**Introducción a la Programación**

**I Semestre – 2022**

**Integrantes:**

Samuel Garcés Castillo – 2022129139

Carlos Daniel Guzmán Ramírez – 2022437782

**Tarea Programada #1**

**Profesora:**

Laura Guiselle Coto Sarmiento

**Tutores:**

Daniel Araya Sambucci

**Fecha de entrega:**

Jueves 28 de abril

**Estatus de la entrega:**

**Índice**

Enunciado del proyecto 3

Justificación de eliminación de olores 3

Conclusiones del trabajo 5

Reglamento de trabajo 6

Agendas y Minutas 7

Cronograma 9

Estadísticas de tiempos 10

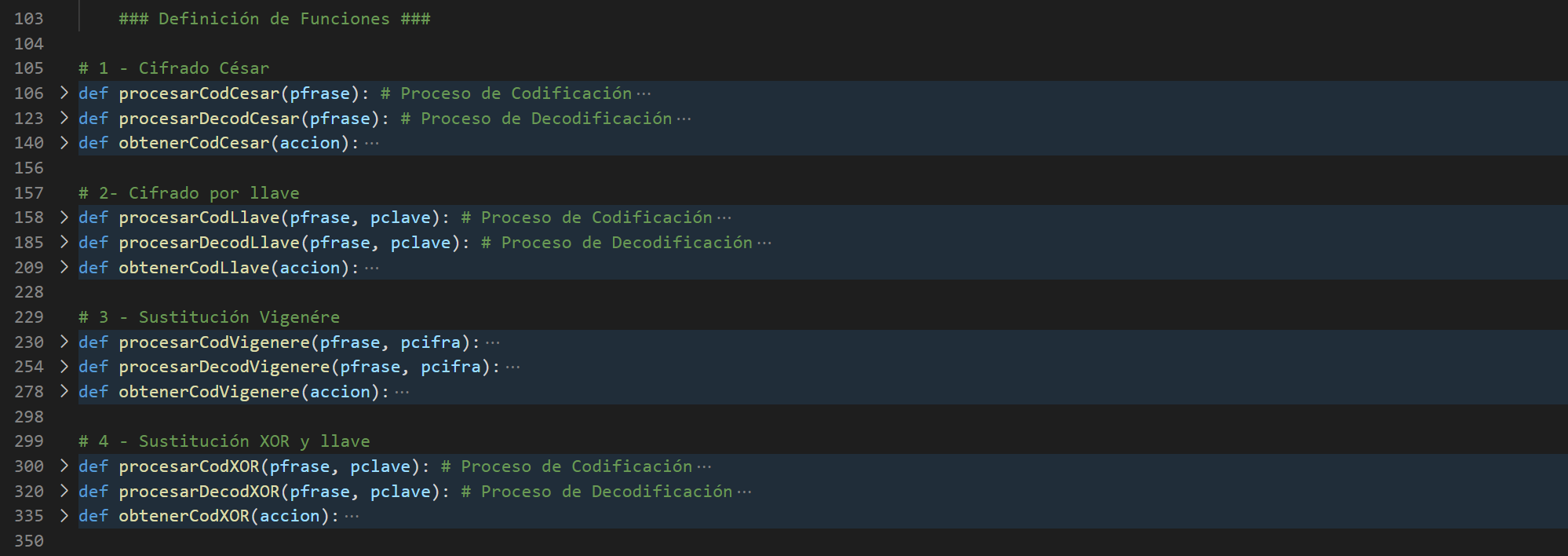
**Enunciado del proyecto:**

Este trabajo fue realizado bajo las especificaciones indicadas en el documento “TP1\_Criptografía\_2022.pdf”, el cual fue entregado y consultado a través de la plataforma del TEC Digital, en el apartado de “Documentos públicos de Taller de Programación GR 2”. Puede consultarse dicha información a través del siguiente enlace:

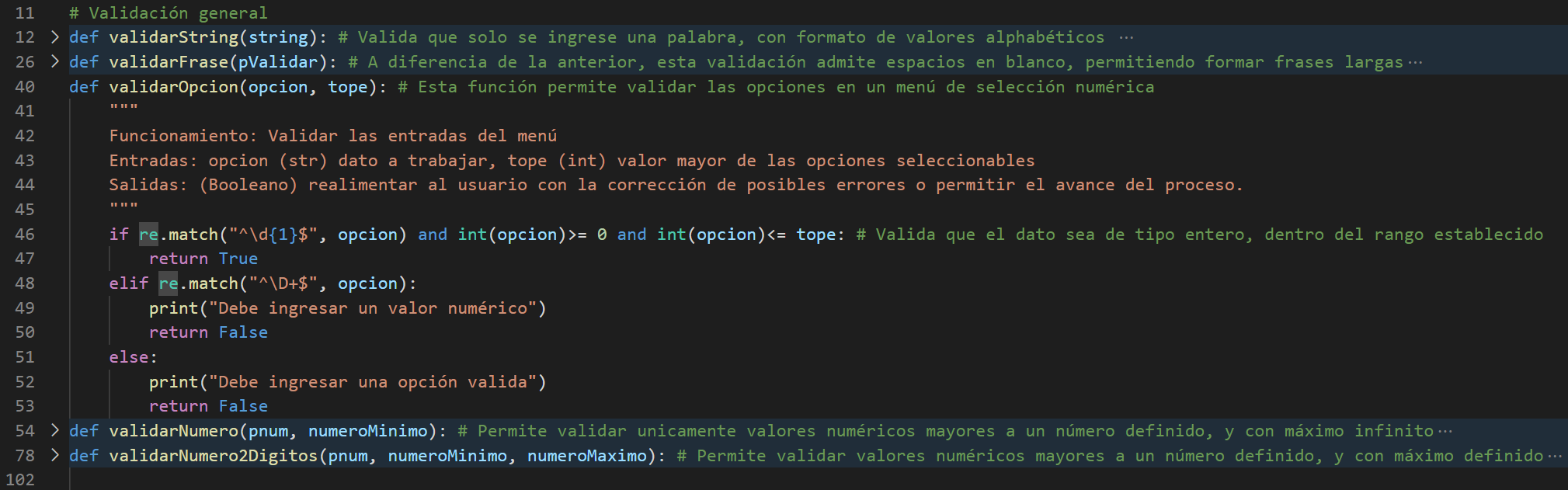
<https://tecdigital.tec.ac.cr/dotlrn/classes/CA/IC1803/S-1-2022.CA.IC1803.2/file-storage/view/public%2Ftareas-programadas%2FTP1_Criptograf%C3%ADa_2022.pdf>

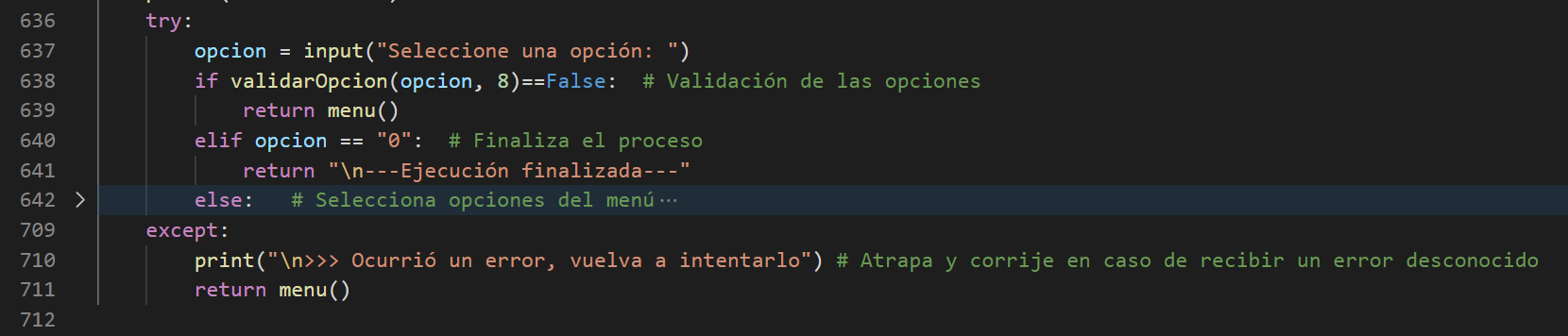
**Eliminación de olores:**

1. **Rigidez:** El código está hecho de manera simple, siguiendo una serie de patrones que han sido cuidadosamente documentados, permitiendo así una fácil manipulación y actualización del código.

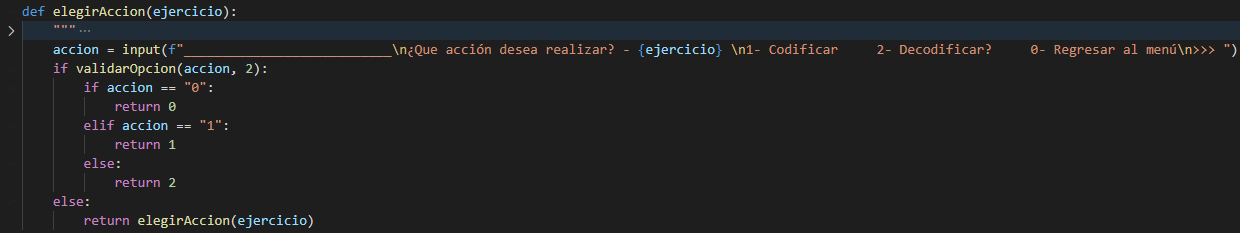


1. **Fragilidad:** Fueron creadas una serie de validaciones recursivas para cada tipo de entradas de datos que fueran necesarias a lo largo del código. Además, se implementó un try-catch fuera de las llamadas respectivas a cada función, permitiendo así el amortiguamiento de posibles errores.

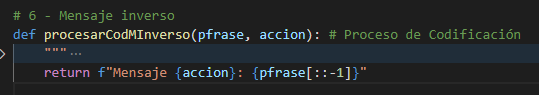




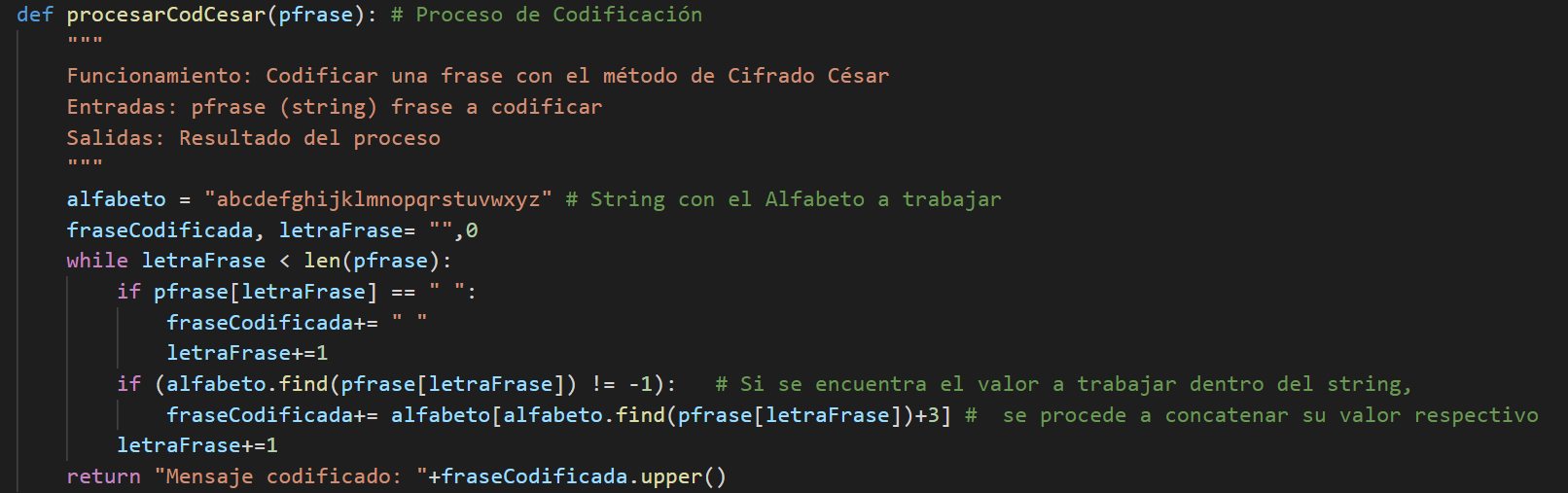
1. **Inmovilidad:** Las funciones fueron hechas para usarse varias veces en todo el código de manera que sea fácil de reutilizar a lo largo de este.



1. **Viscosidad:** El programa permite realizar todo tipo de cambio sin afectar el funcionamiento del código en general.
2. **Complejidad innecesaria:** Se buscaron formas de hacer las funciones de la manera más simple posible dentro de los temas vistos en clase.



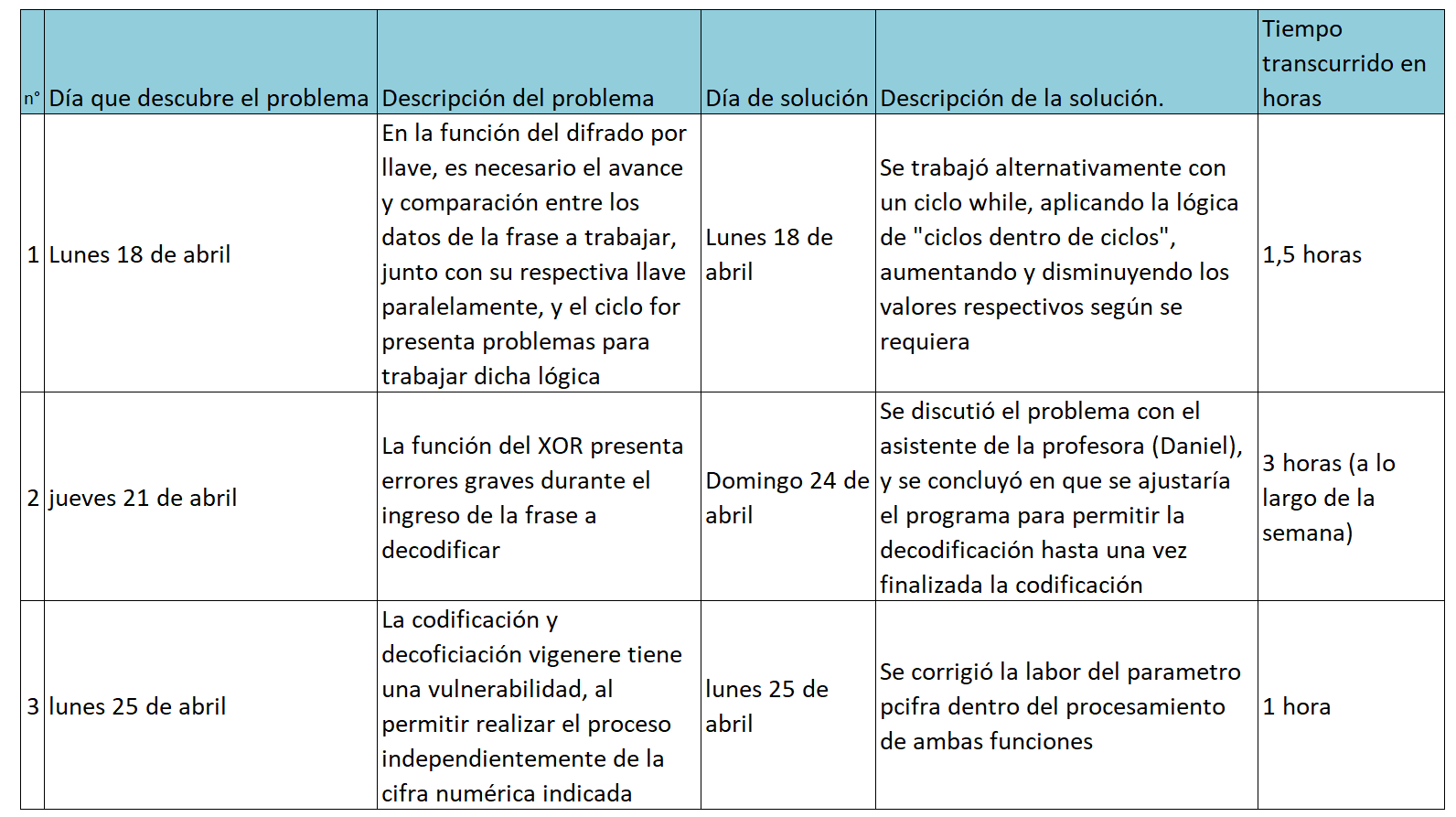
1. **Repetición innecesaria:** Ninguna función fue ‘copiada’ y ‘pegada’, no se repiten pedazos de códigos innecesarios. En caso de necesitarse una función en repetidas ocasiones, se define la función de manera que pueda ser utilizada en cualquier momento.
2. **Opacidad:** El código está escrito de manera que sea muy fácil de leer e interpretar, ya sea en el nombre de las variables o en la forma en que se realizan las funciones.



**Conclusiones del trabajo**

**Problemas encontrados y soluciones a los mismos**

A lo largo de la realización del trabajo, diversos desafíos fueron enfrentados para alcanzar el éxito. Entre ellos se pueden destacar los siguientes:

****

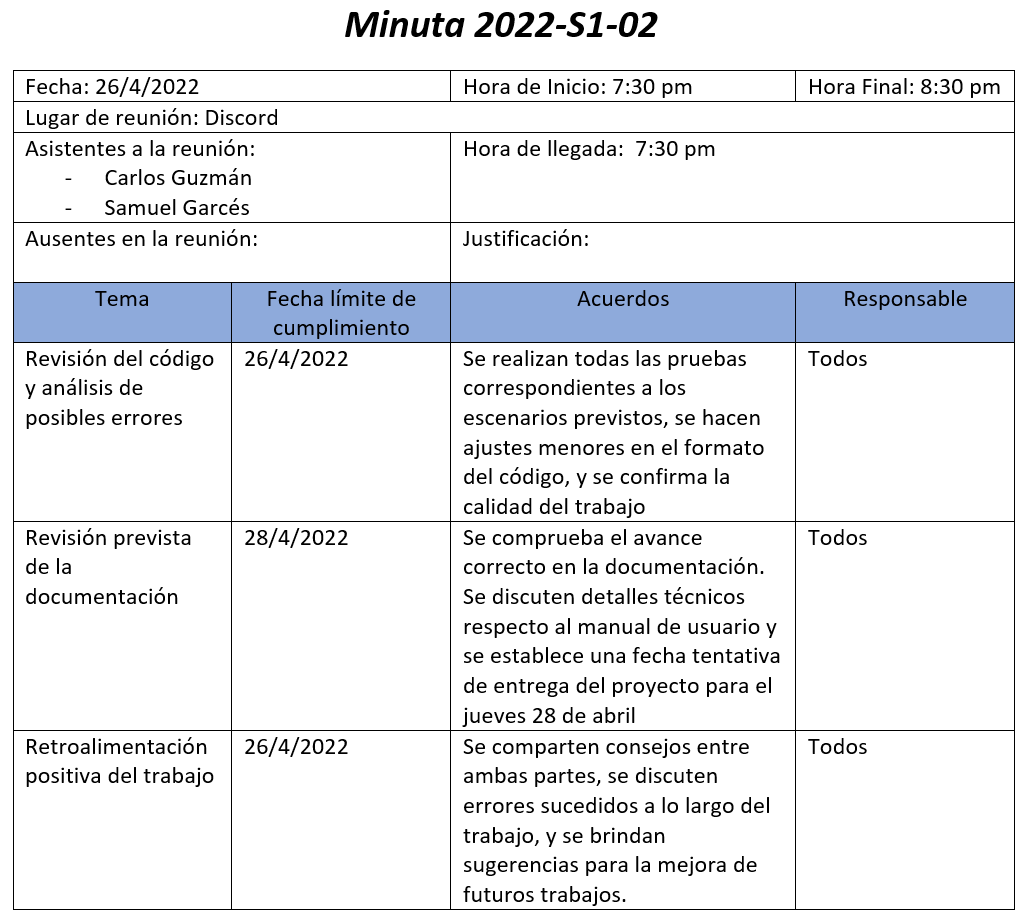
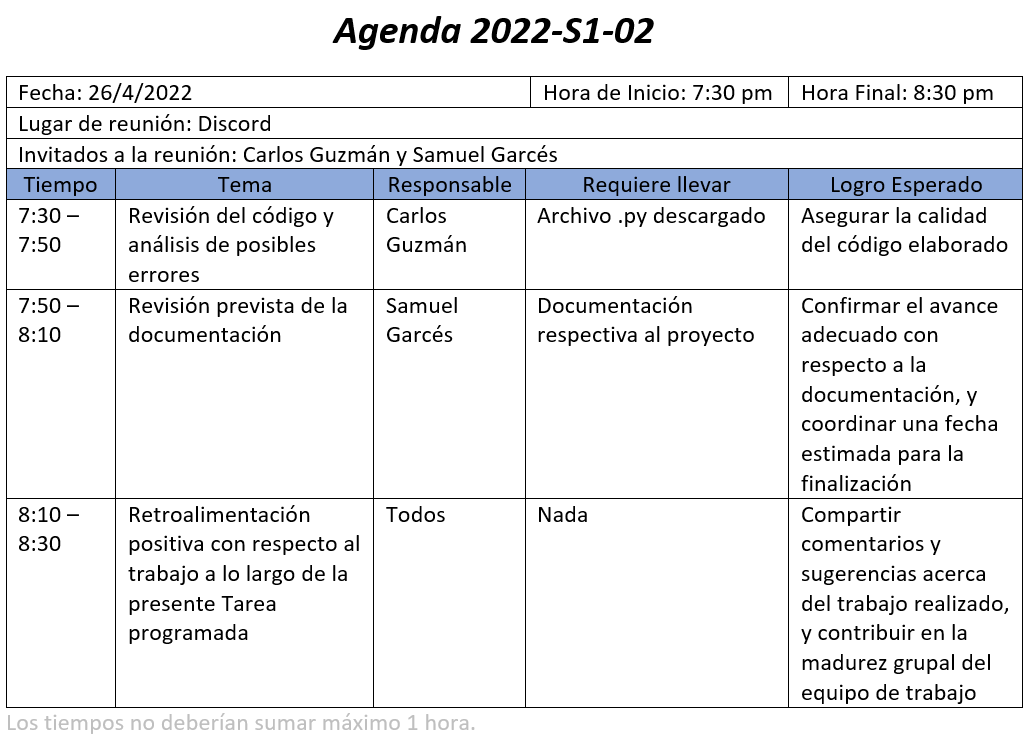
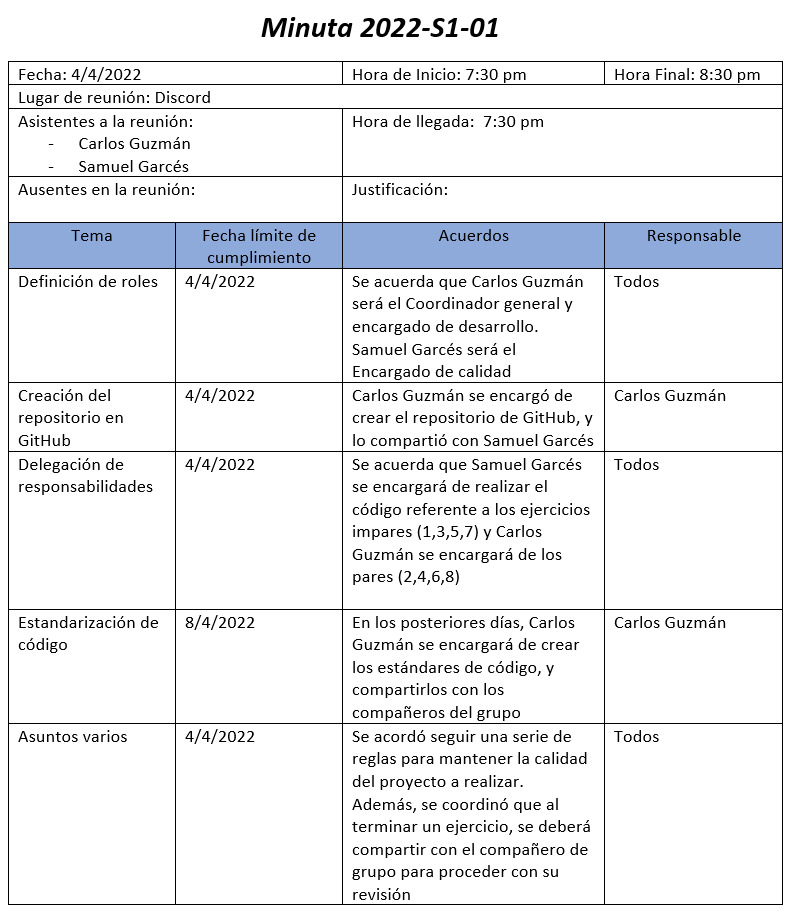
**Aprendizajes obtenidos**

|  |  |  |
| --- | --- | --- |
| **N°** | **Carácter personal** | **Carácter técnico** |
| 1 | Trabajo en equipo | Uso de la lógica de módulos para la optimización del código |
| 2 | Distribución de las labores | Codificación de if-else en una sola línea |
| 3 | Búsqueda de soluciones diversas - autodidacta | Codificación de ciclos “for” en una sola línea |
| 4 | Solución de problemas | Validación recursiva de los datos de entrada |
| 5 | Optimización del tiempo | Utilización del formato f“...” para concatenar strings |
| 6 | Autocritica | Creación de menús complejos |
| 7 | Reconocimiento de errores | Utilización de funciones estratégicas de listas |
| 8 | Análisis crítico de código ajeno | Utilización estratégica de strings combinados con listas |
| 9 | Simulación de un ambiente laboral real | Aplicación práctica de expresiones regulares |

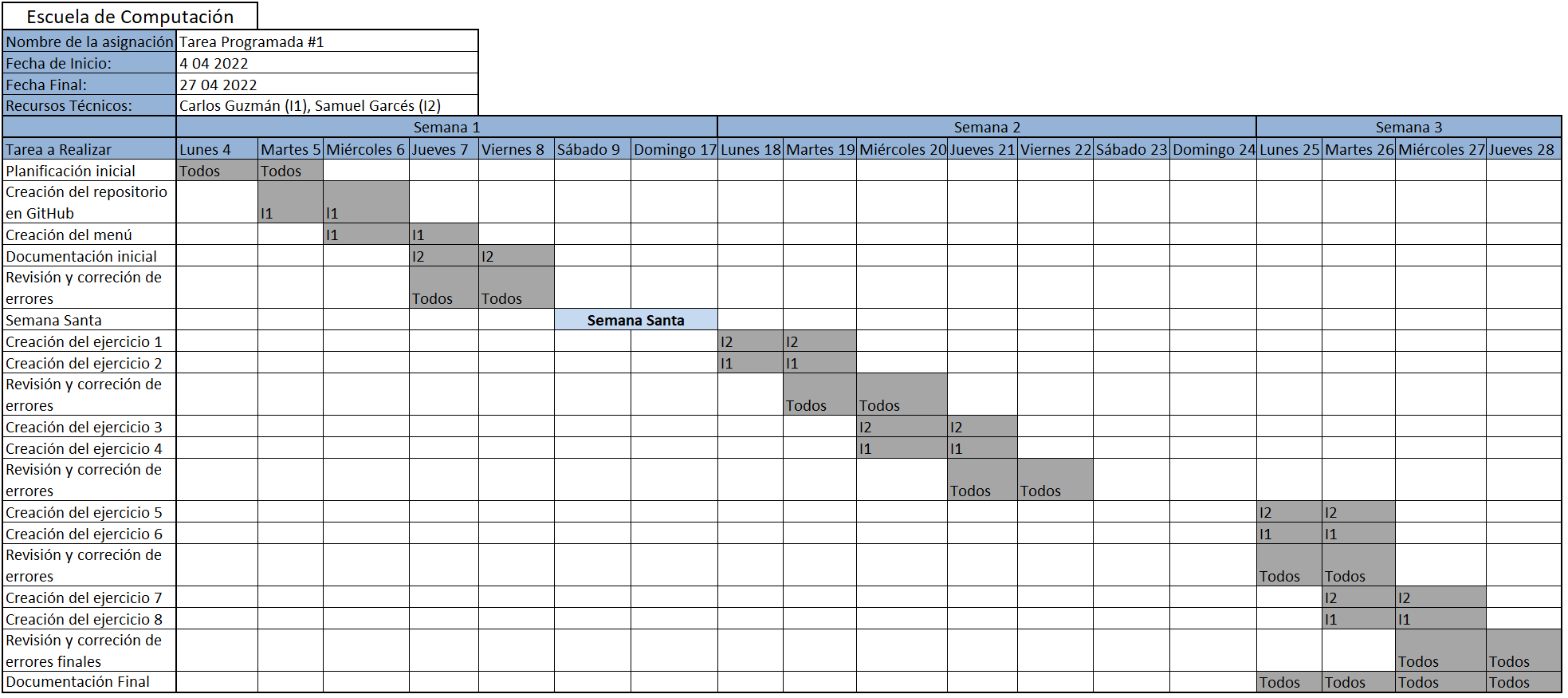
**Reglamento de trabajo**

1. La plataforma GitHub será el medio oficial para la manipulación de los archivos y el código de software.
2. Cada integrante debe trabajar un mínimo de 4 horas por semana en sus labores correspondientes.
3. Debe cumplirse con una técnica de trabajo limpia y ordenada en todo momento.
4. Debe seguirse una lógica de programación libre de olores de software.
5. Debe trabajarse única y exclusivamente en la rama personal de cada integrante, y se debe informar al coordinador general cuando se solicite el agregado a la rama principal.
6. En caso de ser necesario, debe informarse a los demás integrantes del equipo acerca de cualquier cambio que se realice en la rama principal.
7. Cualquier comentario, sugerencia, crítica constructiva o demás aporte que se brinde a cualquiera de los integrantes deberá realizarse de manera respetuosa, y de la misma manera se debe interpretar de manear proactiva para el beneficio de ambas partes.
8. No se tolerarán discusiones acaloradas o faltas de respeto ante ninguno de los integrantes de este equipo. Cualquier falta a esta regla podrá ser motivo de amonestación.

**Agendas y minutas**



**Cronograma**

****

**Estadísticas de tiempos**

|  |  |
| --- | --- |
| **Actividad realizada** | **Horas** |
| Análisis de problemas | 8 |
| Diseño de algoritmos | 5 |
| Investigación de métodos diversos | 6 |
| Programación | 14 |
| Documentación del programa | 2 |
| Pruebas del programa | 4 |
| Elaboración del manual de usuario | 2 |
| Elaboración de documentación del proyecto | 6 |
| **TOTAL** | 47 horas |