**Sudoku** **Solver**

Contents

[Limbajul de programare 3](#_Toc99085594)

[Ce este sudoku? 3](#_Toc99085595)

[Bibloteca principala 5](#_Toc99085596)

[Fereastra principala 6](#_Toc99085597)

[Introducerea numerelor 7](#_Toc99085598)

[Panoul de introducere a numerelor dupa apasarea butonului de start: 9](#_Toc99085599)

[Panoul de introducere a numerelor dupa introducere: 9](#_Toc99085600)

[Submeniul 10](#_Toc99085601)

[Programul de procesare a mesajelor: 10](#_Toc99085602)

[Programul de executare: 11](#_Toc99085603)

# Limbajul de programare

C++ este un limbaj de programare general, compilat. Este un limbaj multi-paradigmă, cu verificarea statică a tipului variabilelor ce suportă programare procedurală, abstractizare a datelor, programare orientată pe obiecte. În anii 1990, C++ a devenit unul dintre cele mai populare limbaje de programare comerciale, rămânând astfel până azi.

Bjarne Stroustrup de la Bell Labs a dezvoltat C++ (inițial denumit C cu clase) în anii 1980, ca o serie de îmbunătățiri ale limbajului C. Acestea au început cu adăugarea noțiunii de clase, apoi de funcții virtuale, suprascrierea operatorilor, moștenire multiplă (engleză multiple inheritance), șabloane (engleză template) și excepții. Limbajul de programare C++ a fost standardizat în 1998 ca și ISO 14882:1998, versiunea curentă fiind din 2003, ISO 14882:2003.

Am conceput un program care rezolva diferite grile de Sudoku, folosinf libraria winapi pe care se bazeaza intregul sistem de operare Windows, astfel rezultand intr-un program pe stilul Microsoft cu care ne-am obsinuit cu totii.

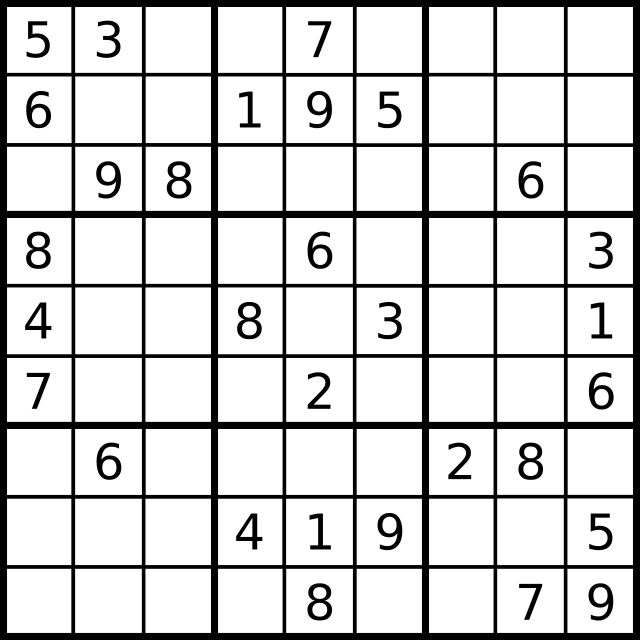
# Ce este sudoku?

Sudoku (din japoneză 数, sû - cifră și 独, doku - unică), este un joc în formă de grilă inventat în 1979 și inspirat de pătratul latin și de problema celor 36 ofițeri a lui Leonhard Euler. Scopul jocului este de a umple această grilă cu cifrele de la 1 la 9 respectând anumite condiții, cu unele cifre fiind de la început dispuse în grilă.

Grila jocului este un pătrat de nouă pe nouă căsuțe, subdivizat în tot atâtea pătrate identice, numite regiuni. Regula jocului este simplă: fiecare rând, coloană sau regiune nu trebuie să conțină decât o dată cifrele de la unu la nouă. Formulat altfel, fiecare ansamblu trebuie să conțină cifrele de la unu la nouă o singură dată.

Cifrele nu reprezintă decât o convenție, relațiile aritmetice între ele nefiind de nici un folos. Orice ansamblu de simboluri distincte: litere, forme, culori, pot fi folosite fără a se modifica regulile jocului. Dell Magazine, primul care a publicat grile, a folosit cifre în publicațiile sale. Dimpotrivă, Scramblets, de la Penny Press, și Sudoku Word, de Knight Features Syndicate, folosesc amândouă litere.

Interesul jocului consistă în simplitatea regulilor sale și în complexitatea soluțiilor sale. Grilele publicate au de obicei un nivel de dificultate indicat, iar editorul are posibilitatea să indice și un timp de rezovare probabil. Cu toate că, în general, grilele ce conțin mai multe cifre completate sunt mai ușoare, inversul nu este în totdeauna adevărat. Dificultatea veritabilă a jocului rămâne totuși în a găsi suita exactă a cifrelor rămase.

Acest joc a inspirat deja mai multe versiuni electronice care aduc un interes diferit rezolvării grilelor Sudoku. Forma sa de tip grilă și folosirea lui într-un scop ludic îl aduc mai aproape de alte jocuri publicate în ziare, cum ar fi [careurile](https://ro.wikipedia.org/w/index.php?title=Careu&action=edit&redlink=1" \o "Careu — pagină inexistentă) și problemele de [șah](https://ro.wikipedia.org/wiki/%C8%98ah).

[Profesorii](https://ro.wikipedia.org/wiki/Profesor) recomandă practicarea jocului Sudoku ca antrenament pentru [gândirea logică](https://ro.wikipedia.org/w/index.php?title=G%C3%A2ndire_logic%C4%83&action=edit&redlink=1" \o "Gândire logică — pagină inexistentă). Nivelul de dificultate poate în acest caz să fie adaptat publicului. Grilele sunt publicate în ziare, dar pot fi și generate cu ajutorul unui [computer](https://ro.wikipedia.org/wiki/Computer).

# Bibloteca principala

Windows API este o [interfață destinată programării aplicațiilor](https://ro.wikipedia.org/wiki/Interfa%C8%9B%C4%83_de_programe_de_aplica%C8%9Bii" \o "Interfață de programe de aplicații) pentru [sistemul de operare](https://ro.wikipedia.org/wiki/Sistem_de_operare) [Microsoft Windows](https://ro.wikipedia.org/wiki/Microsoft_Windows) (API este acronimul din [limba engleză](https://ro.wikipedia.org/wiki/Limba_englez%C4%83" \o "Limba engleză) pentru Application Programming Interface). Windows API este cunoscută, în general, cu numele de Win32 API, însă denumirea Windows API reflectă mai precis capacitățile și utilitatea sa, suportul atât pentru Windows 32 biți, cât și pentru Windows 64 biți.

Microsoft Windows [SDK](https://ro.wikipedia.org/wiki/SDK" \o "SDK) ([en.](https://ro.wikipedia.org/wiki/Limba_englez%C4%83) Software Development Kit) conține documentația și unelte necesare programatorilor pentru a realiza aplicații folosind Windows API.

Prin Win32 API programatorul are acces direct la o mare parte a funcțiilor de nivel de bază (en. low-level) ale [sistemului de operare](https://ro.wikipedia.org/wiki/Sistem_de_operare" \o "Sistem de operare), putând crea aplicații într-un mod foarte flexibil.

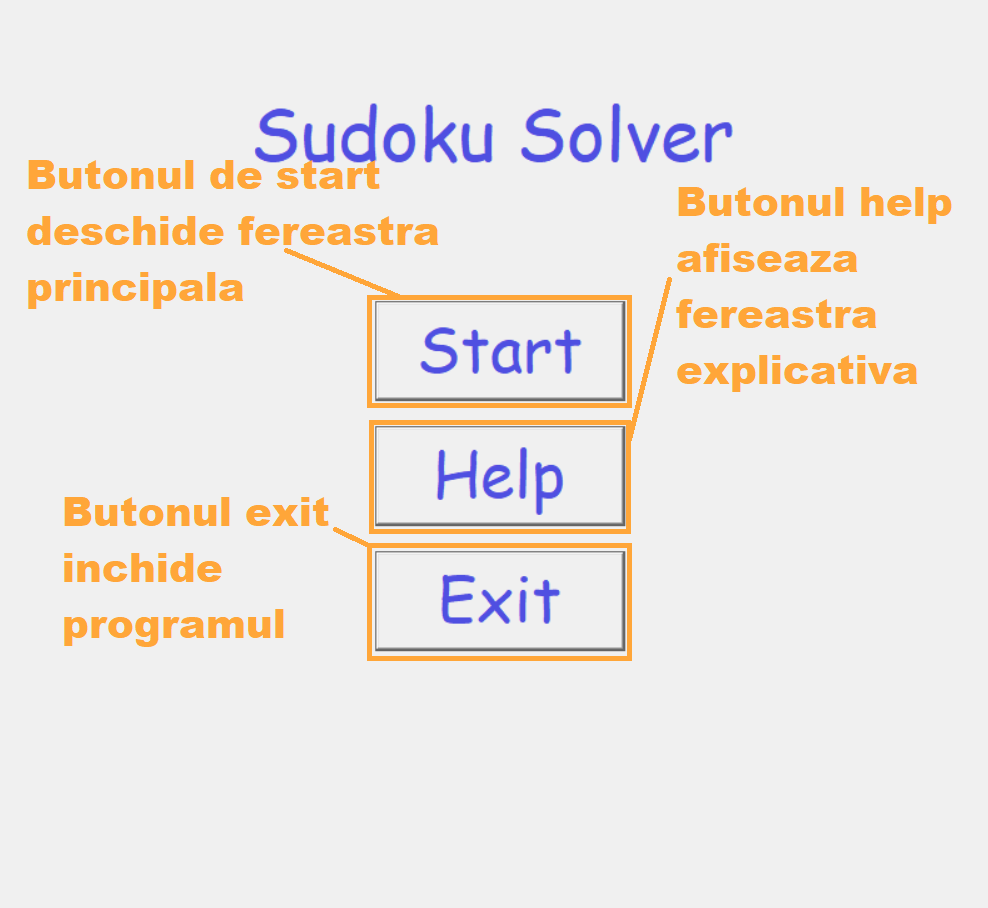
Windows API conține o ofertă de servicii pentru toate aplicațiile bazate pe ferestre grafice (en. windows). Această interfață de [programare](https://ro.wikipedia.org/wiki/Programare" \o "Programare) permite utilizatorilor să realizeze o interfață grafică propriilor aplicații, să acceseze sistemul [computerului](https://ro.wikipedia.org/wiki/Computer" \o "Computer), memoria acestuia, dispozitivele (fie de intrare, fie de ieșire), să implementeze sunete, imagini, video sau funcțiuni de rețea (respectiv [Internet](https://ro.wikipedia.org/wiki/Internet)) în aceste aplicații. Programarea cu Windows API înseamnă primirea, interpretarea, trimiterea de „mesaje” către „ferestre”, sau „controale” (obiecte controlabile - ex. ToolBox, EditBox, Button, Text, CheckBox).

Funcția principală (en. main - WINAPI WinMain) a unei aplicații Win32 conține o „buclă” (structură iterativă) (ex. while() în limbajul [C++](https://ro.wikipedia.org/wiki/C%2B%2B)) în care sunt apelate funcțiile de preluare și traducere a mesajelor trimise de către utilizator (prin intermediul dispozitivelor de intrare, apoi al sistemului de operare, în cazul de față Windows). Mesajele sunt interpretate și trimise mai departe ferestrei active. Mesajele pot fi trimise atât de sistem, asemenea unor mesaje din "subconștient", dacă ar fi să facem o comparație cu [creierul uman](https://ro.wikipedia.org/wiki/Creier" \o "Creier), cât și explicit, "conștient", prin funcția SendMessage(). Fiecărei ferestre i se asociază o procedură responsabilă cu interpretarea mesajelor primite. Spre exemplu, dacă unei ferestre i se trimite un mesaj de „distrugere”, aceasta dispare. Dacă unui control de tip CheckBox i se trimite mesajul de validare, în dreptul său apare binecunoscutul marcaj de validare.

Procesul principal pentru WinApi este WINAPI WinMain fata de Main-ul obisnuit, acesta are un loop de tipul while( cat timp) care asteapta mesaje si le trimite unui decodor care apoi actioneaza intr-un switch si ia fiecare caz particular, apoi fiecare caz este accesat de un cod unic trimis prin parametrul “msg”.

## Fereastra principala

La deschiderea programului esti intampinat de o fereastra cu un meniu principal care consista in butoanele “start”,”help” si “exit”.



## Introducerea numerelor

Dupa apasarea butonului de “Start” pe ecran apare o grila de sudoku(de 9 pe 9 patratele) in care introducem numerele grilei si la apasarea butonului “Solve” in grila , in spatiile goale numerele corecte rezolvari.

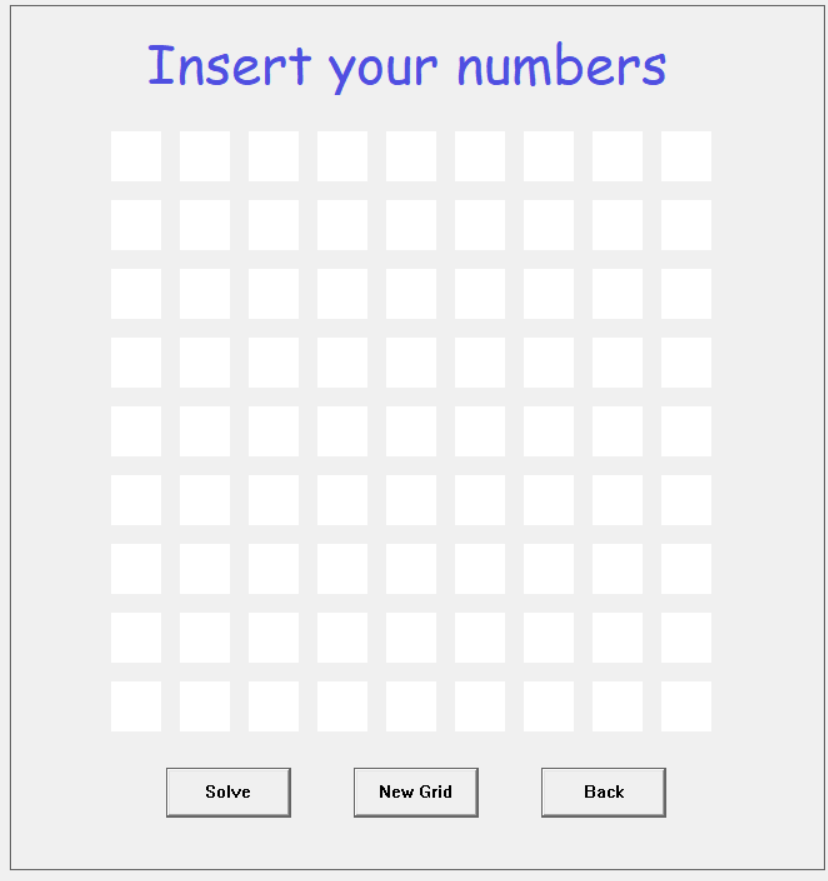
Butonul “New Grid” sterge numerele introduse in grila vor fi sterse iar grila din interiorul programului va fi stearsa.

Butonul “Back” inchide fereastra curenta astfel ne duce la meniul principal.



## 

## Panoul de introducere a numerelor dupa apasarea butonului de start:



case START\_BUT:

ShowWindow(button, SW\_HIDE);

ShowWindow(button2, SW\_HIDE);

ShowWindow(button3, SW\_HIDE);

ShowWindow(textMenu, SW\_HIDE);

platforma = CreateWindow("STATIC", NULL , WS\_BORDER | WS\_CHILD | WS\_VISIBLE| SS\_BITMAP, 175, 40, 650 ,690 ,hwnd ,NULL ,NULL , NULL);

SendMessageW(platforma, STM\_SETIMAGE, IMAGE\_BITMAP,(LPARAM)txt0);

textBox(hwnd);

break;

Când butonul de Start este apăsat, această parte a codului ascunde butoanele meniului principal, inserând apoi textul: „Insert your numbers”. Următorul pas este apelarea subprogramului textBox.

void textBox(HWND hwnd){

solve = CreateWindow("BUTTON", "Solve", WS\_CHILD | WS\_VISIBLE | WS\_BORDER | BS\_PUSHBUTTON

, 300, 650, 100, 40, hwnd,(HMENU)SOLV\_ST ,NULL ,NULL);

newG = CreateWindow("BUTTON", "New Grid", WS\_CHILD | WS\_VISIBLE | WS\_BORDER | BS\_PUSHBUTTON

, 450, 650, 100, 40, hwnd, (HMENU)NEW\_ST ,NULL, NULL);

bacK = CreateWindow("BUTTON", "Back", WS\_CHILD | WS\_VISIBLE | WS\_BORDER

, 600, 650, 100, 40, hwnd, (HMENU)MENU\_PROG ,NULL, NULL);

for(int i=0; i<9; i++){

for(int j=0; j<9; j++){

boxy[i][j] = CreateWindow("EDIT", "", WS\_CHILD | WS\_OVERLAPPED | WS\_GROUP | WS\_VISIBLE

| SS\_CENTER | ES\_NUMBER |ES\_MULTILINE, 80+(55\*j),100+(55\*i),40,40,platforma,NULL,NULL,NULL);

SendMessage(boxy[i][j],EM\_LIMITTEXT,1,NULL);

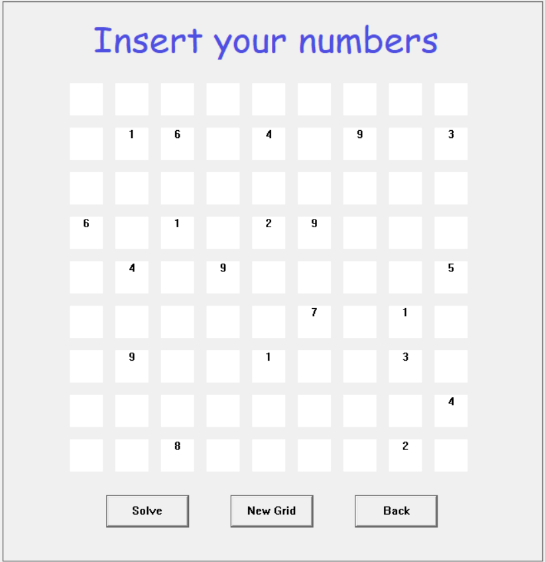
}

}

}

În primă instanță subprogramul crează butoanele solve, new grid și back. Apoi, prin intermediul celor două for-uri se crează optezi și una de casete de introducere a textului, limitate la o singură cifră per casetă.

## Panoul de introducere a numerelor dupa introducere:



int Solve(HWND hwnd){

char temp[2];

for(int i=0; i<9; i++){

for(int j=0; j<9; j++){

GetWindowTextA(boxy[i][j],temp,2);

if(strcmp(temp,"")) grid[i][j] = 0;

grid[i][j]=atoi(temp);

}

}

if(check(grid)){

if (solveSuduko(grid,0,0))

print(grid);

}else MessageBoxW(hwnd,L"There is no solve",L"Warning",MB\_ICONWARNING);

}

Prin intermediul for-ului subprogramul verifică dacă există text introdus în căsuță. În cazul în care aceasta este goală, în matrice va fi introdusă valoarea 0, iar în caz contrar, prin intermediul funcției atoi, textul va fi transformat din caracter în număr întreg, iar apoi este introdus de asemenea în matrice.

bool check(int grid[9][9]){

for (int i=0; i<9; i++){

for(int j=0; j<8; j++){

for(int k=j+1; k<9; k++){

if(grid[i][j] == grid[i][k] && grid[i][k] != 0)

return false;

if(grid[j][i] == grid[k][i] && grid[k][i] != 0)

return false;

}} }

for(int startVal=0; startVal<=6; startVal+=3)

for(int startCol=0; startCol<=6; startCol+=3){

int b=startVal;

while(b<=startVal+2) {

int m=startCol;

while(m<=startCol+2){

for(int i= startVal; i<=startVal+2; i++){

for(int j=startCol; j<=startCol+2; j++){

if(grid[b][m] == grid[i][j] && b!=i && m!=j && grid[b][m] != 0) return false;

}

}

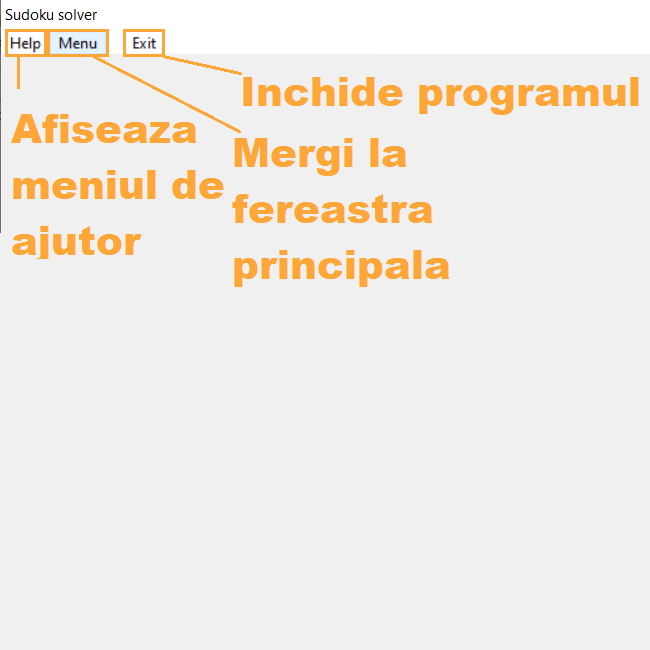
m++;

} b++; } } return true;}

Acest subprogram verifică numerele astfel încât ele să fie așezate conform regulilor jocului Sudoku. După această verificare, va fi apelat subprogramul solveSudoku, care verifică dacă există o posibilă rezolvare pentru datele introduse. Dacă aceasta există, pe ecran va fi afișată soluția, în caz contrar se va afișa mesajul: „There is no solve”.

## Submeniul

Submeniul ferestrei contine optiunile Help, Menu, Info si Exit. “Help” incepe prezentarea imaginilor explicative ale programului, “Menu” utilizeaza functia back si ne aduce la meniul principal, “Info” prezinta titlul si autorul proiectului si “Exit” inchide complet programul.



## 

## Programul de procesare a mesajelor:

int WINAPI WinMain(HINSTANCE currentInstance, HINSTANCE previousInstance, PSTR cmdLine, INT cmdCount) {

///hide console

ShowWindow(GetConsoleWindow(), SW\_HIDE);

/// register the window class

const char\* CLASS\_NAME = "myWin32WindowClass";

WNDCLASS wc = {0};

wc.hInstance = currentInstance;

wc.lpszClassName = CLASS\_NAME;

wc.hCursor = LoadCursor(nullptr, IDC\_ARROW);

wc.hbrBackground = GetSysColorBrush(COLOR\_BTNFACE);

wc.lpfnWndProc = WindowProcessMessages;

RegisterClass(&wc);

///create aplication window

CreateWindow(CLASS\_NAME, "Sudoku solver",

WS\_OVERLAPPED | WS\_VISIBLE, //bodystyle

250, 15, //position

x, y, //size

NULL, NULL, NULL, NULL);

///win loop

MSG msg {};

while (GetMessage(&msg, nullptr, 0, 0)) {

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return 0;

}

## Programul de executare:

Subprogramul WindowProcessMessages preia mesajul si prin intermediul unei instructiuni de tipul switch care pentru fiecare buton si mesaj in parte executa o secventa de cod.

LRESULT WindowProcessMessages(HWND hwnd, UINT msg, WPARAM param, LPARAM lparam) {

switch (msg)

{

case WM\_COMMAND:

switch(param){

case FILE\_MENU\_EXIT:

PostQuitMessage(0);

break;

case NEW\_ST:

emptybox(hwnd);

break;

case FILE\_MENU\_NEW:

MessageBeep(MB\_ICONINFORMATION);

MessageBoxA(hwnd,"Sudoku solver;\n Proiect Realizat de Damean Andrei", "About us",MB\_OK);

break;

case MENU\_PROG:

ShowWindow(platforma,SW\_HIDE);

ShowWindow(textP, SW\_HIDE);

ShowWindow(solve, SW\_HIDE);

ShowWindow(newG, SW\_HIDE);

ShowWindow(bacK, SW\_HIDE);

ShowWindow(button, SW\_SHOW);

ShowWindow(button2, SW\_SHOW);

ShowWindow(button3, SW\_SHOW);

ShowWindow(textMenu, SW\_SHOW);

ct++;

if(ct>=5)

ShowWindow(cat\_pisi, SW\_SHOW);

break;

case HELP\_NON:

loadImage();

ajut = CreateWindow("STATIC",NULL,WS\_VISIBLE | WS\_CHILD | WS\_POPUPWINDOW | SS\_BITMAP,400,80,650,650,hwnd,NULL,NULL,NULL);

for(int i=1; i<=3; i++){

SendMessageW(ajut,STM\_SETIMAGE,IMAGE\_BITMAP, (LPARAM)hAjutor[i]);

ShowWindow(ajut, SW\_SHOW);

Sleep(4000);

}

ShowWindow(ajut, SW\_HIDE);

break;

case START\_BUT:

ShowWindow(button, SW\_HIDE);

ShowWindow(button2, SW\_HIDE);

ShowWindow(button3, SW\_HIDE);

ShowWindow(textMenu, SW\_HIDE);

platforma = CreateWindow("STATIC", NULL , WS\_BORDER | WS\_CHILD | WS\_VISIBLE| SS\_BITMAP, 175, 40, 650 ,690 ,hwnd ,NULL ,NULL , NULL);

SendMessageW(platforma, STM\_SETIMAGE, IMAGE\_BITMAP,(LPARAM)txt0);

textBox(hwnd);

break;

case SOLV\_ST:

Solve(hwnd);

break;

case CAT\_BUT:

pisica(hwnd);

break;

case IESIRE\_BUTT:

PostQuitMessage(0);

return 0;

break;

default:

MessageBeep(MB\_ERR\_INVALID\_CHARS);

}

break;

case WM\_CREATE:

loadImage();

AddMenus(hwnd);

break;

case WM\_DESTROY:

PostQuitMessage(0);

return 0;

default:

return DefWindowProc(hwnd, msg, param, lparam);

}

}