UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Arqueología informática: Implementación de sistemas clásicos de cifrado en Scratch

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: Esteve Romero, Abel
Tutor: Molero Prieto, Xavier

Curso 2018 - 2019

Resum

El present treball se centra en la creació de quatre xifrats clàssics de la història de la criptografia (Disc d'Alberti, Xifrat Vigenère, Xifrat Playfair i Xifrat ADFGVX) per mitjà de l'ús del llenguatge de programació Scratch. També s'explica com s'ha elaborat la pàgina web on estaran explicats els xifrats que s'allotjaran al Museu d'Informàtica de l'Escola Tècnica Superior d'Enginyeria Informàtica amb l'objectiu d'ensenyar a tots els usuaris del museu com funcionen els xifrats de manera didàctica i que interactuïn amb el xifrat gràcies al programa creat. Aquest document principalment es dividirà en tres parts, a la primera part es descriurà cronològicament el context històric de cada xifrat, el funcionament, les seves peculiaritats i fins a un petit exemple. A la segona part es desenvoluparan les bases del llenguatge de programació Scratch i com s'ha adaptat la implementació dels xifrats a les limitacions d'aquest llenguatge. L'última part se centra en l'explicació detallada del procediment seguit per implementar els quatre xifrats i així incitar els usuaris més novells en programació a utilitzar Scratch per aprendre i endinsar-se al món de la programació per a assentar les bases que posseeixen els llenguatges de més alt nivell (variables locals, variables globals, llistes, bucles, operadors, esdeveniments, ...).

Paraules clau: xifrat, llenguatge de programació, Scratch, implementació, pàgina web

Resumen

El presente trabajo se centra en la creación de cuatro cifrados clásicos de la historia de la criptografía (Disco de Alberti, Cifrado Vigenère, Cifrado Playfair y Cifrado ADFGVX) por medio del uso del lenguaje de programación Scratch. También se explica como se ha elaborado la página web donde estarán explicado los cifrados que se alojarán en el Museo de Informática de l'Escola Tècnica Superior d'Enginyeria Informàtica con el objetivo de enseñar a todos los usuarios del museo como funcionan los cifrados de manera didáctica y que interactúen con el cifrado gracias al programa creado. Este documento principalmente se dividirá en tres partes, en la primera parte se describirá cronológicamente el contexto histórico de cada cifrado, el funcionamiento, sus peculiaridades y hasta un pequeño ejemplo. En la segunda parte se desarrollarán las bases del lenguaje de programación Scratch y como se ha adaptado la implementación de los cifrados a las limitaciones de dicho lenguaje. La última parte se centra en la explicación detallada del procedimiento seguido para implementar los cuatro cifrados y así incitar a los usuarios más noveles en programación a utilizar Scratch para aprender y adentrarse en el mundo de la programación para asentar las bases que poseen los lenguajes de más alto nivel (variables locales, variables globales, listas, bucles, operadores, eventos,...).

Palabras clave: cifrado, lenguaje de programación, Scratch, implementación, página web

Abstract

The present work focuses on the creation of four classic ciphers from the history of cryptography (Alberti Disk, Vigenère Cipher, Playfair Cipher, and ADFGVX Cipher) through the use of the Scratch programming language. It is also explained how the web page has been elaborated where the ciphers will be explained that will be housed in the Computer Museum of l'Escola Tècnica Superior d'Enginyeria Informàtica with the aim of teaching all the users of the museum as ciphers work in a didactic way and interact with the encryption thanks to the created program. This document will mainly be divided into three parts, in the first part the historical context of each cipher will be described chronologically, the operation, its peculiarities and even a small example. In the second

part, the foundations of the Scratch programming language will be developed and how we have adapted the implementation of the ciphers to the limitations of that language. The last part focuses on the detailed explanation of the procedure followed to implement the four ciphers and thus encourage the youngest users in programming to use Scratch to learn and enter the world of programming to establish the bases that have the languages of highest level (local variables, global variables, lists, loops, operators, events, ...).

Key words: cipher, programming language, Scratch, implementation, web page

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

En este capítulo se expondrá el motivo por el cual se ha escogido este trabajo, se tratarán de explicar detalladamente todos objetivos a tener en cuenta en toda la realización del trabajo, también se explicará la estructura que se utilizará a lo largo del documento. Por otra parte habrá una bibliografía en la cual se explicará cómo se ha utilizado en la creación de nuestros programas de cifrado en el lenguaje de programación Scratch

1.1 Motivación

La principal motivación para la realización de este proyecto es poder acercar el mundo de la programación a los jóvenes y despertar un interés por él. Es por ello que el Museo de Informática de l'Escola Tècnica Superior d'Enginyeria Informàtica de la Univesitat Politècnica de València ofrece un recorrido por la historia de la informática.

Una de las actividades que ofrece el museo es el taller de Scratch, *Scratch Day*, que intenta acercar a todos los participantes del taller al mundo de la programación, a través de un lenguaje sencillo e interactivo como es Scratch. El presente trabajo servirá como ejemplo para conocer los cifrados que se utilizaron en la historia.

Como contrapunto el trabajo también pretende acercar a los usuarios de la página web del museo en el mundo de la programación y no solamente describir el funcionamiento e historia de los cifrados.

1.2 Objetivos

Los objetivos del presente trabajo es la creación de cuatro cifrado utilizados en la historia usando el lenguaje de programación Scratch y orientado a los jóvenes. Los cifrados a crear son: El cifrado de Alberti (o Disco de Alberti), el cifrado de Vigenère, el cifrado Playfair y cifrado ADFGVX (o cifrado alemán). Posteriormente, se incluirán dichos cifrados en la página web del Museo de Informática.

Además de la creación de los cifrados, el trabajo tiene como objetivos conocer la historia de dichos cifrados de modo que el lector pueda ver las diferencias y sacar sus propias conclusiones. Cabe destacar que se puede encontrar el código de los programas realizados en la página del museo y de esta manera conocer cómo está programado internamente cada cifrado. Se trata de una manera interactiva de aprender tanto acerca de los cifrado como de saber implementarlos.

1.3 Estructura de la memoria

El presente trabajo se estructura en diez capítulos, a continuación describiremos brevemente su estructura:

- **Capítulo 1.** Se presenta la motivación, los objetivos y la estructura de la memoria.
- **Capítulo 2.** Se presentará una aproximación histórica de los cifrados nombrados.
- **Capítulo 3.** Se explicará de manera detallada el entorno y el lenguaje de programación Scratch. Se explicará de manera precisa todos las herramientas que nos ofrece para la creación de programas sencillos y la metodología empleada en nuestros cifrados.
- **Capítulos 4, 5, 6, 7.** Se detallarán los cifrados que hemos realizado, comentando los fundamentos histórico y las fortalezas y las debilidades a la hora de descifrar el criptograma original.
- **Capítulo 8.** Se especificará cómo está estructurada la página web y qué tecnología hemos utilizado para la creación de dicha página web.
- **Capítulo 9.** En este último capítulo se mostrarán las conclusiones obtenidas del trabajo realizado, como los objetivos hemos cumplido y sugerir trabajos futuros.

1.4 Uso de la bibliografía

En esta sección se va ha hablar de la bibliografía utilizada en la realización de este trabajo, centrándonos principalmente en los investigación de los recursos que nos ofrece el lenguaje de programación Scratch.

También hay que comentar que para la composición de la memoria se consultaron trabajos anteriores realizados por antiguos alumnos de la escuela, centrando nuestro interés en uno de ellos, realizado por Le Danny Yang, ya que se trata de otro trabajo sobre los cifrados clásicos [11].

Antes de empezar estudiando los cifrados se ha investigado sobre la criptografía y como ha ido evolucionando los diferentes tipos de cifrados a lo largo de la historia, para ello se han utilizado los siguientes enlaces [4], [9] y [17].

Toda la información utilizada para la explicación del contexto histórico de los cifrados se pueden encontrar en las siguiente páginas [5], [6], [7], [8], [10], [15] y [16]. Para explicar el entorno de programación Scratch y las herramientas que nos ofrece se han usado las páginas [1], [2], [3], [12], [13], [14] y [18].

También se ha buscado información sobre el funcionamiento de cada uno de los cifrados y qué dificultades podrían producirse a la hora de implementarlos. Al explicar un ejemplo de cada cifrado se ha referenciado un enlace a un programa interactivo para consultar el cifrado en cuestión.

1.5 Derechos de autor

Todas las imágenes utilizadas tanto en los programas de cifrado como las que aparecen en la memoria han sido obtenidas libres de derechos de autor y siempre usadas para el ámbito académico. Debido a que no se ha encontrado una imagen correcta del disco de

Alberti, ha sido creada exclusivamente para este trabajo. Hay que añadir que todos los sonidos utilizados en los cifrados pertenecen a la biblioteca propia de Scratch.

CAPÍTULO 2

La criptografía clásica: una mirada al pasado

En este capítulo se va a exponer el contexto histórico de los cifrados (Cifrado de Alberti, Cifrado de Vigenère, Cifrado Playfair y Cifrado ADFGVX) presentados de forma cronológica. También se va a describir su funcionamiento tanto el cifrado del mensaje como el descifrado del criptograma, así como un breve ejemplo de cada cifrado.

2.1 Cifrado de Alberti (1467)

2.1.1. Contexto histórico

Leon Battista Alberti (véase Figura 2.1) nació en Génova en 1404. Fue un teórico del arte, arquitecto y escritor, y junto a Leonardo Da Vinci, fue una de las figuras más representativas del Renacimiento ya que reunía todos los conocimientos de la época.

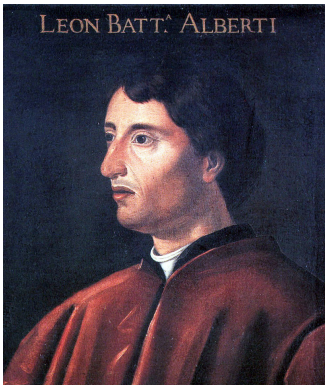


Figura 2.1: Retrato de Leon Battista Alberti.

Hijo del mercader florentino, Lorenzo Alberti y de Bianca Fieschi. Alberti recibió la mejor educación disponible para un noble italiano. Desde 1414 a 1418 estudió los clásicos [1] en la prestigiosa escuela de Gasparino Barzizza en Padua. Más adelante completaría sus estudios estudiando derecho en la universidad de Bolonia.

Alberti reconocía a las matemáticas como el campo común entre las artes y las ciencias, por ello comenzó su tratado *Della Pittura*. Dicho tratado se basaba en el contenido científico de la óptica clásica en la determinación de la perspectiva como instrumento geométrico de la representación artística y arquitectónica.

[1] Estudio de las lenguas, cultura, historia y el pensamiento de la civilización de la Antigua Grecia y Roma.

Debido a su carácter polifacético, Alberti también realizó contribuciones en otros campos como la arquitectura, dado que pasó mucho tiempo estudiando lugares, ruinas y objetos antiguos detalló dichas observaciones en un nuevo tratado *De Re Aedificatoria* (véase Figura 2.2). Dicho tratado fue el primer tratado sobre arquitectura del Renacimiento e influyente en la arquitectura posterior. En el siglo XVIII fue traducido al italiano, francés, inglés y español.

Figura 2.2: De Re Aedificatoria.

Entre las diversas obras de Alberti hay que destacar la obra *De Cifris* o *De componendis cifris* donde trata la criptografía, otro campo donde fue experto. En dicho trabajo explica el cifrado por el que es conocido, se trata de dos discos concéntricos con 24 caracteres cada uno que rotan haciendo que un carácter del disco exterior pueda coincidir con cualquiera del disco interior.

El cifrado de Alberti fue el primer avance significativo desde el cifrado César, hablamos del siglo I a.C, se trata de un sistema polialfabético porque la sustitución de un carácter se basa en función de un criterio de ordenación. También hay que comentar que el historiador criptográfico David Kahn en su libro *The Codebreakers: The comprehensive history of secret communication from ancient times to the internet* [10], escrito en 1967, le nombró como *Father of Western Cryptography*.

2.1.2. Cifrado

Para comenzar a explicar el cifrado, primero hay que dejar claro tres términos que utilizaremos. El *periodo* indica la frecuencia de aparición de los caracteres del mensaje que queramos enviar al receptor, el *incremento* es el número de veces que rota el disco interior en cada periodo y la *configuración inicial* es la configuración o posición en la que se encuentra el disco interno a la hora de cifrar. Hay que tener en cuenta que la combinación de estos parámetros conforman la clave del cifrado.

Los discos que conforman el Disco de Alberti (véase Figura 2.3) presentan diferencias, el disco interno presenta el alfabeto en un cierto orden y con el símbolo & y el disco externo presenta el abecedario ordenado y los dígitos 1, 2, 3 y 4 para formar códigos en el cifrado. Por ejemplo el 214 podría significar "Las naves están listas para zarpar". Alberti recomendó cifrar el código a su vez con el fin de que si se capturara el libro de códigos, el mensaje permaneciera seguro. También hay que tener en cuenta que para mostrar la diferencia entre el disco externo y el interno el alfabeto del disco externo está en mayúsculas y el alfabeto del disco interior en minúsculas.

Figura 2.3: Disco de Alberti.

También hay que tener en cuenta que el disco exterior tienen 24 celdas, hay 20 caracteres y 4 dígitos, se deduce que no todo el alfabeto esta representado, por ello hay que realizar una sustitución de algunos caracteres para que todas las combinaciones estén presentes en el cifrado. En la tabla 2.1 se muestran estas equivalencias.

H	FF
J	II
K	QQ
U	VV
W	XX
Y	ZZ

Tabla 2.1: Tabla de equivalencia en el cifrado Alberti.

A partir de aquí ya se puede explicar cómo funciona el cifrado. El funcionamiento consiste en colocar los discos en la configuración inicial acordada, un carácter del disco exterior se sustituiría por el carácter inmediatamente inferior. La complejidad del cifrado viene dada por el periodo ya que en cada periodo establecido el disco rotaba un incremento (si el incremento es de cinco, el disco interior rotará cinco caracteres en sentido antihorario) y gracias a esta rotación el carácter anterior ahora se sustituye por otro carácter diferente consiguiendo así un sistema polialfabético. En la Figura 2.3 se muestra como el carácter *A* se sustituiría por el carácter *m*.

2.1.3. Descifrado

Durante la época, Alberti afirmó que su cifrado era indescifrable y lo denominó en sus propias palabras como "digno de reyes", porque si no se conoce la clave (periodo, incremento y configuración inicial) es casi imposible descifrar el mensaje.

En el caso de que sí se conociese la configuración inicial, el incremento y el periodo, para el descifrado, habría que realizar los mismos procesos que con el cifrado pero sustituyendo el carácter inferior por el carácter inmediatamente superior.

2.1.4. Ejemplo

Para ilustrar el cifrado se va a comentar un ejemplo [2] los parámetros son el *periodo* es de 5, el *incremento* es de 2 y la *configuración inicial* es la configuración de la figura 2.3. El mensaje que usaremos es *HOLAMUNDO*, como se comenta en la sección 2.1.2 hay que realizar una sustitución de algunos caracteres por lo que el mensaje quedaría como *FFOLAMVNDO*. En la tabla 2.2 se muestra como quedaría cifrado el mensaje.

H		O	L	A	M	U		N	D	O
F	F	O	L	A	M	V	V	N	D	O
D	D	Z	R	P	O	P	P	P	A	S

Tabla 2.2: Ejemplo de cifrado Alberti.

2.2 Cifrado de Vigenère (1553)

2.2.1. Contexto histórico

Para situarnos en el contexto histórico hay que recordar que el cifrado de Alberti es el primer cifrado polialfabético correctamente documentado. En el año 1508 Johannes Trithemius escribe el libro *Poligraphia*, una obra criptográfica dedicada al arte de la esteganografía [3], donde crea la *tabula recta* que es una tabla cuadrada de alfabetos donde cada fila se completa desplazando la fila anterior hacia la izquierda (véase Figura 2.4) y con ella también crea el cifrado Trithemius.

El cifrado Vigenère originalmente descrito por Giovan Battista Bellaso en el libro *La cifra del. Sig. Giovan Battista Bellaso* (véase Figura 2.5) en el año 1553 donde utilizó la *tabula recta* añadiendo una contraseña o clave repetida para cambiar el alfabeto de cada cifrado en cada letra. Mientras que en el cifrado de Alberti y en el de Trithemius se usa un patrón de sustituciones fijos, en el que propuso Bellaso el patrón cambia cada vez que se cambia la contraseña.

Blaise de Vigenère propuso una descripción muy similar pero con una clave más segura ante la corte de Enrique III de Francia en el año 1556, este hecho hizo que en el siglo IX se le atribuyese a Vigenère la invención del cifrado descrito por Bellaso. David Kahn en su libro [10] se hace eco de esta malinterpretación diciendo esto sobre la historia de la criptografía:

> It ignored this important contribution and instead named a regressive and elementary cipher for him though he had nothing to do with it (p 145).

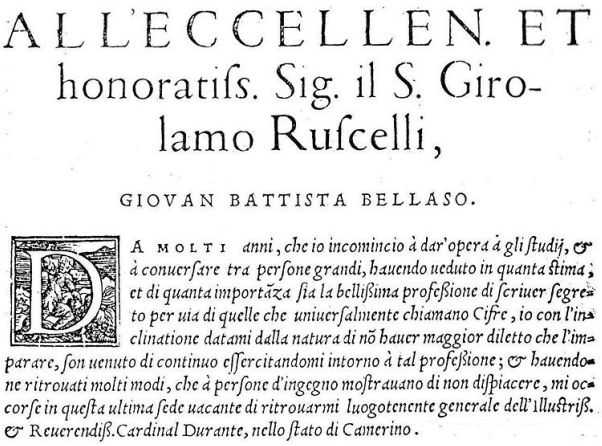
La historia de la criptografía ignora esta importante contribución e insta a nombrar el cifrado elemental y regresivo a Vigenère aunque él no hizo nada.

[2] En el siguiente enlace se puede consultar y probar el cifrado https://www.dcode.fr/alberti-cipher.
[3] Es el estudio y la aplicación de técnicas para la ocultación de mensajes u objetos.

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

Figura 2.4: Tabula Recta.

El cifrado de Vigenère ganó bastante reputación por ser robusto y parecer irresoluble, esto le hizo ganarse el apodo de cifrado indescifrabre (*le chiffre indéchiffrable* en francés), tanto fue así que el notable autor y matemático Charles Lutwidge Dodgson (Lewis Carroll) también lo denominó así en su obra de 1868 *The Alphabet Cipher* y también se describió como tal en la revista *Scientific American* en 1917.



ALL'ECCELLEN. ET
honoratiſs. Sig. il S. Giro-
lamo Ruſcelli,

GIOVAN BATTISTA BELLASO.

DA MOLTI anni, che io incomincio à dar'opera à gli ſtudij, & à conuerſare tra perſone grandi, hauendo ueduto in quanta ſtima, et di quanta importãza ſia la belliſſima profeſſione di ſcriuer ſegreto per uia di quelle che uniuerſalmente chiamano Cifre, io con l'inclinatione datami dalla natura di nõ hauer maggior diletto che l'imparare, ſon uenuto di continuo eſſercitandomi intorno à tal profeſſione; & hauendone ritrouati molti modi, che à perſone d'ingegno moſtrauano di non diſpiacere, mi occorſe in queſta ultima ſede uacante di ritrouarmi luogotenente generale dell'Illuſtriſſ. & Reuerendiſſ. Cardinal Durante, nello ſtato di Camerino.

Figura 2.5: La cifra del. Sig. Giovan Battista Bellaso.

Dicha reputación fue inmerecida ya que Charles Babbage consiguió descifrarlo cerca del año 1854, aunque no publicó dicho trabajo. Kasiski consiguió descifrar el cifrado y publicar dicho descubrimiento en el siglo XIX.

El cifrado de Vigenère fue utilizado en la Guerra Civil Americana, al ser de fácil uso cuando se usan discos de cifrado, los Estados Confederados de América lo utilizaban para el envío de mensajes cifrados aunque la Unión conseguía descifrarlos. En 1918 Gilbert

Vernam propuso una variable al cifrado creando el cifrado Vernam-Vigenère o simplemente cifrado Vernam.

2.2.2. Cifrado

La parte más importante del cifrado de Vigenère es tener la *tabula recta* o tabla de Vigenère a mano, se escribe un mensaje y una clave. Una vez se tiene el mensaje se repite la clave hasta que tenga la misma longitud que el mensaje, estos serán los pasos previos para poder empezar a cifrar.

Una vez realizado el paso anterior se va carácter por carácter, obteniendo la posición de los caracteres del mensaje en el alfabeto utilizado en la tabula recta y se hace lo mismo con la nueva clave, obtenidas todas las posiciones se realiza la suma carácter a carácter del mensaje y de la clave, a la vez que se va realizando el módulo con la longitud de alfabeto y ese resultado es la posición del carácter del mensaje cifrado en el alfabeto.

En términos matemáticos el cifrado se realizaría con la siguiente fórmula.

$$C(X_i) = (M_i + C_i) \setminus mod\ L$$

Donde M_i es la posición del carácter del mensaje en el alfabeto, C_i es la posición del carácter de la clave en el alfabeto y L es la longitud del alfabeto, en nuestro caso es 27 al utilizar el alfabeto español.

2.2.3. Descifrado

Para el descifrado hay que hacer la operación inversa del cifrado con el criptograma y la clave. El descifrado se realiza también carácter a carácter, obtenemos la posición de los caracteres del mensaje cifrado en el alfabeto y realizamos la resta con la posición del carácter de la nueva clave en el alfabeto. Si esa resta es menor o igual a 0 se suma la longitud del alfabeto y se realiza el módulo con la longitud del alfabeto con el resultado pero si esa resta es mayor se realiza directamente el módulo con la longitud del alfabeto. En términos matemáticos el descifrado se realizaría con las siguientes fórmulas.
Si $(MC_i - C_i) \leq 0$ entonces:

$$D(X_i) = (MC_i - C_i + L) \setminus mod\ L$$

Si $(MC_i - C_i) > 0$ entonces:

$$D(X_i) = (MC_i - C_i) \setminus mod\ L$$

Donde MC_i es la posición del carácter del criptograma en la tabula recta, C_i es la posición del carácter de la clave en la tabula recta y L es la longitud del alfabeto, en nuestro caso es 27 al utilizar el alfabeto español.

2.2.4. Ejemplo

En este siguiente ejemplo [4] tenemos un mensaje *hola mundo* y una clave *inf*, las posiciones se han calculado con la Figura 2.4, empezando por el valor 0. En la tabla 2.3 se pueden observar las posiciones de los caracteres del mensaje y la clave, y el mensaje cifrado.

[4] En el siguiente enlace se puede consultar y probar el cifrado https://es.planetcalc.com/2468/.

h	o	l	a		m	u	n	d	o
7	15	11	0	-	12	21	15	3	15
i	n	f	i		n	f	i	n	f
8	13	5	8	-	13	5	8	13	5
15	1	16	8	-	25	26	21	16	20
o	b	p	i		y	z	u	p	t

Tabla 2.3: Ejemplo de cifrado Vigenère.

2.3 Cifrado de Playfair (1854)

2.3.1. Contexto histórico

Otra de las ironías de la historia criptográfica sucede con el cifrado Playfair. Charles Wheatstone (véase Figura 2.6) nombró un dispositivo de cifrado que permitía un nuevo cifrado, parecido al dispositivo de cifrado que inventó en 1817 por el coronel americano Decius Wadsworth, aunque no le dio importancia a ese dispositivo. Wheatstone inventó este cifrado para que fuera usado en los mensajes enviados por telégrafo, para garantizar secretismo a los mensajes pero gracias a Playfair ganó popularidad al utilizarse en uso militar.

El cifrado se nombró cifrado Playfair por Lord Playfair (véase Figura 2.7), primer Lord Playfair de St. Andrews y amigo de Wheatstone. Ellos se entretenían resolviendo los mensajes cifrados en el *London Times*. Una vez descubrieron la correspondencia entre un estudiante de Oxford y su mujer en Londres, cuando él le propuso fugarse Wheatstone insertó un mensaje con el mismo cifrado que decía que su cifrado había sido descubierto.

Figura 2.6: Retrato de Charles Wheatstone.

Playfair describió el cifrado como «el cifrado simétrico descubierto recientemente por Wheatstone» en una cena con miembros del gobierno inglés como el consejero del gobierno Lord Granville, el Principe Albert y el futuro primer ministro Lord Palmerston entre otros. Playfair les explicó el sistema de cifrado y días después recibió dos cartas, de Palmerston y Granville, mostrando que realmente dominaban el cifrado.

Este cifrado, es el primer cifrado en la historia de la criptografía en ser digrámico, es decir, en cifrar cada par de letras a la vez y el resultado de ambas depende de ellas. Wheatstone reconoció que el cifrado funcionaba tanto si se organizaba en un rectángulo como en un cuadrado, empezó a usar en el cifrado un alfabeto completamente mezclado el cual es originado por la transposición de la clave, se trataba de una versión primaria del

Figura 2.7: Retrato de Lord Playfair.

cifrado. Debajo de la palabra clave escribiría las letras restantes del alfabeto y derivando el alfabeto mezclado leyendo las columnas verticalmente.

Dicha característica duro poco debido al menor común denominador, igual que el cifrado de Vigenère. La clave se inscribía en una tabla 5x5 seguido de las letras restantes del abecedario, hay que indicar que las letras *I* y *J* se escribirían en la misma celda. La práctica de este cifrado aunque facilitó la invulnerabilidad en los mensajes también ayudó a la disminución de la seguridad del mismo.

Debido a la falta de seguridad en el cifrado se crearon dos variantes (*Two Square* y *Four Square*) en las que la seguridad y la robustez del cifrado aumentaban. *Two Square* o Dos Cuadros utiliza 2 cuadros de 5x5, existen dos variantes una en horizontal y otra en vertical y el método para cifrar es un poco distinta que en cifrado Playfair. *Four Square* o Cuatro Cuadros utiliza 4 cuadros de 5x5, fue inventado por Felix Delastelle, si se analiza carácter por carácter se trataría de un cifrado de sustitución polialfabético pero si se analiza por pares de caracteres sería un cifrado de sustitución monoalfabético.

Hay que comentar que dicho cifrado aparece en la película de 2007 llamada *National Treasure: Book of Secrets* (La búsqueda 2: El diario secreto), del director *Jon Turteltaub* y protagonizada por *Nicolas Cage* donde se nos sitúa al final de Guerra Civil Americana y es necesario descifrar un mensaje cifrado para descubrir el mensaje que esconde un mapa.

2.3.2. Cifrado

Como se dijo anteriormente este cifrado es digrámico, es decir, se cifra por pares de caracteres, para cifrar se necesita una tabla o matriz de 5x5 y unas sencillas reglas que permiten el cifrado de los pares de caracteres. Primero se dividirá el mensaje en pares de caracteres y después se le aplicarán las reglas de cifrado. Dichas reglas son:

1. Si los caracteres Ox y Oy están en la misma línea se cifrarán con la letra inmediatamente de la derecha. (Si fuera el último el carácter cifrado sería el primero de esa línea).

2. Si los caracteres Ox y Oy están en la misma columna se cifrarán con la letra inmediatamente inferior. (Si fuera el último el carácter cifrado sería el primero de esa columna).

3. Si los caracteres Ox y Oy están en líneas y columnas distintas, no imaginaremos un rectángulo y los caracteres serían las esquinas diagonalmente opuestas, los ca-

racteres cifrados serían los caracteres que corresponderían a las otras esquinas del rectángulo en la línea correspondiente.

4. Si los caracteres Ox y Oy son iguales se añade el carácter *X* entre ellos.

5. Si la longitud del mensaje es impar, lo que implica que la última pareja de caracteres estaría incompleta, se añadiría el carácter *X*.

Un caso especial seria la letra *J* que se sustituiría por la letra *I* y, en el caso del alfabeto español la letra $\tilde{N}$ se sustituiría por la letra *N*.

2.3.3. Descifrado

Para el descifrado primero se dividiría el mensaje por pares de caracteres y se aplicarían las reglas opuestas al cifrado. Dichas reglas son:

1. Si los caracteres Ox y Oy están en la misma línea se descifrarán con la letra inmediatamente de la izquierda. (Si fuera el primero el carácter cifrado sería el último de esa línea).

2. Si los caracteres Ox y Oy están en la misma columna se descifrarán con la letra inmediatamente superior. (Si fuera el primero el carácter cifrado sería el último de esa columna).

3. Si los caracteres Ox y Oy están en líneas y columnas distintas, no imaginaremos un rectángulo y los caracteres serían las esquinas diagonalmente opuestas, los caracteres cifrados serían los caracteres que corresponderían a las otras esquinas del rectángulo en la respectiva línea.

Las reglas número 4 y 5 del cifrado se aplicaría una vez descifrado el mensaje ya que dependerá del contexto del mensaje original, por lo que en el descifrado se podrían eliminar estas reglas.

2.3.4. Ejemplo

En esta parte se mostrará un ejemplo [5] del cifrado Playfair. En este ejemplo la clave sería *info* y el mensaje *hola mundo* por lo que el cifrado será el siguiente. Se empieza creando la tabla con la clave y el resto del alfabeto (véase Figura 2.4), cuando esta creada se dividirá el mensaje en pares de caracteres y se aplicarán las reglas explicadas en la sección 2.3.2, una vez cifrado quedará como en la tabla 2.5

i	n	f	o	a
b	c	d	e	g
h	k	l	m	p
q	r	s	t	u
v	w	x	y	z

Tabla 2.4: Tabla del cifrado Playfair con la clave *info*

.

[5] En el siguiente enlace se puede consultar y probar el cifrado https://www.dcode.fr/playfair-cipher.

h	o	l	a	m	u	n	d	o	(x)
m	i	p	f	p	t	f	c	f	y

Tabla 2.5: Cifrado Playfair con el mensaje *hola mundo*.

2.4 Cifrado ADFGVX (1917)

2.4.1. Contexto histórico

El cifrado más famoso de toda la criptografía es el cifrado ADFGVX, fue creado en por el alemán Fritz Nebel cerca del año 1918. Dicho cifrado se basa en el tablero de Polibio y utiliza la sustitución y la transposición.

El tablero de Polibio usa un cuadro de 5x5, con los lados numerado, donde se insertan los 24 caracteres del alfabeto griego, si se utiliza el alfabeto español se combinarían los caracteres *i/j* y *n/ñ* en las respectivas celdas. Este cifrado utiliza la sustitución para convertir los caracteres en los números correspondientes. En la tabla 2.6 se muestra un ejemplo del tablero de Polibio (el carácter *s* se transformaría en 43).

	1	2	3	4	5
1	a	b	c	d	e
2	f	g	h	i/j	k
3	l	m	n/ñ	o	p
4	q	r	s	t	u
5	v	w	x	y	z

Tabla 2.6: Tablero de Polibio con el alfabeto español.

El cifrado ADFGVX tiene dos variables, una de ellas es la que utiliza un tablero de Polibio de 5x5 donde sustituiría lo numeración de los laterales por ADFGX y otra más compleja donde se utiliza el tablero de Polibio de 6x6 donde se sustituiría la numeración de los laterales por ADFGVX y se introduciría en el tablero los primeros diez dígitos.

La primera versión del cifrado AFGVX, cifrado ADFGX, se usó en las transmisiones de audio entre operadores de radio alemanes en la Primera Guerra Mundial. Hay que comentar que se utilizaron estas letras para el cifrado por que son muy diferenciadas en código morse (véase tabla 2.7), gracias a esto los operadores solo se tenían que aprender los 6 caracteres del cifrado y no todas las combinaciones del código.

A	·	-	-	
D	-	·	·	
F	·	·	-	·
G	-	-	·	
V	·	·	·	-
X	-	·	·	-

Tabla 2.7: Caracteres ADFGVX en codigo Morse.

Georges Paivin (vease Figura 2.8), el mejor criptoanalista en Beau du Chiffre, descubrió el cifrado e intentó descifrarlo. La presencia de 5 caracteres le sugirió que se trataba de un tablero, utilizó lo métodos conocidos de distinción y de transposición y no consiguió descifrarlo. El intenso envío de mensajes entre los alemanes permitió a Paivin tener suficientes mensajes para encontrar un patrón. David Kahn en su libro [10] describe en detalle el método de descifrado utilizado por Paivin.

Figura 2.8: Retrato de Georges Paivin.

Con la introducción del sexto carácter en el cifrado a Paivin le costó más descubrir el método de descifrado, por que no sabía el motivo por el cual ese carácter estaba ahí, cuando interceptó dos mensaje casi idénticos, los analizó y por fin descubrió el método de descifrado.

2.4.2. Cifrado

Para el cifrado se define una configuración inicial del tablero de Polibio. Una vez la configuración inicial está definida se empieza a aplicar la sustitución al mensaje que queremos cifrar. Aplicando la sustitución a un carácter éste se quedará como el carácter de la línea y de la columna donde está situado dicho carácter, por ejemplo, si usamos la configuración de la tabla 2.8 el carácter *M* quedaría como *DG*.

Hasta aquí se ha realizado la parte de sustitución, ahora se efectuará la transposición con el nuevo mensaje. Este nuevo mensaje se ordenará en columnas por debajo de nuestra clave, si en la tabla quedasen espacios libres se añadirán caracteres nulos al final, dichos caracteres nulos serían *DFGVX* añadiendo solo los que sean necesarios para completar la tabla.

Seguidamente se ordenarán las columnas respecto al orden de los caracteres de la clave en el alfabeto y por último solo se tendrá que leer el mensaje columna a columna obviando la clave. Hay que añadir que el mensaje cifrado usualmente se envía en bloques de 5 caracteres.

2.4.3. Descifrado

Para el descifrado habrá que seguir los pasos del cifrado pero en orden inverso. Primero colocamos el mensaje cifrado en columnas con respecto a la clave ordenada alfabéticamente, después ordenamos las columnas con la clave original. Una vez ordenado el mensaje se puede proceder a la sustitución de cada par de caracteres en el tablero de Polibio obteniendo así el mensaje original. Si se diera el caso de que el mensaje original tiene valores nuevos estos aparecerán al final del mensaje original sin aportarle nada nuevo a este.

2.4.4. Ejemplo

Para explicar un ejemplo [6] de este cifrado se empezará con la configuración inicial del tablero de Polibio, dicha configuración se muestra en la tabla 2.8. Una vez se tiene la configuración inicial se empieza a aplicar la sustitución al mensaje que se quiere cifrar, nuestro mensaje es *holamundo*, la sustitución quedará como vemos en la tabla 2.9.

	A	D	F	G	V	X
A	0	F	Q	B	I	Z
D	W	1	X	M	8	E
F	R	N	2	G	T	S
G	K	P	U	3	J	D
V	O	7	H	Y	4	V
X	9	C	L	6	A	5

Tabla 2.8: Tablero de Polibio con la configuración inicial.

h	o	l	a	m	u	n	d	o
VF	VA	XF	XV	DG	GF	FD	GX	VA

Tabla 2.9: Sustitución del mensaje en el cifrado ADFGVX.

Hasta aquí se ha realizado la parte de sustitución, ahora se procederá a la transposición con el nuevo mensaje. Este nuevo mensaje se ordenará en columnas por debajo de nuestra clave *info*, se puede ver nuestro ejemplo en la tabla 2.10, si la tabla no se completa se añadirán caracteres nulos al final, como la tabla no se completa se añaden los caracteres *v* y *x*. Ahora se ordenarán las columnas respecto al orden de los caracteres en el alfabeto, véase tabla 2.11 y por último solo tendremos que leer el mensaje columna a columna.

I	N	F	O
V	F	V	A
X	F	X	V
D	G	G	F
F	D	G	X
V	A	(V	X)

Tabla 2.10: Tabla con la clave.

F	I	N	O
V	V	F	A
X	X	F	V
G	D	G	F
G	F	D	X
V	V	A	X

Tabla 2.11: Tabla con la clave ordenada alfabéticamente.

El mensaje final quedaría tal que así *VXGGVVXDFVFFGDAAVFXX* y si lo se pusiera en bloques de 5 caracteres quedaría *VXGGV VXDFV FFGDA AVFXX*.

[6] En el siguiente enlace se puede consultar y probar el cifrado http://www.grasoft.be/geotools/ADFGVX.htm.

CAPÍTULO 3
El entorno de programación Scratch

En este capítulo se describirá el entorno de programación Scratch además del porqué de dicha elección. También se hablará de las herramientas que nos ofrece para la creación de programas.

3.1 ¿Por qué Scratch?

Aquí se describirán las principales razones por la que se ha escogido este entorno de programación y no otro:

1. **Entorno de programación.** El lenguaje de programación Scratch ha sido concebido con el propósito de que los usuarios sin previos conocimientos de programación puedan crear crear sus programas desde cero o a partir de código de otros ya que permite la compartición de los proyecto de forma fácil y sencilla. El hecho de que los proyectos se puedan reutilizar y modificar su código ahorra una cantidad inmensa de tiempo y trabajo, como indica su logo (véase Figura 3.1).

2. **Facilidad de uso.** A pesar de que el lenguaje de programación Scratch no posee las características de los entornos de programación de alto nivel (corrección de sintaxis, importación de librerías, control de versiones...), es el lenguaje idóneo para los más jóvenes inexpertos ya que se trata de un lenguaje sencillo cuya forma de programar se basa en bloques (similar a los bloques *LEGO*) que se unen entre ellos para añadir funcionalidad al programa que se esté creando.

3. **Compartición de código.** Scratch es de carácter libre, es decir, los proyectos pueden ser compartido, cualquier usuario puede acceder a su código interno y si lo desea importar el proyecto y modificarlo a su gusto para realizar un nueva versión del proyecto. Gracias a esta funcionalidad de Scratch se enriquece tanto al creador del proyecto como al usuario que desea empezar en el entorno de programación.

4. **Curva de aprendizaje.** La curva de aprendizaje es mínima, dicho de otro modo, el tiempo que se requiere para aprender como funciona el entorno de programación Scratch es prácticamente nulo. Una importante parte de esta justificación es la inmensa cantidad de bibliografía que existe hoy en día en la web de Scratch y cursos externos que podemos encontrar en `https://www.udemy.com`. Actualmente Scratch se encuentra en la versión 3.0.

5. **Editor gráfico *online*.** Scratch cuenta con un editor gráfico donde el usuario puede crear o editar imágenes para incorporarlos en su proyecto. En secciones posteriores se describirá con más detalle sobre esta funcionalidad.

6. **Extensiones.** Scratch permite añadir extensiones al proyecto del usuario para poder programar con dispositivos físicos como *micro:bit* o *kits de robótica de LEGO* y otros, por ejemplo, para traducir texto dentro de su proyecto.

7. **Software libre.** Scratch es totalmente gratuito y al alcance de cualquier usuario con ganas de comenzar en el entorno de la programación. Por otra parte, Scratch está traducido a más de 50 idiomas y se sigue traduciendo a más idiomas actualmente.

Figura 3.1: Logo de Scratch con su lema «Imagina, Programa, Comparte».

3.2 Proyecto Scratch

Scratch es una plataforma multimedia de programación con versión de escritorio y versión web. El público para el cual esta diseñado son niños y profesores de escuelas secundarias (para niños más pequeños existe una versión más simple denominada *ScratchJr*). Además ayuda a los usuarios a pensar de forma creativa, razonar sistemáticamente y trabajar colaborativamente ya que permite compartir los proyectos. Todo el contenido de la plataforma se puede encontrar en su propia página web (véase Figura 3.2)[1].

El proyecto se crea en 2003 por la iniciativa del grupo de investigadores *Lifelong Kindergarten* del MIT (*Massachusetts Institute of Technology*) Media Lab, liderado por Mitchel Resnick. La idea de este lenguaje de programación surgió entre los integrantes del grupo y se basó en el lenguaje *LOGO*[2]. Dicho lenguaje de programación era de alto nivel aunque fácil de aprender por lo que Scratch adoptó esta característica.

El término Scratch viene por la técnica usada por los *disk jockeys* denominada *scratching* que se basa en hacer modificaciones en las melodías moviendo el vinilo hacia delante y hacia atrás. En este entorno de programación es algo similar debido a que se pueden mezclar animaciones, sonidos e imágenes, en palabras de Mitchel Resnick:

> *We take the name "Scratch", from the way that hip-hop disk jockeys scratch with music. They take pieces of music and then combine them together in unexpected and creative ways.*

[1] Página web de Scratch https://scratch.mit.edu/.

[2] Guía realizada por Marisa Carro Rubiera. Dpto. de Tecnologías. I.E.S. "Elisa y Luis Villamil" (Vegadeo). CURSO 2008-2009. http://www.edu.xunta.gal/centros/iesricardomella/system/files/logo1_0.pdf.

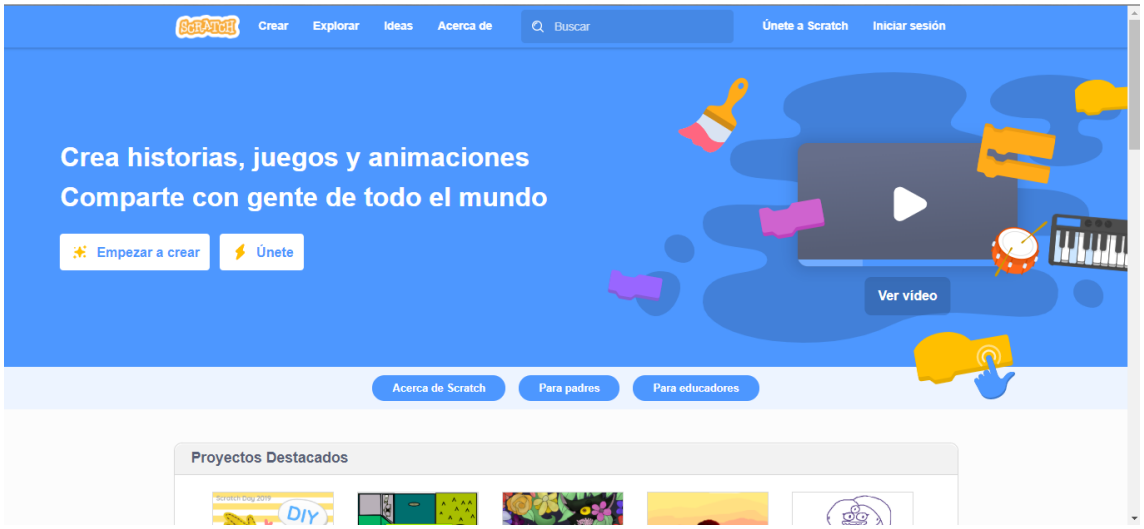


Figura 3.2: Pantalla inicial de la web de Scratch.

Otro significado diferente del nombre de este entorno de programación podría ser debido al término *from scratch* que en inglés significa desde cero ya que en este entorno de programación el público al que va dirigido tiene conocimientos casi nulos de programación.

Scratch es un proyecto de desarrollo cerrado, eso quiere decir, que el grupo de investigación no intenta implicar a los usuarios a contribuir en la elaboración del proyecto sino que se realiza íntegramente por ellos. También es de código abierto, dicho de otra forma, el grupo de investigación libera la totalidad del proyecto para que la comunidad pueda modificarlo y añadir extensiones. La traducción del proyecto se realiza mediante una página externa pero en su servidor de traducciones (`https://www.transifex.com/llk/scratch-editor/`) en el que cualquier usuario puede ayudar a traducir el proyecto en cualquier idioma.

Al final del año 2007 la página web oficial fue publicada coincidiendo con la salida de la segunda versión del proyecto denominada *Scratch 2.0*. Una importante característica fue el componente social en que los usuarios, llamados también *scratchers*, podían compartir sus proyectos con la comunidad, visualizar o modificar los proyectos de otros usuarios. El 2 de enero del año 2019 se actualizó a la versión 3.0 cambiando por completo la apariencia tanto de la web como de su plataforma. En la Figura 3.3 se puede observar la sección de comentarios de otros *scratchers* en un proyecto compartido en la web.

3.3 Objetivo de Scratch

El grupo de investigadores del MIT Media Lab pensó el lenguaje de programación Scratch con un propósito en mente, acercar el arte de la programación a los más jóvenes y de esta forma aprender a programar de manera didáctica y divertida. El componente social juega un importante papel porque ofrece la posibilidad a los usuarios visualizar el código interno de los programas ajenos compartidos en la web y poder modificarlo para crear una nueva versión, y al estar traducido actualmente a más de 50 idiomas facilita en gran medida el proceso de aprendizaje.

Cabe destacar que Scratch permite la inclusión de comentarios en la parte de código con el fin de enriquecer la información por parte del creador del programa.

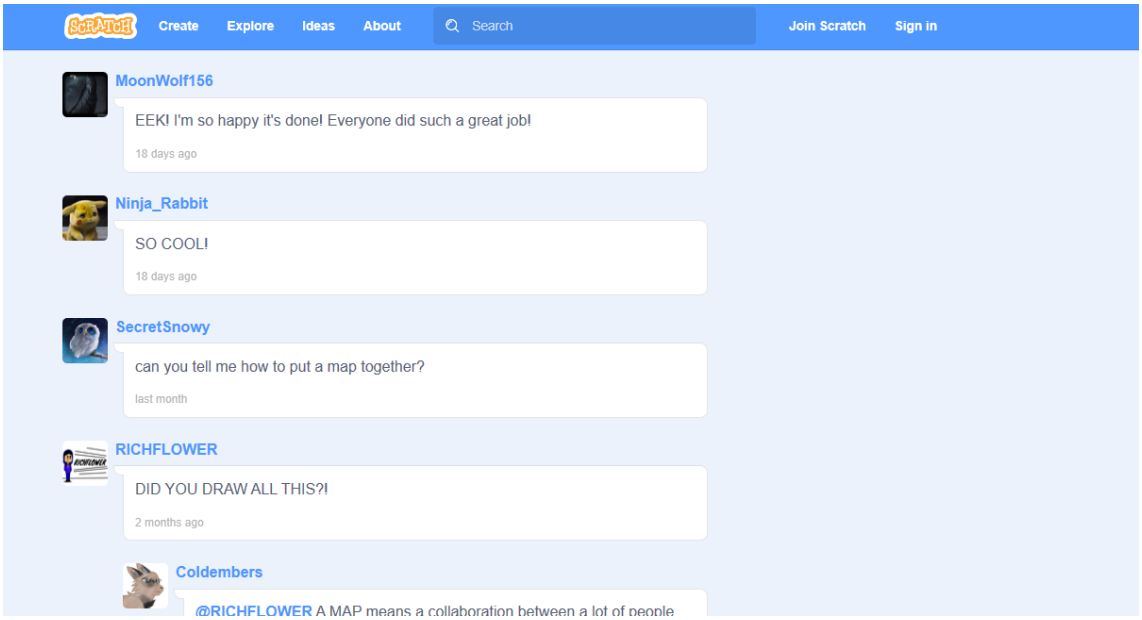


Figura 3.3: Sección de comentarios en un proyecto en la web.

3.4 Scratch y el pensamiento computacional

El pensamiento computacional proviene del término inglés *Computational Thinking* definido por Seymour Papert aunque utilizado por Jeannette M. Wing en 2006. En palabras de Jeannette se define como «*El pensamiento computacional implica resolver problemas, diseñar sistemas y comprender el comportamiento humano, haciendo uso de los conceptos fundamentales de la informática*», en otra palabras es el proceso que implica la formulación de un problema, la forma de expresar la solución en secuencia entendibles por una computadora y el posterior análisis de la ejecución de la solución y su evaluación. En [18] se puede consultar más información acerca del pensamiento computacional.

En Scratch se intenta abarcar este pensamiento computacional estableciendo conceptos como: variables, variables condicionales, operadores, secuencia de instrucciones, métodos y eventos. Por otra parte, Scratch intenta adaptar prácticas habituales en el pensamiento computacional como:

1. Divide y vencerás: ser incremental en la búsqueda de soluciones.
2. Probar y depurar: nada sale a la primera, se comentan y corrigen los errores.
3. Reusar código y modificarlo: no partir de cero con el código.
4. Abstraer, modelar y modular: crear una estructura ordenada para gestionarla con mayor facilidad.

Estas prácticas son muy comunes en los lenguajes de programación de alto nivel por lo que cuando estén dominadas pueden ser trasladadas a lenguajes más complejos ya que estas practicas se enfocan en la forma que tiene una persona de pensar con el propósito de llegar a una solución y no en la manera de programar de un lenguaje en concreto.

La manera de programar de un lenguaje en concreto se denomina sintaxis, es la estructura con la que se organizan los elementos sintácticos como espacios, operadores, variables, entre otros y cada elemento tiene establecido un orden con respecto a otros elementos.

3.5 Panel de objetos

La característica más importante que ofrece Scratch a la hora de crear programas es el panel de objetos. Este panel permite al usuario tener localizados de manera visual todos los objetos que se vayan creando a lo largo del proyecto y acceder al fragmento de código de esos objetos, esto se explicará más adelante en la sección 3.6. Gracias a esta característica los usuarios sin conocimientos de programación pueden tener en todo momento un control visual de todos los objetos creados en cada programa.

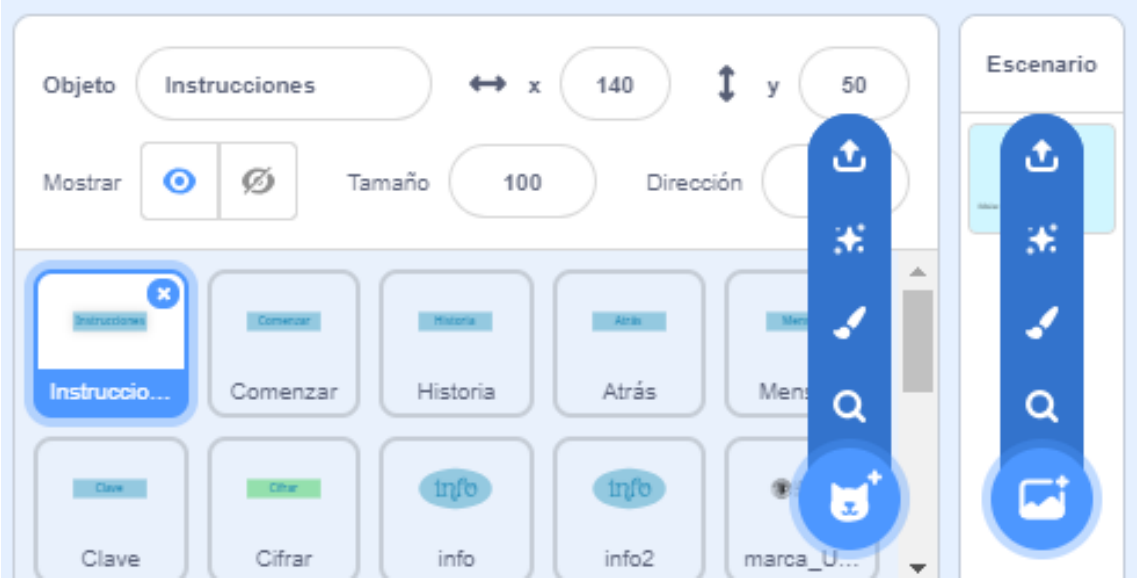


Figura 3.4: Panel de objetos donde se encuentran los objetos del programa.

En la esquina inferior derecha de la Figura 3.4 vemos cuatro iconos que nos permiten la selección de una imagen desde la biblioteca propia de Scratch, crear una imagen desde cero, elegir una imagen al azar de la biblioteca propia del programa o subir una imagen desde nuestro ordenador.

Scratch nos ofrece un editor de imágenes en el cual el usuario puede dibujar su propia imagen o importarla desde su ordenador (véase Figura 3.5). Hay que mencionar dos conceptos muy claros en el mundo de las imágenes que son el mapa de bits (*bitmap*) y la imagen vectorizada (*Scalar Vector Grafics*) (véase Figura 3.6) ya que el Scratch permite convertir la imagen importada de un tipo de imagen al otro.

- Las imágenes de mapa de bits o *bitmap* están formadas por puntos, también llamados píxeles [3], que contienen información del color que representan. El conjunto de *píxeles* ordenado de una manera concreta crea la imagen en sí. La resolución de una imagen viene dada por la cantidad de píxeles que contiene, por ejemplo, una imagen con resolución 1920x1080 quiere decir que se compone de 1920 *píxeles* de ancho y 1080 *píxeles* de alto.

  La principal ventaja que tienen este tipo de imágenes es que tienen menor tamaño en comparación con la misma imagen vectorizada y por lo tanto el tiempo a la hora de transferir la imagen también es menor. Las imágenes de este tipo se suelen guardar con extensión png

- Las imágenes vectorizadas o *Scalar Vector Grafics* están formadas por representaciones de figuras geométricas como rectángulos, círculos o segmentos. Cada figura

[3] Unidad básica de una imagen digital asociado con un color formado por colores primarios.

tiene una fórmula matemática especifica que el ordenador procesará y transformará en una imagen completa que el usuario pueda visualizar y entender.

La principal ventaja que tienen este tipo de imágenes con respecto a las vectorizadas es que al no tener una dimensión absoluta o asociada al tamaño, al aumentar o reducir la resolución de la imagen no se pierde calidad, cosa que con el otro tipo de imagen si que sucedería. Las imágenes de este tipo se suelen guardar con extensión svg

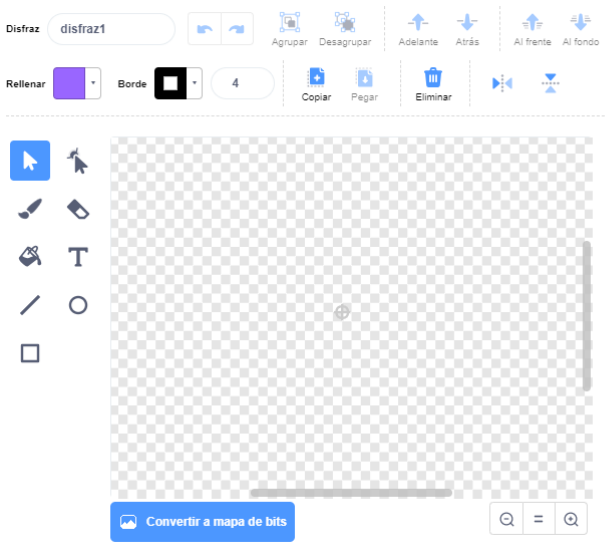


Figura 3.5: Panel editor de imágenes del programa.

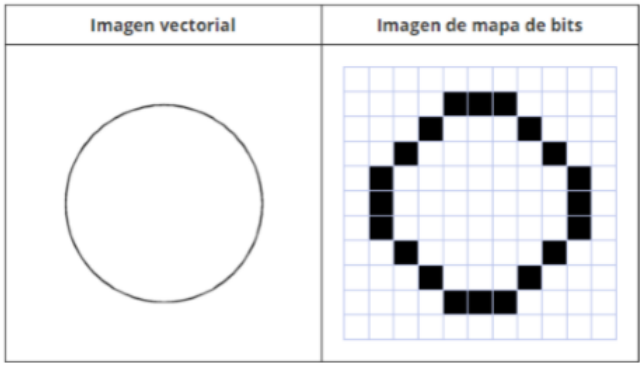


Figura 3.6: Diferencias entre imagen vectorizada e imagen de mapa de bits.

En todos los programas creados para este trabajo se han utilizado imágenes vectoriales debido a que son más fáciles de manipular y al tener un menor tamaño es mucho más fácil transferir una imagen entre distintos programas.

Por otra parte, en el panel de objetos (véase Figura 3.7) también se permite cambiar el nombre al objeto en cuestión, ver y cambiar en la posición que se encuentra, también permite mostrar o ocultar del objeto y cambiarle el tamaño y el ángulo del objeto.

Por último, en la parte derecha del panel de objetos encontramos el objeto que Scratch define por defecto denominado «Escenario», desde donde se maneja todo lo que ocurre en el programa y es el encargado de establecer el fondo del mismo. Una función que se le suele encargar de establecer las variables de inicialización [4] del programa, en nuestro

[4] Valores que se establecen al ejecutar un programa.

caso es donde se establece todo el código importante como es la parte de cifrado o de descifrado.

Figura 3.7: Información sobre un objeto.

3.6 Panel de programas

Cada objeto en Scratch posee su propio panel de código o *scripts* y en este panel es donde los usuarios van ir creando la funcionalidad del programa. Una característica del Scratch es que para programar no se escriben lineas de código sino que se van arrastrando bloques, predefinidos por el lenguaje, simulando las características básicas de un lenguaje de programación. Estos bloques son similares a las piezas de *LEGO*.

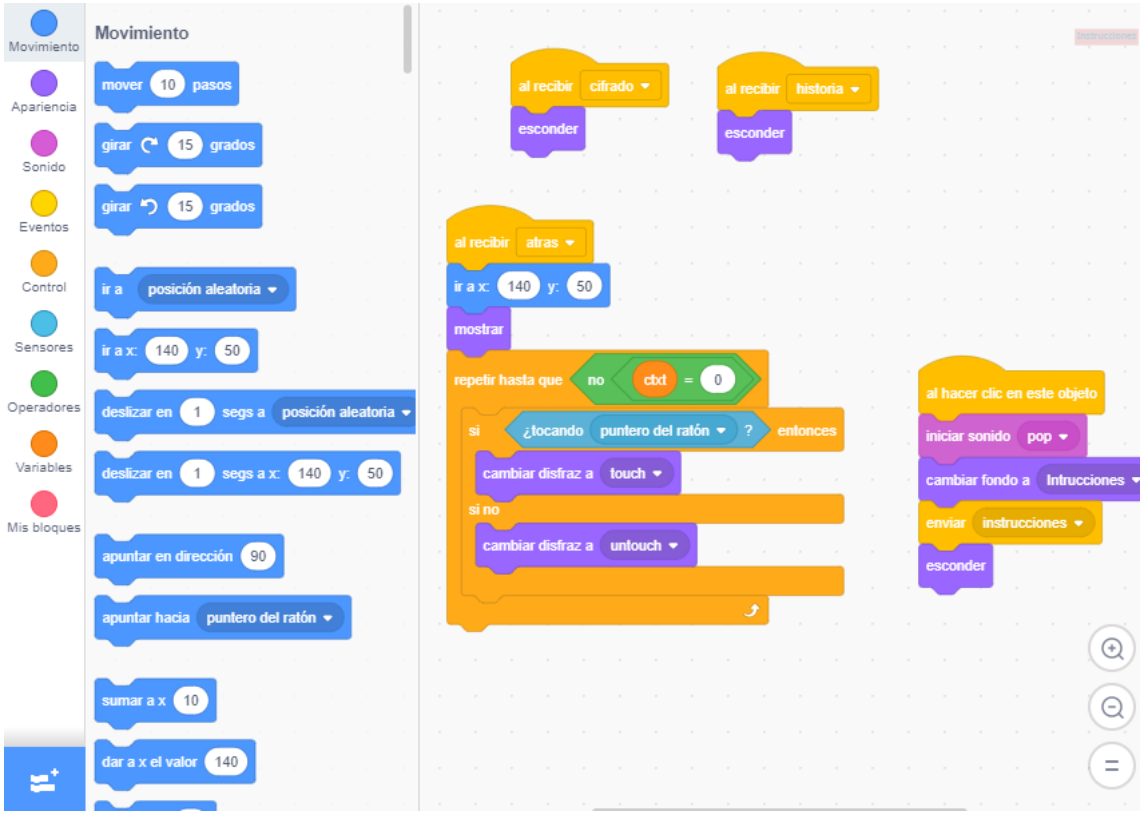


Figura 3.8: Panel de programas donde se muestra el código de un objeto.

En la parte izquierda de la Figura 3.8 se pueden ver todos los tipos de bloques que ofrece el lenguaje de programación Scratch. A continuación se describirán brevemente:

- **Bloque de Movimiento.** Son los bloques responsables de desplazar, detectar y fijar en qué posición se encuentra un objeto dentro del escenario del programa. Habi-

tualmente una vez arranca el programa se suele fijar los objetos en una posición concreta.

- **Bloque de Apariencia.** Estos bloques son los encargados de poder modificar la apariencia de un objeto. Algunos bloques son mostrar un mensaje, fijar el objeto a un tamaño, cambiar el disfraz del objeto, esconder o mostrar el objeto, entre muchos otros.
- **Bloque de Sonido.** Gracias a estos bloques se puede modificar el sonido del programa. Se puede iniciar o detener un sonido, añadir algún efecto al sonido y modificar el volumen del sonido. Cabe destacar que se pueden agregar tanto sonidos de la propia biblioteca del programa como importar nuestros propios sonidos desde nuestro ordenador.
- **Bloque de Eventos.** Estos bloques son los encargados de comunicar objetos distintos en el programa gracias al envío y recepción de mensajes (o señales). Cuando se envía un mensaje existe un objeto que la genera y otro u otros objetos que están esperando ese mensaje para realizar una tarea asociada a la recepción del mensaje, como por ejemplo, hacer clic en el objeto, recibir un mensaje especifico, cuando el fondo cambie a uno especifico, entre otros.
- **Bloque de Control.** Estos bloques son los encargados de proporcionar las instrucciones de iteración y condición, se pueden generar fragmentos de código que incluyan una condición, bucles o también fragmentos de espera, en los que la ejecución se esperará un tiempo fijado por el usuario o hasta que se cumpla una condición específica. Una peculiar funcionalidad de Scratch es que permite la creación de crear clones, o copias, de un objeto.
- **Bloque de Sensores.** Son los bloques que detectan ciertas situaciones en objeto. La situaciones son detectar cuando el cursor esta tocando el botón, cuando un objeto colisiona contra otro, detectar cuando una tecla esta siendo pulsada, activar un cronómetro, etc.
- **Bloque de Operadores.** Gracias a estos bloques se pueden utilizar los operadores matemáticos básicos como la suma, la resta, la multiplicación o la división, también podemos usar los operadores de comparación. Con estos bloques también se pueden hacer operaciones con las variables de caracteres como unir, obtener un carácter o buscar un carácter en una variable.
- **Bloque de Variables.** Estos bloques proporcionan al usuario la opción de crear variables simples o listas (denominados *arrays* en inglés). Scratch permite la opción de crear variables en modo local[5] o global[6], de manera que si son locales solo pueden ser editadas por ese mismo objeto pero si son globales pueden ser editadas por cualquier objeto del programa.
- **Bloque de Mis bloques.** Scratch gracias a este bloque permite al usuario la creación de bloques propios formados con los bloques anteriormente mencionados (véase Figura 3.9). Esta característica intenta simular a los llamados métodos que poseen los lenguajes de más alto nivel.
- **Extensiones.** Scratch permite agregar más bloques al programa para añadirle funcionalidad, estos bloques o extensiones pueden ser desde propios del programa como sensor de vídeo, música o lápiz o externos que permite agregar dispositivos físicos como *micro:bit* o *LEGO WeDo 2.0*.

[5] Solo para este objeto.
[6] Para todos los objetos.

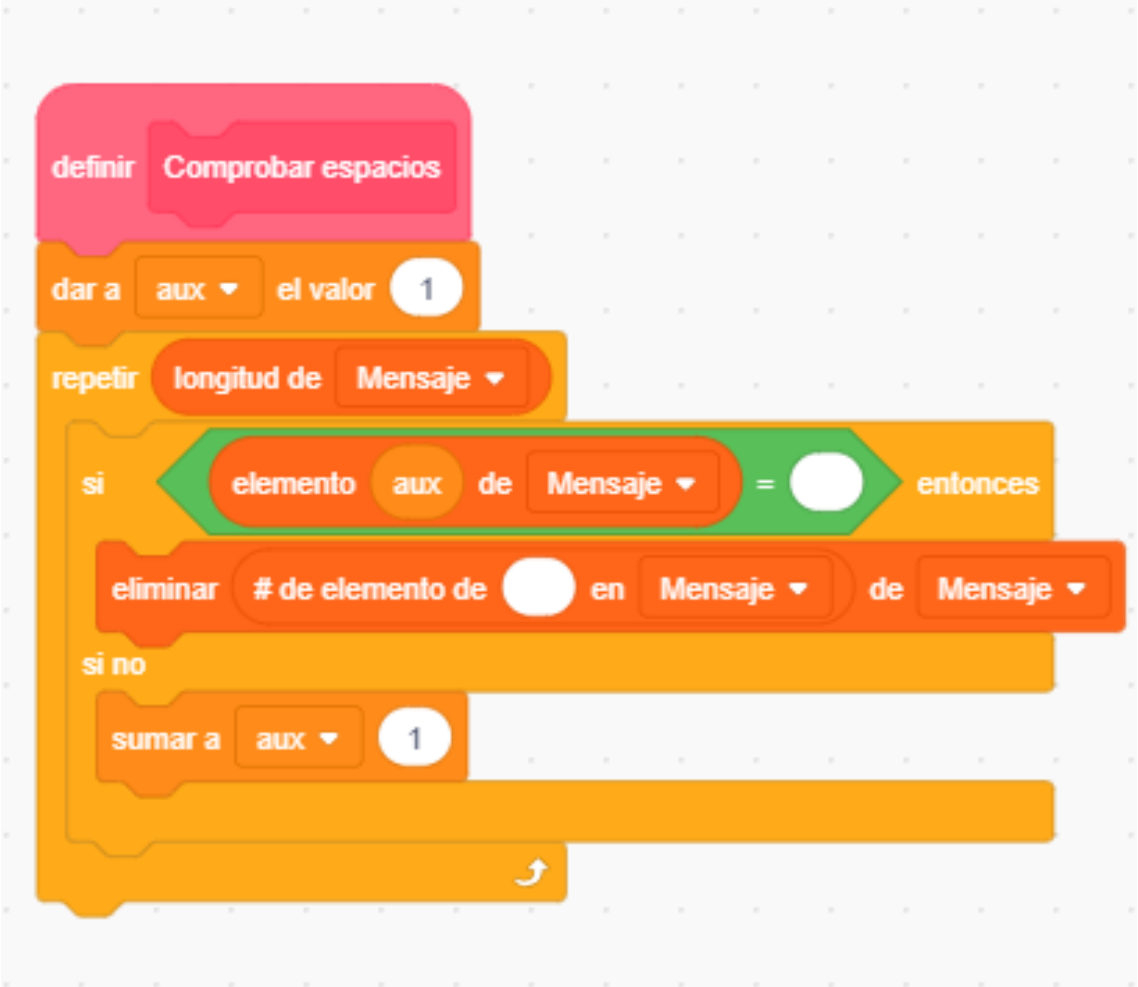


Figura 3.9: Bloque creado en el programa para eliminar espacios en el mensaje.

3.7 Panel de disfraces

En el panel de disfraces el usuario puede gestionar las diferentes apariencias que puede tener un objeto en nuestro programa (véase Figura 3.10). Scratch para representar la animación de los objetos cambia de manera sucesiva los disfraces de ese objeto y así darle una sensación de movimiento del objeto al usuario.

3.8 Panel de sonidos

Este es el último panel que se describirá, al igual que pasa con el panel de programas y de disfraces, cada objeto que se este creando también posee un panel de sonidos (véase Figura 3.11). La labor de este panel consiste en gestionar los distintos sonidos que cada objeto reproduce al ejecutar el programa.

Cuando se importa un sonido ya sea de la biblioteca propia del Scratch o de nuestro ordenador Scratch admite múltiples formatos de audio como *wav*[7] o *mp3*[8]. Scratch ofrece la posibilidad de modificar el audio, añadir efectos o incluso grabar audio con nuestro propio micrófono. Todos los sonidos serán controlados con los bloques de sonido del panel de programas.

[7]Formato de audio sin compresión de datos.
[8]Formato de audio cuyos datos se encuentra comprimidos.

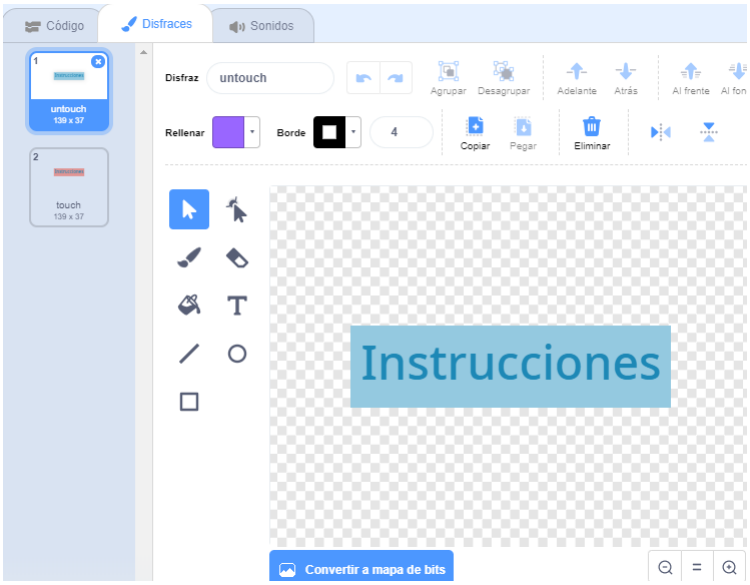


Figura 3.10: Panel de disfraces del objeto «Instrucciones».

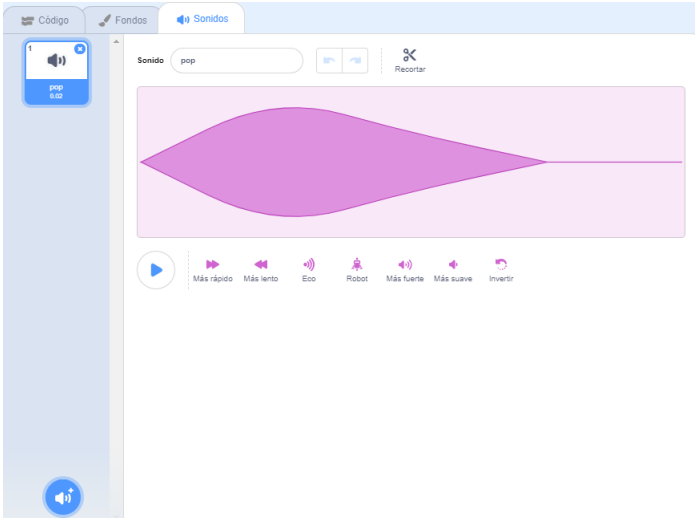


Figura 3.11: Panel de sonido donde se pueden importar y modificar sonidos.

3.9 Metodología en el desarrollo de los cifrados

En esta sección se explicarán los pasos seguidos para realizar los cuatro cifrados que trata el presente trabajo (Disco de Alberti, Vigenère, Playfair y ADFGVX). Se comenzará explicando el motivo de la elección de los cifrados hasta la corrección de los fallos una vez implementados.

En primer lugar se ha investigado que cifrados clásicos de la historia se querían implementar en este trabajo. En trabajos previos ya se había realizado la implementación de cifrados clásicos [11] por lo que esos cifrados se descartaron. Se querían cifrados que se pudieran explicar de manera didáctica a los jóvenes y fueran fáciles de entender, también tenía que haber variedad en los tipos de cifrados que seleccionamos, en otras palabras, que fueran tanto monoalfabéticos como polialfabéticos.

El siguiente paso es la recopilación de toda la información posible, de las características, de la complejidad y de las dificultades de los cifrados que pudiesen surgir al intentar implementarlo en el lenguaje de programación Scratch. Antes de implementar los cifra-

dos se ha de aprender a utilizar el lenguaje aunque no ha sido especialmente difícil al tratarse de un lenguaje por bloques, del cual existe gran cantidad de documentación en la propia página del lenguaje y tutoriales que explican el funcionamiento del mismo. Cabe destacar que al usar este lenguaje en cursos anteriores solo ha hecho falta una pequeña revisión del funcionamiento debido al cambio de versión en el programa.

El proceso de recopilación de información ha sido laborioso debido a que era necesario investigar de cada cifrado que peculiaridades presentaba tanto en el momento de cifrar el mensaje como a la hora de descifrar el mensaje y a causa de las limitaciones de Scratch buscar la manera de trasladar dichas peculiaridades cuando se va a implementar cada cifrado.

A la hora de diseñar los objetos que dan lugar a los cifrados, se ha tenido que empezar la creación desde cero debido a que no existían imágenes con una resolución aceptable o a la hora de importarlas no se adaptaban correctamente al programa. Otro factor es que en la biblioteca propia del Scratch no había ninguna imagen que se adecuara al entorno del cifrado. Todas la imágenes creadas son imágenes vectorizadas dado que ofrecen una gran calidad en la imagen, por ejemplo, para desarrollar la imagen del Disco de Alberti se ha necesitado un programa externo que permitiera la creación y edición de imágenes (vease Figura 3.12).

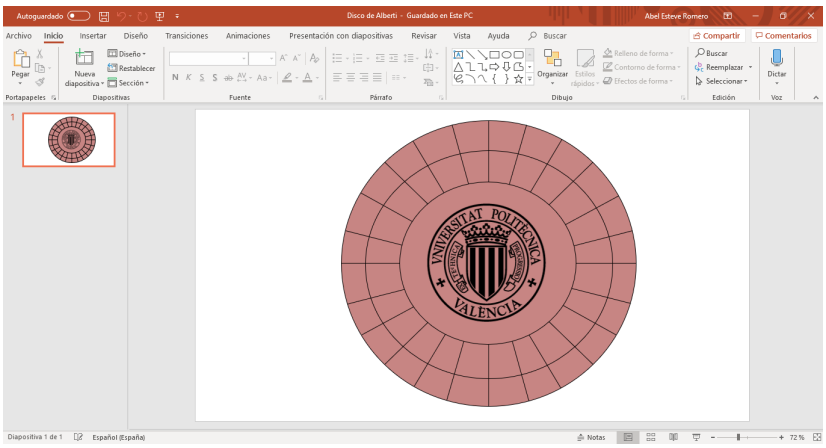


Figura 3.12: Creación del Disco de Alberti en el programa *Microsoft PowerPoint*.

El objetivo principal de la programación en Scratch es conseguir que en cada cifrado el mensaje introducido por el usuario se cifre y se descifre correctamente, por ello antes de diseñar ningún entorno gráfico concreto se ha empezado a trabajar únicamente con las variables para la funcionalidad correcta de cada cifrado (véase Figura 3.13). Una vez el cifrado esta correcto se ha pasado a la creación de cada imagen que conformará el entorno gráfico que el usuario podrá ver.

Se ha pretendido que la pantalla inicial de todos los cifrados sea igual, para que el usuario se sienta cómodo al usar al probar los programas pero debido a la complejidad del algún cifrado se ha debido de cambiar. En tres de nuestros cifrados el menú inicial presenta cuatro botones «Instrucciones», «Cifrado», «Descifrado» e «Historia», en el cuarto cifrado los botones «Cifrado» y «Descifrado» se han sustituido por el botón «Comenzar» junto al logo del Museo de Informática, de la Escuela Técnica Superior de Ingeniería Informática y de la Universitat Politècnica de València (véase Figura 3.14).

Los cuatro cifrados presentan el mismo esquema gráfico tanto de forma visual como de manera interna, ya que de esta manera se ahorra tiempo a la hora de implementar todos los objetos. Los objetos del menú principal presentan las mismas características en todos los cifrados, al pasar el cursor por encima de ellos el disfraz cambia de color

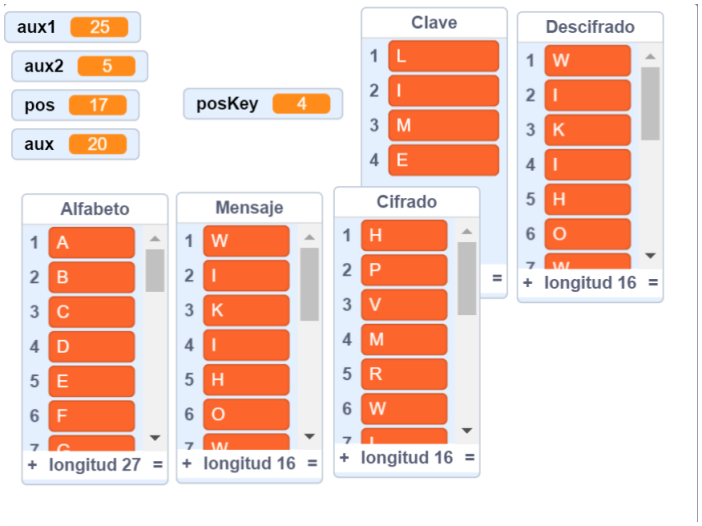


Figura 3.13: Captura del programa antes del implementar el entrono gráfico.

Figura 3.14: Pantalla inicial del cifrado de Vigènere.

mostrando así que se trata de un botón de interacción, los botones dentro del menú de cifrado/descifrado también cambiarán a otro color distinto para mostrar que se trata para la inserción de variables. Hay que comentar que han implementado botones cuya función es mostrar una pequeña información sobre que se debe hacer en cada caso cuando el usuario pasa el cursor por encima del botón (véase Figura 3.15).

Para finalizar, cuando ya está todo el programa diseñado e implementado se consideran una serie de pruebas que debe pasar para corregir cualquier problema o fallo que pueda surgir y así se asegurar que los programas sean robustos antes de ser publicados en la web del Museo de Informática.

3.10 Otros detalles

Cabe señalar un detalle a la hora de implementar en Scratch, debido a ser un lenguaje bastante limitado en comparación con lenguajes de alto nivel como Java, el tratamiento tanto de caracteres en mayúsculas y en minúsculas como el de caracteres con tilde puede

resultar muy fastidioso y por ello se ha decidido que este hecho sea tratado en posibles ampliaciones de los programas.

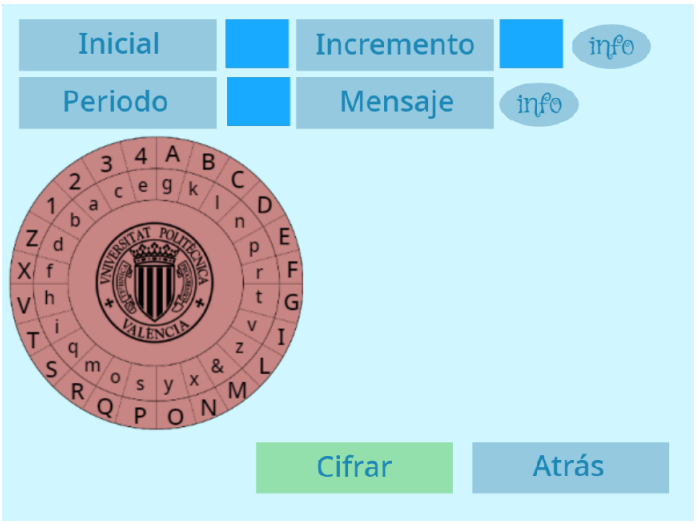


Figura 3.15: Captura del Disco de Alberti con el entorno gráfico.

Por otra parte, los caracteres especiales como @, #, %, & entre otros, tampoco serán tratados ya que podrían dar algún problema cuya solución supondría una dificultad extra, solo nos ceñiremos a utilizar los caracteres en mayúsculas y los números enteros reales dado que el objetivo de este trabajo es mostrar los cifrados al público de la manera más didáctica. Para avisar de ello al usuario se ha implementado un cartel que muestra dicha información (véase Figura 3.16).

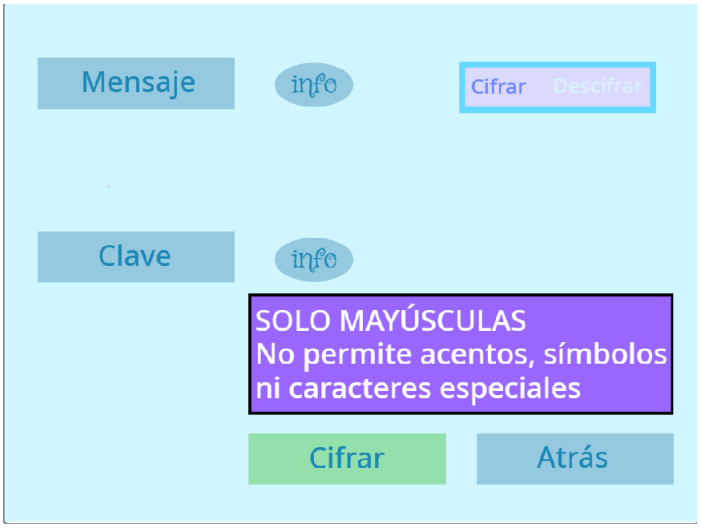


Figura 3.16: Mensaje de información en el cifrado Vigenère.

CAPÍTULO 4

Diseño e implementación del cifrado Alberti en Scratch

El primer cifrado que se va ha describir es el cifrado Alberti. Como se ha mencionado en el capítulo 2 se trata de un cifrado polialfabético porque un mismo carácter se puede cifrar de distintas maneras en función de un criterio, en este caso en función de unos parámetros. En este capítulo se describirá detalladamente toda la implementación del cifrado en Scratch.

4.1 Organización del menú principal

Dentro del programa en cuestión nos encontramos un total de 26 objetos entre los que se encuentran tanto objetos simples como complejos.

El menú principal que nos encontramos al acceder al programa se compone de cuatro botones con los que accederemos a las diferente secciones del programa como «Instrucciones», «Cifrado», «Descifrado» o «Historia» (véase Figura 4.1). Para indicar al usuario que estos botones son de interacción al pasar el cursor por encima por alguno de ellos el disfraz se cambiará. A continuación se mostrará la funcionalidad de estos botones.

Figura 4.1: Menú principal del cifrado de Alberti.

- **Instrucciones.** Al pulsar en este botón el usuario cambiará de sección y se le mostrará una breve explicación sobre cómo funciona el cifrado.

- **Cifrado.** Este botón junto al de «Descifrado» es el más importante ya que ofrece funcionalidad al programa. Al pulsar en el botón el valor de la variable «ctxt» cambiará a *1* indicando al programa que esta en el escenario del cifrado (véase Figura 4.2), también cambiará el fondo al fondo llamado «cifrado» en el cual no aparece ningún texto en el fondo y enviará al programa el mensaje «cifrado».

  Cuando el escenario ha cambiado el usuario podrá introducir los parámetros y el mensaje correspondientes para realizar el cifrado. Cabe destacar que han sido implementados mensajes de advertencia para evitar que el usuario cometa errores innecesarios. Por ejemplo si pasamos con el cursor sobre la etiqueta «info» de al lado del botón «Mensaje» o del botón «Incremento» se mostrará un mensaje que indica que la longitud máxima del mensaje debe ser de 15 caracteres en el caso del mensaje o que la longitud máxima de los parámetros debe ser la indicada.

- **Descifrado.** Cuando el usuario presiona el botón el valor de la variable «ctxt» cambiará a *2* indicando al programa que se encuentra en el escenario del descifrado, también cambiará el fondo al fondo llamado «cifrado» en el cual no aparece ningún texto en el fondo y enviará al programa el mensaje «cifrado».

  Al cambiar de escenario se muestra uno similar al escenario de cifrado pero con la única diferencia es que el botón inferior es «Descifrar» en vez de «Cifrar». Se ha diseñado este escenario de tal manera para que el usuario se sienta familiarizado con el escenario ya que ha sido presentado anteriormente.

- **Historia.** Al pulsar en este botón el usuario cambiará de sección y se le mostrará un breve contexto histórico sobre el cifrado.

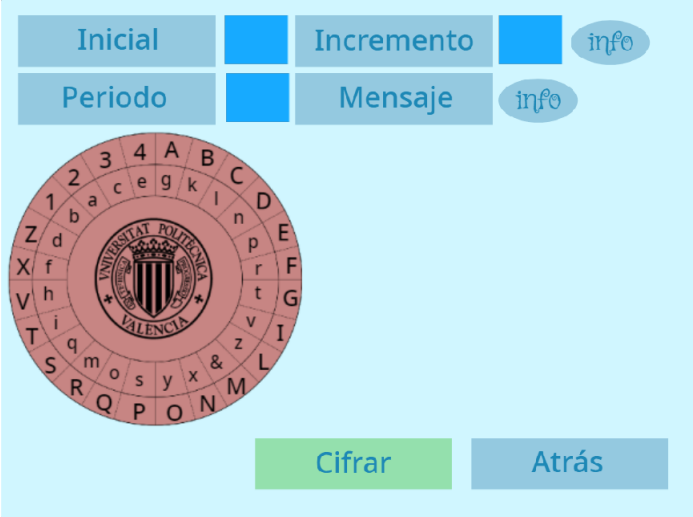


Figura 4.2: Pantalla principal del cifrado de Alberti.

4.2 Objetos del cifrado

En este apartado se hablará de los objetos que aportan funcionalidad al cifrado así como la manera en la que están implementados en el entorno Scratch mostrando un fragmento de código asociado a los distintos objetos. De los 26 objetos que componen el programa solo 7 son los más importantes para el cifrado y que aportan funcionalidad, se describirán a continuación.

1. **Disco de Alberti.** Este objeto no tiene funcionalidad más bien es solo demostrativo, muestra al usuario el disco de Alberti, cada vez que el usuario decide rotar el disco introduciendo un parámetro inicial o al pulsar en cifrar/descifrar rota el disco interior mostrando así la rotación al usuario. Se puede observar como es la apariencia del disco en la Figura 4.2.

   La implementación es muy simple (véase Figura 4.7), al tratarse de un objeto con diferentes disfraces solo hace falta saber cuando se cambia el disfraz del objeto para simular la rotación del disco. Tenemos tanto el alfabeto exterior como el alfabeto interior guardado en respectivas listas, cuando el usuario introduce inicial o se cumple el periodo el alfabeto interior se modifica cambiado la posición de sus caracteres y se envía el mensaje «disco». Cuando el disco reciba este mensaje cambiará el disfraz mostrando la rotación del alfabeto interior al usuario .

2. **Inicial.** La funcionalidad de este botón es permitir al usuario introducir el parámetro inicial que hará que el disco interior del disco de Alberti rote tantos caracteres como parámetro haya introducido el usuario.

   De la manera que se ha implementado este objeto, al pulsar en este botón aparece un cuadro de texto que permite al usuario introducir el parámetro inicial, este parámetro se guardará en la variable temporal «respuesta» y dicha variable se volverá a guardar en «inicial». Ahora el programa comprobará que inicial se encuentra entre el *-1* y el *24*, permitiendo el 0 aunque no surgiría efecto en la rotación, si el parámetro esta en dicho rango se enviará el mensaje «inicial» al programa y si el parámetro no estuviera dentro de ese rango enviará el mensaje «warning_mes» (véase Figura 4.8).

   El programa al recibir el mensaje «inicial» cambiaría el disfraz del cuadro adyacente al objeto «Inicial» al parámetro guardado en la variable «inicial», después enviará en mensaje «rotar» que hará que el alfabeto interno cambie de posición sus caracteres e inmediatamente después el disco cambie de disfraz a este nuevo alfabeto.

   El programa al recibir el mensaje «warning_mes» muestra por pantalla el mensaje de advertencia «Introducir Mensaje o Parámetros válidos (Pulse en info para saber más)» indicando que algún parámetro o el mensaje introducido no es válido (véase Figura 4.3).

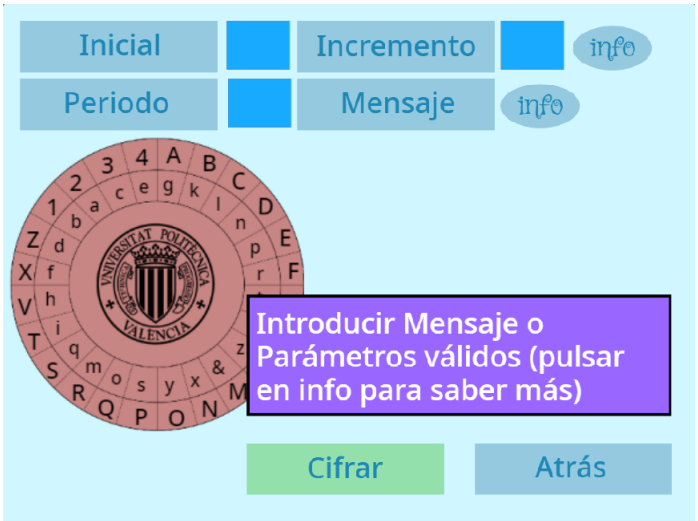


Figura 4.3: Mensaje de advertencia sobre el mensaje y los parámetros.

3. **Incremento.** La funcionalidad de este objeto es posibilitar al usuario introducir el parámetro incremento. Una vez introducido el parámetro el disco interior rotará

tantos caracteres como parámetro haya introducido el usuario cada vez que se cumpla el periodo.

Este botón se ha implementado de tal forma que al pulsar en este botón aparece un cuadro de texto que permite al usuario introducir el parámetro incremento, este parámetro se guardará en la variable temporal «respuesta» y dicha variable se volverá a guardar en «incremento». Ahora el programa comprobará que incremento se encuentra dentro del rango entre el *-1* y el *25*, permitiendo el 0 aunque no surgiría efecto en la rotación, si se cumple el rango se enviará el mensaje «incremento» y si el parámetro no estuviera dentro de ese rango enviará el mensaje «warning_mes» (véase Figura 4.9).

El programa al recibir el mensaje «incremento» cambiaría el disfraz del cuadro contiguo al objeto «incremento» al parámetro guardado en la variable «incremento». Dicha variable se utilizará más adelante a la hora de cifrar.

4. **Periodo.** Este botón realiza la función de permitir al usuario introducir el parámetro periodo. Cada vez que se cumpla ese periodo en el mensaje, el disco de Alberti rotará tantas veces como indica el parámetro incremento.

   Este botón se ha implementado tal que al pulsar en este botón aparece un cuadro de texto que permite al usuario introducir el parámetro periodo, este parámetro se guardará en la variable temporal «respuesta» y dicha variable se volverá a guardar en «periodo». Ahora el programa comprobará que la longitud de periodo se encuentra entre el *0* y el *15*, si es así enviará el mensaje «periodo» y si el parámetro no estuviera dentro de ese rango enviará el mensaje «warning_mes» (véase Figura 4.4).

   El programa al recibir el mensaje «periodo» cambiaría el disfraz del cuadro inmediato al objeto «Periodo» al parámetro guardado en la variable «periodo». Dicha variable se utilizará más adelante a la hora de cifrar.

5. **Mensaje.** Este objeto tiene la utilidad de permitir al usuario introducir el mensaje para cifrar o descifrar.

   La implementación es similar a la de los otros objetos (véase Figura 4.5), cuando el usuario pulsa en el objeto aparece un cuadro de texto que permite al usuario introducir el mensaje, se comprueba que el mensaje no contenga ningún símbolo o carácter especial si es así el mensaje se guardará en la variable temporal «respuesta» y dicha variable se guardará en la lista «Mensaje» pero si no mostrará el mensaje de advertencia de la Figura 3.16. En este punto se comprueba que no tengamos que hacer ningún cambio en los caracteres y si así fuera se utiliza la tabla 2.1 para las equivalencias. Cuando las caracteres están comprobados se verifica que la longitud de la lista «Mensaje» este dentro del rango entro *0* Y *15* caracteres, si es así se crearán tantos clones de «Char_mensaje» como caracteres tenga la lista «Mensaje» para mostrar los caracteres en pantalla, pero si el mensaje no estuviera en el rango se enviaría el mensaje «warning_mens».

6. **Cifrar/Descifrar.** La función de este objeto es permitir al usuario cifrar o descifrar el mensaje con los parámetros incremento y periodo.

   La parte de la implementación de este botón se divide en 3 partes (véase Figura 4.10), al decidir pulsar el botón, al decidir cifrar y al decidir descifrar. Cuando se pulsa el botón se comprueba que el mensaje no este vacío y que los parámetros inicial, incremento y periodo no estén vacíos y estén dentro del rango anteriormente comentados, si no así se enviará el mensaje «warning_mens» y detendrá el código del objeto pero si están dentro del rango y la comprobación es correcta se elimina los caracteres en pantalla. A continuación se comprueba el valor de la variable «ctxt»

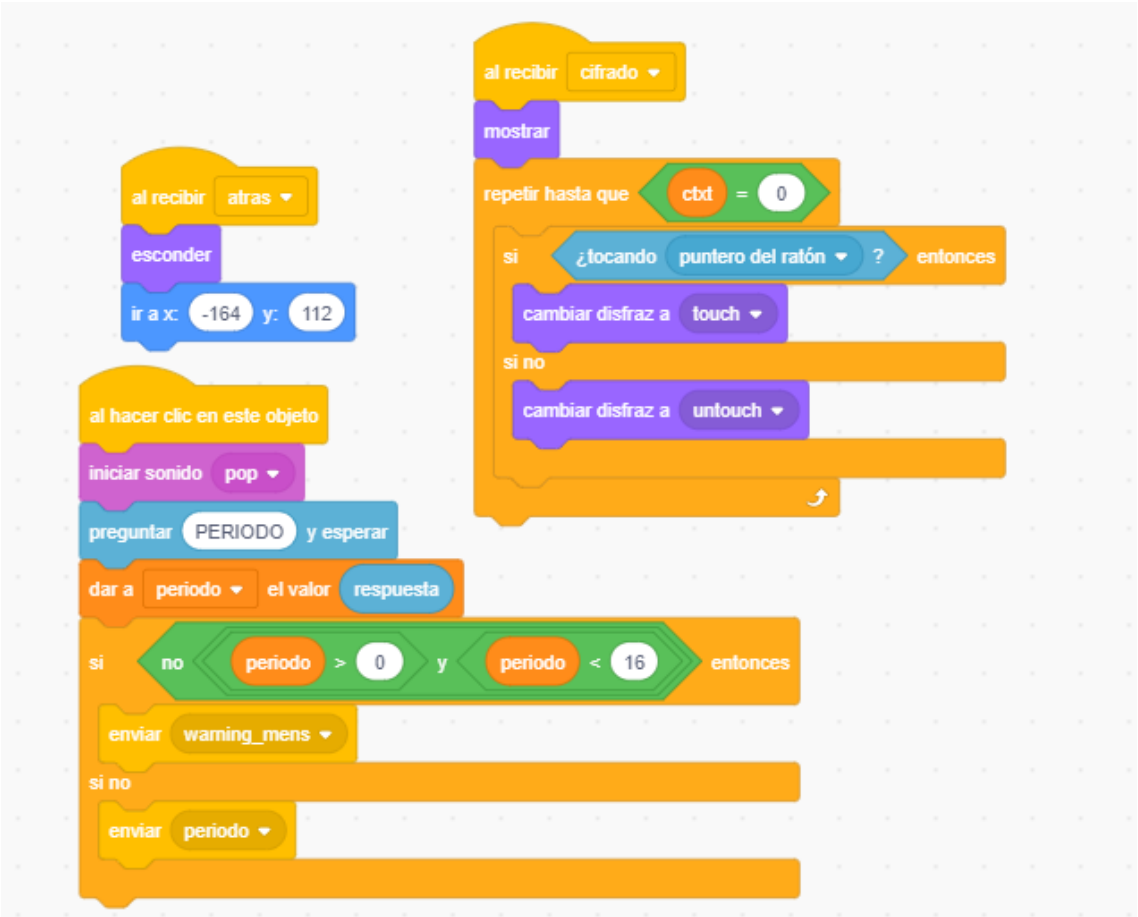


Figura 4.4: Fragmento de código del botón Periodo.

para saber si se trata del cifrado o del descifrado y se cambia al disfraz adecuado, si la variable «ctxt» es igual a *0* se enviará el mensaje «cifrar» al programa pero si la variable «ctxt» es igual a *1* se enviará el mensaje «descifrar» al programa.

El programa al recibir el mensaje «cifrar» elimina todos los caracteres de la lista «Cifrado», se comprobarán carácter a carácter el mensaje y se irá sustituyendo en relación al alfabeto interno del disco de Alberti, repitiendo esto cada periodo y rotando el disco en cada incremento. Este nuevo mensaje se irá guardando en la lista «Cifrado», cuando todo el mensaje este sustituido se crearán tantos clones de «Char_cifrado» como caracteres tenga la lista «Mensaje» para mostrar los caracteres del mensaje cifrado por pantalla (véase Figura 4.11).

El programa al recibir el mensaje «descifrar» el programa elimina todos los caracteres de la lista «Cifrado», se comprobarán carácter a carácter el mensaje y se irá sustituyendo en relación al alfabeto externo del disco de Alberti, repitiendo esto cada periodo y rotando el disco en cada incremento. Este nuevo mensaje se irá guardando en la lista «Cifrado», cuando todo el mensaje sustituido se crearán tantos clones de «Char_cifrado» como caracteres tenga la lista «Mensaje» para mostrar los caracteres del mensaje cifrado por pantalla (véase Figura 4.12).

7. **Atrás.** La utilidad de de este objeto es permitir al usuario volver al menú principal tanto cuando el usuario quiere introducir un mensaje para cifrar o para descifrar, como después de leer las instrucciones o la historia del cifrado.

   Al tratarse de un botón que se encuentra en diferentes escenarios la implementación se podría dividir en diferentes partes. Cuando se pulsa en el objeto se cambia el

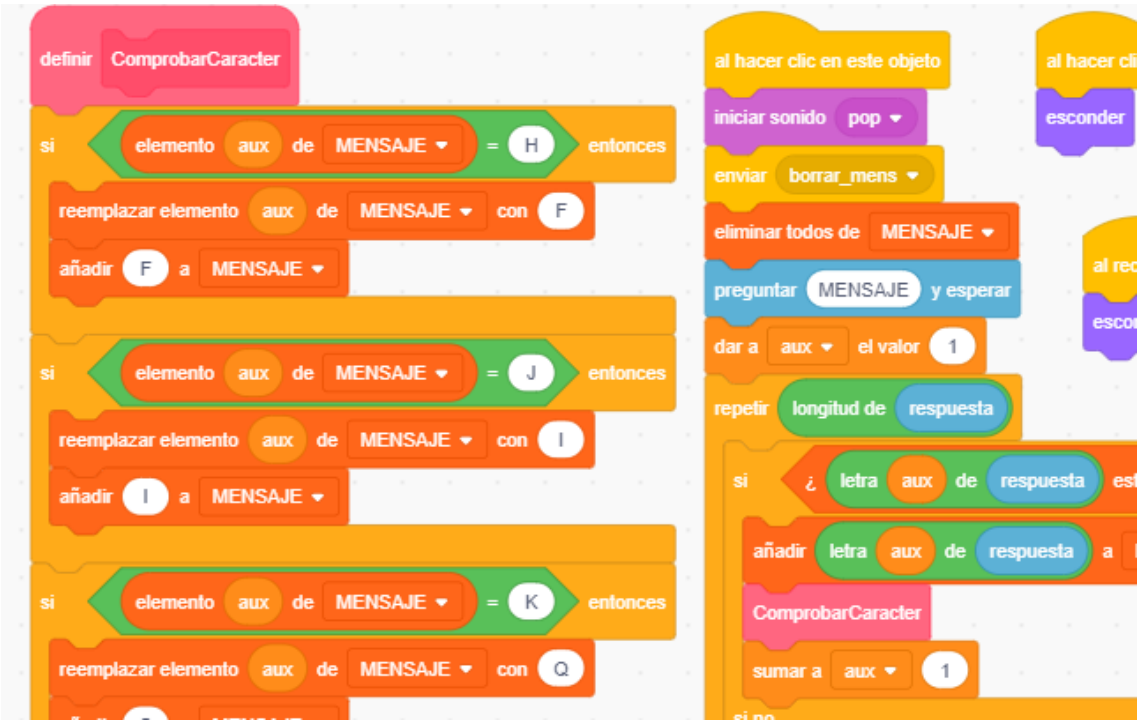


Figura 4.5: Fragmento de código del botón Mensaje.

fondo a principal, la variable «ctxt» cambia al valor *0* y se envía el mensaje «atrás» al programa retornando así al usuario al menú principal.

Cuando el programa recibe el mensaje «atrás» este objeto se oculta al usuario, si recibe el mensaje «historia», «instrucciones» o «cifrado» y dependiendo de la variable «ctxt» tiene un bucle para que cada vez que se pase el cursor por encima del objeto este cambie de disfraz mostrando así que se trata de un objeto de interacción (véase Figura 4.6).

4.3 Posibles ampliaciones

En este apartado se explicarán algunas de las posibles ampliaciones que se pueden realizar al programa para la mejora del cifrado. Una de las ampliaciones es poder introducir tanto mayúsculas como minúsculas en el mensaje así poder implementar otra versión del cifrado en la que no se necesita parámetros, en esta versión del cifrado solo haría falta comprobar la mayúsculas para rotar el disco. Otra ampliación podría ser aumentar el rango del mensaje para que el usuario pudiese introducir un mensaje más largo y que al cifrar o descifrar mostrará el proceso de cifrado. Una última ampliación, que cambiaría el cifrado, sería permitir al usuario cambiar el orden del alfabeto del disco pudiendo así poder cifrar de manera totalmente distinta el mismo mensaje.

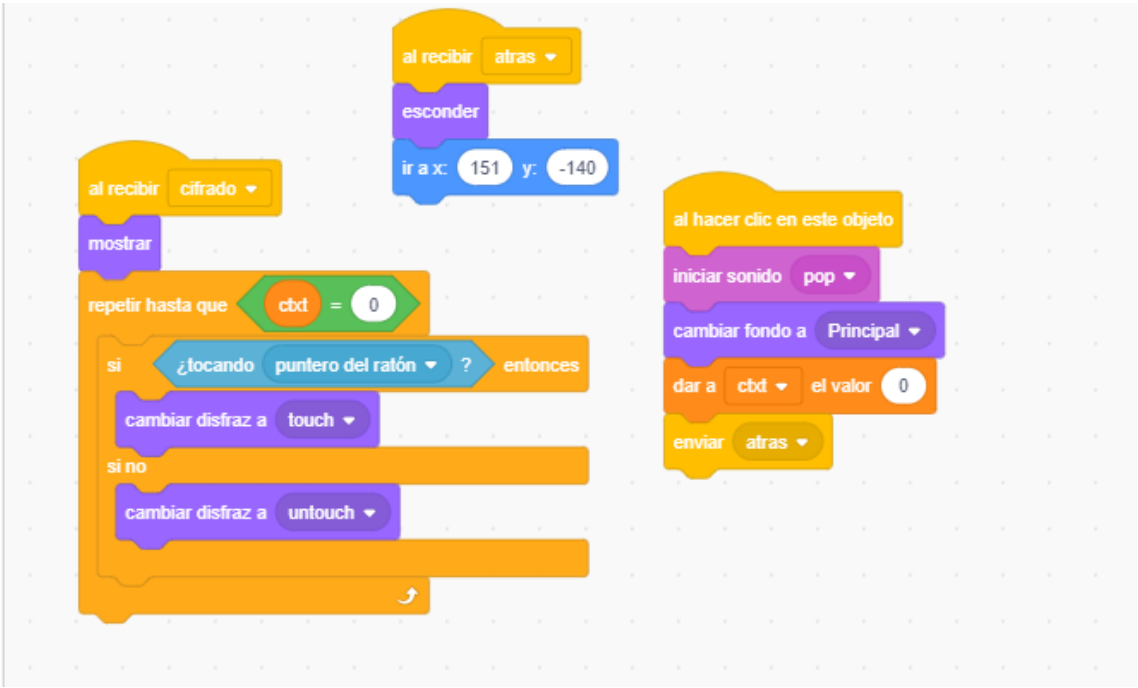


Figura 4.6: Fragmento de código del botón Atrás.

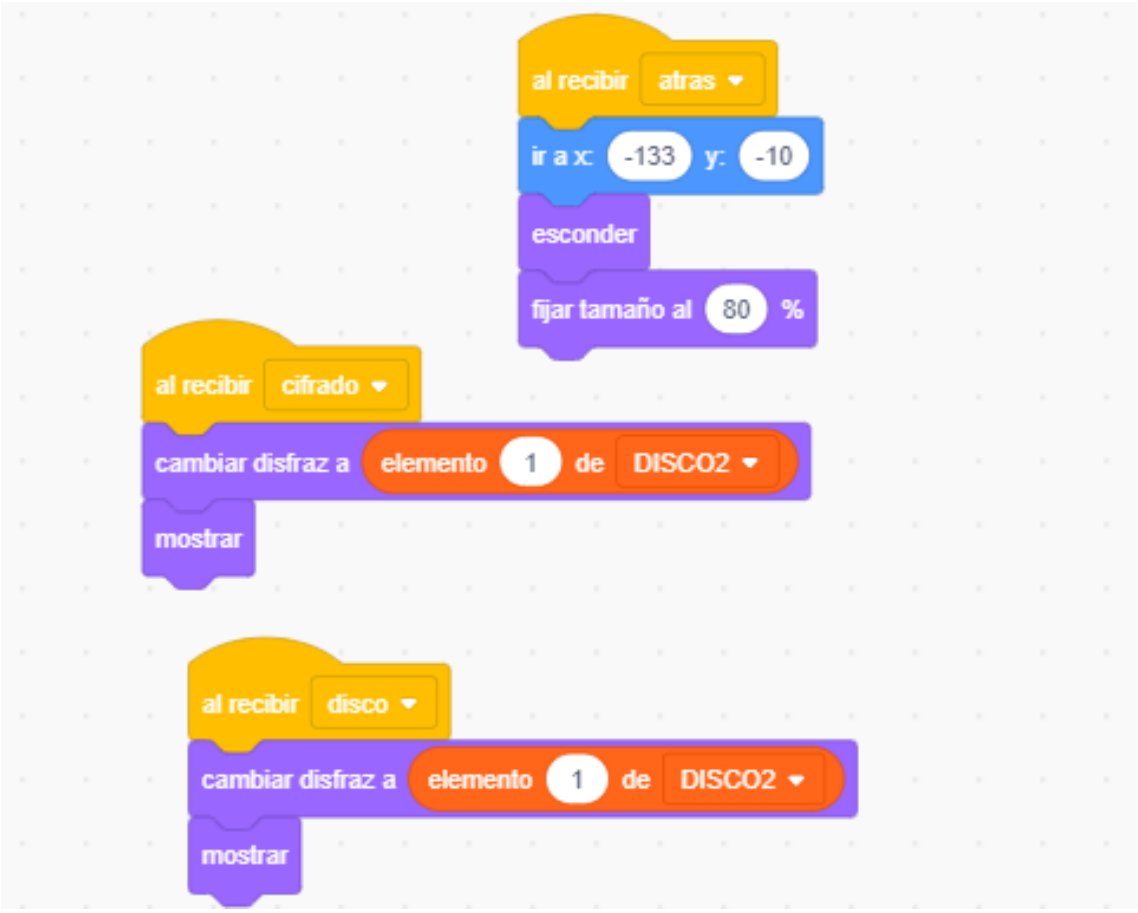


Figura 4.7: Fragmento de código del disco de Alberti.

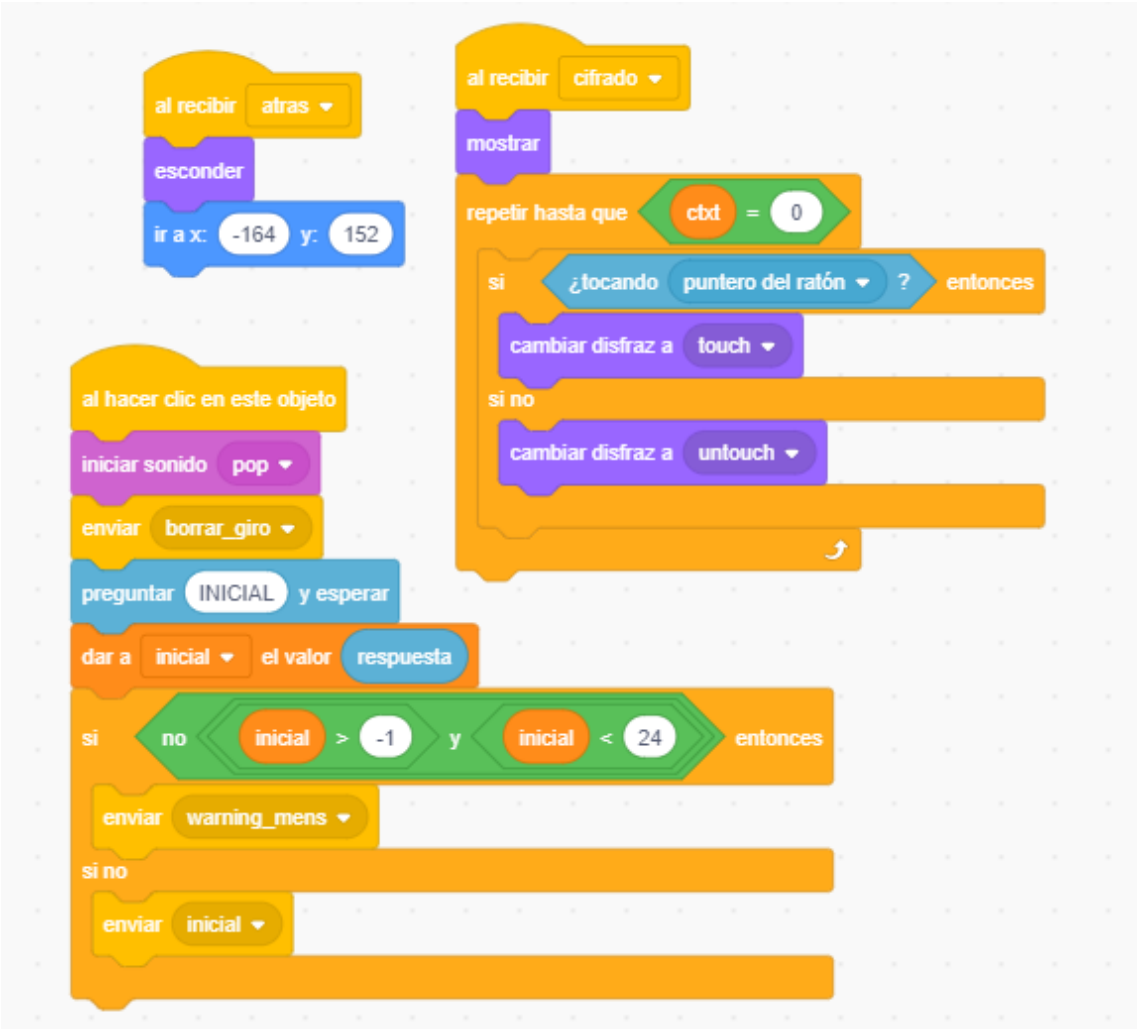


Figura 4.8: Fragmento de código del botón Inicial.

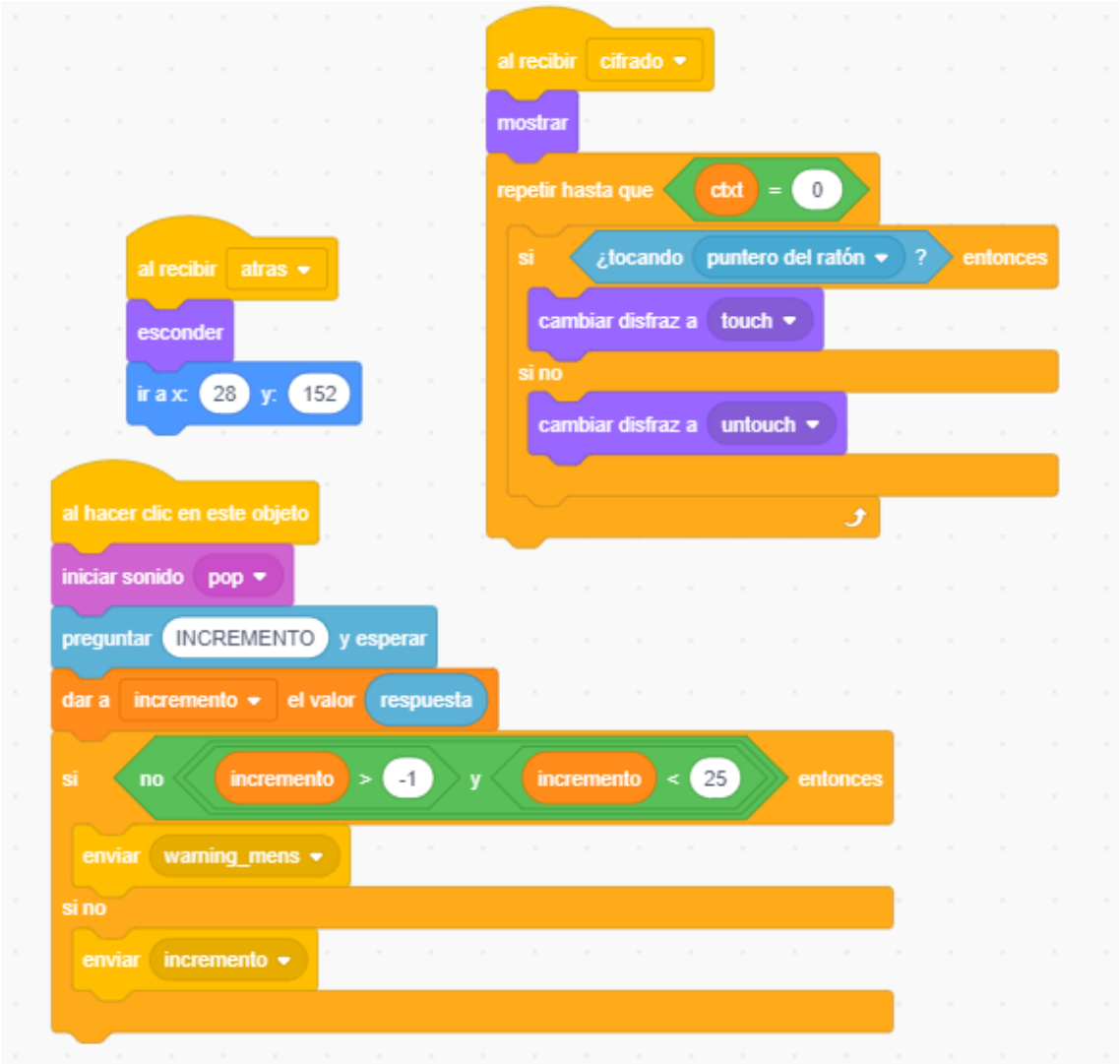


Figura 4.9: Fragmento de código del botón Incremento.

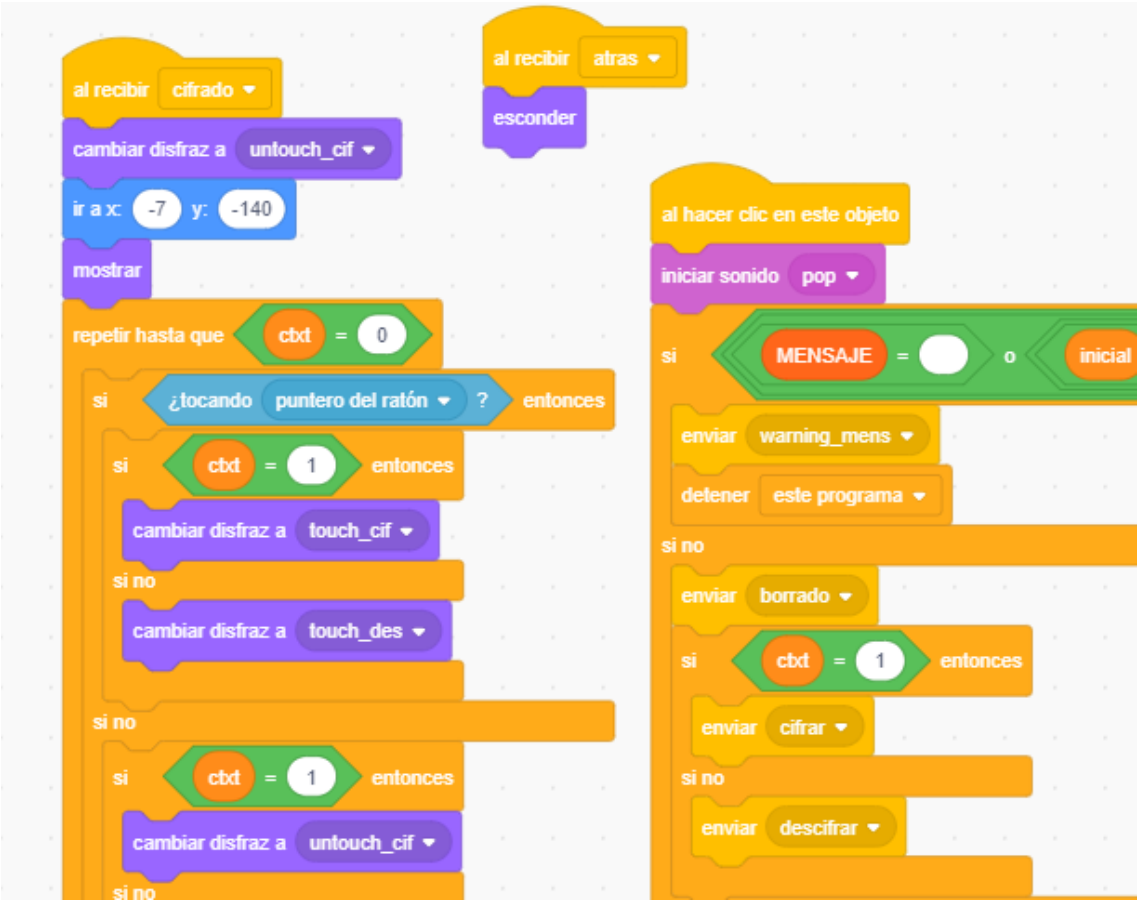


Figura 4.10: Fragmento de código del botón Cifrar.

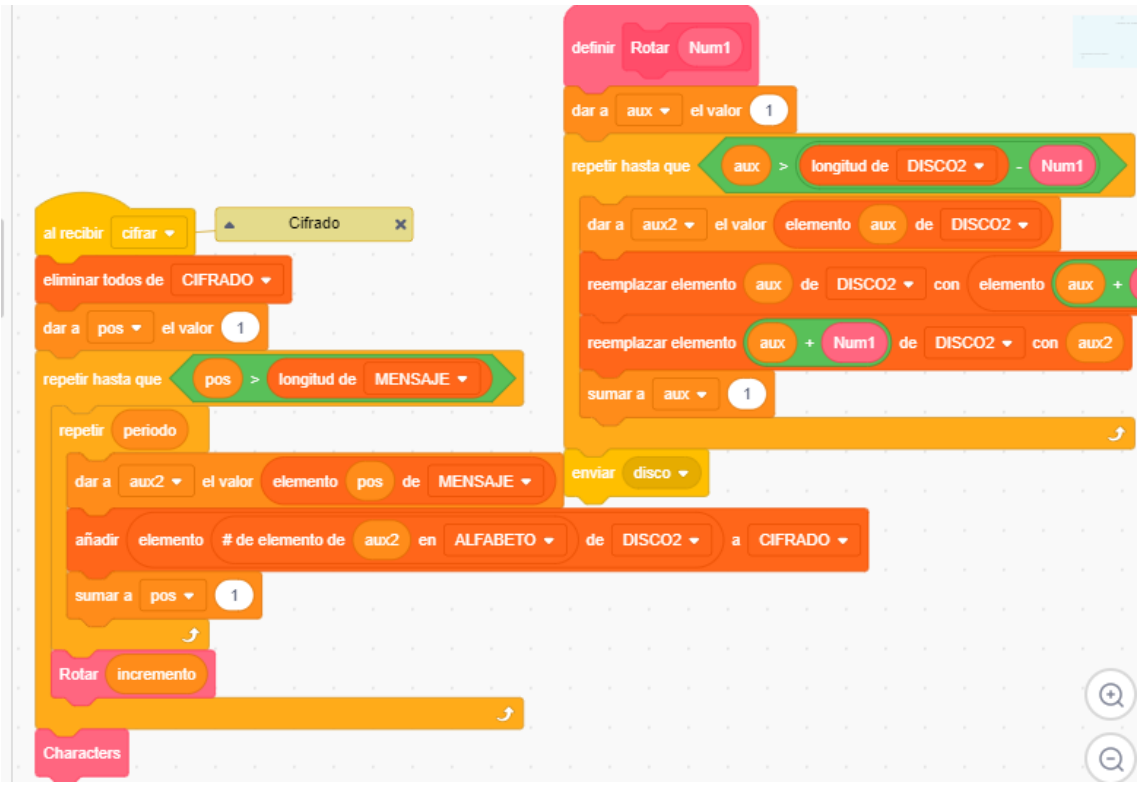


Figura 4.11: Fragmento de código del método Cifrar.

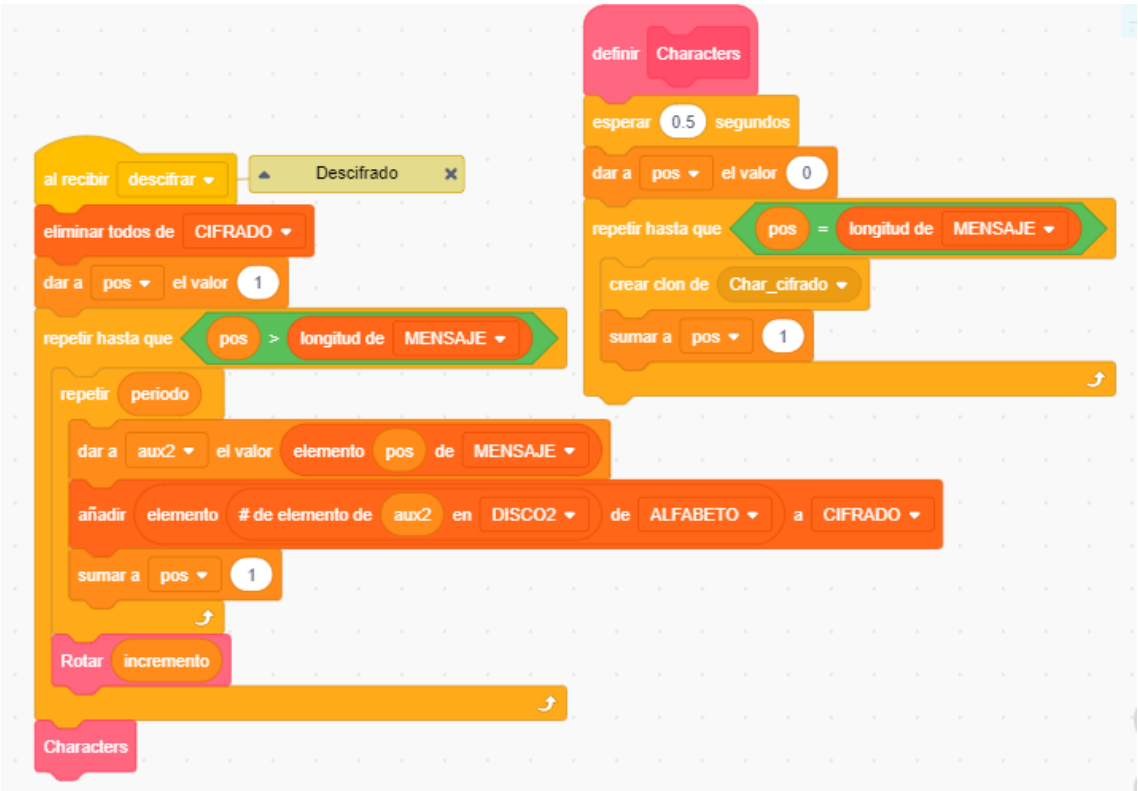


Figura 4.12: Fragmento de código del método Descifrar.

CAPÍTULO 5

Diseño e implementación del cifrado de Vigenère en Scratch

El segundo cifrado que se va a describir es del cifrado de Vigenère. Como se explicó en el capítulo 2 se trata de un cifrado polialfabético implantado desde el año 1508 por Giovan Battista Bellaso. En este capítulo se explicará detalladamente el proceso llevado a cabo para la elaboración y la implementación de este cifrado en Scratch.

Al ser un cifrado que no tiene muchas peculiaridades a la hora de cifrar se ha implementado también en Java para comparar las dos implementaciones. Dicha implementación se encuentra en el Apéndice A.

5.1 Organización del menú principal

Lo primero que nos encontramos al acceder al programa del cifrado es un menú principal sencillo (véase Figura 5.1) que consta de tres botones «Instrucciones», «Comenzar» e «Historia» que ofrecen distintos escenarios al usuario. A continuación se explicará brevemente la función que tiene cada uno.

Figura 5.1: Menú principal del cifrado de Vigenère.

- **Instrucciones.** Al pulsar en este botón nos presentará un nuevo escenario donde se explicará brevemente como funciona el cifrado dentro del programa. Dicho escenario se muestra cuando el programa cambia de fondo al denominado «instrucciones» donde se encuentra descrito el cifrado.
- **Comenzar.** Este botón es el más importante porque permitirá al usuario comenzar a cifrar o descifrar sus mensajes desde un escenario diferente. Al pulsar en este botón el fondo cambiará al llamado «cifrado», donde no aparece ningún texto que impida al usuario la correcta ejecución del cifrado, también se eliminarán los caracteres de las listas «Mensaje» y «Clave» para evitar los fallos que pudieran surgir, se cambia el valor de la variable «ctxt» a *1* y se envía el mensaje «cifrado» al programa. En la sección 5.2 se contará con detalle la actividad de los botones de este escenario.
- **Historia.** Cuando se pulsa en este botón se mostrará un nuevo escenario donde se mostrará una reducida historia sobre el cifrado. Dicho escenario se muestra cuando el programa cambia de fondo al denominado «historia» donde se encuentra un breve contexto histórico del cifrado.

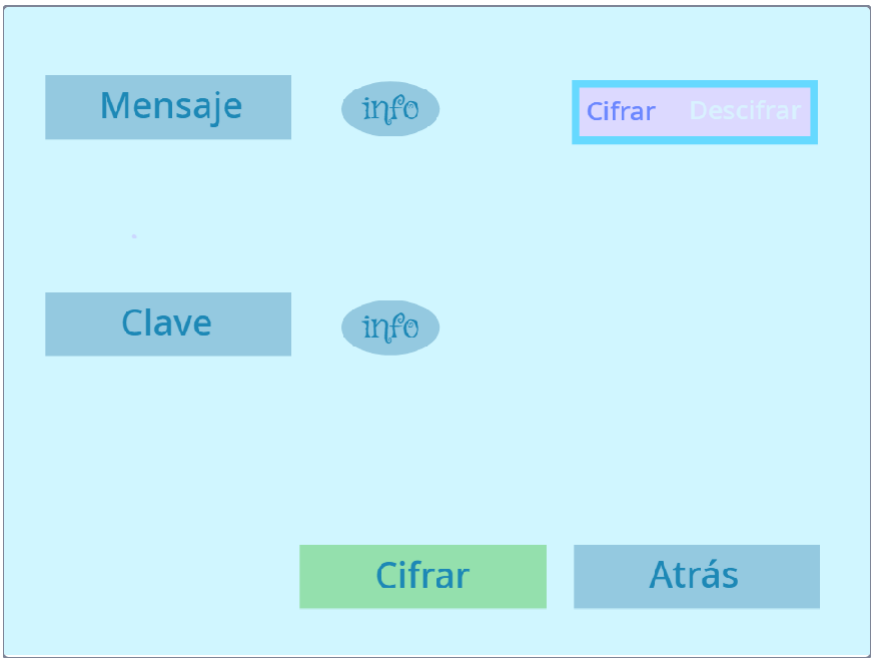


Figura 5.2: Pantalla principal del cifrado de Vigenère.

5.2 Objetos del cifrado

El actual cifrado se constituye de 21 objetos diferentes de los cuales solo 5 proporcionarán funcionalidad al cifrado. A continuación se explicará el funcionamiento de estos 5 objetos (véase Figura 5.2).

1. **Botón para Cifrar o Descifrar.** Este botón tiene la función más importante porque dependiendo de qué opción esté seleccionada el programa cifrará o descifrará el mensaje dependiendo de la clave. Al tener la opción «Cifrar» seleccionada en botón inferior aparecerá Cifrar que permitirá cifrar el mensaje, por el contrario si tenemos la opción «Descifrar» activada aparecerá Descifrar que posibilita la opción de descifrar el mensaje.

La implementación de este botón (véase Figura 5.5) es bastante sencilla. Cuando se pulsa el botón se comprueba que disfraz está seleccionado, si el disfraz seleccionado es el número 1, corresponde al disfraz de cifrar por lo que cambiará el valor de la variable «ctxt» a *1*. Si el disfraz seleccionado es el número 2 que corresponde a la opción de descifrar, cambiará la variable «ctxt» al valor *2*.

2. **Mensaje.** Este botón tiene la función de permitir al usuario introducir el mensaje que quiere utilizar en el cifrado.

   La implementación de este botón es algo simple (véase Figura 5.3). Al pulsar en el botón se borrará el mensaje que haya sido introducido anteriormente para que no puedan surgir errores, se mostrará un cuadro de texto en el que se introducirá el nuevo mensaje. Se comprobará que no se ha introducido ningún carácter especial y ningún símbolo, si no fuese así se enviará el mensaje «warning_char» que mostrará el mensaje de la Figura 3.16. Si la comprobación es correcta el mensaje introducido se guardará en la variable temporal «respuesta» la cual se guardará, carácter a carácter en la lista «Mensaje», una vez realizado se comprueba que la longitud del mensaje esté entre *0* y *24* caracteres si la longitud está dentro de ese rango se crearán tantos clones de «Char_mensaje» como longitud de mensaje los cuales mostrarán por pantalla el mensaje, pero si la longitud del mensaje no estuviera dentro de ese rango se enviará el mensaje «warning_mens» que mostrará el mensaje de error «Introducir mensaje o clave válidos (pulsar en info para saber más)».

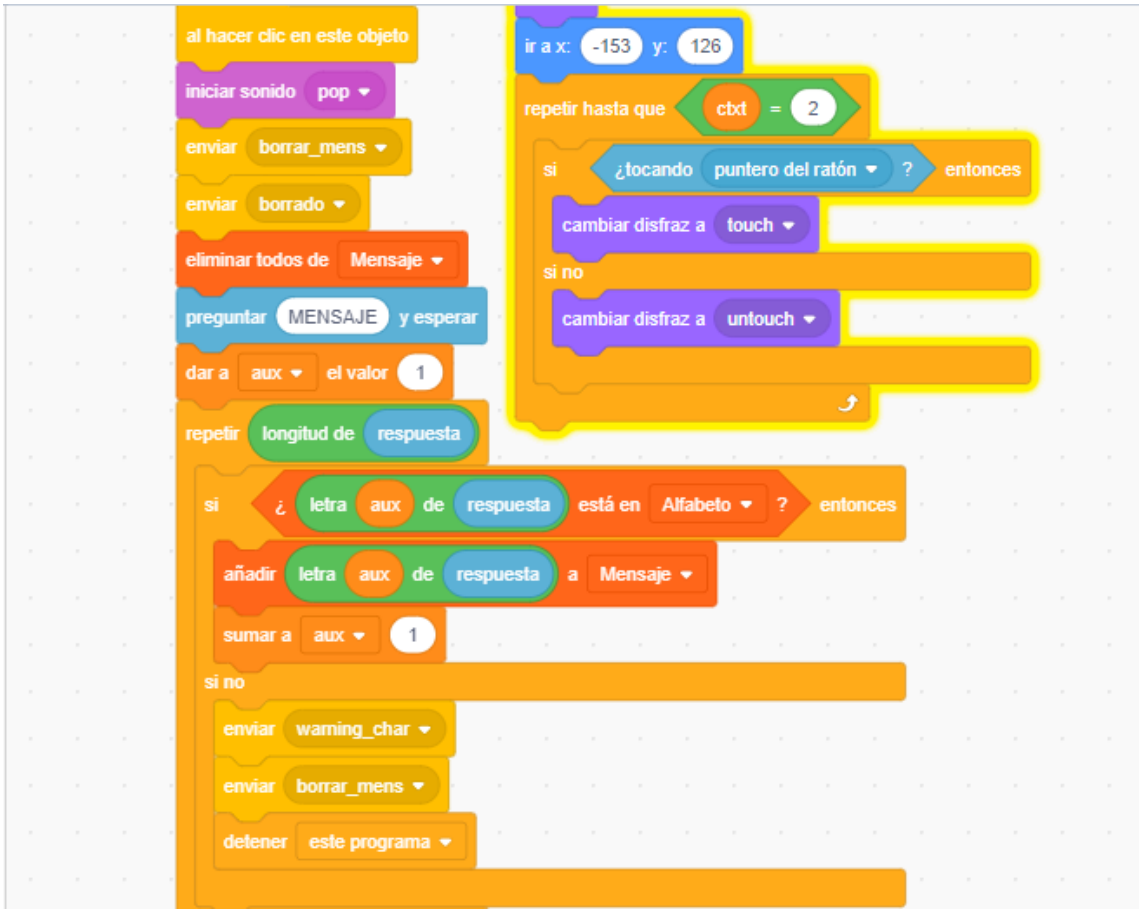


Figura 5.3: Fragmento de código del botón Mensaje.

3. **Clave.** La funcionalidad de este botón es muy parecida a la del botón comentado anteriormente, pero en este caso permite al usuario introducir la clave que se utilizara a lo largo del cifrado.

La implementación es similar al anterior (véase Figura 5.6), al pulsar en el botón se borrará la clave que haya sido introducida anteriormente para evitar problemas, se mostrará un cuadro de texto en el que se introducirá la nueva clave. En este momento se comprobará que no se ha introducido ningún carácter especial ni ningún símbolo, si no fuera así se enviará el mensaje «warning_char» y se mostrará el mensaje de advertencia de la Figura 3.16. Si la comprobación es correcta la clave se guardará en la variable temporal «respuesta» la cual se guardará, carácter a carácter, en la lista «Clave» una vez realizado comprobaremos que la longitud de la clave este entre *0* y *6* caracteres, sin espacios, si la longitud esta dentro de ese rango se crearán tantos clones de «Char_clave» como longitud de clave, los cuales mostrarán por pantalla la clave pero si la longitud de la clave no estuviera dentro de ese rango o se hubiesen introducido espacios se enviará el mensaje «warning_mens» que mostrará el mensaje de error «Introducir mensaje o clave válidos (pulsar en info para saber más)».

4. **Cifrar.** La funcionalidad de este botón es permitir al usuario ejecutar el cifrado, tanto si quiere cifrar el mensaje como si desea descifrarlo.

   La implementación de este botón (véase Figura 5.4) no es compleja, al pulsarlo comprueba que la longitud del mensaje y de la clave no sea nula, la comprobación de que cumplan un rango se comprueba en los botones correspondientes, si no es así se enviará el mensaje se enviará el mensaje «warning_mens» que mostrará el mensaje de error que ya hemos comentado anteriormente, si la comprobación es correcta se borra todo resultado final que estuviese guardado previamente y comprueba la variable «ctxt» si es 1 se envía el mensaje «cifrar», sino se envía el mensaje «descifrar».

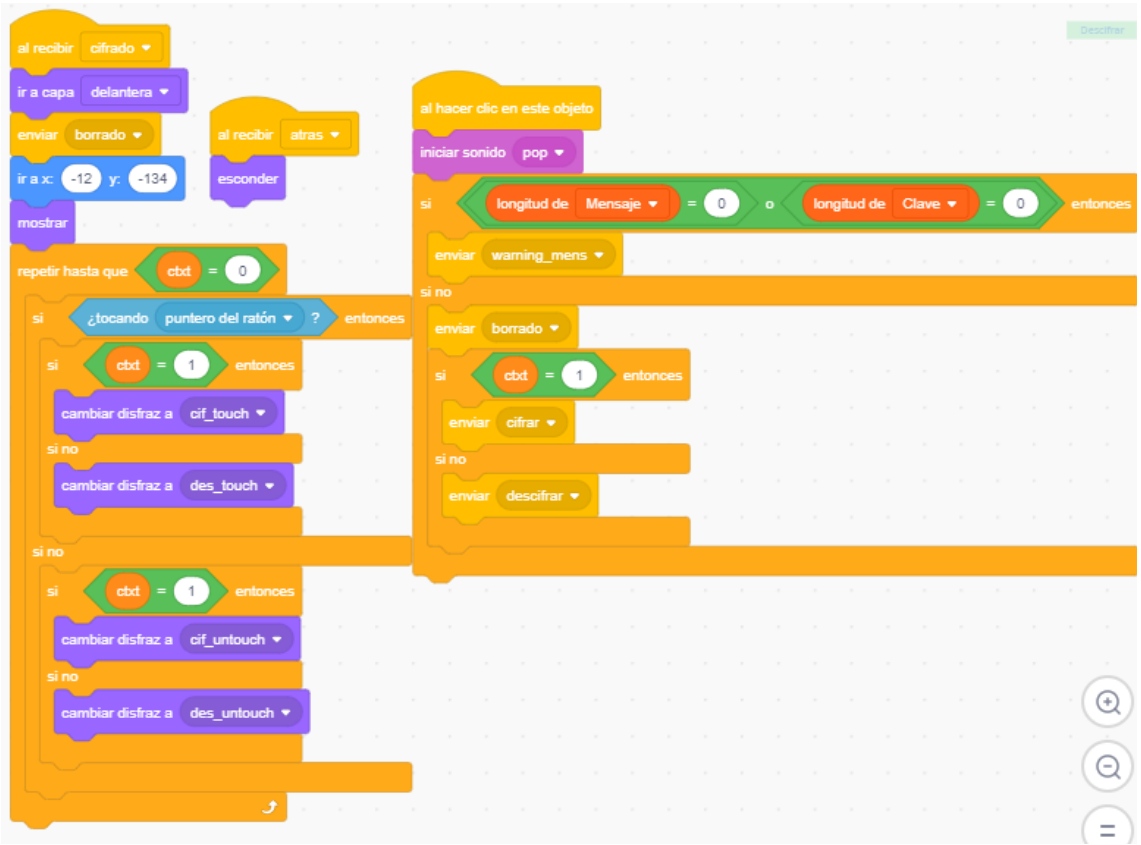


Figura 5.4: Fragmento de código del botón Cifrar.

Cuando el programa recibe el mensaje «cifrar» (véase Figura 5.7) se eliminan todos los espacios del mensaje, si es que tuviera alguno, se eliminan los caracteres de la

lista «cifrado» y se ejecutará el siguiente proceso carácter por carácter del mensaje. Primero se obtiene la posición del carácter del mensaje en un alfabeto predefinido, en este caso el alfabeto español. Luego se obtiene el módulo de esa posición con la longitud de la clave, para saber qué carácter de la clave coincide con el carácter del mensaje y se obtiene la posición de ese carácter de la clave en el alfabeto. A continuación se suman las dos posiciones obtenidas y se aplica el módulo con la longitud del alfabeto. El resultado obtenido nos dice en qué posición del alfabeto se encuentra el carácter cifrado del mensaje, pero debido a las limitaciones del Scratch hay que restarle una unidad a ese resultado para que el cifrado este correcto.

Cuando el programa recibe el mensaje «descifrar» (véase Figura 5.8) se eliminan todos los espacios del mensaje, si es que tuviera alguno, se eliminan los caracteres de la lista «cifrado» y se ejecutará el siguiente proceso carácter por carácter del mensaje. Primero se obtiene la posición del carácter del mensaje en un alfabeto predefinido, en este caso el alfabeto español. Luego se obtiene el módulo de esa posición con la longitud de la clave, para saber que carácter de la clave coincide con el carácter del mensaje y se obtiene la posición de ese carácter de la clave en el alfabeto. A continuación se comprueba la resta de las dos posiciones obtenidas si es menor que 0 se sumará la longitud del alfabeto y se aplica el módulo con la longitud del alfabeto, si es mayor que 0 solo se le aplica el módulo con la longitud del alfabeto. El resultado obtenido nos dice en que posición del alfabeto se encuentra el carácter cifrado del mensaje, pero debido a las limitaciones del Scratch hay que restarle una unidad a ese resultado para que el cifrado este correcto.

5. **Atrás.** Este botón ofrece al usuario volver al menú principal si se encuentra tanto en el cifrado, en las instrucciones del cifrado o en la historia del cifrado.

   La implementación (véase Figura 5.9) es sencilla. Al pulsar el botón la variable cambia su valor «ctxt» a 0 para indicar que queremos volver al menú principal, se cambia el fondo al de la pantalla principal denominado «principal» y envía el mensaje «borrado» que hace que el programa borre cualquier variable que este guardada.

5.3 Posibles ampliaciones

El cifrado ha sido completado aunque hay que comentar que se puede añadir alguna mejora que se explicará como posibles ampliaciones. La primera ampliación posible podría ser añadir más alfabetos al cifrado y permitir al usuario elegir en que alfabeto quiere cifrar el mensaje, en nuestro caso damos predefinido el alfabeto español. Otra ampliación podría ser poder introducir mayúsculas y minúsculas combinadas tanto en la clave como en el mensaje para un cifrado del mensaje más complejo.

Otra posible ampliación interesante sería poder elegir el idioma de trabajo y que cada botón se adaptará a ese idioma haciendo así posible que el mismo programa fuera usado por personas con diferentes idiomas. También se podría introducir alguna mejora en el aspecto de sonido como por ejemplo introducir alguna música de fondo que imitara a la de la época del cifrado, o alguna mejora en el aspecto estético cambiando de color los fondos o los botones dependiendo de donde se encuentre el usuario.

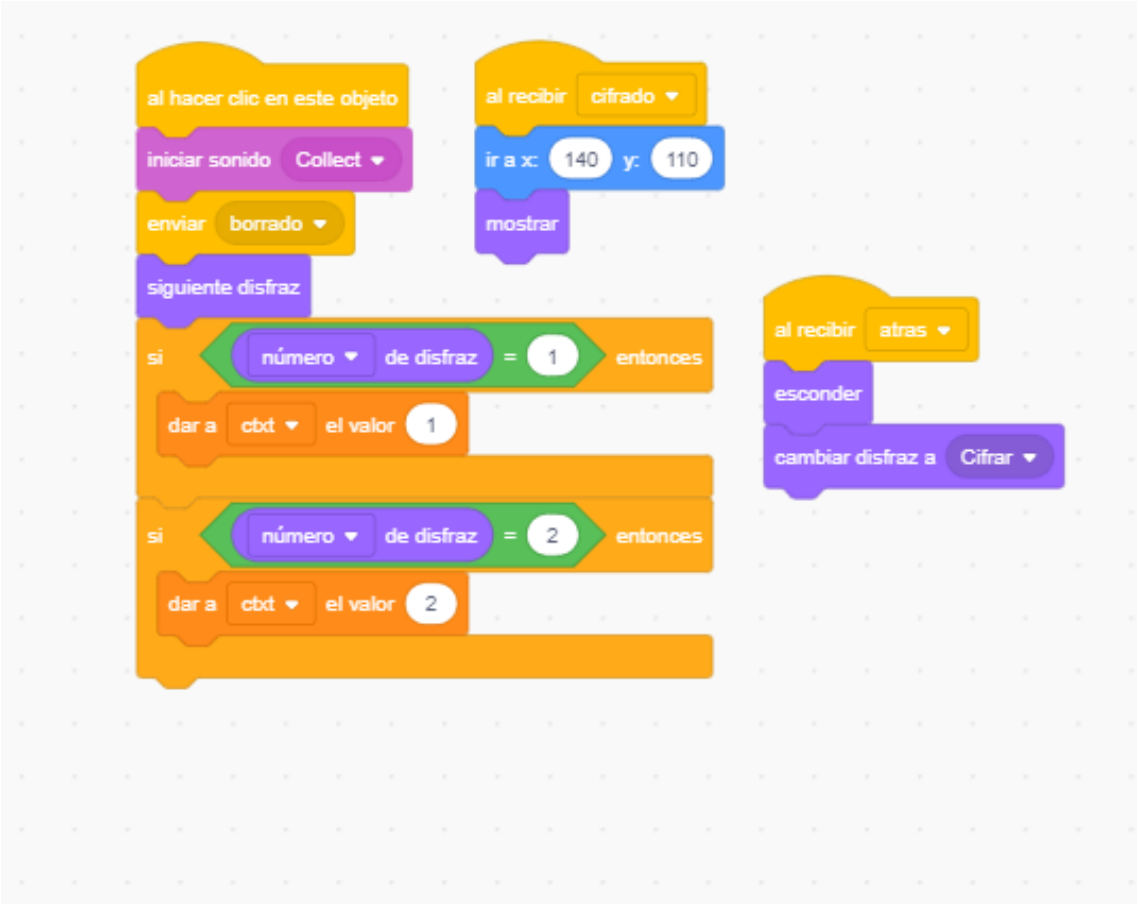


Figura 5.5: Fragmento de código del botón para Cifrar/Descifrar.

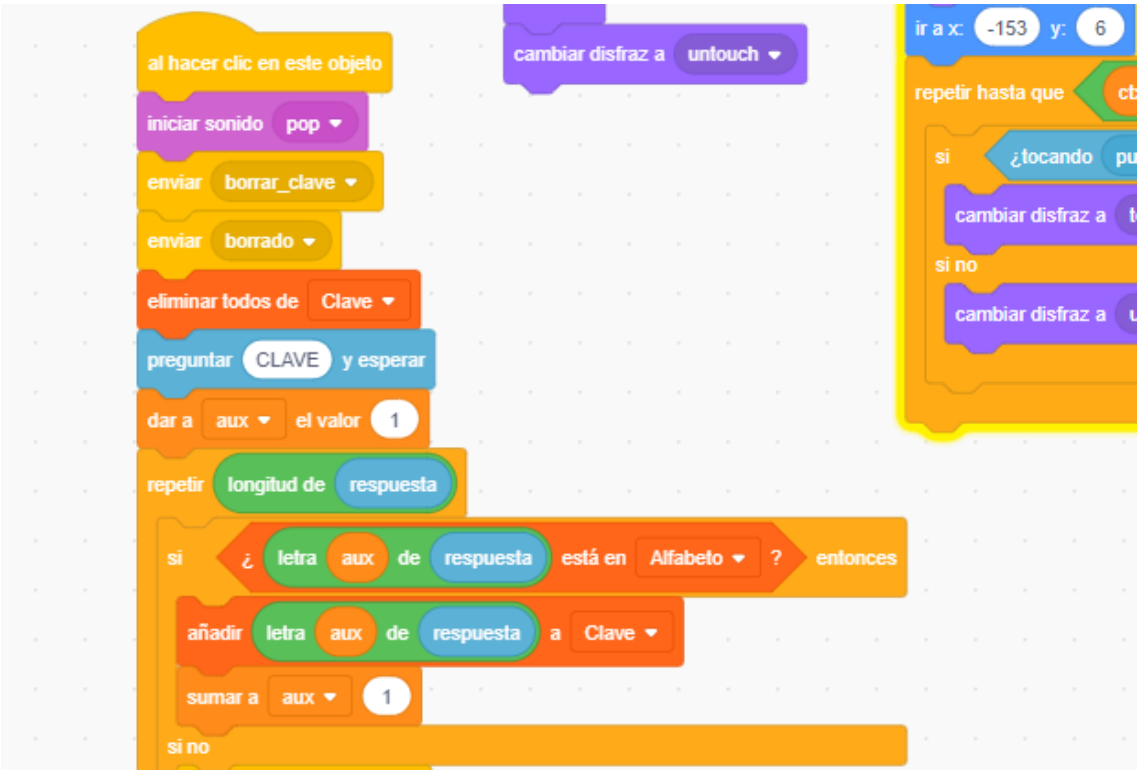


Figura 5.6: Fragmento de código del botón Clave.

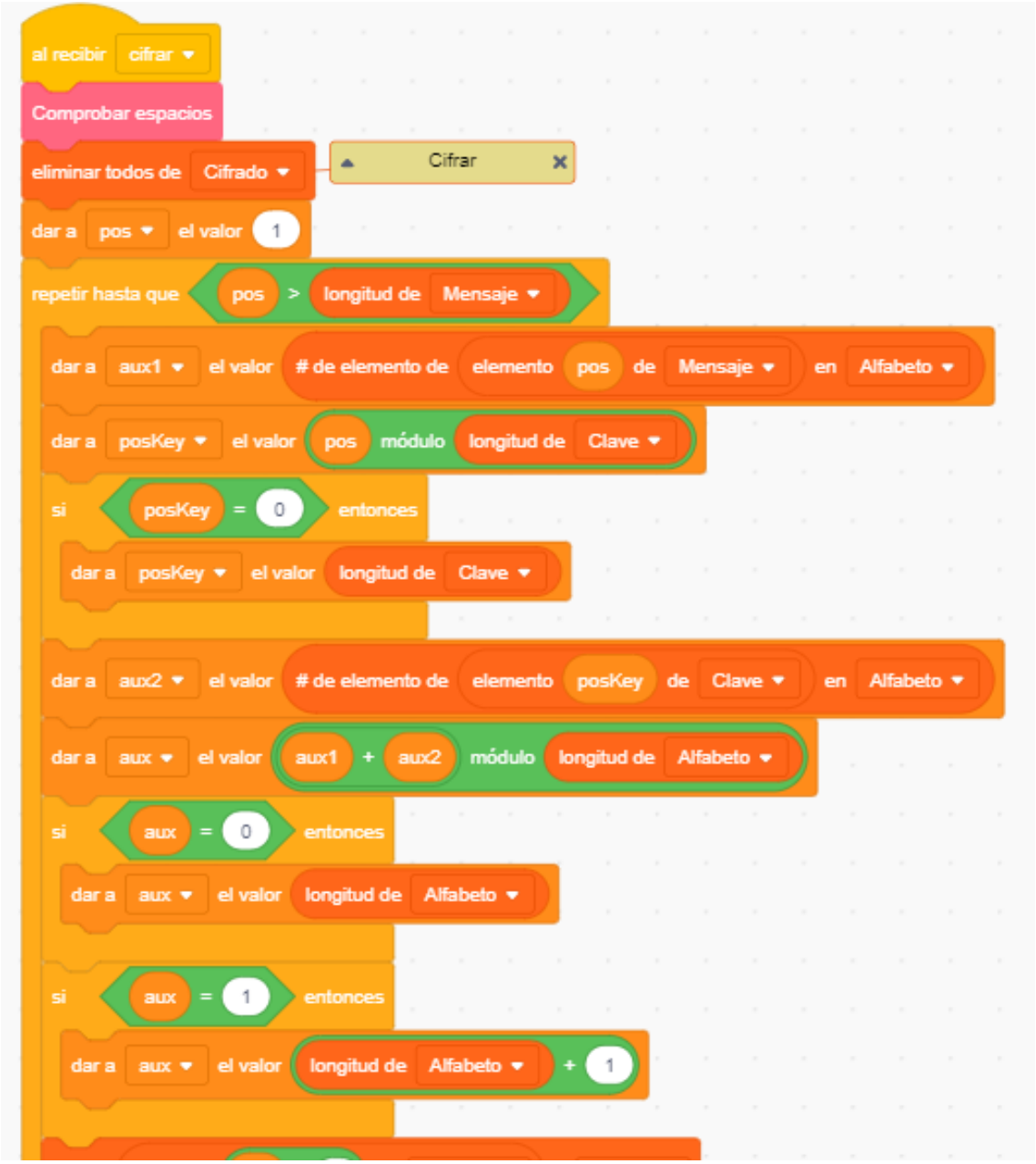


Figura 5.7: Fragmento de código del cifrado.

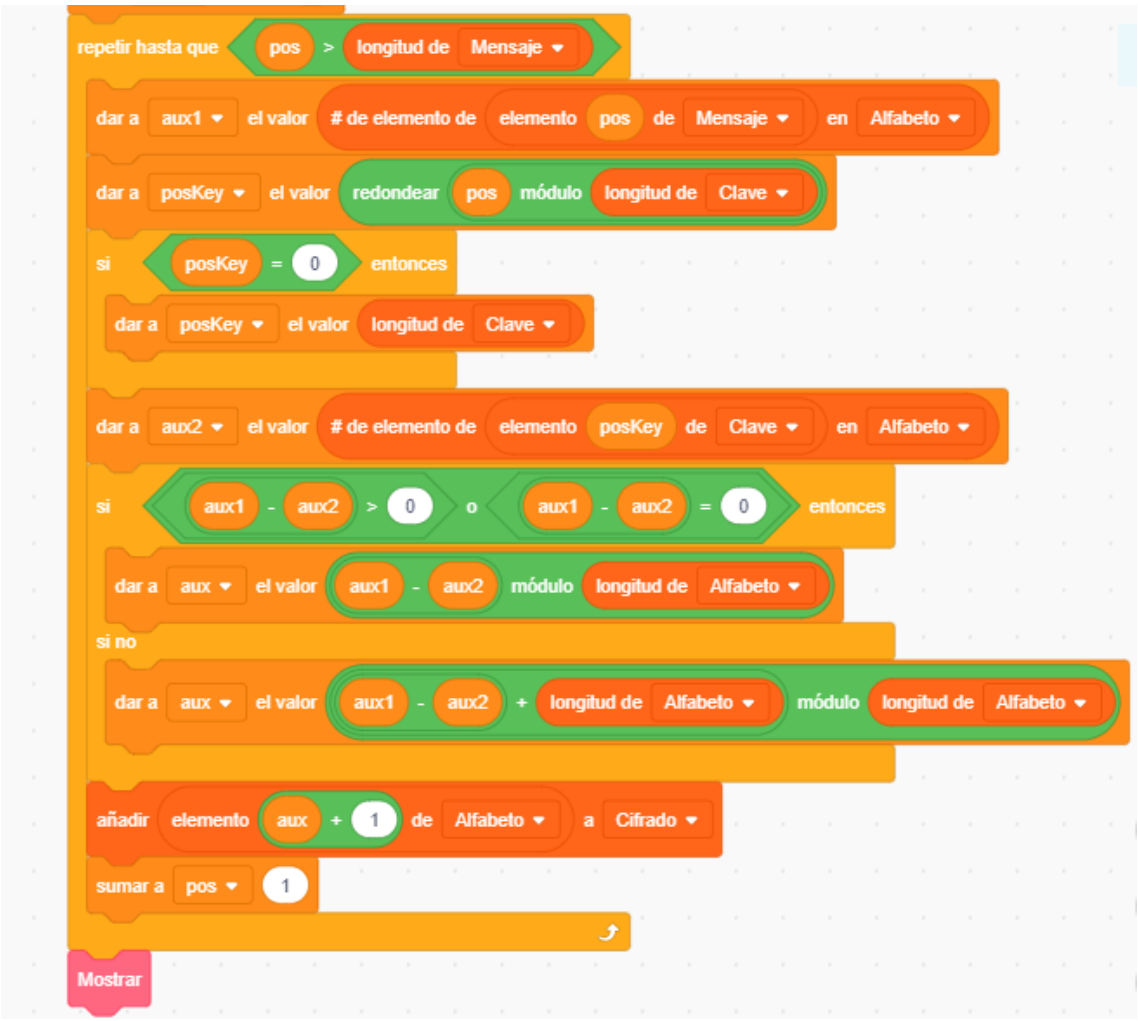


Figura 5.8: Fragmento de código del botón descifrado.

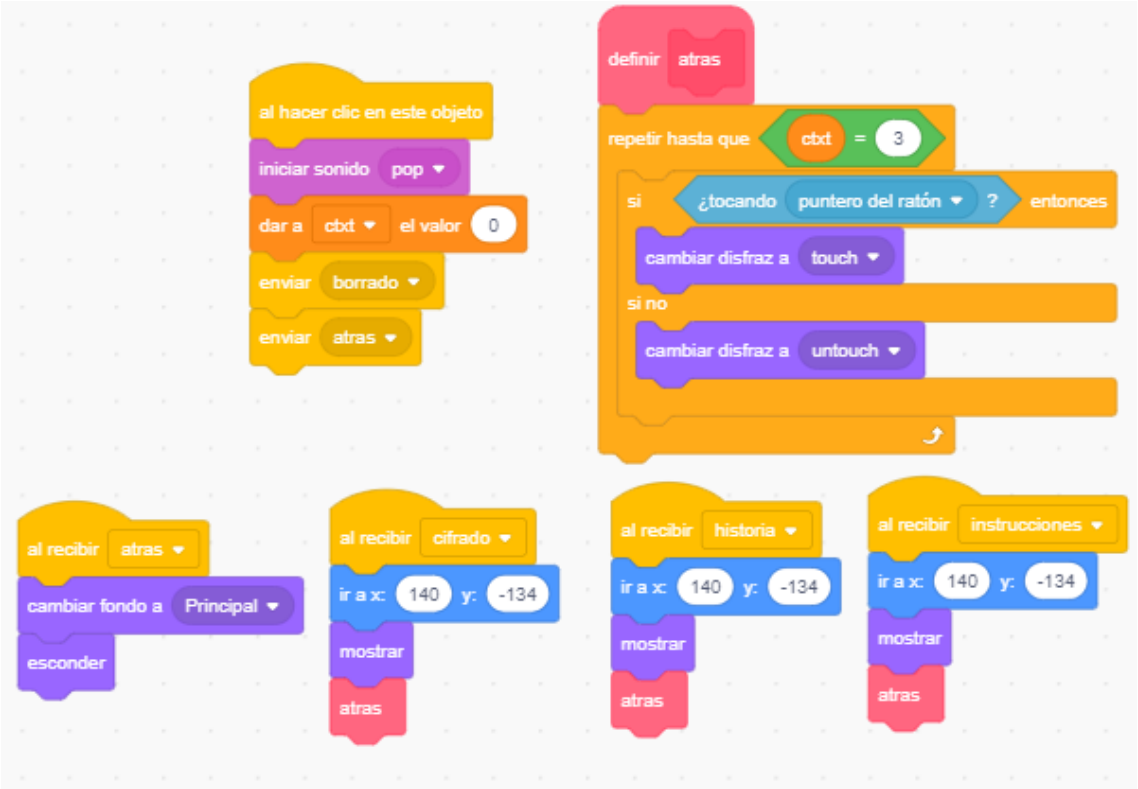


Figura 5.9: Fragmento de código del botón Atrás.

CAPÍTULO 6
Diseño e implementación del cifrado Playfair en Scratch

El siguiente cifrado que se va a describir en este trabajo es el cifrado de Playfair. Como se comentó en el capítulo 2 este cifrado fue inventado por Charles Wheatstone en el año 1854. Se trata de un cifrado monoalfabético, porque un mismo mensaje puede estar cifrado de diversas formas dependiendo de que clave se utilice. En este capitulo se detallará todo el proceso seguido para su implementación.

6.1 Organización del menú principal

Al acceder al programa nos encontramos con un menú principal (véase Figura 6.1) sencillo que se compone de cuatro botones «Instrucciones», «Cifrado», «Descifrado» y «Historia». A continuación se explicará brevemente cada uno de los botones.

Figura 6.1: Menú principal del cifrado de Playfair.

- **Instrucciones.** Al pulsar en este botón se cambiará el escenario, cambiando al fondo «Instrucciones» que muestra unas breves instrucciones de cómo funciona el cifrado.

- **Cifrado.** En el caso de pulsar este botón se mostrará un nuevo escenario donde se permitirá al usuario introducir los datos para poder empezar a cifrar el mensaje. La variable «ctxt» cambiará a *1* indicando al programa que se quiere cifrar, se eliminarán todos los caracteres de las listas «Clave» y «Mensaje», se cambiará al fondo llamado «cifrado», que no tiene ningún texto que impida la correcta ejecución del cifrado y se enviará al programa el mensaje «cifrado».
- **Descifrado.** En el caso de que el usuario pulse este botón se mostrará un escenario que permitirá al usuario introducir los datos para descifrar el mensaje. La variable «ctxt» cambiará a *2* indicando al programa que se quiere descifrar, se eliminarán todos los caracteres de las listas «Clave» y «Mensaje», se cambiará al fondo llamado «cifrado» y se enviará al programa el mensaje «cifrado».
- **Historia.** En el caso de pulsar en este botón el escenario, cambiando el fondo al fondo «Historia» en el que aparece una breve historia sobre el cifrado.

6.2 Objetos del cifrado

El cifrado se constituye de 24 objetos diferentes, solo 5 botones facilitan funcionalidad al cifrado. A continuación se explicará el funcionamiento de estos (Véase Figura 6.2).

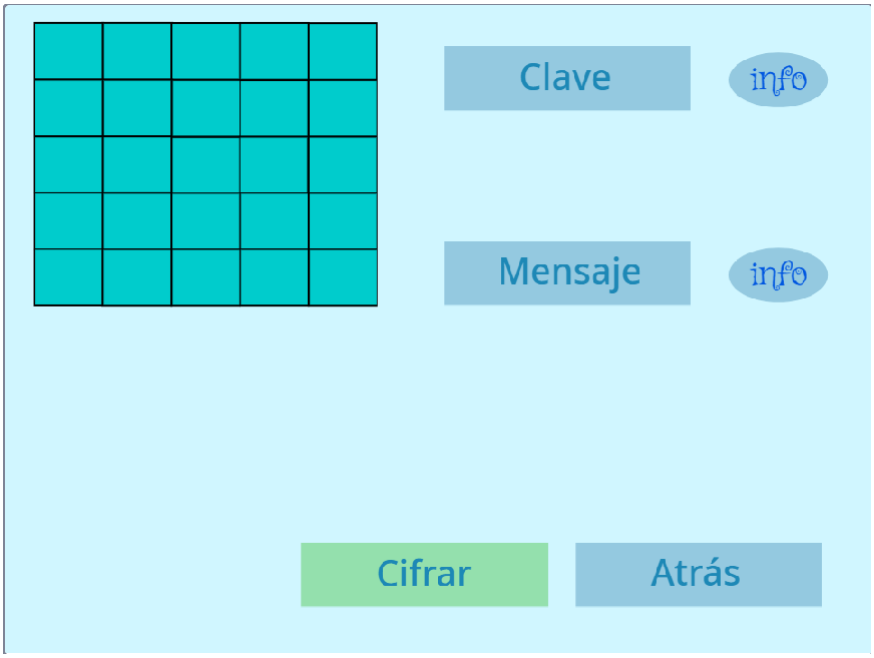


Figura 6.2: Pantalla del cifrado de Playfair.

1. **Clave.** Cuando se pulsa este botón facilita al usuario introducir la clave del cifrado, con la que se creará la tabla que se utiliza para el cifrado o descifrado del mensaje.

   La implementación es la siguiente (véase Figura 6.3). Al pulsar el botón se borrarán los caracteres de la lista «Clave» y se ocultarán los caracteres de la clave que este en pantalla con el mensaje «borrar_clave», se mostrará un cuadro de texto en el que se introducirá la clave que se guardará en la variable temporal «respuesta», se comprobará que la longitud está dentro del rango entre *1* y *6*. Si no se cumple esta condición se enviará al programa el mensaje «warning_mens» pero si se cumple el código seguirá ejecutándose. En este momento se comprueba si se han

introducido caracteres especiales o símbolos. Si se han introducido se envía el mensaje «warning_char» y se detiene el programa, si no se han introducido la variable «respuesta» se guardará carácter a carácter en la lista «Clave» comprobando que no tenga espacios y se verifica si tiene algún carácter que tenga que ser cambiado como pasa con los caracteres *J* y *Ñ* que se cambian por *I* y *N* respectivamente. Una vez verificado todo se crearán tantos clones de «Char_clave» como longitud de Clave y con esto se mostrarán los caracteres de la clave por pantalla. Después se enviará el mensaje «crear_tabla» al programa y mostrará la tabla en pantalla creando tantos clones de «Char_tabla» como longitud tenga la tabla.

El programa al recibir el mensaje «crear_tabla» mostrará por pantalla en mensaje «Creando tabla» y ejecuta el código que crea la tabla. La tabla se crea primero añadiendo los caracteres que tenga la clave y después va revisando el alfabeto, añadiendo por orden los caracteres que todavía no estén introducidos en la tabla.

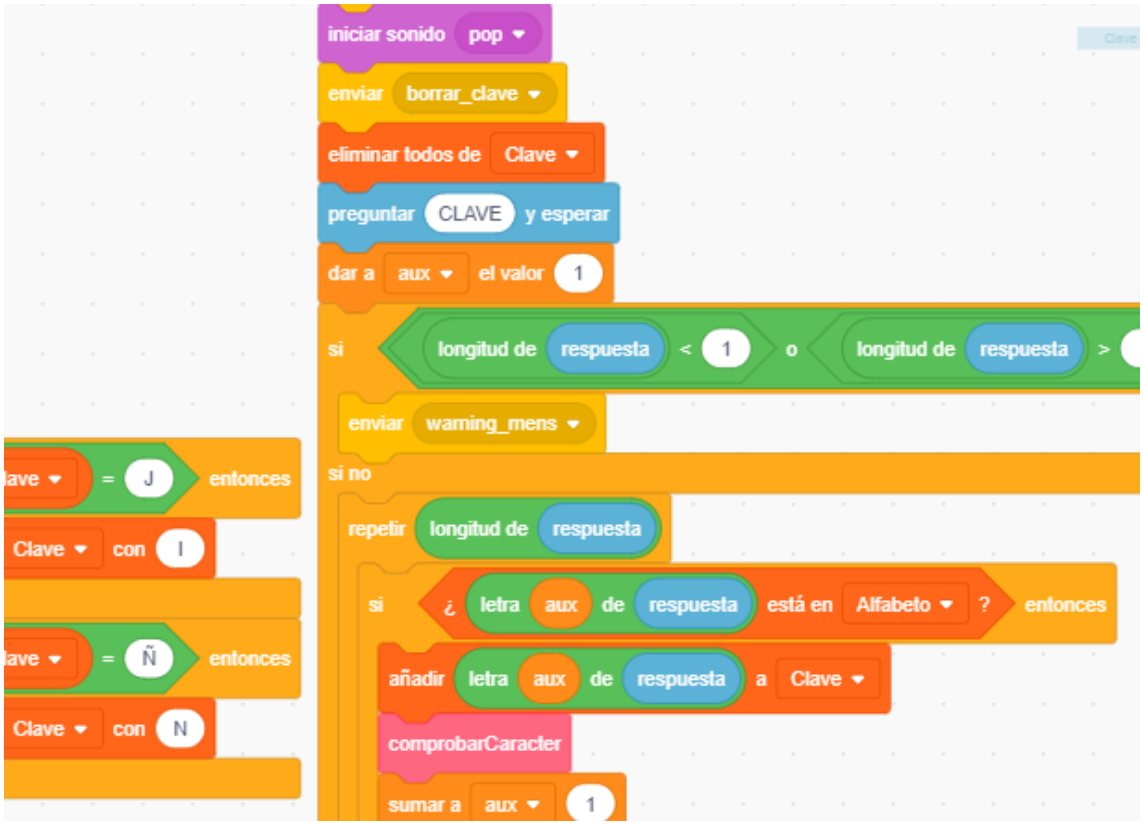


Figura 6.3: Fragmento de código del botón Clave.

2. **Mensaje.** Al pulsar en este botón se permitirá al usuario introducir el mensaje que se utilizará en el cifrado o en el descifrado dependiendo en que escenario se encuentre.

   La implementación (véase Figura 6.5) es bastante parecida a la del botón comentado anteriormente, al pulsar el botón se borrarán los caracteres que estuviese guardado en la lista «Mensaje», también se ocultarán los caracteres del mensaje que estuviesen en pantalla enviando el mensaje «borrar_mens». Se abrirá un cuadro de texto en el que se introducirá el mensaje, el mensaje introducido en el cuadro de texto se guardará en la variable temporal «respuesta», y se comprobará que la longitud de esta variable esta dentro del rango entre *1* y *18*, si la comprobación es incorrecta se enviará el mensaje al programa «warning_mens» pero si la comprobación es correcta se seguirá ejecutando el código. En este momento se comprueba que no

se hayan introducido símbolos ni caracteres especiales. Si es así la variable «respuesta» se guardará en la lista «Mensaje», carácter a carácter, comprobando se hay algún espacio y si hay algún carácter que utilizar las equivalencias en los caracteres como es el caso de los caracteres J y $\tilde{N}$ que se cambian por I y N respectivamente. Si se hubiesen introducido caracteres especiales o símbolos se enviará los mensaje «warning_char» y «borrar_mens» y se detendrá el programa, con eso mensajes el programa mostrará un mensaje de advertencia por los caracteres especiales y los símbolos y borrará el mensaje introducido. Para finalizar si todas las comprobaciones esta correctas se crearán tantos clones de «Char_mensaje» como longitud del mensaje con lo que se mostrarán los caracteres del mensaje por pantalla.

3. **Cifrar.** Al pulsar en este botón se aplicarán las reglas del cifrado al mensaje y mostrarán al usuario los caracteres del mensaje cifrado.

   La implementación del objeto es sencilla (véase Figura 6.6), ya que todo el código del cifrado se encuentra creado en el escenario, el objeto predefinido en Scratch, por los problemas que pudieran surgir en la modificación de este objeto. Primero se vuelve a comprobar que la clave y el mensaje están dentro de los rango comentado anteriormente, la clave entre 1 y 6 caracteres y el mensaje entre 1 y 18 caracteres. Si no estuviesen dentro del rango se envía el mensaje «warning_mens» que muestra por pantalla el mensaje «Introducir Mensaje o Clave válidos (pulsa en info para saber más)», si la clave y el mensaje si que están dentro del rango correspondiente se envía los mensaje «borrado» y «cifrar».

   El programa al recibir el mensaje «borrado» oculta todos los caracteres del mensaje cifrado que estuvieran en pantalla. Al recibir el mensaje «cifrar» ejecuta el código del cifrado, primero elimina todos los caracteres del la lista «Cifrado», después va recorriendo el mensaje por parejas de caracteres y va comprobando las reglas del cifrado comentadas en la sección 2.3.2 y cuando este todo el mensaje comprobado y cifrado se mostrará por pantalla al usuario. Mientras se van comprobando las reglas se muestra por pantalla el mensaje «Procesando ...» para mostrar al usuario que el programa sigue funcionando y está procesando el cifrado.

4. **Descifrar.** La función de este botón es similar al botón «Cifrar» pero con este botón el usuario descifrará el criptograma para mostrar el texto en claro.

   La implementación (véase Figura 6.7) , como en cifrado se encuentra en el escenario. Primero se vuelve a comprobar que la clave y el mensaje están dentro de los rango comentado anteriormente, la clave entre 1 y 6 caracteres y el mensaje entre 1 y 18 caracteres. Si no estuviesen dentro del rango se envía el mensaje «warning_mens» que muestra por pantalla el mensaje «Introducir Mensaje o Clave válidos (pulsa en info para saber más)», si la clave y el mensaje si que están dentro del rango correspondiente se envía los mensaje «borrado» y «descifrar».

   El programa al recibir el mensaje «borrado» oculta todos los caracteres del mensaje descifrado que estuvieran en pantalla. Al recibir el mensaje «descifrar» ejecuta el código del descifrado, primero elimina todos los caracteres del la lista «Cifrado», después va recorriendo el mensaje por parejas de caracteres y va comprobando las reglas del descifrado comentadas en la sección 2.3.3 y cuando este todo el mensaje comprobado y descifrado se mostrará por pantalla al usuario. Mientras se van comprobando las reglas del descifrado se muestra por pantalla el mensaje «Procesando ...» para mostrar al usuario que el programa sigue funcionando y está procesando el descifrado.

5. **Atrás.** La funcionalidad de este botón es sencilla, se trata del botón que permite al usuario volver al menú principal cuando se encuentre en cualquier escenario.

La implementación(véase Figura 6.4) de este botón muestra que al pulsarlo cambiará el fondo al fondo «Principal», cambiará la variable «ctxt» a 0 por lo que el programa saldrá del modo cifrado o descifrado y enviará el mensaje «atrás» al programa. El programa al recibir el mensaje «atrás» esconderá todos los objetos que no pertenecen al menú principal y mostrará los que si pertenecen mostrando así la pantalla principal.

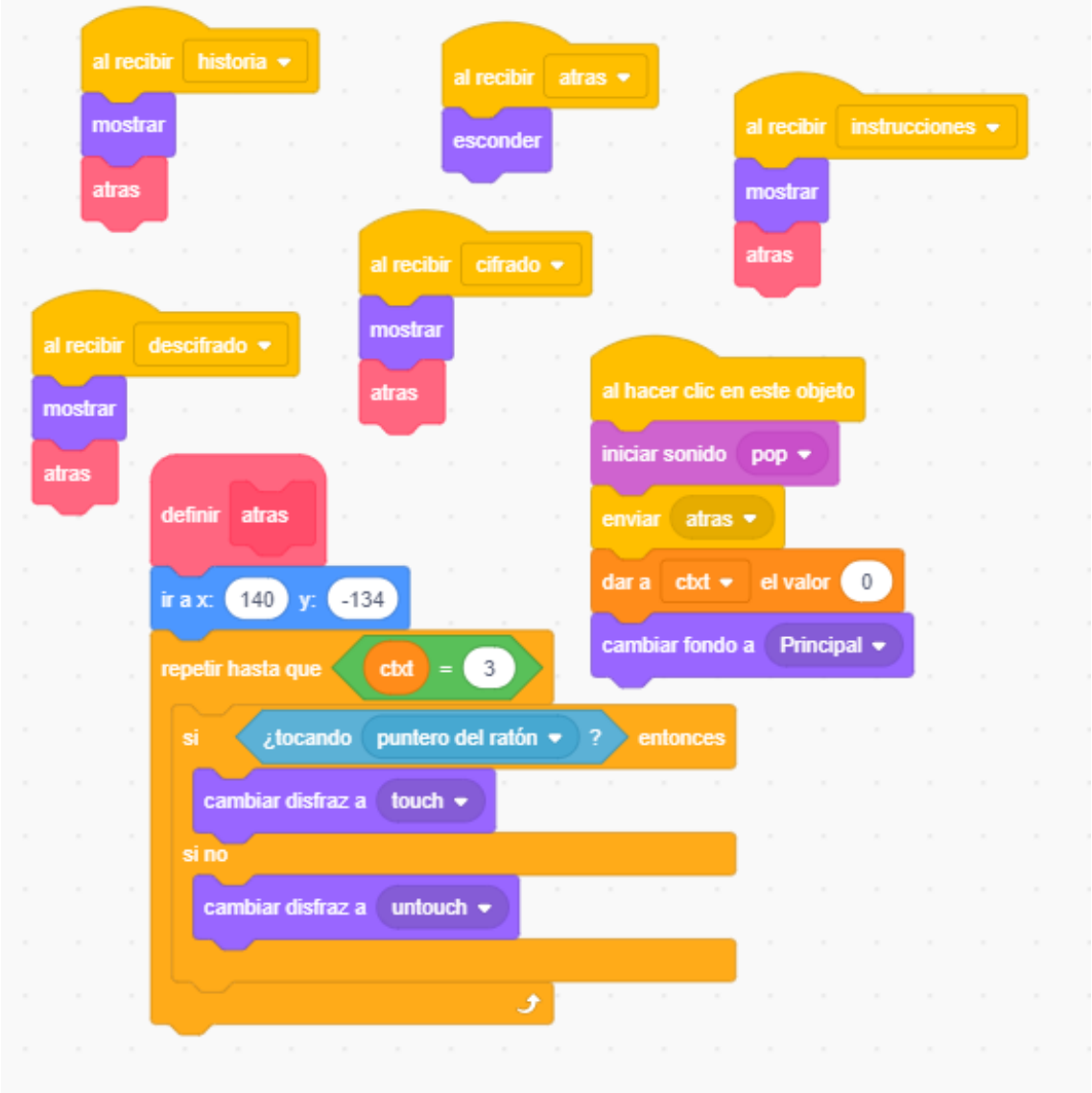


Figura 6.4: Fragmento de código del botón Atrás.

6.3 Posibles ampliaciones

El cifrado ya está completado aunque se puede mejorar añadiendo alguna posibles ampliaciones. La primera ampliación posible sería añadir más alfabetos al cifrado y permitir al usuario poder elegir con que alfabeto quiere cifrar, también se puede añadir alguna opción para cambiar el idioma del programa y dependiendo del idioma que cambia todo el texto del programa al idioma indicado.

Otra posible ampliación posible es poder ver gráficamente la utilización de las reglas del cifrado a la hora de cifrar o descifrar. A la hora de la implementación y la creación

de objetos utilizar los caracteres en mayúscula y en minúsculas es un trabajo prolijo y monótono que ocupa más tiempo del deseado por ello una ampliación sería permitir la introducción de la clave y el mensaje tanto con mayúsculas y minúsculas o combinadas.

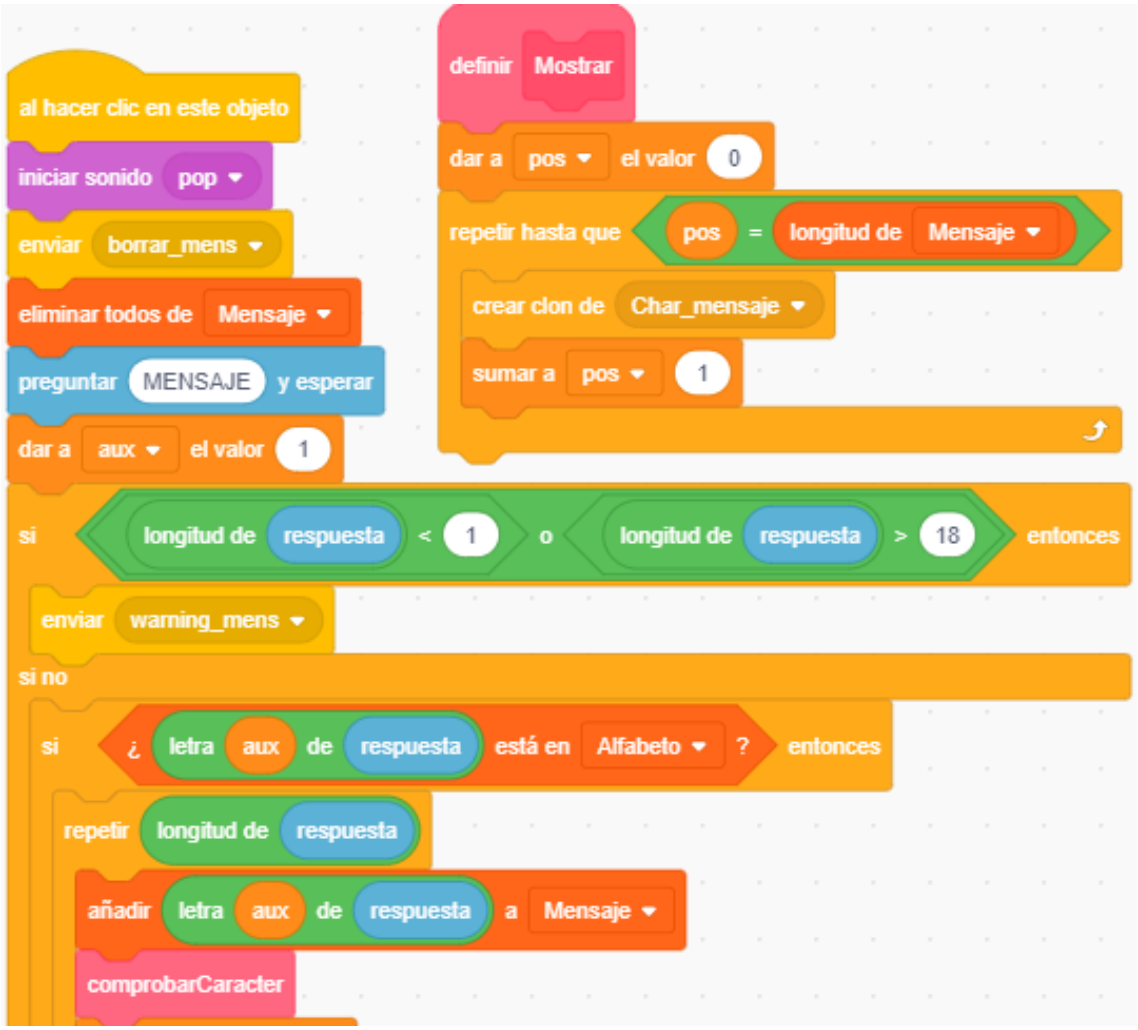


Figura 6.5: Fragmento de código del botón Mensaje.

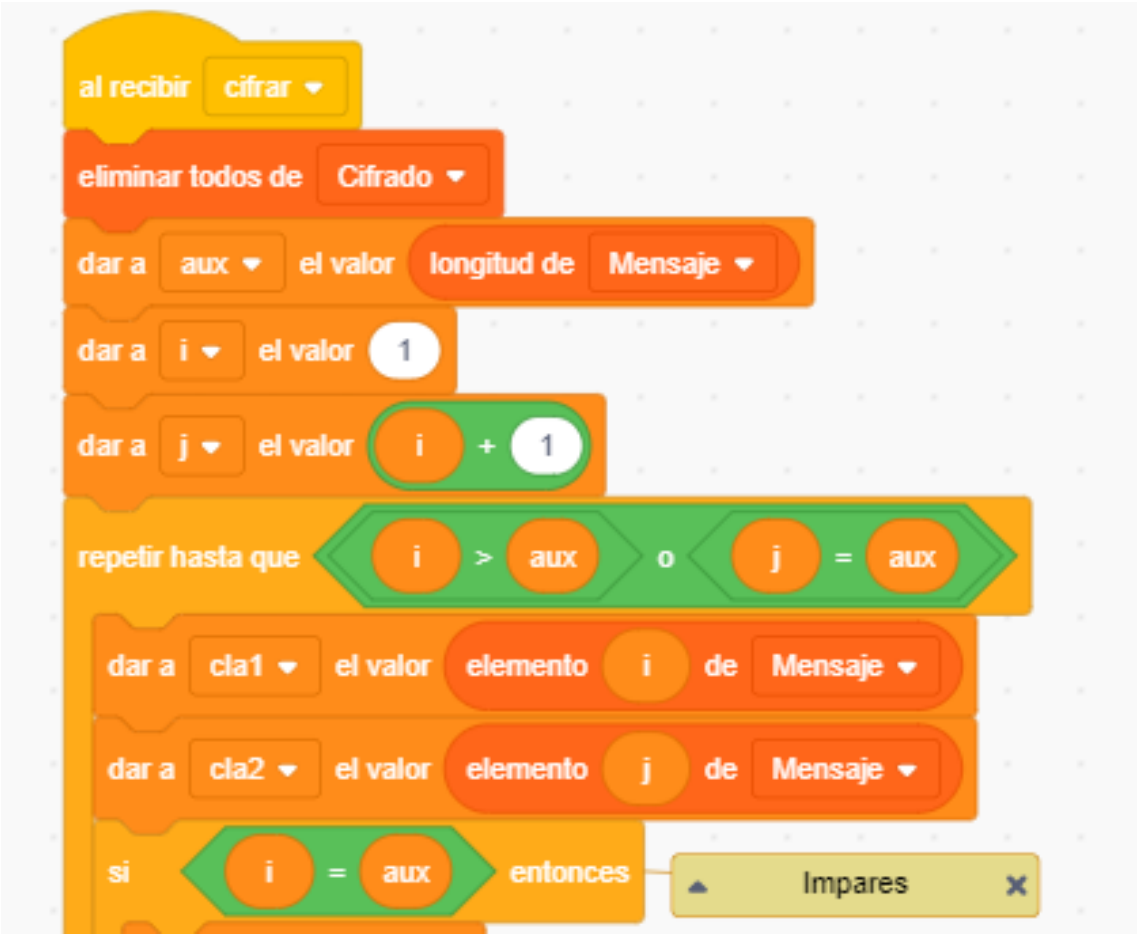


Figura 6.6: Fragmento de código del cifrado.

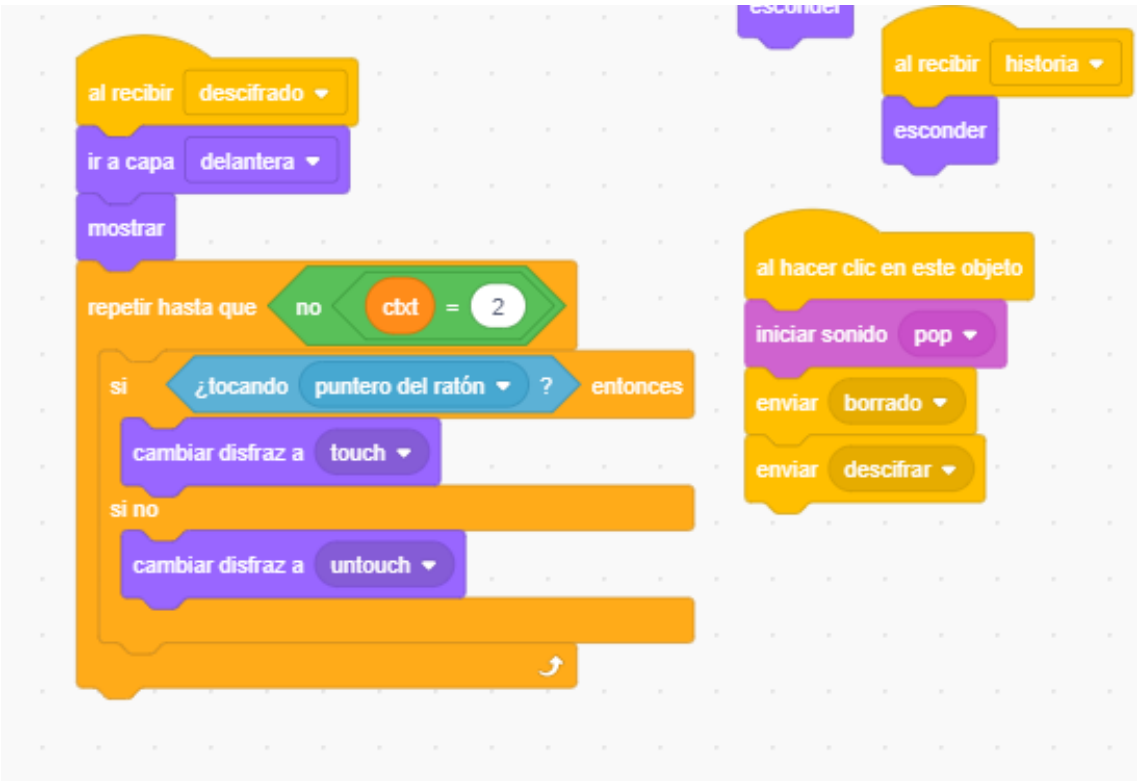


Figura 6.7: Fragmento de código del botón Descifrar.

CAPÍTULO 7

Diseño e implementación del cifrado ADFGVX en Scratch

En este capitulo se va a explicar como se ha implementado el cifrado ADFGVX. Tal y como se explicó en la sección 2.4 se trata de un cifrado monoalfabético que consta de dos partes, una primera parte de sustitución y otra de transposición.

Hay que comentar que en comparación con otros cifrados la implementación de este cifrado ha costado más tiempo del que parecía debido a la dificultad de la creación de las tablas en la parte de la transposición, también debido a que las fuentes bibliográficas son escasas y en muchas no se explica correctamente todo el proceso de cifrado.

7.1 Organización del menú principal

Al acceder al programa nos encontramos una estructura bastante similar a la de los otros programas, en ella nos encontramos un menú principal (véase Figura 7.1) con cuatro botones «Instrucciones», «Cifrado», «Descifrado» y «Historia» que dividen las distintas partes del programa. A continuación se explicará brevemente la funcionalidad de cada uno.

- **Instrucciones.** Cuando pulsamos este botón el escenario cambiará y se nos mostrará una breve explicación sobre como funciona el cifrado cambiando el escenario al llamado «Instrucciones» y enviando el mensaje «instrucciones» que hace que todos los objetos que no estén incluidos en este escenario se oculten.

- **Cifrado.** Este botón junto al de «Descifrado» es el más importante debido a que se introduce al usuario en el escenario donde le permite introducir el mensaje y la clave para cifrar, también muestra el tablero de Polibio con la configuración inicial del cifrado. Al pulsar el botón la variable «cif» cambia el valor a *1* indicando que que estamos en el escenario del cifrado, la variable «pos» cambia el valor a *0* ya que se trata de una variable del código del cifrado, también se eliminan los caracteres de las listas «Clave» y «Mensaje», para finalizar se cambia el fondo al llamado «Cifrado» en el cual no aparece ningún texto que impida la correcta ejecución del cifrado y se envía el mensaje «cifrado» al programa.

- **Descifrado.** Igual que con el botón de «Cifrado» este botón permite al usuario introducir el mensaje y la clave para descifrar, también muestra el tablero de Polibio con la configuración inicial del descifrado. Al pulsar el botón la variable «cif» cambia el valor a *2* indicando que que estamos en el escenario del descifrado, la variable

«pos» cambia el valor a *0* ya que se trata de una variable del código del cifrado, también se eliminan los caracteres de las listas «Clave» y «Mensaje», para finalizar se cambia el fondo al llamado «Cifrado» en el cual no aparece ningún texto que impida la correcta ejecución del cifrado y se envía el mensaje «cifrado» al programa.

- **Historia.** Si queremos conocer una breve historia sobre el cifrado y como este surgió, se pulsará en este botón que cambiará el escenario y nos mostrará dicha historia cambiando el escenario al llamado «Historia» y enviando el mensaje «historia» que mostrará los objetos de este escenario y ocultara los que no estén incluidos.

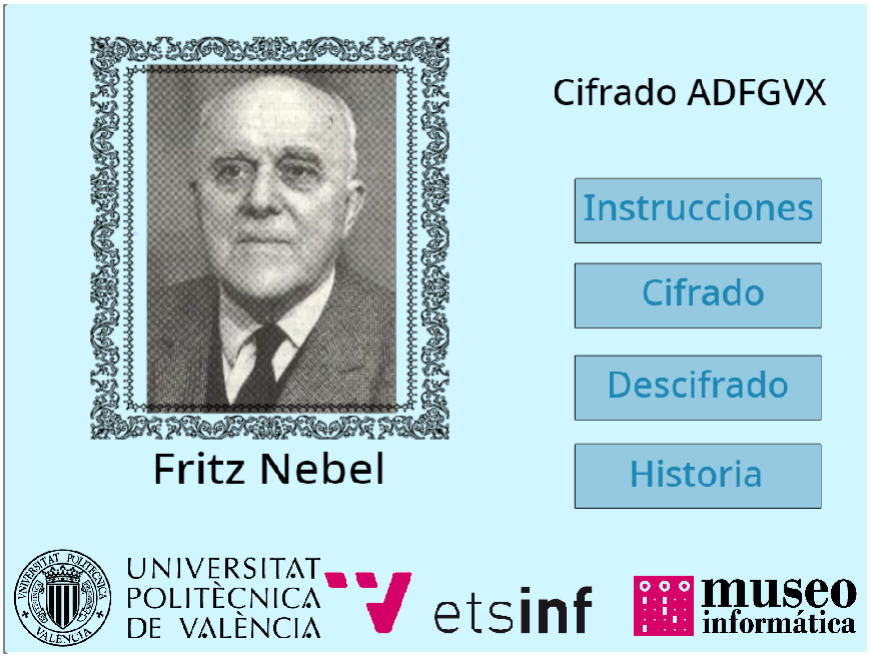


Figura 7.1: Menú principal del cifrado ADFGVX.

7.2 Objetos del cifrado

El cifrado se compone de 27 objetos diferentes, de los cuales solo 7 de ellos proporcionaran funcionalidad al cifrado. A continuación se explicará el funcionamiento.

1. **Tablero de Polibio.** El tablero de Polibio (véase Figura 7.2) en este cifrado solo es demostrativa, es decir solo muestra la configuración inicial que utilizará el cifrado en la parte de la sustitución, sin la posibilidad de cambio en la organización de los caracteres por parte del usuario.

2. **Clave.** Este botón es de los más importantes en este cifrado, la utilidad de este botón es permitir al usuario introducir la clave que quiere utilizar a la hora de cifrar o de descifrar.

   La implementación (véase Figura 7.6) es sencilla, al pulsar en este botón se abre un cuadro de texto que permite al usuario introducir la clave con la cual quiere cifrar o descifrar, dicha clave se guardará en la variable temporal «respuesta», se comprueba que no se hayan introducido caracteres especiales ni símbolos así la variable «respuesta» se guardará carácter a carácter en la lista «Clave» pero si la comprobación es errónea se envía los mensajes «warning_char» y «borrar_clave» y se detendrá el código. Si todo es correcto se procede a comprobar que la longitud

Figura 7.2: Pantalla del cifrado con la tabla, la clave y el mensaje.

de la clave este dentro del rango entre *3* y *6* caracteres. Si la longitud no se encuentra en entre ese rango se enviará el mensaje «warning_mens» que mostrará en pantalla «Introducir Mensaje o Clave válidos (pulsar en info para saber más)» pero si la clave se encuentra dentro de ese rango de caracteres se crearán tantos clones de «Char_clave» como longitud de clave, que se mostrarán en pantalla haciendo visible la clave que ha introducido el usuario.

3. **Mensaje.** Este botón también es de los más importantes en este cifrado, la utilidad de este botón sirve para que el usuario introduzca el mensaje que quiere utilizar en el cifrado o en el descifrado. Hay que hacer una clara división entre el mensaje del cifrado y el del descifrado ya que tienen diferentes características.

   La implementación (véase Figura 7.7) es sencilla, al pulsar en este botón se abre un cuadro de texto que permite al usuario introducir el mensaje con el que cifrar o descifrar,el mensaje introducido se guardará en la variable temporal «respuesta» y se se comprueba que no se han introducido símbolos ni caracteres especiales. Si se han introducido se envían los mensajes «warning_char» y «borrar_mens» y se detendrá el código pero si no se han introducido símbolos o caracteres especiales la variable «respuesta» se guardará carácter a carácter en la lista «Mensaje». Ahora se procede a comprobar que la longitud del mensaje esté dentro del rango entre *6* y *18* caracteres si nos encontramos en el cifrado o dentro del rango entre *la longitud de la clave introducida* y *43* caracteres si nos encontramos en el descifrado, para ello primero se tiene que comprobar que la clave ha sido introducida. Si la longitud del mensaje no se encuentra en ese rango se envía el mensaje «warning_mens» que mostrará el mensaje de advertencia «Introducir Mensaje o Clave válidos (pulsar en info para saber más)» pero si el mensaje si se encuentra dentro de ese rango se crearán tantos clones de «Char_clave» como longitud del mensaje, que se mostrarán en pantalla haciendo visible el mensaje que ha introducido el usuario. Hay que comentar que como el mensaje del descifrado puede estar escrito en bloques de 5 caracteres primero se tienen que eliminar los espacios entre los bloques.

4. **Precifrar/Predescifrar.** Este botón se encuentra en el escenario del cifrado y del descifrado, dependiendo del escenario aparecerá un disfraz y otro. La funcionalidad de

este botón es cambiar el escenario y mostrar al usuario la primera tabla del cifrado, donde se encuentra la clave y la sustitución del mensaje usando la configuración inicial.

La implementación de ese botón (véase Figura 7.8) se divide partes, en la primera parte se comprueba que la clave y el mensaje no estén vacíos y estén dentro de sus respectivos rangos si no es así se envía el mensaje «warning_mens». Ahora se eliminan los espacios del mensaje para que no den problemas a la hora de la transposición y en la creación de la tabla. En la segunda parte se comprueba la variable «cif» si el valor es igual a *1* significa que el usuario se encuentra en el cifrado por lo que envía el mensaje «precifrar» al resto del programa y si es igual a *2* se encuentra en el descifrado por lo que envía el mensaje «predescifrar» al resto del programa. En la tercera parte se envía el mensaje de «borrado» que borrará los caracteres que estén en pantalla y finalmente se enviará el mensaje «tablas» con el que se mostrará la primera tabla del cifrado en pantalla. Al enviar el mensaje «precifrar» o «predescifrar» aparece en pantalla en el mensaje «Procesando ...» indicando al usuario que se esta procesando el precifrado o el precifrado del mensaje.

Cuando el programa recibe el mensaje «Precifrar» se traduce en la creación de las tablas del cifrado. Primero, se crea la primera tabla con la clave y la sustitución del mensaje mediante la configuración inicial. Debido a que Scratch no permite la creación de matrices habrá que introducir todos los caracteres en un lista, en nuestro caso la lista es «tablaPrimaria», en dicha lista se guardará primero la clave y después la sustitución, carácter a carácter, del mensaje original. Segundo, hay que comprobar que la tabla creada esté completa, es decir que la longitud de la tabla sea múltiplo de la longitud de la clave, si no fuese así habrá que añadir tantos caracteres nulos al final de la tabla hasta que dicha condición sea correcta. Tercero, hay que ordenar la clave con respecto la posición que ocupan sus caracteres en el alfabeto, con esta nueva clave se reorganiza la tabla primaria. Por último, se creará una nueva tabla que se guardará en la lista «tablaSecundaria» se trata de una reorganización de las columnas de la primera tabla con respecto a la nueva clave.

Al recibir el mensaje «Predescifrar» el programa hará el proceso contrario, primero reorganizará la clave en función de la posición de sus caracteres en el alfabeto creará la tabla primaria con esa nueva clave y el mensaje introducido por el usuario que se guardará en la lista «tablaPrimaria». Una vez creada esta tabla el programa volverá a organizar la clave a la introducida por el usuario y creará la tabla secundaria con la reorganización por columnas de la tabla primaria con respecto a esta clave que se guardará en la lista «tablaSecundaria».

Tanto la tabla primaria como la secundaria tiene las dimensiones respecto a la longitud de la clave y a la longitud del mensaje, por lo tanto con cada clave y con cada mensaje se crearán tablas con dimensiones diferentes. Al acabar la creación de las dos tablas aparecerá en pantalla la tabla primera.

5. **Siguiente.** La utilidad de este botón es ocultar al usuario los caracteres de la tabla primaria y mostrar los caracteres de la tabla secundaria mostrando en pantalla que se ha cambiado de tabla, primero muestra el mensaje «tabla primaria» y al pulsar muestra el mensaje «tabla secundaria» (véase Figura 7.3).

   La implementación de este botón es simple (véase Figura 7.9), al pulsar el botón se enviará el mensaje «borrado» al resto del programa que eliminará los caracteres que esta en pantalla de la tabla primaria, se cambiará la variable «ctxt» a 2 para indicar que el usuario se encuentra en la tabla secundaria, y para finalizar se enviará los mensajes «tablas» que mostrará por pantalla los caracteres de la nueva tabla y

«mostrar» con el que aparecerán los nuevos botones en pantalla ocultando los que no pertenecen a este escenario.

Figura 7.3: Pantalla con la tabla primaria.

6. **Atrás/Anterior/Fin.** Aunque es un mismo botón se presenta con diferentes disfraces en diferentes escenarios, al pulsar este botón se mostrará al usuario el escenario anterior ocultando los objetos que no pertenezcan al escenario correspondiente.

   La implementación de este botón (véase Figura 7.10) es bastante simple, al pulsar en el botón se comprueba que la variable «ctxt» es *1* o *2* si es así se le dará el valor *0* y se enviará el mensaje «anterior», esto significa, que el usuario ha decidido volver al escenario donde se introducía la clave y el mensaje y que el botón presentaba el disfraz «Atrás». Si la comprobación de la variable es incorrecta se cambiará la variable «ctxt» al valor *0*, se cambiará el valor de la variable «cif» a *0*, también se cambiará el fondo a «Principal» y se enviará el mensaje «atras», eso significa que el usuario quiere volver al menú principal y que el objeto presentaba o el disfraz «Atrás» o el disfraz «Fin».

7. **Cifrar/Descifrar.** Este botón se encuentra en el escenario donde se encuentra la tabla secundaria (véase Figura 7.4). La funcionalidad de este botón es mostrar por pantalla los caracteres del mensaje cifrado o descifrado.

   La implementación del botón (véase Figura 7.11) se basa en comprobar si nos encontramos en cifrado o en el descifrado, para ello comprobamos la variable «cif», si el valor es *0* el usuario se encuentra en el cifrado y por lo tanto se envía el mensaje «cifrar» pero si la variable «cif» tiene el valor *1* el usuario se encuentra en el descifrado y se envía el mensaje «descifrar».

   El programa al recibir el mensaje «cifrar» muestra por pantalla el mensaje «Procesando ...» que indica al usuario que está procesando el cifrado del mensaje. El código del cifrado recorre la tabla secundaria columna a columna, obviando la clave, y carácter a carácter se va guardando en la lista «Cifrado», al necesitar bloques de 5 caracteres para el resultado final se añadirá un espacio cada 5 caracteres. Una vez recorrida toda la tabla se crearán tantos clones de «Char_cifrado» como longitud del «Cifrado» mostrando así el mensaje cifrado en bloques de 5 caracteres.

El programa al recibir el mensaje «descifrar» muestra por pantalla el mensaje «Procesando ...» que indica al usuario que está procesando el descifrado del mensaje. El código del programa recorre la tabla secundaria, a partir de la clave, y sustituye cada par de caracteres por el carácter correspondiente de la configuración inicial, dichos caracteres se guardarán en la lista «Cifrado». Cuando todos los caracteres de la tabla estén sustituidos se crearán tantos clones de «Char_cifrado» como longitud del «Cifrado» y se mostrará por pantalla el mensaje original.

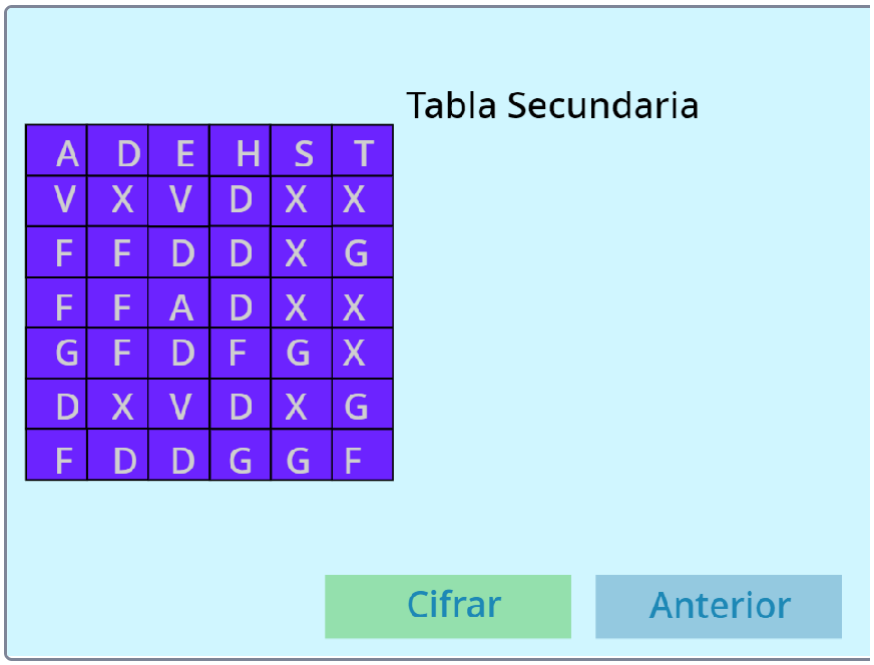


Figura 7.4: Pantalla con la tabla secundaria.

Antes de mostrar tanto el mensaje cifrado como el mensaje original también aparecerá en pantalla la clave introducida por el usuario para la mayor comprensión de la tabla y el mensaje «Mensaje Cifrado» o «Mensaje Descifrado» si se encuentra en el cifrado o en el descifrado respectivamente (véase Figura 7.5).

7.3 Posibles ampliaciones

El programa que realiza el cifrado ya ha sido totalmente implementado aunque hay alguna posibles ampliaciones que se podrían desarrollar para mejorar el programa. La primera ampliación podría ser la modificación por parte del usuario de la configuración inicial en el tablero de Polibio, por lo que un mismo mensaje podría estar cifrado de múltiples formas. Otra ampliación podría ser poder introducir mayúsculas y minúsculas combinadas tanto en la clave como en el mensaje para añadir mayor complejidad al cifrado.

Otra posible ampliación seria poder elegir el idioma de trabajo y que tanto los botones como el alfabeto se adaptaran a ese idioma. Una última ampliación importante podría ser conseguir ver la reorganización de la tabla primaria a la tabla secundaria de manera gráfica para que el usuario viera visualmente el proceso y poder visualizar los cambios presentes entre las dos tablas.

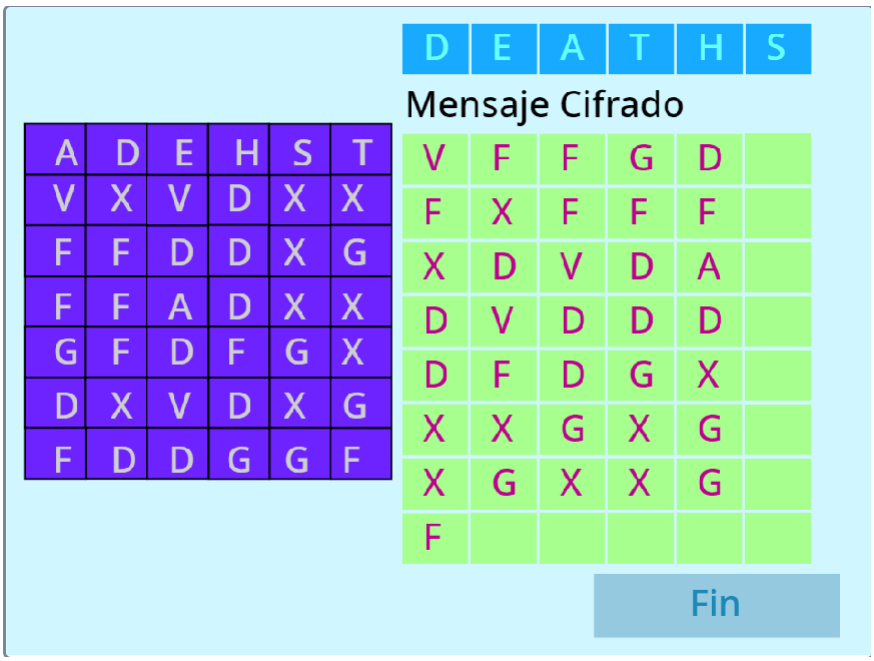


Figura 7.5: Pantalla con el mensaje cifrado.

Figura 7.6: Fragmento de código del botón Clave.

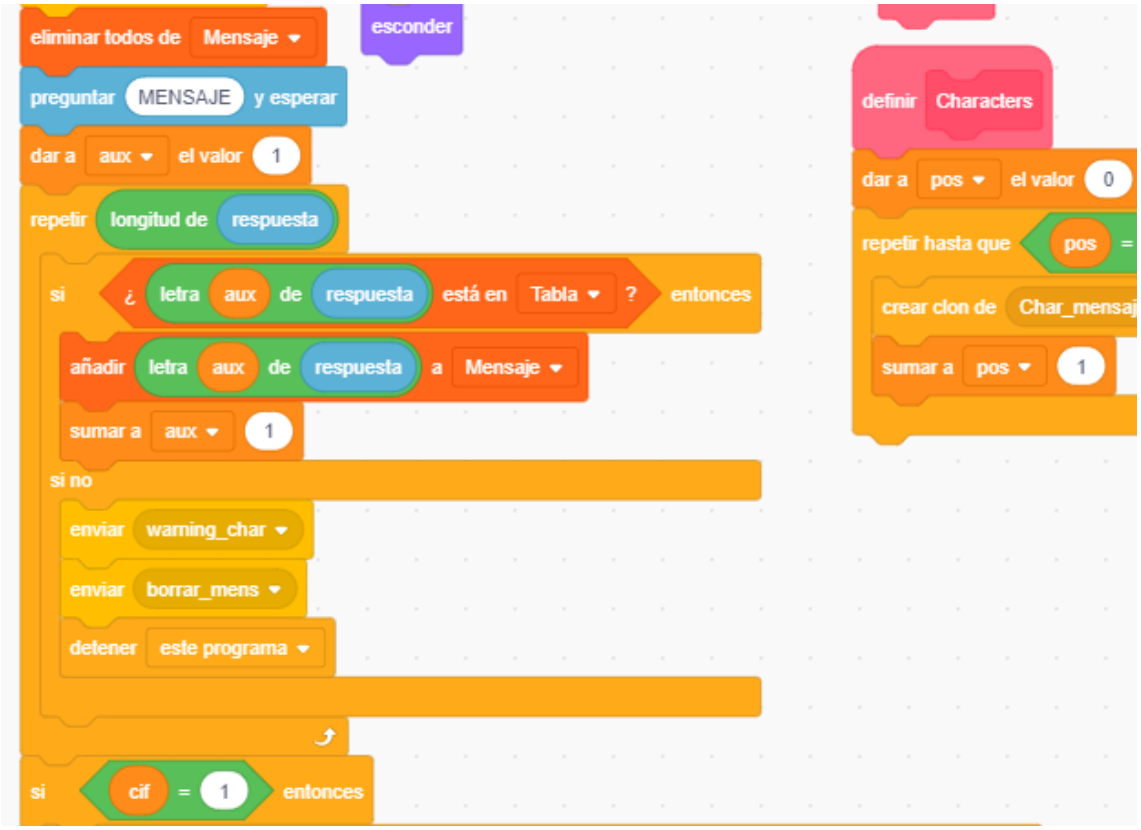


Figura 7.7: Fragmento de código del botón Mensaje.

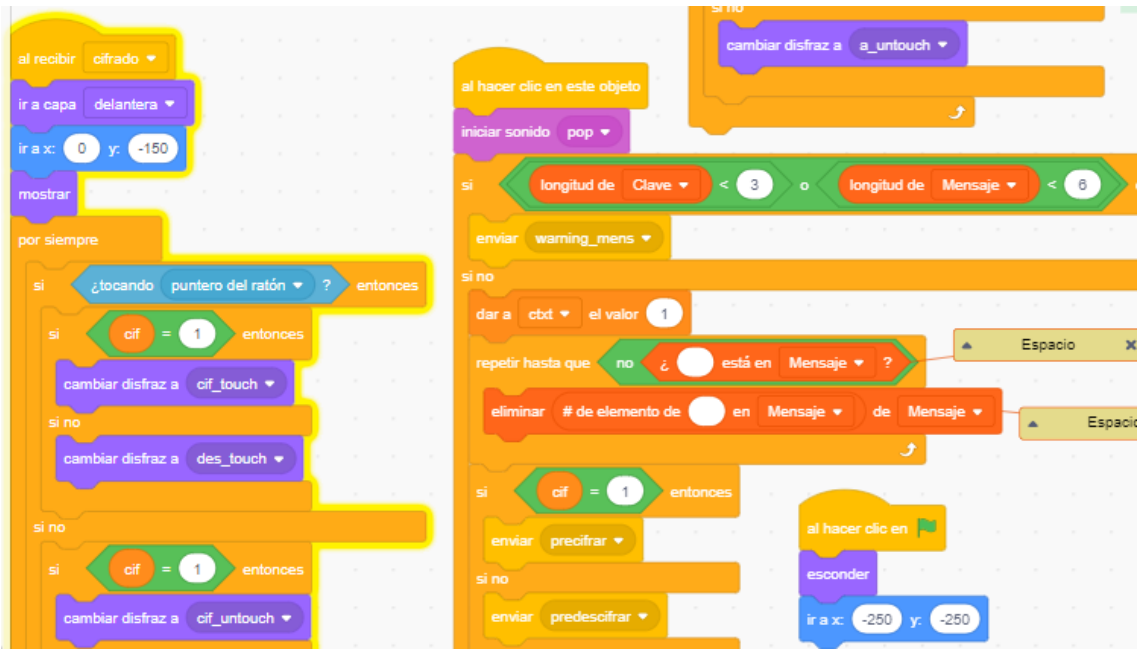


Figura 7.8: Fragmento de código del botón Precifrar.

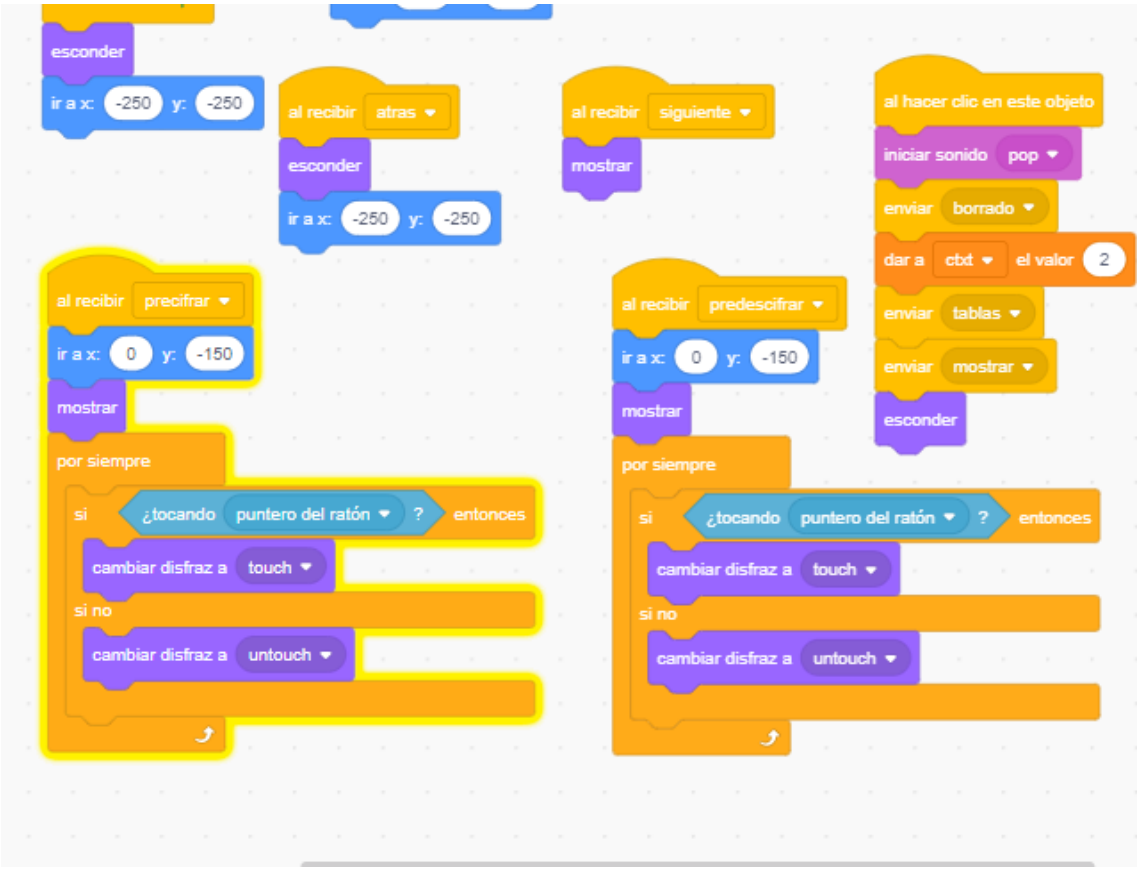


Figura 7.9: Fragmento de código del botón Siguiente.

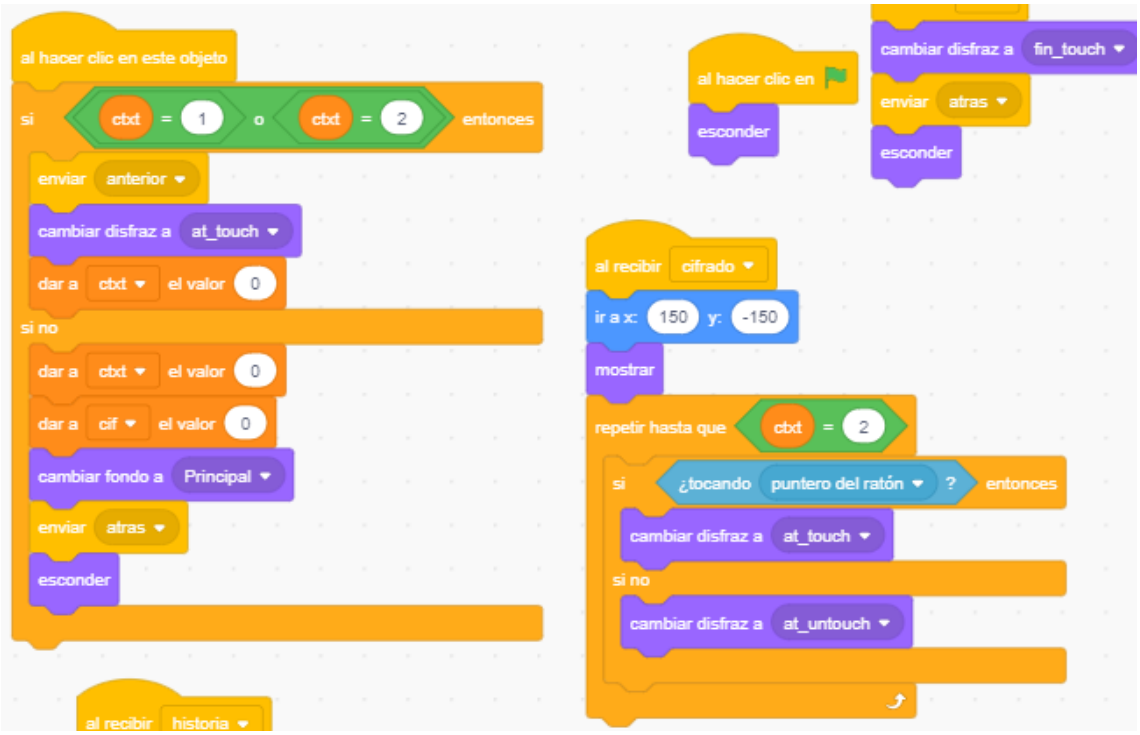


Figura 7.10: Fragmento de código del botón Atrás.

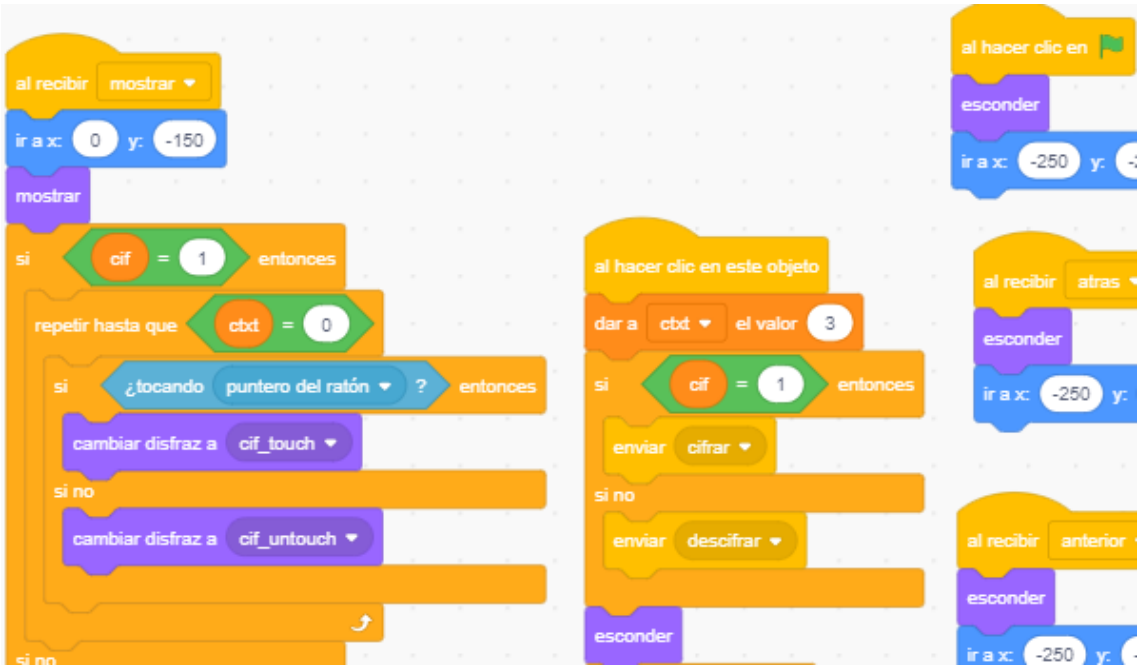


Figura 7.11: Fragmento de código del botón Cifrar.

CAPÍTULO 8
Diseño e implementación de la página web

En este capítulo se va exponer cómo se ha diseñado e implementado la página web del Museo de Informática [1] con los cuatro programa que se han creado en el lenguaje de programación Scratch. En la página web tendrá una breve descripción de cada unos de los cifrados, las instrucciones que el usuario tendrá que seguir para ejecutar el programa en Scratch y el mismo programa creado con un enlace a la página de Scratch, donde está compartido con otros usuarios y donde se pueden ver los comentarios y el código interno que permite la modificación y creación de una nueva versión del programa.

8.1 Estructura de la página web

Como se comentó en el primer capítulo de este trabajo, el principal propósito de este trabajo es la creación de cuatro cifrado clásicos utilizados en la historia mediante el lenguaje de programación Scratch. Además de una página web explicativa que se incluirá en la página web del Museo de Informatica de l'Escola Tècnica Superior d'Enginyeria Informàtica con el objetivo de que las personas que visiten el museo puedan disfrutar de una parte de la historia de la criptografía conociendo cómo funcionan estos cuatro cifrados y animar a los jóvenes visitantes con pocos o nulos conocimientos en criptografía o programación a dar sus primeros gracias al sencillo lenguaje de programación como Scratch.

Si nos fijamos en la Figura 8.1 podemos observar cómo la estructura de la página web esta dividida en apartados correspondientes a cada cifrado. En cada uno de estos apartados se incluye una breve descripción del cifrado incluso una muy breve historia, las instrucciones que debe seguir el usuario cuando desee ejecutar el programa del cifrado en Scratch. Posteriormente se encuentra el proyecto de Scratch embebido en la página web donde el usuario puede comprobar el funcionamiento del cifrado, también se encuentra un enlace a la página web de Scratch donde se encuentra el proyecto donde cualquier usuario puede realizar comentarios sobre el proyecto y revisar el código interno para aprender como se ha implementado e incluso realizar una nueva versión cambiando alguna característica o incluso añadir alguna funcionalidad.

En la parte final de la página se encuentra el apartado «Notas» donde se ha introducido el nombre del creador de la página web y de los programas en Scratch y el nombre del tutor de este trabajo.

[1] Página web del museo `museo.inf.upv.es`.

Figura 8.1: Captura de la página web.

8.2 Tecnologías empleadas

Ahora se explicarán las tecnologías empleadas en la creación de la página web tanto cuales son sus características como que utilidad proporcionan a la página web. La página web ha sido creada en el programa Visual Code Studio, un entorno de programación de la empresa Microsoft. Hay que comentar que este entorno de desarrollo está disponible tanto para Windows, Linux y macOS y que permite el desarrollo en múltiples lenguajes de programación como pueden ser C++, Java, Phyton, entre muchos otros hasta los más recientes como Clojure, F#, o Razor. En la Figura 8.2 se puede ver un fragmento del código de la página web.

Para nuestra página web se han utilizados las tecnologías HTML, CSS y Javascript, que son las comúnmente utilizadas. Cada una de estas tecnologías que se han empleado ha proporcionado una característica concreta la página web dentro del proceso de creación. A continuación se explicará cual es la finalidad de cada una de las tecnologías:

- **HTML.** Acrónimo de Lenguaje de Marcas de HiperTexto (en inglés *HyperText Markup Language*), sirve de estándar para la creación de paginas web y se basa en una etiquetas rodeadas por corchetes angulares que marcan el inicio y el fin de cada objeto que el navegador (Microsoft Edge, Chrome, Firefox, ...) interpreta y da forma para que el usuario pueda visualizar correctamente.

  El uso de esta tecnología en la creación nuestra página web ha servido para dar estructura tanto al texto como al resto de contenido que se ha introducido.

```
</header>
<!-- end-header --><section id="headline" style="">
    <div class="container">
      <h3 style="">Arqueología Informática: Implementación de sistemas clásicos de cifrado en Scra
    </div>
</section>
      <div class="breadcrumbs-w"><div class="container"><div id="crumbs"><a href="http://museo.inf

<section id="main-content" class="container">
<!-- Start Page Content -->
<div class="row-wrapper-x"><div class="row-wrapper-x" style="text-align: left;">
<p style="text-align: center;"><span style="color: #d60066;"><strong><span style="font-size: 2em
<h3 style="text-align: center; color: #404040;"><strong>Abel Esteve Romero</strong></h3>
<p>Sección creada por Abel Esteve Romero. Realizaremos un recorrido histórico en la criptografía d
el siglo XV con el Disco de Alberti hasta el siglo XX  con el cifrado ADFGVX comentando también el
de Blaise de Vigenère (año 1553) y el cifrado Playfair (año 1854). La finalidad de este trabajo es
puedan conseguir aprender como funcionan los cifrados y en que contexto histórico se crearon. Adic
se añade un pequeño programa donde se puede probar el cifrado.</p>
```

Figura 8.2: Fragmento de código de la página web.

- **CSS.** Acrónimo de *Cascading Style Sheets* (en español hojas de estilo en cascada), se trata de un lenguaje que define las hojas de estilo, es decir define el aspecto final de la página web. Aunque HTML tiene características para cambiar el aspecto de la página que se este creando, este lenguaje esta creado para separar el contenido de la página y su presentación. CSS permite definir todos los aspectos gráficos como son las fuentes, los márgenes, los colores, las imágenes, entre muchos otros.

  En nuestra página web esta utilizada para darle un aspecto característico y visual aunque no aporte ninguna característica extra.

- **JavaScript.** Es un lenguaje de programación que se utiliza en páginas HTML para crear funciones que se ejecutarán y añadirán dinamismo a la página web que se este creando. Al ser un lenguaje que funciona desde el lado del cliente, no necesita el uso de compiladores, el mismo navegador es capaz de interpretar código en JavaScript. Hay que comentar que también se puede ejecutar desde el lado del servidor.

  En la creación de nuestra página web se ha utilizado está tecnología para añadirle funciones que serán las encargadas de actuar con el servidor en el que esta alojada la página web.

CAPÍTULO 9
Conclusiones

En este último capítulo se propondrán las conclusiones finales a que se han alcanzado en la realización de este trabajo. Se expondrá un resumen final donde se explicarán los objetivos alcanzado, también se incluirá un apartado donde se explicarán algunas consideraciones para un trabajo futuro.

9.1 Conclusiones finales

Como se ha podido demostrar con este trabajo es posible, con un lenguaje de programación orientado a los jóvenes como es Scratch, la creación de cifrado simple en la actualidad pero con cierta complejidad y que supusieron una revolución histórica en la historia de la criptografía en el periodo que fueron desarrollados. Para la creación e implementación de estos cifrados en el lenguaje Scratch, se ha tenido que realizar trabajo de investigación sobre el momento y contexto histórico de su creación y como han influido a la historia de la criptografía. Todos los artículos y publicaciones utilizados en este trabajo se pueden encontrar en la bibliografía.

Como se explicó en el capítulo 3, el lenguaje de programación Scratch tiene como objetivo animar e introducir a los usuarios noveles en el mundo de la programación, ya que se trata de un lenguaje intuitivo basado en bloques, que no requiere escribir código como tal sino que cada bloque contiene una funcionalidad y el usuario al agrupar u organizar los bloques en un mismo conjunto, como si fueran distintos bloques de *LEGO* ensamblados entre sí, genera un fragmento de código totalmente funcional. Estos fragmentos de código creados por el usuario forman parte de un objeto en el entorno de programación Scratch. Estos objetos, creados por el usuario o predefinidos por Scratch, se comunican entre ellos, de forma que cada vez que se añade un objeto al proyecto este puede añadir una nueva característica. La funcionalidad final del proyecto finalizado se origina al tener todos los fragmentos de código, de cada objeto, interactuando entre sí.

Si bien es cierto que Scratch facilita el aprendizaje en el mundo de la programación respecto a otros lenguaje de programación de más alto nivel, resulta ser un lenguaje limitado y para la creación de algunos programas no es la mejor elección ya que se puede tardar más intentado adaptar la implementación de ese programa a este lenguaje de programación que implementar esa solución en otro lenguajes de programación. En nuestro caso nos hemos encontrado con dos limitaciones. La primera de ellas es que a la hora de comprobar un carácter guardado en una variable, Scratch no detecta si esta en mayúsculas o en minúsculas, por ello se ha decidido implementar todas las variables en mayúsculas y avisar de ello al usuario para que no cometa errores. La segunda es a la

hora de cifrar los mensajes: se ha tenido que limitar la longitud de estos porque Scratch no posee una *scrollview*[1] que haría falta para cifrar mensajes largos.

Lo primero que se ha realizado en este trabajo ha sido introducir el contexto histórico en el que se sitúa cada uno de los cifrados, para poder ver como ha afectado la creación de los cifrados a la historia de la criptografía, también se ha explicado el funcionamiento de cada uno así como un breve ejemplo. Seguidamente se ha razonado el motivo por el cual se ha escogido el lenguaje de programación Scratch para la creación de nuestros programas y qué características y limitaciones nos ofrece a la hora de crear nuestro programas y compartirlos con otros usuarios.

Una vez se ha explicado el motivo por el cual se ha elegido el entorno de programación Scratch, se ha detallado cómo se han implementado los diferente cifrados en el entorno. Primero describiendo como se estructuraba el cifrado correspondiente y después explicando detalladamente cada objeto importante que aporta funcionalidad al cifrado con su respectivo fragmento de código. Finalizando con unas posibles ampliaciones que podrían llevarse a cabo en cada cifrado.

Finalmente se ha descrito la página web del Museo de Informatica que se ha creado para alojar una pequeña descripción de cada cifrado, las instrucciones para el funcionamiento del programa y el mismo programa. Por otra parte también se incluye un enlace a la página donde está compartido el proyecto para que los usuarios puedan comentar el proyecto y acceder al código interno para saber cómo funciona o añadirle alguna nueva funcionalidad con la creación de una nueva versión del programa.

Se espera que tanto el presente trabajo como la página web animen y ayuden a los usuarios sin experiencia o con pocos conocimientos de programación a iniciarse en el mundo de la programación o en el mundo de la criptografía de un modo divertido y didáctico gracias a la manera de compartir conocimientos y proyectos del lenguaje Scratch.

9.2 Trabajo futuro

Para finalizar este último capítulo hay que mencionar que en los capítulos donde se explica el diseño e implementación de cada cifrado (capítulos 4, 5, 6 y 7) se ha incluido una sección final donde se comentan posibles ampliaciones que se pueden realizar a los programas. Algunas de estas ampliaciones añadirían funcionalidad al cifrado para llegar a un número mayor de usuarios. En la actualidad existen diversos cifrados que se usaron a lo largo de la historia, por lo tanto, se podría considerar cifrados clásicos. Cualquier usuario con un mínimo dedicación al mundo de la criptografía clásica y con conocimientos de Scratch podría implementar en dicho entorno. Uno de estos cifrado podría ser la Cifra de Delastelle que presenta dos variantes que se explicarán a continuación.

Félix Delastelle, nombrado anteriormente por la variante *Four Square* del cifrado Playfair, describió alrededor del año 1901 el cifrado bífido y el cifrado trífido (en inglés *bifid cipher* y *trifid cipher*). Se trata de dos cifrados donde el uso de la clave es opcional ya que es necesario una configuración inicial en las tablas que puede conseguirse con la inclusión de la clave o de forma totalmente al azar, se podría decir que se trata de cifrados entre el cifrado Playfair y el cifrado ADFGVX.

El cifrado bífido se basa en una matriz 5x5 (o 6x6, depende del alfabeto utilizado) con una configuración inicial como se puede ver en la tabla 9.1. El mensaje se sustituye carácter a carácter por la posición que ocupa en tabla, primero el eje de ordenadas y luego el eje de abscisas. Esta sustitución se coloca debajo del mensaje original en dos

[1] Funcionalidad que poseen los lenguajes de alto nivel que permite al usuario poder desplazar el documento de manera vertical u horizontal para poder visualizar todo el contenido.

filas, en la primera fila la posición del carácter en el eje de ordenadas y en la segunda el eje de abscisas. Para finalizar se agrupan las posiciones en parejas y se sustituyen por el carácter correspondiente en la tabla, primero se agrupa la primera fila, si se diera el caso el último dígito de la primera fila se agrupa con el primero de la segunda y después la segunda fila. Un ejemplo del cifrado se puede observar en la tabla 9.2.

	1	2	3	4	5
1	B	G	W	K	Z
2	Q	P	N	D	S
3	I	O	A	X	E
4	F	C	L	U	M
5	T	H	Y	V	R

Tabla 9.1: Tabla del cifrado bífido.

m	e	n	s	a	j	e	d	e	p	r	u	e	b	a
4	3	2	2	3	3	3	2	3	2	5	4	3	1	3
5	5	3	5	3	1	5	4	5	2	5	4	5	1	3
L	P	A	O	O	V	I	E	Y	Y	Z	M	S	M	W

Tabla 9.2: Mensaje de prueba en cifrado bífido.

El cifrado trífido se basa en tres matrices 3x3 con una configuración inicial. El mensaje se sustituye carácter a carácter por la tabla y la posición que ocupa en tabla, primero el eje de ordenadas y luego el eje de abscisas como en la tabla 9.3. Esta sustitución se coloca debajo del mensaje original en tres filas, en la primera fila la tabla en la que se encuentra el carácter, en la segunda fila la posición del carácter en el eje de ordenadas y en la tercera el eje de abscisas.

Para finalizar se agrupan las posiciones en tríos y se sustituyen por el carácter correspondiente en la tabla correspondiente, primero se agrupa la primera fila, si se diera el caso el último dígito o últimos dos dígitos de la primera fila se agrupa con los dos primero o el primero de la segunda y después la segunda fila, y si se diera el caso el último dígito o últimos dos dígitos de la segunda fila se agrupa con los dos primero o el primero de la tercera y finalmente la tercera fila. Un ejemplo del cifrado se puede observar en la tabla 9.4.

1	1	2	3	**2**	1	2	3	**3**	1	2	3
1	F	E	L	1	S	T	B	1	O	P	Q
2	I	X	M	2	C	G	H	2	U	V	W
3	A	R	D	3	J	K	N	3	Y	Z	&

Tabla 9.3: Tablas del cifrado trífido.

m	e	n	s	a	j	e	d	e	p	r	u	e	b	a
1	1	2	2	1	2	1	1	1	3	1	3	1	2	1
2	1	3	1	3	3	1	3	1	1	3	2	1	1	3
3	2	3	1	1	1	2	3	2	2	2	1	2	3	1
E	T	F	Q	I	B	D	A	R	L	W	F	K	C	J

Tabla 9.4: Mensaje de prueba en cifrado trífido.

En los dos cifrados se ha mostrado un ejemplo con el mensaje *mensaje de prueba*, sin espacios, en la tabla del cifrado bífido se ha utilizado una configuración inicial totalmente al azar y en las tablas del cifrado trífido se ha usado la clave *FELIX MARIE DELASTELLE* que al no tener que repetir caracteres se ha quedado como clave final *FELIXMARDST*.

Presentados estos cifrados sería interesantes que los usuarios que leyeran este trabajo se animarán a implementarlos en el entorno Scratch con el objetivo de ampliar la biblioteca de proyectos relacionados con la criptografía clásica que se pueden alojar en la página web del Museo de Informatica de l'Escola Tècnica Superior d'Enginyeria Informàtica para que los visitantes tengan un mayor abanico de proyectos en Scratch relacionado con los cifrados clásicos que descubrir y aprender.

Bibliografía

[1] Scratch - Imagine, Program, Share. Consultar en https://scratch.mit.edu/.

[2] Discuss Scratch. Consultar en https://scratch.mit.edu/discuss/.

[3] Paint Editor - Scratch Wiki. Consultar en https://en.scratch-wiki.info/wiki/Paint_Editor.

[4] Caballero, P. (1997) *Introducción a la Criptografia*. Madrid: Ra-ma.

[5] Christensen, Chris. (2010) *Cryptography of the Vigenère Cipher*. Northern Kentucky University. Consultado en http://cort.as/-JeKI el 18 de abril de 2019.

[6] Christensen, Chris. (2006) *Polygraphic Ciphers*. Northern Kentucky University. Consultado en http://cort.as/-JeKc el 18 de abril de 2019.

[7] Christensen, Chris. (2006) *ADFGVX Cipher*. Northern Kentucky University. Consultado en http://cort.as/-JeKi el 18 de abril de 2019.

[8] Christensen, Chris. (2015) *Fractionating Ciphers*. Northern Kentucky University. Consultado en http://cort.as/-JeLY el 18 de abril de 2019.

[9] Granados, G. (10 de julio de 2006). Introducción a la criptografía. *Revista Digital Universitaria*, 7(7):1-17. Consultado en http://cort.as/-JYuB el 09 de abril de 2019.

[10] Kahn, D. (1996). *The Codebreakers: The comprehensive history of secret communication from ancient times to the internet*. New York: Scribner.

[11] Le Danny Yang. (2016). *Arqueología informática: la criptografía clásica con Scratch*. Trabajo Fin de Grado.Escola Tècnica Superior d'Enginyeria Informàtica. Univesitat Politècnica de València.

[12] McManus, S. (2013) *Scratch programming in Easy Steps*. Leamington Spa: East Steps Limited.

[13] Prudencio, M. (2013). Una herramienta lúdica de iniciación a la programación, Scratch. *Linux Magazine*, 28:78-82.

[14] Serrano, G. (2012). *Programación Scratch para niños*. United States: Createspace

[15] Singh, S. (2000). *Los códigos secretos el arte y la ciencia de la criptografía, desde el antiguo Egipto a la era Internet*. New York: Anchor.

[16] Smith, L. D. (2012). *Cryptography - the science of secret writing*. United Kingdom : Lightning Source UK Ltd.

[17] Soler Fuensanta, J. R., & López-Brea Esplau, F. J. (2016). *Mensaje secretos: la historia de la criptografía española desde sus inicios hasta los años 50*. Valencia: Tirant lo Blanch.

[18] Wing, J.M. (2006). Computational Thinking. Viewpoint. *Communications of the ACM,* 49(3):33-35.

APÉNDICE A

Implementación del cifrado Vigenère en Java

```
import java.util.Scanner;
public class Vigenere
{
  public static void main (String g[])
  {
    Scanner sc = new Scanner(System.in);
        char matriz[][] = new char [27][27];
    char abecedarioM[] = {'a','b','c','d','e','f','g','h','i','j','k','l','m','
        n','o','p','q','r','s','t','u','v','w','x','y','z'};


        String mensaje="";
    String maxClave="";
    String clave="";

    System.out.println("");
    System.out.println("Enter message:");
    mensaje=sc.nextLine();
    System.out.println("");
    mensaje=mensaje.toLowerCase();
    System.out.println("Enter key:");
    clave=sc.nextLine();
    System.out.println("");
    clave=clave.toLowerCase();

    int [] mensajeP= new int[mensaje.length()];
    int [] claveP= new int[mensaje.length()];
    int [] cifrado= new int[mensaje.length()];
    char [] cifradoFinal= new char[mensaje.length()];

    int numero=1;
    String opc="";
    System.out.println("Select one option:");
    System.out.println("0: Show Tabula Recta");
    System.out.println("1: Cipher");
    System.out.println("2: Decipher");
    System.out.println("");
    opc=sc.nextLine();

    if ((opc.equals("0")|opc.equals("Show Tabula Recta"))){
      System.out.println("");
      for (int i=0;i<27;i++){
          for (int j=0; j<27; j++){
                    matriz[i][j]=abecedarioM[numero-1];
          numero++;
```

```
45                      if(numero>27) { numero=1; }
46          }
47                  numero++;
48            if(numero>27) { numero=1; }
49              }
50
51          for (int x=0;x<27;x++) {
52            for (int y=0; y<27;y++){
53              System.out.print(matriz[x][y] + " ");
54            }
55                   System.out.println("");
56              }
57      }
58      else if ((opc.equals("1")|opc.equals("Cipher"))){
59
60              for (int x=0;x<=mensaje.length()-1;x++){
61                  maxClave+=clave.charAt(x %clave.length());
62        }
63        System.out.println("");
64        System.out.println(mensaje);
65              System.out.println(maxClave);
66
67        for (int x=0; x<mensaje.length(); x++){
68          for(int y=0; y<abecedarioM.length; y++){
69          if(abecedarioM[y]==mensaje.charAt(x)) { mensajeP[x]=y+1; }
70          if(abecedarioM[y]==maxClave.charAt(x)) { claveP[x]=y+1; }
71          }
72          cifrado[x]= (mensajeP[x] + claveP[x]) %abecedarioM.length ;
73        }
74
75        for (int i=0; i<cifrado.length; i++){
76          for(int j=0; j<abecedarioM.length; j++){
77            if(cifrado[i]==j){
78            cifradoFinal[i]=abecedarioM[j-1];
79            }
80          }
81        }
82
83        System.out.println("");
84              System.out.println("Cipher text");
85
86      }
87      else if ((opc.equals("2")|opc.equals("Decipher"))){
88
89        for (int x=0;x<=mensaje.length()-1;x++){
90                      maxClave+=clave.charAt(x %clave.length());
91        }
92
93        System.out.println("");
94        System.out.println(mensaje);
95              System.out.println(maxClave);
96
97        for (int x=0; x<mensaje.length(); x++){
98          for(int y=0; y<abecedarioM.length; y++){
99          if(abecedarioM[y]==mensaje.charAt(x)) { mensajeP[x]=y+1; }
100         if(abecedarioM[y]==maxClave.charAt(x)) { claveP[x]=y+1; }
101         }
102         if((mensajeP[x] - claveP[x])>=0){
103         cifrado[x]= (mensajeP[x] - claveP[x]) %abecedarioM.length ;}
104         else{
105           cifrado[x]= (mensajeP[x] - claveP[x]+abecedarioM.length) %
                abecedarioM.length ;
106         }
107       }
```

```
108         for (int i=0; i<cifrado.length; i++){
109           for(int j=0; j<abecedarioM.length; j++){
110             if(cifrado[i]==j){
111             cifradoFinal[i]=abecedarioM[j-1];
112             }
113           }
114         }
115                 System.out.println("");
116                 System.out.println("Clear text");
117       }
118       else{
119         System.out.println("Wrong option");
120         System.out.println("");
121       }
122         System.out.println(cifradoFinal);
123         System.out.println("");
124       }
125 }
```