

Documento de análisis del proyecto

Juegos: Tatetí-Ahorcado



6 de julio de 2021

Sofía sanabria y manuel buslón

Índice

[1 Planteamiento del problema 2](#_Toc76245137)

[2 Análisis de soluciones alternativas 2](#_Toc76245138)

[2.1 Ahorcado 2](#_Toc76245139)

[2.1.1 Para el problema de la lista de palabras 2](#_Toc76245140)

[2.1.2 Para esconder la palabra 2](#_Toc76245141)

[2.1.3 Para imprimir el estado de la horca 3](#_Toc76245142)

[2.2 Ta Te Ti 3](#_Toc76245143)

[2.2.1 Guardar el estado actual 3](#_Toc76245144)

[2.2.2 Finalizar partida 3](#_Toc76245145)

[3 Justificación de la solución elegida 4](#_Toc76245146)

[4 Desarrollo de la solución 4](#_Toc76245147)

[5 Conclusiones 4](#_Toc76245148)

# Planteamiento del problema

# Análisis de soluciones alternativas

## Ahorcado

### Para el problema de la lista de palabras

* Para solucionar el problema se puede implementar directamente una lista de palabras sobre la cual el programa elija aleatoriamente una, otra forma de hacerlo podría ser implementar varias listas permitiendo elegir al usuario que lista de palabras quiere usar y a partir de ella elegir una.
* Otras formas tendrían en cuenta el uso de archivos de texto con las palabras, en estos casos una forma sería utilizar etiquetas que definan una sección de palabras que el programa cargará en una lista o tener una carpeta que contenga los archivos de texto representando las categorías con sus palabras dentro.

### Para esconder la palabra

* Como primera opción a partir de la palabra seleccionada se puede imprimir el carácter que lo esconde por la cantidad de letras que tenga la palabra, para luego ir reemplazando los caracteres por la letra acertada en su respectivo lugar.
* Una solución diferente sería almacenar las letras de la palabra y las letras descubiertas en listas, para luego reemplazar las letras no descubiertas de las palabras por el carácter que las esconde.
* Para solucionar el problema de colocar el guion bajo y que este se confunda formando solo una raya, se coloca un espacio entre cada letra.

### Para imprimir el estado de la horca

* Se puede plantear imprimir línea por línea el dibujo de la horca y agregar cada parte del cuerpo a las líneas que correspondan dependiendo de la cantidad de errores.
* Una manera un poco extraña sería imprimir texto con el estado de la horca, el problema de esta opción es que es algo perturbadora e incomoda de imaginar al pensar que describe que aparece cada parte con los errores.
* Otra opción es almacenar cada parte del cuerpo en una lista y agregarlas a otra a medida que se van obteniendo errores.

## Ta Te Ti

### Guardar el estado actual

* La forma en la que se guardará el estado del tablero puede estar representada de distintas formas, como: una lista con la ficha que pertenece a esa posición, otra lista que tenga tres elementos y a su vez otros tres elementos.
* Un diccionario que tenga como claves las coordenadas en el tablero y como valor el elemento que está en ese lugar.

### Finalizar partida

* Para ello se puede recorrer cada una de las filas, columnas y diagonales para ver si coinciden o es un empate.
* Comparar directamente los valores de las posiciones que se necesitan para ganar y si ninguna de estas coincide y no están vacíos es un empate.

# Justificación de la solución elegida

## Ahorcado

* Se optó por hacer una carpeta con archivos de texto con cada categoría como nombre de este, es más limpia en cuanto a código y organización, ya que cuando aumenta el número de palabras disminuye la legibilidad del código.
* La forma elegida de ocultar la palabra es óptima en cuanto a eficiencia, ya que coloca un guion bajo por cada letra oculta junto a un espacio para evitar confusiones a la hora de ver la ubicación de las letras.
* Elegimos la opción de tener la horca prediseñada para ir agregando cada parte del cuerpo según cuántos errores se cometen, las partes del cuerpo están en una lista.

## Ta Te Ti

* Se eligió guardarlo en un diccionario ya que es más fácil para acceder de esta manera.
* Para saber si la partida finalizó se recorre y compara cada casilla utilizando estructuras de repetición.

# Desarrollo de la solución

# Conclusiones