Tecnológico de Costa Rica

Escuela de Computación

Taller de Programación

Tercer Proyecto de Programación

Documentación del Proyecto

Prof. William Mata

Carlos Daniel Calderon Salazar

2019222174

Grupo 3

Martes 26 de Noviembre, 2019

Tabla de contenido

Enunciado del Proyecto
OBJETIVOS DE LA TAREA

- Reforzar los conocimientos acerca de la metodología de desarrollo de programas de mayor escala: o Entender el problema. o Diseñar la solución: dividir el problema en problemas más pequeños. o Codificar la solución. o Probar y evaluar el programa.

- Aplicar y reforzar aspectos del lenguaje Python 3 o Uso de estructuras condicionales y de repetición de procesos. o Desarrollo de funciones. o Utilización de archivos. o Utilización de estructuras de datos nativas de Python y TDA (Tipos de Datos Abstractos). o Uso de clases (POO) en la funcionalidad de los jugadores del Top-10.

- Ampliar el conocimiento acerca del desarrollo de GUI en Python.

- Aplicar buenas prácticas de programación: documentación interna y externa del programa, reutilización de código, nombres significativos, eficiencia del programa, evaluar alternativas, uso de técnica divide y vencerás (dividir el problema en partes, desarrollar cada una de esas partes), etc.

- Usar algún software de control de versiones de software, por ejemplo Git o algún otro que usted decida. En los próximos días se dará un taller sobre este tipo de software.

- Validación de los datos de entrada: todos los datos de entrada se deben validar según restricciones que se indican en cada uno de ellos.

- Fomentar en el estudiante la investigación: temas no tratados en el curso pero necesarios para hacer el proyecto. Dichos temas deben ser explicados detalladamente en la documentación del proyecto.

DEFINICIÓN DEL PROYECTO: SUDOKU

Sudoku es un pasatiempo que se popularizó internacionalmente después del año 2000 cuando diversos periódicos empezaron a publicarlo en la sección de pasatiempos.

Consiste en llenar con dígitos del 1 al 9 cada una de las casillas de una cuadrícula que tiene un total de 9 x 9 casillas. La cuadrícula la podemos ver como una matriz de 9 filas y 9 columnas. A la vez esta cuadrícula se divide en subcuadrículas de 3 x 3 casillas. Se toman como base algunos dígitos fijos en la cuadrícula, lo cual determina el nivel de dificultad para completar el juego. Aunque usualmente se usan dígitos, lo que interesa es que sean grupos de 9 elementos diferentes: por ejemplo 9 letras, 9 colores, 9 frutas, etc.

La regla de este juego es que un mismo elemento no se puede repetir: - En una misma fila (casillas horizontales) - En una misma columna (casillas verticales) - En las subcuadrículas (3x3)

Además, los elementos que aparecen al inicio del juego quedan fijos, no se pueden cambiar.

Ejemplo de un Sudoku: ya tiene los dígitos preestablecidos para empezar a jugar.

¿ Se puede llenar la casilla de la fila 3 columna 1 con un 4 ? ¿ Por qué ? ¿ Se puede llenar la casilla de la fila 4 columna 7 con un 8 ? ¿ Por qué ? ¿ Se puede llenar la casilla de la fila 1 columna 6 con un 9 ? ¿ Por qué ? ¿ Se puede llenar la casilla de la fila 9 columna 5 con un 6 ? ¿ Por qué ?

El programa tendrá un menú principal desde el cual se accederá la funcionalidad del programa, es decir, lo que el programa hace. Usted puede agregar otras funcionalidades que vayan a mejorar el producto. En la interfaz gráfica ponga atención a los diferentes elementos como son los tamaños de letras, colores, formas, menús, botones, cuadros de texto, etc. Puede hacer cambios a la interfaz siempre y cuando cumpla con los requerimientos del programa que se indican seguidamente.

REQUERIMIENTOS DEL PROGRAMA

A) Jugar

Esta opción permite jugar el Sudoku. Cuando se da esta opción se muestra una pantalla como la siguiente según la configuración y las partidas registradas para los diferentes niveles de dificultad en las otras opciones del menú: Nivel del juego, si va a usar el reloj y los elementos para llenar la cuadrícula.

El programa tiene una serie de partidas que previamente han sido registradas y de ahí selecciona aleatoriamente una partida según el nivel de dificultad configurado. Python tiene funciones para generar números aleatorios que pueden servir para seleccionar alguna de las partidas de tal forma que siempre se elija una al azar. Puede usar otros algoritmos para esta selección aleatoria de partidas. Documente cuál algoritmo de selección aleatoria usó. En una misma corrida del programa, si hay n partidas para un nivel, primero se deben escoger las n partidas aleatorias antes de volver a repetirlas.

Uso de los botones:

Cuando el jugador pica este botón se inicia el juego. Una vez que se inicia el juego, el jugador selecciona un elemento picándolo en el panel de elementos y luego pica en la casilla de la cuadrícula en donde quiere ponerlo. Cuando pica el elemento se señala con una flecha como se muestra en el ejemplo, y cuando lo coloca la flecha desaparece. Si quiere seleccionar otro elemento lo pica para ponerlo en donde sea permitido. Cuando el jugador pone un elemento se deben hacer las validaciones para que la jugada cumpla con las reglas del juego, de lo contrario se le envía alguno de estos mensajes:

- JUGADA NO ES VÁLIDA PORQUE EL ELEMENTO YA ESTÁ EN LA FILA - JUGADA NO ES VÁLIDA PORQUE EL ELEMENTO YA ESTÁ EN LA COLUMNA - JUGADA NO ES VÁLIDA PORQUE EL ELEMENTO YA ESTÁ EN LA CUADRÍCULA - JUGADA NO ES VÁLIDA PORQUE ESTE ES UN ELEMENTO FIJO En estos casos ponga también en modo de parpadeo (blinking) el elemento que ya está en el juego para que el jugador lo identifique de inmediato. Cuando el jugador continúe (da <ENTER> u otra tecla después de ver el error) quitamos el modo de parpadeo de ese elemento.

El juego termina cuando el jugador llena todas las casillas de la cuadrícula, ahí para el reloj o el timer y despliega el mensaje ¡ EXCELENTE ! JUEGO COMPLETADO.

En este momento debe determinar si este jugador debe registrarlo en el Top 10. El Top 10 es un archivo donde el programa registra las mejores 10 marcas por cada nivel de dificultad (los jugadores que tarden menos en completar el juego). Si tenemos las 10 marcas y el jugador actual hace un mejor tiempo que esas marcas, hay que eliminar la marca con mayor

INICIAR JUEGO

tiempo para seguir teniendo un máximo de 10 marcas por nivel. La marca es un TDA que contiene el nombre y el tiempo (horas, minutos, segundos) que un jugador tardó en completar un juego. Note que si usa el timer hay que calcular la duración del juego. Cuando un juego es completado el programa regresa a la opción de Jugar.

Otras consideraciones: - Antes de iniciar el juego el jugador debe dar un nombre (string de 30 caracteres máximo). - Luego de dar el botón INICIAR JUEGO, este botón se deshabilita. - En caso de haber configurado la opción de Timer, el jugador puede dejar el tiempo configurado o modificarlo antes de INICIAR JUEGO. El tiempo empieza a correr cuando le den INICIAR JUEGO. - En caso de no usar el reloj o el timer no debe aparecer esa parte en la pantalla. Para el uso del timer alguna de sus partes (horas, minutos, segundos) debe ser mayor a cero. - En caso de haber configurado la opción de Timer y éste llegue a 0 y el juego no haya terminado se envía el mensaje TIEMPO EXPIRADO. ¿ DESEA CONTINUAR EL MISMO JUEGO (SI O NO) ?. Si responde SI entonces el timer pasa a ser reloj inicializado con el tiempo que se había establecido en el timer. Por ejemplo si el timer estaba para 1 hora y 30 minutos, ahora el reloj debe marcar que ya ha pasado 1 hora y 30 minutos y sigue contando el tiempo. Si responde NO el juego finaliza regresando a la opción de Jugar. - En caso de usar el reloj las horas pueden llegar hasta 23, los minutos entre 0 y 59 y los segundos entre 0 y 59. Hay que realizar estas validaciones. - En caso de no existir alguna partida para el nivel seleccionado se da el mensaje DEBE GRABAR AL MENOS UNA PARTIDA DE ESTE NIVEL. Luego lo envía al menú principal. - En caso de picar una casilla y no haya seleccionado previamente un elemento se envía el error FALTA QUE SELECCIONE EL ELEMENTO.

En caso de que ocurra un error el programa enviará un mensaje de error. El error se mantendrá en pantalla hasta que el usuario le indique al programa que continúe para que pueda corregir la situación.

Elimina la última jugada dejando la casilla vacía. Puede borrar todas las jugadas que ha hecho, por tanto use una pila de jugadas. Sugerencia: use un TDA implementando una pila donde cada elemento va a contener la fila y columna en donde se jugó. Cada vez que se hace una jugada se agrega a la pila, y si seleccionan este botón, se toma la última jugada agregada en la pila (el manejo de la pila es tipo LIFO: Last In First Out, último en entrar primero en salir) y se borra la casilla respectiva de la cuadrícula.

Otras consideraciones: - En caso de que ya no hayan mas jugadas para borrar según la pila hay que enviar el mensaje NO HAY MAS JUGADAS PARA BORRAR. - Se puede seleccionar esta opción solamente si el juego ha iniciado de lo contrario hay que enviar el mensaje NO SE HA INICIADO EL JUEGO. - La pila de jugadas realizadas se reinicia en cada juego.

Cuando el jugador selecciona esta opción se le pregunta

¿ ESTA SEGURO DE TERMINAR EL JUEGO (SI o NO) ?

Si responde SI termina de inmediato el juego y se vuelve a mostrar otro juego como si estuviera entrando a la opción de Jugar .

Si responde NO sigue jugando con el mismo juego.

Se puede seleccionar esta opción solamente si el juego ha iniciado de lo contrario hay que enviar el mensaje NO SE HA INICIADO EL JUEGO.

BORRAR JUGADA

TERMINAR JUEGO

Cuando el jugador selecciona esta opción se le pregunta

¿ ESTA SEGURO DE BORRAR EL JUEGO (SI o NO) ?

Si responde SI vuelve a la opción de Jugar usando la misma partida pero eliminando todas las jugadas que hizo.

Si responde NO sigue jugando con el mismo juego.

Se puede seleccionar esta opción solamente si el juego ha iniciado de lo contrario hay que enviar el mensaje NO SE HA INICIADO EL JUEGO.

Esta opción se puede usar en cualquier momento. Detiene el reloj si se esta usando. Despliega una sola pantalla con los records de los mejores 10 primeros jugadores por cada nivel: aquellos que hicieron menos tiempo para completar el juego. En caso de no tener los 10 jugadores en algún nivel se despliegan los que se tengan. El Top 10 se guarda en el archivo “sudoku2019top10.dat”. Represente cada jugador como un objeto (clase de POO) y tenga 3 listas de objetos, una por cada nivel de dificultad. Estas listas se guardan en ese archivo. Al menos use métodos para establecer los valores de los objetos y para obtenerlos.

TOP 10

NIVEL DIFÍCIL: JUGADOR TIEMPO 1- Nombre jugador 1:30:15 2- Nombre jugador 1:32:55 … 10-

BORRAR JUEGO

TOP 10

NIVEL INTERMEDIO: 1- Nombre jugador 1:10:21 2- Nombre jugador 1:35:55 … 10-

NIVEL FÁCIL:

1- Nombre jugador 0:7:23 2- Nombre jugador 0:10:55 … 10-

Luego de que el usuario vea esta información el programa regresa a donde estaba jugando y sigue el conteo en el reloj.

Este botón se puede usar en cualquier momento que el juego haya iniciado. Guarda en el archivo “sudoku2019juegoactual.dat” el juego actual (cuadrícula) con

su configuración (nivel, uso del reloj, etc.) y nombre del jugador El objetivo es que el jugador pueda en cualquier momento guardar el juego y posteriormente continuarlo. Este archivo solo va a contener una partida. En caso de que haya una partida en el archivo, se borra y se guarda la del momento.

Este botón se puede usar solamente cuando un juego no se haya iniciado. Trae del archivo “sudoku2019juegoactual.dat” el juego que tenga con su configuración y lo pone en la pantalla como el juego actual. El juego continúa cuando el jugador usa el botón de INICIAR JUEGO.

GUARDAR JUEGO

CARGAR JUEGO

B) Configurar

Esta opción es para indicar las condiciones con que se va a jugar. Contiene los siguientes datos que se van a guardar en el archivo “sudoku2019configuración.dat” : (los valores por omisión –o default- están señalados con el círculo en rojo)

1. Nivel: o o o Fácil o Intermedio o Difícil

2. Reloj: o o o Si o No o Timer

En caso de seleccionar Timer puede poner los siguientes tiempos como una recomendación, pueden ser modificados por el jugador en esta opción o en la de Jugar:

Para el nivel fácil: 30 minutos Para el nivel intermedio: 1hora Para el nivel difícil: 2 horas

Para el timer las horas pueden estar entre 0 y 4, los minutos entre 0 y 59 y los segundos entre 0 y 59. El timer debe tener al menos uno de estos valores. Hay que realizar estas validaciones y enviar los mensajes respectivos en caso de errores.

Horas Minutos Segundos

3. Panel de elementos para llenar la cuadrícula:

C) Partidas

Grabe las partidas usadas por el juego en el archivo “sudoku2019partidas.dat”.

Las partidas se van a grabar en tres diccionarios: - Diccionario de partidas de nivel fácil - Diccionario de partidas de nivel intermedio - Diccionario de partidas de nivel difícil

Estructura de los elementos de cada diccionario: LLAVE: número de partida (entero) VALOR: partida

Cada partida es una lista con 9 sublistas, y cada sublista contiene 9 elementos. Las sublistas representan las filas y los elementos de cada sublista las columnas.

Con esta partida de ejemplo de nivel fácil tendríamos lo siguiente:

diccionario de partidas de nivel fácil: { 1: [# partida número 1 [“5”, “3”, “”, “”, “7”, “”, “”, “”, “”], [“6”, “”, “”, “1”, “9”, “5”, “”, “”, “”], [“”, “9”, “8”, “”, “”, “”, “”, “6”, “”], [“8”, “”, “”, “”, “6”, “”, “”, “”, “3”], [“4”, “”, “”, “8”, “” , “3”, “”, “”, “1”], [“7”, “”, “”, “”, “2”, “”, “”, “”, “6”], [“”, “6”, “”, “”, “”, “”, “2”, “8”, “”], [“”, “”, “”, “4”, “1”, “9”, “”, “”, “5”], [“”, “”, “”, “”, “8”, “”, “”, “7”, “9”]], …, n: [… # partida número n] }

Busque algunas partidas (al menos 3 por cada nivel) y grábelas directamente en el archivo.

D) Ayuda

Esta opción desplegará el manual del usuario.

E) Acerca de

Esta opción la usaremos para desplegar información “Acerca del programa” donde pondremos al menos los datos del nombre del programa, la versión, la fecha de creación y el autor.

F) Salir

Esta opción se usa para salir del programa.

DOCUMENTACIÓN DEL PROYECTO

REQUISITOS PARA REVISAR EL PROYECTO a- El programa debe tener documentación interna. b- Para el desarrollo del proyecto debe usar un software de control de versiones. c- La nota de la documentación del proyecto, indicada abajo, sirve para aceptar o rechazar el proyecto: se revisan los proyectos que cumplan con esa documentación en un 90% o más. d- El programa debe usar una interfaz tipo GUI.

Enviar vía tecDigital, sección EVALUACIONES / PROGRAMAS, una carpeta comprimida (.rar, .zip, etc.) de nombre programa3_su_nombre que contenga las siguientes partes:

o Parte 1: Documentación del proyecto (nombre: documentación_sudoku.PDF). ▪ Portada. (1 p) ▪ Contenido. (2 p) ▪ Enunciado del proyecto. (2 p) ▪ Temas investigados (material no estudiado en el curso). (15 p) • Por cada uno de estos temas debe poner el marco teórico: de qué trata, cómo se usa. ▪ Diseño y explicación de la solución: estructuras de datos, archivos y otros aspectos importantes de la solución. (25 p). ▪ Conclusiones del trabajo: (15 p) • Problemas encontrados y soluciones a los mismos. • Aprendizajes obtenidos. ▪ Estadística de tiempos: un cuadro que muestre el detalle de las actividades que realizó y las horas invertidas en cada una de ellas. La estadística permite medir el esfuerzo dedicado al trabajo en términos de actividades y tiempos, lo cual puede ser una base para calcular el esfuerzo requerido en futuros trabajos. (5 p)

Ejemplos de actividades: Actividad Realizada Horas Análisis del problema Diseño de algoritmos Investigación de ... Programación Documentación interna Pruebas Elaboración del manual de usuario Elaboración de documentación del proyecto

Etc. TOTAL

▪ Rúbrica de evaluación y análisis de resultados (PONGA LA HOJA DE LA RÚBRICA EN PÁGINA NUEVA). (15 p) • Tome la rúbrica de evaluación y por cada concepto calificado Usted debe indicar el % de avance y el análisis de resultados de su proyecto. o 100: Totalmente desarrollado. No hace falta análisis excepto que requiera hacer alguna observación. o Un % específico, por ejemplo 80 significaría un desarrollo parcial del 80%. En el análisis indicar tres partes: ¿qué hace?, ¿qué falta?, ¿ por qué no se completó ? o 0: No desarrollado. En el análisis indicar el motivo. • Partes que desarrolló adicionales a los requerimientos.

Concepto Puntos Puntos obtenidos

Avance 100/%/0

Análisis de resultados

Menú 1 Opción Jugar (despliegue del juegol) 15 Botón Iniciar Juego (incluye creación del archivo del Top 10) 10 Botón Borrar Jugada 5 Botón Terminar Juego 2 Botón Borrar Juego 5 Botón Top 10 10 Botón Guardar Juego 5 Botón Cargar Juego (incluye el despliegue del mismo) 15 Opción Configurar 10 Ayuda (manual de usuario) 10 Acerca de 1 Salir 1 Reloj y Timer tiempo real 10 TOTAL 100 Partes desarrolladas adicionalmente

▪ Manual de usuario (nombre: manual_de_usuario_sudoku.). (20 p)

o Parte 2: Programa fuente (nombre: sudoku.py) y los objetos necesarios para ejecutar el programa.

IMPORTANTE: CONOCIMIENTO DE LA SOLUCIÓN PRESENTADA. En la revisión del trabajo, el estudiante debe demostrar un completo dominio de la solución que implementó, tanto desde el punto de vista técnico (uso de Python) como de la funcionalidad del programa. La revisión se puede hacer individualmente o en grupos, examinando el programa o temas específicos aplicados en el programa. Última línea

"… mira con optimismo el estudio que estás haciendo.

Estás aquí porque te estás formando para la vida.

Estás entrenando tu cerebro y tu inteligencia para ser una persona de bien que aporte muchas cosas a una sociedad actual carente de muchos valores. Pon

energía y entusiasmo que el estudio puede ser pesado, pero encontrarás muchos beneficios con tus logros alcanzados.

DIOS nos siga bendiciendo …”

Temas Investigados

POO

La programación orientada a objetos (POO, u OOP según sus siglas en inglés) es un paradigma de programación que viene a innovar la forma de obtener resultados. Los objetos manipulan los datos de entrada para la obtención de datos de salida específicos, donde cada objeto ofrece una funcionalidad especial.

Muchos de los objetos prediseñados de los lenguajes de programación actuales permiten la agrupación en bibliotecas o librerías, sin embargo, muchos de estos lenguajes permiten al usuario la creación de sus propias bibliotecas.

Está basada en varias técnicas, como las siguientes:

- herencia.
- cohesión.
- abstracción.
- polimorfismo.
- acoplamiento.
- encapsulación.

La POO tiene sus raíces en la década del 60 con el lenguaje de programación Simula que en 1967, el cual fue el primer lenguaje que posee las características principales de un lenguaje orientado a objetos.

Smalltalk (de 1972 a 1980) es posiblemente el ejemplo canónico, y con el que gran parte de la teoría de la POO se ha desarrollado. Más su uso se popularizó a principios de la década de 1990.

En la actualidad, existe una gran variedad de lenguajes de programación que soportan la orientación a objetos.

Los objetivos de la POO son:

- Organizar el código fuente
- Re-usar código fuente en similares contextos.

POO en Python

El mecanismo de clases de Python agrega clases al lenguaje con un mínimo de nuevas sintaxis y semánticas.

En Python las clases es una mezcla de los mecanismos de clase encontrados en C++ y Modula-3.

Como es cierto para los módulos, las clases en Python no ponen una barrera absoluta entre la definición y el usuario, sino que más bien se apoya en la cortesía del usuario de no “forzar la definición”.

Sin embargo, se mantiene el poder completo de las características más importantes de las clases: el mecanismo de la herencia de clases permite múltiples clases base, una clase derivada puede sobrescribir cualquier método de su(s) clase(s) base, y un método puede llamar al método de la clase base con el mismo nombre.

“Los objetos pueden tener una cantidad arbitraria de datos.”

En terminología de C++, todos los miembros de las clases (incluyendo los miembros de datos), son *públicos*, y todas las funciones miembro son *virtuales*.

Como en Modula-3, no hay atajos para hacer referencia a los miembros del objeto desde sus métodos: la función método se declara con un primer argumento explícito que representa al objeto, el cual se provee implícitamente por la llamada.

Como en Smalltalk, las clases mismas son objetos. Esto provee una semántica para importar y renombrar.

A diferencia de C++ y Modula-3, los tipos de datos integrados pueden usarse como clases base para que el usuario los extienda.

También, como en C++ pero a diferencia de Modula-3, la mayoría de los operadores integrados con sintaxis especial (operadores aritméticos, de subíndice, etc.) pueden ser redefinidos por instancias de la clase.

(Sin haber una terminología universalmente aceptada sobre clases, haré uso ocasional de términos de Smalltalk y C++. Usaría términos de Modula-3, ya que su semántica orientada a objetos es más cercana a Python que C++, pero no espero que muchos lectores hayan escuchado hablar de él).

Algunas particularidades de POO en Python son las siguientes:

- Todo es un objeto, incluyendo los tipos y clases.
- Permite herencia múltiple.
- No existen métodos ni atributos privados.
- Los atributos pueden ser modificados directamente.
- Permite “monkey patching”.
- Permite “duck typing”.
- Permite la sobrecarga de operadores.
- Permite la creación de nuevos tipos de datos.

Función .trace

El trace módulo le permite rastrear la ejecución del programa, generar listados anotados de cobertura de estados de cuenta, imprimir las relaciones entre la persona que llama y la persona que llama y enumerar las funciones ejecutadas durante la ejecución de un programa. Se puede usar en otro programa o desde la línea de comandos.

Método After

Tkinter proporciona una variedad de funciones integradas que desarrollan una GUI (interfaz gráfica de usuario) interactiva y destacada. **after()**La función también es una función universal que se puede utilizar directamente en la raíz, así como con otros widgets.

Parámetros:
parent *: es el objeto del widget o ventana principal, cualquiera que esté usando esta función.*
ms *: es el tiempo en milisegundos.*
función *: que se llamará.*
**** args*** *: otras opciones.*

Widget topLevel

Los widgets de Toplevel funcionan como ventanas administradas directamente por el administrador de ventanas. No necesariamente tienen un widget principal encima de ellos.

Su aplicación puede usar cualquier cantidad de ventanas de nivel superior.

Parámetros

- **opciones** : aquí está la lista de las opciones más utilizadas para este widget. Estas opciones se pueden usar como pares clave-valor separados por comas.

No Señor.	Opción y descripción
1	**bg** El color de fondo de la ventana.
2	**bd**

	Ancho del borde en píxeles; el valor predeterminado es 0.
3	**cursor** El cursor que aparece cuando el mouse está en esta ventana.
4	**clase_** Normalmente, el texto seleccionado dentro de un widget de texto se exporta para ser la selección en el administrador de ventanas. Establezca exportselection = 0 si no desea ese comportamiento.
5	**fuente** La fuente predeterminada para el texto insertado en el widget.
6	**fg** El color utilizado para el texto (y mapas de bits) dentro del widget. Puede cambiar el color de las regiones etiquetadas; Esta opción es solo la predeterminada.
7	**altura** Altura de ventana.
8	**alivio** Normalmente, una ventana de nivel superior no tendrá bordes tridimensionales a su alrededor. Para obtener un borde sombreado, configure la opción bd más grande que su valor predeterminado de cero, y establezca la opción de relieve en una de las constantes.
9	**anchura** El ancho deseado de la ventana.

Diseño y explicación de Soluciones

Dentro del programa sinceramente no tuve mucha dificultad en desarrollarlo ya que de pequeño solía resolverlos con mi abuelo por lo que la lógica, la definición, el juego y la técnica ya los tenía aprendidos y solo me hacía falta llevarlos a cabo.

Se hizo larga la programación principalmente al tener que hacer 4 tipos diferentes de partidas por lo que aunque casi todas llevaran la misma lógica, algo les terminaba cambiando. El poder ensamblar todo, sin que tuviera pulgas o errores era complicado ya que había que hacer mil pruebas por cada tipo de juego para estar seguro que nada se fuera a ir o que hubieran pulgas, la interfaz fue algo sencillo, no tan complicado pero bonito a plena vista, para que el usuario pueda tener momentos de relajación mientras juega, ya que una de las técnicas más importantes dentro del juego está en la paciencia y en ver que todo esté bien.

Conclusiones del Trabajo

Problemas Encontrados

Como lo dije antes, sinceramente no hubieron mayores problemas a la hora de realizar el proyecto, si soy sincero casi fueron nulos ya que todos iban desarrollándose de poco a poco con un poco de cabeza para cada planteamiento de alguna idea, en lo que sí puedo decir que duré tiempo haciendo y no por dificultad, sino por precisión fue en las validaciones por cuadrante del sudoku, ya que necesitaba calcular cada paso exacto para que no hubieran márgenes de error, por dicha todo se logró desarrollar y se logró plantear. Otro de los problemas el cual si fue bastante grande fue a la hora de guardar y cargar ya que el programa me estaba modificando las plantillas del juego por lo que a la hora de cargar el juego cuando leía alguna plantilla la colocaba con los números que el usuario había definido, se creaba un tipo de alias y aunque yo realizara el .copy() siempre me modificó las plantillas por lo que tuve que gastar un poco de memoria para hacer variables con valores exactamente iguales. Luego de esto el programa ya funcionó.

Aprendizajes aprendidos

Dentro de los aprendizajes que tuve estuvo el de aprender más sobre POO ya que esto me ayuda a avanzar para mis próximos cursos y gracias a esto ya supe cómo utilizarlo mejor. Otra de las cosas que aprendí y mejoré a nivel personal fue la agilidad de pensamiento y de análisis que tuve para este proyecto porque si lo mido en tiempos para este proyecto dediqué casi que menos de la mitad de tiempo que en otros y para mí en lo personal quedó mucho más limpio y bonito.

Estadísticas de tiempo

Actividad Realizada	Horas
Análisis del problema	5
Diseño de algoritmos	5
Investigación de…	5
Programación	72
Documentación Interna	3
Pruebas	7
Elaboración del manual de usuario	1
Elaboración de la documentación del proyecto	2
Etc.	4
Total	104 (4/5 días aprox)

Rúbrica de evaluación y análisis de resultados

Concepto	Puntos	Puntos obtenidos	Avance 100/%/0	Análisis de resultados
Menú	1	1	100	
Opción Jugar (despliegue del juego)	15	15	100	
Botón Iniciar Juego (incluye creación del archivo del Top 10)	10	10	100	
Botón Borrar Jugada	5	5	100	
Botón Terminar Juego	2	2	100	
Botón Borrar Juego	5	5	100	
Botón Top 10	10	10	100	
Botón Guardar Juego	5	5	10	
Botón Cargar Juego (incluye el despliegue del mismo)	15	15	100	
Opción Configurar	10	8	95	Por tiempo me faltó la configuración especial
Ayuda (Manual de usuario)	10	10	100	
Acerca de	1	1	100	
Salir	1	1	100	
Reloj y Timer tiempo real	10	10	100	
Total	**100**	98		
Partes desarrolladas adicionalmente		4		