

Universidad Simón Bolívar

Departamento de Computación y Tecnología de la Información

Laboratorio de Algoritmos y Estructuras I CI-2691

Prof. Carmen Rosseline Rodríguez

Especificaciones del Proyecto

4 En Línea

Integrantes:

José Barrera. Carnet: 15 - 10123

Alfredo Cruz. Carnet: 14 – 10261

Sartenejas, abril de 2018

Introducción

El juego “4 En Línea” tiene como finalidad colocar cuatro fichas en línea (horizontal, vertical o diagonal) en un tablero vertical. En cada turno el jugador, deja caer una ficha en alguna de las 7 columnas. Cuando se encuentran alineadas 4 fichas del mismo jugador este queda ganador del juego y se concluye la partida actual. Sin embargo, puede llenarse el tablero y que no se logre ninguna conexión en cuyo caso se declara un empate. O puede que el jugador decida retirarse y automáticamente conceder la victoria al rival.

El problema a resolver consiste en lograr desarrollar en el lenguaje de programación Python y en el pseudo-lenguaje GCL, una adaptación del juego 4 En Línea. Para hallar la solución al problema implementaremos el código utilizando la estructura de análisis descendente, partiendo de una carta estructurada (que se anexará más adelante). Ésta permite tener una presentar una perspectiva más clara del código, pues divide y jerarquiza las tareas a realizar en pequeños procedimientos y funciones más fáciles de entender. Los cuales nos permitirán vislumbrar la solución del problema de forma más clara y organizada.

Descripción de la finalidad de las definiciones de tipos de datos:

Para hacer más legible el código y facilitar su lectura, decidimos establecer los siguientes tipos de datos al inicio del juego:

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Tipo** | **Descripción** |
| nombreusuario | String/Entrada | Nombre del Usuario |
| nivel | Integer/Entrada | Dificultad de la Máquina |
| partida | Integer/Entrada | Empezar Una Nueva Partida(0), Cargar Una Anterior(1) o Salir del Juego (else) |
| seguir | Bool/Entrada | Si se desea rendirse |
| x | Integer/Entrada | Fila donde el usuario desea jugar |
| y | Integer/Entrada | Columna donde el usuario desea jugar |
| A | List | Matriz Tablero de Juego |
| G | List | Tablero de Resultados |
| jugando | Bool | Controla el ciclo de jugar partida |
| turno | Integer | Contador de turnos de la partida |
| juegauser | Integer | Si le toca jugar al usuario |
| ganador | Integer | Ganador de la partida finalizada: Empate(0), Usuario(1) o Máquina (2) |
| dentrodeljuego | Bool | Controla el ciclo del juego |
| movida | Bool | Controla el ciclo de intentar una juagada para el Nivel 1 |
| i | Integer | Fila donde la Máquina juega |
| j | Integer | Columna donde la Máquina juega |
| datosdejuego | Class | Atributos: nombre, turno, nivel, A, i, j (valores de juego) |
| anterior | datosdejuego | Almacena los datos de juego de una partida anterior. |
| actual | datosdejuego | Almacena los datos de juego la partida actual. |

Manera en que la máquina hace la toma de decisiones:

Para el turno de la máquina, las jugadas siguen 2 estrategias bien diferenciadas de acuerdo al nivel de dificultad (1 o 2) que selecciona el usuario:

Nivel 1: Se intenta un par de coordenadas random hasta que estas se ubiquen en una casilla válida para jugar. Entonces se ejecuta la jugada.

Nivel 2:

al momento de moverse en el tablero, se utiliza un random para asignarle una posición posible luego de haber almacenado en un arreglo las posiciones válidas a las que se podría mover dependiendo de su valor obtenido en el dado. • La máquina puede realizar sospechas en habitaciones donde haya estado anteriormente, puede sospechar de cartas de su propia mano y no puede sospechar de cartas que ya le hayan refutado, pero sí de una habitación que ya haya sido refutada puesto que no existe la restricción de no moverse a una habitación refutada. • La máquina solo puede acusar cuando tiene las cartas del sobre, es decir, cuando tiene la posibilidad de ganar.

Conclusiones

La realización del presente proyecto de laboratorio, nos permitió implementar la estructura de análisis descendente para resolver un problema de complejidad sin precedentes dentro del curso. Durante su desarrollo, se hizo evidente que el dividir la tarea en pequeños problemas que trabajan de forma interconectada, realmente permite atacar los problemas de forma más eficaz. Esto debido a que se puede analizar cada componente como una nueva mini-tarea, enfocando la capacidad de resolución de problemas en algo más simple, que incluso pudiese aplicarse en futuras macro-tareas.

Nos dimos cuenta, además, de que el análisis descendente es un proceso que no es ajeno a los cambios, pues las primeras soluciones a cada mini-tarea, por lo general no escapaban de mejoras en pro de simplificar el código, hacerlo más legible e inclusive agregarle nuevas funcionalidades al programa.

En cuanto a los algoritmos y las estructuras de datos, podemos concluir que fijar las herramientas adecuadas es determinante para desarrollar soluciones elegantes a problemas concretos. Un aspecto básico que, de descuidarse, deviene en una pérdida de tiempo para comprender el código tanto por los lectores como por el propio desarrollador.

Referencias

* https://www.hasbro.com/es-es/customer-service/toy-game-instructions