1. La función toma una posición determinada y una jugada posible. El primer valor de la tupla Jugada es la posición y el segundo la cantidad que se le quiere restar a la fila.

La función recorre la lista de posiciones hasta llegar a la fila en la que quiere realizar la jugada. Mientras la va recorriendo va agregando el valor de esa fila (la cantidad de piedras que hay en la fila), a la nueva posición que va a devolver.

Cuando llega a la fila en la que quiere hacer la jugada, pregunta si al restar lo que se le quiere quitar y la cantidad de la fila son iguales, entonces devuelve el resto de la nueva posición que estuvimos armando (para que no me devuelva una fila con 0 piedras). Y si al restar sobra algo en la fila, lo agregamos a la nueva posición que va a devolver.

Defino que, si entra por terminal una lista vacía, devuelve una lista vacía. Y le agregué un pequeño filtro para que cuando se quiera jugar quitándole a una fila más piedras de las que tiene, o si se quiere jugar en un numero de fila que no existe, devuelva indefinido.

1. Como la función devuelve un conjunto de combinatoria de tuplas, podemos verlo como el producto cartesiano entre la cantidad de filas y la cantidad de elementos por fila. Tomo la posición ingresada, y voy creando un conjunto de tuplas donde voy ingresando por cada fila existente en la posición dada, todas las posibles combinaciones de jugadas según la cantidad de piedras que haya en la fila.

Tomo la primera fila y un conjunto de todos los posibles movimientos dentro de la fila, y en “prodCartesiano” creo un conjunto con las tuplas de cada jugada posible dentro de la fila. Luego este conjunto de jugadas de la primera fila, la uno con los de las siguientes filas, recursando “generarProductoDesde”.

1. Tomo una posición ingresada por pantalla y todas las posibles jugadas de esa posición.

Entonces pregunto ¿si de las posibles jugadas juego la primera, la nueva posición es ganadora? Si es ganadora, y no tengo más posibles jugadas, entonces la mía es perdedora, si hay otra posible jugada, sigo con la siguiente. Si la siguiente posición es perdedora, la mía es ganadora.

La recursión funciona así: (que quede claro que cuando digo “siguiente posición”, es la posición resultante de una jugada) Cuando consulto sobre la siguiente posición, esGanadoraAux