NOMBRE:

ALGORITMOS DE EXPLORACION Y BACKTRACKING

OBJETIVOS:

Los objetivos a conseguir en esta práctica son:

- Profundizar y poner en práctica los conocimientos acerca de la técnica de programación conocida como Vuelta atrás o (*Backtracking*).
- Uso de la recursión para la implementación de esta técnica.

INTRODUCCIÓN.

La técnica de Vuelta atrás es uno de los métodos más usados en problemas abordados mediante estrategias de Prueba y error, ya que ofrece la posibilidad de volver a un punto anterior de la cadena de acciones para encontrar una solución a un problema, cuando se demuestra que las acciones tomadas hasta el momento no conducen a una solución.

Esta técnica genera un árbol de opciones (aunque no lo implemente físicamente), que va explorando en profundidad. Es lo que se conoce exploración del espacio de soluciones. Si la opción tomada no es adecuada para resolver el problema, vuelve hacia atrás para encontrar el nodo inmediatamente anterior que tiene alguna otra opción para explorar. La exploración generalmente acaba cuando se encuentra una solución, aunque se pueden construir algoritmos que no se agoten en este punto y realicen búsquedas exhaustivas de todas las soluciones existentes.

DESARROLLO: SUDOKU

Sudoku es un rompecabezas matemático de colocación que se surgió en Japón en 1986 y se dió a conocer en el ámbito internacional en el año 2004. Muchos periódicos internacionales publican diariamente estos puzzles en su sección de pasatiempos. Sudoku es una abreviatura japonesa que significa, los números deben aparecer solo una vez (“number singly”). Por lo que el objetivo del juego 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. No se debe repetir ninguna cifra (del 1 al 9) en una misma fila, columna o subcuadrícula.

A continuación se muestra un ejemplo del rompecabezas Sudoku. En la figura 1 se muestra la configuración inicial, es decir se muestra el puzzle con algunos números ya dispuestos en celdas. En la figura 2 se muestra la solución final dada la configuración inicial de la figura 1.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Figura 1: Configuración inicial

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Figura 2: Solución final

1. Crea un programa (“***Practica7.cpp***”) que implemente el algoritmo anterior y permita visualizar por pantalla el puzzle de entrada asi como la solución final. Se debe utilizar la estrategia de backtracking para resolver el problema de Sudoku obteniendo una solución única (ya que un sudoku está bien planteado, si la solución es única). Para ello se implementará un algoritmo para resolver el puzzle de forma recursiva

2. Además se debe calcular cuantas llamadas recursivas se realizan para los casos (puzzle 1, 2 , 3 y 4) que se proponen a continuación (se ha de hacer notar que en los casos propuestos los espacios vacios se han representado con cel número 0) ¿Es significante la diferencia en el número de llamadas recursivas para los casos propuestos?¿Por qué?

Puzzle 1	Puzzle 2	Puzzle 3	Puzzle 4
830509060	000701008	500000002	130080700
062108700	005490000	002010080	080600000
700030000	007020900	010500007	050004000
210000030	803200609	000086013	400050000
005306900	500000002	060000070	201000809
070000086	602007504	350120000	000060003
000010008	004010300	200009040	000100060
007805340	000072100	070030900	000009040
050204071	100508000	400000001	008020031