UNIVERSIDAD DE LA FRONTERA

Taller 5 - Algoritmos y Paradigmas- ICC507
Estructuras de datos en python - 2024-2

Carlos Cares, Ph.D.
carlos.cares@ceisufro.cl

Problema 1. (20%).

El juego de las Torres de Hanoi, en su versión más simple, se trata de 3 torres que contienen 4 discos de diferente ancho. Al principio la primera torre tiene todos los discos, cada disco puede estar en una torre sólo si es el único disco o si hay un disco de mayor tamaño bajo él. El juego consiste, en una mínima cantidad de movimientos, dejar todos los discos en la última torre, considerando que existe la restricción de mover (o tocar) un disco a la vez.

Construya un programa python que, dado una entrada que representa cada torre por las letras A, B y C,, una por línea. Luego de la letra podrían venir números indicando los discos de arriba hacia abajo etiquetados por su ancho. El número representa el ancho del disco. De este modo la situación de la figura se representaría en la entrada de la siguiente manera:

A 1 2 3 4
B
C

La entrada tendrá una línea adicional que es la propuesta de un movimiento como un par de letras, por ejemplo A B, lo que indica que se desea mover el disco de la primera torre a la segunda torre. La salida debe ser la siguiente: si el movimiento es posible, entonces se escribe la nueva configuración. Si el movimiento NO es posible, la salida debe indicar “No hubo movimiento” y se repite la configuración de entrada. Por ejemplo:

Ejemplo 1
Entrada
A 1 2 3 4
B
C
A B

Salida
A 2 3 4
B 1
C

Ejemplo 2
Entrada
A 3 4
B 1
C 2
B A

Salida
A 1 3 4
B
C 2

Ejemplo 3
Entrada
A 3 4
B 1
C 2
A B
No hubo movimiento (3 es más ancho que 2)
A 3 4
B 1
C 2

Si la entrada no tiene este patrón, el programa debe escribir un mensaje que diga “torres o discos mal etiquetados”, ya sea que no se usan las letras de torres o se usan números o símbolos diferentes a los considerados para los discos.

UNIVERSIDAD DE LA FRONTERA

Problema 2. (20%).

Corrija el programa anterior para que se pueda “jugar” continuamente, es decir que el programa parte con la configuración inicial (todos los discos en la torre A) y sólo lea las solicitudes de movimiento una tras otra.

Problema 3. (20%).

Modifique el programa anterior para que funcione de la siguiente manera, el programa debe detectar cuando se ha finalizado, escribiendo el mensaje “logrado” en la movida <número de movida>. Pero el programa debe detectar además si ha llegado a una configuración repetida, diciendo simplemente “jugada repetida” en la movida <lista de números de movidas>. Use como tope 16 movidas. Si se alcanzan las 16 movidas sin llegar al estado deseado el programa debe decir “no logrado”, y terminar.

Problema 4. (20%).

Sobre su solución del problema 2 o sobre su solución del problema 3, agregue en la salida del estado del juego una línea con las jugadas posibles por ejemplo:

Ejemplo 1
A 1 2 3 4
B
C
Movidas posibles A>B, A>C
(lectura)

Ejemplo 2
A 3 4
B 2
C 1
Movidas posibles B>A, C>1

Problema 5. (20%).

En esta solución se pide que construya un programa que encuentre una solución al juego de la siguiente manera. Considere una representación de un estado como una lista que contenga a su vez una lista por cada torre, una cuarta lista con las movidas posibles una quinta lista con los nodos correspondientes a las movidas hechas y una

sexta lista que contiene los estados anteriores por los que ha pasado el juego. Note que debe haber una correspondencia en las movidas posibles y los nuevos estados alcanzados. Con esta estructura se pide que genere un programa que solucione la torre de Hanoi para otros casos buscando si el juego ha alcanzado el estado deseado.

Ejemplo de estado inicial con movidas posibles pero sin los próximos estados
[[1,2,3,4],[],[],[[0,1],[0,2]], [ ] , [ ]]

Si agregan los nodos del primer nivel, la estructura debe quedar así:
[[1,2,3,4],[],[],[[0,1],[0,2]], [
[[2,3,4], [1], [], [[0,2], [1,2], [1,0]], [ ]],
[[2,3,4], [], [1], [[0,1], [2,1], [2,0]], [ ]],
] [[[1,2,3,4],[],[]]]]

No que la lista de los estados anteriores se crea simplemente agregando las 3 primeras listas que representan las torres, en una lista, como el último elemento de la sexta lista. Si el nodo que representa un estado es el estado objetivo (osea [[], [], [1,2,3,4]]), entonces la respuesta al juego es el conjunto de estados en la sexta lista y eso es lo que se debe escribir en consola.

Recomendación. Separe en funciones la generación de opciones de movidas, y luego la aplicación de las movidas sobre el estado actual. El programa se probará cambiando una configuración diferente en cantidad de discos pero mantendrá las 3 torres. Por este motivo el estado final debe configurarse en cada prueba.