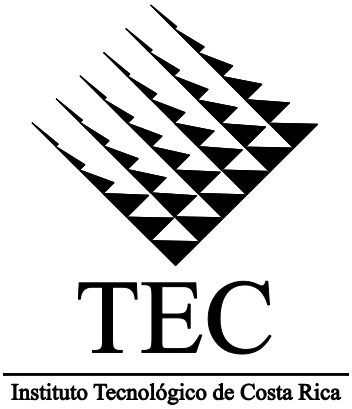
Instituto Tecnológico de Costa Rica



Escuela de Computación

TI14-01 Taller de programación

Tarea Programada #2

Profesor:

Andrei Fuentes

Estudiantes:

Juan Fabricio Soto Mejías
Benjamín Calvo de León.
José Andrés Hernández S.
Carlos Alberto Campos F.

I semestre 2012

Tabla de Contenidos

Tabla de Contenidos 2
Descripción del problema 3
Librerías Usadas 3
Análisis de Resultados: 4
Escenarios de Prueba: 4
Conclusiones Personales 5
Anexos: 6

Descripción del problema

Con respecto a la elaboración del juego del laberinto, en distinción de un simple juego, este debe tener la capacidad de generar laberintos de diferentes tamaños según lo decida o seleccione el usuario, además las opciones permitirán escoger si se quiere buscar la solución de modo usuario o automático. En el modo usuario, se iniciara el laberinto a partir de un archivo txt, el mismo se ira recorriendo de forma manual para encontrar el resultado mediante el uso de las teclas; por otro lado en la opción automática, el programa deberá generar matrices de manera random. En ambas deberá existir la opción de solucionar, la cual permitirá que el usuario al presionarla se active el solucionador del Laberinto. Lo que se pretende es mediante la librería pygame de python elaborar una serie de funciones que elabore el juego de acuerdo a lo solicitado por el usuario. Y de forma aleatoria, tendrá que indicar cuando se ha encontrado la solución. Todo laberinto sin excepción se le mostrara al usuario teniendo salida con una llegada a la solución definida.

Librerías Usadas

Para la resolución de la tarea decidimos utilizar tres librerías: Pygame, Random y Tkinter. La librería de mayor importancia en este proyecto es, sin lugar a dudas, es la librería de Pygame. Pygame es un módulo para el lenguaje de programación Python en el cual se crean juegos mucho más eficientes y complejos que con la herramienta Tkinter. Decidimos diseñar nuestro juego de laberinto en esta herramienta ya que podría decirse que dominamos la librería Tkinter, (cuya dificultad es prácticamente mínima) y nos llamaba la atención las posibilidades de usar dicha herramienta para la creación de los mismos. Para la generación de laberintos automáticos se requirió de una pieza muy importante de la librería Random: Randomint. Gracias a dicha utilidad logramos hacer laberintos casi cien por ciento realizables todas las veces que era llamado el algoritmo de generación automática de mapas. Después de cuatro días y medio de leer la documentación, programas y tutoriales logramos crear nuestro laberinto de forma completamente autónoma, sin necesidad de ingresar algún dato que no sea el de elección al modo automático. Después de muchos intentos fallidos e investigaciones infructuosas, no logramos realizar con efectividad la colisión entre Sprites, así que nos vimos en

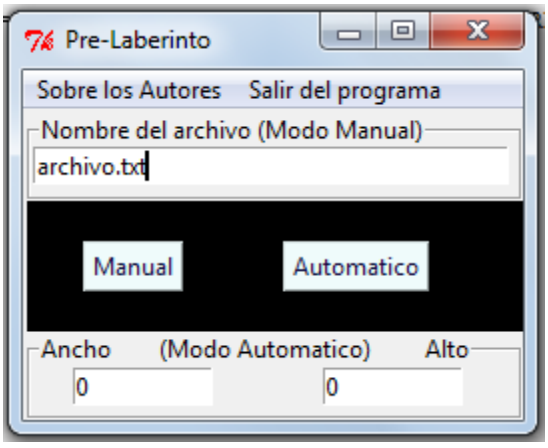
la necesidad de crear rectángulos detrás de los sprites para determinar las colisiones por medio de Rects. Otro pequeño problema, a los cuales llamamos como “los poderes de yoda” fue el de poder presionas las teclas de movimiento horizontal y vertical (w,a,s,d) al mismo tiempo que las flechas, lograba el poder movilizarse en medio de dos paredes diagonalmente. El segundo “poder” (que es en realidad un pequeño bug) es el de mover el mouse sobre la superficie al mismo tiempo que se presionaba una tecla de movimiento. Esto causaba que el jugador se pudiera desplazar en medio de las paredes, sin detectar colisiones; sin embargo, si movimientos. Esto se puede solucionar removiendo el cursor o eliminando la opción de (w,a,s,d), sin embargo, decidimos dejarlo para respetar “el poder de yoda”.

Análisis de Resultados:

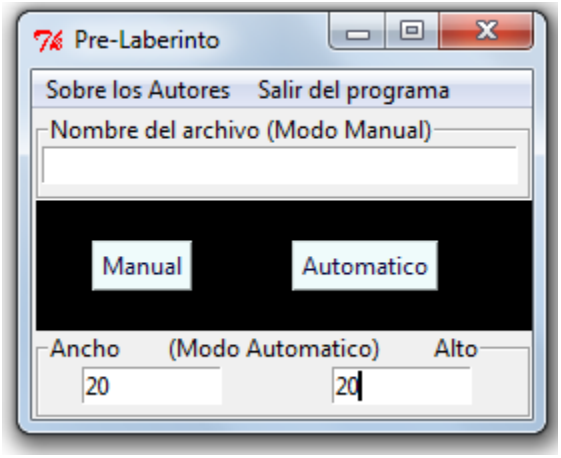
De acuerdo a los escenarios de prueba llevados a cabo, el programa cumple con las expectativas esperadas, tanto por parte de los usuarios como de los creadores, de forma que el modo usuario y automático trabajan de manera eficiente. En cuanto a la opción de solucionar “Backtracking”, logramos implementarla y hacer que funcionara.

Escenarios de Prueba:

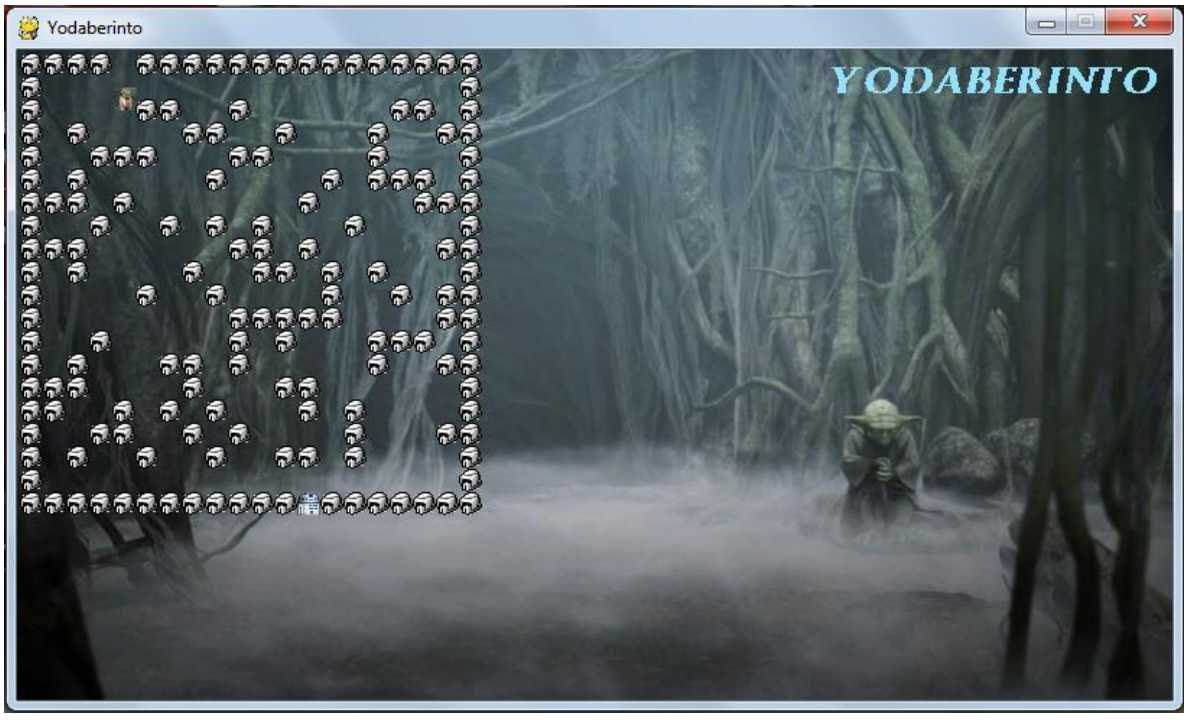
1. En este primer ejemplo utilizamos la lectura de un archivo.txt para generar el juego, la cual contiene una matriz (De forma manual).



2. En el segundo generamos el laberinto de manera manual, en otras palabras, con la función que existe dentro del código para generar matrices.



En ambos genero el siguiente resultado:



Conclusiones Personales

Definitivamente se puede afirmar que aprendimos la lección. Se necesita de un mejor planeamiento y definitivamente más información sobre cómo usar las

librerías de Python. Debemos admitir que fue un verdadero dolor de cabeza que en un par de días se avanzara tan poco y terminara en frustración a las cuatro de la mañana del día de entrega. Fue verdaderamente reconfortante que el laberinto funcionara de una forma estética y agradable a la vista (al menos según mi opinión y la de mis compañeros) y un poco desilusionante el hecho de no poder integrar a la interfaz gráfica el modo automático y el modo usuario, así como el poder preguntar al usuario si deseaba extraer un txt con una matriz. Realmente nos sentimos decepcionados… Sin embargo, estamos orgullosos del trabajo que realizamos, pues fue verdaderamente gratificante cuando se lograba avanzar; también lo estamos de haber creado nuestro primer juego programado por nosotros mismos.

Anexos:

Manual de Usuario para Yodaberinto.