Escuela de Ingenieria en Computacion
Taller de programacion
Grupo 1

Juguemos Katamino

Kendal Chacon Chaves
Oscar Daniel Rojas Rodriguez
Jose Mario Castro Cruz

Prof: Alex Brenes Brenes

24 de Mayo de 2024

0.1. Resumen Ejecutivo

El katamino es un juego de meza estilo rompecabezas, en el cual su mecánica se basa en un tablero dividido en N cuadros de alto y M cuadros de ancho, dicho tablero debemos llenarlo con una serie de piezas dadas que deben ser encajadas en el tablero de modo que no sobre ninguna pieza además de que en el tablero no quede ninguno cuadro sin ocupar. Dicho esto se nos presenta un problema en el cual tenemos que resolver Kataminos por medio de un algoritmo, normalmente resolver un katamino es una tarea que a una persona le puede tomar una cantidad alta de tiempo debido a que las piezas tienen en el peor de los casos hasta 8 posiciones diferentes y descubrir una posible solución nos puede toar cierto tiempo, pero como estamos en la era digital podemos diseñar un algoritmo que por medio de Backtracking pruebe todas las posibles combinaciones en cuestión de segundos en una computadora.

Nuestra solución al problema se divide en varias partes:

- Input :La funcion separador divide una cadena de entrada en listas de caracteres para poder procesar la forma de las piezas
- manipulación de piezas :Tenemos las funciones rotar y espejo que tienen como fin dar con todas las posibles posiciones y rotaciones de las piezas además de la función recortar que elimina las filas y columnas vacías
- verificación y manipulación de tablero :Tenemos las funciones isvalid, adentrodeltableroyaaprendiahacerestojajaja ,ponerpiezas y remover; que tienen como fin ir colocando piezas y ser las estructuras que intervienen en el backtracing.
- generación de posibles combinaciones :funcabra es la encargada de generar todas las posibles formas de cada pieza y las almacena en una lista
- Backtracking :Backtrack Realiza el algoritmo de backtracking para intentar todas las posibles combinaciones de piezas en el tablero

Como resultado de toda esta lógica obtuvimos un Código el cual es capaz de resolver kataminos dentro del rango de tiempo aceptable, además de que es un Código relativamente más corto que la mayoría lo cual hace que comprenderlo sea un poco más fácil y menos tedioso. Sumado a esto se puede notar que es un código que cumplió exactamente con todas las condicionales las cuales se nos exigen en el planteamiento del proyecto tales como los datos de entrada y salida.

0.2. Introducción

En el ámbito de la programación, la resolución de problemas constituye uno de los principales ejes alrededor de los cuales gira esta ocupación. La capacidad para abordar y resolver problemas complejos es fundamental para tener éxito en un mercado laboral en el cual existe muchísima oferta. En este contexto, se nos presenta la tarea de desarrollar un algoritmo utilizando la técnica de backtracking para encontrar una solución a un problema específico. Esta técnica es conocida por su capacidad para explorar todas las posibles soluciones por medio de fuerza bruta sin dejar ninguna posibilidad por fuera

El desafío consiste en diseñar un algoritmo que, mediante backtracking, encuentre una solución válida si existe, o retorne -1 en caso contrario. Este documento tiene como objetivo proporcionar una documentación exhaustiva de todos los aspectos relacionados con la elaboración de la solución. Se detallarán los pasos seguidos en el proceso de desarrollo, los desafíos e inconvenientes encontrados, y las estrategias empleadas para superarlos. Además, se presentarán estadísticas detalladas sobre el rendimiento del código, incluyendo tiempos de ejecución y eficiencia en el uso de recursos. Se van a detallar datos sobre cómo fue la elaboración de la solución, convertirlo a código Python, inconvenientes y desafíos que surgieron en la elaboración del mismo. Además, al final se va a hacer una conclusión en la cual se van a sintetizar los resultados obtenidos y una visión de parte de los integrantes del grupo acerca de que piensan del algoritmo y detalles que aprendieron o desean

0.3. Marco Teórico

Para la elaboracion de dicho proyecto se utilizaron varias herramientas las cuales son:

- Python(Lenguaje de Programacion)
  Python es un lenguaje de programación potente y fácil de aprender. Tiene estructuras de datos de alto nivel eficientes y un simple pero efectivo sistema de programación orientado a objetos. La elegante sintaxis de Python y su tipado dinámico, junto a su naturaleza interpretada lo convierten en un lenguaje ideal para scripting y desarrollo rápido de aplicaciones en muchas áreas, para la mayoría de plataformas.[1]

  En las navidades de 1989 Van Rossum, mientras trabajaba en un centro de investigación holandés (CWI), decidió empezar un nuevo proyecto como pasatiempo personal. Pensó en darle continuidad a ABC, un lenguaje de programación que se desarrollo en el mismo centro en el que estaba trabajando.

  Sin embargo, el proyecto no llegó mucho más lejos por las limitaciones del hardware de la época, así que Van Rossum decidió darle una segunda vida a su idea y partiendo de la base que tenía, empezó a trabajar en Python.[2]

- Visual Studio Code(IDE)
  Visual Studio Code, al que conocemos también como VSCode, es un editor de código para programadores gratuito, de código abierto y multiplataforma. Está desarrollado por Microsoft, una compañía con una dilatada experiencia en la creación de IDEs (entornos de desarrollo integrados), que ha conseguido plasmar su larga tradición en el sector para ofrecer una herramienta ligera y práctica que la comunidad ha adoptado en masa[3]
  Se podria decir que es uno de los IDE´s mas utilizados debido a la gran cantidad de extensiones que tiene disponible sin importar el lenguaje de programación[3]

- GitHub
  Github es un portal creado para alojar el código de las aplicaciones de cualquier desarrollador, y que fue comprada por Microsoft en junio del 2018. La plataforma está creada para que los desarrolladores suban el código de sus aplicaciones y herramientas, y que como usuario no sólo puedas descargarte la aplicación, sino también entrar a su perfil para leer sobre ella o colaborar con su desarrollo.[4] Es una herramienta muy recomendable debido a que varias personas pueden trabajar en un mismo proyecto, con la salvedad de que se pueden guardar varios versiones anteriores del mismo, por si fuera necesario tomar un paso hacia atras.

- Equipo de Pruebas
  Las pruebas fueron realizadas en una Lenovo Ideadapad 5 con las siguientes especificaciones: Procesador AMD Ryzen 5 5500U 2.10GHz, 6 Nucleos y 12 hilos, 8GB de memoria Ram y sistema operativo Windows 11.

- Repositorio Utilizado
  https://github.com/Kendallch75/Proyecto-2-Taller-de-programacion

- Descripcion del Juego
  El cuaderno de Katamino incluye varias páginas con numerosos retos para un jugador, que varían en función de su dificultad. En ellos, es necesario colocar todas las piezas correspondientes, haciéndolas encajar. Dependiendo del reto escogido y de su nivel de dificultad, se delimita el espacio del tablero con la guía. A continuación, se cogen las piezas que se indican.

  El objetivo es colocarlas en este espacio, de manera que no sobre ningún hueco. Una vez completado, se deshace y se pasa a un nuevo reto, de mayor dificultad. Debido a su gran número y a su dificultad creciente (algunos pueden llegar a ser casi imposibles) completarlos todos es una tarea que puede llevar varios meses. En caso de hacerlo, o de quedarse estancado, siempre se puede intentar superar antiguos rompecabezas, pero en menos tiempo.[5]

0.4. Solución

Acontinuacion se van a describir todas las funciones utilizadas para el algoritmo de solucion de Katamino

- La función separador(input) toma una cadena de caracteres como entrada y la divide en una lista de caracteres individuales, facilitando el procesamiento de las descripciones de las piezas en el juego de Katamino.

- La función rotar(matrix) recibe una matriz que representa una pieza y la rota 90 grados en sentido horario, generando una nueva orientación de la pieza. Esta función es fundamental para explorar todas las posibles combinaciones de piezas en el tablero durante el algoritmo de backtracking.

- La función espejo(matrix) refleja una matriz horizontalmente, lo que permite generar espejos de cada pieza.Se basa sobre todo en la transpuesta y es esencial para considerar todas las formas posibles de cada pieza durante el proceso de búsqueda de soluciones en el juego de Katamino.

- La función isfull(matrix) verifica si una matriz está completamente llena, es decir, si todas sus celdas contienen algún valor distinto de ".". Esta función se utiliza para determinar si se ha completado el tablero con todas las piezas colocadas.

- La función recortar(matrix) elimina las filas y columnas vacías alrededor de una matriz que representa una pieza. Esto ayuda a reducir el espacio de búsqueda y a eliminar las partes vacías de las piezas, lo que simplifica el proceso de colocación de las piezas en el tablero.

- La función ponerpiezas(E, piz, fila, col) intenta colocar una pieza en el tablero en una posición específica y actualiza el tablero si la colocación es válida. Esta función verifica si la colocación de la pieza viola las reglas del juego antes de actualizar el tablero.

- La función remover(E, piz, fila, col) elimina una pieza previamente colocada del tablero, restaurando las celdas ocupadas por esa pieza a su estado original (".") en el tablero.

- La función isvalid(E, piz, fila, col) verifica si una pieza puede ser colocada en una posición específica del tablero sin violar las reglas del juego. Esto incluye verificar si la pieza no entra en conflicto con otras piezas ya colocadas.

- La función adentrodelatablayaaprendiahacerestojajajajajaj(E, piz, fila, col) verifica si una pieza puede ser colocada dentro de los límites del tablero desde una posición dada, evitando que se coloque parcialmente fuera del tablero.

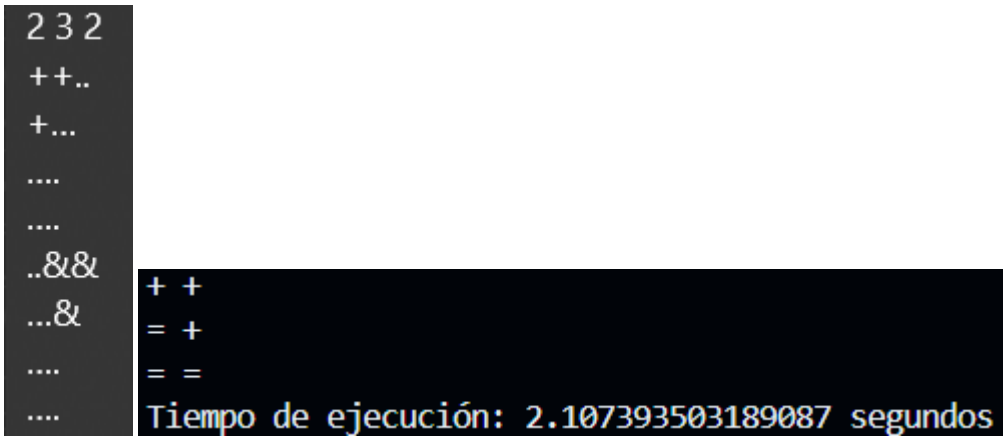
- La función funccabra(PL, PTEE) genera todas las posibles rotaciones y espejos de cada pieza y las almacena en una subpila dentro de una pila de las Piezas totales. Esto amplía las opciones disponibles durante el proceso de colocación de las piezas en el tablero.

- La función bactrack(E, PT, i) realiza el algoritmo de backtracking para intentar todas las combinaciones de piezas en el tablero. Comenzando desde la esquina superior izquierda del tablero, esta función prueba todas las piezas en todas las posiciones válidas hasta encontrar una solución válida o hasta que se hayan explorado todas las posibilidades.

0.5. Resultados de las pruebas

A continuación se muestran ejemplos de los inputs que se utilizaron seguidos de sus respectivos outputs en los cuales se incluye en tiempo de ejecución de cada caso

- Prueba 1

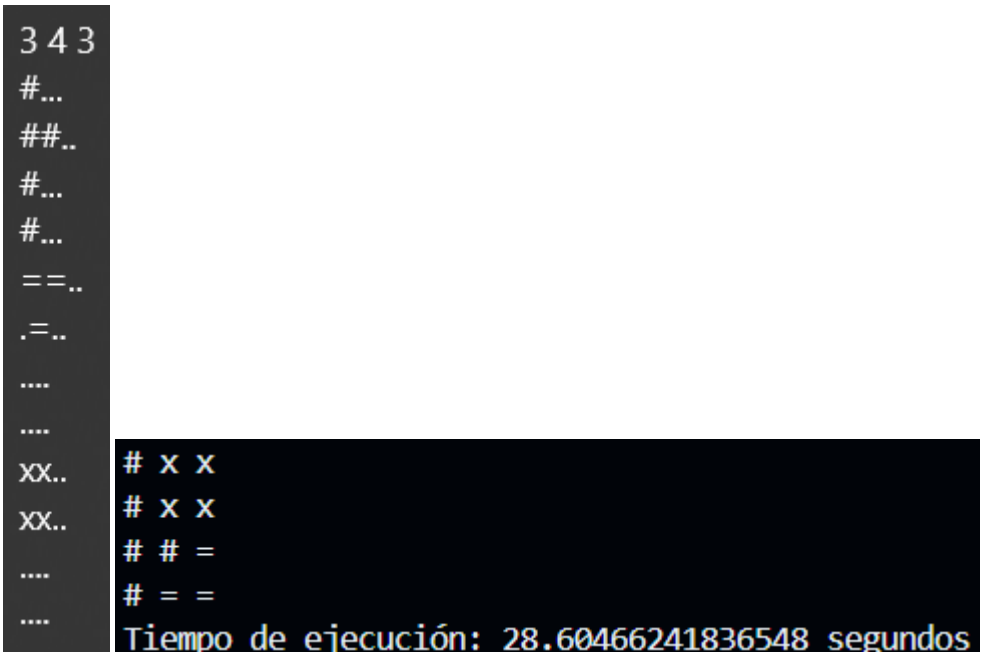


- Prueba 2

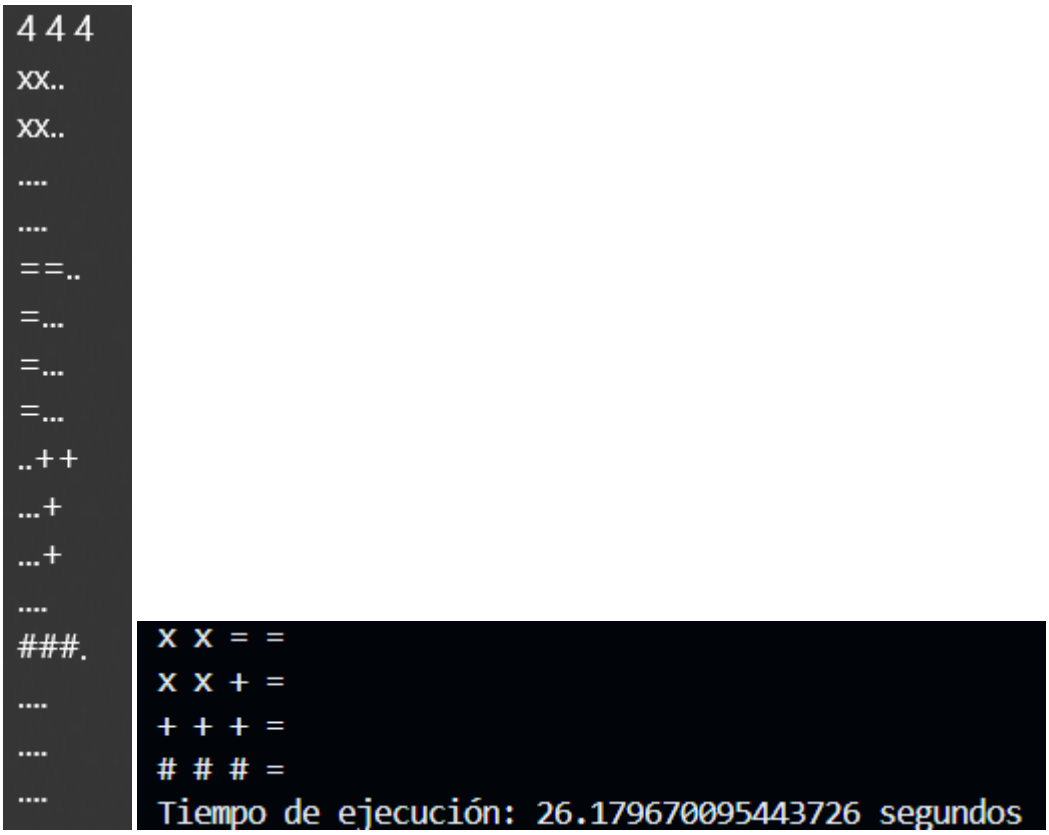
```
5 5 6
==..
.=..
....
....
+...
++..
.+..
.+..
&&..
&&..
&...
....
---.
---.
....
....
%%..
%%..
....
....
##..
....
....
....
```

```
= = + + +
= + + & &
- - & & &
- - % % #
- - % % #
Tiempo de ejecución: 5.393056154251099 segundos
```

- Prueba 3


3 4 3
#...
##..
#...
#...
==..
.=..
....
....
xx..
xx..
....
....
x x
x x
=
= =
Tiempo de ejecución: 28.604662418365548 segundos

- Prueba 4



Como se puede observar en las imágenes de los casos de pruebas mostrados anteriormente, se nota un tiempo de ejecución relativamente bajo(mayormente por debajo de las 30 segundos) aunque ya hablando en tiempo real(el lapso de tiempo desde que damos enter hasta que nos da el resultado estamos por debajo del segundo), resultados bastante alentadores que demuestran una correcta manipulación del algoritmo utilizando las líneas de Código justas y necesarias para un correcto y eficiente funcionamiento

0.6. Conclusiones

El proyecto de Juguemos Katamino es una propuesta interesante porque nos enseña lo universal que puede ser la utilización de un algoritmo basado en backtracking en problemas que tienen bastantes posibles soluciones, además que el ver como la computadora genera las soluciones con tal rapidez nos pone a reflexionar sobre todo lo que ha avanzado la tecnología y como nos puede ayudar en nuestro día a día a solucionar problemas mediante procesos computaciones que se ejecutan en segundos o a veces hasta milésimas. En general el grupo siente una satisfacción bastante grande al haber encontrado una solución a este problema que de entrada se plantea bastante complejo, pero en su desarrollo las ideas se van iluminando y tomando forma.

0.7. Aprendizajes

- Daniel- Aprendi a separar los stings de una entada, a utilizar de manera correcta el bactacking, a rotar las matrices, pero especialmente a utilizar de manera correcta las pilas.
- Kendal-Siento que aprendí muchísimo sobre la manipulación tanto de pilas como de matrices además de que Daniel me ayudo mucho en comprender los algoritmos y la lógicas detrás del backtracking

0.8. Bibliografia

[1] "El tutorial de Python," Python Documentation. https://docs.python.org/es/3/tutorial/

[2] Tokio School, "La historia de Python. Las versiones de un lenguaje único - Tokio School," Tokio School, Jan. 31, 2024. https://www.tokioschool.com/formaciones/cursos-programacion/python/historia/

[3]F. G. De Zúñiga, "¿Qué es Visual Studio Code y cuáles son sus ventajas?," Blog De arsys.es, Apr. 19, 2024. https://www.arsys.es/blog/que-es-visual-studio-code-y-cuales-son-sus-ventajas

[4]Y. Fernández, "Qué es Github y qué es lo que le ofrece a los desarrolladores," Xataka, Oct. 30, 2019. https://www.xataka.com/basics/que-github-que-que-le-ofrece-a-desarrolladores

[5]L. G. Abarca, "Katamino: un rompecabezas al estilo Tetris para uno y dos jugadores • Consola y Tablero," Consola Y Tablero, Dec. 13, 2018. https://consolaytablero.com/2015/04/06/katamino-un-rompecabezas-al-estilo-tetris-para-uno-y-dos-jugadores/