UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Arqueología Informática: Diseño e implementación de la máquina Enigma en Scratch

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: Juan Antonio López Ramírez

Tutor: Xavier Molero Prieto

Curso 2018-2019

Resum

El present treball es centra en el disseny i implementació de la màquina Enigma, una màquina clàssica de xifrat utilitzada durant la Segona Guerra Mundial, mitjançant el llenguatge de programació Scratch.
Quan la màquina estigui finalitzada, estarà disponible a la pàgina web del Museu d'Informàtica de l'Escola Tècnica Superior d'Enginyeria Informàtica amb l'objectiu de demostrar la capacitat del llenguatge Scratch, fer que més usuaris comencin a usar-lo per aprendre programació de manera entretinguda i difondre el patrimoni cultural informàtic de les últimes dècades del segle XX.
Pel que fa a la memòria, primer es realitza un repàs del context històric i social al qual pertany la màquina Enigma. Després s'aprofundeix en el funcionament de la màquina, els seus usos, quins models es van arribar a construir i en què consisteixen cada un d'ells. Per acabar, es dissenya i s'implementa la màquina Enigma en Scratch amb l'objectiu de donar la mateixa funcionalitat que una màquina d'aquella època.

Paraules clau: màquina Enigma, Scratch, Museu d'Informática, programació, patrimoni cultural informàtic

Resumen

El presente trabajo se centra en el diseño e implementación de la máquina Enigma, una máquina clásica de cifrado utilizada durante la Segunda Guerra Mundial, mediante el lenguaje de programación Scratch.
Cuando la máquina esté finalizada, estará disponible en la página web del Museo de Informática de la Escuela Técnica Superior de Ingeniería Informática con el objetivo de demostrar la capacidad del lenguaje Scratch, hacer que más usuarios empiecen a usarlo para aprender programación de manera entretenida y difundir el patrimonio cultural informático de las últimas décadas del siglo XX.
Respecto a la memoria, primero se realiza un repaso dentro del contexto histórico y social al que pertenece la máquina Enigma. Después se ahonda en el funcionamiento de la máquina, sus usos, qué modelos se llegaron a construir y en qué consisten cada uno de ellos. Para terminar, se diseña y se implementa la máquina Enigma en Scratch con el objetivo de dar la misma funcionalidad que una máquina de aquella época.

Palabras clave: máquina Enigma, Scratch, Museo de Informática, programación, patrimonio cultural informático

Abstract

The current work focuses on the design and implementation of the Enigma machine, a classic encryption machine used during World War II, using Scratch as the programming language.
Once finished, it will be available on the Museo de Informática de la Escola Tècnica Superior d'Enginyeria Informàtica's website with the aim of demonstrating the ability of the Scratch language, to make more users start using it to learn programming in an entertaining way and disseminate the computer cultural legacy of the last decades of the 20th century.
Regarding the memory, first a review is made within the historical and social context to which the Enigma machine belongs. Afterwards, how the machine works, its uses, what models were built and what each of them consists of. To finish, the Enigma machine is

designed and implemented in Scratch with the aim of giving the same functionality as a machine of that time.

Key words: Enigma machine, Scratch, Museum of Informatics, programming, computer cultural legacy

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

En este capítulo explicaremos las causas que nos han llevado a escoger este tema para realizar el trabajo. Se aclarará el objetivo que se intenta alcanzar y se describirá al lector la estructura del proyecto que seguiremos durante todo el documento. Finalmente, habrá una sección donde se hablará de la bibliografía usada, sobre todo se ahondará en las herramientas y los recursos que hemos empleado para hacer el programa con el lenguaje de programación Scratch.

1.1 Motivación

La principal causa para realizar este proyecto es despertar el interés de la gente ya no solo en la máquina Enigma como uno de los mejores métodos de cifrado, sino también en la criptografía en general como ámbito de estudio.

El Museo de Informática de la Escola Tècnica Superior d'Enginyeria Informàtica de la Universitat Politècnica de València ofrece numerosas actividades culturales, tales como conferencias, exposiciones o una parte dedicada a la educación. En este ámbito se pueden encontrar talleres para acercar a los visitantes a la historia de la informática e introducirlos en el mundo de la programación. Es por ello que nosotros nos centraremos en el taller de Scratch, debido principalmente a que se trata de un lenguaje sencillo e interactivo que nos servirá de gran ayuda para realizar nuestro simulador de la Máquina Enigma.

Por tanto, se trata de un proyecto de divulgación, tanto de una técnica de cifrado clásica, como de un lenguaje de programación visual de carácter educativo.

1.2 Objetivos

En este trabajo nos disponemos a diseñar e implementar una máquina Enigma con todos sus componentes, tales como el clavijero o el panel de configuración interna de rotores. Además, nuestro simulador de la máquina Enigma incluirá los distintos modelos que se fabricaron de ella, los cuales son:

1. Wehrmacht - UKW = B
2. Wehrmacht - UKW = C
3. Kriegsmarine M3 - UKW = B
4. Kriegsmarine M3 - UKW = C

5. Kriegsmarine M4 - UKW = B
6. Kriegsmarine M4 - UKW = C

Los cuatro primeros modelos funcionan con una clave de tres rotores, mientras que los dos últimos con una clave de cuatro. La 'B' y la 'C' del final de cada modelo indican con qué reflector están trabajando.
El funcionamiento de los diversos componentes de la máquina Enigma, junto con otros conceptos que se mencionan en este apartado, serán explicados más adelante.

1.3 Estructura de la memoria

El trabajo realizado se compone de seis capítulos. En este apartado describiremos brevemente el contenido de cada uno de ellos.

- **Capítulo 1. Introducción al trabajo:** Se explican las motivaciones por las que se escogió este tema para realizar el trabajo; y los objetivos a alcanzar.
- **Capítulo 2. Contexto histórico:** En este capítulo se ahondará en los sucesos históricos que dieron lugar a la creación de la máquina Enigma, así como en las circunstancias que hicieron que este mecanismo llegase a ser un importante elemento en los acontecimientos de la Segunda Guerra Mundial. Además, se comentará la influencia que tuvo una vez terminada la guerra.
- **Capítulo 3. Funcionamiento de la máquina Enigma:** En esta parte se expondrá con detalle el funcionamiento interno de la máquina para cifrar, y por tanto se explicará el papel que desempeñan elementos importantes como el clavijero o la clave del mensaje, tanto en los modelos de tres rotores como en los de cuatro.
- **Capítulo 4. Scratch como entorno de programación:** Se verán de forma detallada las técnicas que nos proporciona este lenguaje para la creación de nuestro simulador.
- **Capítulo 5. Diseño e implementación de la máquina Enigma:** Aquí describiremos cómo hemos hecho el simulador de la máquina mediante las herramientas proporcionadas por Scratch; así como sus instrucciones de uso.
- **Capítulo 6. Conclusiones:** Por último se hará un breve resumen del funcionamiento del simulador realizado y comprobaremos si se han alcanzado los objetivos planteados en un principio.

1.4 Uso de la bibliografía

En este apartado de la memoria hablaremos de la bibliografía usada para poder llevar a cabo la realización del trabajo. En especial, vamos a centrarnos en la información que se ha obtenido de la web para poder exponer la historia y el funcionamiento de la máquina Enigma.

Toda la información referente a los fundamentos teóricos, aplicaciones y análisis de la seguridad de ciertos métodos criptográficos se ha obtenido de varios libros disponibles en la biblioteca de la propia escuela ([1], [2], [3], [4]). Además, la obra de Alan Turing en el desencriptado de las comunicaciones nazis fue consultada en un libro dedicado a su vida, su trabajo y su legado ([5]). Del mismo modo, un resumen de la máquina Enigma

se puede encontrar en la página web del Museo, en un apartado dedicado a la historia de la informática ([6]).

Para la redacción de la memoria se consultó, en su mayoría, el sitio web *Crypto Museum*[1], cuyo objetivo es preservar la historia de las distintas máquinas de cifrado que han existido ([8], [9], [10], [11], [12], [13], [14], [15]).

Por otro lado, los conocimientos sobre el funcionamiento de las distintas máquinas Enigma y sus procedimientos a la hora de cifrar o descifrar han sido adaptados de una página web[2] dedicada a la criptología y que cuenta con simuladores de diversas máquinas clásicas de cifrado ([16], [17]).

Por último, todo lo relacionado con Scratch se buscó en la enciclopedia libre dentro de su propia página web ([18], [19]). Además, el concepto de pensamiento computacional ha sido introducido en este trabajo tal como lo definió la informática teórica Jeannette Wing ([20]).

[1] https://www.cryptomuseum.com/index.htm
[2] http://users.telenet.be/d.rijmenants/index.htm

CAPÍTULO 2
Contexto histórico

La historia de la máquina Enigma empieza en 1915, con la invención de las máquinas de cifrado basadas en rotores. Este tipo de mecanismos surgieron alrededor de aquella época en países como Estados Unidos, Suecia, Países Bajos y Alemania.

2.1 Origen

A principios del siglo XX, la expansión de las telecomunicaciones aumentó la necesidad de crear métodos de cifrado automatizados y sustituir los métodos manuales [6].

Así, en 1915, dos oficiales navales holandeses, Theo A van Hengel y RPC Sprengler, produjeron máquinas de cifrado basadas en rotores para el Departamento de Guerra holandés [7]. Posteriormente, en 1917, inventores como el americano Edward Hebern o el holandés Hugo Koch crearon aparatos similares en sus respectivos países.

En 1918, justo al final de la Primera Guerra Mundial, el ingeniero alemán Arthur Scherbius inventó lo que hoy conocemos como la máquina Enigma. Después de años de intentar mejorar su invención y de haber obtenido la patente NL10700 de Hugo Koch para un dispositivo similar [8], la primera máquina Enigma vio la luz en 1923.

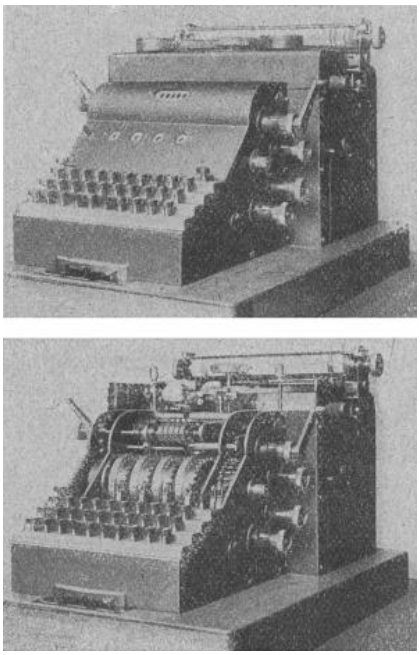

Figura 2.1: *Die Handelsmaschine* completa y vista desde dentro.

Esta máquina, llamada *Die Handelsmaschine* (véase la figura 2.1), imprimía directamente en una hoja de papel como una máquina de escribir. Un año más tarde le sucedió el *Die Schreibende Enigma* (véase la figura 2.2), que tenía un mecanismo de impresión mejorado, basado en la sustitución de la rueda de impresión giratoria, que no era fiable a altas velocidades, por barras tipográficas más comunes en las máquinas de escribir de la época [9].

Figura 2.2: *Die Schreibende Enigma*.

2.2 Del papel a la luz

Al principio, la máquina Enigma daba muchos problemas debido a su poca fiabilidad a la hora de imprimir, tanto con la rueda de impresión de los primeros modelos como con las barras tipográficas. Además, solo podía ser rentable en el mercado de gama alta porque su construcción era muy costosa.

Por este motivo, Scherbius desarrolló una máquina que producía su salida en un panel de luces en lugar de en papel. Este primer modelo fue diseñado en 1924 y recibió el nombre de *Gluhlampenmaschine* (máquina de luces) o, comúnmente denominada, Enigma A [10]. El mecanismo se encuentra alojado en una caja de madera y se parece bastante a los modelos Enigma posteriores, excepto que las teclas están dispuestas en orden alfabético (ABCDE...) en lugar del orden más común de las máquinas de escribir (QWERTZ...).

Más tarde, llegarían los modelos Enigma B (cuya novedad era que el rotor derecho avanzase automáticamente al pulsar una tecla [11]) y Enigma C (bastante similar al anterior, y sería la última versión que tuviese las teclas en orden alfabético [12]).

A diferencia del Enigma que imprimía a papel, las máquinas con panel de luces tenían un reflector que hacía que la máquina se correspondiera simétricamente (esto se explicará debidamente en el capítulo 3). Como resultado, los ajustes de la máquina para el cifrado y descifrado de textos eran idénticos, lo que mejoró enormemente su utilidad.

Figura 2.3: Modelo Enigma B con disposición de letras en orden alfabético.

2.3 Usos militares

En 1926, el diseño de Enigma fue mejorado drásticamente. Se desarrolló un nuevo chasis y se introdujo el diseño de teclado estándar alemán (QWERTZ...). También se hizo más accesible la tapa superior, de modo que era más fácil modificar los ajustes básicos como la clave del mensaje o la posición de los rotores. Además, el reflector podía ajustarse ahora en 26 posiciones diferentes y se situó a la izquierda de las tres ruedas de cifrado, por lo que a veces se piensa (erróneamente) que esta máquina es un Enigma de cuatro ruedas. Este modelo recibió el nombre de Enigma D [13] y llegó a convertirse en la base de la mayoría de máquinas posteriores.

Un año más tarde, se empezaron a desarrollar nuevos modelos partiendo de la Enigma D. En primer lugar estaba la Enigma Comercial, que más tarde se conocería como Enigma K. Todas las máquinas Enigma comerciales tenían un mecanismo simple de rotación de ruedas, parecido al cuentakilómetros de un coche. La rueda de la derecha realiza un solo paso en cada pulsación de tecla. Después de que la rueda de la derecha haya completado una vuelta completa, la rueda central da un solo paso y así sucesivamente.

En aquella época, el ejército alemán (Reichswehr, conocido más tarde como Wehrmacht) comenzó a mostrar interés por este tipo de máquinas. Así, se desarrolló para este una variante especial del Enigma D comercial, con tres rotores de cifrado y un reflector fijo. Además, esta nueva máquina tenía una placa de conexiones, también llamada clavijero, que añadía una capa extra al cifrado y estaba situada en la parte delantera. El primer prototipo estuvo listo en 1927 y fue conocido como Enigma D del Reichswehr, con una placa de conexiones de un solo extremo.

La versión final se terminó en 1932 con un clavijero mejorado de doble terminación, para que pudiera ser conectado de extremo a extremo. Esta versión era conocida por el Reichswehr (ahora: Wehrmacht) como el Enigma-I (véase la figura 2.4) y solo estaba disponible para el ejército. Hasta ese momento, todos los modelos comerciales de Enigma estaban disponibles gratuitamente en el mercado internacional. Esto cambiaría en 1932, cuando el ejército alemán reclamó los derechos exclusivos de la máquina. A partir de entonces, todas las ventas comerciales e internacionales debían ser aprobadas por el ejército.

Además del Enigma-I, en 1929 se hizo una nueva versión del *Schreibende Enigma* llamada Enigma H y conocida por el ejército como Enigma-II (véase la figura 2.4). Esta se vendió al ejército húngaro, pero nunca fue muy popular debido a su alto precio. Aparte, los alemanes también seguían vendiendo máquinas Enigma a los suizos y al ejército holandés. Este último compró algunos modelos hasta 1938.

A mediados de los años 30, cuando el ejército alemán se preparaba para la guerra, comenzó a encargar máquinas Enigma-I en grandes cantidades para el Heer (Ejército de tierra) y la Luftwaffe (Fuerza aérea). Para la Kriegsmarine (Armada alemana) se desarrolló un modelo similar y compatible con el Enigma-I. Se conocía como el Enigma M1 (1934), al que le seguirían el Enigma M2 (1938) y finalmente el Enigma M3 (1940).

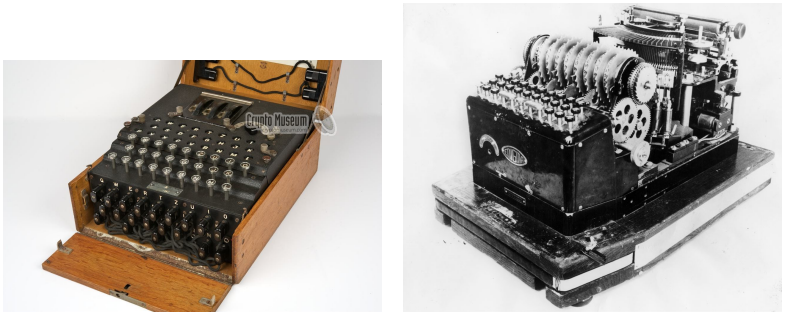

Figura 2.4: Enigma-I (1927) y Enigma-II (1929), modelos del ejército alemán.

2.4 Segunda Guerra Mundial. Descifrando Enigma

En 1933, la Oficina Polaca de Cifrado obtuvo acceso a los procedimientos operativos de la máquina Enigma que utilizaba el ejército alemán. Hans Thilo Schmidt, un playboy alemán que trabajaba en la Oficina Alemana de Cifrado, vendió información al servicio secreto francés. Los franceses, que dieron a Schmidt el nombre en clave Asché, reenviaron dicha información a los polacos para que pudiesen reconstruir la máquina.

A partir de ese año, los polacos interceptaron y descifraron gran parte de las comunicaciones de radio de los alemanes. En 1938 se observaba un aumento en el número de mensajes enviados por los alemanes y parecía claro que Alemania se estaba preparando para la guerra.

Durante todo este tiempo, los alemanes habían estado utilizando un *Grundstellung* (configuración básica) común para todo el tráfico de mensajes mediante la Enigma. Sin embargo, en septiembre de 1938, se abandona este procedimiento. Alrededor de la misma época, dos nuevos rotores (IV y V) se suman a los tres existentes, multiplicando por diez el número máximo de ajustes posibles.

Mientras tanto, los polacos habían construido su propio equivalente de la Wehrmacht Enigma con un panel de conexiones añadido en la parte de atrás. El cableado de los dos rotores adicionales fue recuperado y se añadieron adecuadamente en la réplica polaca. Con la guerra inminente, los polacos comenzaban a buscar formas de sacar sus conocimientos del país antes de que fuera demasiado tarde.

Dillwyn Knox fue un papirólogo brtánico que, desde 1925, había estado tratando de romper el código de la máquina Enigma y tuvo su primer éxito en abril de 1937, cuando descifró la Enigma K (una de las versiones comerciales de la Enigma D), que el gobierno alemán había proporcionado a Francisco Franco durante la guerra civil española. En 1938, cuando Alemania comenzó a utilizar la Enigma para la comunicación entre Alemania y España, Knox montó un ataque para intentar romper la Enigma militar, pero no tuvo éxito porque no fue capaz de resolver el cableado del disco de entrada.

Ese mismo año, el Servicio de Inteligencia Secreto británico (o MI6) comenzó a discutir la máquina Enigma con el Deuxième Bureau, la oficina de cifrado francesa, de la que adquirieron los datos que los franceses habían obtenido del espía alemán Asché. En enero de 1939 tuvo lugar el primer encuentro polaco-francés-británico en París (Francia),

donde el MI6 estaba representado por Dilly Knox, Hugh Foss y Alastair Denniston. Knox describió un sistema de colocación de varillas que él mismo había desarrollado, pero los polacos fueron instruidos por sus superiores para que no revelasen ninguna información vital en ese momento.

A pesar de eso, Knox había conseguido impresionar a los polacos. En julio de 1939, a las puertas de la guerra, se organizó una segunda reunión. Esta tuvo lugar en Polonia, en una instalación de la Oficina Polaca de Cifrado en un bosque cerca de Pyry, al sur de Varsovia (Polonia). En esta reunión, los polacos revelaron sus logros y dieron a los franceses y a los británicos una copia de la máquina que estaban utilizando los alemanes.

La diferencia a la hora de intentar descifrar la Enigma era que Knox había aplicado la lingüística, mientras que los polacos habían utilizado la teoría matemática de la permutación. Gracias a esta teoría, consiguieron deducir el secreto del cableado interno de la Enigma y, a partir de esta debilidad, crearon una máquina para descifrar los códigos alemanes. Esta recibió el nombre de *Bomba kryptologiczna* (bomba criptológica) [14].

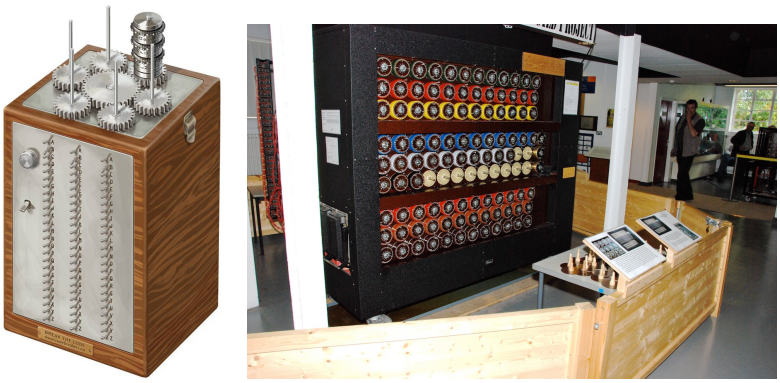

Figura 2.5: Bomba polaca (izquierda) y británica (derecha).

Inmediatamente después de la reunión de Pyry, la Oficina Polaca de Cifrado destruyó todos sus documentos y equipos secretos (lo que incluía a la Bomba), mientras que los criptoanalistas escaparon a Francia. Unas semanas más tarde, en agosto de 1939, los británicos fundaron Bletchley Park.

En septiembre de ese mismo año, Alemania invadió Polonia y dos días después, Francia y Reino Unido declararon la guerra a Alemania. La Segunda Guerra Mundial había empezado, semanas después de que los polacos compartieran sus secretos. En la primera etapa de la guerra, los polacos seguían trabajando en Enigma desde la Oficina Francesa de Cifrado.

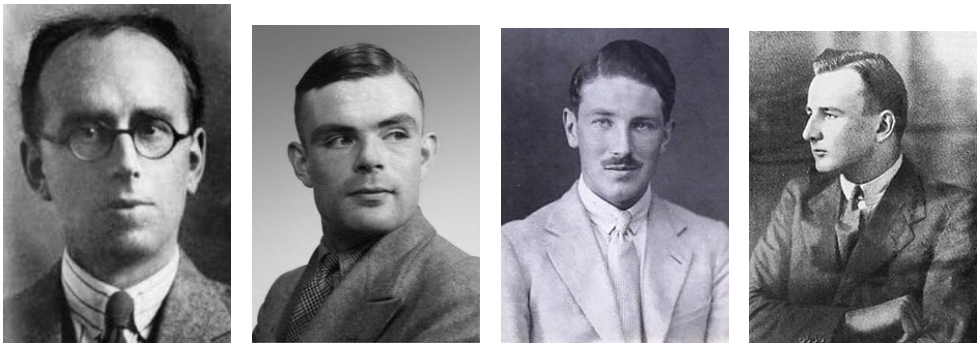

Figura 2.6: Miembros importantes de Bletchley Park. De izquierda a derecha: Knox, Turing, Welchman y Milner-Barry.

Bletchley Park era una finca en la pequeña ciudad de Bletchley (Milton Keynes, Reino Unido), a unos 70 kilómetros al norte de Londres, que se convertiría en la sede del Government Code and Cypher School, la Oficina Británica de Cifrado. El lugar fue elegido porque tenía conexiones ferroviarias directas con Londres, Cambridge y Oxford, lo que permitía que los científicos y el personal del ejército viajaran de manera discreta.

Las primeras personas en llegar a Bletchley Park (BP) fueron criptógrafos profesionales, matemáticos, ajedrecistas y personas con habilidades organizativas. Entre ellos se encontraban Dillwyn Knox, Gordon Welchman, Alan Turing y Stuart Milner-Barry. Milner-Barry era un jugador de ajedrez, mientras que Welchman y Turing eran matemáticos de Cambridge.

Al principio, los mensajes Enigma se rompían utilizando métodos sencillos de lápiz y papel. Pero a medida que aumentaba el volumen del tráfico de mensajes, Turing comenzó a buscar soluciones automatizadas.

Basándose en la Bomba polaca y en la información que fue pasada por los polacos poco antes del comienzo de la Segunda Guerra Mundial, Turing desarrolló la Bomba británica [14]. En comparación con la anterior, esta tenía un enfoque completamente diferente, ya que se basaba en la suposición de que un texto plano conocido (o adivinado) está presente en una determinada posición del mensaje. La primera máquina fue entregada en Bletchley Park en marzo de 1940.

En las primeras etapas de la guerra, los británicos podían leer la mayoría de los mensajes de radio de la Fuerza aérea alemana (Luftwaffe) y una parte modesta del tráfico del Ejército (Heer). Los mensajes navales, por otro lado, suponían un problema, ya que sus procedimientos operativos eran mucho más complicados. La Armada alemana (Kriegsmarine) utilizaba tres rotores adicionales (VI, VII y VII), cuyo cableado era desconocido hasta ese momento. Esos rotores adicionales se usaban exclusivamente en la Marina y no eran compartidos con otras partes del Ejército. Aparentemente, la Kriegsmarine utilizaba un procedimiento complejo que incluía varios libros de códigos, libros de mensajes cortos y tablas de sustitución.

En 1941, Turing descubrió como funcionaba el cableado de los rotores adicionales y el procedimiento de los indicadores para los mensajes navales. Ayudado por la obtención de una gran cantidad de códigos de un submarino alemán, capturado en mayo de ese mismo año, Turing consiguió averiguar el funcionamiento de la Enigma M3 de la Armada y descifrar gran parte del tráfico naval.

Ante esto, la Armada alemana introdujo un nuevo modelo de la Enigma en febrero de 1942, con un nuevo rotor situado entre el extremo izquierdo y el reflector. Además, cambiaron el sistema de indicadores e introdujeron nuevos libros de códigos. La nueva máquina se conocía como Enigma M4 (véase la figura 2.8) y era usada exclusivamente por la sección U-Boot de la Kriegsmarine [15]. Las Bombas, fabricadas para la Enigma de tres rotores, no funcionaban con este nuevo modelo.

La Enigma M4 tuvo un gran impacto en La Batalla del Atlántico. Como ya no era posible leer los mensajes que provenían o iban dirigidos a los submarinos alemanes, era imposible determinar su ubicación, lo que provocaba enormes pérdidas de barcos, personas y suministros.

Sin embargo, en octubre de 1942 se capturaron nuevos libros de códigos de un submarino alemán. Mientras tanto, Turing había elaborado los nuevos procedimientos navales y el cableado del rotor adicional.

Como la Bomba solo era adecuada para atacar a las máquinas Enigma de tres rotores, se desarrollaron varias soluciones. Una Bomba de tres rotores, que contenía el equivalente a treinta y seis máquinas Enigma, fue modificada para convertirse en una Bomba

Figura 2.7: Enigma M4 de cuatro rotores utilizada por la Marina alemana.

de cuatro rotores correspondiente a veinticuatro Enigmas. Aunque el resultado fue una máquina bastante lenta, funcionó.

Finalmente, se construyeron variantes de una verdadera Bomba de cuatro rotores. Algunas de ellas incluían una cuarta rueda extra rápida y un circuito electrónico de detección basado en válvulas. Para entonces, los Estados Unidos de América ya habían entrado en la guerra y, tras una larga discusión, se decidió compartir el conocimiento sobre la tecnología de la Bomba con los Aliados americanos.

Esta decisión permitió a los estadounidenses desarrollar su propia Bomba de cuatro rotores. Mientras que los británicos sufrían escasez de materiales, los estadounidenses tenían suficientes suministros y recursos para asignarles financiación y capacidad de producción. Así, el desarrollo de la primera Bomba estadounidense comenzó a finales de 1942 y a mediados de 1943 ya estaba lista.

A finales de 1943, se instalaron más de ciento veinte máquinas y, durante el resto de la guerra, los Estados Unidos se encargaron de romper la mayor parte de los mensajes Enigma basados en cuatro rotores (es decir, el tráfico submarino), dejando al Reino Unido las comunicaciones de tres rotores.

2.5 Consecuencias

Una vez acabada la guerra, toda la historia del descifrado de la máquina Enigma se mantuvo en secreto durante muchos años. Salvo algunas excepciones, las personas involucradas siguieron con sus vidas y la mayoría de las Bombas fueron desmanteladas. Las máquinas Enigma capturadas terminaron en las bóvedas de la Oficina Británica de Cifrado y la NSA (Agencia de Seguridad Nacional de los Estados Unidos), o fueron entregadas a otros países.

En algunos lugares como Noruega, Alemania y Austria, la Enigma-I se utilizó durante muchos años después de la guerra, hasta que fue reemplazada por equipos más nuevos y mejores. Se cree que la máquina también se utilizó en varios países africanos.

No hay informes sobre el uso de la Enigma por parte de la Unión Soviética, aunque es bastante seguro que capturasen algunas máquinas. Durante mucho tiempo se asumió que no tenían conocimiento alguno de los logros de los Aliados en la Segunda Guerra Mundial, pero ahora parece probable que estuvieran bien informados, ya que en 1956 los rusos introdujeron la primera versión de una máquina de cifrado basada en rotores muy avanzada que se llamaba FIALKA.

Esta máquina tenía diez ruedas de cifrado y presentaba pasos de rotor irregulares, con las ruedas moviéndose en ambas direcciones. El clavijero fue sustituido por un lector de tarjetas y permitía el uso de letras y números. Tenía un perforador de cinta y un lector incorporado, e imprimía la salida directamente en una tira de papel. Era usada por todos los países del Pacto de Varsovia.

Por su parte, los estadounidenses empezaron a desarrollar su máquina basada en rotores, llamada KL-7, que se convirtió en el mecanismo principal de cifrado de la OTAN en la era de la posguerra. A diferencia de Enigma, el KL-7 tenía ocho rotores, siete de los cuales se movían en un complejo patrón de pasos irregulares.

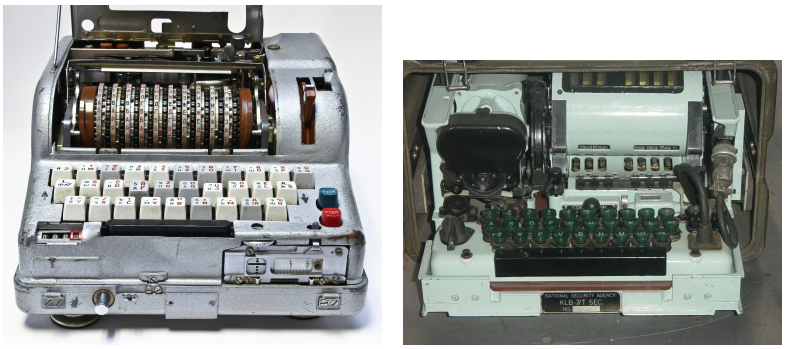

Figura 2.8: FIALKA (izquierda) y KL-7 (derecha), las máquinas de cifrado basadas en rotores creadas por la Unión Soviética y Estados Unidos, respectivamente.

CAPÍTULO 3
Funcionamiento de la máquina Enigma

En el presente capítulo vamos a explicar la mecánica y la estructura interna de la máquina Enigma-I, utilizada por las divisiones de tierra y aire de la Wehrmacht; y de las máquinas M3 y M4, que eran usadas por la Kriegsmarine. Además, describiremos el procedimiento que seguía el ejército alemán para el cifrado de mensajes. Por último, comentaremos la seguridad que presentaba la máquina desde un punto de vista criptográfico.

3.1 Especificaciones técnicas

La máquina Enigma es un dispositivo electromecánico que consiste en un teclado, un panel de luces (que representa el alfabeto), un panel de conexiones (clavijero), un reflector y tres o cuatro rotores, dependiendo del modelo.

La máquina tiene un compartimento para una batería de 4 voltios. Algunas versiones tienen un interruptor para seleccionar entre la batería interna o una alimentación externa, mientras que otras tienen un transformador para conectar la máquina a la red eléctrica.

Cuando se pulsa una tecla, se emite un pulso que viaja por un circuito eléctrico, pasando por los rotores y el reflector, hasta llegar a una de las veintiséis bombillas del panel de luces, que representa la letra encriptada. Hay que tener en cuenta que, al pulsar una tecla, primero se envía la corriente a los rotores y, posteriormente, realiza su recorrido a través del circuito. Esto quiere decir que el giro de los rotores tiene lugar antes que el cifrado de la letra pulsada.

La bombilla que se enciende, es decir, la letra encriptada, depende de las conexiones del clavijero, la disposición de los rotores y el reflector.

3.1.1. Panel de conexiones o clavijero

Este tablero permite reconfigurar el cableado interno de la máquina y aporta más resistencia criptográfica que un rotor adicional (véase el apartado 3.3), ya que una máquina Enigma sin clavijero puede ser descifrada utilizando métodos manuales.

Para conectar un par de letras se emplea un cable. Si, por ejemplo, conectamos la 'H' con la 'U', ambas letras se intercambian antes y después de la codificación por parte de los rotores. Aunque se pueden utilizar hasta trece cables a la vez, normalmente solo se usaban diez.

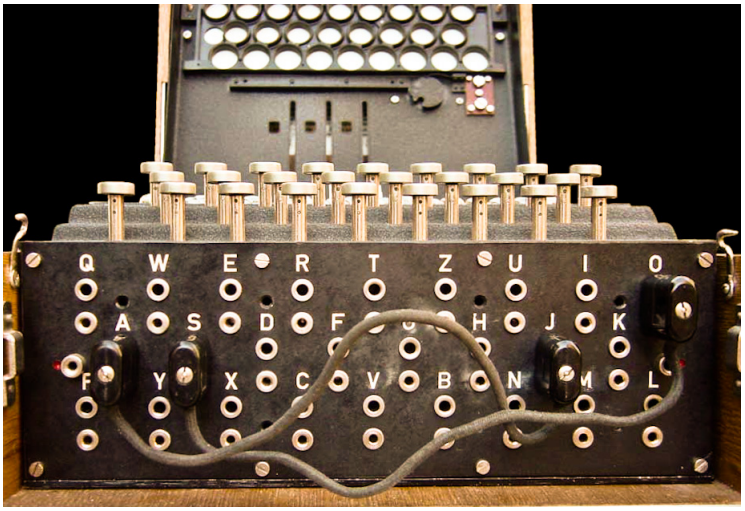

Figura 3.1: Tablero de conexiones.

3.1.2. Reflector

El reflector (*Umkehrwalze* en alemán, abreviado UKW) es un componente que envía los impulsos eléctricos, que llegan desde los rotores, de vuelta en orden inverso. El reflector tiene 26 pines en el lado derecho. Cada uno de ellos está conectado a otro del mismo conjunto, formando 13 parejas.

En la tabla 3.1, cada columna representa los emparejamientos que hay en cada uno de los cuatro reflectores. De esta forma, en el reflector B, el contacto de la letra 'A' se encuentra cableado con el de la letra 'Y' y viceversa, formando un bucle.

Físicamente, cada pin se encuentra en el lado derecho del reflector, obteniendo así una encriptación recíproca. La ventaja de este diseño es que el cifrado y el descifrado se realizan con el mismo proceso electromecánico y los mismos ajustes.

Sin embargo, una letra nunca se puede encriptar por ella misma. Esta es una propiedad que abrió la puerta al criptoanálisis, haciendo el trabajo más fácil para los descifradores.

B	C	B fino	C fino
Y - A	F - A	E - A	R - A
R - B	V - B	N - B	D - B
U - C	P - C	K - C	O - C
H - D	J - D	Q - D	J - E
Q - E	I - E	U - F	N - F
S - F	O - G	Y - G	T - G
L - G	Y - H	W - H	K - H
P - I	R - K	J - I	V - I
X - J	Z - L	O - L	M - L
N - K	X - M	P - M	W - P
O - M	W - N	X - R	Z - Q
Z - T	T - Q	Z - S	X - S
W - V	U - S	V - T	Y - U

Tabla 3.1: Los cuatro reflectores

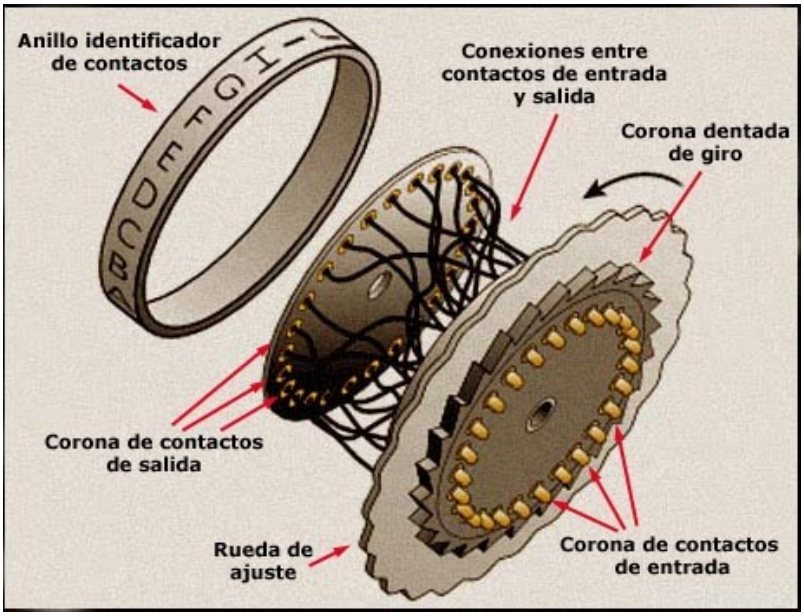

Figura 3.2: Elementos del rotor.

3.1.3. Rotores

Los rotores, o ruedas de cifrado, son los elementos más importantes de la máquina. Estos discos redondos, de aproximadamente 10 centímetros de diámetro, están hechos de metal o baquelita.

Un disco consiste en una carcasa redonda giratoria con las letras de la A a la Z o los números del uno al veintiséis, y una muesca. El centro del rotor se encuentra fijado en la rueda selectora que contiene todos los rotores de la máquina, y contiene veintiséis contactos con resorte en el lado derecho, conectados a veintiséis contactos planos en el lado izquierdo.

El cableado representa una encriptación de sustitución y es diferente para cada rotor. Además, es posible cambiar estas conexiones para que una letra (o número) de un extremo del rotor esté en contacto con otra distinta a su pareja original (véase el apartado 3.1.4).

La combinación de varios rotores, en posiciones siempre cambiantes entre sí, es lo que hace que el cifrado sea tan complejo. Cada rotor tiene a su izquierda una muesca y a su derecha 26 dientes, que son utilizados por el mecanismo de pasos para el avance de los rotores (véase el apartado 3.1.5).

En un principio, la máquina fue introducida con tres rotores. En 1939, el conjunto se amplió a cinco rotores, marcados con los números romanos I, II, III, IV y V, todos con una sola muesca. La Kriegmarine amplió este grupo de rotores añadiendo otros tres, VI, VII y VIII, cada uno con dos muescas.

En 1942, la Enigma M4 introdujo un cuarto rotor. Para ello, los reflectores anchos B y C que se usaban en la versión de tres rotores fueron sustituidos por reflectores finos (con los mismos nombres), dejando espacio para el cuarto rotor especial. Habían dos rotores que podían ocupar ese espacio, los llamados Beta y Gamma, y tenían contactos de resorte en ambos lados. Esto los hacía incompatibles con los otros ocho rotores.

El cableado interno de los rotores realiza la encriptación real. Como puede apreciarse en la tabla 3.2, las letras de cada una de las columnas representan los veintiséis contactos del lado izquierdo y los veintiséis pines del lado derecho. La señal viaja primero de de-

I	II	III	IV	V	VI	VII	VIII	Beta	Gamma
E - A	A - A	B - A	E - A	V - A	J - A	N - A	F - A	L - A	F - A
K - B	J - B	D - B	S - B	Z - B	P - B	Z - B	K - B	E - B	S - B
M - C	D - C	F - C	O - C	B - C	G - C	J - C	Q - C	Y - C	O - C
F - D	K - D	H - D	V - D	R - D	V - D	H - D	H - D	J - D	K - D
L - E	S - E	J - E	P - E	G - E	O - E	G - E	T - E	V - E	A - E
G - F	I - F	L - F	Z - F	I - F	U - F	R - F	L - F	C - F	N - F
D - G	R - G	C - G	J - G	T - G	M - G	C - G	X - G	N - G	U - G
Q - H	U - H	P - H	A - H	Y - H	F - H	X - H	O - H	I - H	E - H
V - I	X - I	R - I	Y - I	U - I	Y - I	M - I	C - I	X - I	R - I
Z - J	B - J	T - J	Q - J	P - J	Q - J	Y - J	B - J	W - J	H - J
N - K	L - K	X - K	U - K	S - K	B - K	S - K	J - K	P - K	M - K
T - L	H - L	V - L	I - L	D - L	E - L	W - L	S - L	B - L	B - L
O - M	W - M	Z - M	R - M	N - M	N - M	B - M	P - M	Q - M	T - M
W - N	T - N	N - N	H - N	H - N	H - N	O - N	D - N	M - N	I - N
Y - O	M - O	Y - O	X - O	L - O	Z - O	U - O	Z - O	D - O	Y - O
H - P	C - P	E - P	L - P	X - P	R - P	F - P	R - P	R - P	C - P
X - Q	Q - Q	I - Q	N - Q	A - Q	D - Q	A - Q	A - Q	T - Q	W - Q
U - R	G - R	W - R	F - R	W - R	K - R	I - R	M - R	A - R	L - R
S - S	Z - S	G - S	T - S	M - S	A - S	V - S	E - S	K - S	Q - S
P - T	N - T	A - T	G - T	J - T	S - T	L - T	W - T	Z - T	P - T
A - U	P - U	K - U	K - U	Q - U	X - U	P - U	N - U	G - U	Z - U
I - V	Y - V	M - V	D - V	O - V	L - V	E - V	I - V	F - V	X - V
B - W	F - W	U - W	C - W	F - W	I - W	K - W	U - W	U - W	V - W
R - X	V - X	S - X	M - X	E - X	C - X	Q - X	Y - X	H - X	G - X
C - Y	O - Y	Q - Y	W - Y	C - Y	T - Y	D - Y	G - Y	O - Y	J - Y
J - Z	E - Z	O - Z	B - Z	K - Z	W - Z	T - Z	V - Z	S - Z	D - Z

Tabla 3.2: Cableado de rotores en la Enigma-I, M3 y M4

recha a izquierda a través de los rotores hacia el reflector y luego regresa de izquierda a derecha. Si nos fijamos en el rotor I vemos que, de derecha a izquierda, la 'A' se cifra en una 'E', la 'B' en una 'K', y la 'C' en una 'M'.

Es necesario comentar que las relaciones entre las letras de cada columna son las predeterminadas de cada rotor, pero pueden variar dependiendo de los ajustes internos de este. Es decir, la letra 'A' del rotor I puede hacer contacto con cada uno de los veinticinco pines restantes, no tiene por qué estar conectada a la letra 'E'.

3.1.4. Ajustes del rotor

Salida	I	Entrada
A	A - E	A
B	B - K	B
C	C - M	C
D	D - F	D
E	E - L	E
F	F - G	F
G	G - D	G
H	H - Q	H
I	I - V	I
J	J - Z	J
K	K - N	K
L	L - T	L
M	M - O	M
N	N - W	N
O	O - Y	O
P	P - H	P
Q	Q - X	Q
R	R - U	R
S	S - S	S
T	T - P	T
U	U - A	U
V	V - I	V
W	W - B	W
X	X - R	X
Y	Y - C	Y
Z	Z - J	Z

Tabla 3.3: Rotor I en forma inicial. Si al rotor le llega una 'A' (columna derecha), viajará por el cableado interno hasta la letra 'E' (parte derecha de la columna central), esto hace que el rotor I devuelva la letra 'E', que se encuentra en la quinta posición de la parte izquierda de la columna central. Por tanto, la letra de salida es la quinta del alfabeto, es decir, la 'E' (columna izquierda).

Como hemos comentado anteriormente, en el exterior del núcleo del cableado hay un anillo móvil con 26 letras (o números) y una muesca. Este anillo es giratorio y se puede colocar en cualquiera de las 26 posibles posiciones. La posición del anillo se denomina *Ringstellung*, y al modificarla se cambia la posición del alfabeto en relación con el cableado interno.

Supongamos que estamos trabajando con el rotor I. Su configuración interna predeterminada se muestra en la tabla 3.3. Después de esto, el rotor realiza un paso, por lo que su nueva disposición es la que se muestra en la tabla 3.4.

Salida	I	Entrada
A	B - K	A
B	C - M	B
C	D - F	C
D	E - L	D
E	F - G	E
F	G - D	F
G	H - Q	G
H	I - V	H
I	J - Z	I
J	K - N	J
K	L - T	K
L	M - O	L
M	N - W	M
N	O - Y	N
O	P - H	O
P	Q - X	P
Q	R - U	Q
R	S - S	R
S	T - P	S
T	U - A	T
U	V - I	U
V	W - B	V
W	X - R	W
X	Y - C	X
Y	Z - J	Y
Z	A - E	Z

Tabla 3.4: Rotor I después de cifrar una letra por primera vez. Si le llega una 'A', mandará la señal a la letra 'K', que se encuentra en la décima posición de la parte izquierda de la columna central. Entonces, la letra de salida será la décima del alfabeto, la 'J'.

En estos dos últimos cifrados hemos tenido en cuenta que el rotor I se encuentra en la configuración predeterminada (con el *Ringstellung* puesto a A-01). Sin embargo, cambiar este ajuste haría que el encriptado fuese distinto.

Salida	I	Entrada
A	Z - J	A
B	A - E	B
C	B - K	C
D	C - M	D
E	D - F	E
F	E - L	F
G	F - G	G
H	G - D	H
I	H - Q	I
J	I - V	J
K	J - Z	K
L	K - N	L
M	L - T	M
N	M - O	N
O	N - W	O
P	O - Y	P
Q	P - H	Q
R	Q - X	R
S	R - U	S
T	S - S	T
U	T - P	U
V	U - A	V
W	V - I	W
X	W - B	X
Y	X - R	Y
Z	Y - C	Z

Tabla 3.5: Rotor I con el ajuste B-02. Así, al enviar una 'A', el rotor la cambia por una 'J', que se encuentra en la decimoprimera posición de la parte izquierda, por lo que se encripta en la decimoprimera letra del alfabeto, la 'K'.

Como la primera letra de este rotor es la 'E', con el *Ringstellung* puesto a B-02 esta se encontraría ahora cableada a la 'B'. El resto de letras se cablearían en orden con las 25 restantes, obteniendo la configuración de la tabla 3.5.

En todos los ejemplos hemos tenido en cuenta un solo rotor, pero lo normal es tener tres rotores (cuatro en la máquina M4), conectados uno tras otro. De esta manera, la letra de salida de un rotor se convierte en la letra de entrada del rotor de su izquierda.

La combinación del cableado, la posición del rotor y su desplazamiento consiguen un cifrado complejo. Por eso cada paso de un rotor produce una trayectoria completamente diferente a través de los tres (o cuatro) rotores.

3.1.5. El mecanismo de paso a paso de los rotores

Cuando hablamos de las posiciones de los rotores, se utiliza la siguiente notación: II, IV, VI significa que el rotor II se encuentra a la izquierda, el IV en el medio y el VI en la derecha.

Un rotor hace girar al de su izquierda dependiendo de su punto de rotación. Si el punto de rotación del rotor I es 'Q', esto quiere decir que cuando el rotor I cambie de la letra 'Q' a la letra 'R', el rotor de su izquierda, sea cual sea, girará en un paso.

En la tabla 3.6, se puede observar los puntos de rotación de cada rotor. Cabe destacar los casos del rotor VI, VII y VIII, en el que los tres comparten dos puntos; y el de Beta y Gamma, que no tienen ninguno debido a que a su izquierda solo puede estar el reflector.

Cuando el rotor central se encuentre en su punto de rotación al mismo tiempo que el rotor de la derecha, se producirá un doble paso. Por ejemplo, si se están utilizando los rotores I, II y III con una secuencia 'AEV', entonces al pulsar una tecla la secuencia pasará a ser 'BFW', ya que tanto el rotor II como el III se encuentran en sus respectivos puntos de rotación. De igual forma, el segundo rotor se moverá cada vez que haga girar al primero, ya que si no se quedará en su punto de rotación. Partiendo del mismo ejemplo, la secuencia 'AEB' precederá a 'BFC', en lugar de 'BEC', porque si pasase esto en la siguiente secuencia volvería a girar el primer rotor (obteniendo 'CED') y así sucesivamente.

En el modelo M4, este mecanismo no funciona con el cuarto rotor, por lo que este nunca se mueve y solo puede ser ajustado manualmente.

Rotor	Punto de rotación
I	Cuando el rotor pasa de 'Q' a 'R'
II	Cuando el rotor pasa de 'E' a 'F'
III	Cuando el rotor pasa de 'V' a 'W
IV	Cuando el rotor pasa de 'J' a 'K'
V	Cuando el rotor pasa de 'Z' a 'A'
VI, VII y VIII	Cuando el rotor pasa de 'Z' a 'A' o de 'M' a 'N'

Tabla 3.6: Puntos de rotación

3.2 Procedimiento de envío y recepción de mensajes

3.2.1. Procedimiento de la Wehrmacht

Los operadores de la Wehrmacht configuraban la máquina de acuerdo con un diario de claves. Este contenía el día (*Tag*), la selección y el orden del rotor (*Walzenlage*), el ajuste de los anillos (*Ringstellung*), las conexiones del clavijero (*Steckerverbindungen*) y los grupos de identificación (*Kenngruppen*). Los días se escribían en orden inverso para que el operario pudiese cortar uno cuando los ajustes de este caducasen.

Tag	Walzenlage	Ringstellung	Steckerverbindungen	Kenngruppen
31	I-II-V	06-22-14	PO-ML-IU-KJ-NH-YT-GB-VF-RE-DC	EXS-TGY-IKJ-LOP
30	III-IV-II	017-04-26	BN-VC-XS-WQ-AZ-GT-YH-JU-IK-PM	KIJ-TFR-BVC-ZAE
29	V-I-III	15-02-09	ML-KJ-HG-FD-SQ-TR-EZ-IU-BV-XC	QZE-TRF-IOU-TGB

Tabla 3.7: Ejemplo de diario de claves para tres días

Para identificar la clave, utilizada en un mensaje en particular, el operador tenía que insertar un grupo de identificación de cinco letras. Este grupo, llamado *Buchstabenkenngruppe*, se componía de dos letras escogidas al azar y uno de los cuatro posibles *Kenngruppen* de tres letras del diario de claves. Las dos letras podían unirse al principio del grupo o al final. En el caso de la tabla 3.7, algunos ejemplos de un correcto *Buchstaben-*

kenngruppe para el día 31 son 'TVEXS', 'TGYZA' o 'LOPXY'. Estos grupos solían estar al principio del mensaje y tenían que omitirse durante el cifrado y el descifrado.

Así, cuando el destinatario miraba este primer grupo, reconocía qué clave se usó para ese mensaje en particular. Si el mensaje se dividía en varias partes, el operador debía insertar un nuevo grupo en el *Buchstabenkenngruppe* para cada una de las partes.

Para encriptar un mensaje, el operador tenía que seleccionar una posición aleatoria para los rotores (la llamada clave del mensaje, *Spruchschlussel* en alemán), que tenía que ser única para cada mensaje. Esto evitaba el uso excesivo de las mismas configuraciones secretas para un día determinado.

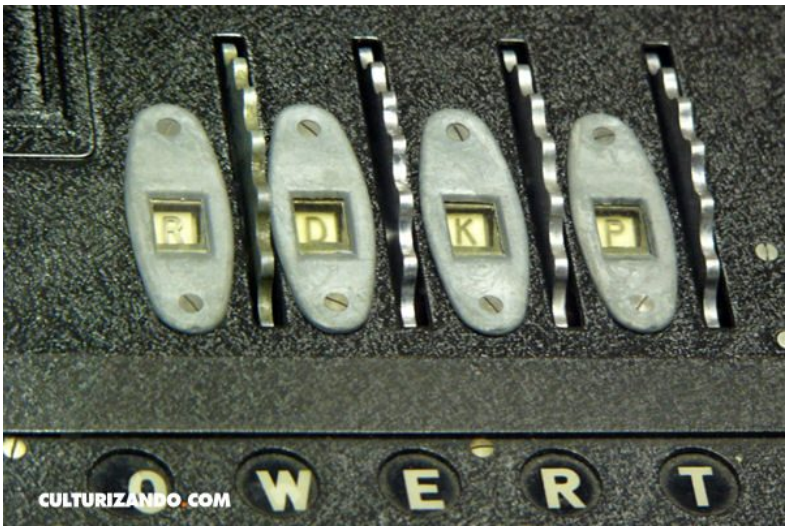

Figura 3.3: La clave del mensaje se configuraba moviendo los rotores una vez que la máquina estaba cerrada y lista. En la imagen, la clave es 'BDKP'.

Como el mensaje debía mantenerse en secreto, utilizaron el siguiente procedimiento: El operador encargado de cifrar seleccionaba una posición básica (*Grundstellung*) y una clave de mensaje aleatorias. Por ejemplo, 'EHZ' y 'XWB'. La primera se configuraba en el *Spruchschlussel* y la segunda se cifraba, obteniendo 'TBS'. Luego, se encripta el mensaje con 'XWB' como clave de este. Finalmente, se envía la posición inicial 'EHZ' y la clave del mensaje 'TBS' junto con el mensaje cifrado.

De esta manera, a las 15:00 del día 31, se transmitía un mensaje de 49 letras:

U6Z DE C 1500 = 49 = EHZ TBS =
TVEXS QBLTW LDAHH YEOEF
PTWYB LENDP MKOXL DFAMU
DWIJD XRJZ=

Para descifrarlo, el operario procedía en orden inverso al encriptado: Para empezar, seleccionaba la Wehrmacht Enigma I con el reflector B. Después, mirando el diario de claves, seleccionaba los rotores, ajustándolos al *Ringstellung* especificado, y conectaba los claves pertinentes en el clavijero. Luego, configuraba la clave del mensaje a 'EHZ' y descifraba 'TBS', obteniendo como resultado 'XWB'. Finalmente, se daba el valor 'XWB' a la clave del mensaje y se descifraba el resto del texto, ignorando el grupo de identificación 'TVEXS'.

3.2.2. Procedimiento de la Kriegsmarine

El método de envío y recepción de mensajes empleado por la armada era mucho más complejo. Su diario de claves estaba compuesto por dos partes:

- *Innere Einstellung* (ajustes internos). Contenía los tres rotores y sus respectivos ajustes de anillo, el rotor Beta o Gamma y el reflector. Esto sólo era para los días impares del mes.

- *Aussere Einstellung* (ajustes externos). Incluía los cables y la posición básica de inicio (*Grundstellung*) para cada día del mes.

Existía una clave adicional para los oficiales y otra especial para la comunicación privada entre el Capitán y el Comandante de los submarinos.

El sistema de grupos de identificación (*Kenngruppen*) era completamente diferente al del ejército de tierra. Además de usarlos en los diarios de claves, la Kriegsmarine utilizó estos grupos en sus redes de cifrado principales para determinar la clave del mensaje, denominados *Kenngruppenbuch*. El *Kenngruppenbuch* contenía las siguientes partes:

- *Zuteilungsliste* (lista de asignaciones), que indicaba al operador qué tabla debía utilizar para una determinada red de cifrado.

- *Tauschtafelplan* (tablero de punteros), que mostraban al operador, dada una tabla determinada, la columna usada para escoger los trigramas adecuados.

- Las tablas con el *Kenngruppen*.

El operador debía seleccionar dos trigramas del *Kenngruppenbuch*. Uno sería el *Schlüsselkenngruppe* (grupo indicador de la clave), para saber qué clave usar; y otro el *Verfahrenkenngruppe* (grupo indicador de la encriptación) para obtener la clave del mensaje.

Con la Enigma en el *Grundstellung* (posición básica para ese día) el operador escribía la clave del mensaje en el *Verfahrenkenngruppe*, que se usaría como posición inicial para cifrar el mensaje. Los dos trigramas juntos (*Schlüsselkenngruppe* y *Verfahrenkenngruppe*) eran el indicador del mensaje.

Este indicador se sometió a un cifrado de sustitución adicional con una tabla de bigramas llamada *Doppelbuchstabentauschtafel* (tabla de conversión de dos letras). Esta tabla era recíproca, de modo que si, por ejemplo, el bigrama 'BK' era codificado por 'MC', el bigrama 'MC' se decodificaba por 'BK'.

El operador escribía los dos trigramas del indicador de mensajes uno debajo del otro, pero agregando, aleatoriamente, una letra al principio del primer trigrama y otra al final del segundo. Los bigramas se tomaban verticalmente del indicador de mensajes y se codificaban de acuerdo con la tabla de bigramas. Los dos grupos de cuatro letras resultantes (el indicador de mensajes codificado) se añadían al principio y al final del mensaje.

Los mensajes de la Kriegsmarine Enigma tenían un formato de grupos de cuatro letras. Algunos mensajes se codificaban con los libros de códigos llamados *Kurzsignalheft* y *Wetterkurzschlussel* antes de cifrarse con la Enigma. Esto se hacía porque el primero convertía palabras, números y todo tipo de expresiones y frases técnicas y operativas en códigos de cuatro letras. Por otro lado, el segundo convertía informes meteorológicos completos en códigos de veintitrés o veinticuatro letras.

3.3 Resistencia criptográfica

Para calcular la seguridad matemática de la Enigma tenemos que encontrar todos los ajustes posibles de la máquina. Por tanto, hay que tener en cuenta la selección y el orden

de los rotores, su cableado, la configuración del *Ringstellung* de cada uno de ellos, su posición inicial al principio del mensaje, el reflector y las configuraciones del clavijero.

Los criptógrafos alemanes sabían que un solo rotor se podía cablear de muchas formas distintas. En total, el número posible de estas era 4×10^{26}.

Desafortunadamente para ellos, los descifradores aliados conocían la máquina, los rotores y su cableado interno. Por tanto, solo tenían que tener en cuenta las diferentes formas en que se podía configurar la máquina. Esto se denomina seguridad práctica, que es mucho menor que la teórica en el caso de la Enigma.

Para los aliados, solo había veintiséis variaciones diferentes para un mismo rotor, es decir, las veintiséis posiciones que este podía tener en la máquina. De este modo, no tuvieron que buscar entre el inmenso número de cableados posibles.

Los criptógrafos alemanes cometieron el error de ignorar que la seguridad de un dispositivo nunca puede depender del secreto del sistema (como el cableado de los rotores), porque tarde o temprano este se verá comprometido. Únicamente puede depender del secreto de la clave (la selección de los rotores y el clavijero).

Supongamos que estamos trabajando con el modelo Enigma-I de la Wehrmacht, con el reflector B ancho y con la posibilidad de escoger entre cinco rotores. Utilizamos diez cables de enchufe en el clavijero, que era el número de cables por defecto.

Para seleccionar tres rotores entre un total de cinco posibles, existen 60 combinaciones $(5 \times 4 \times 3)$.

El cableado interno de cada rotor se puede ajustar en cualquiera de las veintiséis posiciones. Por lo tanto, con tres rotores hay 17.576 posiciones de rotor diferentes $(26 \times 26 \times 26)$.

El anillo de cada rotor funciona de manera similar al cableado, pero difiere en que solo se tienen en cuenta los anillos del rotor central y del derecho, porque no hay ningún rotor a la izquierda del tercero. Por tanto, hay 676 combinaciones de anillos (26×26).

Por último, falta saber las posibles combinaciones del clavijero. La ecuación para calcular el número de posibilidades de conectar pares de letras de un total de 26 con un número dado de cables es:

$$\frac{N!}{(N-2)! \times n! \times 2^n}$$

Donde N es el número total de letras, en este caso N=26; y n es el número de cables, es decir, n=10. Por tanto, tenemos 150.738.274.937.250 combinaciones posibles.

Teniendo en cuenta todo lo anterior, se procede al cálculo:

$$60 \times 17{,}576 \times 676 \times 150{,}738{,}274{,}937{,}250 \approx 1{,}07 \times 10^{23}$$

Así, las maneras distintas en las que se podía configurar la máquina de la Wehrmacht era del orden de 10^{23}.

Esto es comparable a una clave de 77 bits.

Es necesario aclarar algo sobre este resultado. En realidad, el período máximo de los rotores, esto es, el número de pasos antes de que la máquina se repita, es ligeramente inferior a 17.576, debido al doble paso del mecanismo de los rotores. Aun así, es un número prácticamente insignificante en comparación con las combinaciones del clavijero.

La máquina de la Wehrmacht podía equiparse con el reflector B o C. Por lo general, siempre se utilizaba el mismo, ya que el uso de diferentes reflectores creaba problemas

logísticos, de procedimiento y prácticos. Además, el hecho de elegir entre dos reflectores solo duplicaría el espacio de la clave.

La adición de un cuarto rotor en la Enigma M4 de la Kriegsmarine para mejorar su seguridad fue una oportunidad perdida. Como este nuevo rotor era inmóvil, solo complicó la máquina en un factor de 26 y podía considerarse como un nuevo reflector regulable.

Sin embargo, la introducción de ocho rotores en las versiones M3 y M4 fue un enfoque mucho mejor, ya que aumentaron las combinaciones de rotores de 60 a 336.

Si tomamos el modelo M4 para el cálculo del tamaño de la clave, esta máquina usa tres rotores de un total de ocho, lo que proporciona 336 combinaciones de rotor $(8 \times 7 \times 6)$.

A esto se le añade un cuarto rotor especial sin anillo, que puede ser Beta o Gamma. Estos no son compatibles con los otros rotores y sólo se pueden colocar en la cuarta posición, por lo que tenemos 2 opciones.

Los cuatro rotores se pueden ajustar en cualquiera de las 456.976 posiciones $(26 \times 26 \times 26 \times 26 \times 26)$.

Aunque la M4 daba a elegir entre un reflector B o C más pequeños que los originales, generalmente nunca se cambiaban, de modo que no vamos a incluirlos en nuestros cálculos.

Una vez más, sólo había dos anillos involucrados, ya que el tercer rotor no pisaba el cuarto, y este nunca se movía. La M4 también se suministró con 10 cables de conexión.

Sabiendo todo esto, calculamos:

$$336 \times 2 \times 456{,}976 \times 676 \times 150{,}738{,}274{,}937{,}250 \approx 3{,}1 \times 10^{25}$$

Por lo que las combinaciones posibles de la M4 eran del orden de 10^{25}.

Esto es comparable a una clave de 84 bits, por lo que, en comparación, la Kriegsmarine M4 era 291 veces más fuerte que la Wehrmacht.

Sabiendo todo esto, se puede afirmar que el descifrado de la máquina Enigma fue un gran desafío. El espacio de clave de 77 bits que poseía era extraordinario en la época de los años 40, ya que una búsqueda exhaustiva era imposible por aquél entonces; y es un tamaño considerable hasta para los estándares informáticos de hoy en día.

CAPÍTULO 4
El entorno y lenguaje de programación Scratch

En esta parte vamos a explicar en qué consiste el entorno de programación Scratch, centrándonos en su funcionamiento, en el propósito que persigue y en las utilidades que nos proporciona para realizar nuestro simulador de la máquina Enigma.

4.1 Proyecto Scratch

Figura 4.1: Logo de Scratch.

Scratch es un lenguaje de programación educativo, gratuito y de código abierto desarrollado por el Massachusetts Institute of Technology (MIT) con más de cuarenta millones de usuarios registrados y cuarenta y dos millones de proyectos compartidos. También es una plataforma multimedia de programación que cuenta con una versión de escritorio y una versión web. Está dirigida principalmente a estudiantes y profesores, ya que con ella se aprende a realizar juegos, historias, animaciones y otras aplicaciones o programas de forma fácil para iniciarse en el mundo de la programación [18]. Todo su contenido se encuentra disponible en su página web.[1]

Este proyecto tiene su origen en el año 2003 por iniciativa del grupo de investigación Lifelong Kindergaten del MIT, con el físico y programador Mitchel Resnick a la cabeza. Este grupo creía que, debido al constante cambio del mundo, la mejor idea para aprender es enseñando a diseñar, a expresarse y a emplear toda la creatividad posible. Resnick tomó como inspiración el lenguaje Logo2, creado por Seymour Papert, cuyo concepto de

[1] https://www.scratch.mit.edu/

Figura 4.2: Mitchel Resnick, creador de Scratch.

programación en bloques se basaba en las figuras de LEGO. De esta manera, los niños que quisieran aprender a programar lo verían como si se tratase de un juego.

El nombre que Resnick y su equipo le pusieron a este entorno proviene de una técnica usada por disc-jockeys en la música hip hop. Esta consistía en mover los vinilos hacia delante y hacia atrás con las manos, de forma que se reproducen nuevos sonidos, que forman una melodía particular. Scratch busca hacer algo similar, pero añadiendo en la mezcla imágenes o gráficos. Así se pueden conseguir animaciones o programas creativos.

El proyecto de Scratch es de desarrollo cerrado, ya que la elaboración del lenguaje de programación corre a cargo del grupo de investigación. En un futuro, se espera que el código fuente sea liberado para que la comunidad pueda modificar el programa y añadir extensiones.

El desarrollo de la primera versión de Scratch duró desde 2003 hasta 2007 e incluía varias interfaces y experimentos con diferentes bloques y características (véase la figura 4.3). En 2006 fue lanzada solo como servicio en línea, mientras que en marzo de 2007 salió como versión de escritorio. A partir de ese momento, cualquier usuario de Scratch podía visualizar y modificar otros proyectos de la comunidad, además de realizar los suyos propios. En 2013 surgió la versión 2.0 con la novedad de crear y editar los proyectos online, sin necesidad de descargarlo ni instalarlo. En 2018 salió Scratch 3.0 [19], y supuso el cambio de programación en Flash a HTML5, por lo que a partir de entonces también se pueden crear proyectos desde tabletas y visualizarlos en teléfonos móviles.

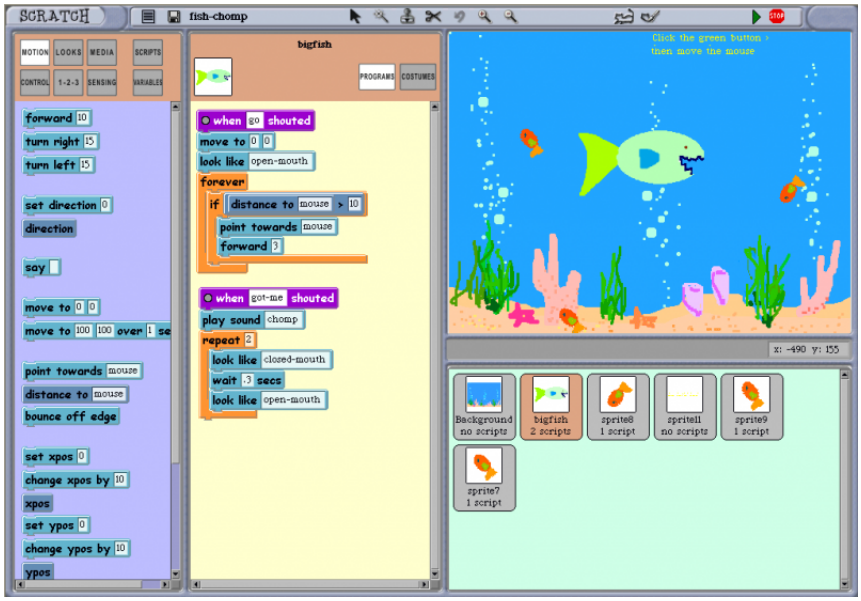

Figura 4.3: Una primera versión de Scratch de 2004.

4.2 Objetivo de Scratch

Scratch se desarrolló con el propósito de que cualquier persona interesada en el ámbito de la programación pudiese aprender de manera entretenida, creativa e interactiva. Principalmente, la plataforma de Scratch ha de resultar útil para la gente inexperta en este tema.

Scratch intenta inspirar en los usuarios un mayor entendimiento de las nuevas tecnologías, especialmente en los algoritmos de programación. Las principales características que tiene Scratch son:

- **Sociabilidad**. En Scratch es muy importante crear una comunidad para participar en el trabajo de los usuarios. Mediante la opción 'Compartir', presente tanto en la versión web como en la versión de escritorio, nuestro proyecto queda subido a la página web para que cualquier otro usuario pueda ejecutarlo desde el navegador o desde local.

  Una vez que un proyecto está subido, cualquier usuario puede modificarlo creando una copia del original. Así se consigue un trabajo de colaboración entre usuarios de diferentes lugares, por lo que ha sido necesario traducir Scratch a un gran número de idiomas para extenderlo de manera internacional.

- **Facilidad**. Programar en Scratch es agrupar bloques de código según la funcionalidad que le queramos dar a nuestro programa o aplicación. No podemos conectar dos o más bloques si la sintaxis que se forma al juntarlos es errónea. Esto ayuda a la hora de prevenir fallos. Un bloque tiene un color que determina su funcionalidad, por lo que su utilización y el determinar con qué otros bloques se ha de juntar se hace sencillo. Otras opciones que facilitan su uso es el poder mover una pila de bloques o ejecutarlos de forma separada al resto del programa.

- **Aprendizaje**. Una forma efectiva de entender algo es involucrarse en aquello que se está realizando. Scratch intenta que sus usuarios se impliquen lo máximo posible en sus proyectos mediante la importación de imágenes, audios o vídeos que ellos consideren oportunos, así como poder dibujar lo que queramos. Esto le da un toque

personal. Por otro lado, se procura que las herramientas tengan un amplio abanico de usos para que en Scratch se puedan programar tanto juegos sencillos como simuladores complejos. Esto ofrece diversidad en el desarrollo de programas.

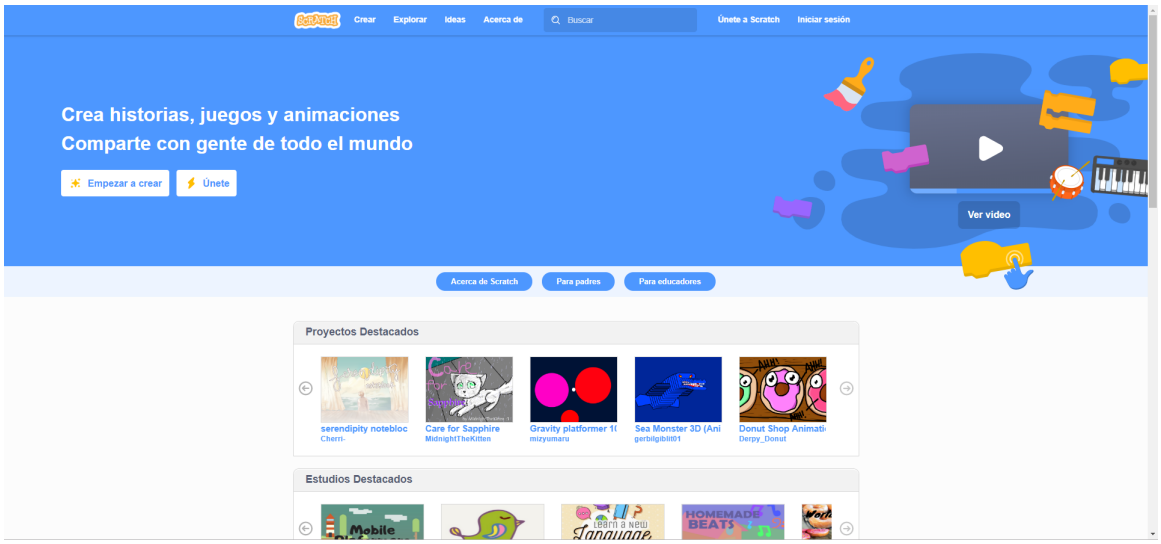

Figura 4.4: Página principal de la web de Scratch.

4.3 El pensamiento computacional

Según lo definió la teórica e ingeniera informática Jeannette Wing [20], el pensamiento computacional es el conjunto ordenado de pensamientos dedicados a plantear problemas y hallar sus soluciones, así como que puedan estar representadas por una persona o un ordenador que procese toda esa información.

Scratch construye su pensamiento computacional basándose en las nociones de procesos iterativos, instrucciones secuenciales o paralelas, eventos, variables, operadores, condicionales y datos. Para poner en práctica todas estas ideas, se emplean las siguientes técnicas:

- Reutilizar código para no empezar desde cero.
- Intentar ser incremental en la búsqueda de soluciones
- Probar una primera versión del código y, si se cometen errores, corregirlos posteriormente.
- Crear una estructura coherente y ordenada para resolver la complejidad.

Estos hábitos también se emplean en lenguajes de programación de alto nivel. Si conseguimos dominarlos en Scratch, podremos adaptarlos a otros lenguajes más complejos.

En resumen, podemos afirmar que el pensamiento computacional nos ayuda a resolver problemas complejos de manera algorítmica, ayudando a aumentar la eficacia de los procesos.

4.4 El entorno de programación

En esta sección explicaremos la estructura que presenta Scratch a la hora de programar y las distintas herramientas que nos proporciona. Dado que este trabajo se ha realizado con la versión de escritorio, nos centraremos en ella, aunque realmente hay pocas diferencias entre esta y la versión de la página web.

Para empezar, haciendo uso de la figura, vamos a comentar los elementos que presenta la interfaz. Estos son:

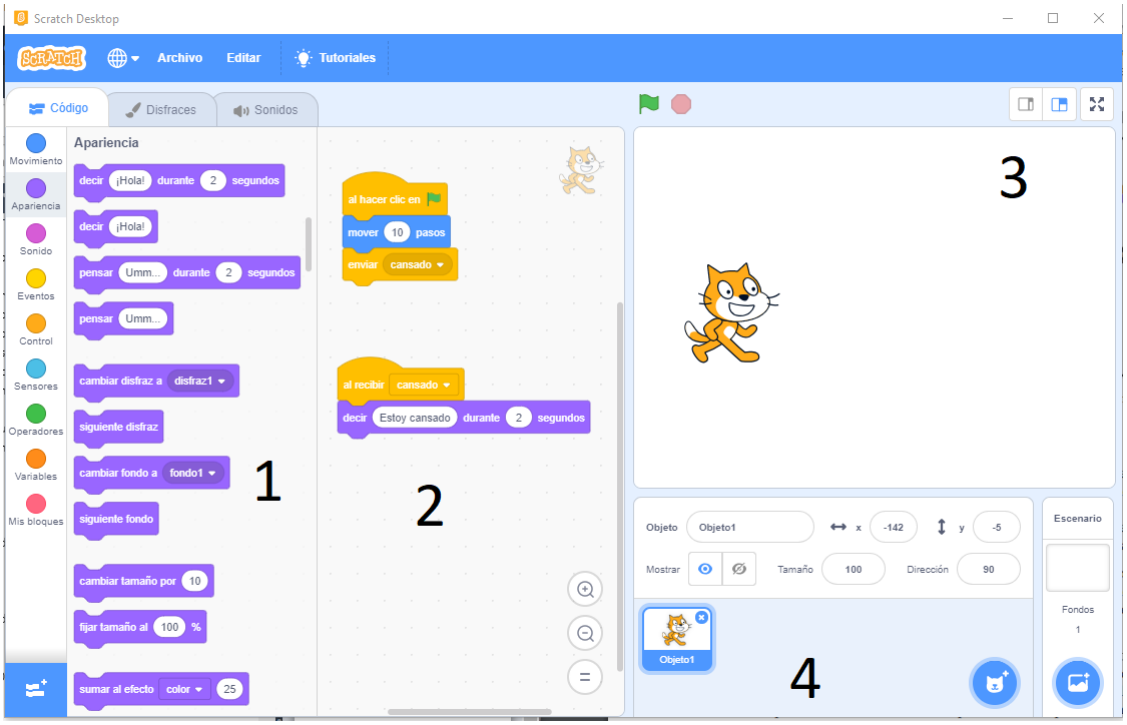

Figura 4.5: Ejemplo de programa hecho en Scratch, con las distintas regiones de la interfaz separadas por números.

- **Columna de bloques.** Es la zona que indica el número 1 de la figura 4.5. El color del bloque hace referencia a su tipo (movimiento, operadores, variables...). Si pulsamos en uno de estos tipos, nos mostrará su correspondiente grupo de instrucciones. Para usarlos, simplemente hay que arrastrarlos al área de programa, o bien pulsarlos y la instrucción se realizará en ese instante. Si se arrastran bloques a esta zona, quedarán eliminados.

- **Área de programa.** Es la zona central, marcada con el número 2 en la figura 4.5. Aquí se arrastran los bloques para programar. Solo se pueden juntar bloques si la sintaxis lo permite. En el ejemplo de la figura , no podemos situar el bloque de morado antes de la instrucción de recibir un mensaje. Por otro lado, si tenemos un conjunto de bloques juntos, no hace falta moverlos, eliminarlos o clonarlos por separado, se pincha en el bloque que está en la cima y la acción que se quiera realizar se aplica a todo el conjunto. Para saber el punto inicial por donde ha de empezar la secuencia de instrucciones, su utiliza el bloque de la bandera verde.

- **Escenario.** Es la región señalada con el número 3 en la figura 4.5. Consiste en un sistema de coordenadas cartesianas en dos dimensiones, con un tamaño de 480 píxeles de ancho y 360 de alto. Aquí se observa el resultado de toda la programación rea-

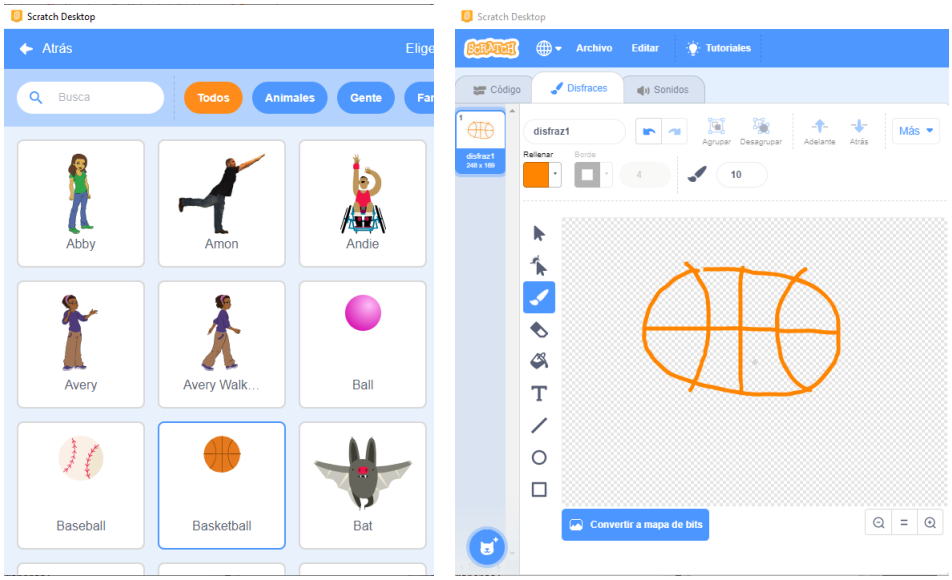

Figura 4.6: Distintas formas de crear objetos en Scratch, mediante la biblioteca (izquierda) o dibujando (derecha).

lizada en el área de programa. Para visualizarlo, debemos pulsar la bandera verde de la parte superior.

- **Espacio de objetos.** Es el área donde se encuentran los distintos objetos y escenarios que se van a usar, así como los mecanismos necesarios para crearlos o importarlos. Está indicado con un 4 en la figura 4.5.

Cada objeto tiene una parte de código que sólo se ejecuta sobre él. Esto es para que el usuario pueda estructurar su código de forma más ordenada y le ayuda a que sepa qué fragmento de código le corresponde a cada objeto, gracias a las imágenes que hay para reconocerlos. Los objetos se pueden crear de varias maneras (véase la figura 4.6):

- A través de la biblioteca de Scratch, en la que aparecen ordenados y separados por temáticas.
- Creando uno nuevo desde cero, partiendo del editor de dibujo que proporciona Scratch.
- Cargando una imagen de nuestro ordenador al programa.

Cuando creamos un objeto, podemos ver sus características (nombre, tamaño, posición, etc.) en el espacio de objetos y modificarlas.

Los fondos se crean de forma similar y es recomendable ajustarlos al tamaño del escenario de la interfaz (480 píxeles de anchura y 360 de altura).

En cuanto al código del programa, la distinción que tiene Scratch con respecto a otros lenguajes de programación es el uso de bloques predefinidos en lugar de líneas. Cada bloque es una instrucción que realiza el programa.

Los bloques que tienen el mismo color indican que pertenecen al mismo tipo, de forma que al usuario le es más fácil reconocer el cometido de cada bloque mientras programa. Existen nueve tipos de bloques:

- **Movimiento.** Hacen que el objeto se sitúe en distintos lugares del escenario, moverlo hacerle girar, etc.

Figura 4.7: Algunos bloques de tipo 'Control' en Scratch.

- **Apariencia.** Pueden hacer cambiar el aspecto del objeto haciendo que aparezca y desaparezca, ponerle efectos o modificar sus disfraces.
- **Sonido.** Son los bloques que modifican el volumen de los sonidos, ponen o quitan música, etc.
- **Eventos.** Gestionan los sucesos de la agrupación de instrucciones, inicializando otros grupos o enviando y recibiendo mensajes.
- **Control.** Gestionan el flujo de las instrucciones. Son bloques que simulan el funcionamiento clásico de los bucles 'while' o de las estructuras de control 'if-then' o 'if-then-else'. Tambíen pueden clonar objetos.
- **Sensores.** Detectan estados del programa durante su ejecución, por ejemplo preguntarle algo al usuario y, en función de su respuesta, realizar una u otra acción.
- **Operadores.** Son los bloques que representan las operaciones algebraicas, como el sumar o multiplicar; y las conectivas lógicas, como el 'and' o el 'or'. También hay instrucciones de manipulación de cadenas de caracteres que nos resultará útil durante la implementación de la máquina Enigma.
- **Variables.** Sustituyen a los bloques de 'Datos' de versiones anteriores de Scratch. Permiten declarar variables y cambiar sus valores.
- **Mis bloques.** Scratch permiten crear bloques como si fueran métodos o funciones con parámetros. Cada bloque de este grupo tiene una entrada, que puede ser numérica, lógica o simplemente una etiqueta.

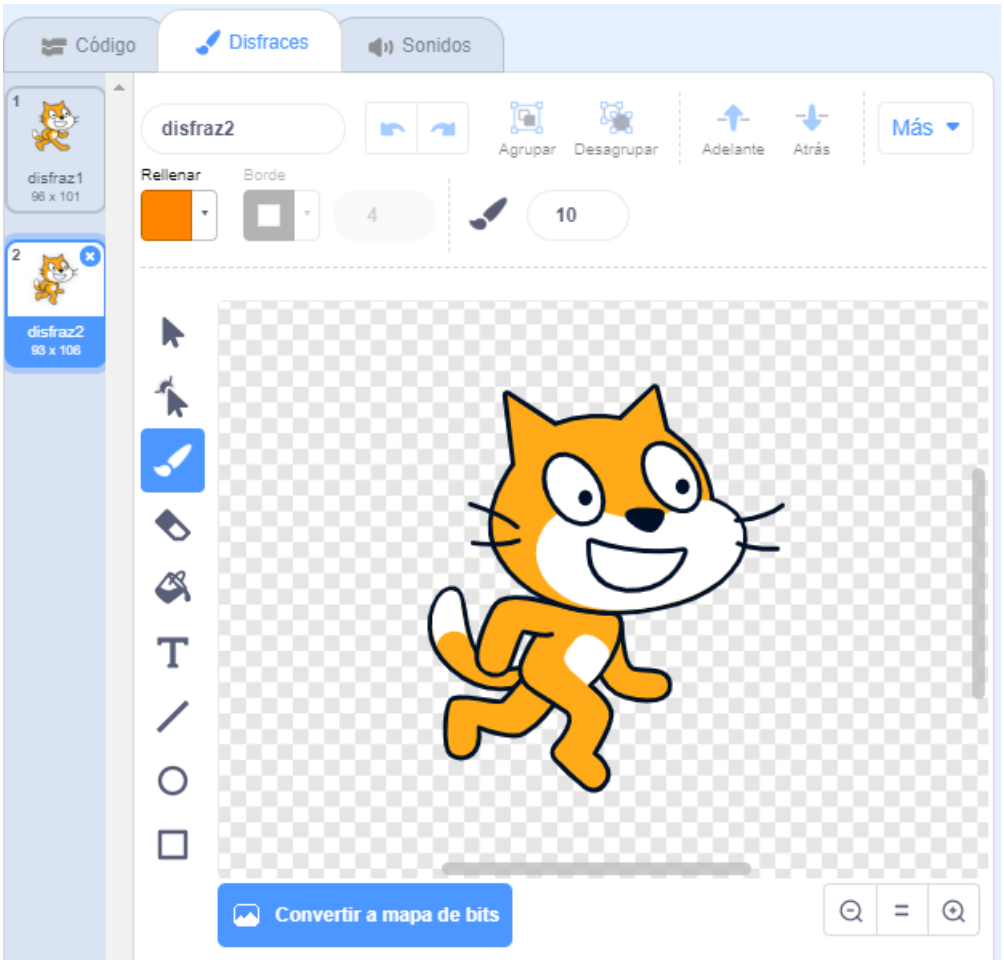

Figura 4.8: Los dos disfraces que tiene el gato de Scratch.

Aunque no vamos a explicar cada bloque por separado independientemente de su tipo, en el capítulo 5 se describirá cada instrucción que sea importante para la implementación del simulador, en especial aquellas cuyo funcionamiento sea difícil de intuir.

Por último, nos encontramos con el panel multimedia, que incluye las pestañas de disfraces y de sonidos.

En realidad, en la figura 4.6, cuando creábamos un objeto seleccionando el balón de baloncesto o dibujándolo, lo que hacíamos era dotar al objeto de un disfraz. Mediante los disfraces se pueden hacer animaciones para cada uno de los objetos. Un objeto puede tener varios disfraces y alternar entre ellos con los bloques de 'Apariencia'. Por ejemplo, el famoso gato de Scratch consiste en dos disfraces (véase la figura 4.8), uno con las piernas del gato en una posición y el otro con otra postura. De este modo, al alternar ambos disfraces y mover el objeto, se simula que está caminando.

Igual que con los disfraces, cada objeto dispone de un panel de sonidos (véase la figura 4.9). Aquí se pueden comprobar todos los sonidos que se asignan a un objeto, pudiendo reproducirlo en el instante de ejecución que elijamos, modificar su duración, añadirle efectos, recortarlo, etc. Podemos conseguir sonidos desde la biblioeca de Scratch, grabarlos con un micrófono o importándolos desde local.

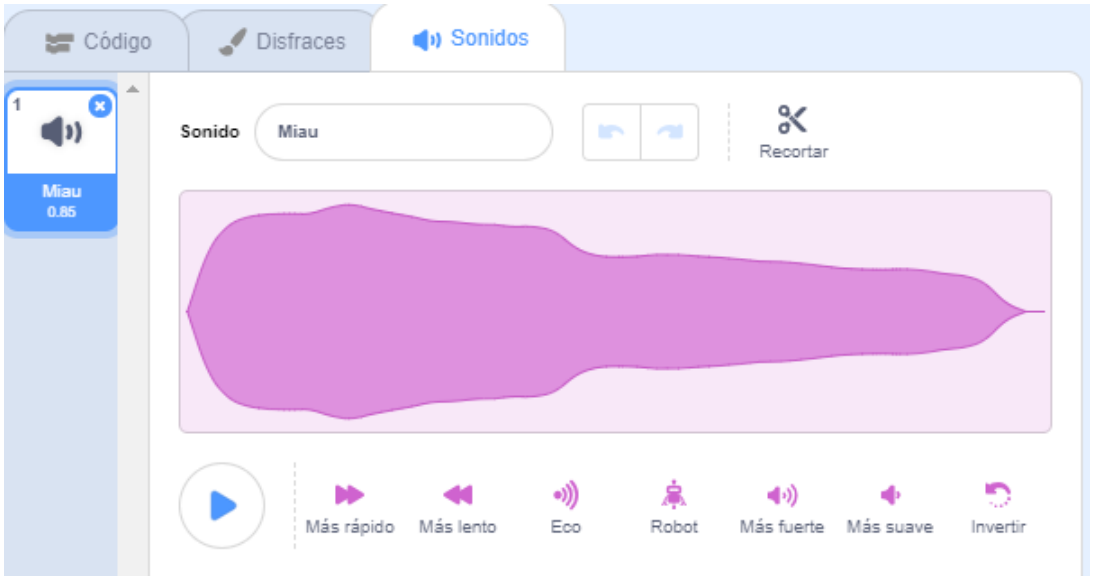

Figura 4.9: Sonido del gato de Scratch.

4.5 ¿Por qué escoger Scratch?

En síntesis, las principales causas que nos han llevado a escoger este lenguaje de programación son:

- **Compartición de proyectos.** Gracias a que cualquier proyecto compartido en la página web de Scratch puede ser modificado por otro usuario, las herramientas que usa una determinada persona pueden resultar útiles para otra.
- **Fácil utilización.** Scratch es un lenguaje ideal para que niños o adultos ajenos al mundo de la programación aprendan. La simplicidad del método de bloques permite obtener nociones que a futuro pueden servir en lenguajes de programación de más alto nivel.
- **Entorno de programación.** El ambiente de trabajo que proporciona Scratch es intuitivo y ordenado, al mismo que tiempo que las herramientas que facilita no son escasas en comparación con otros lenguajes más complejos, a pesar de que no posea algunas características útiles de estos, como puede ser la importación de librerías o correctores automáticos de sintaxis.
- **El factor social.** No es necesario tener una cuenta de Scratch para poder hacer uso del programa. Además, el desarrollo del pensamiento lógico y computacional mejora la comprensión de un ámbito de la informática que en principio es complicado. Al mismo tiempo, se trata de un software libre y gratuito, multilenguaje y disponible en diversos sistemas operativos. Todos estos hechos podrían explicar el gran éxito de Scratch, que a día de hoy cuenta con más de cuarenta millones de proyectos y usuarios.

CAPÍTULO 5

Diseño e implementación del simulador de la máquina Enigma en Scratch

En este capítulo vamos a describir el procedimiento seguido para implementar la máquina Enigma en el lenguaje Scratch. Además, explicaremos las instrucciones del simulador realizado.

5.1 Metodología empleada

En primer lugar, se han buscado otros simuladores de la Enigma[1] y vídeos en la red que detallaban su funcionamiento[2]. Esto ha ayudado a, cuando tenemos implementados determinados aspectos de la máquina en nuestro simulador, comprobar que funcionan correctamente.

Una vez se ha probado suficientemente la máquina, pasamos a la búsqueda de materiales y medios. En nuestro caso, se buscan imágenes, enlaces y cualquier posible recurso que refuerce su entendimiento. Como indicamos en el primer capítulo de este trabajo, la información que nos ha servido de apoyo para comprender la máquina Enigma se encuentra en una página web dedicada a la criptografía [16].

Después, se ha elaborado un esbozo de cómo estaría estructurado el simulador y la función de cada una de sus partes. En la figura se puede apreciar un primer bosquejo que servirá de base para el posterior diseño del simulador.

A continuación se ha adaptado ese boceto a Scratch, relacionando cada fase y elemento con escenarios y objetos, respectivamente. En posteriores apartados se explicará como se han creado o importado los diferentes componentes al entorno de Scratch.

Posteriormente hacemos el diseño íntegro del simulador con todas sus partes en Scratch y procedemos a la implementación de la Enigma. Por último, se han cifrado textos de un tamaño considerable en nuestro simulador y en otros, para verificar que el programa entero resultante es correcto.

En cuanto a la distintas versiones que existen de la máquina Enigma, el simulador cuenta con aquellas que pueden considerarse más importantes debido a su papel histórico (véase el capítulo 2). Estas son la versión Wehrmacht Enigma-I y la Enigma M3 y M4 utilizadas por la Kriegsmarine.

[1] http://enigmaco.de/enigma/enigma_es.html
[2] https://www.youtube.com/watch?v=XK_1gUo8YDE&t=1s

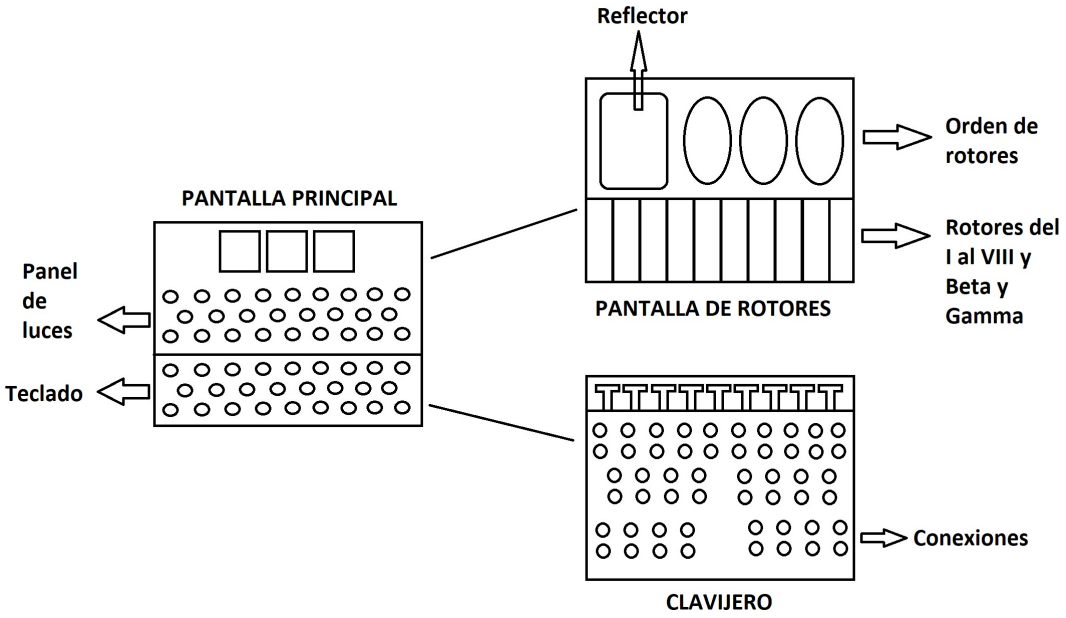

Figura 5.1: Esbozo realizado para el diseño del simulador.

5.2 Organización

El simulador está compuesto por cinco pantallas:

- **Presentación.** Es la primera pantalla que nos encontramos al ejecutar el simulador (véase la figura 5.2). Desde ella podemos acceder a las instrucciones de uso.
- **Instrucciones.** Aquí se encuentran los controles y funciones del programa, separados por las tres pantallas esenciales del simulador, tal como aparece en la figura 5.3. Una vez pasemos esta pantalla, solo podremos volver a ella desde la principal.
- **Principal.** Es la pantalla donde se cifra o descifra una letra, una palabra, un texto, etc. Para hacerla, hemos tomado una fotografía de una máquina Enigma real (véase la figura 5.4).

  Por un lado, tiene el teclado y el panel de luces en la parte inferior y central, respectivamente. Si pulsamos una tecla en nuestro teclado, su correspondiente en el teclado del simulador se ensombrecerá. Al soltar la tecla, se iluminará la letra cifrada en el tablero de luces, la cual depende de como estén configurados el clavijero y los rotores.

  Por otro lado, dispone de un monitor de tres o cuatro partes (dependiendo del modelo de la Enigma) donde podemos formar la clave del mensaje. Desde esta pantalla podemos acceder a las instrucciones, a los rotores o al clavijero.
- **Clavijero.** Desde aquí establecemos las conexiones entre distintas letras pulsando sobre ellas de manera consecutiva. En la figura 5.5 se puede apreciar su aspecto.
- **Rotores.** Es la pantalla donde configuramos todo lo relacionado con ellos: orden, selección, cableado interno, etc. Además, podemos cambiar el modelo de la máquina Enigma. En total, hay seis versiones a elegir: dos versiones de la Wehrmacht, cada una con un reflector distinto; dos modelos Kriegmarine M3 con los mismos reflectores que la Wehrmacht y dos modelos Kriegsmarine M4 con los reflector B y C finos.

Figura 5.2: Pantalla de inicio del simulador.

Figura 5.3: Pantalla de instrucciones del simulador.

Figura 5.4: Pantalla principal del simulador, realizada a partir de una imagen de una máquina Enigma de cuatro rotores.

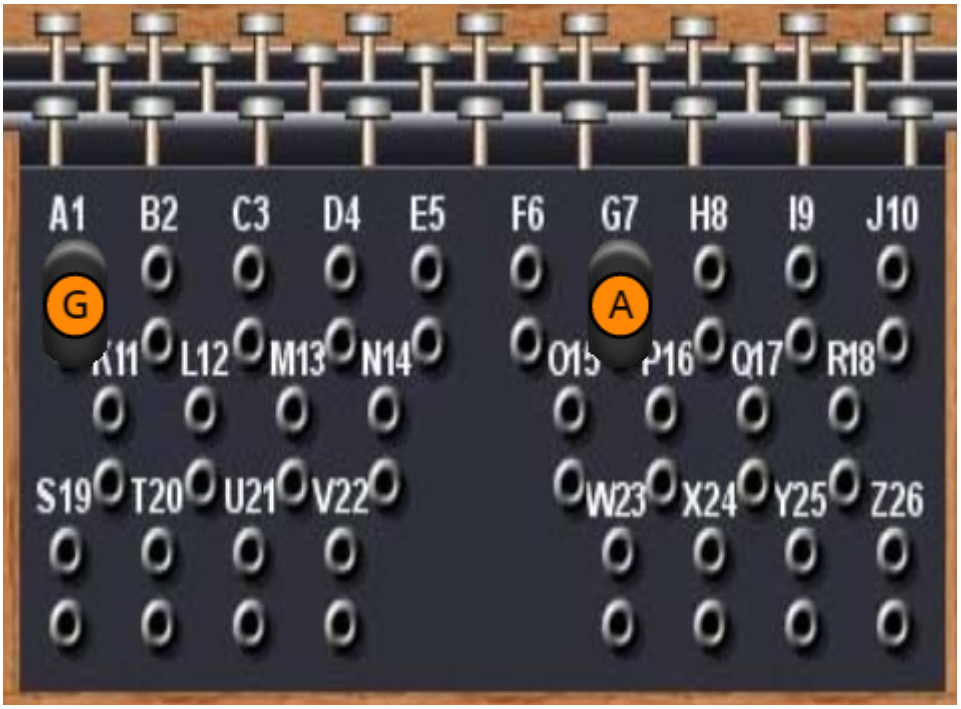

Figura 5.5: Pantalla del clavijero con las letras 'A' y 'G' cableadas. Para deshacer esta conexión, basta con pulsar una de ellas en nuestro teclado.

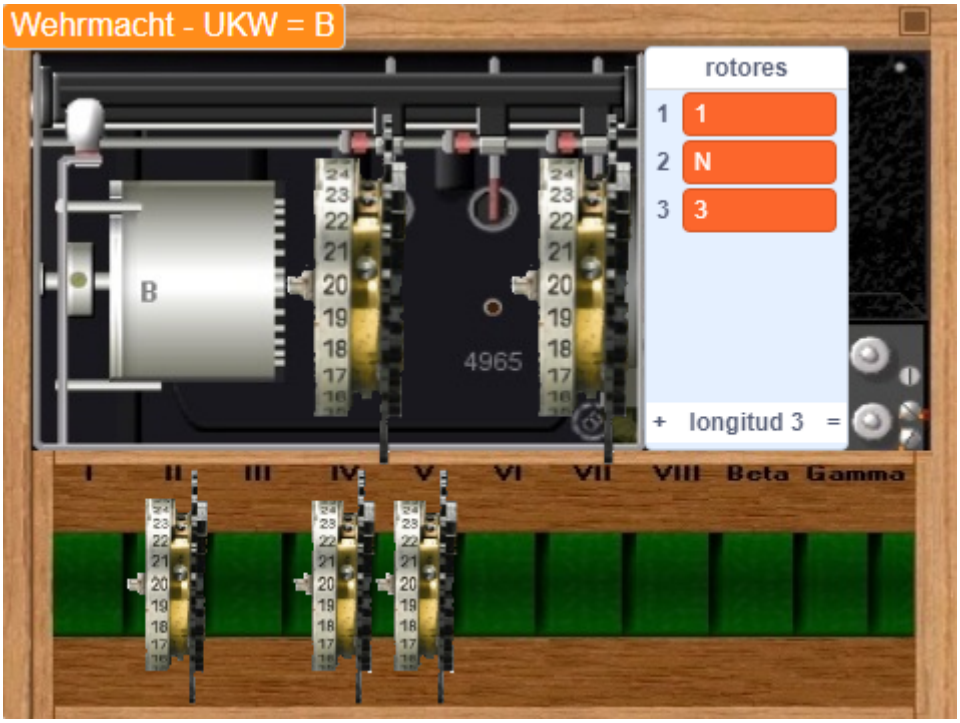

Figura 5.6: Pantalla de rotores. En la parte superior izquierda se indica el modelo de Enigma que se está usando. En la parte derecha, la lista de Scratch muestra qué rotores se están usando y su orden ('N' quiere decir que no hay ninguno en esa posición). En la parte inferior, tenemos los rotores que se pueden usar en ese modelo seleccionado. Para poder volver a la pantalla principal, debemos haber dispuesto todos los rotores en orden, sin dejar ningún hueco en la parte superior.

En la siguiente sección, explicaremos en profundidad los aspectos más importantes de la implementación de este simulador.

5.3 Implementación

5.3.1. Herramientas básicas utilizadas

Para empezar, hemos definido una serie de variables y listas que nos ayudan a orientarnos durante la programación. Por un lado están las variables principales, que son los rotores (figura 5.7), sus alfabetos (figura 5.8), y los displays (figura 5.9 y 5.10). Por otro lado, tenemos variables auxiliares ('aux', 'pos', 'cont') y variables que sirven a un determinado propósito dentro de los métodos del programa.

Dentro de este último grupo tenemos 'begin', que nos servirá para saber cuándo pasamos de un fondo a otro (véase el apartado 5.3.2); 'clavesAlfanuméricas', que es una palabra con todos las letras del alfabeto ordenadas ('ABCD...') y que usaremos a lo largo de la implementación; 'controlPlugboard' y 'letraPlugboard' se usan en la implementación del clavijero y se explicarán más adelante, igual que 'posX1', 'posX2', 'posY1', 'posY2'; 'entrada', que la utilizamos en el proceso de cifrado; 'letra' guarda la letra que estamos pulsando para cifrar; 'letraCifrada' es la letra que el simulador devuelve como salida al finalizar el cifrado; 'modelo' indica con qué versión de Enigma estamos trabajando; 'modeloSelec' es similar a la anterior, pero tiene un valor númerico en vez de una cadena de caracteres; 'mostrandoRotor' vale '1' si estamos modificando el *Ringstellung* de un rotor y '0' en caso contrario; 'girar1', 'girar2' y 'girado2' se utilizan en el proceso de cifrado para saber cuando girar el primer y el segundo rotor; y 'sentido', que indica si estamos

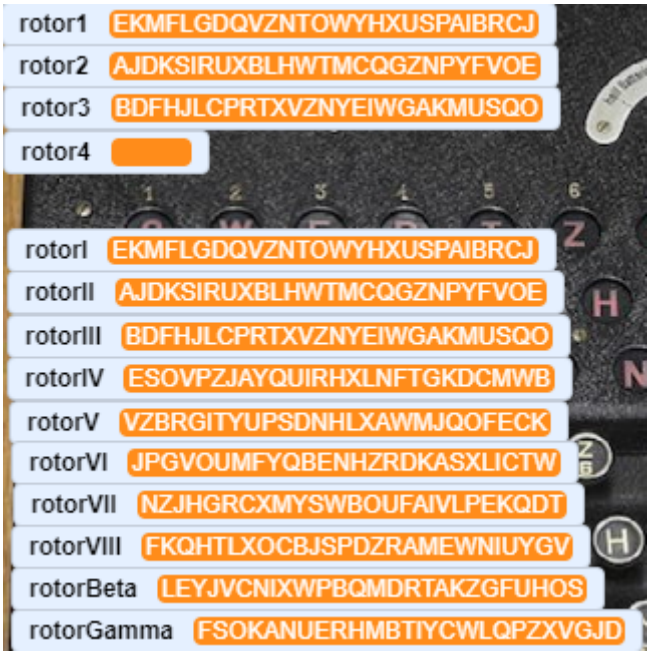

Figura 5.7: Variables dedicadas a los rotores. Las que acaban en números romanos representan los diferentes rotores que podía tener la máquina Enigma y no se pueden modificar. Por otro lado, las que acaban en números del 1 al 4 representan los rotores que están seleccionados en ese instante de ejecución, y pueden cambiar a lo largo de la simulación. En el ejemplo se puede apreciar que se trata de un modelo de tres rotores, cuyos elegidos son el I, el II y el III, en ese mismo orden.

modificando un determinado componente de la clave del mensaje en orden correcto ('1') o inverso ('0').

Con respecto a las listas, tenemos un conjunto de ellas que implementan los reflectores (véase la figura 5.11) y otra llamada simplemente 'reflector', que indica cuál de los cuatro posibles se está usando en ese momento. También hay dos listas que reflejan los puntos de rotación de los rotores (véase la figura 5.12). Después están 'listaPlugboard1' y 'listaPlugboard2', que hacen las veces de una base de datos para llevar un control sobre las conexiones del clavijero (figura 5.13). Al mismo tiempo, hay una lista encargada de guardar los nombres de los diferentes modelos de la Enigma que vamos a implementar, llamada 'modelos' (figura 5.14). Por último, hay una lista 'rotores' que indica cuáles son los rotores que se están utilizando (véase el escenario de los rotores del apartado 5.3.2 para observar el uso que tiene esta lista).

Para terminar con este apartado, hemos creado tres métodos con el tipo de instrucciones 'Mis bloques'. Por una parte, 'posicion' es una función a la que se le pasan como parámetros una palabra y una letra. Así, guarda en la variable 'pos' la posición que ocupa esa letra en la palabra. Por otra parte, 'rotacionPositiva' es un método que, dada una palabra, elimina la primera letra y la coloca al final, guardando el resultado en la variable 'aux', sin modificar la palabra original. Finalmente, 'rotacionNegativa' hace algo parecido a la anterior, pero esta vez elimina la última letra y la coloca al principio, de nuevo sin modificar la palabra que se pasa como argumento. En la figura 5.15 se pueden apreciar estos métodos.

Cabe destacar el uso que hemos hecho de los bloques 'enviar' y 'Al recibir', de tipo 'Eventos', para implementar mensajes entre distintos objetos del programa. A pesar de que muchos de ellos pueden considerarse métodos como los de la clase 'Mis bloques', se ahondará en ellos en el apartado 5.3.3.

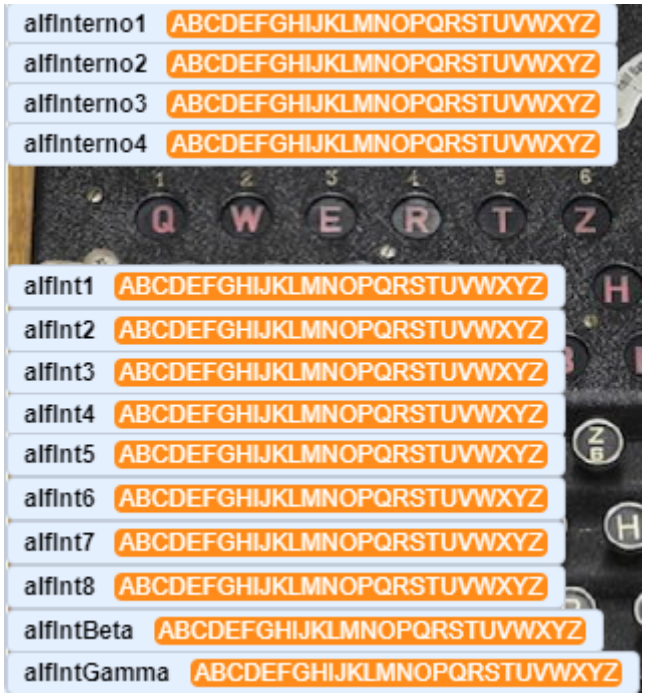

Figura 5.8: Variables de los alfabetos. Los del tipo 'alfInt' son los alfabetos internos de cada rotor, es decir, representan su anillos. Cada vez que modifiquemos el *Ringstellung* de un cierto rotor, cambiará su variable 'alfInt' asociada. Una vez que estén modificados todos los 'alfInt' de los rotores que vamos a usar, se pasará el valor a las variables tipo 'alfInterno'. Por ejemplo, si usamos el rotor I en la configuración de *Ringstellung* B-02, su valor será 'ZABCD...'. Si lo situamos en la posición 3, al cambiar a la pantalla principal para cifrar o descifrar, 'alfInterno3' tendrá ese valor.

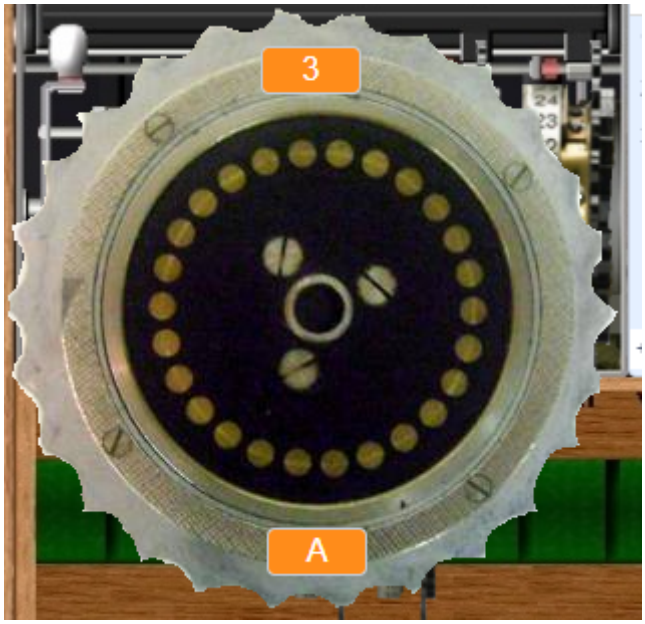

Figura 5.9: Los displays de tipo 'DisplayAlfInterno' se utilizan cuando estamos configurando el *Ringstellung* de un cierto rotor (que se identifica con la variable 'DisplayRotor') y nos indican su configuración. En el ejemplo, observamos el rotor 3 configurado en A-01, por lo que en ese instante 'DisplayRotor' vale '3' y 'DisplayAlfInterno3' tiene un valor 'A'.

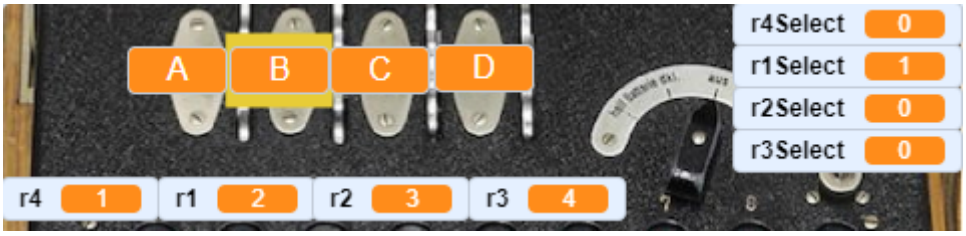

Figura 5.10: Las variables 'r1Display', 'r2Display', 'r3Display' y 'r4Display' representan las distintas componentes de la clave del mensaje. Hay que tener en cuenta que 'r4Display' es la que se encuentra en el extremo izquierdo, el resto están ordenadas del 1 al 3. Las variables 'r' indican la posición que tiene esa componente de la clave en el alfabeto ('r2' es '3' porque la tercera componente de la clave, que está asociada al segundo rotor, es 'C'). Por último, las variables 'Select' indican si la componente está siendo seleccionada en ese instante ('1') o no ('0').

Figura 5.11: Listas de Scratch que implementan los cuatro reflectores de la máquina Enigma, donde el índice y el valor de cada elemento son los emparejamientos de pines de cada uno de ellos. 'B3' y 'C3' representan los reflectores anchos, mientras que 'B4' y 'C4' los finos.

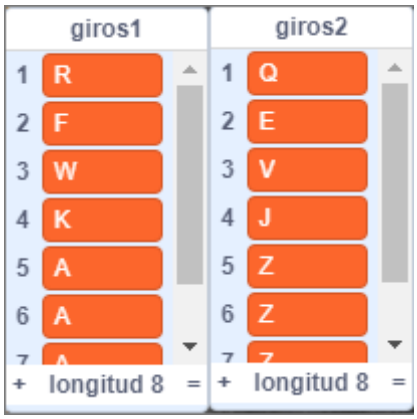

Figura 5.12: Listas que representan los puntos de rotación de cada rotor (véase el apartado 3.1.5 para más información).

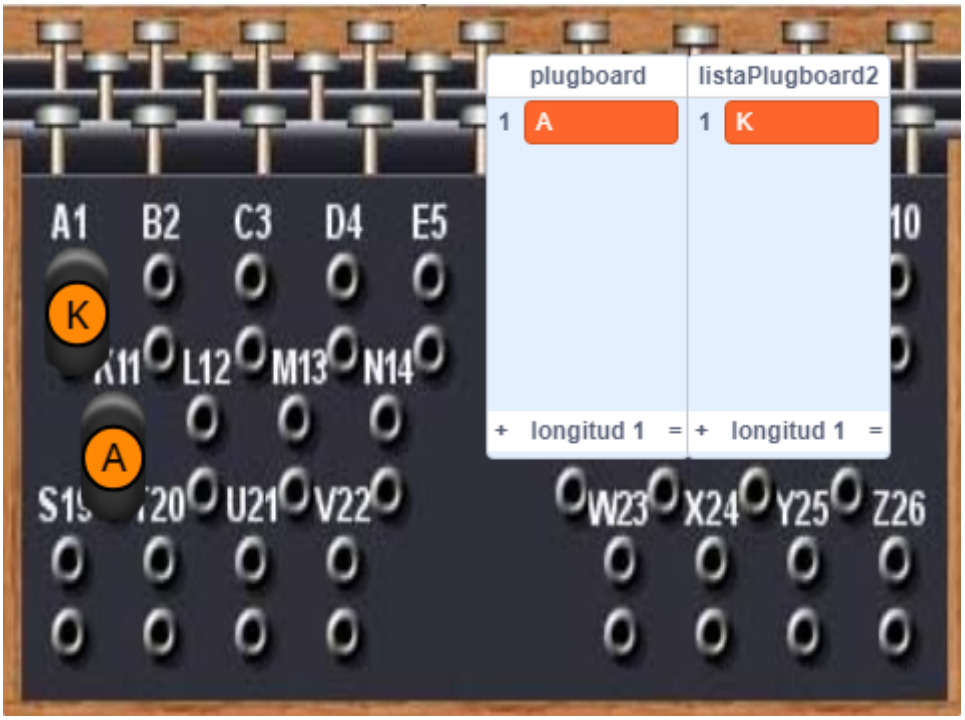

Figura 5.13: Un ejemplo de uso de 'listaPlugboard1' y 'listaPlugboard2'.

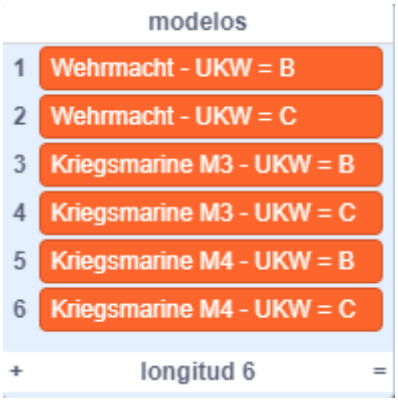

Figura 5.14: En 'modelos' se almacenan los nombres de las versiones de la máquina Enigma presentes en el simulador.

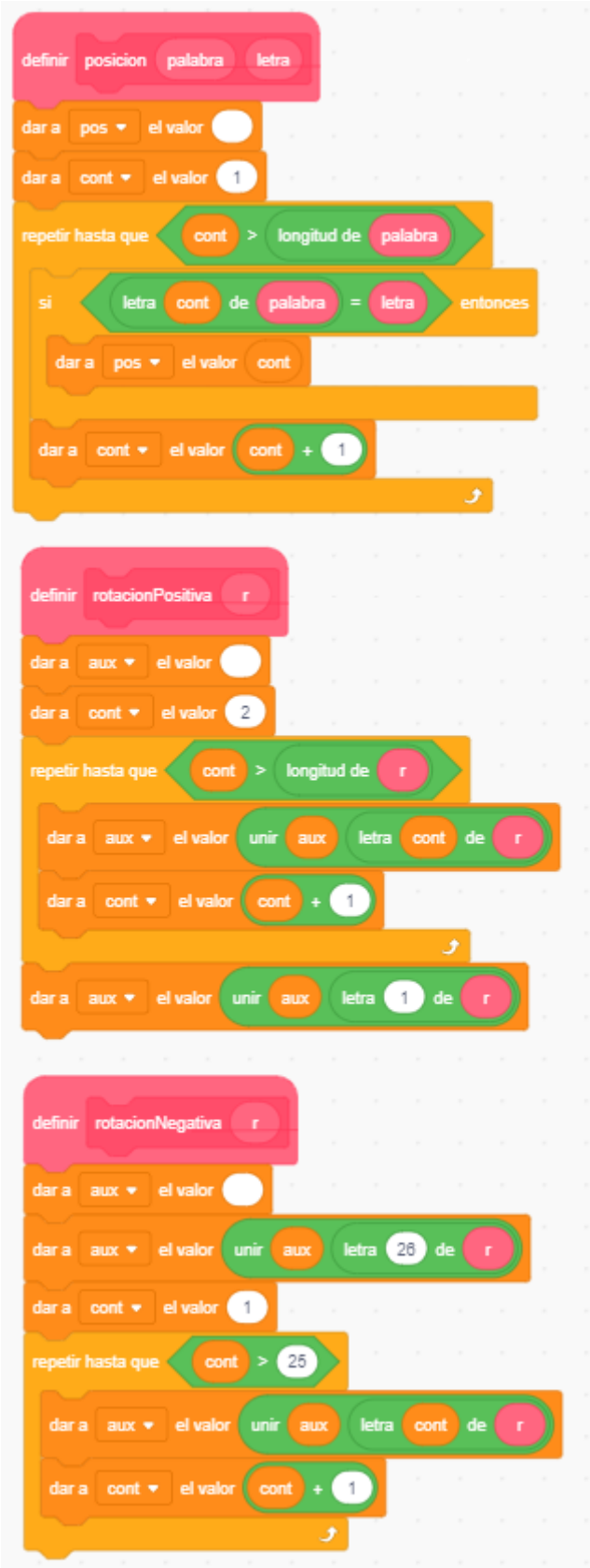

Figura 5.15: Las tres funciones de tipo 'Mis bloques' creadas en este simulador.

Figura 5.16: Los distintos fondos con sus respectivos números de identificación en Scratch.

5.3.2. Escenarios

Hemos importado diez fondos que representan las cinco pantallas del simulador. En realidad, son los mismos que se pueden apreciar en la sección 5.2, con la diferencia de que la pantalla de rotores tiene 6 fondos asociados, uno por cada modelo de Enigma que queremos implementar. También hemos importado un clip de audio llamado 'sonido-Maquina' que representa el cambio de un escenario a otro[3]. El resto de información que aporta cada fondo y que no se encuentra en la figura 5.16 se agrega como variable visible, objeto o disfraz de un determinado objeto, debido a que los fondos no tienen disfraces.

El código empleado para los escenarios tiene la función principal de hacer que el usuario se vaya desplazando entre ellos. Por otra parte, debido a que hay objetos y variables que no deben ser visualizados en un determinado fondo, se han de esconder mediante los bloques de instrucciones pertinentes. Las distintas transiciones entre fondos son:

- Desde la pantalla principal hasta los rotores y viceversa, solo hay que pulsar la tecla 'espacio' y se ejecutará el conjunto de bloques de código que se puede apreciar en la figura 5.17 y en la 5.18.
- Entre la pantalla principal y el clavijero es necesario pulsar la tecla '1' y se realizará el grupo de bloques de la figura 5.19.
- Dentro de la pantalla de rotores, para moverse entre los diferentes modelos se pulsan las flechas izquierda y derecha, tal como se explica en las figuras 5.20 y 5.21.
- Moverse entre la pantalla de presentación, de instrucciones y la principal, como se puede observar en la figura 5.22. Este método está en un objeto a parte debido a que los fondos '9' y '10' tienen un disfraz de la presentación al simulador y las instrucciones, respectivamente.

[3] https://clyp.it/ji2augsn

Figura 5.17: Primera parte del código utilizado para moverse entre la pantalla principal y los rotores. Al pulsar 'espacio', si el número de fondo es '1' quiere decir que estamos en la pantalla principal y tenemos que ir a los rotores. Por tanto, iniciamos el sonido, escondemos los displays junto con su selector y apagamos el panel de luces. Posteriormente, mostramos los modelos seleccionados con 'modeloSelec', la lista de rotores y, dependiendo del valor de 'modelo', se nos mostrará el fondo que corresponda.

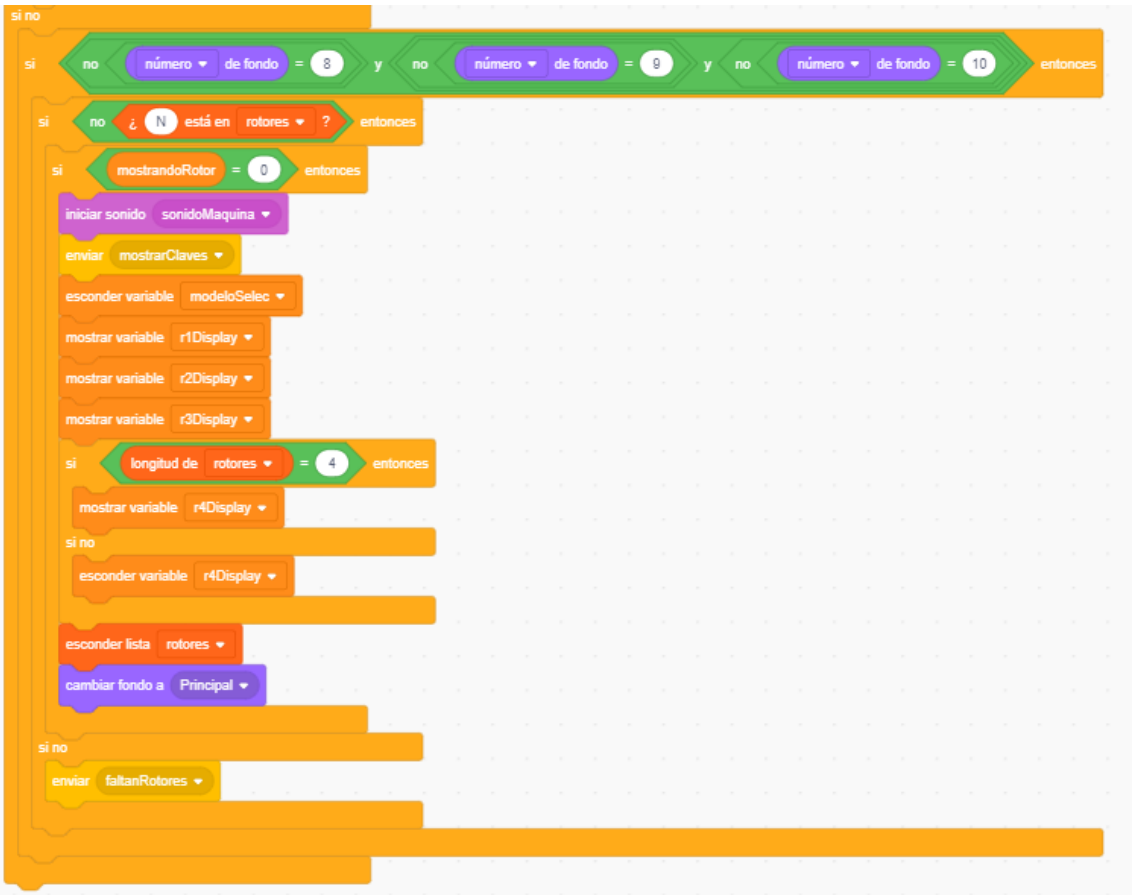

Figura 5.18: Segunda parte del código usado para moverse entre la pantalla principal y los rotores. Al pulsar 'espacio', si el fondo no es '1', debemos comprobar que no estamos ni en el '8', el '9' o el '10' (el resto de fondos son los de los rotores). Si es así, comprobamos que no falta ningún rotor en la lista 'rotores' (en caso de que falten, 'faltanRotores' mandará un aviso). Luego, verificamos que no estamos cableando ningún rotor. Entonces, iniciamos el sonido, escondemos el modelo de Enigma y la lista de rotores, cambiamos el fondo al principal y mostramos las claves y los displays (si la longitud de 'rotores' es 4, quiere decir que estamos en un modelo M4, por lo que se visualizará el cuarto display).

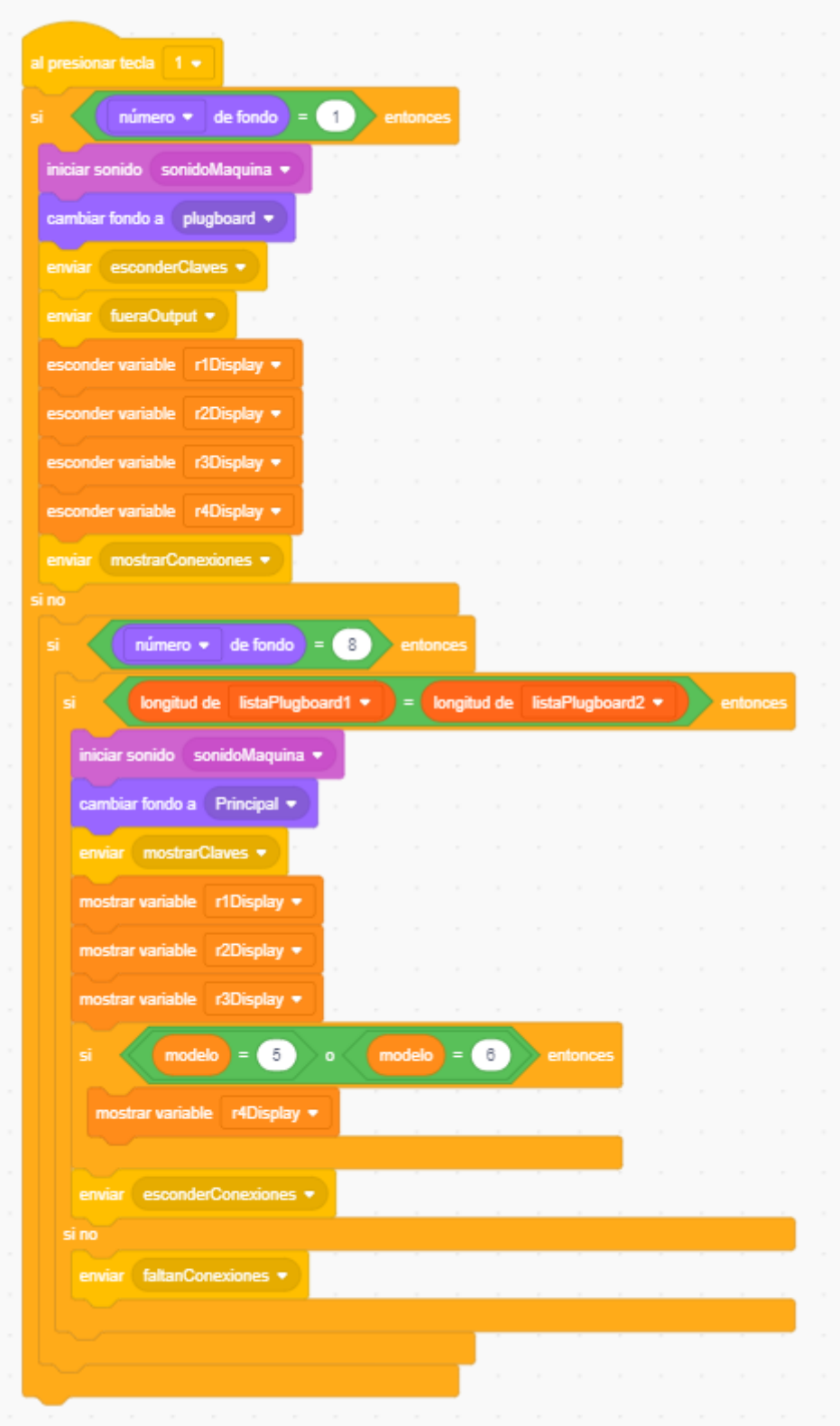

Figura 5.19: Código utilizado para desplazarse entre la pantalla principal y el clavijero. Por un lado, si el número de fondo es '1', quiere decir que estamos en la pantalla principal. Por tanto, se iniciará el sonido, escondemos los displays junto con su selector y apagamos el panel de luces. Finalmente, mostramos las conexiones que hayan en el clavijero. Por otro lado, si el número de fondo es '8', nos encontramos en el clavijero. Ahora verificamos que no hay conexiones a medias comprobando que las listas 'listaPlugboard1' y 'listaPlugboard2' tiene la misma longitud (si no es así, 'faltanConexiones' nos enviará un aviso). Si se cumple, iniciamos el sonido, cambiamos de fondo, mostramos los tres displays (si el modelo es M4 mostramos el cuarto display) y escondemos las conexiones del clavijero.

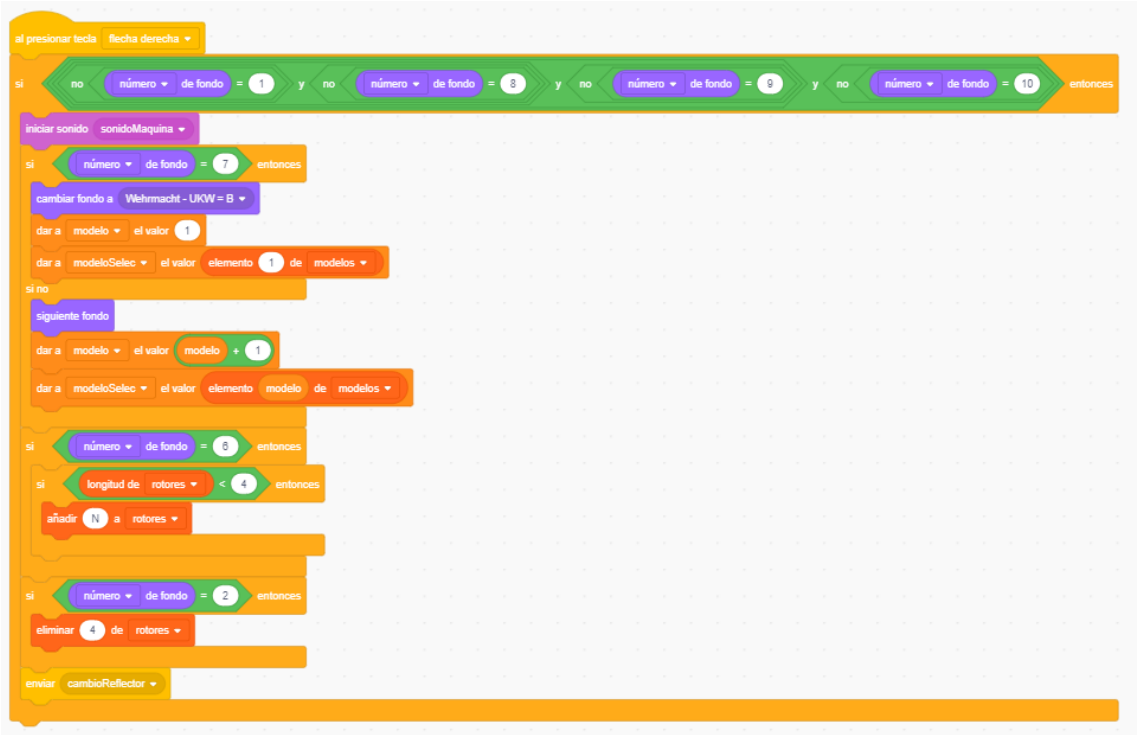

Figura 5.20: Parte de código encargada de cambiar de modelo de Enigma cuando se pulsa la flecha derecha. Al hacerlo, se comprueba que estamos en uno de los fondos de los rotores. Si el número de fondo es '7', quiere decir que estamos en el último fondo de rotores, por lo que nos movemos al primero; si no es así, avanzamos hacía el siguiente. Cuando pasamos al fondo '6', estamos pasando al modelo M4, por lo que añadimos un cuarto elemento a la lista 'rotores'. De forma similar, cuando pasamos al fondo '2', nos desplazamos a una versión de tres rotores, por lo que eliminamos el cuarto componente de esa lista. Finalmente, enviamos 'cambioReflector', ya que con el orden preestablecido de modelos siempre estamos cambiando de reflector.

Figura 5.21: Parte de código que se ejecuta al pulsar la flecha izquierda en un fondo de rotores. Debido a que no hay un bloque en Scratch que nos mueva hasta el fondo anterior, se sustituye el 'if-else' de la figura 5.20 por el código de la figura actual.

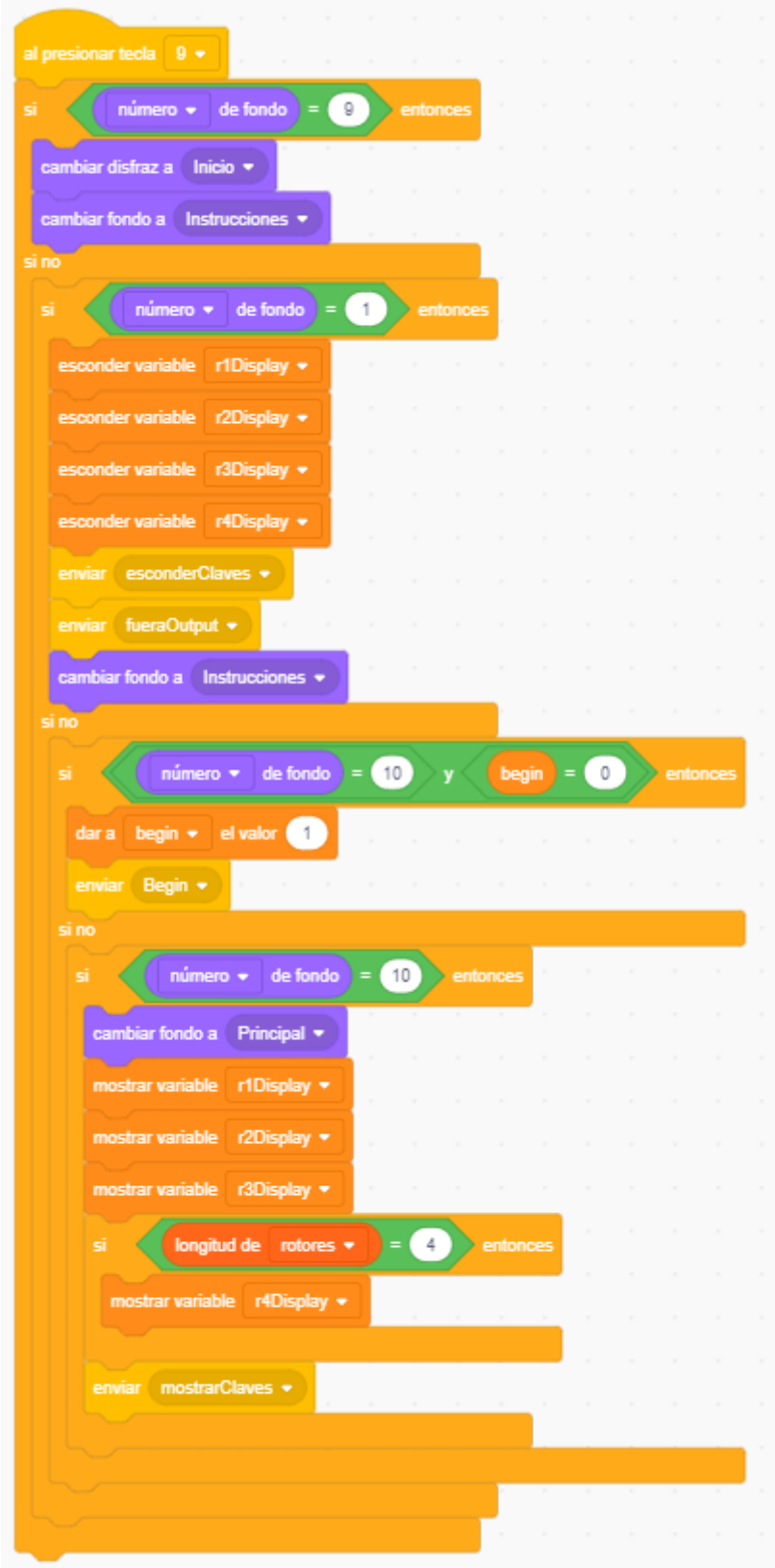

Figura 5.22: Grupo de bloques que permite desplazarse entre la pantalla de presentación, instrucciones y principal. Al pulsar '9', si estamos en el fondo número '9' (pantalla de presentación), cambiamos de disfraz para quitar el texto de presentación y cambiamos el fondo a la pantalla principal. Si estamos en la pantalla principal, escondemos los displays junto con su selector, apagamos el panel de luces y cambiamos el fondo a la pantalla de instrucciones. Por último, si estamos en la pantalla de instrucciones hay dos contextos posibles. Si venimos de la pantalla de presentación, hay que inicializar el programa con 'Begin' y movernos a la pantalla principal; y si venimos de la pantalla principal simplemente mostramos displays y cambiamos de fondo.

5.3.3. Objetos y disfraces

En total, el simulador en Scratch cuenta con 44 objetos y 3 clips de audio. Uno de estos es 'sonidoMaquina', que ya se ha utilizado en el apartado 5.3.2. Los otros dos son 'tecleo'[4] y 'sonidoRotor'[5], el primero para simular el sonido de teclear una máquina de escribir y el segundo emula el ruido que hacen los rotores al moverse.

El primer objeto que tenemos es 'Inicio/Instrucciones', que se encarga de implementar el código necesario para moverse entre las tres primeras pantallas (véase el apartado 5.3.2 para más información). Además, es donde comienza la ejecución de nuestro programa, como se puede apreciar en la figura 5.23. Cuenta con dos disfraces, uno de ellos transparente, que se utilizará cuando dejemos la pantalla de presentación; y el otro consiste en un texto de bienvenida al que asignamos el nombre 'saludos'.

Luego, se ha creado el objeto 'Inicialización', que formatea las variables y las deja en su valor original, tanto si es la primera vez que usamos el simulador como si lo acabamos de reiniciar. Se puede ver un ejemplo de esto en la figura 5.24. Como se puede apreciar, las variables han de inicializarse cuando se recibe el mensaje 'Begin', enviado cuando se pasa por la pantalla de presentación, la de instrucciones y la principal, en ese orden (véase las figuras 5.23 y 5.22). En este objeto también se encuentran las implementaciones de dos métodos que muestran mensajes de error. Uno es 'faltanRotores' y el otro 'faltanConexiones', a los que se les llama cuando dejamos rotores sin colocar o conexiones incompletas en el clavijero, respectivamente (véase la figura 5.25).

Posteriormente, tenemos el objeto 'Inputs'. Su función es iluminar en la pantalla principal la letra que el usuario haya pulsado en su teclado. Su disfraz es un círculo negro que se posiciona en la letra presionada. En las figuras 5.26 y 5.27 se puede observar el código implementado en esta parte.

A continuación está el objeto 'Cifrado', que podemos considerar uno de los más importantes, pues su cometido principal es encriptar una determinada letra. Para ello, se implementa el evento 'cifrar' (véase las figuras 5.28, 5.29, 5.30 y 5.31), que representa el recorrido por los rotores y alfabetos internos (teniendo en cuenta los puntos de rotación y la disposición del clavijero), lo que da como resultado una nueva letra. Esta se envía al objeto 'Outputs' con el mensaje 'letraCifrada'. Además de eso, tiene otro método llamado 'clavesModificadas' (véase la figura 5.32) que se encarga de modificar el rotor y el alfabeto interno correspondiente cuando cambiamos un display (si, por ejemplo, el selector cambia la última componente por la derecha de la clave del mensaje, hay que girar el tercer rotor y su alfabeto de manera equivalente).

Después, tenemos 'Outputs', que se encarga de iluminar la letra cifrada en el panel de luces. De manera similar al objeto 'Inputs', tiene por disfraz un círculo que se posiciona sobre la letra encriptada. Es de color blanco y tiene un efecto de desvanecimiento para simular el alumbrado (véase la figura 5.33). El código que implementa esta función se puede apreciar en la figura 5.34.

Luego, está 'Claves', que representa el selector de la clave del mensaje. El bloque de código inicial y los eventos implementados en él se muestran en la figura 5.35. La función principal de este objeto es cambiar el valor de los displays (véase la figura 5.36) mediante un disfraz, que consiste en un rectángulo amarillo móvil a lo largo de las cuatro componentes de la clave (figura 5.37).

Por otro lado, 'Displays' se encarga de hacer girar los componentes de la clave cuando estamos cifrando y un rotor hace girar al de su izquierda, como se refleja en la figura 5.38.

[4] https://clyp.it/ncbsfntc
[5] https://clyp.it/vkqzlvt2

También es posible que estemos modificando los rotores en su correspondiente pantalla, en cuyo caso el orden anterior ya no sirve y es necesario reiniciar los displays de la clave al volver a la pantalla principal. Por eso se envía el mensaje 'reinicioDisplays' (véase la figura 5.39, para que todos vuelvan a la configuración inicial.)

Otro punto son los diez objetos dedicados a los diez posibles rotores. Cada uno de estos objetos contiene un disfraz igual para todos. El disfraz tiene un tamaño de 150 si el rotor se está usando y de 100 si está en la alfombra verde para ser elegido. Por defecto, cuando empieza la simulación los tres rotores que se emplean son el I, el II y el III, en ese orden. El código de inicio será como el de la figura 5.40. Como los objetos solo pueden ser visualizados en la pantalla de rotores, cada uno tiene una parte de código dedicado a cuando transitamos de esta pantalla a la principal (véase las figuras 5.41, 5.42 y 5.43) . Para usar un rotor, hay dos tipos de código realizado, uno para los rotores del I al VIII (5.44) y otro para Beta y Gamma (5.45). Esto se debe a que los dos últimos solo pueden servir como cuarto rotor. Los rotores se van completando de derecha a izquierda cuando se pulsa sobre ellos. Finalmente, los rotores del VI al VIII solo están disponibles en los modelos M3 (numerados como 3 y 4) y Beta y Gamma en los M4 (5 y 6), por lo que también se ha tenido que implementar este aspecto (véase la figura 5.46 y 5.47 para más detalles).

Al mismo tiempo, 'rotorConf' se encarga de la configuración interna de los rotores, es decir, del *Ringstellung*. Para modificar el cableado de un rotor basta con pulsar el número en nuestro teclado ('9' y '0' para Beta y Gamma, respectivamente) y nos aparecerá un disfraz, que consiste en un rotor visto desde frente, como se muestra en la figura 5.9. En las figuras 5.48 y 5.49 se puede ver como se ha implementado la selección y la modificación de un rotor. En este objeto también se ha echado mano de los métodos 'posicion', 'rotacionPositiva' y 'rotacionNegativa' vistos anteriormente.

Por último, falta hacer funcionar el clavijero. Para ello, hay 26 objetos, uno por cada letra, que sirven para representar los extremos de los cables. Así, si queremos cablear 'A' con 'K', en el escenario saldrán los objetos 'cableA' y 'cableK', cada uno en la posición del otro, como en la figura 5.13. El código que muestra y oculta estos objetos cuando cambiamos de pantalla está en la figura 5.50. Mientras tanto, los bloques encargados de conectar y desconectar una letra se observan en la figura 5.51. Finalmente, hay que tener en cuenta los posibles casos que suceden al pulsar una determinada letra. Estos se han implementado en las figuras 5.52 y 5.53.

Figura 5.23: El presente grupo de bloques inicia la ejecución del simulador. Al pulsar en la bandera verde, escondemos las variables 'Display', 'modeloSelec', la lista de rotores y enviamos 'esconderClaves' para que no se pueda ver el selector de la clave del mensaje. Estas medidas se toman por si antes de pulsar en la bandera, nos encontramos en otro fondo. Finalmente, modificamos el fondo, cambiamos el disfraz a 'saludos' para que nos aparezca el mensaje de bienvenida y lo situamos en la posición -71 del eje X y 18 del Y (Esquina inferior izquierda del escenario).

Figura 5.24: Ejemplo de inicialización de algunas variables.

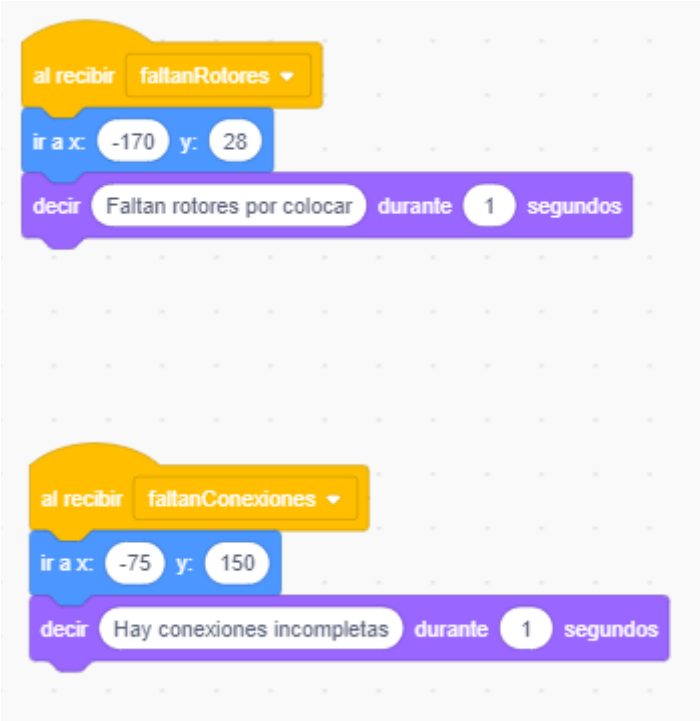

Figura 5.25: Los métodos 'faltanRotores' y 'faltanConexiones'.

Figura 5.26: Bloques que se ejecutan al inicio del simulador. El efecto 'desvanecer' le da un toque estético, mientras que el bucle infinito hace que solo se pueda pulsar una tecla al mismo tiempo.

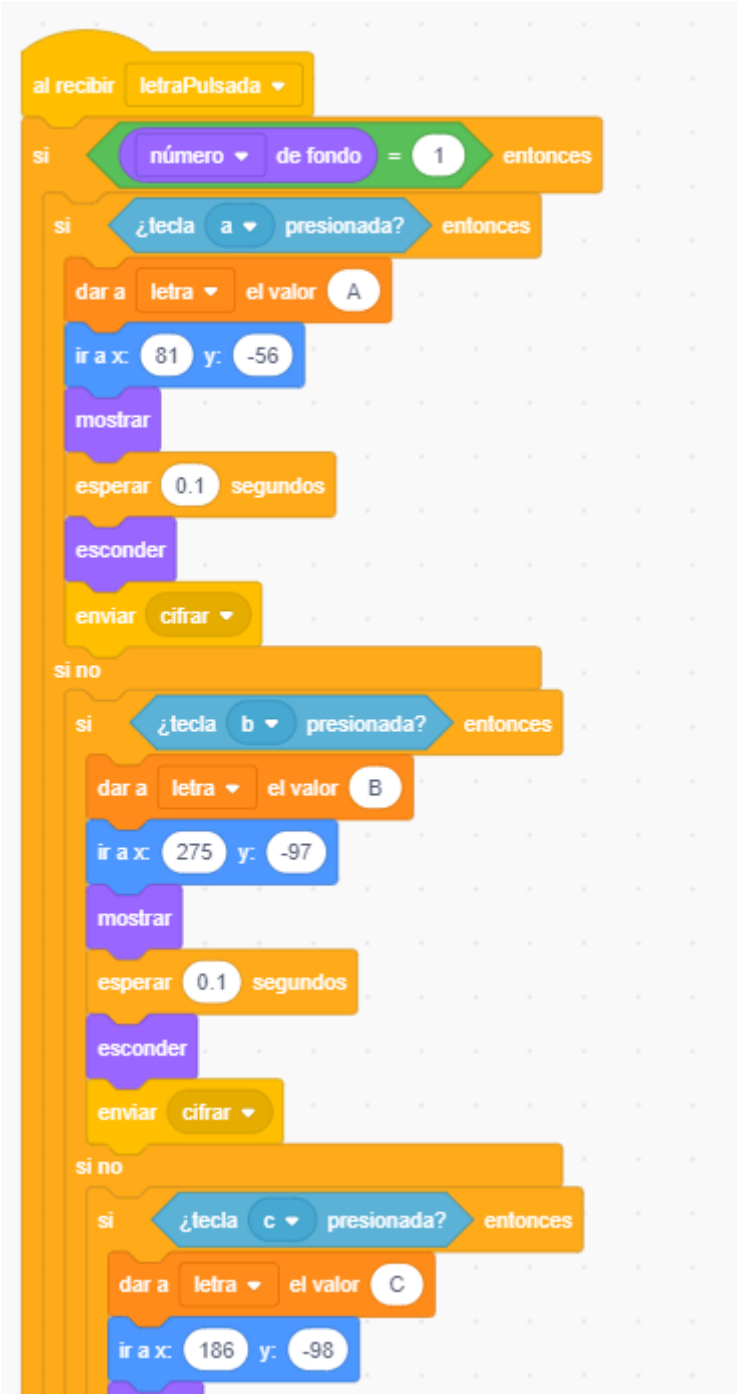

Figura 5.27: Código para simular el estado de pulsar una tecla. Dependiendo de la letra que sea, se asignará su valor a la variable 'letra', se mostrará el disfraz negro en su posición durante un breve período de tiempo y se enviará el mensaje 'cifrar' para que otro objeto encripte dicha letra.

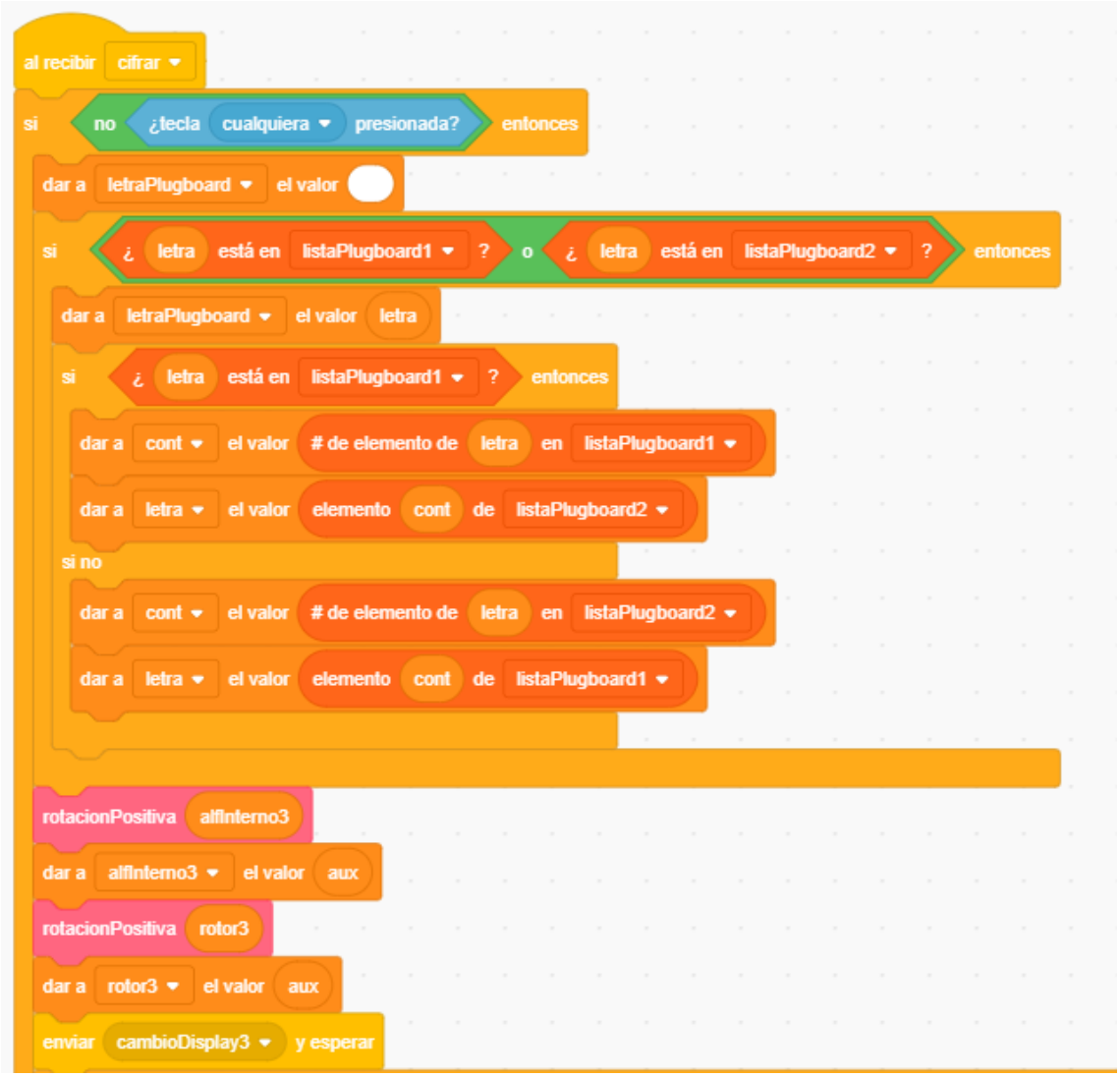

Figura 5.28: Primera parte del código de cifrado. Después de verificar que no hay una tecla pulsada para evitar concurrencia entre distintos encriptados, se comprueba si la letra a cifrar tiene alguna conexión en el clavijero. En caso afirmativo, se intercambia el valor de 'letra' por su par del clavijero y guardamos la original en la variable 'letraPlugboard'. Posteriormente, se gira el tercer rotor, su alfabeto y su display con el mensaje 'cambioDisplay3'.

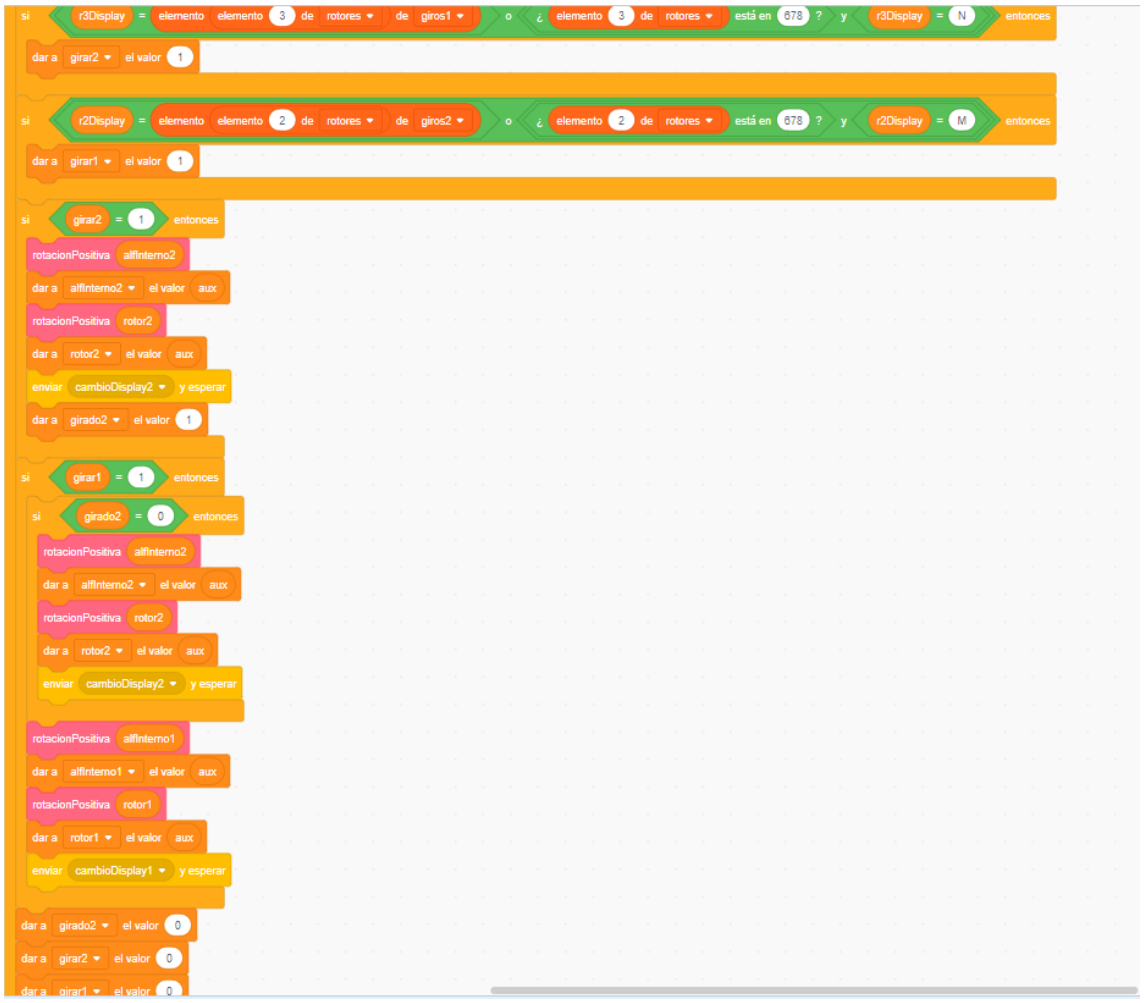

Figura 5.29: Segunda parte del código de cifrado. Aquí se implementan los puntos de rotación. Primeramente, se comprueba si el tercer y el segundo rotor están sus respectivos puntos de rotación. Como el rotor VI, VII y VIII tienen dos puntos de rotación, se añade una segunda condición opcional (con un 'OR' lógico) y la letra que lo representa, que en los tres casos es la 'N'. Luego, si los dos rotores están en su punto de rotación, se asigna el valor '1' a la variable 'girar2' y 'girar1'. Después, se hacen girar los correspondientes alfabetos, rotores y displays. La variable 'girado2' sirve como guarda para impedir que el segundo rotor gire dos veces en una misma pasada. Finalmente, se reinician 'girar1', 'girar2' y 'girado2'.

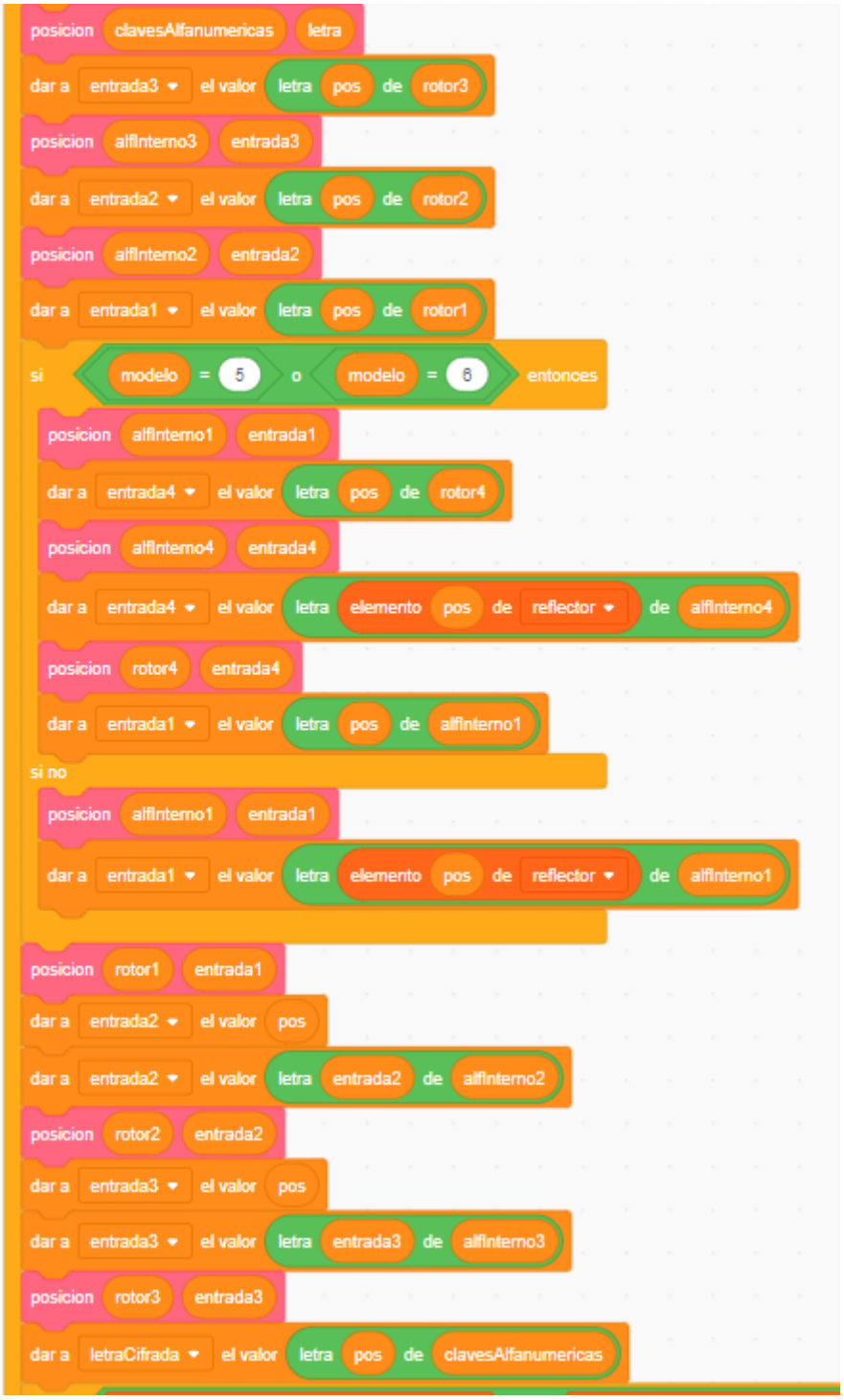

Figura 5.30: Tercera parte del código de cifrado. Esta corresponde al viaje que realiza el pulso electromecánico a través de los rotores, haciendo que la salida de un rotor sea la entrada del siguiente (véase las tablas 3.3, 3.4 y 3.5 para más información). Si el número de la variable 'modelo' es '5' o '6', estamos en una versión M4, por lo que el pulso tiene que pasar por el cuarto rotor. Finalmente, el reflector manda la señal de vuelta y el resultado final se guarda en la variable 'letraCifrada'.

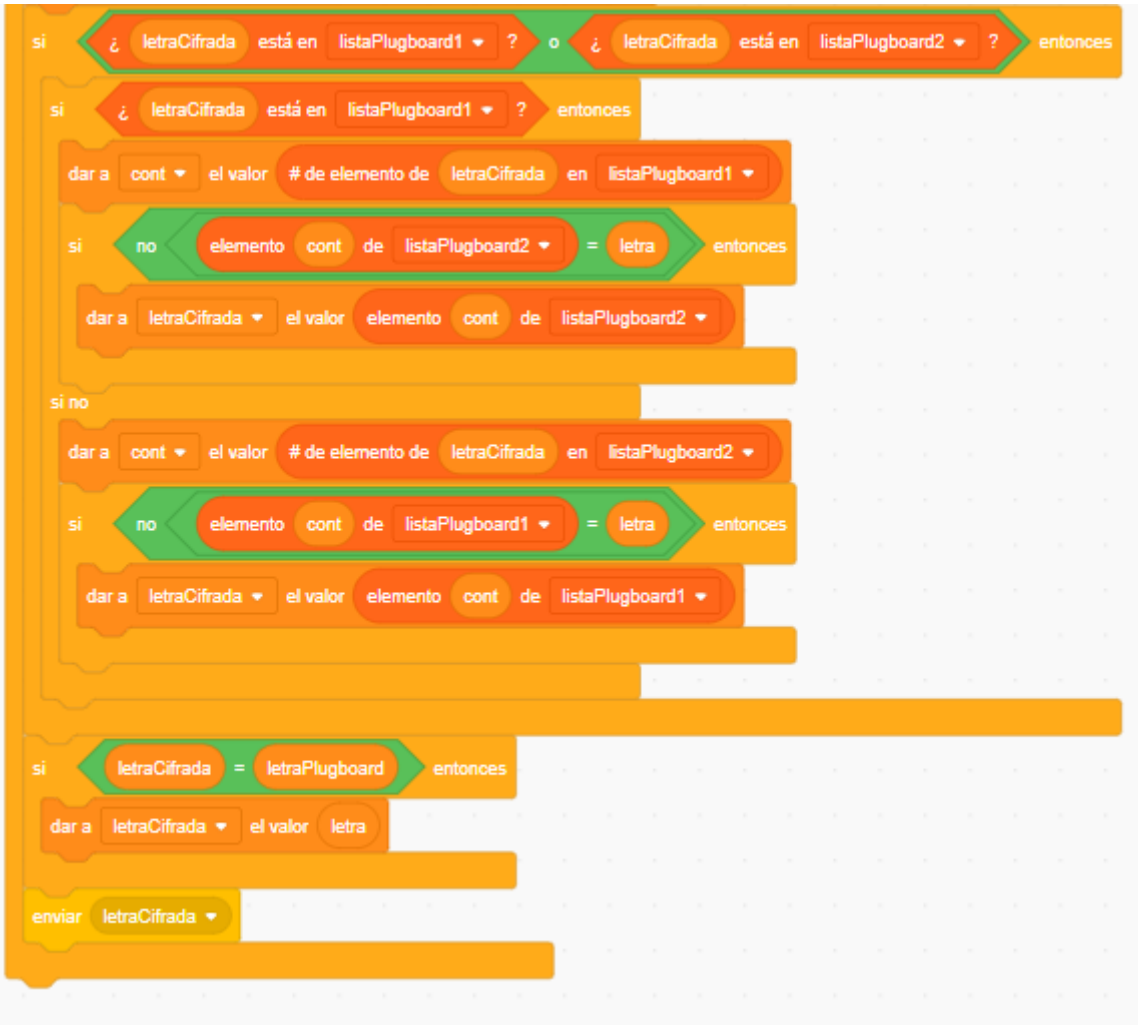

Figura 5.31: Cuarta y última parte del código de cifrado. Una vez que hemos cifrado la letra, tenemos que comprobar si la resultante está cableada en el clavijero. Para ello, verificamos si la letra cifrada está en 'listaPlugboard1' o 'listaPlugboard2'. En caso afirmativo, comprobamos que la letra a la que está conectada no es la misma que la original, y le damos ese nuevo valor a 'letraCifrada'. Esto último se debe a que puede darse el caso de que una letra se cifre por otra a la que está conectada, formando un bucle. Por ejemplo, si 'A' se cifra por 'B' y hay una conexión 'BA', se ignoraría el cableado, debido a que una letra no se puede encriptar por ella misma. Finalmente, se enviaría el mensaje 'letraCifrada' (no confundir con la variable homónima).

Figura 5.32: El evento 'clavesModificadas'. Si 'sentido' tiene valor '1', se mueve el alfabeto y el rotor del display seleccionado hacia delante (por ejemplo, si vale 'ABC...Z' pasa a valer 'BC...ZA'). Si tiene valor '0', gira en orden inverso (Si vale 'ABC...Z' pasa a valer 'ZABC...').

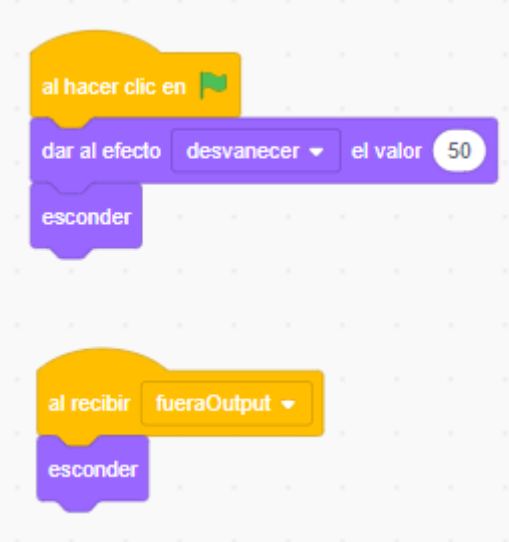

Figura 5.33: Dos grupos de bloques del objeto 'Outputs'. El primero, hace desvanecer el objeto de forma parcial y lo esconde hasta el cifrado. El evento 'fueraOutput' simplemente lo esconde, y se usa principalmente cuando se cambia de pantalla y el tablero de luces está encendido.

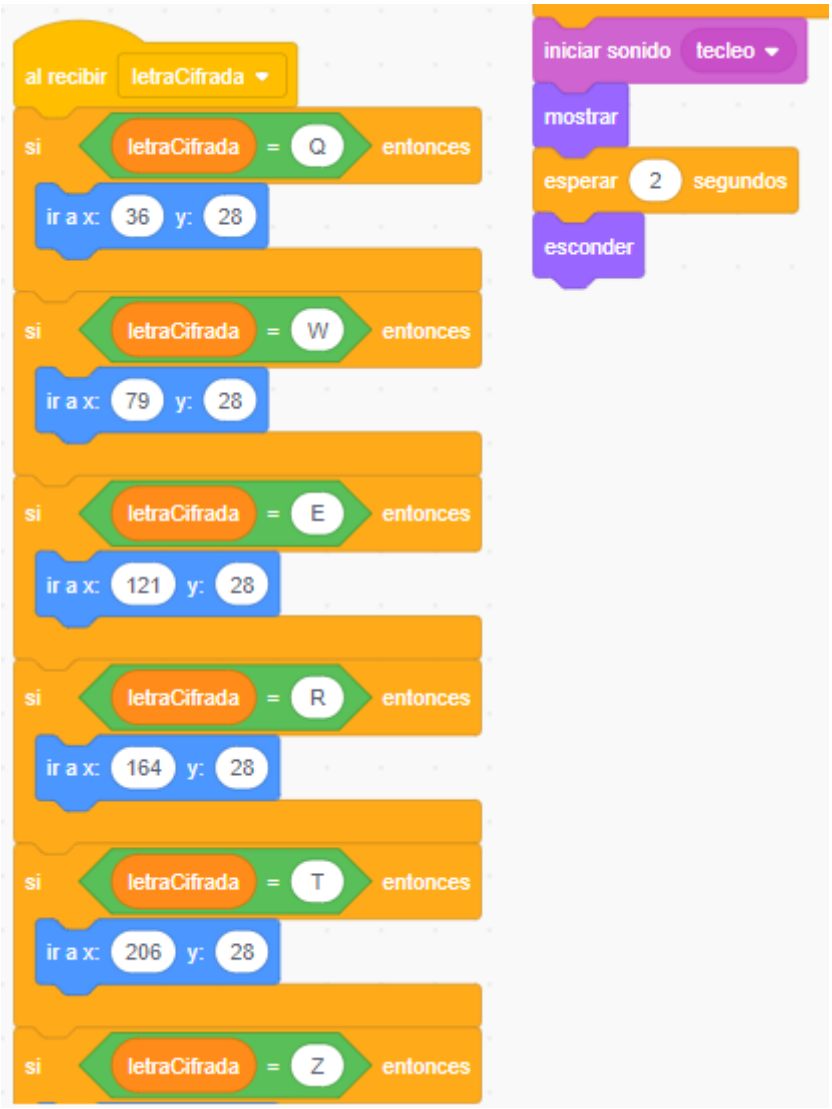

Figura 5.34: Bloques encargados de iluminar el panel de luces. Cuando llega el mensaje 'letraCifrada' desde 'Cifrado', el objeto 'Outputs' se desplaza hacia donde está esa letra, y se muestra durante dos segundos.

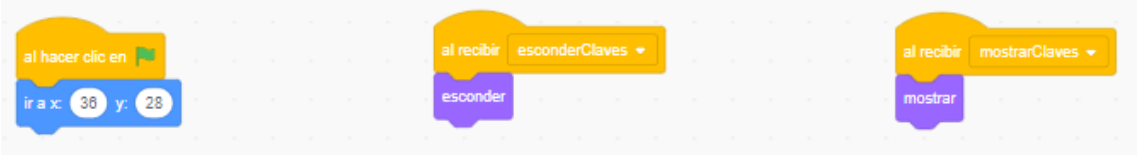

Figura 5.35: Primeros bloques del objeto 'Claves'. Al empezar la simulación, el selector se mueve al último componente de la clave por la derecha, ya que esta es su posición inicial predefinida. 'esconderClaves' oculta el selector y 'mostrarClaves' lo vuelve a enseñar.

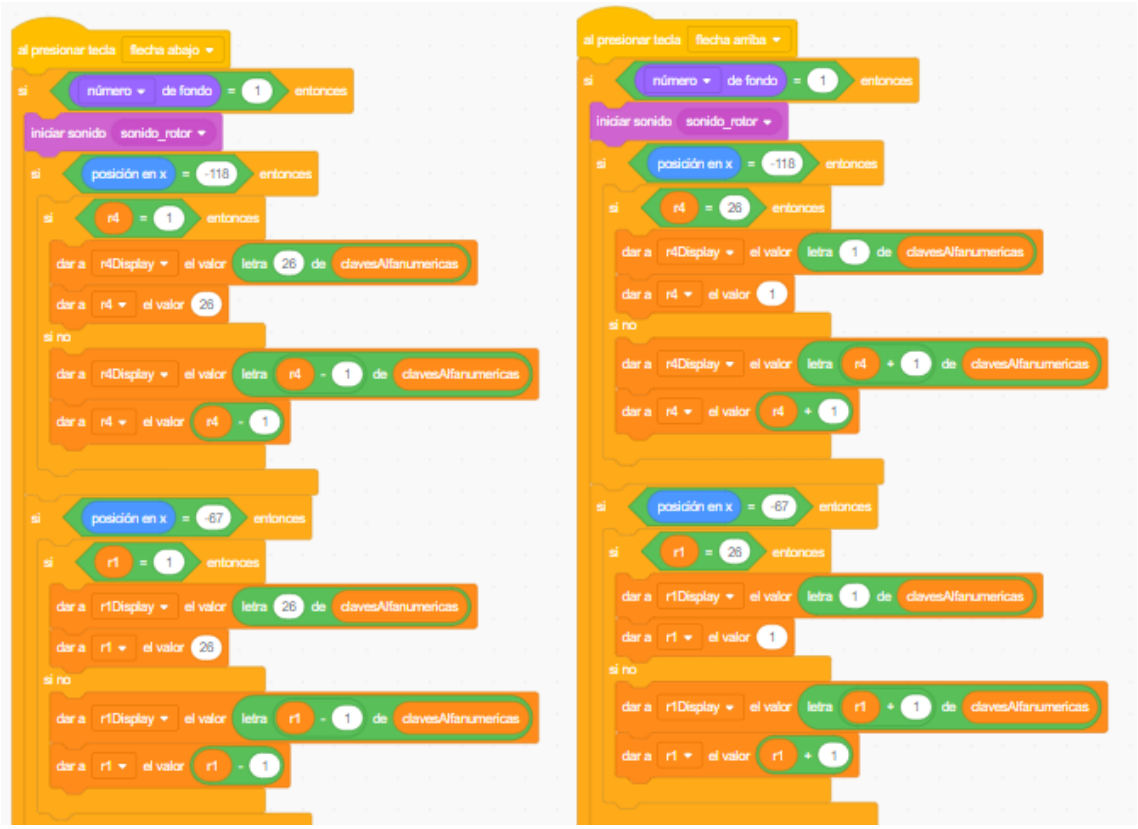

Figura 5.36: Bloques que implementan el cambio de un determinado display. Cuando se pulsa la flecha abajo, se comprueba la posición en el eje X para saber qué componente estamos modificando. Luego, si la variable 'r' vale '1', como vamos en orden inverso, pasará a valer '26' y el display valdrá la letra 'Z'; en cualquier otro caso, 'r' pasará a valer 'r' menos 1 y el display la letra anterior en orden alfabético. Aunque no se aprecia en la imagen, como esto último afecta a los alfabetos y rotores internos, damos valor '0' a 'sentido' y enviamos el mensaje 'claveModficadas' para que el objeto 'Cifrado' los modifique como es debido. De igual forma sucede al pulsar la flecha arriba, solo que 'r' y la variable del display se modifica en orden alfabético hacia delante y el valor que se la da a 'sentido' es '1'.

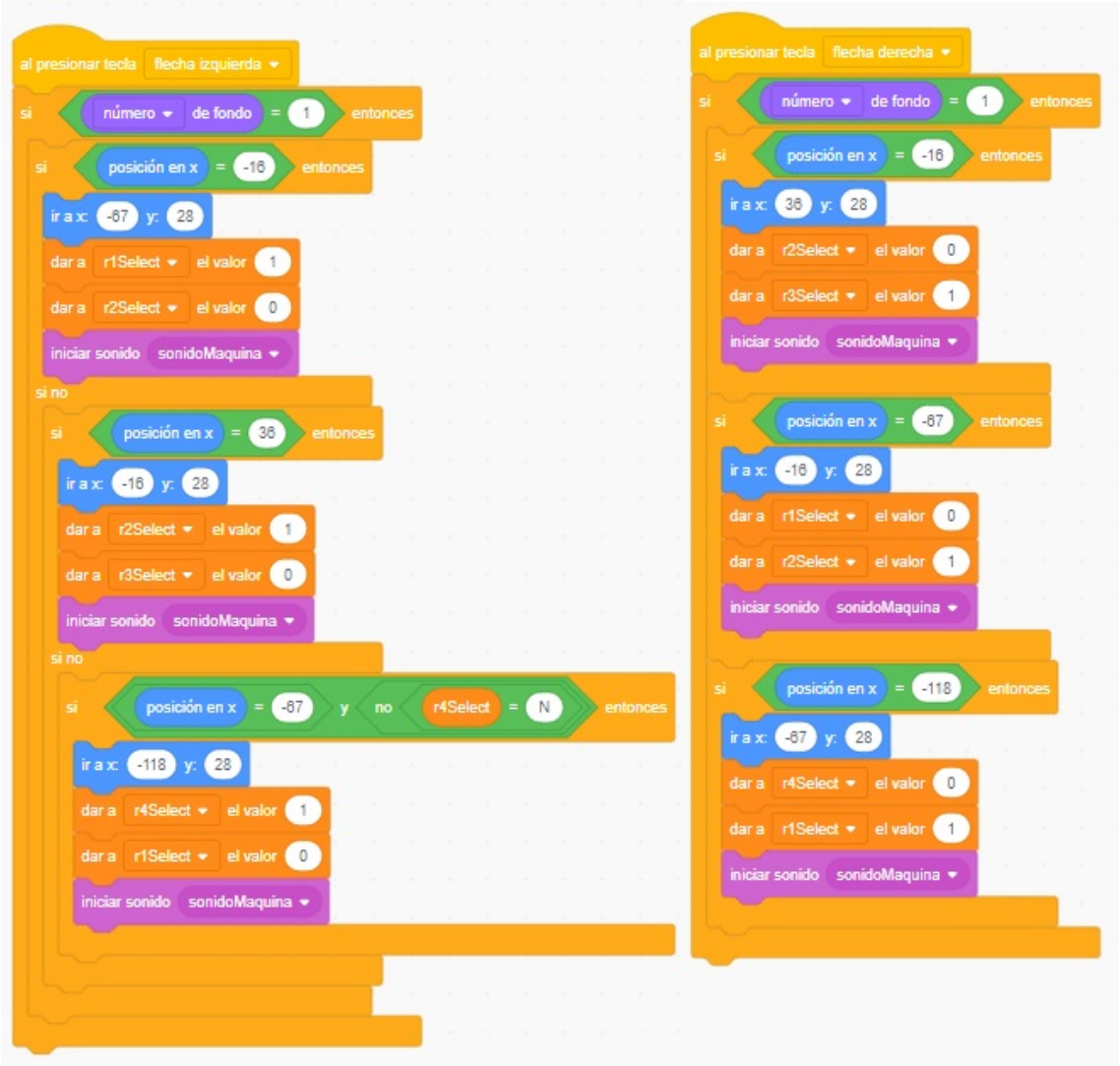

Figura 5.37: Bloques correspondientes al movimiento del selector de la clave. Cuando nos desplazamos, hay que cambiar el valor de la variable 'Select' para saber qué rotor hemos dejado y a cuál hemos pasado. Para saber si estamos en un modelo de tres rotores, 'r4Select' valdrá 'N'. En cualquier otro caso, el modelo será de cuatro rotores.

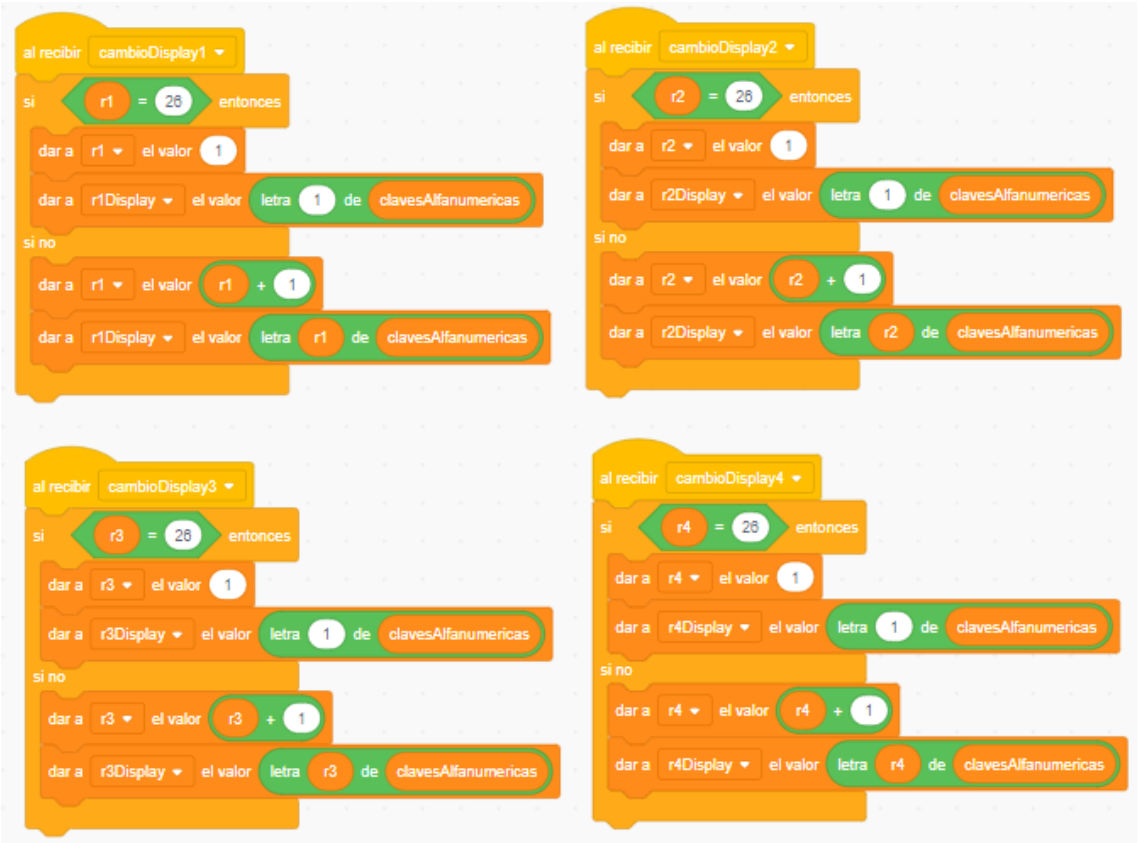

Figura 5.38: Eventos que modifican los cuatro displays cuando se está encriptando una letra.

Figura 5.39: Evento que reinicia todos los displays, bien cuando hemos seleccionado nuevos rotores, o bien cuando hemos cambiado de orden los que ya teníamos.

Figura 5.40: Código de inicio de los rotores. A la izquierda, el de los rotores I, II y III, que como están seleccionados por defecto, han de tener un tamaño de 150 y estar ordenados debidamente. A la derecha, la posición inicial del resto de rotores, que han de tener un tamaño de 100 y estar en sus respectivas posiciones para poder ser elegidos.

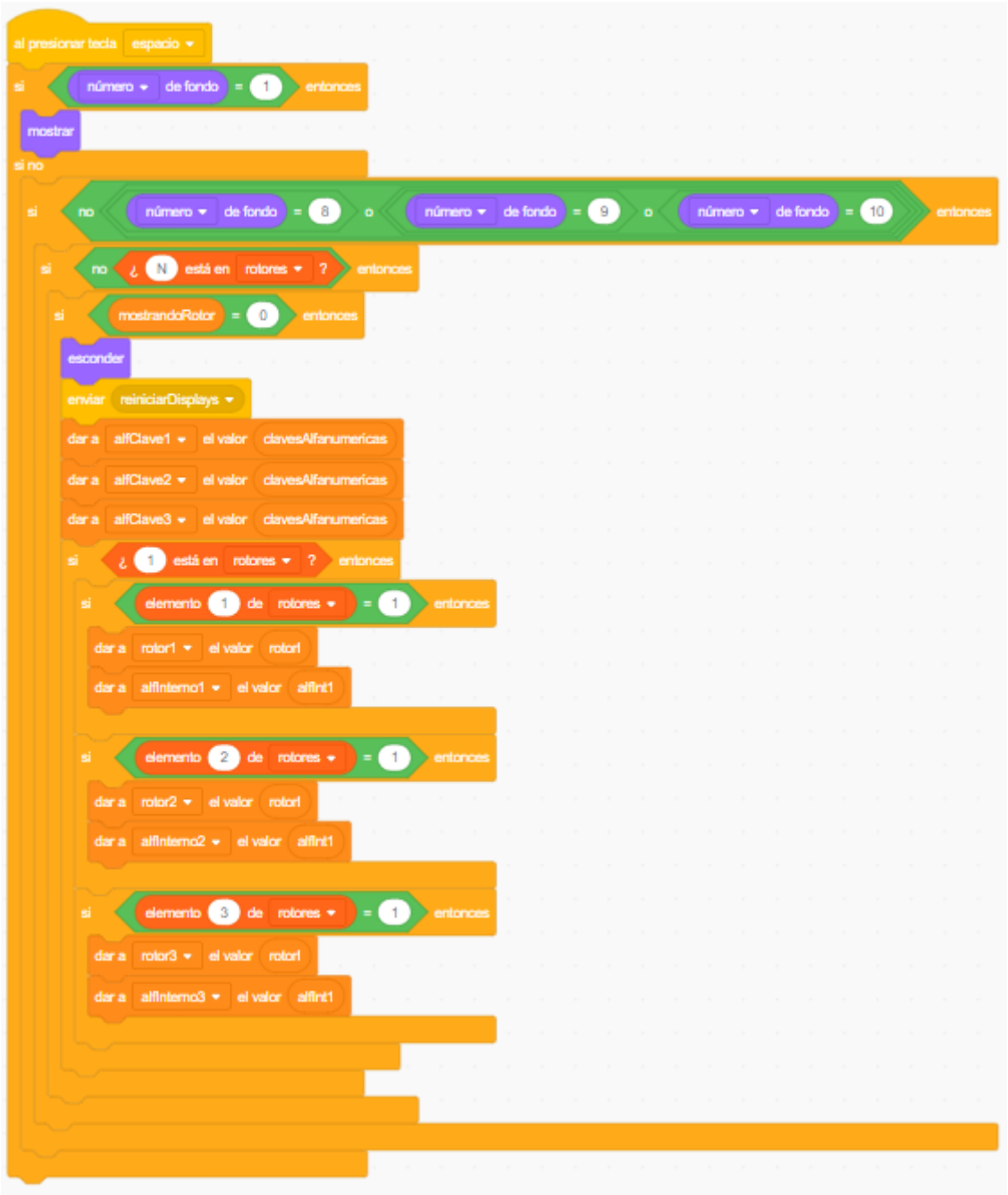

Figura 5.41: Bloques de los rotores I, II, III, IV y V cuando se transita entre la pantalla principal y la de rotores. Si pulsamos espacio y estamos en la principal, cambiamos de pantalla, por lo que mostramos el rotor. Si no es así y estamos en una de las pantallas de rotores (los fondos 8, 9 y 10 son el clavijero, la pantalla de presentación y las instrucciones, respectivamente), entonces comprobamos que no nos hemos dejado ningún rotor sin colocar, que no estamos modificando el *Ringstellung* de ningún rotor y, posteriormente, preparamos los alfabetos y rotores para el cifrado.

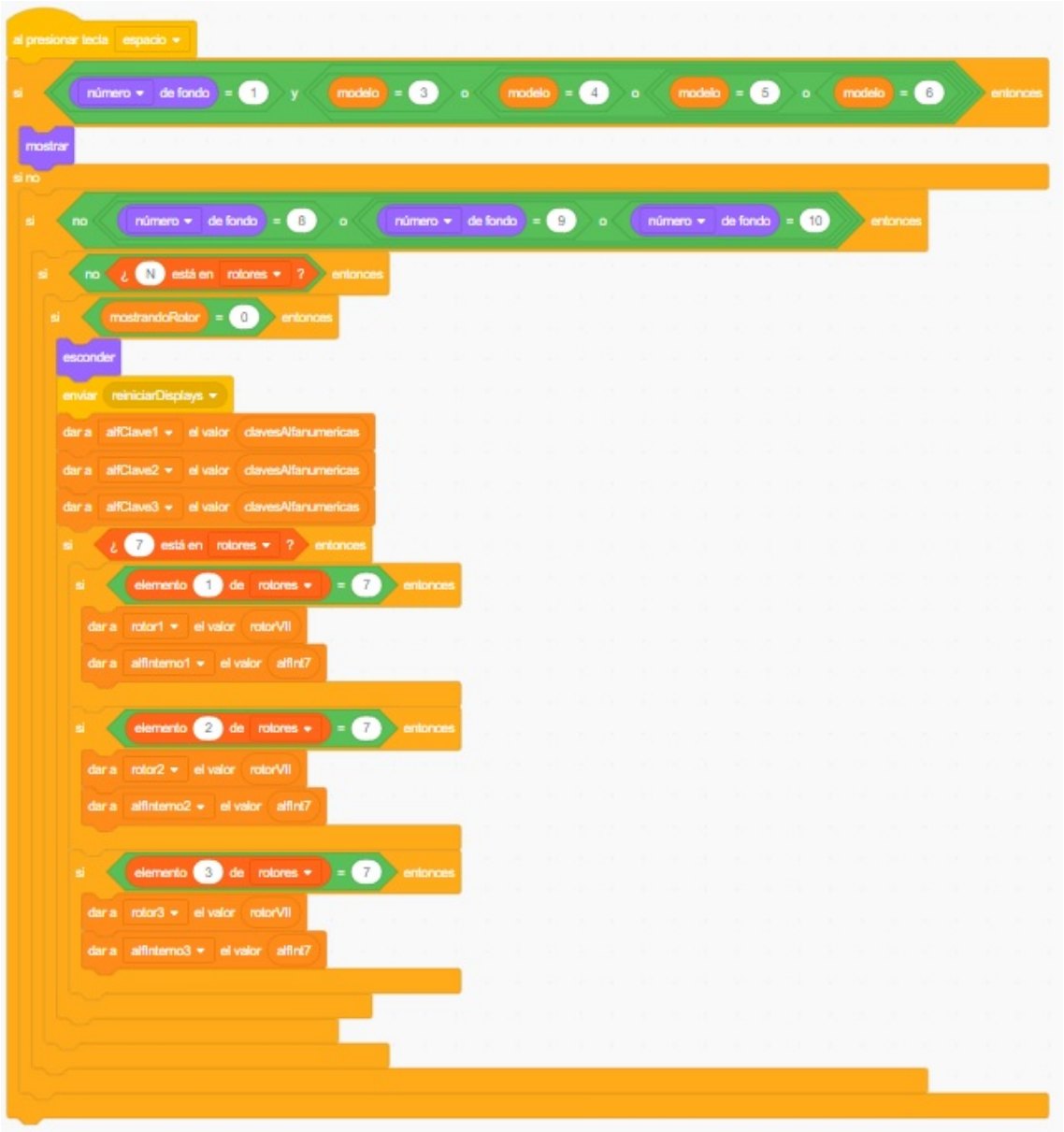

Figura 5.42: Bloques de los rotores VI, VII y VIII cuando se transita entre la pantalla principal y la de rotores. Dichos rotores se han de poder ver cuando, al pulsar espacio, estemos en el fondo principal y el modelo sea de una Kriegsmarine ('modelo' tenga un valor del 3 al 6). Si queremos ir a la pantalla principal, entonces nos aseguramos de que no estemos en otro fondo que no sea el de rotores. Luego, si no falta ningún rotor por colocar y nos estamos modificando internamente ninguno, asignamos a los alfabetos y los rotores su valor correspondiente.

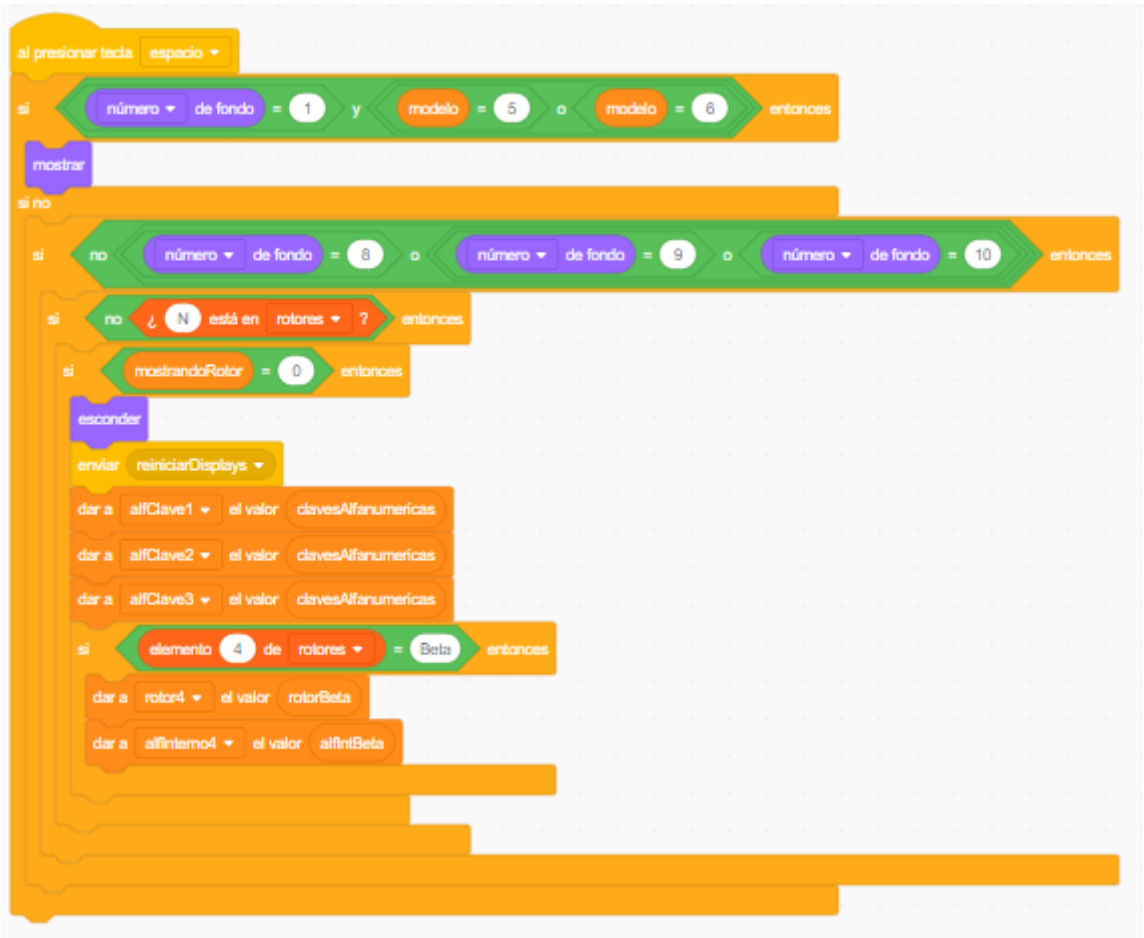

Figura 5.43: Bloques de los rotores Beta y Gamma cuando se transita entre la pantalla principal y la de rotores. Es muy similar al de la figura 5.41 y 5.42, pero aquí solo se tienen en cuenta los modelos 5 y 6 (correspondientes a la M4) cuando cambiamos de pantalla.

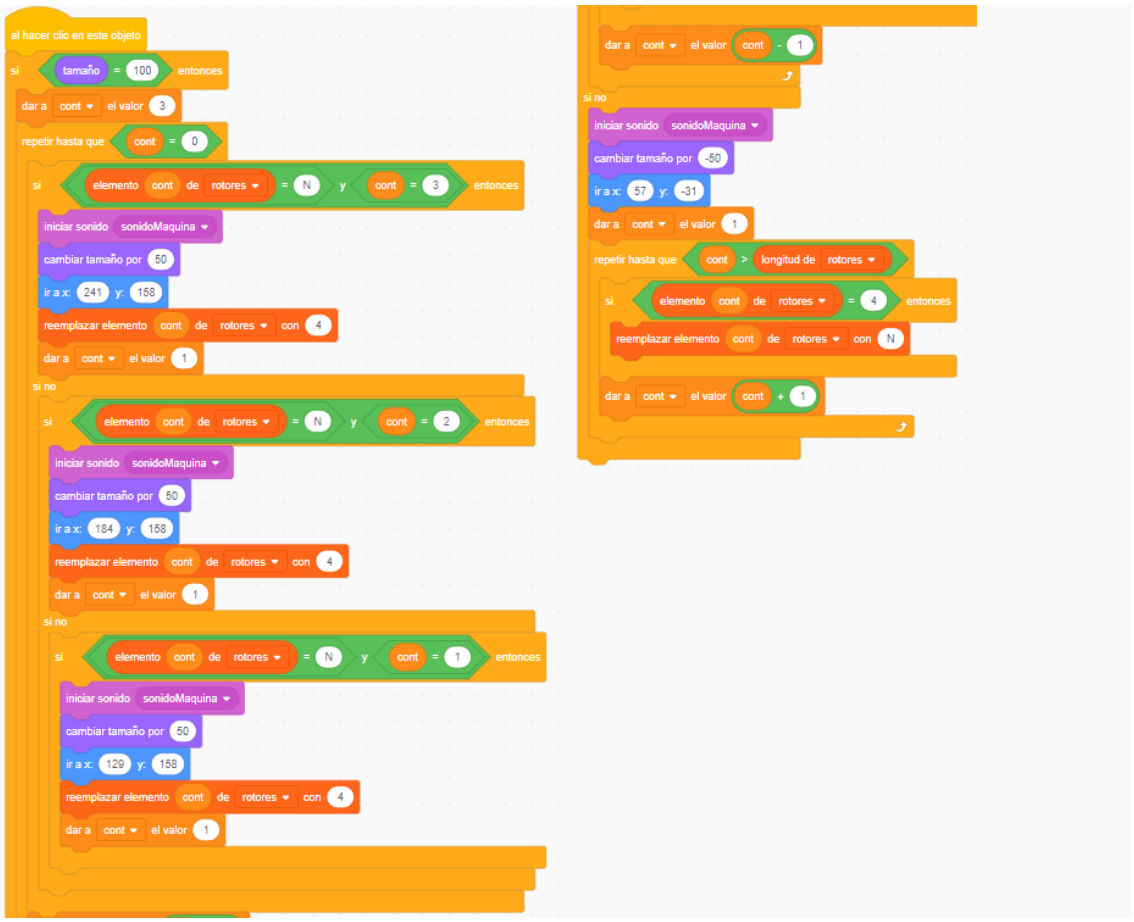

Figura 5.44: Grupo de bloques encargado de colocar un rotor para su uso. Este es el específico para los rotores del I al VIII. Si el tamaño del rotor es 100, quiere decir que no se está usando, por lo que vamos comprobando, de derecha a izquierda, si hay un hueco para colocarlo; además de adjuntarlo a la lista 'rotores'. Si el tamaño no es 100, quiere decir que se está usando, así que lo dejamos en su sitio y lo borramos de la lista 'rotores'.

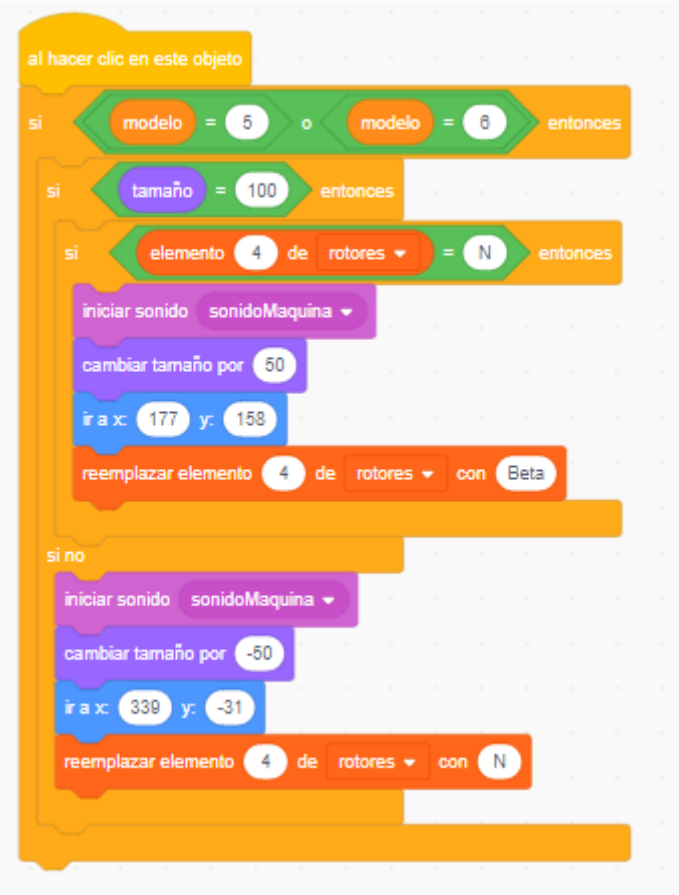

Figura 5.45: Grupo de bloques encargado de colocar un rotor para su uso. Este es el específico para los rotores Beta y Gamma. Si pulsamos en el rotor, el modelo ha de ser de una máquina M4 (5 o 6). Una vez verificado esto, si el tamaño es 100 y no hay ningún rotor ocupando la cuarta posición, se coloca en dicho lugar y se añade el nombre del rotor a la lista 'rotores'. Si el tamaño no es 100, entonces debemos devolver el rotor a su posición inicial y eliminarlo de la lista 'rotores'.

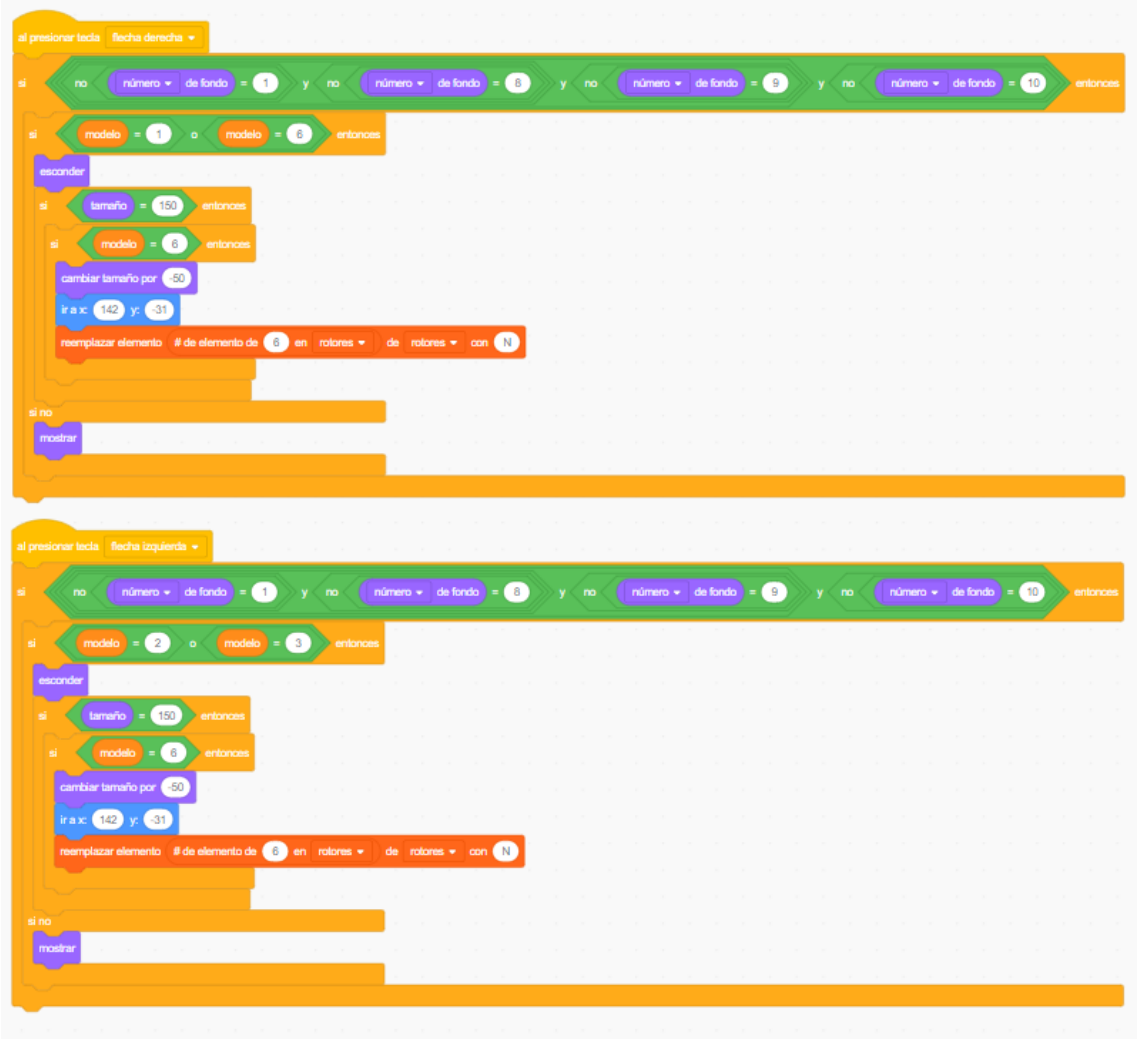

Figura 5.46: Conjunto de instrucciones que representan el protocolo que deben seguir los rotores VI, VII y VIII cuando se cambia de modelo de Enigma en la pantalla de rotores. Si vamos hacia delante pulsando la flecha derecha, cuando pasemos del modelo 6 al 1 y del 1 al 2 han de desaparecer (ya que no estan disponibles en la versión Wehrmacht). Además, si estamos usando alguno en ese instante, le cambiamos el tamaño, lo movemos a su posición inicial y lo borramos de la lista 'rotores'. Si transitamos entre otros modelos, simplemente mostramos los rotores. Si vamos hacia atrás con la flecha izquierda, hay que hacer el proceso inverso, teniendo en cuenta que ahora las transiciones son del modelo 3 al 2 y del 2 al 1.

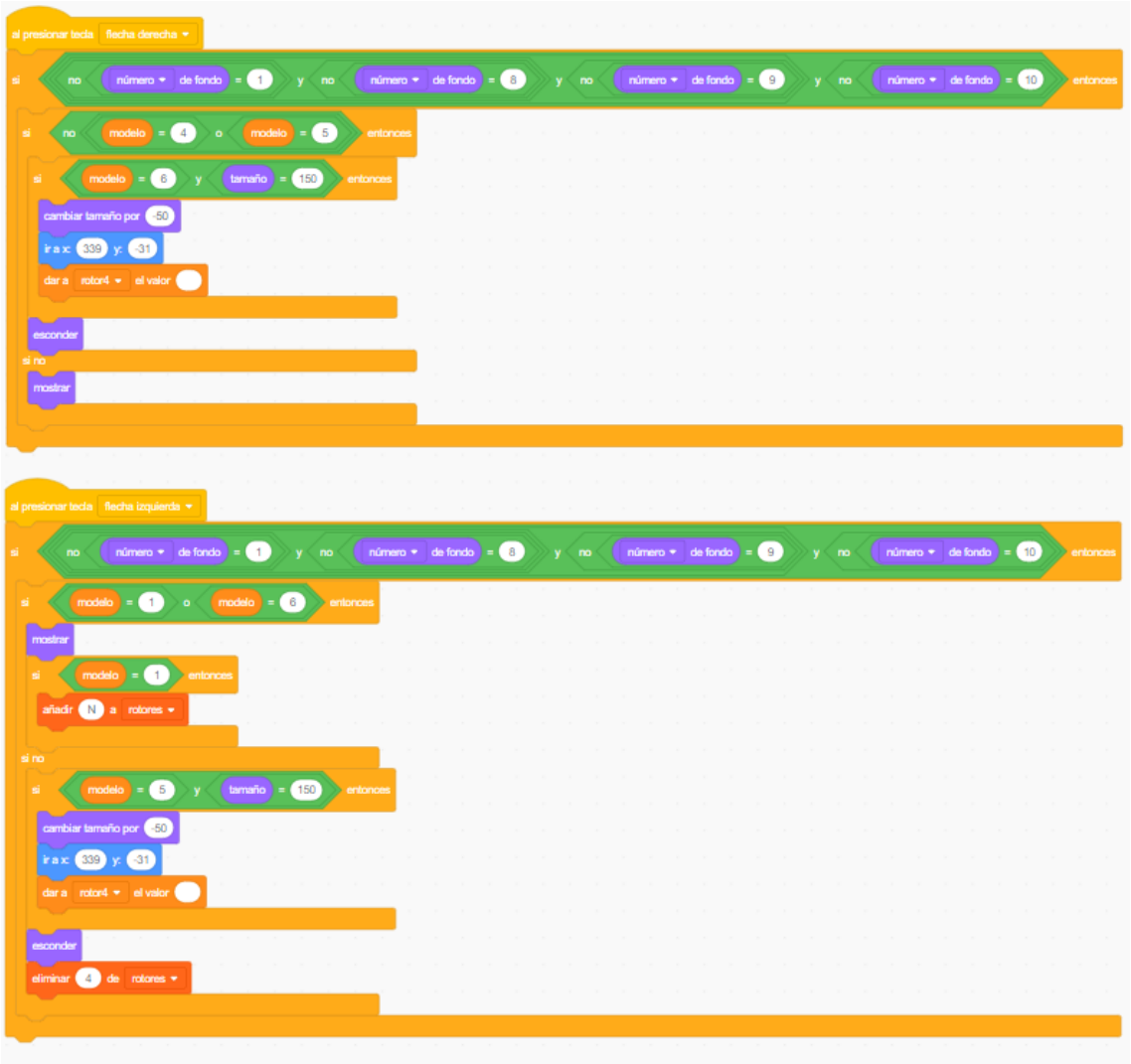

Figura 5.47: Conjunto de instrucciones que representan el protocolo que deben seguir los rotores Beta y Gamma cuando se cambia de modelo de Enigma en la pantalla de rotores. Si vamos hacia delante con la flecha derecha, no transitamos del modelo 4 al 5 ni del 5 al 6, pero sí del 6 al 1 (pasamos de la versión M4 a la Wehrmacht) y estamos usando el rotor, entonces lo devolvemos a su posición inicial y lo ocultamos. El caso opuesto a esta sentencia es que haya una transición del modelo 4 al 5 y del 5 al 6. En ambos casos es pasar a un modelo M4, por lo que se muestran los rotores. Si vamos hacia atrás con la flecha izquierda y transitamos del modelo 1 al 6 o del 6 al 5, entonces mostramos el rotor. Si se produce la primera transición, además de mostrarlo, añadimos un cuarto elemento a la lista 'rotores', indicando que ya podemos usar cuatro rotores. Si, cuando pulsamos la flecha izquierda, hacemos otras transiciones, escondemos el rotor y eliminamos el cuarto elemento de 'rotores'. Además, si la transición es del modelo 5 al 4 y estamos usando el rotor, lo devolvemos a su posición inicial.

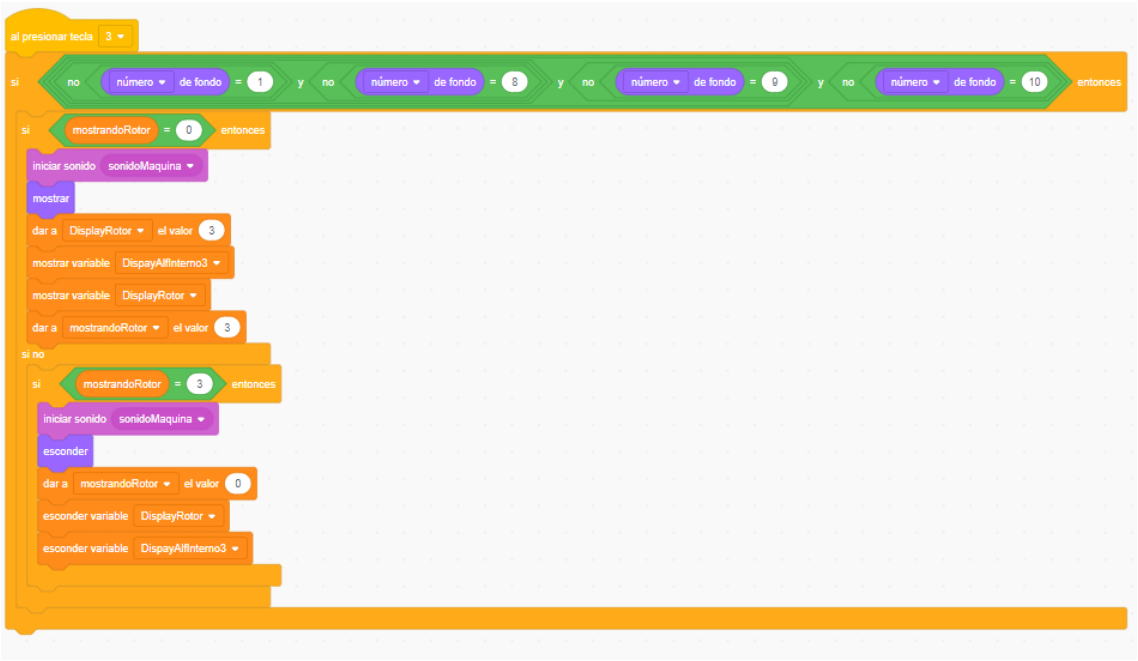

Figura 5.48: Ejemplo de selección de un rotor para modificar su configuración interna. Cuando pulsamos el número del rotor que queremos cambiar, siempre que nos encontremos en una pantalla de rotores, comprobamos la variable 'mostrandoRotor'. Si está a '0', quiere decir que no estamos modificando ningún rotor, por lo que asignamos el número identificador del rotor a 'mostrandoRotor' y a 'DisplayRotor', mostrando en pantalla este último, junto con el disfraz y 'DisplayAlfInterno', que representa la configuración actual (Por ejemplo, si estamos en A-01, 'DisplayAlfInterno' tiene valor 'A'). Si 'mostrandoRotor' tiene por valor el número del rotor que hemos pulsado, entonces ocultará el disfraz y dará por hecho que hemos acabado de modificar el cableado interno. No podremos modificar otro rotor hasta que no terminemos con el actual.

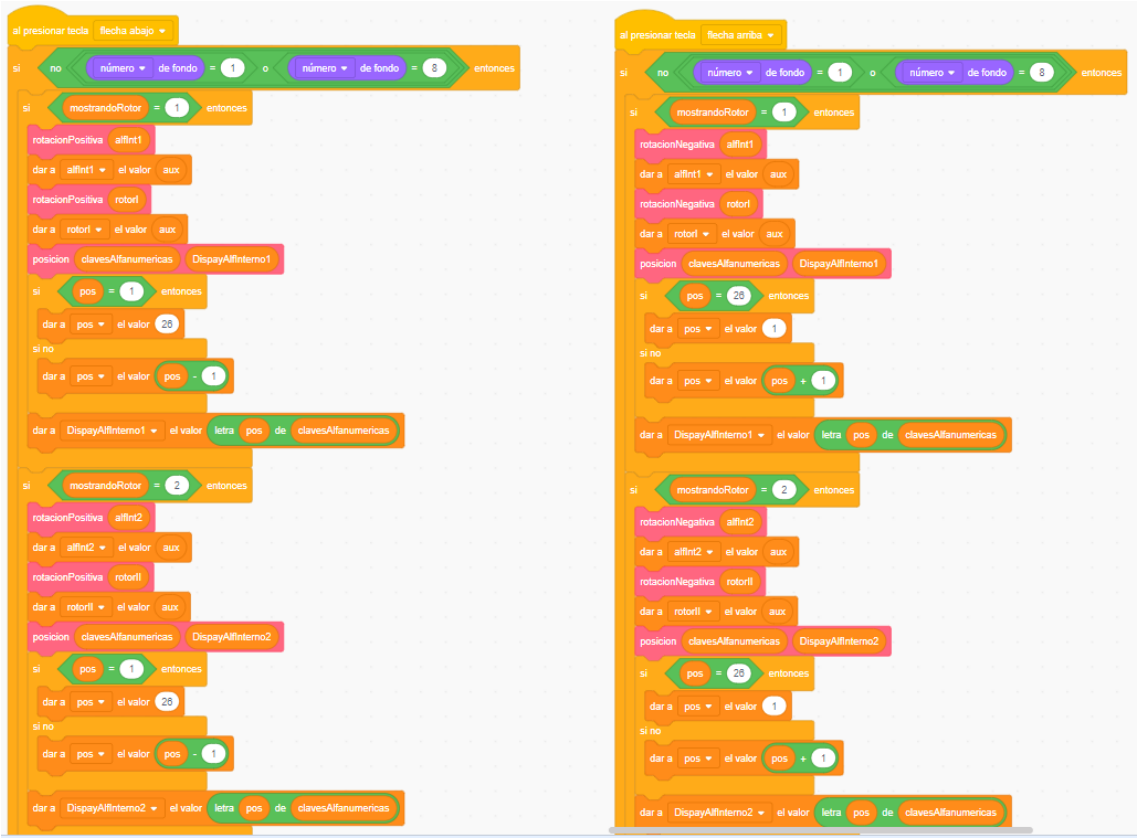

Figura 5.49: Bloques dedicados a modificar el cableado interno de los rotores. Como se puede apreciar, solo se modificará el rotor y el alfabeto interno de aquél cuyo número de identificación sea el valor de 'mostrandoRotor'.

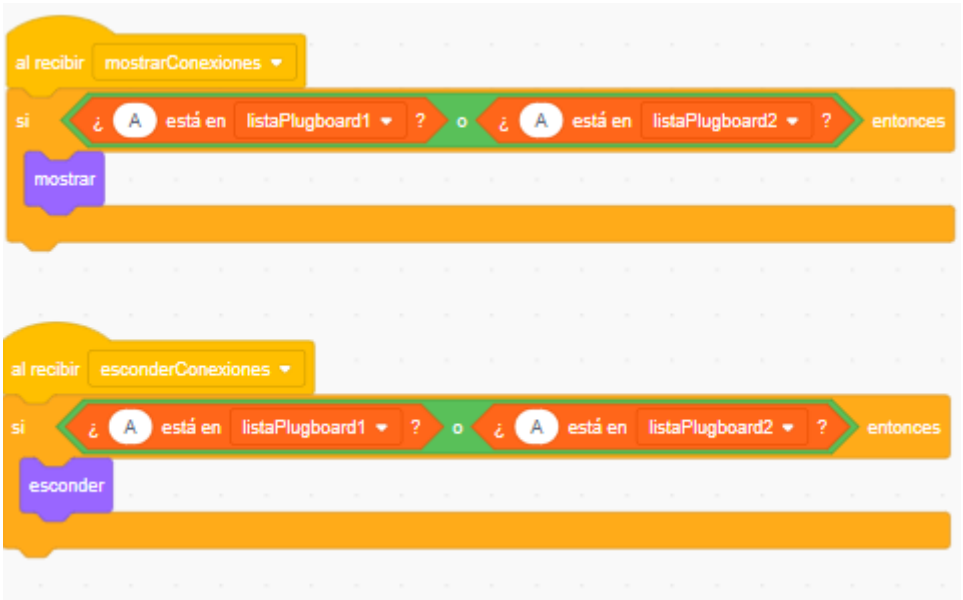

Figura 5.50: Eventos para mostrar y esconder las conexiones del clavijero cuando se cambia de pantalla. Basta con verificar si esa letra está en 'listaPlugboard1' o 'listaPlugboard2' para saber si tiene alguna conexión.

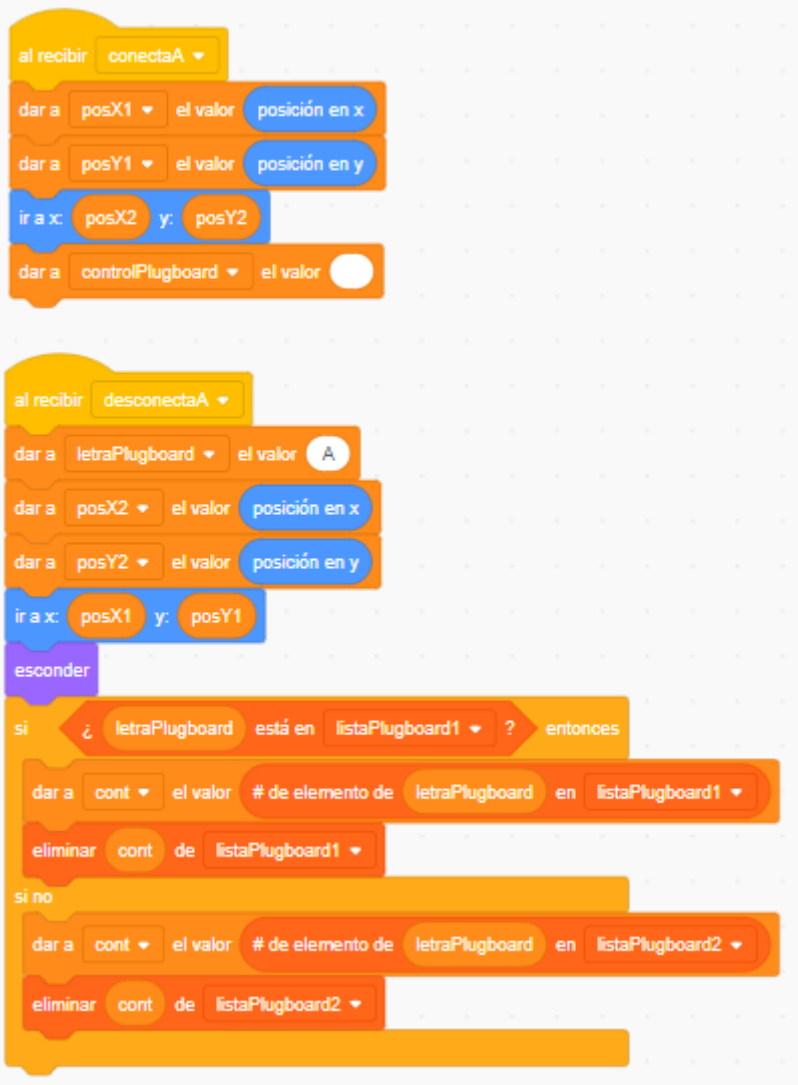

Figura 5.51: Eventos para cablear o desconectar una letra. Para lo primero, hay que tener en cuenta que el objeto que nos ha enviado el mensaje ha dejado sus coordenadas en posX2 y posY2, por lo que nuestro objeto dejará las suyas en posX1 y posY1 y se moverá a la posición del otro. Finalmente, asignará a 'controlPlugboard la cadena vacía para indicar que se ha completado la conexión por parte del presente objeto. Para el segundo evento, nuestro objeto tiene una conexión, de modo que guardará su letra en 'letraPlugboard' y su posición actual en 'posX2' y 'posY2'. Luego se moverá a 'posX1' y 'posY1', su posición original, y se esconderá. Por último, hay que borrarla de la 'listaPlugboard' en la que se encuentre.

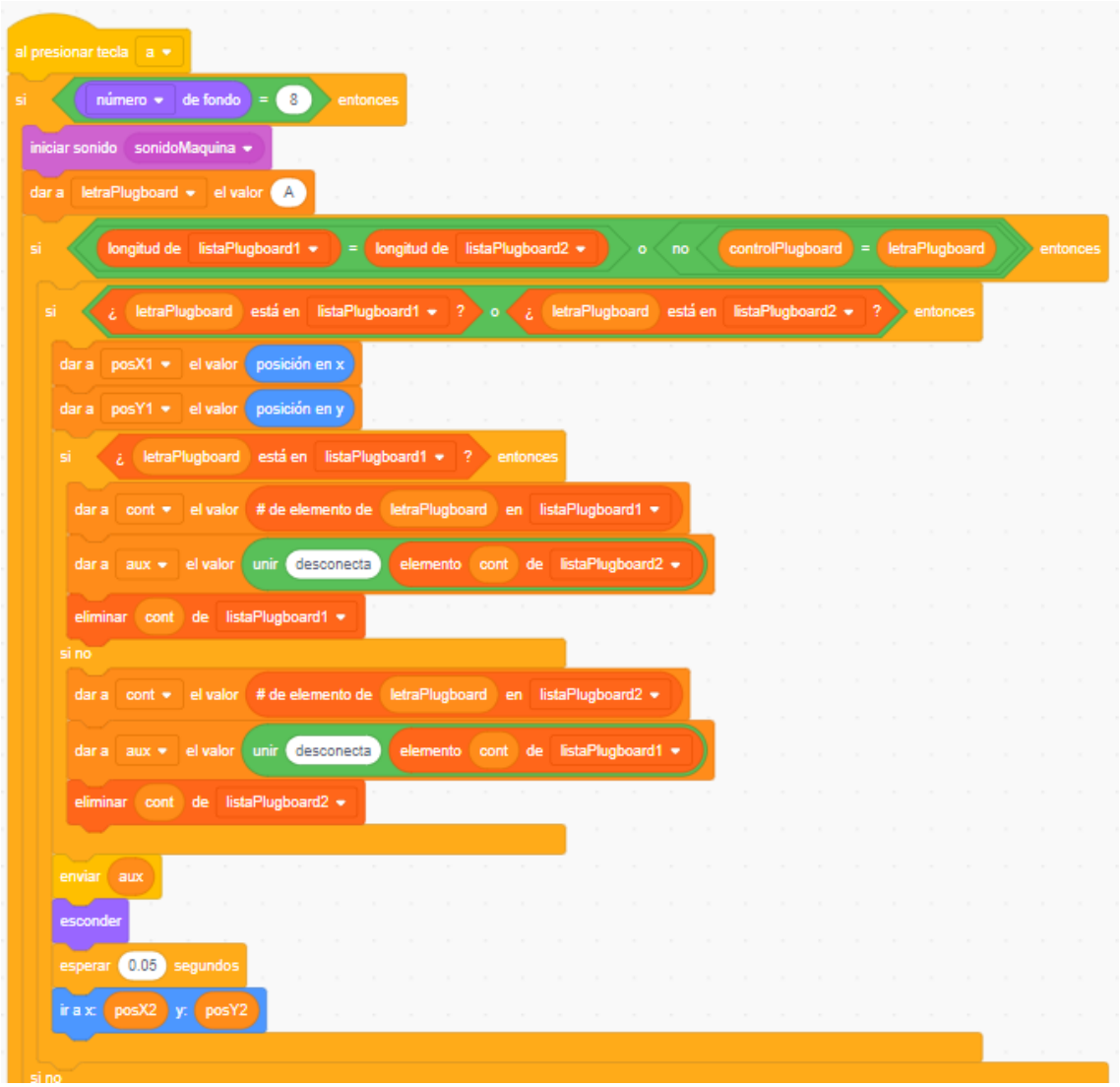

Figura 5.52: Primera parte del código empleado en las conexiones de una determinada letra en el clavijero. Aquí se implementa el caso en que la letra pulsada esté cableada y queramos desconectarla. Para ello se verifica que las listas 'listaPlugboard' sean del mismo tamaño, para que no haya ninguna conexión incompleta; o que 'controlPlugboard' y 'letraPlugboard' tengan distintos valores, lo que indica que no hemos pulsado esa letra previamente. Después, se guarda su posición en 'posX1' y 'posY1' (que se enviará su par con el mensaje 'desconecta', ya que es la posición inicial de este), se elimina de 'listaPlugboard', vuelve a su posición inicial y se esconde.

Figura 5.53: Segunda parte del código empleado en las conexiones de una determinada letra en el clavijero. Aquí se implementan tres casos distintos. En primer lugar, si la letra pulsada no tiene ninguna conexión y no hay conexiones incompletas, 'controlPlugboard' tendrá como valor la cadena vacía. Entonces se dará a 'controlPlugboard' el valor de la letra pulsada para indicar que ahora sí que hay una conexión inacabada, se añade la letra a 'listaPlugboard1', se muestra en pantalla el objeto y se guardan sus posiciones en 'posX1' y 'posY1'. En segundo lugar, si 'controlPlugboard' no tiene el mismo valor que 'letraPlugboard', es porque hay que realizar una conexión entre la letra que hemos pulsado y otra que ya está en el clavijero. Para ello, se reinicia 'controlPlugboard', se añade la letra pulsada a 'listaPlugboard2', se guarda su posición en 'posX2' y 'posY2', se mueve a la posición 'posX1' y 'posY1' y se envía el mensaje 'conecta' a la otra letra. Finalmente, si 'controlPlugboard' y 'letraPlugboard' tienen el mismo valor, quiere decir que queremos quitar del clavijero una letra que no está cableada. Se procede eliminando esa letra de 'listaPlugboard1', reiniciando 'controlPlugboard' y escondiendo el objeto.

CAPÍTULO 6
Conclusiones

En este último capítulo de la memoria, se plantean las consideraciones finales que hemos obtenido a lo largo del proyecto realizado. Se mostrará un resumen y se realizarán unas comprobaciones finales sobre nuestro simulador, comparándolo con otro ya existente. Por último, se analizarán los objetivos que hemos alcanzado en el presente trabajo.

6.1 Observaciones finales

En el trabajo actual se ha comprobado que con un lenguaje de programación orientado a un gran número de usuarios como Scratch, es posible la creación de una máquina clásica de cifrados de una cierta complejidad con relativa facilidad, a pesar de que en su época de apogeo supuso una revolución a la hora de encriptar mensajes para un bando de un importante conflicto mundial. Para la creación de esta máquina se ha realizado un trabajo de investigación empleando varios sitios web, como se puede apreciar en bibliografía del final de este documento.

Tal como se explicó en el capítulo 4 dedicado al lenguaje de programación Scratch, este introduce a las personas inexpertas en la programación dentro de este mundo, con independencia de la edad que tengan. Esto se debe en gran medida a que Scratch es un lenguaje intuitivo inspirado en los bloques de juguete de LEGO, por lo que el usuario solo ha de arrastrar y organizar debidamente dichos bloques para producir un programa funcional, sin necesidad de escribir líneas de código. Este programa consta de diversos objetos dentro del entorno de programación donde se encuentran los conjuntos de bloques creados. Estos objetos se comunican entre sí, de esta forma una vez finalizamos el proyecto tendríamos el simulador completamente operativo.

No deja de ser cierto que Scratch está limitado a la hora de crear ciertos tipos de programas o métodos. En nuestro caso, a la hora de crear las letras internas de los rotores y los reflectores hemos tenido que utilizar variables y listas, cuando podría haber sido mucho más útil y práctico el uso de diccionarios[1], que Scratch no tiene en su haber.

Por eso, lo primero que hemos hecho ha sido contar la historia de la máquina Enigma para que el lector pueda llegar a entender la importancia que tuvo durante la Segunda Guerra Mundial y la revolución que supuso para el mundo de la criptografía. Después se ha explicado el funcionamiento de este mecanismo, cómo lo usaban las Potencias del Eje (sobre todo Alemania) para establecer sus redes de comunicación sin que los Aliados pudieran enterarse y la seguridad desde el punto de vista matemático que ofrecía

[1] Herramienta presente en el lenguaje de programación *Python*, que consiste en una colección no ordenada de valores que son accedidos a traves de una clave. Es decir, en lugar de acceder a la información mediante el índice numérico es posible acceder a los valores a través de sus claves, que pueden ser de diversos tipos.

la máquina. Posteriormente, se ha comentado por qué hemos escogido Scratch como entorno de programación para nuestro simulador y qué herramientas nos ofrece para la elaboración de las distintas funcionalidades de la máquina.

Una vez que se ha presentado Scratch, se ha detallado cómo hemos implementado cada uno de los objetos del programa para darle la funcionalidad que poseen las seis versiones de la máquina Enigma que se han tenido en cuenta. Para ello, se ha explicado uno a uno estos objetos junto con sus respectivos disfraces y partes de código.

6.2 Comprobaciones

Realizar la verificación de nuestro simulador ha consistido en cifrar un texto de 90 palabras con el emulador de la máquina Enigma disponible en la página web de *Cypher machines and cryptology*[2]. Para ello, se ha establecido una configuración aleatoria de la máquina, que es:

- Modelo: Kriegsmarine M3 - UKW = C
- Orden de rotores: VII, VI, IV
- Ajuste de anillo: 06, 22, 14
- Clave del mensaje: AAA
- Clavijero: PO, ML, IU, KJ, NH, YT, GB, VF, RE, DC
- Texto: 'Tony, me alegro de que hayas vuelto al complejo. No me gusta la idea de que te pasees por la mansion a solas. Todos necesitamos una familia. Mi fe se basa en la gente, imagino. En los individuos. Y me alegra decir que, en su mayoria, no me han decepcionado. Por eso yo tampoco puedo decepcionarlos. Crei que no contandote lo de tus padres te evitaba ese sufrimiento, pero ahora me doy cuenta de que era a mi a quien se lo evitaba. Te pido perdon.'

El mensaje cifrado resultante, con todas las letras agrupadas, es:

'NWEDFNPSSHHDZXJCFPLLYXTJMOMYPVB
LHFRHVAPSAZHQRPQOFPBDDBGSWKGFLSZ
CYLCDJPPGSPBWAFDUPYKIGVSACVYFIJC
ZVACORAOJMEYFCCHWJYINMEHNJOELRZF
CTVWLZOOTBOEHFXENMFLCSAYXNQMZQXA
LNNGDYJPHXFRGYTMLOYBQUZZVWRGQEQA
MUDVHKQRNPSXFNFPKWNMHDELDNDLKHUCD
PNLJLNJEFKGNLYXQHQCZCLLQNBGYCZGXHA
AZSOABIMIYDQJODOXPWSDAYHFDEHFQUOIH
BANKDQXCSAZLWKMAVKJBBNYRWTNDLLVRMX
UIQHKORMKFJDBYCCWBWRA'

Este resultado coincide con el cifrado que nuestro simulador realiza teniendo la misma configuración. Al pie de esta página se facilita un enlace a un vídeo donde se demuestra esto.[3]

[2] http://users.telenet.be/d.rijmenants/en/enigmasim.htm
[3] https://youtu.be/jegEpbHdZhw

6.3 Objetivos alcanzados

Llegados a este punto, se puede afirmar que se han cumplido los objetivos expuestos en el apartado 1.2, que son la adaptación de seis versiones de la máquina Enigma a un programa de simulación por ordenador y la implementación de todos sus componentes fundamentales.

Bibliografía

[1] Caballero Gil, Pino. *Introducción a la criptografía*. RA-MA Editorial, Madrid, segunda edición, 2002.

[2] Hernández Encinas, Luis. *La criptografía*. Editorial CSIC y los libros de la Catarata, Madrid, 2016.

[3] Gómez Urgellés, Joan. *El lenguaje secreto de los números: codificación y criptografía*. RBA y National Geographic, Barcelona, 2014.

[4] Singh, Simon. *The Code Book: Science of Secrecy from Ancient Egypt to Quantum Cryptography*. Doubleday, Nueva York, 1999.

[5] De León Rodríguez, Manuel y Timón, Agata. *Rompiendo Códigos: Vida y Legado de Turing*. Editorial CSIC y los libros de la Catarata, Madrid, 2014.

[6] El Enigma, Museo de Informática de la Universitat Politècnica de València. Consultar en https://histinf.blogs.upv.es/2011/11/04/2248/

[7] Karl de Leeuw. The Dutch Invention of The Rotor Machine, 1915-1923. *Cryptologia*, Vol. XXVII, nº1, pp. 73-94, enero, 2003.

[8] Dutch Patent NL10700, extracto del 7 de octubre de 1919. Consultar en https://www.cryptomuseum.com/crypto/enigma/patents/files/NL10700.pdf

[9] Die schreibende Enigma. Printing Enigma machine, 1924-1926. Consultar en https://www.cryptomuseum.com/crypto/enigma/pe26/index.htm

[10] Enigma A. Glow lamp machine, 1924. Consultar en https://www.cryptomuseum.com/crypto/enigma/a/index.htm

[11] Enigma B. Glow lamp Enigma machine, 1924-1925. Consultar en https://www.cryptomuseum.com/crypto/enigma/b/index.htm

[12] Enigma C. Glow lamp Enigma machine, 1925. Consultar en https://www.cryptomuseum.com/crypto/enigma/c/index.htm

[13] Enigma D. Commercial Enigma A26. Consultar en https://www.cryptomuseum.com/crypto/enigma/d/index.htm

[14] Bombe. Breaking the Enigma cipher. Consultar en https://www.cryptomuseum.com/crypto/bombe/index.htm#uk

[15] Enigma M4. Naval 4-wheel Enigma. Consultar en https://www.cryptomuseum.com/crypto/enigma/m4/index.htm

[16] Technical Details of the Enigma Machine. Consultar en http://users.telenet.be/d.rijmenants/en/enigmatech.htm

[17] Enigma Message Procedures. Consultar en http://users.telenet.be/d.rijmenants/en/enigmaproc.htm

[18] Scratch. Consultar en https://en.scratch-wiki.info/wiki/Scratch

[19] Scratch 3.0. Consultar en https://en.scratch-wiki.info/wiki/Scratch_3.0

[20] Jannette Marie Wing. *Computational Thinking. Viewpoint.* Communications of the ACM, 49(3):33-35. 2006.