {

COORD pos = {x, y};

SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), pos);

}

void\* moverMarquesina(void \*marq){

int a, b, c, n=0,letra=39,pos=1,cont=39,aux,cont1=39,auxg;

char t[50] ="UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE",auxt[39]=" ";

do{

for (a=0;a<42;a++){

aux=pos;

for(b=39;b>cont;b--){

gotoxy(pos,1);

cout<<t[b];

pos--;

}

aux++;

pos=aux;

cont--;

Sleep (75);

if(a==40){

break;

}

}

for(a=3;a<70;a++){

gotoxy(a-1,1);

cout<<" ";

gotoxy(a,1);

cout<<t;

Sleep (75);

}

pos=70;

auxg=69;

for (a=0;a<41;a++){

gotoxy(auxg,1);

cout<<" ";

aux=pos;

for(b=0;b<=cont1;b++){

gotoxy(pos,1);

cout<<t[b];

pos++;

}

cont1--;

aux++;

pos=aux;

auxg++;

Sleep (75);

}

cont1=39;

letra=39;

pos=1;

cont=39;

c=0;

}while (c=1);

}

1. **Conclusiones**

La recursividad hace que el resolver estos problemas sean un poco más cortos, pero con más grado de complejidad a la hora de entenderlos como funcionan.