Algoritmia y Complejidad

Ejercicios tema 5

Laboratorio Miércoles 15:00-17:00

Integrantes:

Carlos Javier Hellín Asensio
Diego Gutiérrez Marcos

Curso: 2020-2021

Problema 4

Descripción del algoritmo:
Para este algoritmo lo que se ha hecho es una recursividad donde el caso base es cuando el próximo movimiento del caballo ya ha superado el número total de casillas (esto es filas * columnas) y sino, con dos bucles se empieza a recorrer todo el tablero y si es movimiento válido (usando la fórmula del enunciado) y esa casilla todavía no se ha visitado, entonces se da como visitada y se llama recursivamente para el siguiente movimiento, es decir, movimiento + 1.
Este movimiento sirve para luego mostrar el tablero con los movimiento que ha ido realizando el caballo como se muestra a continuación en los casos de prueba.
Para que el caballo pueda pasar por la misma casilla en las siguientes llamadas recursivas, se pone esa casilla como no visitado y al igual que en el caso de no poder recorrer todo el tablero con el caballo se muestra que no hay solución.

Casos de prueba:
En este caso de prueba hemos escogido un tablero de 4x4 la cual no puede recorrer todo el tablero

```
No se ha encontrado solución para que el caballo recorra todo el tablero de 4 x 4
con el caballo en la posición inicial de (0, 0)
```

En este caso de prueba hemos escogido un tablero 5x5 en la que sí puede recorrer todo el tablero utilizando los movimientos del caballo.
Funcionamiento del algoritmo:
Nuestro algoritmo tiene como funcionalidad de explicar si el caballo puede moverse en todas las casillas del tablero y si puede mostrar el tablero siendo cada casilla el número en el que el caballo ha sido movido. En este caso el caballo comienza en la posición 0,0 como se ve indicado con un 1 en el tablero de la imagen de abajo. Lo que hace el algoritmo es recorrer todas las casillas y comprobar si el movimiento del caballo es válido y no se ha visitado en cada una de ellas Cuando una de esas casillas cumple con estas condiciones como por ejemplo en la casilla 2,1 en el segundo movimiento y entonces se vuelve a llamar al mismo algoritmo para realizar las mismas comprobaciones en la casilla encontrada(en este caso 2,1) y sumándole un 1 al movimiento para guardar que el movimiento en el que se realizó era el segundo. Este algoritmo irá sucediendo hasta que haya rellenado todo el tablero o que se pare porque no haya posibilidad de que el caballo realice un movimiento. En este caso que he hemos explicado el caballo puede moverse

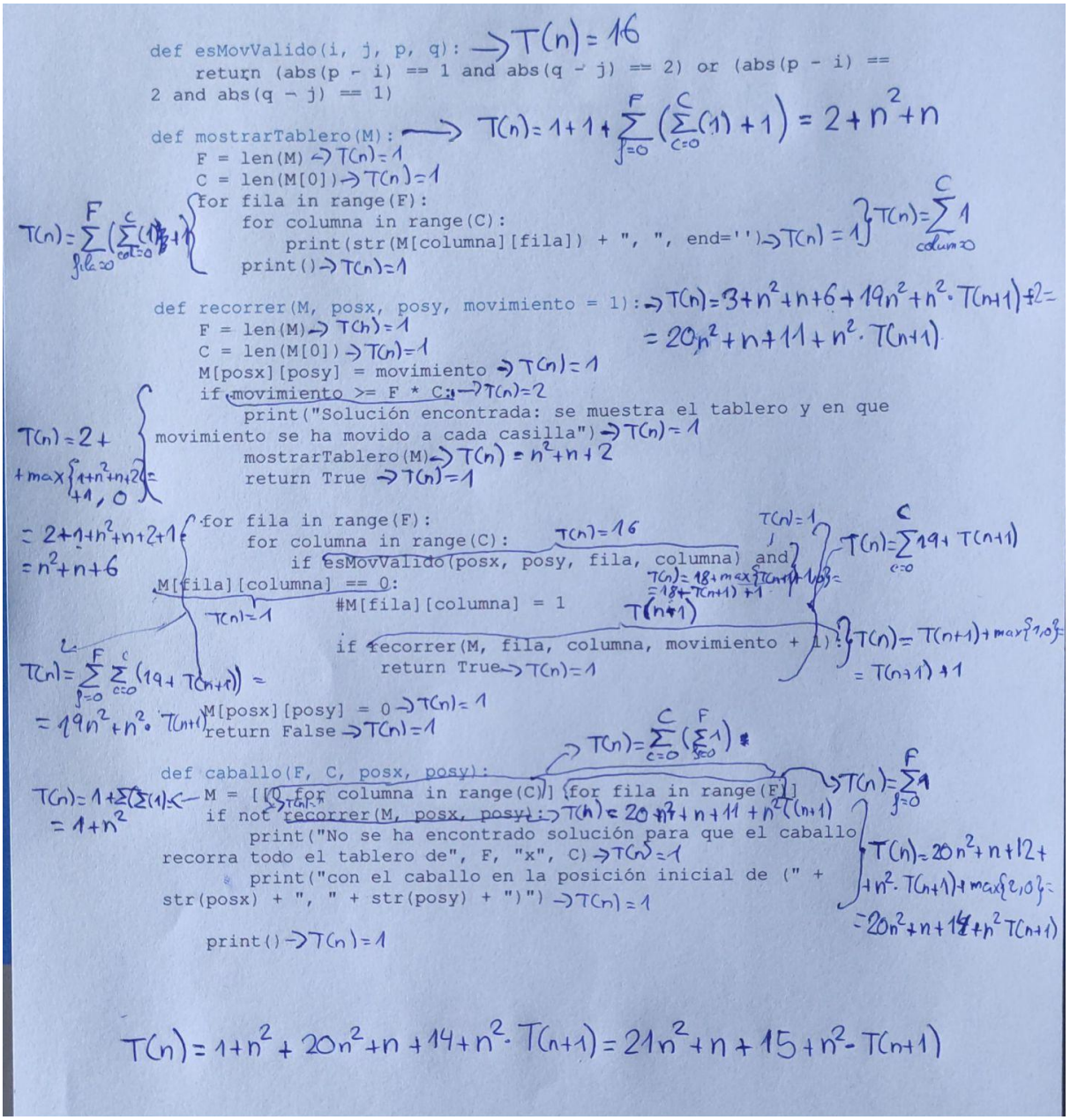
por todo el tablero y entonces se muestra el tablero indicando el momento en el que ha realizado cada movimiento.

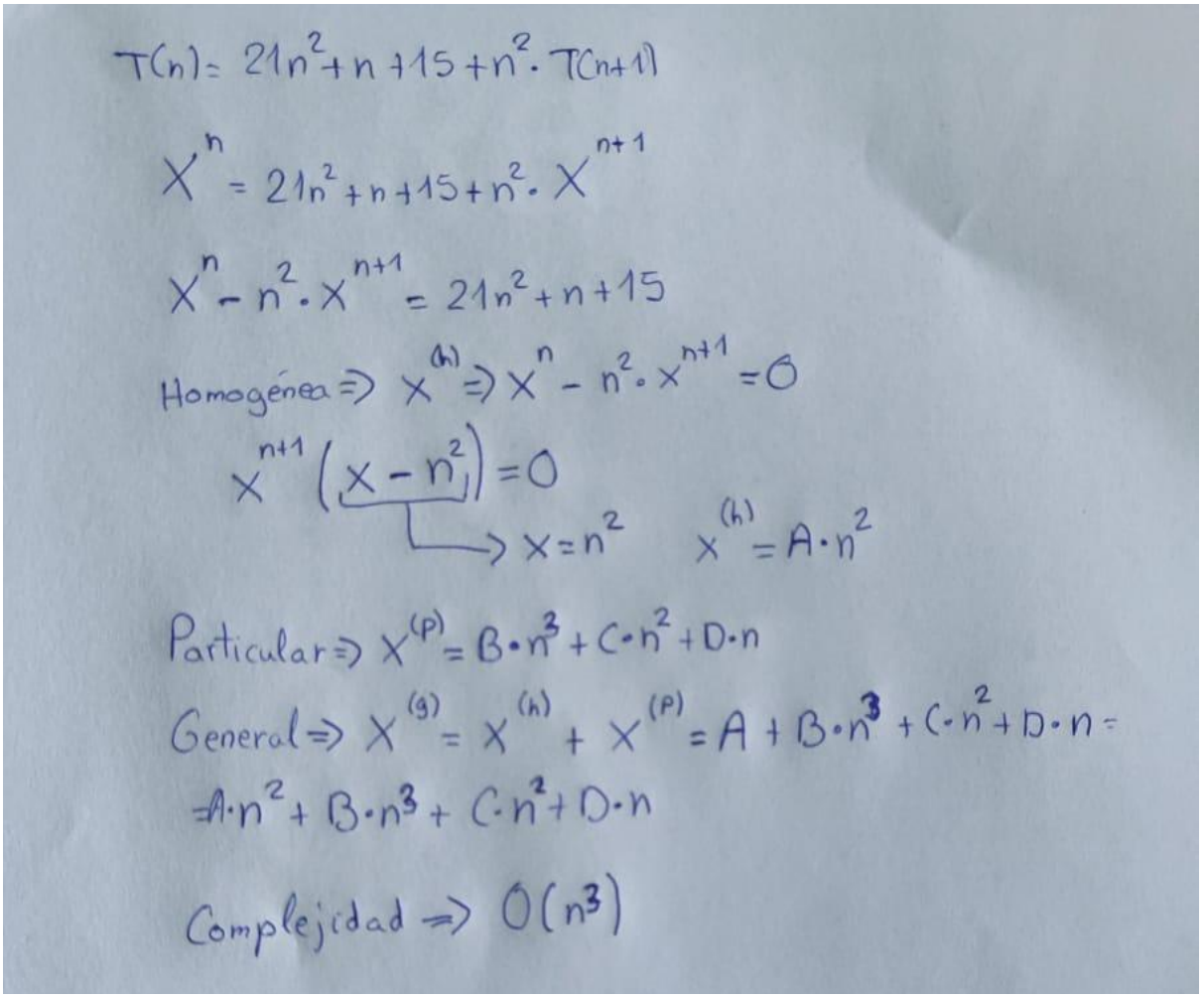
```
Solución encontrada: se muestra el tablero y en que movimiento se ha movido a cada casilla
1, 6, 17, 12, 23,
18, 11, 22, 7, 16,
5, 2, 13, 24, 21,
10, 19, 4, 15, 8,
3, 14, 9, 20, 25,
```

Este caso de prueba es el indicado en el enunciado, el cual tiene un tablero de 8x8 y que siempre tiene solución independientemente de la posición en la que comenzase el caballo

```
Solución encontrada: se muestra el tablero y en que movimiento se ha movido a cada casilla
1, 10, 31, 64, 33, 26, 53, 62,
12, 7, 28, 25, 30, 63, 34, 51,
9, 2, 11, 32, 27, 52, 61, 54,
6, 13, 8, 29, 24, 35, 50, 41,
3, 18, 5, 36, 49, 40, 55, 60,
14, 21, 16, 23, 46, 57, 42, 39,
17, 4, 19, 48, 37, 44, 59, 56,
20, 15, 22, 45, 58, 47, 38, 43,
```

Cálculo de la complejidad:





Problema 6

Descripción del algoritmo:
El algoritmo se implementa con backtracking como se pide en el enunciado. Primero tenemos el caso base de si la longitud de la cadena es 1 y el primer carácter de dicha cadena es el que buscamos, entonces se muestra la solución por pantalla.
En caso contrario, se hace un bucle de la longitud - 1 de la cadena para obtener el primer y segundo carácter, si estos pertenecen a la tabla de sustitución M entonces se realiza la sustitución de los dos caracteres por el valor que se haya en la tabla. Una vez hecho, se añade como posible solución y se llama recursivamente al algoritmo con la cadena sustituida en busca de la siguiente sustitución.
En el caso de no llegar a la solución de encontrar al final el carácter que buscamos, se quita la última solución y probamos con los otros dos siguientes caracteres de la cadena para ver si llegamos a la solución (aquí es donde se realiza el backtracking).

En el caso de que el primer o segundo carácter no pertenezca a la tabla (es decir, no es a, b, c o d) se da un aviso de error y se termina el algoritmo.

Casos de prueba:
En este caso de prueba, buscamos todos los casos posibles en los que se pueda reducir la cadena “acabada” al carácter “d” cambiando los valores de las cadenas con los valores que correspondan de la tabla de sustitución M.
Funcionamiento del algoritmo:
En este ejemplo comenzamos cogiendo los dos primeros caracteres de la cadena “acabada” que son ‘a’ y ‘c’ y comprobamos que se encuentran en la tabla de sustitución M. Si están como es este caso, queremos sustituir con un valor de la tabla M los dos primeros caracteres de la cadena. Para encontrar ese valor debemos buscar en la posición correspondiente dependiendo de los 2 caracteres que tenemos. Si tenemos el carácter ‘a’ una posición de la fila o la columna en la que buscar es 0, si es el carácter ‘b’ es 1, si es el ‘c’ es 2 y si es el ’d’ es 3. Como los dos caracteres son a y c tenemos los valores 0 y 2. El valor del primer carácter indica el número de la fila mientras que el segundo indica el número de la columna, en este caso el número de la fila es 0 y el de la columna es 2. Por tanto al buscar en la tabla en la posición 0,2 encontramos el valor a y con ese los sustituimos por los dos primeros caracteres. Este proceso se sigue repitiendo hasta que únicamente queda un carácter al final. Si ese carácter final coincide con ‘d’ pues borramos la solución anterior porque no es correcta y realizamos el proceso de sustitución utilizando los dos siguientes caracteres. Por ejemplo, este caso ocurre en el primer carácter al que llegamos como solución ya que se nos queda como solución ‘b’. Al borrarlo como solución volvemos a la cadena de dos caracteres que teníamos en ese momento ‘aa’, como no podíamos escoger otros caracteres volvemos a borrar esta cadena quedándonos para trabajar con la cadena ‘bda’. Al cambiar los valores escogido como caracteres por los dos siguientes nos volvía a ocurrir lo mismo se nos quedaba como carácter final uno diferente a ‘d’ en este caso quedaba ’a’ así que volvíamos a quitar soluciones hasta la cadena de 4 caracteres que se mantenía igual. Tras cambiar los caracteres escogidos realizamos otra vez el proceso de sustitución de dos caracteres por uno de la tabla M y finalmente conseguimos llegar a nuestra primera opción posible que acababa en ‘d’. Este proceso se sigue llevando a cabo de esta forma hasta que todas las soluciones posibles han aparecido por pantalla como hemos mostrado en las imágenes de debajo

```
A continuación se muestran todas las posibles soluciones de sustituir acabada hasta llegar a d
['acabada', 'aabada', 'bbada', 'aada', 'ada', 'da', 'd']
['acabada', 'aabada', 'bbada', 'aada', 'ada', 'ad', 'd']
['acabada', 'aabada', 'bbada', 'aada', 'aad', 'ad', 'd']
['acabada', 'aabada', 'bbada', 'bcda', 'bca', 'da', 'd']
['acabada', 'aabada', 'bbada', 'bcda', 'bcd', 'bc', 'd']
['acabada', 'aabada', 'bbada', 'bbda', 'ada', 'da', 'd']
['acabada', 'aabada', 'bbada', 'bbda', 'ada', 'ad', 'd']
['acabada', 'aabada', 'bbada', 'bbda', 'bbd', 'ad', 'd']
['acabada', 'aabada', 'bbada', 'bbad', 'aad', 'ad', 'd']
['acabada', 'aabada', 'bbada', 'bbad', 'bcd', 'bc', 'd']
['acabada', 'aabada', 'bbada', 'bbad', 'bbd', 'ad', 'd']
['acabada', 'aabada', 'abada', 'acda', 'ada', 'da', 'd']
```

Este caso de prueba es el mismo caso que estamos utilizando en la imagen anterior lo único que en azul se ve reflejada una de las posibles soluciones que nos indica el enunciado.

```
['acabada', 'acacda', 'aacda', 'aacd', 'bcd', 'bc', 'd']
['acabada', 'acacda', 'aacda', 'aacd', 'aad', 'ad', 'd']
['acabada', 'acacda', 'aacda', 'aacd', 'aac', 'bc', 'd']
['acabada', 'acacda', 'abcda', 'bcda', 'bca', 'da', 'd']
['acabada', 'acacda', 'abcda', 'bcda', 'bcd', 'bc', 'd']
['acabada', 'acacda', 'abcda', 'abca', 'bca', 'da', 'd']
['acabada', 'acacda', 'abcda', 'abca', 'ada', 'da', 'd']
['acabada', 'acacda', 'abcda', 'abca', 'ada', 'ad', 'd']
['acabada', 'acacda', 'abcda', 'abcd', 'bcd', 'bc', 'd']
['acabada', 'acacda', 'abcda', 'abcd', 'abc', 'bc', 'd']
['acabada', 'acacda', 'abcda', 'abcd', 'abc', 'ad', 'd']
['acabada', 'acacda', 'acada', 'aada', 'ada', 'da', 'd']
['acabada', 'acacda', 'acada', 'aada', 'ada', 'ad', 'd']
['acabada', 'acacda', 'acada', 'aada', 'aad', 'ad', 'd']
['acabada', 'acacda', 'acada', 'acda', 'ada', 'da', 'd']
['acabada', 'acacda', 'acada', 'acda', 'ada', 'ad', 'd']
```

Cálculo de la complejidad:

```
def pertenece(caracter):
    return "a" <= caracter <= "d"

def indice(caracter):
    return ord(caracter) - ord("a")

def sustitucion(cadena, caracterFinal, solucion = []):
    n = len(cadena)
    if not solucion:
        solucion.append(cadena)
    if n == 1 and cadena[0] == caracterFinal:
        print(solucion)
    for i in range(n - 1):
        primerCaracter = cadena[i]
        segundoCaracter = cadena[i + 1]
        if pertenece(primerCaracter) and pertenece(segundoCaracter):
            valor = M[indice(primerCaracter)][indice(segundoCaracter)]
            cadenaSustituida = "".join([cadena[:i], valor, cadena[i + 2:]])
            solucion.append(cadenaSustituida)
            sustitucion(cadenaSustituida, caracterFinal, solucion)
            solucion.pop()
        else:
            print("El carácter", primerCaracter, "o",
segundoCaracter, "de la cadena de texto no pertenece a la tabla de
sustitución M")
            return
```

$T(n) = 8 + 1 + 3 + \max\{9 + 1 + 1 + T(n-1), 1 + 1\} = 7 + 9 + 1 + 1 + T(n-1) = 18 + T(n-1)$

$T(n) = \sum_{i=0}^{n-1} 2 + 19 + T(n-1) = \sum 21 + T(n-1) = 21n + n \cdot T(n-1)$

$T(n) = 8 + 21n + n \cdot T(n-1)$

$T(n) = 8 + 21n + T(n-1) \cdot n$

$X^n = 8 + 21n + X^{n-1} \cdot n$

$X^n - n \cdot X^{n-1} = 21n + 8$

Homogénea $\Rightarrow X^{(h)} \Rightarrow X^n - n \cdot X^{n-1} = 0$

$X^{n-1}(x - n) = 0$

$\rightarrow x = n \qquad X^{(h)} = A \cdot n$

Particular $\Rightarrow X^{(p)} = B \cdot n^2 + C \cdot n$

General $\Rightarrow X^{(g)} = X^{(h)} + X^{(p)} =$

$= A \cdot n + B \cdot n^2 + C \cdot n$

Complejidad $\Rightarrow O(n^2)$