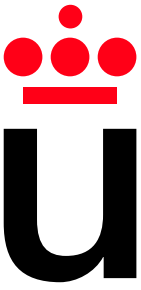


Universidad
Rey Juan Carlos

Escuela Técnica Superior
de Ingeniería Informática

Grado en Diseño y Desarrollo de Videojuegos

Curso 2022-2023

Trabajo Fin de Grado

APLICACIÓN WEB BASADA EN EL FRAMEWORK FLUTTER PARA LA ENSEÑANZA DE ALGORITMOS RECURSIVOS DE MANERA VISUAL

Autor: Juan Ignacio Castro Carmona
Tutor: Manuel Rubio Sánchez

Agradecimientos

Para comenzar, me gustaría expresar mi agradecimiento a Manuel Rubio Sánchez, mi tutor, por su apoyo y motivación durante todo el proceso, así como por su dedicación para guiarme y asesorarme en todo momento en la realización de este trabajo.

En segundo lugar, quiero mostrar mi agradecimiento a mi familia por el respaldo que me han brindado y por creer siempre en mí. Han sido una gran fuente de motivación y han estado a mi lado en todo momento.

Finalmente, pero no por eso menos importante, quiero dar las gracias a mis amigos, quienes me han dado su apoyo incondicional.

Resumen

La recursividad, un paradigma de programación que puede resultar desafiante para los principiantes, requiere un cambio de pensamiento en comparación con la programación iterativa. Con este fin, se ha creado RecuAlg, una aplicación web que guía a los estudiantes en el diseño visual y paso a paso de algoritmos recursivos. Mediante un diagrama, proporciona casos base y recursivos, demostrando cómo se divide el problema en subproblemas más pequeños y cómo resolverlos. Además, genera código Python en tiempo real para ilustrar la implementación del algoritmo, permitiendo a los estudiantes ver su funcionamiento en la práctica y aplicar técnicas de programación recursiva para resolver problemas complejos.

Para desarrollar RecuAlg, se eligió el framework Flutter y el lenguaje Dart debido a su potencia en el desarrollo de interfaces de usuario y su capacidad de compilación nativa en la web. De esta manera, se garantiza la disponibilidad para todos a través de una aplicación web en línea, uno de los principales objetivos del proyecto.

Palabras clave: Recursión, Diseño de algoritmos, Aplicación web, RecuAlg, Dart, Flutter, Herramienta didáctica, Python

Abstract

Recursion, a programming paradigm that can be challenging for beginners, requires a shift in thinking compared to iterative programming. With this in mind, RecuAlg, a web application, has been created to guide students in the visual design and step-by-step development of recursive algorithms. Through a diagram, it provides base and recursive cases, demonstrating how the problem is divided into smaller subproblems and how to solve them. Additionally, it generates real-time Python code to illustrate the algorithm's implementation, allowing students to see it in action and apply recursive programming techniques to solve complex problems.

To develop RecuAlg, the Flutter framework and the Dart language were chosen due to their power in user interface development and native compilation capability for the web. This ensures availability for everyone through the online availability of the web application, which is one of the main project objectives.

Palabras clave: Recursion, Algorithm design, Web application, RecuAlg, Dart, Flutter, Didactic tool, Python

Índice de contenidos

Índice de tablas

Índice de figuras

1 Introducción

En este capítulo se proporciona el contexto y alcance del proyecto. Se explicará la motivación que llevó a su realización y se presentarán los objetivos que se esperan cumplir. Asimismo, se describirá la estructura del documento y se comentará los temas a tratar en cada uno de los capítulos.

1.1. Contexto

En nuestros días, la tecnología se ha vuelto fundamental como instrumento para la enseñanza y el aprendizaje. En este sentido, las aplicaciones web son una opción cada vez más popular para facilitar la adquisición de conocimientos. La enseñanza de algoritmos recursivos puede resultar un reto para muchos estudiantes, debido a su complejidad y abstracción. Por ello, el desarrollo de una aplicación web que permita el desarrollo de estos algoritmos de manera visual puede facilitar comprensión de estos y resultar de gran utilidad para el proceso educativo.

En particular, en lo que se refiere a la enseñanza de algoritmos recursivos, se trata de una técnica de programación en la que una función se llama a sí misma con el fin de descomponer un problema en subproblemas más simples que se resuelven mediante la misma función. A pesar de su potencial para resolver problemas, su implementación puede resultar complicada para los estudiantes de programación, especialmente para aquellos que se inician en el tema. Con RecuAlg se busca facilitar este proceso mediante la guía paso a paso y el uso de esquemas.

1.2. Alcance

El alcance de este trabajo abarcará el diseño e implementación de una aplicación web para enseñar el diseño de algoritmos recursivos. La aplicación se enfocará en los conceptos fundamentales de los algoritmos recursivos. Para lograr esto, se llevará a cabo la identificación de los requisitos funcionales y no funcionales, los casos de uso, la elección de una metodología de desarrollo, así como la selección de las herramientas y tecnologías apropiadas. Además, se implementarán funcionalidades de diseño visual de algoritmos y se integrarán diversos ejercicios.

El fin de este trabajo es desarrollar una aplicación web de fácil acceso y uso que facilite la enseñanza y comprensión de algoritmos recursivos. Para lograr esto, se han establecido los siguientes objetivos generales que serán desarrollados en el capitulo 2.

1.3. Estructura del documento

Con el propósito de ofrecer una visión general del contenido de esta memoria, a continuación se enumeran los capítulos que la componen, cada uno de ellos con una breve descripción de los temas que se abordan en cada uno de ellos.

- **Capítulo 2: Objetivos.** Se especifican y describen tanto los objetivos generales como los específicos.
- **Capítulo 3: Estado del arte.** Se analizan diversas metodologías para la enseñanza de la recursividad y se examinan las alternativas actuales, identificando tanto sus aspectos positivos como negativos.
- **Capítulo 4: Descripción informática.** Se especifican los requisitos funcionales y no funcionales, se definen los casos de uso, se describe la metodología utilizada durante el desarrollo, se explican y justifican las herramientas seleccionadas para llevar a cabo el proyecto, se detalla la estructura y desarrollo del proyecto, y se presenta un manual de usuario que indica cómo realizar todas las acciones posibles.
- **Capítulo 5: Líneas futuras.** Se proponen una serie de mejoras que se pueden incorporar al proyecto y se definen dos rumbos iniciales que éste puede tomar.
- **Capítulo 6: Conclusion.** Se analiza si se han alcanzado con éxito los objetivos propuestos inicialmente.

2

Objetivos

En este capítulo se expondrán los objetivos del proyecto, los cuales se dividen en dos categorías: los objetivos generales, que son de carácter amplio y representan la visión global que se busca alcanzar; y los objetivos específicos, asociados a cada objetivo general, que detallan las metas específicas que se deben cumplir.

2.1. Desarrollar una aplicación para la enseñanza de la recursividad

El objetivo principal de este proyecto es desarrollar una aplicación para facilitar la enseñanza del diseño de algoritmos recursivos a través de una herramienta visual e interactiva. La aplicación permite a los usuarios entender, practicar y afianzar sus conocimientos en recursividad. La metodología utilizada para la enseñanza se basará en el enfoque descrito en el artículo *Problems in Comprehending Recursion and Suggested Solutions* [5] y en el libro *Introduction to Recursive Programming* [6]. Además, la aplicación ofrece retroalimentación en tiempo real a los usuarios al generar código de python, lo que les permite detectar y corregir errores de manera inmediata. De esta manera, se espera que la aplicación sea de gran ayuda tanto para los estudiantes que estén aprendiendo sobre recursividad como para los profesores que enseñan esta materia. Los objetivos específicos son:

- Facilitar la comprensión del funcionamiento e implementación de los algoritmos recursivos utilizando los diagramas presentes en el libro anteriormente

mencionado. Es importante destacar que estos diagramas sirven específicamente para abordar un subconjunto de problemas y su finalidad es ayudar a los estudiantes a asimilar los conceptos clave de la recursividad.

- Permitir a los usuario crear algoritmos recursivos de manera visual y sencilla.
- Incluir una amplia variedad de ejercicios y problemas que permitan a los usuarios adquirir, practicar y afianzar sus conocimientos en diseño de algoritmos recursivos.
- Ofrecer retroalimentación en tiempo real a los usuarios, que les permita detectar y corregir errores de manera inmediata.

2.2. Adaptar la aplicación a asignaturas de la URJC

Se desarrollará una herramienta de enseñanza que podrá ser utilizada en la asignatura de Diseño y Análisis de Algoritmos e incluso en Introducción a la Programación en diferentes grados de la Universidad Rey Juan Carlos (URJC). Esta herramienta permitirá a los estudiantes aplicar y reforzar los conocimientos adquiridos en clase a través de ejercicios prácticos.

Además de su utilidad para la enseñanza, también se contempla que la herramienta pueda ser utilizada en experimentos de innovación docente. Estos experimentos tendrán como objetivo evaluar la efectividad pedagógica de la metodología descrita en el artículo *Problems in Comprehending Recursion and Suggested Solutions*[5] y en el libro *Introduction to Recursive Programming* [6], así como los diagramas presentes en este último. Los objetivos específicos para lograr este objetivo general son los siguientes:

- Incluir ejercicios correspondientes a las asignaturas previamente mencionadas, de manera que estén alineados con los objetivos de aprendizaje de dichas materias.
- Aplicar la metodología y utilizar los diagramas mencionados anteriormente para que futuros experimentos de innovación docente puedan ponerlos a prueba y evaluar su efectividad.

2.3. Garantizar su disponibilidad para cualquier usuario

Por último, otro objetivo importante de la aplicación es garantizar su disponibilidad para cualquier persona, sin importar su ubicación o el dispositivo que utilicen. Para lograrlo, la aplicación se desarrollará como una aplicación web interactiva y se optimizará para ser compatible la mayoría de dispositivos y con los principales navegadores web. Algunos objetivos específicos relacionados con esta meta son:

- Desplegar la aplicación en un sitio web público.
- Optimizar la aplicación para que funcione correctamente en la mayoría de los navegadores web y dispositivos.
- Ofrecer la aplicación de forma gratuita para que cualquier usuario interesado pueda utilizarla sin limitaciones económicas.

3

Estado del arte

En este capítulo, se presentará el estado del arte en relación con la enseñanza de algoritmos recursivos. Se revisarán artículos y libros relevantes que abordan este tema. El objetivo es analizar el conocimiento existente y las aplicaciones relacionadas con la enseñanza de la recursión, identificando tanto los enfoques exitosos como las oportunidades de mejora. A través de esta revisión, se busca obtener una comprensión más sólida de las estrategias y herramientas utilizadas en la enseñanza de algoritmos recursivos.

3.1. Artículos sobre la enseñanza de algoritmos recursivos

En este apartado, se examinarán y analizarán diferentes artículos que abordan la enseñanza de algoritmos recursivos. Estas fuentes proporcionarán una visión detallada de los enfoques, metodologías y resultados obtenidos en la enseñanza de este tema. Con ellos, se explorarán los problemas comunes que enfrentan los estudiantes al comprender y aplicar la recursión en la programación. Estos recursos ofrecerán una base sólida para comprender los desafíos y las mejores prácticas asociadas con la enseñanza de algoritmos recursivos.

Antes de adentrarnos en el análisis de los artículos más relevantes para el desarrollo de este trabajo, cabe destacar los siguientes:

1. ***Introducing Students to Recursion: A Multi-facet and Multi-tool Approach*** [7]: En el capítulo cuatro de este artículo se trata el desarrollo del pensamiento recursivo y presenta cómo se pueden introducir a los estudiantes diferentes conceptos de recursividad ejemplificando la resolución de esta tarea con los ejercicios de Fibonacci e impresión de los dígitos de un número.

2. ***Recursion in Gradual Steps (Is Recursion Really that Difficult?)*** [8]: En este artículo se trata la dificultad de la recursión, argumentando que esta no radica en el propio concepto de recursión sino que su problemática radica en los mecanismos de la programación imperativa y propone una introducción gradual a través de los siguiente tres campos: gramática, programación funcional y programación imperativa.

3.1.1. *Teaching Recursion in a Procedural Environment: How Much Should We Emphasize the Computing Model?*

El articulo *Teaching recursion in a procedural environment - How much should we emphasize the computing model?* [9] se enfoca en un estudio realizado en estudiantes universitarios de segundo año de ciencias de la computación. El objetivo del estudio fue determinar cuánto énfasis se debe poner en el modelo informático al enseñar recursión. El estudio se divide en dos fases:

En la primera fase, se administró una prueba al comienzo del curso para evaluar las dificultades de los estudiantes en la formulación recursiva. Como resultado, se encontró que 32 de los 42 alumnos dieron soluciones erróneas en la parte recursiva de la prueba.

En la segunda fase del estudio, los estudiantes se dividieron en dos grupos: el Grupo 1 se utilizó como grupo de control y continuó estudiando la recursión con énfasis en los mecanismos de recursión, mientras que el Grupo 2 se enfocó en el nivel declarativo y abstracto. Los resultados mostraron que enfatizar el nivel declarativo y abstracto mejoró significativamente la capacidad de formulación de soluciones recursiva de los estudiantes.

Además, en el artículo se hacen referencias a los siguientes articulos *Animating recursion as an aid to instruction* [10] y *Integrating algorithm animation into a learning environment* [11], donde se concluye que la animación que ilustra el "modelo de copias" de la recursión puede mejorar la comprensión y evaluación de funciones recursivas. Sin embargo, en el primer estudio se les pasó un cuestionario a los alumnos en el que indicaron que la visualización no les ayudaba a la hora de desarrollar los algoritmos recursivos.

3.1.2. *Problems in Comprehending Recursion and Suggested*

El artículo *Problems in Comprehending Recursion and Suggested Solutions*[5] se enfoca en las dificultades que enfrentan los estudiantes al aprender la recursión en programación. El autor explica que, a pesar de ser una herramienta poderosa para resolver problemas complejos, muchos estudiantes tienen dificultades para comprenderla.

El autor identifica tres problemas principales que contribuyen a estas dificultades: la falta de exposición al pensamiento declarativo, la falta de comprensión de la abstracción funcional y la ausencia de una metodología adecuada para expresar soluciones recursivas.

Para abordar estos problemas, el autor propone desarrollar habilidades de pensamiento declarativo y abstracción funcional en los estudiantes, al tiempo que se proporciona una metodología clara para expresar soluciones recursivas. Esta metodología consta de los siguientes pasos:

1. **Identificar el caso base**: Se trata de identificar el problema más sencillo que se puede resolver sin recurrir a la recursión.
2. **Simplificar el problema**: Se reduce gradualmente el tamaño del problema original mediante una rutina de simplificación, hasta alcanzar el caso base.
3. **Aplicar la recursión**: Se llama a la misma función con un problema más simple como argumento y se combinan los resultados para obtener la solución final.

El autor resalta la importancia de utilizar ejemplos concretos ara ayudar a los estudiantes a comprender mejor los conceptos abstractos de la recursión.

3.2. Aplicaciones para la enseñanza de algoritmos recursivos

En este apartado, se examinarán y analizarán diferentes aplicaciones relacionadas con la enseñanza de algoritmos recursivos. Se evaluarán las características, ventajas y desventajas de estas aplicaciones existentes, con el objetivo de identificar oportunidades de mejora que puedan ser aprovechadas en el contexto de este proyecto. A través de esta revisión, se buscará comprender cómo las aplicaciones existentes han abordado la enseñanza de algoritmos recursivos, y qué aspectos pueden ser considerados para el desarrollo de una nueva herramienta o enfoque que facilite y mejore el aprendizaje en este campo.

3.2.1. Recursion Tree Visualizer

Recursion Tree Visualizer [1] es una aplicación que permite visualizar el proceso de recursión a través de la creación de un árbol de recursión.

La herramienta es muy visual y muestra el código de los algoritmos como se puede ver en la Figura 3.1, lo que facilita la comprensión de los procesos de recursión.

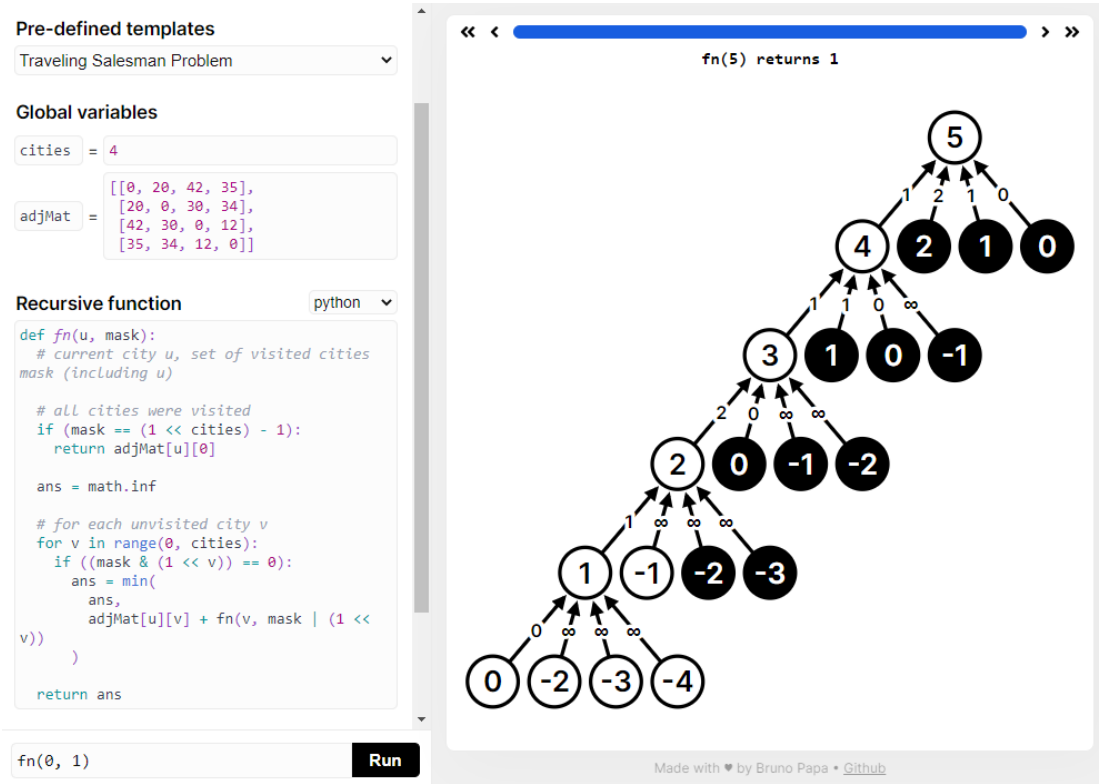


Figura 3.1: Interfaz. Fuente Recursion Tree Visualizer [1]

Además, como se puede apreciar en la Figura 3.2 esta herramienta cuenta con varios ejemplos de algoritmos implementados para que los usuarios puededan utilizarlos para practicar y experimentar con la recursividad.

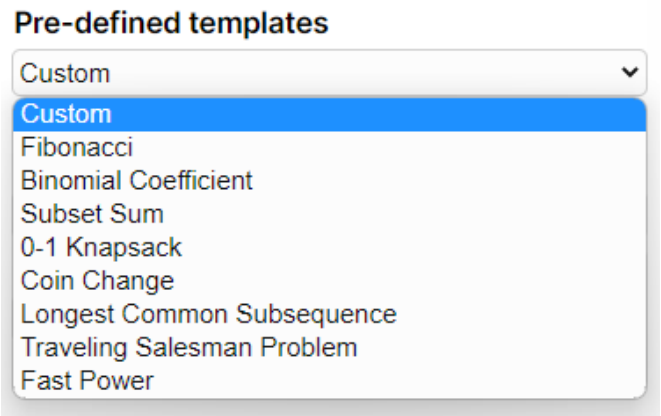


Figura 3.2: Ejemplos. Fuente Recursion Tree Visualizer [1]

Como puntos positivos, podemos mencionar que Recursion Tree Visualizer es una de las pocas aplicaciones enfocadas exclusivamente en la recursividad. La herramienta también es muy usable y sencilla.

Por otro lado, la herramienta tiene un gran problema en su enfoque ya que no es una aproximación acertada para aprender recursividad. Se basa en comprender el árbol de recursión y no en pensar de forma recursiva, es decir, en pensar en los casos bases, recursivos, cómo el problema se divide y se resuelve. Por lo tanto, aunque Recursion Tree Visualizer puede ser una herramienta útil para visualizar la recursión, no es la mejor opción para aquellos que quieran aprender recursividad.

3.2.2. SRec

SRec [12] es una aplicación que proporciona a los usuarios la capacidad de visualizar de manera animada algoritmos recursivos en los sistemas operativos Windows y Linux. Esta herramienta ofrece múltiples vistas, como el grafo de dependencia y el árbol de recursión, que permiten al usuario analizar y comprender visualmente el funcionamiento de los algoritmos recursivos.

Uno de los aspectos más destacados de SRec es su amplia capacidad de configuración y personalización. Los usuarios tienen la libertad de ajustar y adaptar la aplicación según sus necesidades y preferencias. Además, SRec permite exportar las animaciones generadas en formatos gráficos estándar, lo que facilita su uso y compartición en distintos contextos educativos.

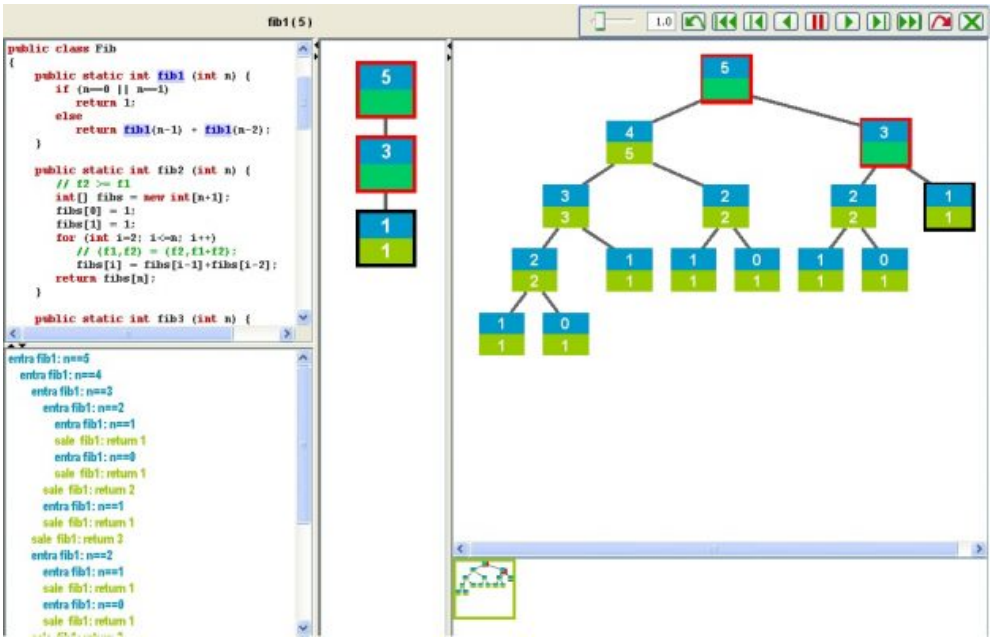


Figura 3.3: Interfaz SRec. Fuente SRec

Otra ventaja importante de SRec es la posibilidad de modificar el código del algoritmo directamente dentro de la aplicación, esto se puede ver en la Figura 3.3. Esto permite a los usuarios experimentar y probar diferentes implementaciones, lo que contribuye a un aprendizaje más práctico y profundo de los conceptos relacionados con la recursividad.

Sin embargo, es importante tener en cuenta que aunque SRec brinda la visualización de diferentes vistas relacionadas con la recursión, no está diseñada específicamente con el propósito de ayudar a los usuarios a comprender los conceptos de la recursión.

Además, es necesario destacar que el uso de SRec requiere la instalación en el sistema operativo correspondiente y la configuración inicial para su correcto funcionamiento. Esto puede implicar un proceso adicional para los usuarios, aunque una vez realizado, proporciona una experiencia interactiva y enriquecedora en el aprendizaje de algoritmos recursivos.

3.2.3. Visualgo

Visualgo [2] es una aplicación web que permite visualizar diferentes algoritmos y estructuras de datos de manera gráfica. La herramienta tiene como objetivo ayudar a los usuarios a comprender mejor cómo funcionan estos algoritmos y estructuras de datos a través de visualizaciones interactivas.

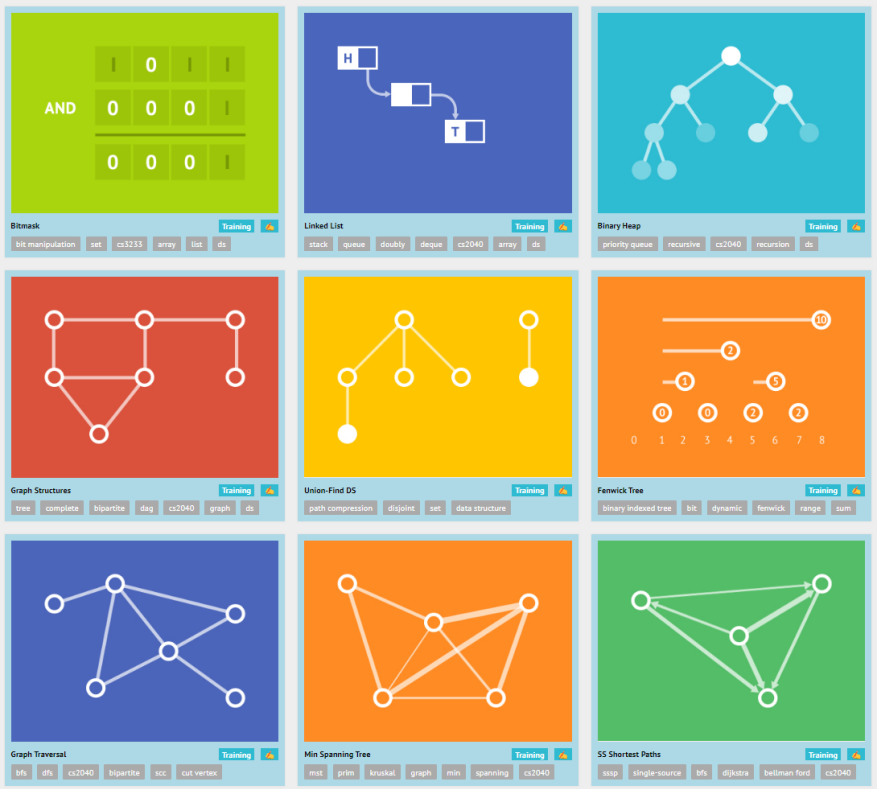


Figura 3.4: Algoritmos. Fuente CodingBat [2]

Entre los puntos fuertes de Visualgo, cuenta con una amplia variedad de algoritmos, lo que la hace muy completa. como se puede ver en la Figura 3.4.

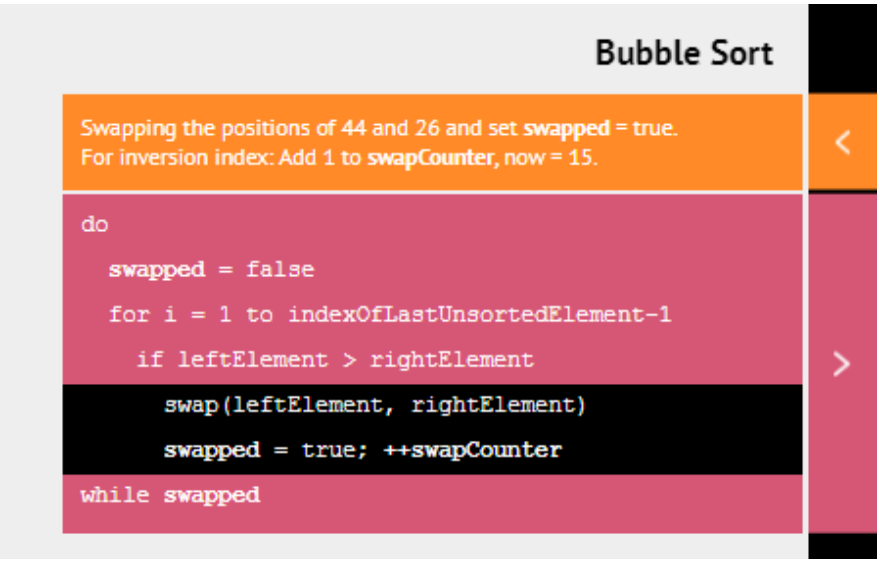


Figura 3.5: Menú de código. Fuente CodingBat [2]

Otro de sus puntos positivos es que muestra el código como se puede notar en la Figura 3.5, esto permite al usuario entender el funcionamiento del algoritmo. Esto es especialmente útil para aquellos que quieran profundizar en el estudio de algoritmos y estructuras de datos.

(a) Menús cerrados.

(b) Menús abiertos.

Figura 3.6: Menús de ejercicio. Fuente Visualgo [2]

Sin embargo, por el contrario, Visualgo resulta muy confusa y difícil de usar debido a que los menús están escondidos en menús desplegables en los bordes de la pantalla y todos estos menús están representados con el mismo icono como se ve en la Figura 3.6a, lo que hace que sea difícil averiguar qué funciones e información contiene cada uno hasta que los desplegamos como se observa en la Figura 3.6b. Aunque la herramienta es visualmente estética, la falta de claridad en su diseño la hace menos usable para aquellos que no estén familiarizados con su uso.

3.2.4. CodingBat

CodingBat [3] es una plataforma en línea que brinda una amplia variedad de ejercicios de programación en varios lenguajes, como Python y Java. Estos ejercicios están diseñados para apoyar a los estudiantes en la mejora de sus habilidades en programación y resolución de problemas a través de la práctica.

Figura 3.7: Ejercicios del primer grupo de ejercicios. Fuente CodingBat [3]

El mayor punto positivo de CodingBat es la gran cantidad de ejercicios que ofrece como se puede ver en la Figura 3.7, lo que aporta un gran valor a esta solución.

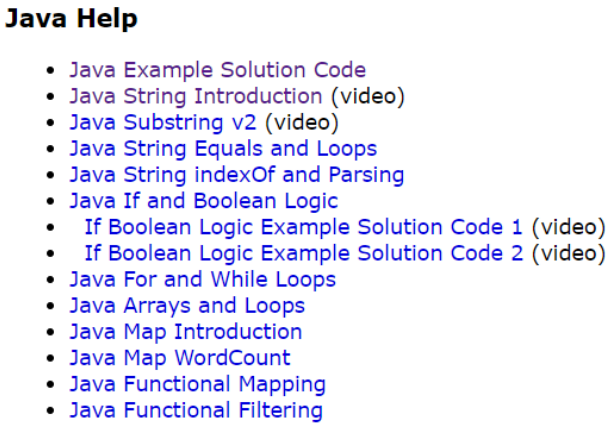


Figura 3.8: Sección de ayudas. Fuente CodingBat [3]

Otro aspecto destacable es que la herramienta también ofrece recursos complementarios como videos y artículos explicativos, como se puede observar en la Figura 3.8 que son muy útiles para el aprendizaje.

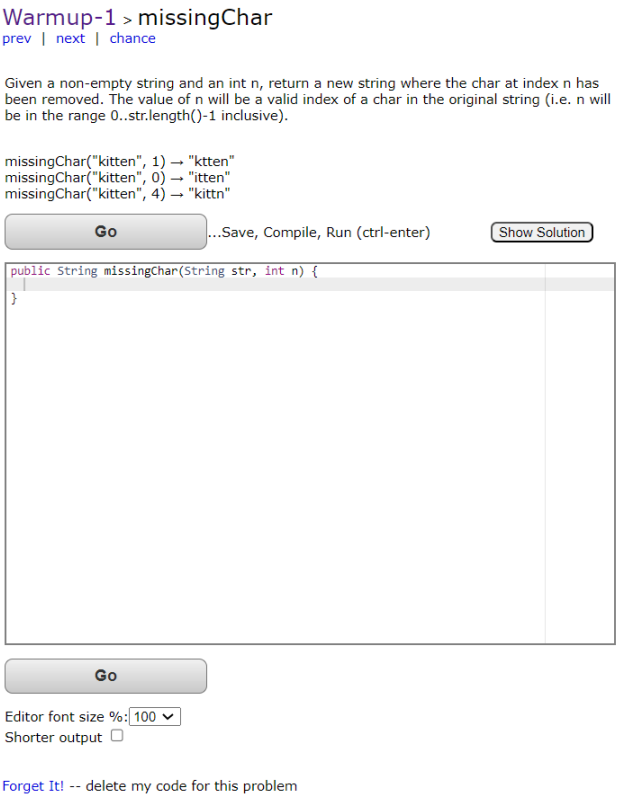


Figura 3.9: Interfaz de resolución de ejercicio. Fuente CodingBat [3]

No obstante, el mayor punto negativo de la plataforma es su interfaz como se puede apreciar en la Figura 3.9, que no es visualmente atractiva ni clara. Además, no ofrece una manera visual para resolver los ejercicios, sino que se tienen que resolver mediante código, lo que puede intimidar a los estudiantes novatos. Otro aspecto negativo es que, aunque la plataforma ofrece algunos ejercicios sobre algoritmos recursivos, no está especializada en este enfoque.

3.3. *Introduction to recursive programming*

El libro *Introduction to Recursive Programming* [6] ofrece una introducción detallada y completa sobre la recursión y sirve como una guía para aprender a pensar y programar de manera recursiva. Este libro ofrece una gran cantidad de ejemplos y problemas además de ofrecer una metodología para abordar el diseño de algoritmos recursivos.

El mayor punto fuerte de *Introduction to Recursive Programming* es cómo enseña a afrontar el diseño de algoritmos recursivos. La metodología que utiliza se basa en definir el tamaño del problema, definir los casos base, cómo descomponer el problema en otros similares pero de menor tamaño y definir los casos recursivos

mediante el uso de la inducción y diagramas. Estos conceptos se explican de manera clara y detallada, lo que facilita su comprensión y aplicación en la programación.

El único punto débil que se podría mencionar es que se limita al medio del libro y, por ende, no permite la interacción con los diagramas de resolución de los ejemplos y problemas. Es precisamente en ese aspecto donde RecuAlg se centra principalmente, en la interacción con los problemas.

4

Descripción informática

En este capítulo se especificarán los requisitos funcionales y no funcionales, se definirán los casos de uso, se describirá la metodología utilizada durante el desarrollo, se explicarán y justificarán las herramientas seleccionadas para llevar a cabo el proyecto, se detallará la estructura y desarrollo del mismo, y se presentará un manual de usuario que indica cómo realizar todas las acciones posibles.

4.1. Análisis

En este apartado, se llevará a cabo un análisis detallado de los requisitos funcionales, requisitos no funcionales y casos de uso del sistema. Estos elementos son fundamentales para comprender y definir las características y funcionalidades clave del proyecto.

En primer lugar, se especificarán los requisitos funcionales, que describen las acciones y funciones específicas que el sistema debe ser capaz de realizar. Posteriormente se identificarán y definirán los requisitos no funcionales, que se refieren a las características y cualidades generales del sistema.

Y por ultimo se definirán los casos de usos que representan una tarea que el usuario puede realizar en el sistema.

4.1.1. Requisitos funcionales

A continuación se presentan los requisitos funcionales que debe contener el sistema. Estos requisitos definen las funcionalidades específicas que deben ser implementadas para satisfacer las necesidades del usuario. Cada requisito se enfoca en una tarea o función específica que debe ser posible realizar en el sistema.

RF 1. Indicación de la ubicación del usuario: El sistema debe mostrar en todo momento en qué pantalla de la aplicación se encuentra el usuario.

RF 2. Visualización de todos los ejercicios: El sistema debe permitir al usuario ver en todo momento los ejercicios disponibles para realizar.

RF 3. Selección de ejercicio: El sistema debe permitir al usuario acceder a los ejercicios.

RF 4. Mostrar indicaciones del ejercicio: El sistema debe mostrar al usuario el enunciado del ejercicio.

RF 5. Agregar casos base: El sistema debe permitir al usuario agregar tantos casos base como considere necesario.

RF 6. Quitar casos base: El sistema debe permitir al usuario eliminar los casos base previamente agregados.

RF 7. Selección o cambio de condición caso base: El sistema debe permitir que el usuario asigne y/o cambie la condición de ejecución de un caso base.

RF 8. Selección o cambio de operación de caso base: El sistema debe permitir al usuario asigne y/o cambie la operación a realizar si se cumple la condición del caso base.

RF 9. Agregar casos recursivos: El sistema debe permitir al usuario agregar tantos casos recursivos como considere necesario.

RF 10. Quitar casos recursivos: El sistema debe permitir al usuario eliminar los casos recursivos previamente agregados.

RF 11. Selección o cambio de condición en caso recursivo: El sistema debe permitir que el usuario asigne y/o cambie la condición de ejecución de un caso recursivo si no es la condición por defecto.

RF 12. Asignación automática de condición en caso recursivo: El sistema debe asignar la condición por defecto al último caso recursivo agregado.

RF 13. Selección o cambio de operación de división de problema en casos recursivos: El sistema debe permitir que el usuario asigne y/o cambie la operación de división del problema.

RF 14. Selección o cambio de operación de resolución de subproblema en casos recursivos: El sistema debe permitir que el usuario asigne y/o cambie la operación de resolución de subproblema.

RF 15. Introducción de parámetros en caso recursivo: El sistema debe permitir al usuario introducir parámetros en casos recursivos para ver cómo sus decisiones afectan a los resultados.

RF 16. Mostrar solución esperada para los parámetros dados: El sistema debe mostrar al usuario la solución esperada para los parámetros que este haya introducido.

RF 17. Mostrar subproblema: El sistema debe mostrar al usuario el resultado del subproblema según los parámetros introducidos y la opción del subproblema.

RF 18. Mostrar solución del subproblema: El sistema debe mostrar al usuario la solución al subproblema generado.

RF 19. Generar código en tiempo real: El sistema debe mostrar al usuario en tiempo real el código en el lenguaje Python que realiza las acciones que este va seleccionando.

RF 20. Verificación del ejercicio: El sistema debe permitir al usuario verificar si el ejercicio realizado es correcto o incorrecto.

RF 21. Mostrar errores del ejercicio: El sistema debe indicar los errores que tenga la solución del usuario.

RF 22. Copiar código: El sistema debe permitir al usuario copiar el código que se ha generado durante la realización del ejercicio.

4.1.2. Requisitos no funcionales

A continuación se presentarán los requisitos no funcionales que deben ser considerados en el desarrollo del sistema. Estos requisitos definen los criterios de calidad que deben ser cumplidos para satisfacer las expectativas del usuario en cuanto a aspectos como rendimiento, usabilidad y compatibilidad. Cada requisito se enfoca en una característica específica que debe ser evaluada y medida para garantizar un sistema eficiente y confiable.

RNF 1. Rendimiento: Todas las operaciones de la aplicación deben completarse en menos de un segundo, medido desde la selección de la opción hasta la finalización de la tarea.

RNF 2. Usabilidad: Un nuevo usuario debe ser capaz de completar una tarea en la aplicación sin ayuda en menos de cinco minutos.

RNF 3. Compatibilidad: El sistema debe funcionar sin errores en los principales navegadores (Google Chrome, Safari, Microsoft Edge y Mozilla Firefox).

4.1.3. Casos de uso

En este apartado se describirán los casos de uso del sistema, los cuales representan las distintas funcionalidades y acciones que los usuarios pueden llevar a cabo. Con el fin de facilitar la comprensión y visualización de estas interacciones, se presentará un diagrama de casos de uso que ilustrará de forma gráfica la relación entre los actores y el sistema.

Posteriormente, se detallarán y definirán individualmente cada uno de los casos de uso identificados en el diagrama. Cada caso de uso incluirá su objetivo, las condiciones previas requeridas, los pasos a seguir, las condiciones posteriores, así como posibles extensiones e inclusiones.

Diagrama de casos de uso

En la Figura 4.1 se muestra el diagrama de casos de uso completo de RecuAlg, donde se destacan las principales acciones que los usuarios pueden llevar a cabo. Este diagrama proporciona una visión general de las funcionalidades del sistema y ayuda a comprender de manera visual las interacciones entre los actores y el sistema.

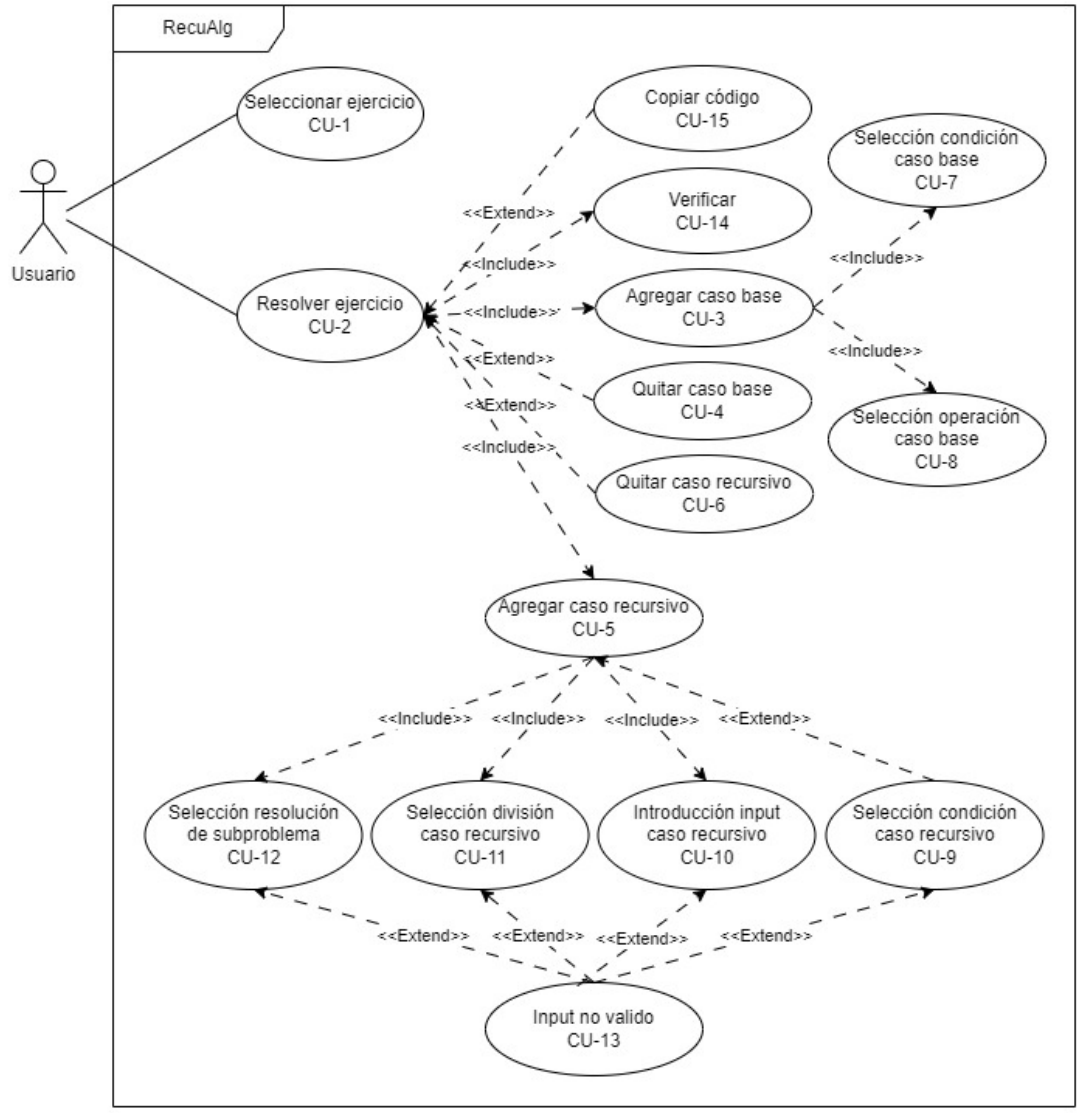


Figura 4.1: Diagrama de casos de uso.

Descripción casos de uso

A continuación, se presentarán todos los casos de uso identificados en el sistema RecuAlg. Cada caso de uso incluirá su identificador, nombre, los actores involucrados, una descripción detallada, las precondiciones necesarias, el escenario principal y las postcondiciones, así como posibles extensiones e inclusiones relacionadas.

Tabla 4.1: CU-1: Seleccionar ejercicio

Identificador	CU-1
Caso de uso	Seleccionar ejercicio
Actores	Usuario
Descripción	Permite al usuario seleccionar un ejercicio para resolver
Extensión	-
Inclusión	-
Escenarios	
Variante 1	
Descripción	Selección desde la pantalla de "Selección de Ejercicio"
Precondición	El usuario se encuentra en la pantalla de selección de ejercicio
Escenario	1. El usuario accede a la pantalla de "Selección de Ejercicio" 2. El usuario visualiza la lista de ejercicios disponibles 3. El usuario selecciona un ejercicio de la lista
Postcondición	El sistema carga la pantalla de "Resolución de Ejercicio" con el ejercicio seleccionado
Variante 2	
Descripción	Selección desde el menú lateral
Precondición	-
Escenario	1. El usuario abre el menú lateral desde la esquina superior izquierda en caso de que este no esté abierto 2. El usuario selecciona la opción "Ejercicios" 3. El usuario visualiza la lista de ejercicios disponibles 4. El usuario selecciona un ejercicio de la lista 5. El sistema carga la pantalla de "Resolución de Ejercicio" con el ejercicio seleccionado
Postcondición	El sistema carga la pantalla de "Resolución de Ejercicio" con el ejercicio seleccionado

Tabla 4.2: CU-2: Resolver ejercicio

Identificador	CU-2
Caso de uso	Resolver ejercicio
Actores	Usuario
Descripción	Permite al usuario resolver un ejercicio
Precondición	Haber seleccionado un ejercicio para resolver
Escenario	1. El usuario accede a la pantalla de "Resolución de Ejercicio" 2. El usuario visualiza el enunciado del ejercicio seleccionado 3. El usuario agrega casos base al ejercicio (condición y operación) 4. El usuario agrega casos recursivos al ejercicio (input, condición, división de problemas y resolución de subproblemas) 5. El sistema genera código Python en tiempo real que representa el algoritmo recursivo 6. El usuario ejecutar el código generado para verificar su funcionamiento 7. El usuario puede copiar el código generado a través de un botón
Postcondición	-
Extensión	-
Inclusión	1. Agregar caso base 2. Quitar caso base 3. Agregar caso recursivo 4. Quitar caso recursivo 5. Verificar ejercicio

Tabla 4.3: CU-3: Agregar caso base

Identificador	CU-3
Caso de uso	Agregar caso base
Actores	Usuario
Descripción	Permite al usuario agregar un caso base al ejercicio en resolución
Precondición	Estar resolviendo un ejercicio
Escenario	1. El usuario selecciona la opción de agregar un caso base 2. El usuario selecciona la condición del caso base mediante un menú desplegable 3. El usuario selecciona la operación del caso base mediante un menú desplegable
Postcondición	-
Extensión	-
Inclusión	1. Selección de condición 2. Selección de operación

Tabla 4.4: CU-4: Agregar caso recursivo

Identificador	CU-4
Caso de uso	Agregar caso recursivo
Actores	Usuario
Descripción	Permite al usuario agregar un caso recursivo al ejercicio en resolución
Precondición	Estar resolviendo un ejercicio
Escenario	1. El usuario selecciona la opción de agregar un caso recursivo 2. El usuario ingresa el input del caso recursivo 3. El usuario selecciona la condición del caso recursivo 4. El usuario selecciona la forma en la que el problema se divide en subproblemas más pequeños 5. El usuario selecciona cómo se resuelven los subproblemas generados
Postcondición	El sistema muestra la solución esperada para el input y la solución del subproblema
Extensión	Input no valido.
Inclusión	1. Introducción de input 2. Selección de condición 3. Selección de división del problema 4. Selección de resolución del problema

Tabla 4.5: CU-5: Quitar caso base

Identificador	CU-5
Caso de uso	Quitar caso base
Actores	Usuario.
Descripción	Permite al usuario eliminar un caso base del ejercicio en resolución
Precondición	1. Estar resolviendo un ejercicio 2. Tener al menos un caso base agregado
Escenario	El usuario selecciona eliminar caso base
Postcondición	El sistema elimina el último caso base agregado
Extensión	-
Inclusión	-

Tabla 4.6: CU-6: Quitar caso recursivo

Identificador	CU-6
Caso de uso	Quitar caso recursivo
Actores	Usuario
Descripción	Permite al usuario eliminar un caso recursivo del ejercicio en resolución
Precondición	1. Estar resolviendo un ejercicio 2. Tener al menos un caso recursivo agregado
Escenario	El usuario selecciona eliminar caso recursivo
Postcondición	El sistema elimina el último caso base agregado
Extensión	-
Inclusión	-

Tabla 4.7: CU-7: Selección condición caso base

Identificador	CU-7
Caso de uso	Selección condición caso base
Actores	Usuario
Descripción	Permite al usuario seleccionar de una lista dada una condición para el caso base
Precondición	Tener al menos un caso base agregado
Escenario	1. El usuario presiona sobre el menú desplegable de opciones de condición para el caso base 2. El usuario selecciona de la lista dada una de las opciones
Postcondición	1. El sistema muestra en el caso base la opción seleccionada 2. El sistema actualiza el código generado con el valor seleccionado
Extensión	-
Inclusión	-

Tabla 4.8: CU-8: Selección operación caso base

Identificador	CU-8
Caso de uso	Selección operación caso base
Actores	Usuario
Descripción	Permite al usuario seleccionar de una lista dada una operación para el caso base
Precondición	Tener al menos un caso base agregado
Escenario	1. El usuario presiona sobre el menú desplegable de opciones de operación para el caso base 2. El usuario selecciona de la lista dada una de las opciones
Postcondición	1. El sistema muestra en el caso base la opción seleccionada 2. El sistema actualiza el código generado con el valor seleccionado
Extensión	-
Inclusión	-

Tabla 4.9: CU-9: Selección condición caso recursivo

Identificador	CU-9
Caso de uso	Selección condición caso recursivo
Actores	Usuario
Descripción	Permite al usuario seleccionar de una lista dada una condición para el caso recursivo
Precondición	Tener al menos un caso recursivo agregado
Escenario	1. El usuario presiona sobre el menú desplegable de opciones de condición para el caso recursivo 2. El usuario selecciona de la lista dada una de las opciones
Postcondición	1. El sistema muestra en el caso recursivo la opción seleccionada 2. El sistema actualiza el código generado con el valor seleccionado
Extensión	Input no valido
Inclusión	-

Tabla 4.10: CU-10: Introducción input caso recursivo

Identificador	CU-10
Caso de uso	Introducción input caso recursivo
Actores	Usuario.
Descripción	Permite al usuario introducir los parámetros para el caso recursivo
Precondición	Tener al menos un caso recursivo agregado
Escenario	1. El usuario presiona sobre el recuadro que pone "Input" del caso recursivo 2. El usuario introduce el input para el caso recursivo
Postcondición	El sistema muestra en el caso recursivo el input introducido
Extensión	Input no valido
Inclusión	-

Tabla 4.11: CU-11: Selección división del problema

Identificador	CU-11
Caso de uso	Selección división del problema
Actores	Usuario.
Descripción	Permite al usuario seleccionar de una lista dada una operación de división del problema para el caso recursivo
Precondición	Tener al menos un caso recursivo agregado
Escenario	1. El usuario presiona sobre el menú desplegable de opciones de división del problema para el caso recursivo 2. El usuario selecciona de la lista dada una de las opciones
Postcondición	1. El sistema muestra en el caso recursivo la opción seleccionada 2. El sistema actualiza el código generado con el valor seleccionado 3. El sistema muestra el resultado de la división del problema con la opción seleccionada
Extensión	Input no valido
Inclusión	-

Tabla 4.12: CU-12: Selección resolución de subproblema

Identificador	CU-12
Caso de uso	Selección resolución de subproblema
Actores	Usuario
Descripción	Permite al usuario seleccionar de una lista dada una operación de resolución de los subproblemas para el caso recursivo
Precondición	Tener al menos un caso recursivo agregado
Escenario	1. El usuario presiona sobre el menú desplegable de opciones de resolución del subproblema para el caso recursivo 2. El usuario selecciona de la lista dada una de las opciones
Postcondición	1. El sistema muestra en el caso recursivo la opción seleccionada 2. El sistema actualiza el código generado con el valor seleccionado
Extensión	Input no valido
Inclusión	-

Tabla 4.13: CU-13: Input no valido

Identificador	CU-13
Caso de uso	Input no valido
Actores	Usuario.
Descripción	Permite al usuario saber si el input que ha introducido es válido para las condiciones seleccionadas
Precondición	1. Tener al menos un caso recursivo agregado 2. El input no cumple con el formato pedido 3. El input no es válido para la selección de división del problema y/o resolución del subproblema.
Escenario	El usuario ingresa un valor que no cumple con el formato requerido y/o elige opciones incorrectas para la división del problema y/o la resolución del subproblema
Postcondición	El sistema muestra el mensaje “Input no valido” en el caso recursivo
Extensión	-
Inclusión	-

Tabla 4.14: CU-14: Verificar

Identificador	CU-14
Caso de uso	Verificar
Actores	Usuario
Descripción	Permite al usuario ejecutar el código que se ha generado mediante sus decisiones para verificar la validez de este
Precondición	Estar resolviendo un ejercicio
Escenario	El usuario presiona el botón verificar
Postcondición	1. El sistema muestra el mensaje "Correcto" si la resolución del ejercicio es válida 2. El sistema muestra el mensaje "Incorrecto" si la resolución del ejercicio es inválida
Extensión	-
Inclusión	-

Tabla 4.15: CU-15: Copiar código

Identificador	CU-15
Caso de uso	Copiar código
Actores	Usuario
Descripción	Permite al usuario copiar el código que se ha generado durante la resolución del ejercicio
Precondición	Estar resolviendo un ejercicio
Escenario	El usuario presiona sobre el botón copiar código
Postcondición	El sistema guarda en el portapapeles del dispositivo el código
Extensión	-
Inclusión	-

4.2. Metodología

Kanban es una metodología ágil de gestión visual de procesos y flujos de trabajo que tiene como fin mejorar la eficiencia, la calidad y la capacidad de entrega del trabajo en una organización. Kanban tiene su origen en el sector manufacturero japonés, específicamente en Toyota, y se ha utilizado durante décadas para mejorar la eficiencia y calidad en la producción.

La metodología Kanban se fundamenta en el uso de tableros Kanban y tarjetas Kanban para visualizar el flujo de trabajo y el progreso de las tareas. Las tarjetas Kanban representan las tareas y elementos del proyecto, y se mueven a lo largo

del tablero en columnas que representan las diferentes etapas del proceso. Esto permite visualizar de manera clara el estado de las tareas y el progreso general del proyecto.

En Kanban se utiliza un límite de Work In Progress (WIP) para controlar la cantidad de trabajo en progreso en cada etapa del proceso. Establecer un límite de WIP ayuda a evitar la sobrecarga del equipo, a identificar cuellos de botella en el proceso y a mejorar la eficiencia en la entrega.

Para el desarrollo de este proyecto en particular, se han definido cuatro columnas:

- **Backlog:** donde se sitúan todas las tareas identificadas para la realización del proyecto.
- **ToDo:** se sitúan las tareas que se han elegido realizar primero, tras analizar su importancia y relevancia respecto a otras del Backlog.
- **Doing:** donde se encuentran las tareas que se están realizando.
- **Done:** están todas las tareas ya completadas.

4.3. Herramientas utilizadas

En este apartado se presentan las herramientas seleccionadas y utilizadas durante el desarrollo del proyecto, junto con una explicación de las razones que respaldaron su elección.

4.3.1. Flutter

Flutter es un framework de código abierto desarrollado por Google para la creación de aplicaciones nativas multiplataforma. Flutter utiliza el lenguaje de programación Dart y ofrece un conjunto de herramientas y widgets personalizables que permiten crear interfaces de usuario de forma rápida y fluida [4].

Se ha elegido Flutter para el desarrollo de este proyecto por varias razones. En primer lugar, ofrece una excelente experiencia de desarrollo. Su arquitectura y herramientas facilitan la creación de interfaces de usuario complejas. Además, Flutter cuenta con una gran cantidad de paquetes y complementos disponibles que permiten agregar fácilmente funcionalidades adicionales a la aplicación.

Otra razón por la cual se ha elegido Flutter es su capacidad para crear aplicaciones multiplataforma. Con Flutter, es posible desarrollar una aplicación

una sola vez y ejecutarla en diferentes plataformas, como iOS, Android, web y escritorio. Esto es posible gracias a la compilación Ahead of Time (AOT).

También se ha decidido utilizar Flutter sobre otras alternativas como React-Native, Xamarin o Ionic porque, según mi criterio, Flutter ofrece una mejor experiencia de usuario y una mayor flexibilidad en el diseño y la personalización de interfaces de usuario. Además, Flutter es más rápido en la ejecución de código y en la carga de la aplicación en comparación con otras alternativas, lo que resulta en una mejor experiencia para el usuario.

4.3.2. Widgets

En el framework de Flutter, los widgets son elementos gráficos que representan una sección de la interfaz de usuario de la aplicación. Los widgets se emplean para construir la interfaz de usuario de la aplicación y ofrecen una gran variedad de alternativas de personalización para manejar su aspecto y funcionalidad.

Flutter tiene una gran cantidad de widgets predefinidos que se pueden utilizar para construir rápidamente interfaces de usuario. Estos widgets predefinidos incluyen botones, campos de texto, listas, menús desplegables, tablas e imágenes, entre otros. Además, Flutter permite crear widgets personalizados para cubrir las necesidades específicas de cada aplicación.

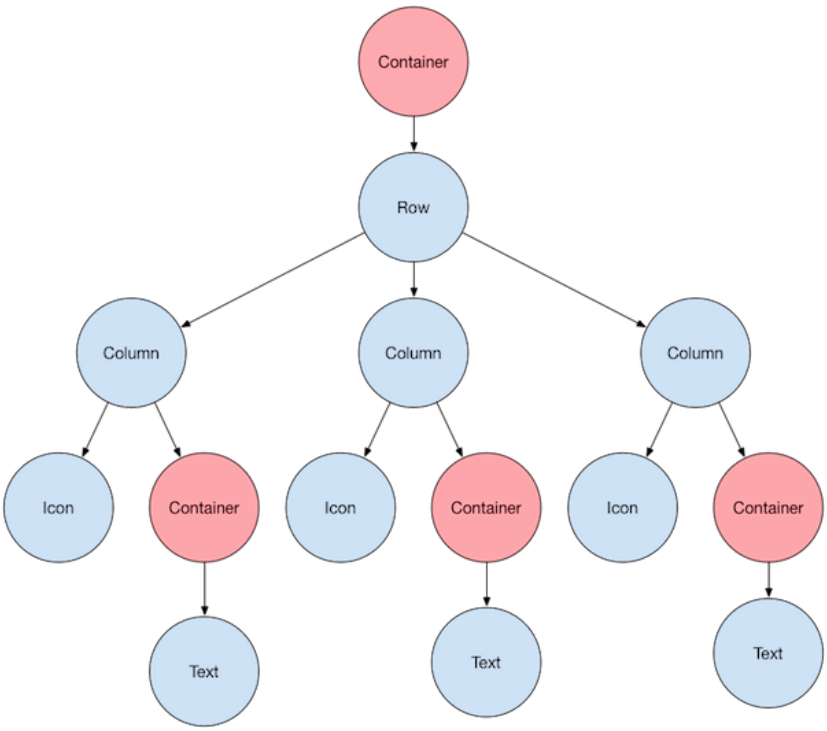


Figura 4.2: Árbol de widgets. Fuente Flutter [4]

Los widgets de Flutter se organizan en un árbol de widgets como se puede ver en la Figura 4.2, donde cada widget puede tener cero o más widgets secundarios. Esto permite construir interfaces de usuario complejas y dinámicas que, al cambiar los valores de los datos que representan o el estado de la aplicación, se actualizan automáticamente.

4.3.3. Dart

Dart es el lenguaje de programación utilizado en el framework de Flutter. Es un lenguaje de programación de código abierto desarrollado por Google, diseñado para optimizar la construcción de interfaces de usuario. Permite hacer cambios interactivos gracias a la actualización de la aplicación en tiempo de ejecución y compila a código máquina para plataformas móviles y de escritorio, y a JavaScript para aplicaciones web [13].

Dart cuenta con una amplia biblioteca de paquetes y extensiones, y una de las características más destacadas es su capacidad para realizar compilación de código en tiempo real Just-in-Time (JIT), lo que permite una rápida iteración en el proceso de desarrollo y una mayor facilidad para depurar el código. También cuenta con una compilación anticipada AOT, que transforma el código fuente en código de máquina nativo antes de que se ejecute el programa.

4.3.4. Librerías

En este apartado se presentan las librerías utilizadas para el desarrollo de la aplicación. La elección de estas ha sido fundamental para la implementación de ciertas funcionalidades, así como para optimizar la eficiencia y el rendimiento de la aplicación. En particular, se destacan dos librerías: GetX y Pocketpy. Cada una de ellas ha sido seleccionada por sus características y capacidades específicas, y se describen con más detalle en los subapartados siguientes:

GetX

GetX [14] es una extensión para Flutter que proporciona herramientas y funciones que simplifican el desarrollo de aplicaciones. GetX se enfoca en proporcionar una gestión eficiente del estado de la aplicación y la navegación entre pantallas.

Una de las características más destacadas de GetX es su capacidad para gestionar el estado de la aplicación de una manera sencilla y eficiente. GetX facilita utiliza el patrón de arquitectura de diseño Modelo-Vista-Controlador (MVC) para separar la lógica de la aplicación del código de la interfaz de usuario, lo que facilita el mantenimiento y la escalabilidad del código. GetX también proporciona un

sistema de observación de cambios que actualiza automáticamente la interfaz de usuario cuando cambia el estado de la aplicación.

También proporciona soporte para la gestión de dependencias que permite registrar un objeto como una dependencia y luego acceder a él en cualquier lugar de la aplicación. De esta manera, se pueden separar las dependencias de la lógica de la aplicación y mejorar la modularidad y escalabilidad del código.

Además, GetX incluye un sistema de navegación avanzado que simplifica la navegación entre pantallas.

Pocketpy

Pocketpy [15] es una librería que permite ejecutar código de Python en Dart. La librería implementa una máquina virtual de Python en Dart, lo que significa que puede interpretar y ejecutar código de Python directamente en Dart.

4.4. Estructura del proyecto

En este apartado, se presentará la estructura detallada del proyecto, con el objetivo de brindar una visión general de la organización de los diferentes elementos que conforman la aplicación. Se explicará la jerarquía de carpetas, los archivos más relevantes y la interacción entre los distintos módulos.

La estructura del proyecto es fundamental para garantizar un desarrollo ordenado y escalable de la aplicación y resulta esencial para entender cómo se ha organizado y cómo funciona el proyecto en su conjunto. El proyecto se divide principalmente en las siguientes carpetas:

4.4.1. Widgets

La carpeta “widgets” contiene todos los widgets personalizados, que son abstracciones de una parte de la interfaz de usuario de la aplicación. Todo lo que se muestra en pantalla en una aplicación Flutter es un widget. Estos widgets a su vez están divididos en cuatro carpetas:

Cases

La carpeta “cases” contiene los widgets que representan tanto los casos base como los casos recursivos.

En la Figura 4.3 y Figura 4.4 se encuentra estos widget identificados y delimitados para a continuación describir cada uno de estos brevemente:

1. El widget representado por la Figura 4.4a modela un caso base y, para ello, contiene dos menús desplegables: el primero (Figura 4.4b) permite seleccionar la condición para ejecutar la operación de ese caso, y el segundo (Figura 4.4c) permite seleccionar la operación a realizar si se cumple la condición.

2. La Figura 4.3a representa la implementa un caso recursivo y, para ello, contiene un menú desplegable (Figura 4.3b) para seleccionar la condición que permite ejecutar el caso. Además, define un esquema que consta de tres filas:

   a) En la primera fila se encuentra un recuadro para introducir los parámetros (Figura 4.3c), seguido de una flecha que lleva al último elemento de la fila. En este último elemento se muestra el resultado esperado (Figura 4.3d) para el ejercicio segun los parámetros introducidos.

   b) En la segunda fila, a la izquierda, se encuentra un menú desplegable para seleccionar cómo se va a dividir el problema en subproblemas (Figura 4.3e), y a la derecha, otro menú desplegable para seleccionar qué operación se realizará en la subsolución para llegar al resultado esperado (Figura 4.3f).

   c) En la última fila, se encuentra a la izquierda el recuadro donde se muestra el resultado de la división del problema (Figura 4.3g); y a la derecha, la flecha que lleva al recuadro donde se muestra la solución del subproblema (Figura 4.3h).

3. Las Figuras 4.4d y 4.3i muestran el widget que define la vista de los botones de agregación y eliminación de un caso base o recursivo.

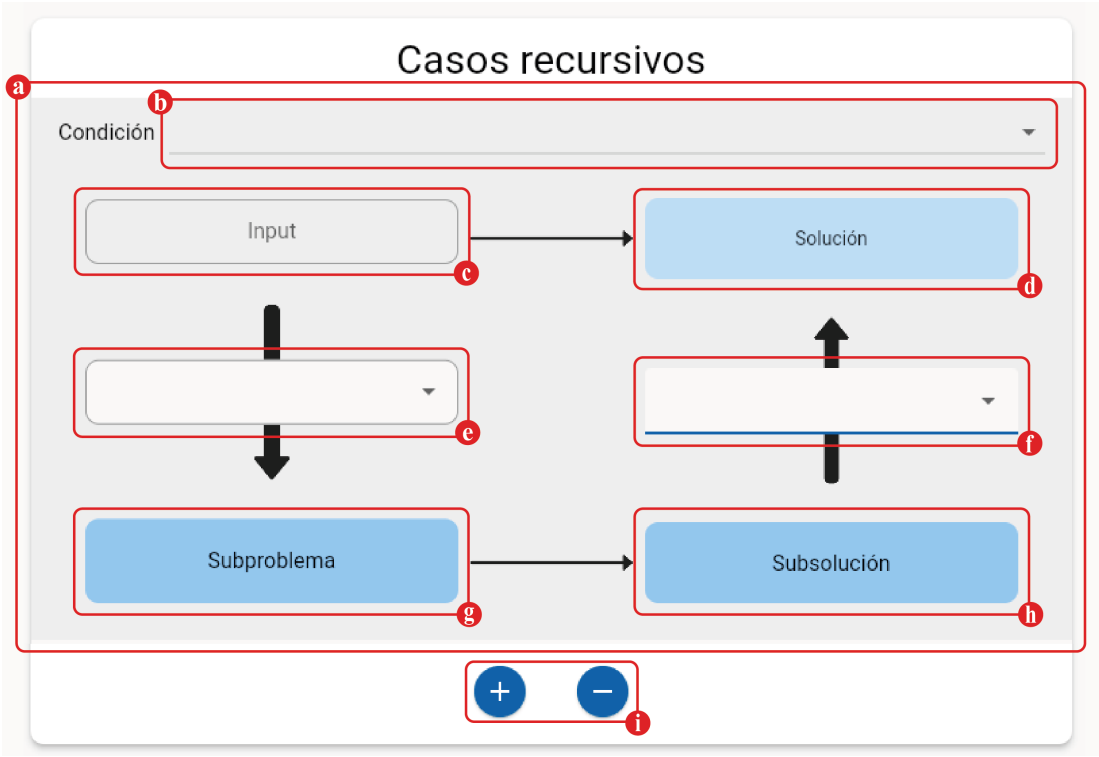


Figura 4.3: Caso recursivo.

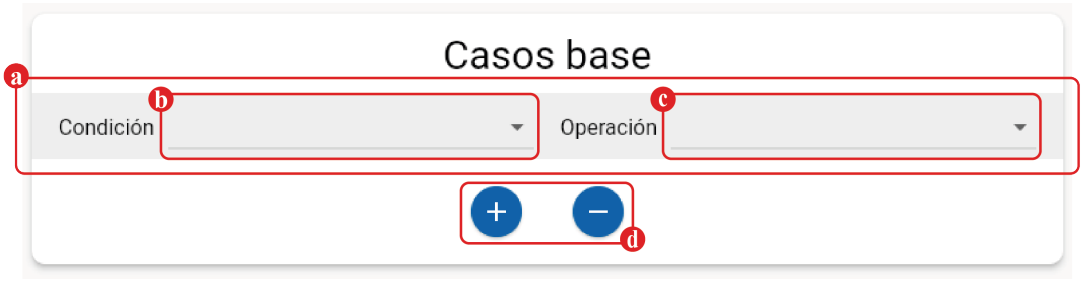


Figura 4.4: Caso base.

Code

Dentro de la carpeta “Code” se pueden encontrar los widgets necesarios para conformar la representación visual del código, tales como la consola, el botón para ejecutar el código y el botón para copiar el código.

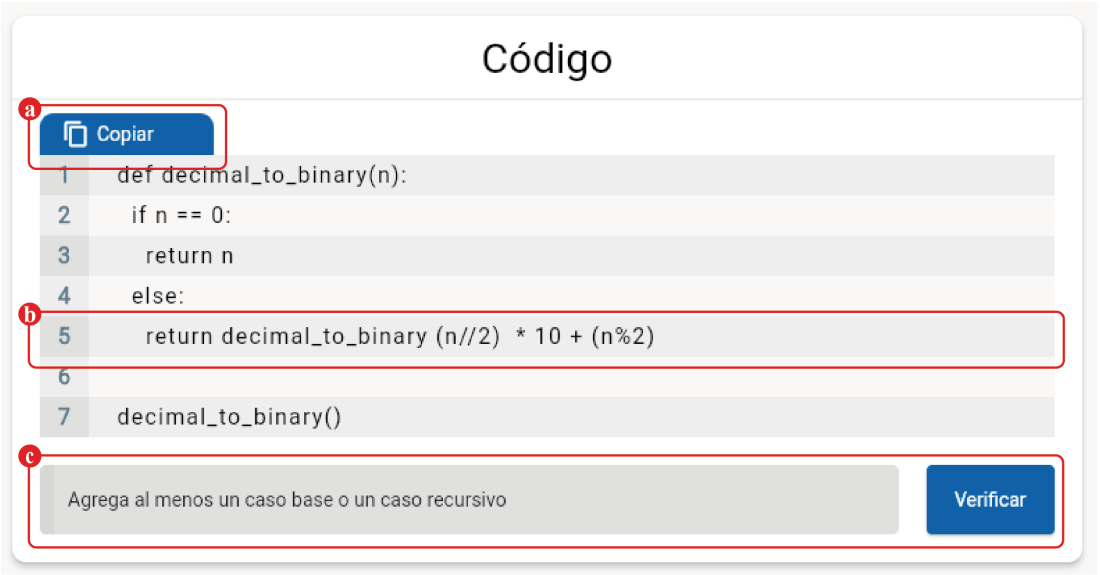


Figura 4.5: Widgets del código.

En Figura superior 4.5 se observan los widgets personalizados indicados y delimitados, que serán descritos brevemente a continuación:

1. La Figura 4.5a representa el widget que define el botón de copiar el código e implementa dicha funcionalidad.
2. Las lineas de código están representadas por el widget que se puede observar en la Figura 4.5b. Este contiene el índice de la línea y el código asignado a la misma.
3. Por último, la Figura 4.5c define el pie del contenedor, donde a la izquierda se encuentra la representación de la consola de la máquina virtual de Python; y a la derecha, el botón de verificación del código.

Layout

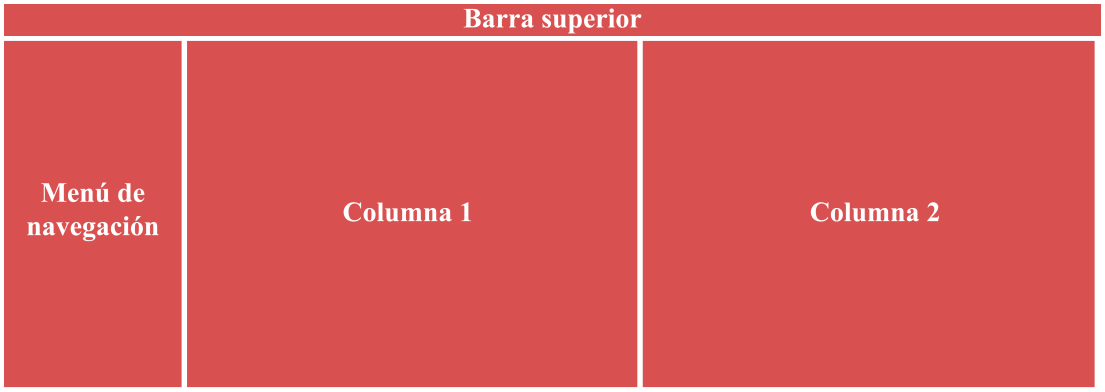
Layout es un conjunto de cuatro widgets que representan las distintas estructuras de la interfaz de usuario dependiendo del tamaño de la ventana donde se ejecute la aplicación.

El primer widget es **layout_constructor**, que recibe los elementos de cada pantalla de la aplicación e implementa los otros tres widgets de la carpeta que definen la estructura de la interfaz según el tamaño de la ventana. **layout_constructor** alterna los widgets que implementa según el tamaño de la ventana. Así, la aplicación se adapta a cualquier pantalla en la que se ejecute.

Los widgets que implementa layout_constructor son:

desktop_scaffold: define la distribución de los widgets de la pantalla cuando esta es muy alargada, distribuyéndolos en una barra superior para indicar dónde nos encontramos y una sección inferior. A la izquierda se encuentra el menú de navegación y a la derecha, dos columnas que se utilizan para distribuir los widgets del ejercicio como se ve en la Figura 4.6.

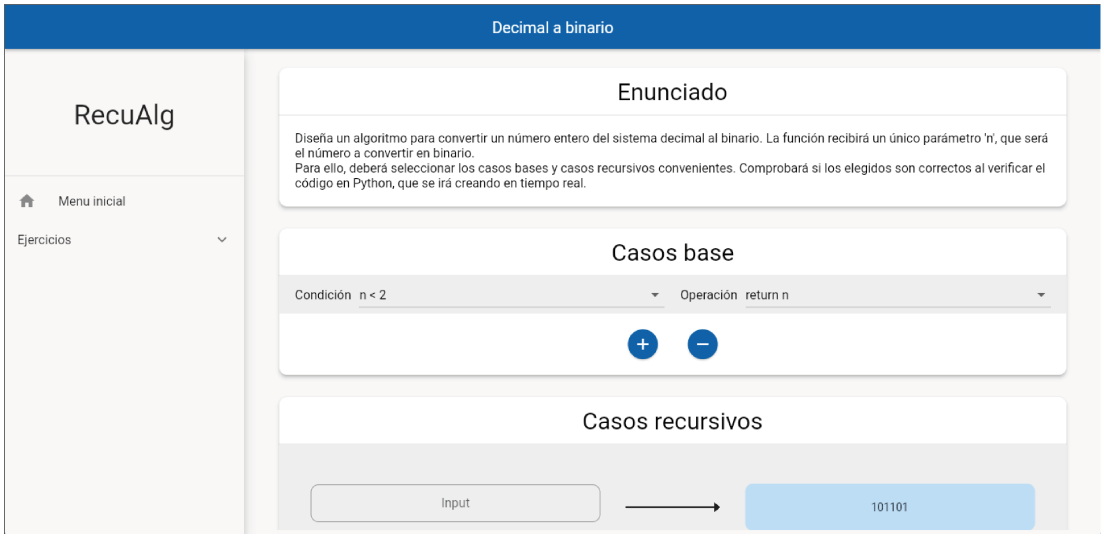
(a) desktop_scaffold



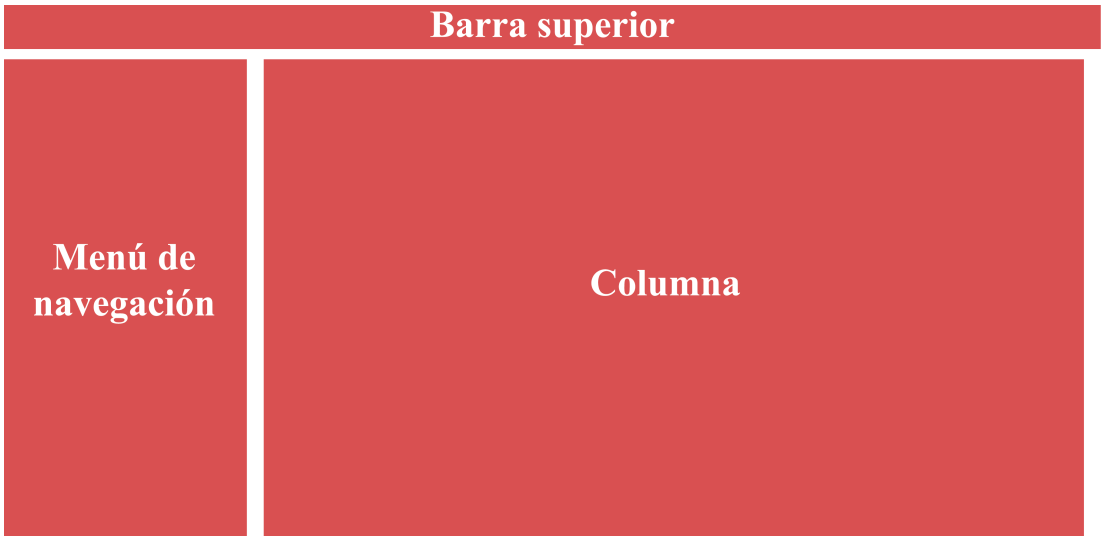
(b) Distribución desktop_scaffold

Figura 4.6: Widget desktop_scaffold.

tablet_scaffold: define la distribución de los widgets de la pantalla cuando tiene un ancho medio, distribuyéndolos en una barra superior para indicar dónde nos encontramos y en la parte inferior, se divide en la izquierda el menú de navegación y en la derecha una única columna para distribuir los widgets de los ejercicios como se ve en la Figura 4.7.



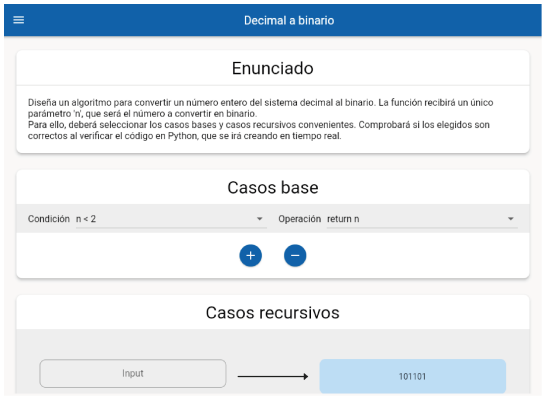
(a) tablet_scaffold



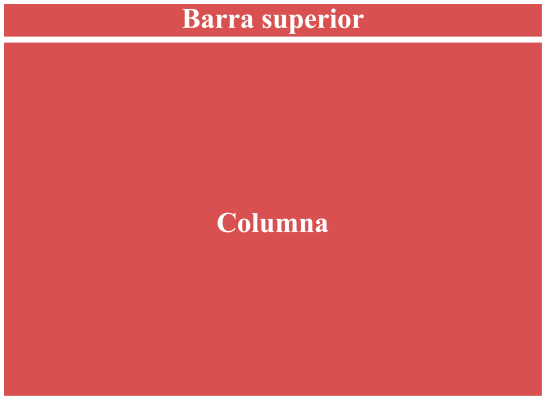
(b) Distribución tablet_scaffold

Figura 4.7: Widget tablet_scaffold.

mobile_scaffold: define la distribución de los widgets de la pantalla cuando tiene un ancho reducido, distribuyéndolos en una barra superior que indica dónde nos encontramos. En la izquierda de esta barra se encuentra un botón para desplegar el menú de navegación, que en esta distribución se encuentra oculto. En la parte inferior, se encuentra una única columna donde se distribuyen los widgets del ejercicio como se ve en la Figura 4.8.



(a) mobile_scaffold



(b) Distribución mobile_scaffold

Figura 4.8: Widget mobile_scaffold.

Others

La carpeta "Others" contiene una variedad de pequeños widgets que se utilizan a lo largo de toda la aplicación. Algunos de los widgets más destacados son:

1. La Figura A.1 representa el widget que define el menú de navegación de la aplicación, el cual es una columna que muestra el nombre de la aplicación, debajo de esta se encuentra el botón para acceder al menú de selección de ejercicios (el menú principal) y la lista de ejercicios para poder acceder de manera rápida a ellos.

2. El widget que define el contenedor de las distintas secciones de la pantalla de resolución de ejercicios (enunciado, casos base, casos recursivos y código) es representado por la Figura 4.9. Este widget implementa:

   a) El título del contenedor.

   b) El contenido del widget que puede ser cualquier otro widget como el enunciado, los contenedores de casos o el código.

   c) El pie del widget ofrece la opción de alojar otros widgets adicionales, lo que resulta especialmente útil si se desea incluir botones para agregar o eliminar casos, en el caso de que el contenedor albergue casos base o recursivos. También es posible agregar la consola del código Python en caso de que el contenedor albergue código.

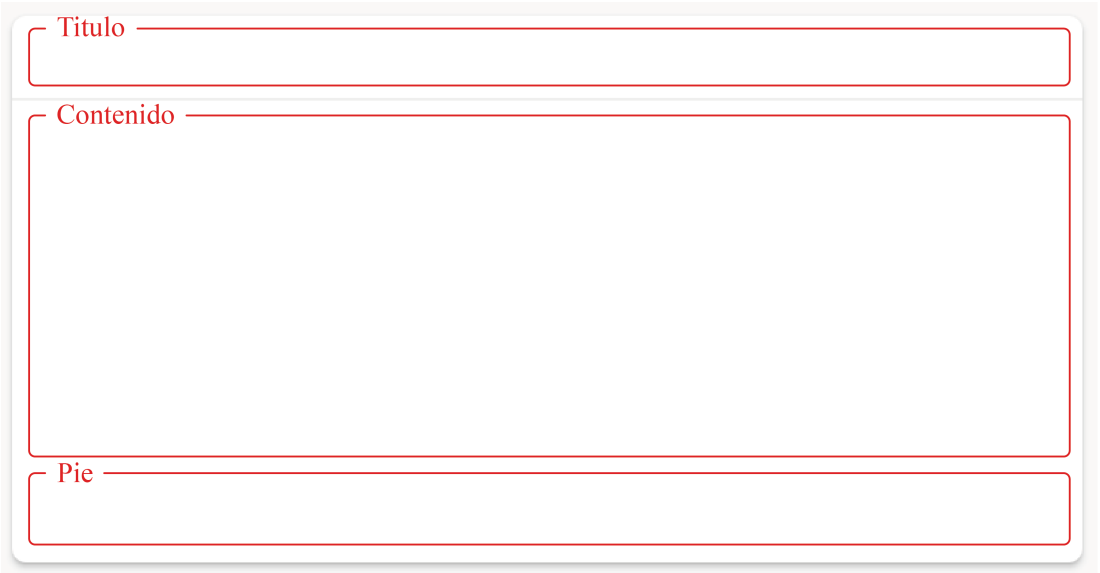


Figura 4.9: Widgets contenedor.

3. El selector de ejercicio como se puede observar en la Figura 4.10, este widget implementa el título del ejercicio, la imagen que lo representa y el botón para ir a la resolución del ejercicio seleccionado. La información de los ejercicios la obtiene del "exercises_provider".

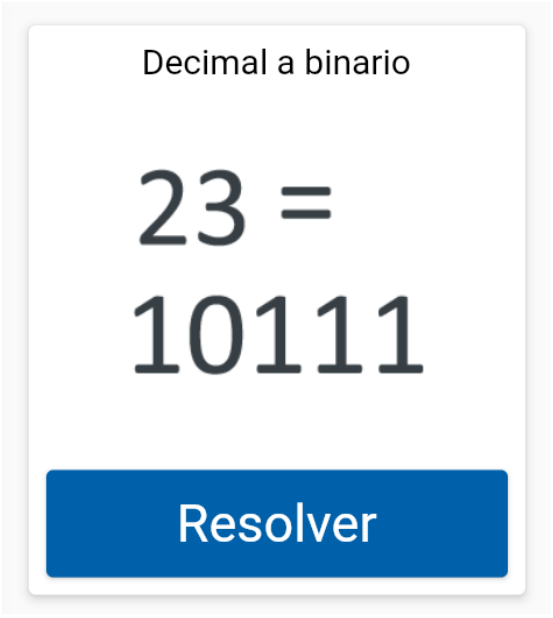


Figura 4.10: Selector de ejercicio.

4.4.2. Models

La carpeta "models" de un proyecto de software contiene las clases de modelo utilizadas en la aplicación para representar los datos. En RecuAlg, la carpeta "models" concretamente contiene:

- Las clases que definen los casos para comprobar la validez de las respuestas a los ejercicios.
- Las clases que definen las opciones dentro de los menús desplegables. Estas opciones almacenan tanto el texto que se mostrará como el código en Python que representa dicha opción para poder generar el código del algoritmo en Python.
- Las clases que definen los ejercicios.

4.4.3. Screens

En la carpeta "screens" se encuentran las definiciones de la interfaz de las dos ventanas que se encuentran en la aplicación, la ventana de ejercicio y la de selección de ejercicios, así como sus controladores que se encargan de implementar la lógica. Ambas pantallas implementan el widget personalizado "layout_constructor.dart", que permite reposicionar los distintos widgets de estas pantallas según el tamaño de la ventana donde se ejecuta la aplicación.

4.4.4. Utils

La carpeta "util" contiene dos scripts importantes para la aplicación:

- El primero implementa la máquina virtual de Python para poder ejecutar el código generado en los ejercicios.
- El segundo se encarga de la transformación de los inputs de los casos recursivos, que son representados en cadenas de texto, al o los tipos de datos requeridos para calcular la solución de los distintos ejercicios.

4.4.5. Assets

La carpeta "assets" contiene los recursos gráficos de la aplicación, como imágenes e iconos. En el caso específico de RecuAlg, esta carpeta contiene:

- Los iconos utilizados en la interfaz de usuario para representar visualmente los ejercicios en la pantalla de selección de ejercicios.
- Las imágenes de las flechas que se usan para la representación gráfica del esquema de los casos recursivos.
- El archivo de imagen utilizado como favicon de la aplicación.

4.4.6. Constants

Por último, la carpeta "constants" contiene constantes que se utilizan en todo el proyecto, como la definición de la paleta de colores y el estilo de los textos.

4.5. Utilización de GetX

Cabe mencionar con y donde se ha usado la librería GetX ya que esta ha resultado de gran ayuda a la hora de la realización del proyecto por los motivos por las características mencionadas en la Sección 4.3.4.

4.5.1. Configuración

Los pasos necesarios para usar GetX en la aplicación Flutter se describen a continuación:

1. Instalar el paquete "get" e importarlo en el archivo principal del proyecto.
2. En el método de la clase "MyApp", que normalmente usa "MaterialApp" para acceder a los widgets predefinidos y a las clases esenciales de la aplicación, se debe utilizar "GetMaterialApp" en lugar de "MaterialApp" para aprovechar las características adicionales de GetX.

Para utilizar los controladores de GetX:

1. La clase que vaya a ser un controlador debe extender de "GetxController".

Para utilizar el gestor de escenas de GetX:

1. Es necesario especificar las rutas de navegación de la aplicación a través de la propiedad "getPages", que recibe una lista de "getPage()" con dos argumentos: el nombre que se le da a la ruta y una función anónima que devuelve la página asociada a la ruta.

2. Luego, desde otro script que importe el paquete "get", se pueden llamar, entre otras funciones, a las siguientes:

   a) **Get.to:** Este método permite navegar a una nueva pantalla pasando una instancia de la pantalla como parámetro.

   b) **Get.off:** Este método permite navegar a una nueva pantalla, eliminando la pantalla anterior de la pila de rutas, es decir, no se puede volver atrás con la flecha de retroceso.

   c) **Get.offAll:** Este método permite navegar a una nueva pantalla y eliminar todas las rutas anteriores de la pila. Esto significa que el usuario no puede regresar a la pantalla anterior.

   d) **Get.offAll:** Este método permite navegar a una nueva pantalla y eliminar todas las rutas anteriores de la pila. Esto significa que el usuario no puede regresar a la pantalla anterior.

   e) **Get.back:** Este método permite regresar a la pantalla anterior en la pila de rutas.

4.5.2. Gestión de escenas

El gestor de escenas de GetX es una herramienta que permite crear y gestionar varias escenas en una aplicación Flutter. Básicamente, cada escena es una pantalla o una vista que el usuario puede ver y manipular. En el caso de RecuAlg, la aplicación tiene dos escenas: la primera es el selector de ejercicios que funciona como pantalla principal y la segunda escena es la de resolución de ejercicios. Para utilizar el gestor de escenas de GetX, se han definido dos rutas de la aplicación: una para la pantalla de selección de ejercicios y otra para la pantalla de resolución de ejercicios. Para navegar entre estas pantallas, los botones cuentan con llamadas a "Get.offAllNamed" para navegar a una ruta específica y eliminar todas las rutas anteriores de la pila.

4.5.3. Controladores

Los controladores de GetX se utilizan para separar la lógica de la vista en una aplicación Flutter, permitiendo una mejor organización y mantenimiento del código. Esto ayuda a hacer que la aplicación sea más escalable y fácil de probar, ya que los controladores pueden ser reutilizados en diferentes partes de la aplicación. En el controlador, además de ayudar a separar la lógica de la vista y de permitir definir las variables y funciones que puede realizar, también se pueden definir variables observables que se pueden observar mediante el uso de los widgets Obx() o Rx() y que se actualizan automáticamente cuando el valor de una variable observable cambia.

4.5.4. Gestor de dependencias

El gestor de dependencias de GetX es una herramienta que te permite manejar la creación, inyección y manejo de instancias de una forma sencilla y eficiente en tu aplicación Flutter. Con este gestor, puedes instanciar y obtener cualquier clase de forma automática y eficiente en cualquier lugar de tu aplicación.

El principio básico detrás del gestor de dependencias es que, en lugar de crear manualmente una instancia de una clase, se delega esta tarea al gestor de dependencias. Esto permite que el gestor de dependencias se encargue de las dependencias de la clase y de la resolución de estas dependencias.

El gestor de dependencias de GetX ofrece una serie de funciones y métodos para manejar la creación y manejo de instancias. Algunas de las funciones principales incluyen:

1. **Get.lazyPut:** esta función se utiliza para crear una instancia de una clase de forma perezosa (lazy). La instancia solo se creará cuando se solicite por primera vez.
2. **Get.put:** esta función se utiliza para crear una instancia de una clase y almacenarla en la memoria caché del gestor de dependencias. La instancia se creará inmediatamente y se almacenará en caché para su uso posterior.
3. **Get.find:** esta función se utiliza para recuperar una instancia previamente creada de una clase. Si la instancia aún no se ha creado, se creará y almacenará en caché antes de ser devuelta.

En el proyecto, el gestor de dependencias se ha utilizado para instanciar los controladores de Getx usados en la aplicación, como el controlador de la resolución de ejercicios y el controlador que maneja la máquina virtual de Python. Estos se han instanciado mediante “Get.lazyPut” para crear la instancia cuando esta vaya a ser usada.

Para recuperar los controladores se ha usado “Get.find” pasándole el tipo de clase del controlador que se quiere recuperar. Se ha hecho de esta manera ya que, por ejemplo, solo se tiene un controlador de la máquina virtual de Python y por ende no es necesario especificar de otra manera. En el caso del controlador de los ejercicios, no ha sido necesario ya que antes de instanciar el controlador, este se ha inicializado en una variable.

4.6. Ejercicios implementados

En este apartado se presentarán los ejercicios implementados en RecuAlg, junto con el razonamiento al que se espera que el usuario llegue para resolver cada

uno de ellos.

Longitud de una lista

El razonamiento consiste en contar la cantidad de elementos en una lista de forma recursiva. Si la lista está vacía, se devuelve 0. Si la lista no está vacía, se suma 1 al resultado y se llama a la función nuevamente con la lista sin el primer elemento. Esto se repite hasta que la lista esté vacía.

Máximo común divisor

El razonamiento se basa en el algoritmo de Euclides. Se divide el número más grande entre el número más pequeño. Si el residuo es 0, entonces el máximo común divisor es el número más pequeño. Si el residuo no es 0, se llama recursivamente a la función con el número más pequeño y el residuo como argumentos.

Multiplicación

El razonamiento es simple y se basa en el concepto matemático de multiplicación. Para multiplicar dos números, se puede pensar en sumar el primer número consigo mismo el segundo número de veces. Si el segundo número es 0, se devuelve 0. Si el segundo número no es 0, se suma el primer número y se llama recursivamente a la función con el primer número y el segundo número menos 1.

Es primo

El razonamiento implica verificar si un número es divisible por algún otro número menor que él. Si el número es menor o igual a 2, se devuelve True si es igual a 2 y False si es menor que 2. Si el número es divisible por algún divisor, se devuelve False. Si el divisor al cuadrado es mayor que el número, se devuelve True. De lo contrario, se llama recursivamente a la función con el número y el siguiente divisor.

Cálculo del n-ésimo término de Fibonacci

El razonamiento se basa en la definición recursiva de la secuencia de Fibonacci. Si el número es 0 o 1, se devuelve el número mismo. De lo contrario, se llama recursivamente a la función con los dos términos anteriores sumados.

Mayor valor de una lista

El razonamiento implica comparar el primer elemento de la lista con el mayor valor del resto de la lista. Si la lista tiene solo un elemento, se devuelve ese elemento. De lo contrario, se compara el primer elemento con el resultado recursivo obtenido al llamar a la función con el resto de la lista y se devuelve el valor máximo.

Suma de los n primeros números

El razonamiento se basa en la suma recursiva de los números desde n hasta 1. Si n es igual a 1, se devuelve 1. De lo contrario, se suma n al resultado recursivo de la suma de los números desde n-1 hasta 1.

Suma de una lista

El razonamiento consiste en sumar recursivamente los elementos de una lista. Si la lista está vacía, se devuelve 0. De lo contrario, se suma el primer elemento con el resultado recursivo de la suma de los elementos restantes de la lista.

Evaluación de un polinomio

Uno de los razonamientos posibles es el basado en la evaluación del polinomio utilizando el método de Horner. Se utiliza un enfoque recursivo donde se multiplica el coeficiente del término más alto por la potencia de x y se suma al resultado recursivo de evaluar el polinomio con los coeficientes restantes.

Pasar de decimal a binari

El razonamiento implica dividir el número decimal entre 2 y obtener el residuo. Luego se llama recursivamente a la función con el cociente obtenido hasta que el cociente sea 0. El resultado se obtiene concatenando los residuos en orden inverso.

Cálculo de la potencia

El razonamiento se basa en la propiedad de la potencia que implica multiplicar la base por sí misma el exponente de veces. Si el exponente es 0, se devuelve 1. De lo contrario, se multiplica la base por el resultado recursivo de la potencia de la base y el exponente menos 1.

Suma lenta

El razonamiento consiste en sumar dos números utilizando un enfoque recursivo que imita la suma manual. Se incrementa el primer número en 1 y se decrementa el segundo número en 1 hasta que el segundo número sea 0. El resultado es el primer número incrementado por la suma recursiva de los dos números restantes.

Inversión de una cadena de texto

El razonamiento para invertir una cadena de forma recursiva consiste en que si la cadena está vacía, se devuelve una cadena vacía. De lo contrario, se llama recursivamente a la función con la cadena sin el primer carácter y se concatena el primer carácter al resultado.

Evaluar si una cadena de texto es palíndromo

El razonamiento para verificar si una cadena es igual a su inversa consiste en que si la cadena tiene 0 o 1 caracteres, se devuelve True. De lo contrario, se verifica si el primer y último carácter son iguales, y se llama recursivamente a la función con la cadena sin el primer y último carácter.

Evaluar si dos cadenas de texto son iguales

El razonamiento implica comparar las cadenas carácter por carácter. Si ambas cadenas son vacías, se devuelve True. Si tienen diferente longitud, se devuelve False. De lo contrario, se compara el primer carácter de ambas cadenas y se llama recursivamente a la función con las cadenas sin el primer carácter.

Evaluar si un número contiene un determinado dígito

El razonamiento implica verificar si el último dígito de un número es igual al dígito objetivo. Si el número es 0, se devuelve False. Si el último dígito es igual al dígito objetivo, se devuelve True. De lo contrario, se llama recursivamente a la función con el número dividido por 10.

Ordenar una lista de enteros mediante Merge sort

El razonamiento pasa por primero verificar si la lista a ordenar tiene menos de 2 elementos, en cuyo caso ya está ordenada y se devuelve tal cual. De lo contrario,

se divide la lista a la mitad y se aplica merge_sort recursivamente a cada mitad. Luego, se utiliza la función merge para combinar las dos listas ordenadas en una sola lista ordenada. La función merge compara los elementos de las dos listas y los va insertando en orden ascendente en la lista resultante. Este proceso de división y combinación se repetira hasta que todas las sublistas estén ordenadas y se obtenga la lista final ordenada.

Permutación de inversión de bits

Si la longitud de 'a' es menor o igual a 2, simplemente hay que devolver la lista original. De no ser asi se tendrá que dividir la lista por la mitad y recursivamente aplicar la función 'invbitperm'. Tras esto se podrá intercalar los elementos de las dos mitades utilizando la función 'interleave'. Al repetir este proceso de manera recursiva se realizará la permutación de inversión de bits.

5

Líneas futuras

A continuación se presentan ciertas mejoras que incrementarían el valor de la aplicación actual y dos posibles enfoques que la aplicación podría adoptar. Cada una de estas vías podría ser complementaria a las mejoras generales planteadas previamente.

Entre las mejoras generales, se destacan las siguientes:

1. Adición de una mayor cantidad de ejercicios.
2. Soporte para múltiples idiomas.
3. Opción de registro para guardar el progreso y llevar un seguimiento del aprendizaje mediante la recopilación de datos..

Entre los dos enfoques que podría tomar la aplicación en un principio se encuentra el de convertirse en una plataforma educativa y el de incorporar herramientas de gamificación.

5.1. Plataforma educativa

Una posible vía futura para RecuAlg es transformarla en una plataforma educativa. En concreto, se espera que la aplicación no se limite a proporcionar únicamente ejercicios de algoritmos recursivos, sino que ofrezca la posibilidad a los

estudiantes de registrarse en la plataforma para almacenar su progreso y evolución. Además, se podrían ofrecer herramientas para que los docentes puedan inscribirse y supervisar el aprendizaje de sus alumnos, añadir nuevos ejercicios personalizados según las necesidades y responder a las dudas planteadas por los estudiantes dentro de la propia aplicación. Se sugiere incorporar un sistema de evaluación para que los estudiantes puedan medir su propio progreso y los profesores puedan realizar un seguimiento del rendimiento de sus estudiantes. Cabe destacar que la plataforma no se limitaría únicamente a los algoritmos recursivos, sino que podría abarcar la enseñanza de todo tipo de algoritmos. Además, se podrían introducir una gran cantidad de ejercicios y permitir cambiar el idioma de la aplicación para ayudar a un mayor número de estudiantes.

5.2. Gamificación

Otra posible vía futura para RecuAlg es la incorporación de herramientas de gamificación. Aunque esta vía se descartó en un primer momento debido a que no se consideró relevante para el alcance inicial del proyecto.

La gamificación puede ser una herramienta poderosa para motivar a los estudiantes y hacer que el proceso de aprendizaje sea más divertido e interesante. Además, puede fomentar la competencia entre estudiantes, lo que podría motivarlos a esforzarse más para mejorar su rendimiento.

Se podrían incorporar recompensas o insignias por completar ejercicios o por alcanzar ciertos niveles de progreso, con el objetivo de estimular el aprendizaje y el compromiso de los estudiantes con la plataforma.

La incorporación de herramientas de gamificación en RecuAlg podría hacer que la aplicación sea más atractiva y efectiva para los estudiantes, lo que podría resultar en un mayor aprendizaje y compromiso con el proceso educativo.

6

Conclusiones

Tras el desarrollo de RecuAlg se puede concluir que se han alcanzado los objetivos planteados en este Trabajo de Fin de Grado (TFG). La aplicación ha sido diseñada con el propósito de enseñar el diseño de algoritmos recursivos de forma visual, basándose en la metodología expuesta en el artículo titulado *Problems in Comprehending Recursion and Suggested Solutions* [5] y en el libro *Introduction to Recursive Programming* [6].

La inclusión de una amplia variedad de ejercicios, muchos de los cuales se utilizan en la enseñanza de la recursividad en las asignaturas de Diseño y Análisis de Algoritmos e Introducción a la Programación, permite a los usuarios adquirir, practicar y fortalecer sus conocimientos en el diseño de algoritmos recursivos. Además, la retroalimentación en tiempo real posibilita la detección y corrección inmediata de errores. Todo esto hace que la aplicación sea de gran ayuda tanto para los estudiantes que están aprendiendo sobre recursividad como para los profesores que enseñan esta materia.

Asimismo, la aplicación puede ser utilizada para poner a prueba y evaluar la efectividad pedagógica de la metodología descrita en el artículo y libro mencionados anteriormente, así como los diagramas presentados en el libro, en futuros experimentos de innovación docente.

Además, se ha logrado el objetivo de hacer que la aplicación esté disponible para cualquier usuario, en cualquier lugar y dispositivo, gracias a su publicación en un sitio web público. También se ha optimizado para ser compatible con los principales navegadores web para que todos los usuarios puedan acceder a esta.

Bibliografía

[1] B. Papa, "Recursion tree visualizer," Website. [Online]. Available: https://recursion.vercel.app/

[2] N. U. o. S. Dr. Steven Halim, "Visualgo," Website. [Online]. Available: https://visualgo.net/en

[3] P. Nick, "Codingbat," Website. [Online]. Available: https://codingbat.com/java

[4] G. LLC, "Flutter." [Online]. Available: https://flutter.dev/

[5] R. Sooriamurthi, "Problems in comprehending recursion and suggested solutions," *ACM SIGCSE Bulletin*, vol. 33, no. 10, 2001.

[6] M. Rubio-Sánchez, *Introduction to Recursive Programming*, 1st ed. Taylor & Francis, 2018.

[7] M. Syslo and A. Kwiatkowska, "Introducing students to recursion: A multi-facet and multi-tool approach," pp. 124–137, 2014.

[8] J. Á. Velázquez-Iturbide, "Recursion in gradual steps (is recursion really that difficult?)," pp. 310–314, 2000.

[9] D. Ginat and E. Shifroni, "Teaching recursion in a procedural environment—how much should we emphasize the computing model?" *ACM SIGCSE Bulletin*, vol. 31, no. 3, 1999.

[10] D. Wilcocks and I. Sanders, "Animating recursion as an aid to instruction," *Computers & Education*, vol. 23, no. 3, 1994.

[11] C. Kann, R. W. Lindeman, and R. Heller, "Integrating algorithm animation into a learning environment," *Computers & Education*, vol. 28, no. 4, 1997.

[12] D. Arroyo Cortés and J. Á. Velázquez Iturbide, "Srec versión 1.8: Manual de uso," Departamento de Lenguajes y Sistemas Informáticos I, Universidad Rey Juan Carlos, Tech. Rep., 2017.

[13] G. LLC, "Dart." [Online]. Available: https://dart.dev/

[14] B. Jonny, "Getx." [Online]. Available: https://pub.dev/packages/get

[15] blueloveTH, "Pocketpy." [Online]. Available: https://pub.dev/packages/pocketpy

Anexo

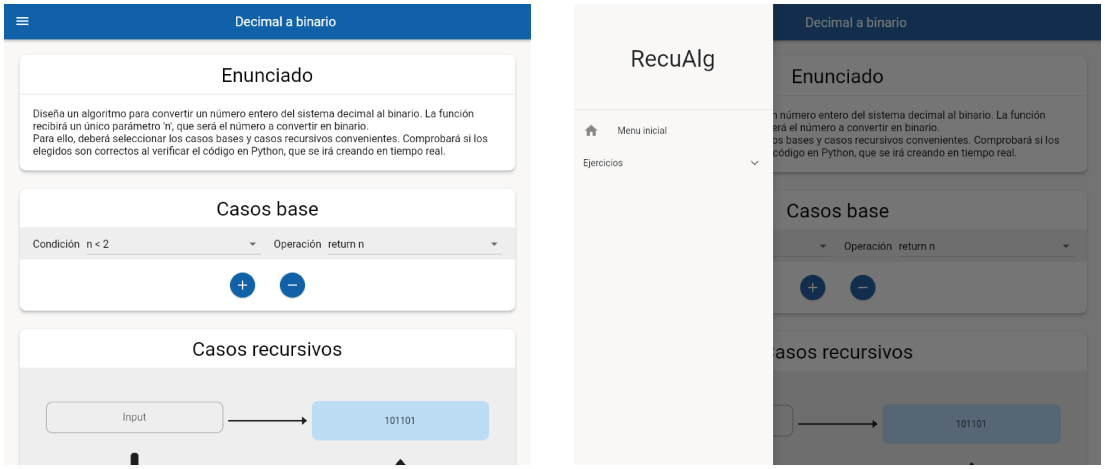
A

Manual de usuario

En este apartado se presenta el manual de usuario, que tiene como objetivo guiar al usuario a través de todas las acciones posibles dentro del sistema. Este manual proporciona una descripción detallada de cada función y acción, acompañada de explicaciones claras, imágenes ilustrativas y texto paso a paso para garantizar una comprensión completa y facilitar la interacción con el sistema.

Abrir menú de navegación

En caso de que el menú de navegación no esté desplegado como se puede ver en la Figura A.1a debido al reducido tamaño de la ventana donde es ejecutada la aplicación el usuario debe presionar sobre las tres líneas horizontales ubicadas en la barra superior a la izquierda, lo que desplegará el menú de navegación Figura A.1b.



(a) Menú de navegación cerrado. (b) Menú de navegación abierto.

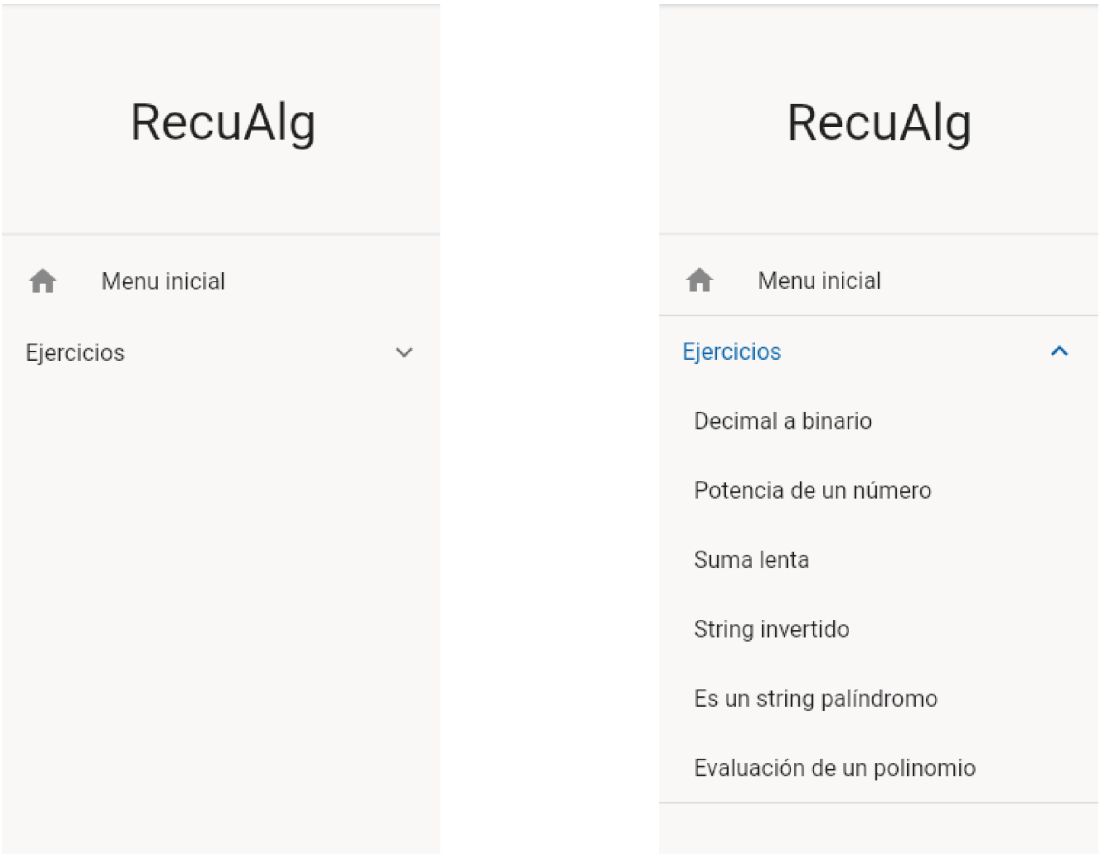
Figura A.1: Menú de navegación

Cerrar menú de navegación

Si el menú de navegación no se encuentra superpuesto a otros elementos de la interfaz, este no se podrá cerrar. Por el contrario si el menú de navegación se encuentre desplegado y superpuesto al resto de elementos de la interfaz veasé Figura A.1b el usuario podrá cerralo presionando fuera de este.

Desplegar listado de ejercicios

Para desplegar el listado de ejercicios, debemos ir al menú de navegación. En caso de que este no sea visible, primero deberemos abrirlo. Dentro del menú de navegación, se debe presionar el botón que dice "Ejercicios" veasé Figura A.2b. Al hacer esto, se desplegará el listado de ejercicios A.2a.



(a) Listado plegado (b) Listado desplegado

Figura A.2: Listado de ejercicios desplegable

Plegar listado de ejercicios

Para plegar el listado de ejercicios, debemos ir al menú de navegación. En caso de que este no sea visible, primero deberemos abrirlo. Dentro del menú de navegación, se debe presionar el botón que dice "Ejercicios" veasé Figura A.2a. Al hacer esto, se plegará el listado de ejercicios veasé Figura A.2b.

Selección de ejercicio

Para seleccionar un ejercicio, hay dos opciones disponibles:

- Desde la pantalla de selección de ejercicio A.3, el usuario puede interactuar con los botones que se encuentran en cada recuadro que representa un ejercicio.
- Desde el menú de navegación A.1b. En caso de que el menú de navegación no esté desplegado primero el usuario tendrá que desplegarlo y una vez visible, deberá desplegar el listado de ejercicios A.2 y posteriormente seleccionar uno de los ejercicios listados.

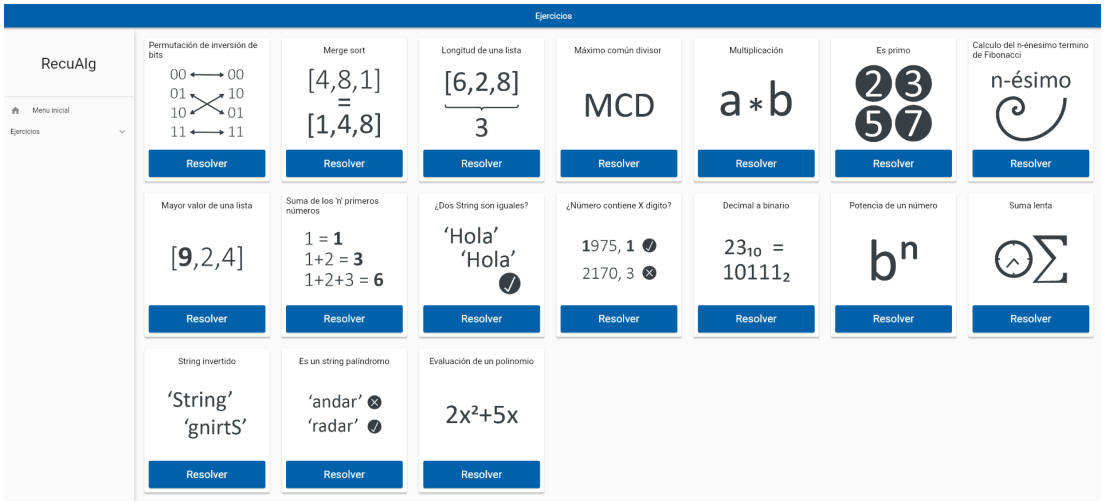


Figura A.3: Selector de ejercicios

Ir al menú de selección de ejercicio

Desde cualquier pantalla de la aplicación, el usuario puede acceder al menú de selección de ejercicio. Si el menú de navegación no está desplegado, primero deberá desplegarlo y una vez visible, el usuario deberá presionar sobre el botón que dice "Menú inicial" veasé Figura A.1.

Resolución de ejercicio

Para realizar cualquier ejercicio, el usuario debe seguir los siguientes pasos:

Casos base

Dentro del recuadro de casos base, el usuario puede realizar las siguientes acciones:

- Agregar caso base: Para hacerlo, debe presionar el botón azul con el símbolo "+" ubicado dentro del recuadro de casos base en la parte inferior A.4c.
- Eliminar caso base: Para eliminar el último caso base agregado, debe presionar el botón azul con el símbolo "-" ubicado dentro del recuadro de casos base en la parte inferior A.4d.
- Seleccionar condición: Para seleccionar la condición del caso base, el usuario debe presionar sobre el menú desplegable A.4a, lo que mostrará un pequeño recuadro con todas las opciones posibles. Luego, debe elegir una de ellas presionándola.
- Seleccionar operación: Para seleccionar la operación del caso base, el usuario debe presionar sobre el menú desplegable A.4b, lo que mostrará un pequeño recuadro con todas las opciones posibles. Luego, debe elegir una de ellas presionándola.

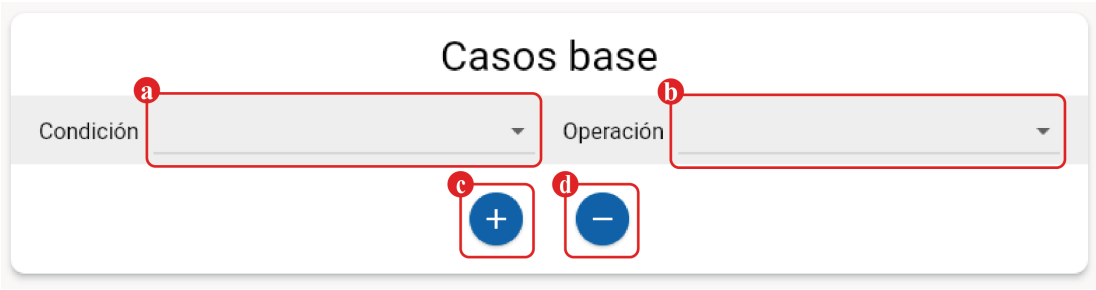


Figura A.4: Casos base

Casos recursivos

Dentro del recuadro de casos recursivos, se pueden realizar las siguientes acciones:

- Agregar caso base: Se agrega un caso recursivo presionando sobre el botón azul con el símbolo "+" situado en la parte inferior del recuadro de casos recursivos A.5h.
- Eliminar caso base: Se elimina el último caso recursivo agregado presionando sobre el botón azul con el símbolo "-" situado en la parte inferior del recuadro de casos recursivos A.5i.
- Seleccionar condición: Si se tienen más de dos casos recursivos, se deben seleccionar las condiciones de ejecución de todos los casos recursivos, excepto el último. Para ello, primero hay que presionar sobre el menú desplegable A.5a, lo que mostrará un pequeño recuadro con todas las opciones posibles para elegir. Luego, se elige una de ellas presionando sobre ella.
- Introducir input: Para introducir un input en el caso recursivo, primero se debe presionar sobre el recuadro de input A.5b y, posteriormente, se pueden introducir los valores de entrada para el ejercicio. Si el input es incorrecto, ya sea porque no cumple con la condición del caso recursivo o porque el input no corresponde con los valores que se le deben pasar, se mostrarán mensajes de error en los demás recuadros. Para introducir cadenas de texto, estas tienen que estar delimitadas por comillas simples (' '). Para introducir listas, estas deben estar delimitadas por corchetes ([ ]) y sus elementos deben estar separados por comas (,). Para introducir números decimales, se deben indicar con un punto (.) y para indicar más de un input, estos deben estar separados por comas.
- Seleccionar descomposición del problema: Para seleccionar la operación de descomposición del problema, primero hay que presionar sobre el menú desplegable A.5d, lo que mostrará un pequeño recuadro con todas las opciones posibles para elegir. Luego, se elige una de ellas presionando sobre ella.
- Seleccionar resolución del subproblema: Para seleccionar la operación de resolución del subproblema, primero hay que presionar sobre el menú desplegable A.5c, lo que mostrará un pequeño recuadro con todas las opciones posibles para elegir. Luego, se elige una de ellas presionando sobre ella.

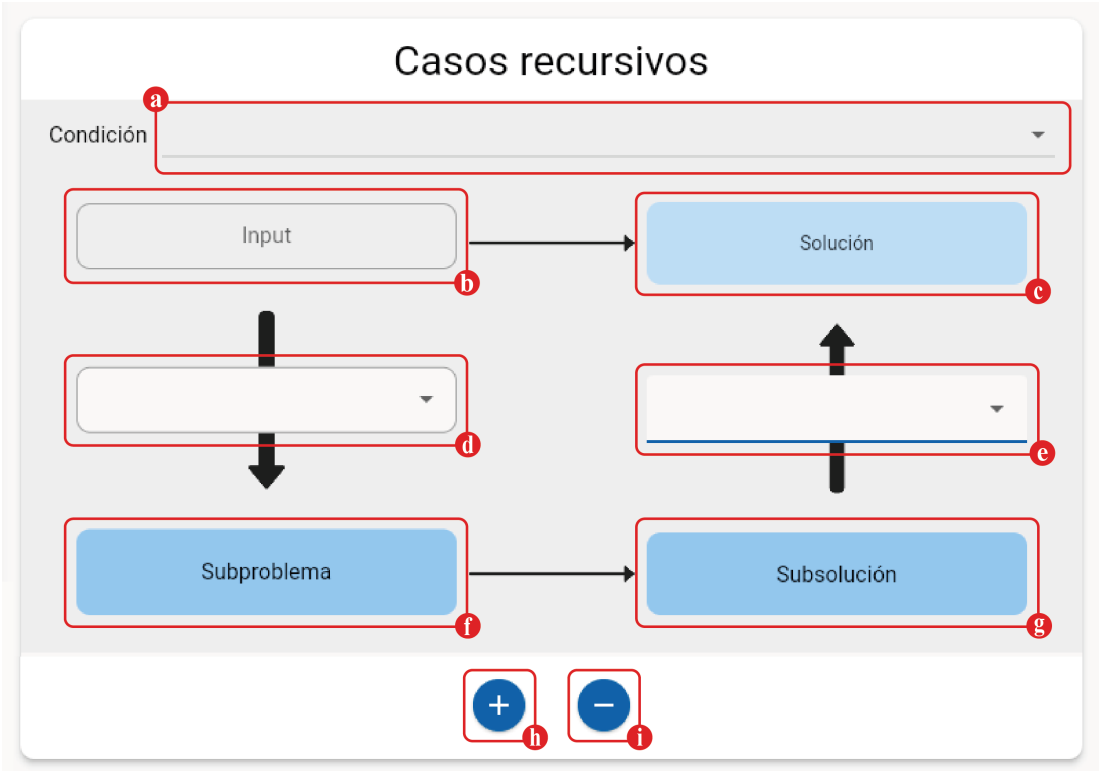
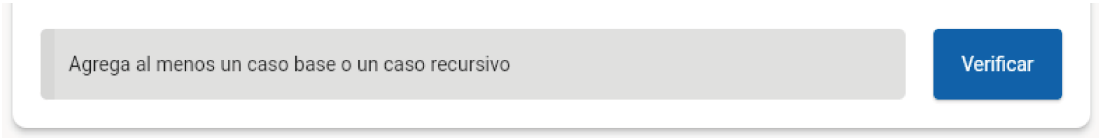


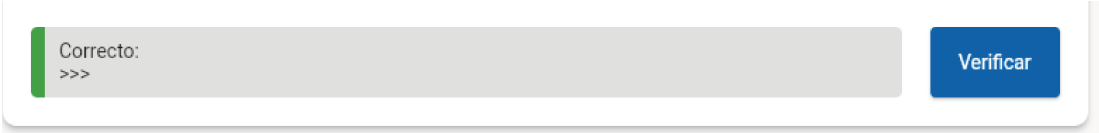
Figura A.5: Casos recursivos

Verificación del ejercicio

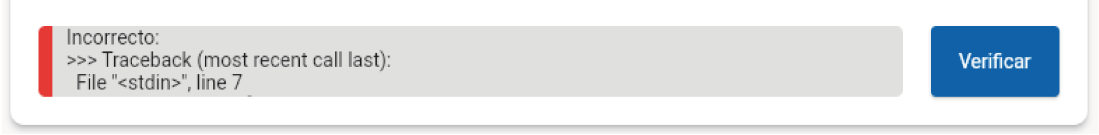
A medida que se agregan, eliminan y seleccionan opciones de los casos bases y recursivos, se generará en tiempo real el código Python para la resolución del problema.



(a) Consola por defecto



(b) Código correcto



(c) Código incorrecto

Figura A.6: Estados consola

Para verificar que el usuario ha agregado los casos bases y recursivos y ha seleccionado las opciones correctas para el ejercicio que está realizando, debe presionar el botón de verificar que se encuentra a la derecha de la consola veasé Figura A.7a. Esto hará que se ejecute el código, por defecto la consolo se muestra como se ve en la Figura A.6a, si el código ejecutado tiene errores, se mostrarán en la consola veasé Figura A.6c. En caso de que el código sea correcto, se indicará el resultado y qué ejercicio está bien veasé Figura A.6b.

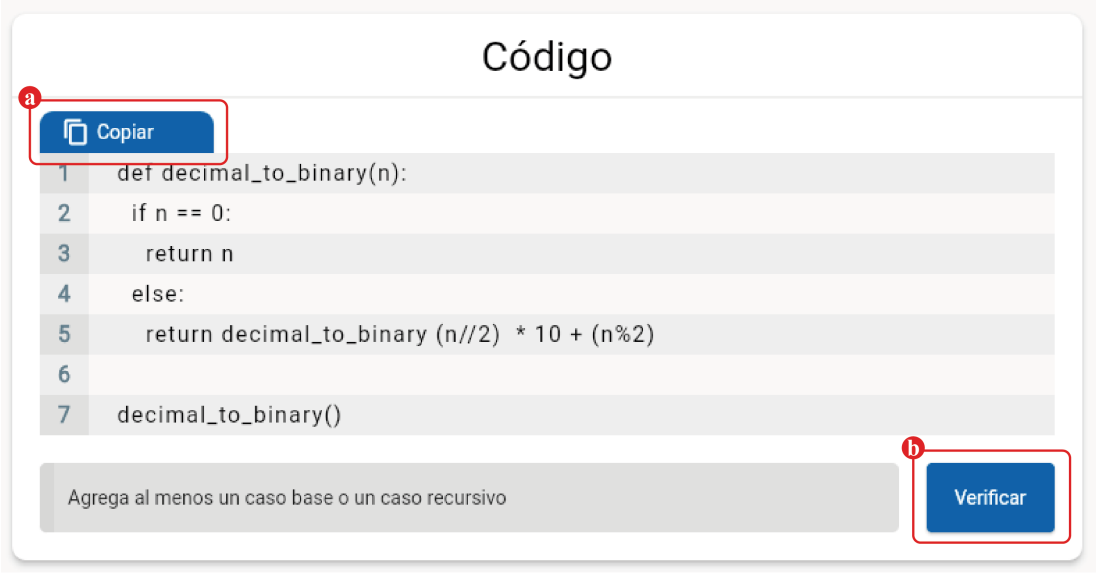


Figura A.7: Código

Copiar código en el portapapeles

Para copiar el código Python generado con nuestras elecciones, se debe presionar el botón que se encuentra sobre el código y que dice “copiar” A.7b. Esto copiará el código generado en el portapapeles del dispositivo donde se está ejecutando la aplicación.