Universidad del Valle de Guatemala
Facultad de Ingeniería

Laboratorio #2
Esquemas de detección y corrección de errores

Marco Antonio Jurado Velasquez 20308
Cristian Eduardo Aguirre Duarte 20231

Introducción

Los esquemas de detección y corrección de errores son utilizados en comunicación de datos para garantizar la integridad de la información . Estos algoritmos permiten detectar si ha ocurrido algún error durante la transmisión o el almacenamiento como el algoritmo CRC-32 y, en algunos casos, como lo es con el algoritmo de Hamming incluso corregirlos.

El algoritmo de Hamming es de detección y corrección de errores desarrollado por Richard Hamming. Este algoritmo realiza la adición de bits de paridad a un mensaje para detectar y así poder corregir errores de un solo bit en la cadena de mensaje codificada. Se colocan o se suman estos bits en casillas específicas para así poder diferenciar a la hora de encontrar un error en que bit específico se encuentra el error. Es utilizado por ejemplo en la memoria RAM de las computadoras.

CRC (Cyclic Redundancy Check) es un algoritmo de detección de errores que utiliza operaciones matemáticas de división. CRC-32 es una variante común del algoritmo que utiliza polinomios de 32 bits para calcular el valor de comprobación. En este caso se utilizó la variante vista en el curso. Al recibir el mensaje, el receptor también calcula el valor de verificación y lo compara con el valor recibido. Si ambos valores coinciden, se asume que el mensaje no tiene errores. CRC-32 se utiliza en sistemas de almacenamiento.

Resultados

Para la sección de resultados se utilizaron 9 tramas similares en ambos algoritmos y además 1 trama específica por algoritmo pues de este modo se comprobaría la incertidumbre del respectivo algoritmo. Las tramas utilizadas son:

Trama original (sin hacer cambios en la trama codificada)

- 1001
- 1100
- 1010

Tramas modificadas para detectar errores en un bit (se hace cambio de un bit en la trama codificada

- 1101
- 1011
- 1111

Tramas modificadas con dos bits para detectar errores (se hace cambio de dos bits en la trama codificada).

- 10011
- 11011
- 11001

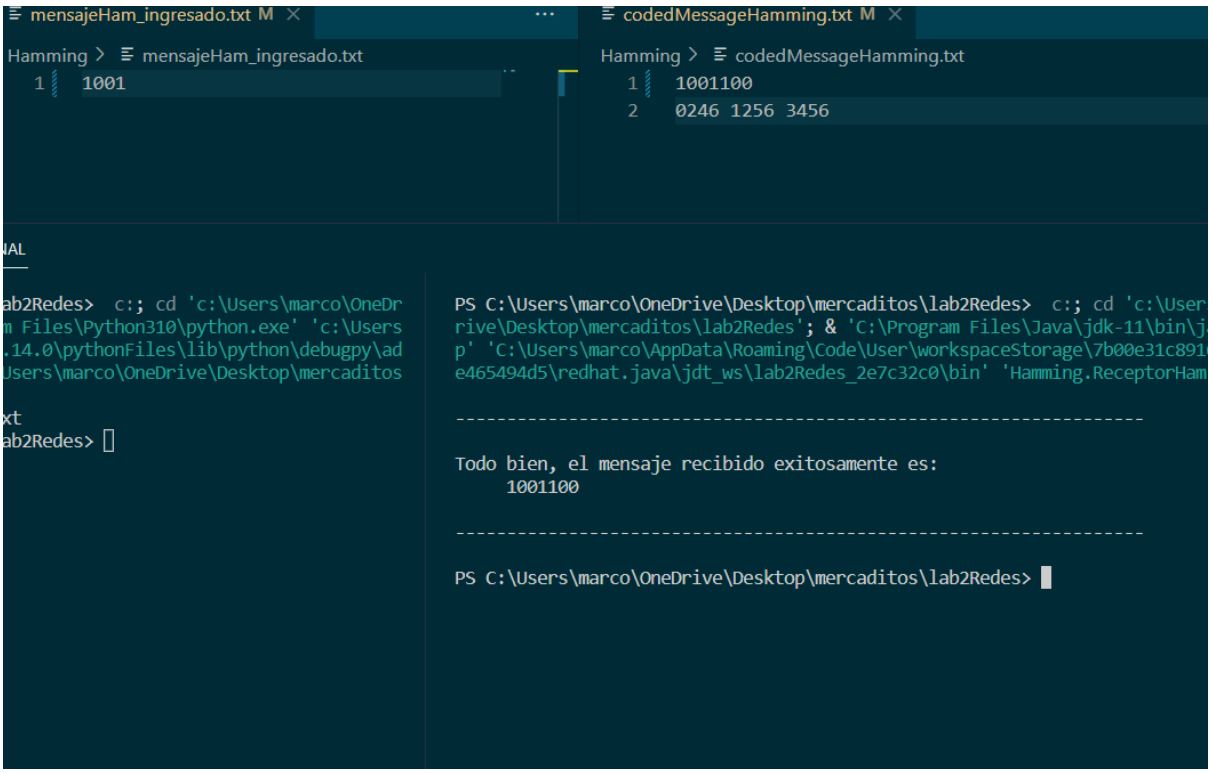
Tramas específicas para comprobación de fallas en algoritmo

- Hamming
    - 1010
- CRC-32
    - XXXXX

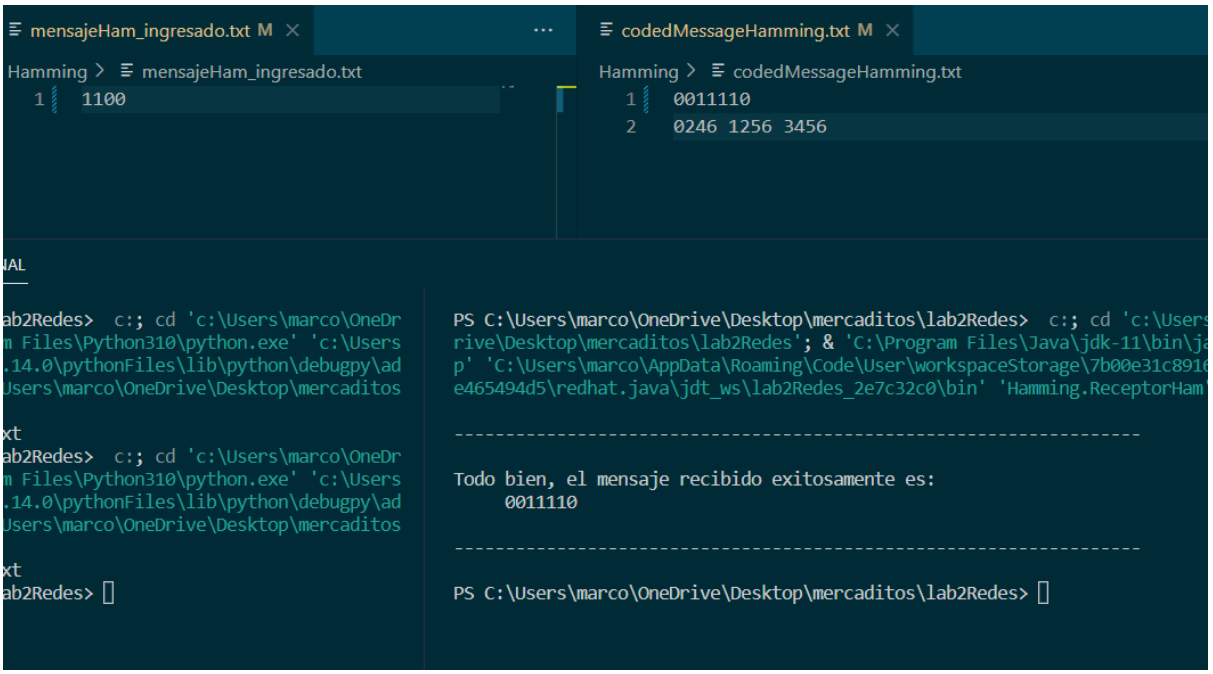
Hamming

Trama original

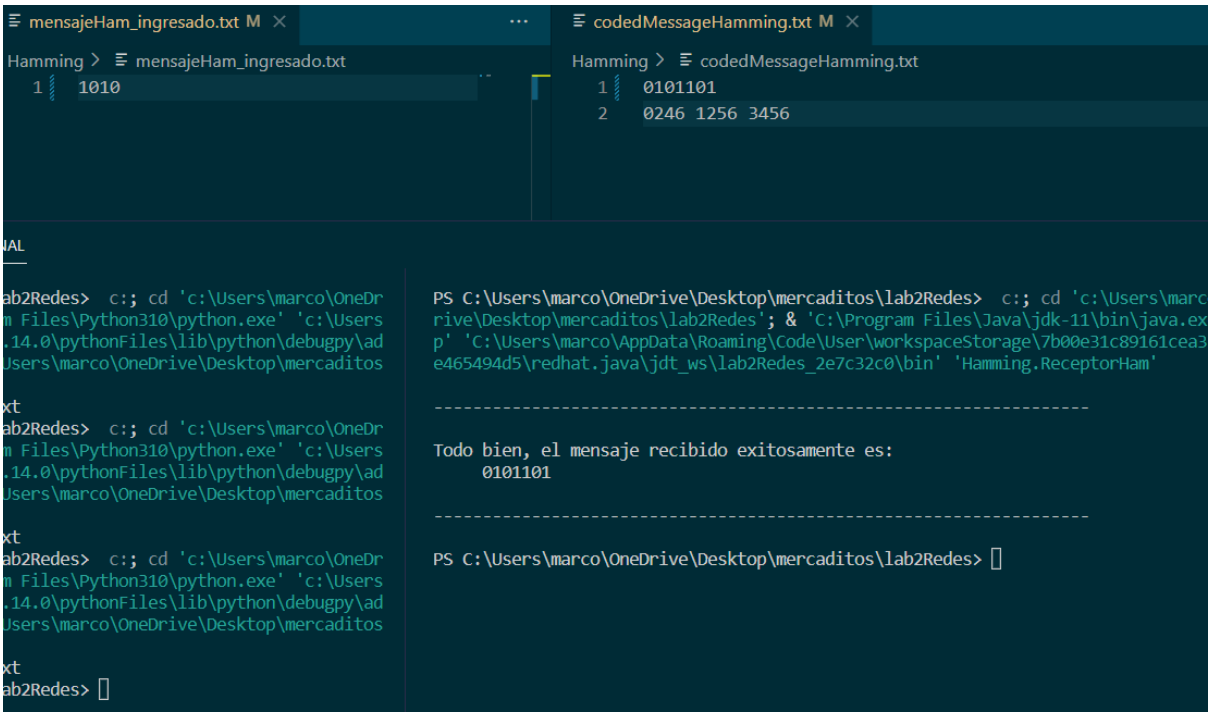
- **1001**
    - **Trama codificada generada: 1001100**
    - **Resultado obtenido previo a procedimiento de recepción**



- **1100**
  - **Trama codificada generada: 0011110**
  - **Resultado obtenido previo a procedimiento de recepción**

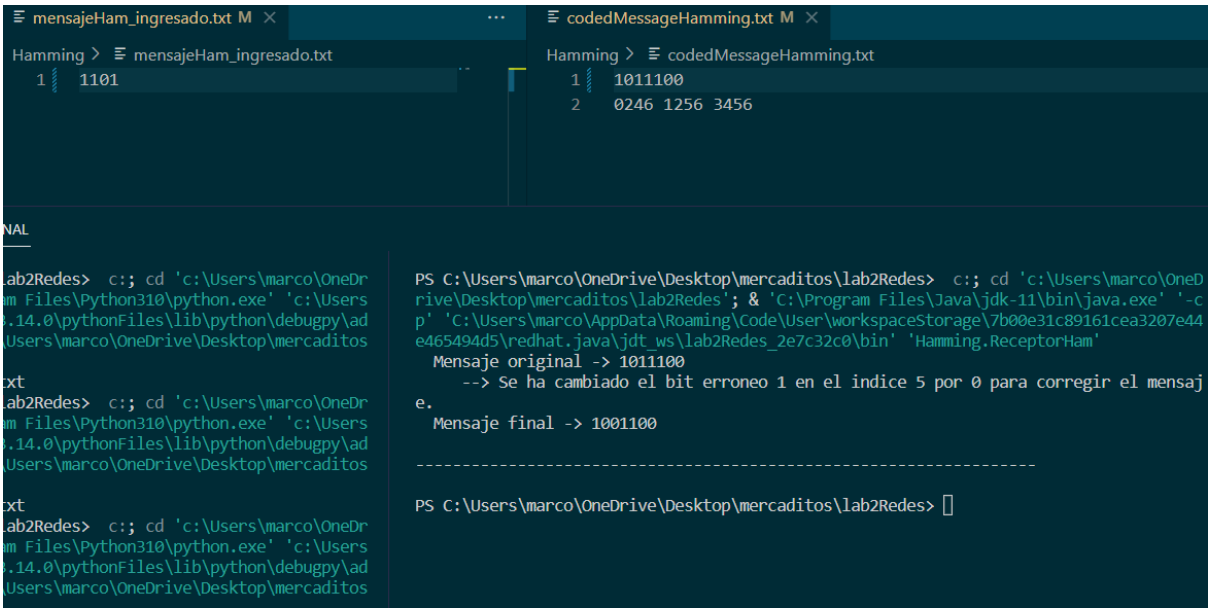


- **1010**
  - **Trama codificada generada: 0101101**
  - **Resultado obtenido previo a procedimiento de recepción**

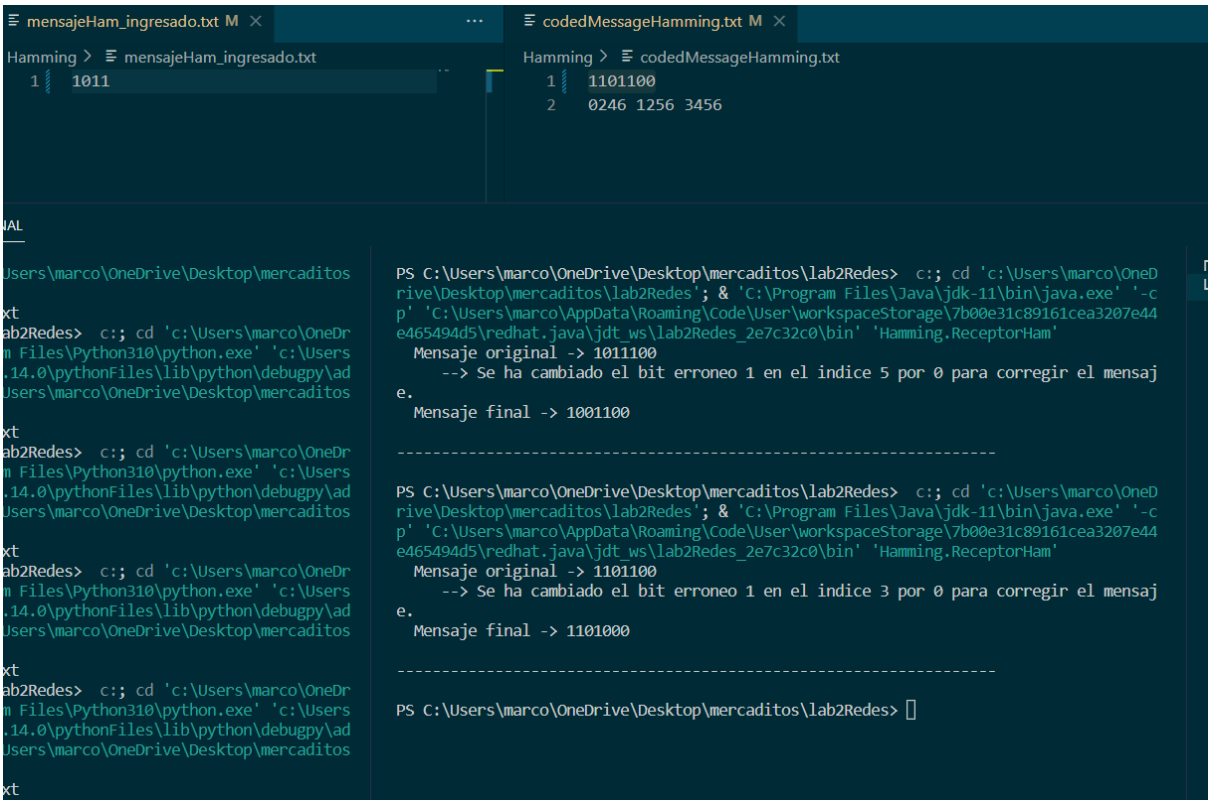


Tramas modificadas para detectar errores.

- **1101**
  - **Trama codificada generada: 1010100 -> 1011100**
  - **Resultado obtenido previo a procedimiento de recepción**

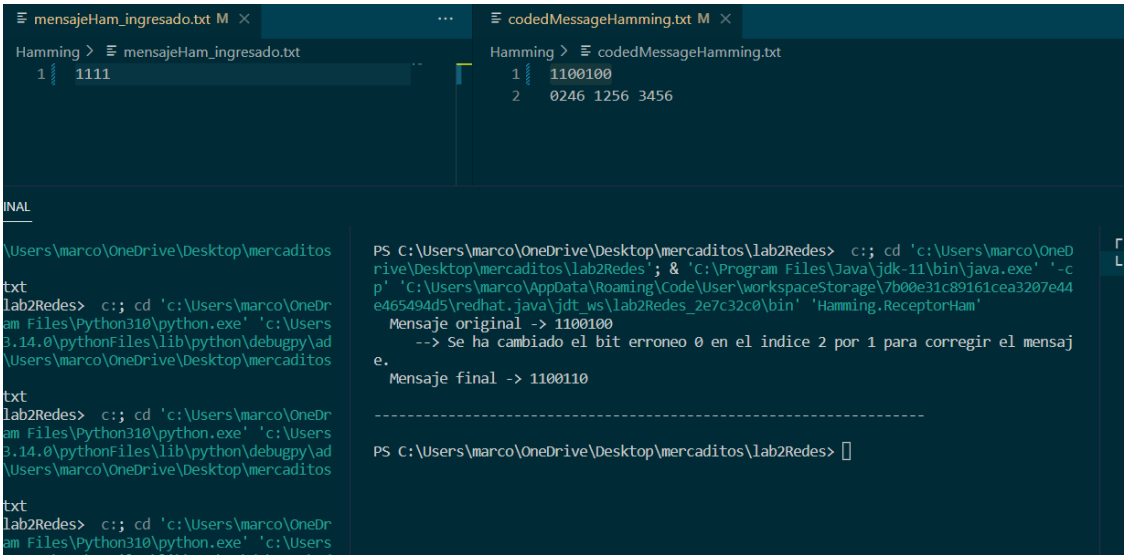


- **1011**
  - **Trama codificada generada: 1100100 -> 1101100**
  - **Resultado obtenido previo a procedimiento de recepción**



En este caso se realizó un cambio de un bit pero el valor obtenido posterior a decodificar no es el mismo al mensaje recibido. Esto no se debe a una mala implementación pero a una falla en el algoritmo donde al verificar los bits de paridad y realizar los cambios se determina que el mensaje final codificado satisface las reglas del algoritmo. Esto es una de las fallas del algoritmo de hamming.

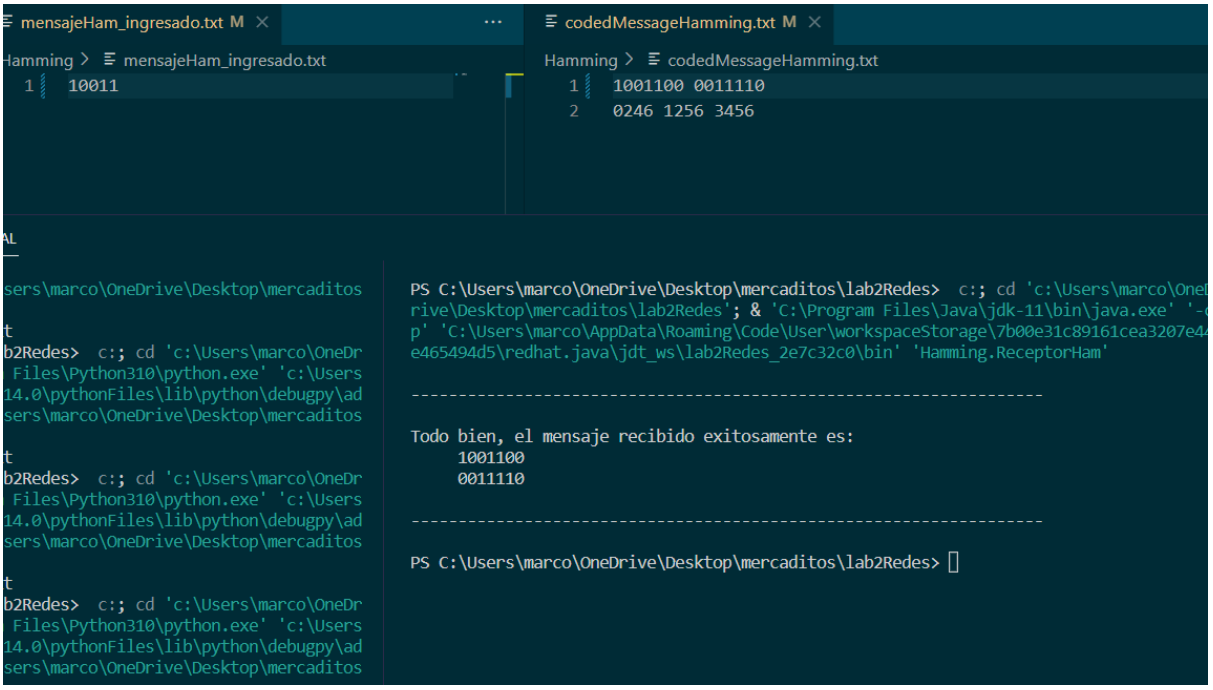
- **1111**
  - **Trama codificada generada: 1110100 -> 1100100**
  - **Resultado obtenido previo a procedimiento de recepción**



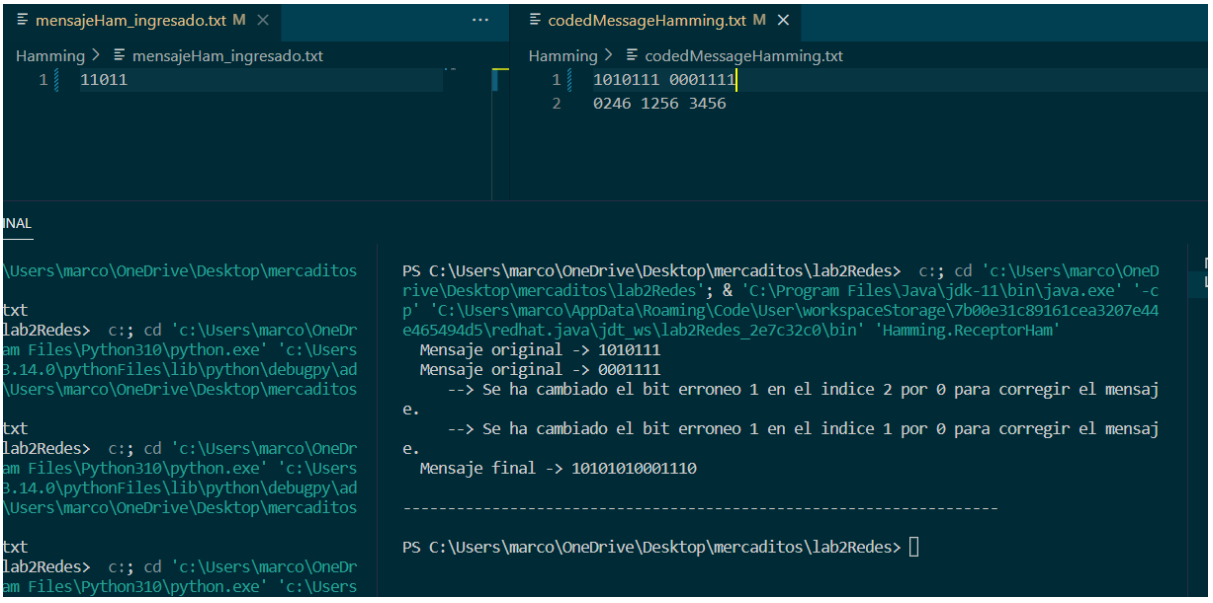
De igual manera que la trama anterior al revisar los bits de paridad podemos ver que la trama generada satisface las reglas del mismo algoritmo.

Tramas modificadas con dos bits para detectar errores.

- **10011**
  - **Trama codificada generada: 1001100 0001111-> 1001100 0011110**
  - **Resultado obtenido previo a procedimiento de recepción**

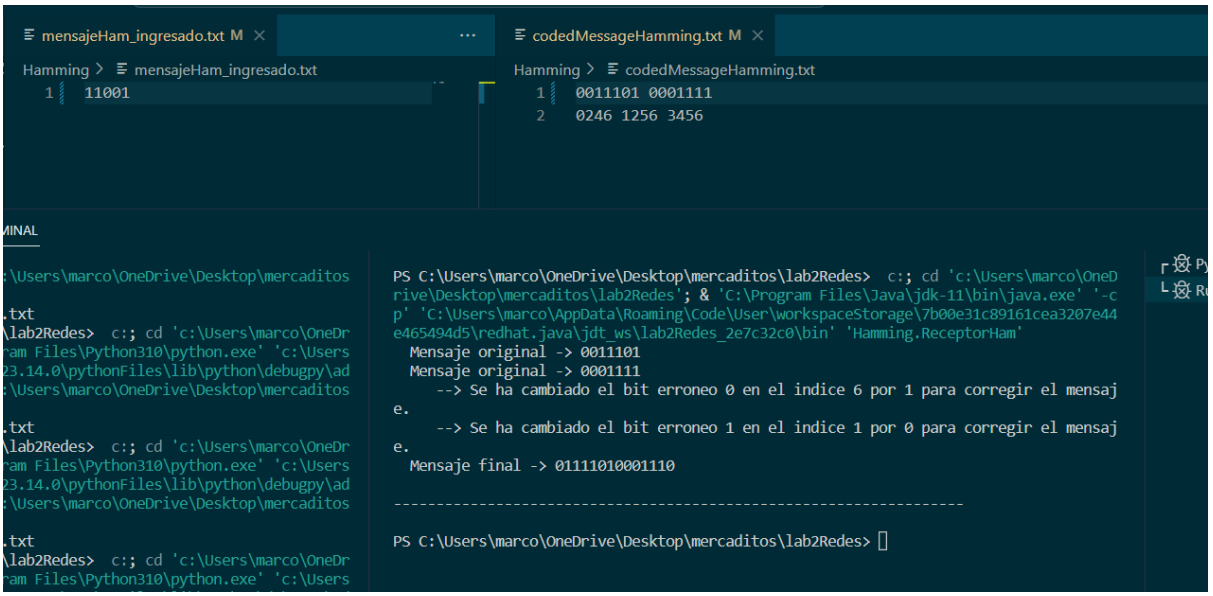


- **11011**
  - **Trama codificada generada: 1010100 0001111 -> 1010111 0001111**
  - **Resultado obtenido previo a procedimiento de recepción**



- **11001**

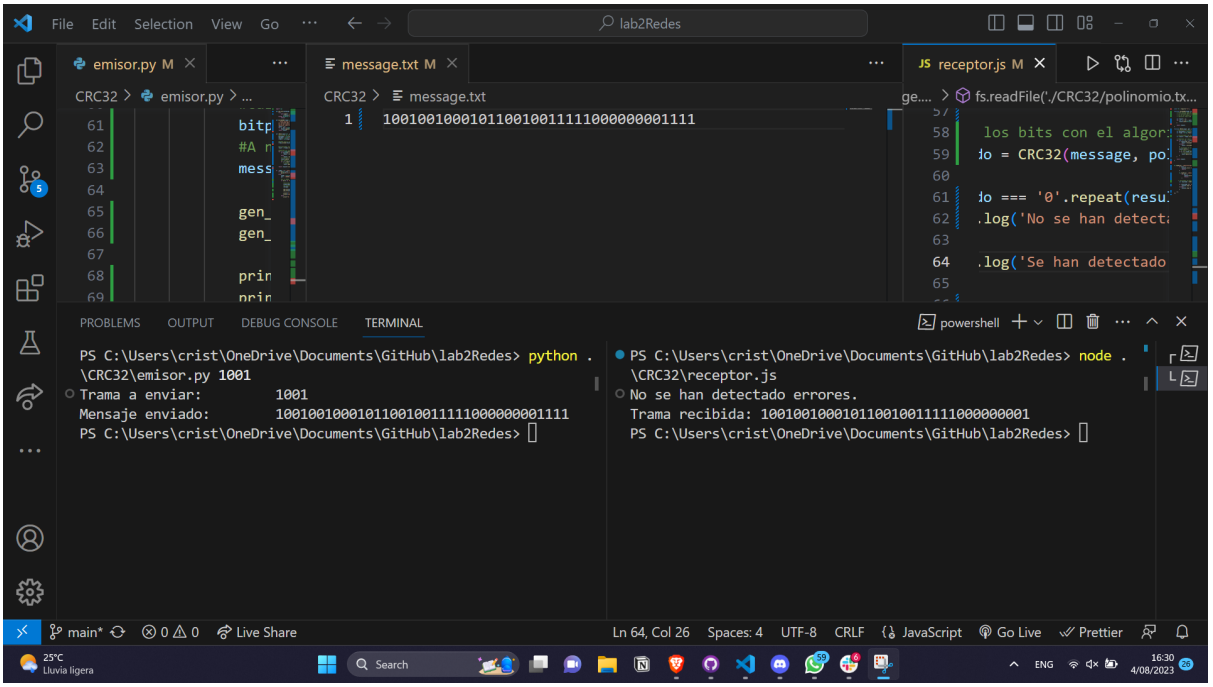
- **Trama codificada generada: 0011110 0001111 -> 0011101 0001111**
- **Resultado obtenido previo a procedimiento de recepción**



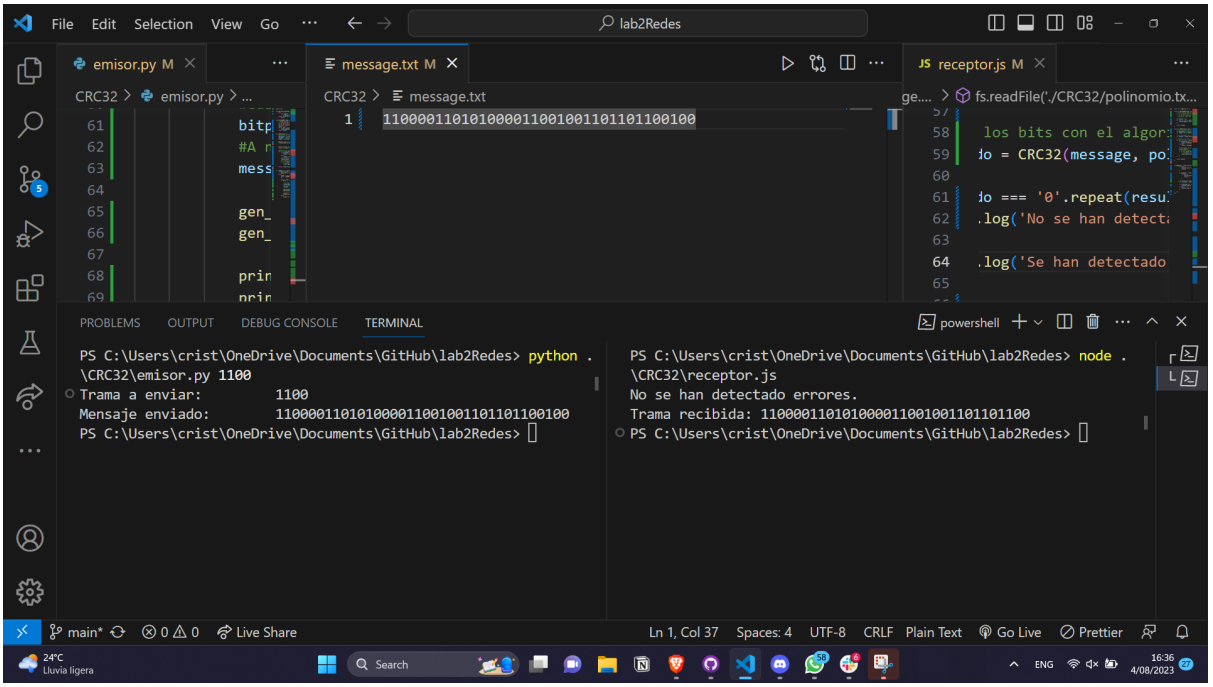
CRC-32

Trama original

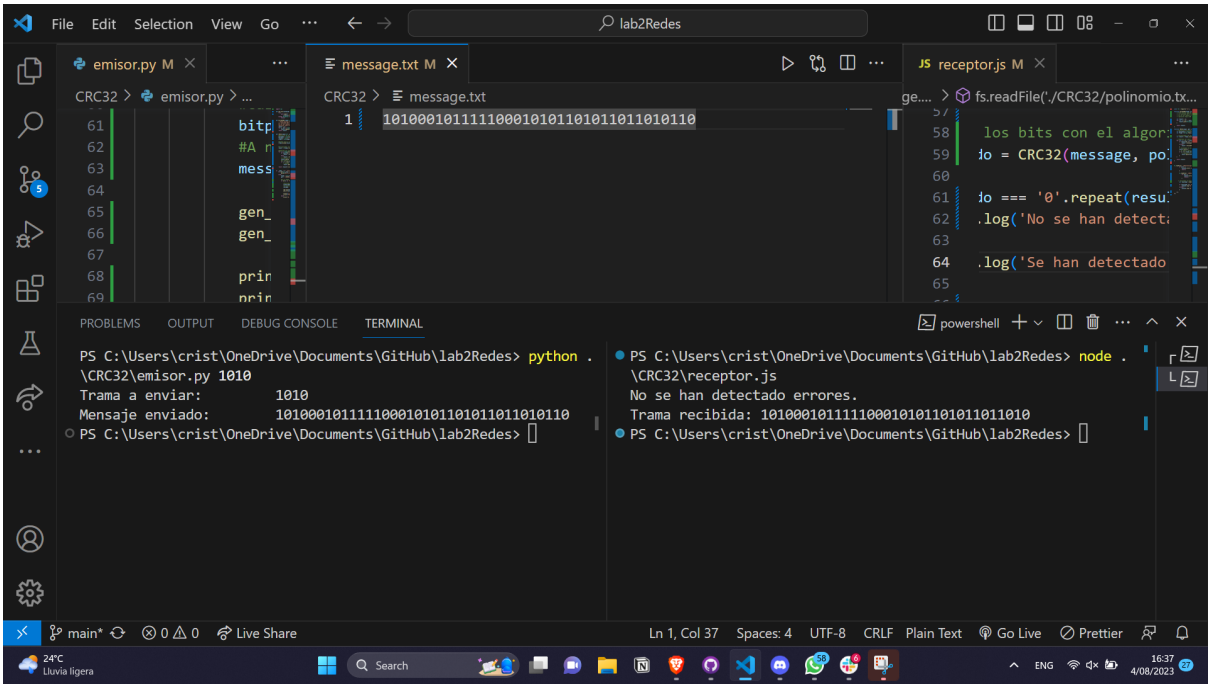
- **1001**
  - **Trama codificada generada:**
    10010010001011001001111100000000111 1



- **1100**
  - **Trama codificada generada:**
    11000011010100001100100110110110010 0

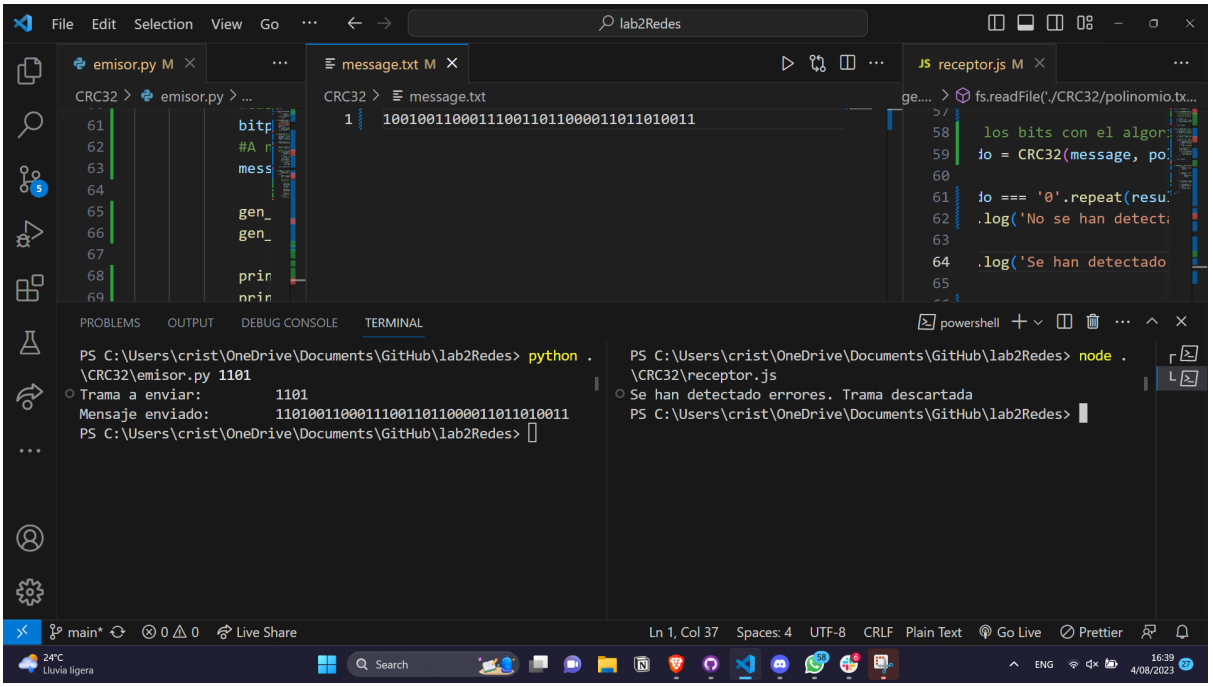


- **1010**
  - **Trama codificada generada:**
    10100010111110001010110101101101011 0110

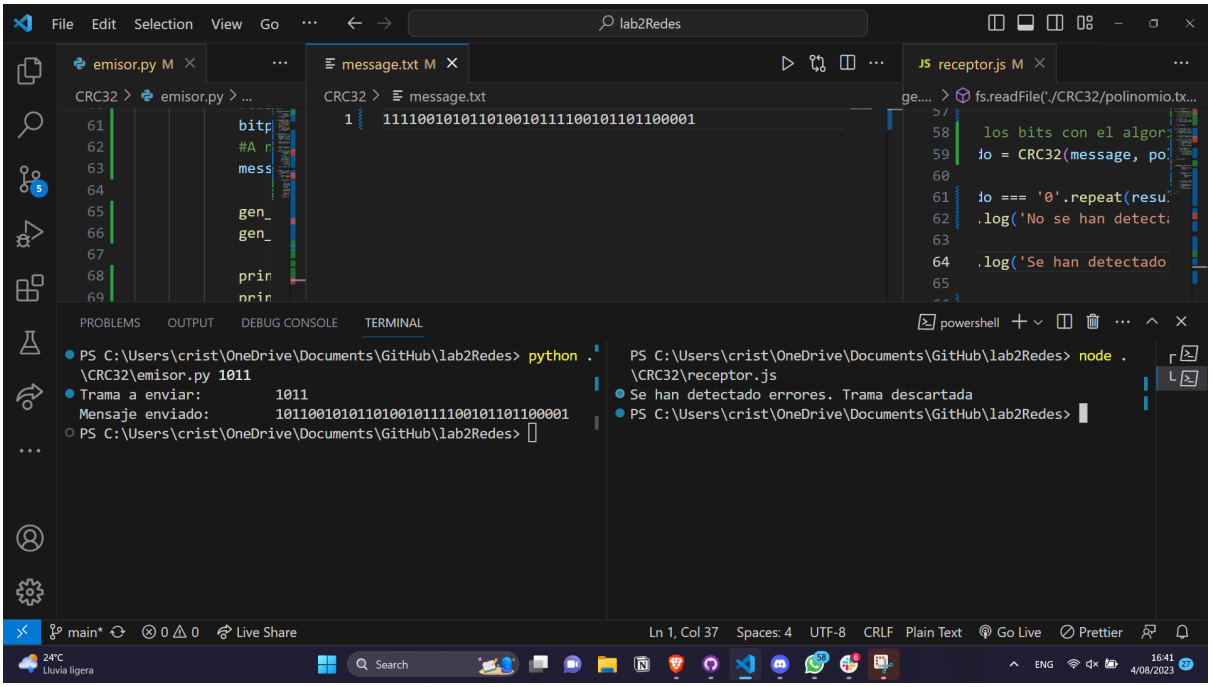


Tramas modificadas para detectar errores.

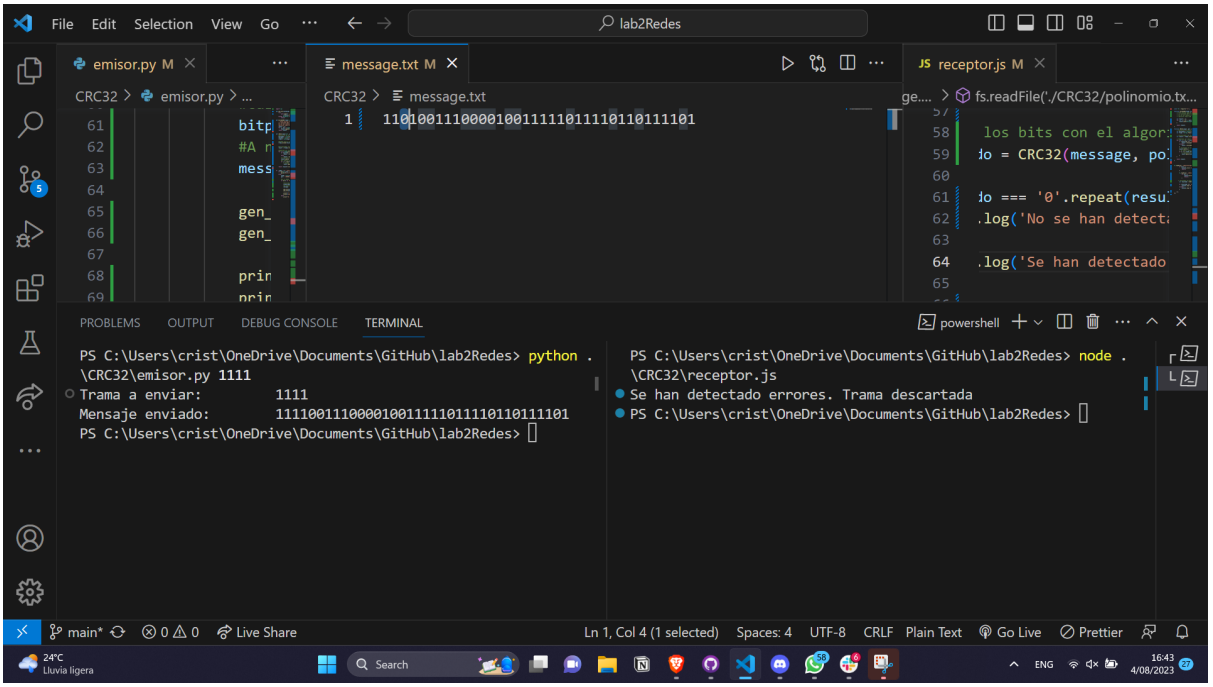
- **1101**
  - **Trama codificada generada:**
    11010011000111001101100001101101 0011
  - **Trama modificada a:**
    10010011000111001101100001101101 0011



- **1011**
  - **Trama codificada generada:**
    10110010101101001011110010110110001 1
  - **Trama modificada a:**
    1 1110010101101001011110010110110001 1

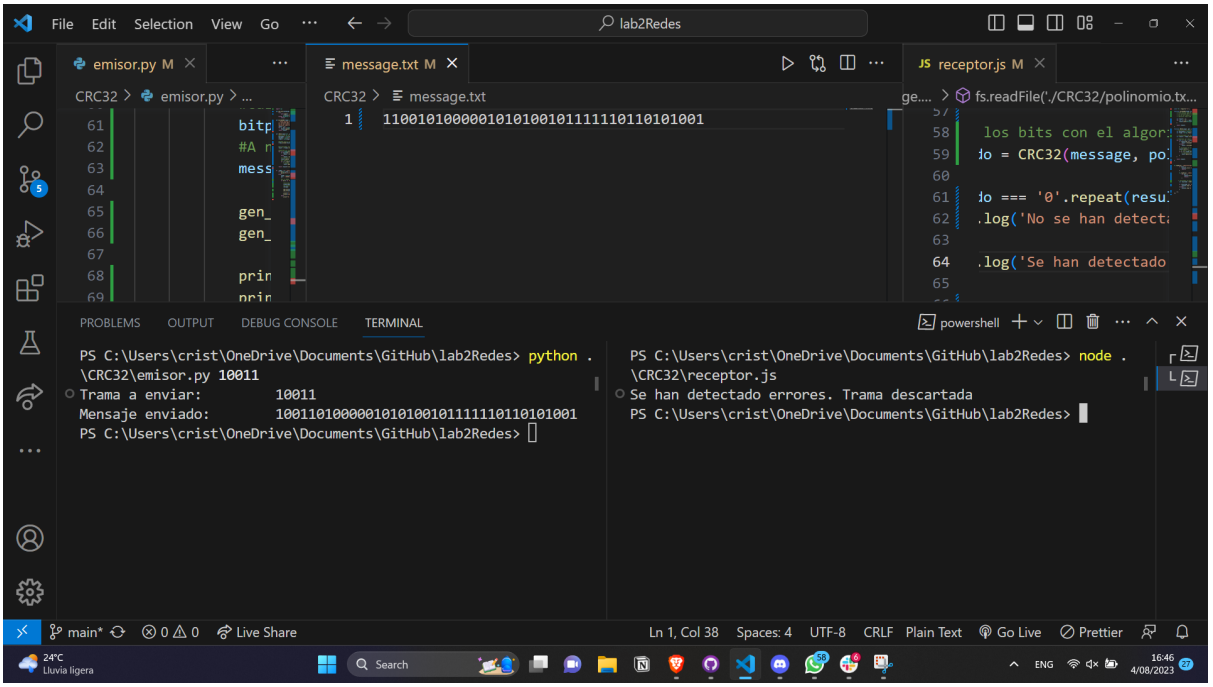


- **1111**
  - **Trama codificada generada:**
    11110011100001001111101111011011 1 1
  - **Trama modificada a:**
    1 1010011100001001111101111011011 1 1

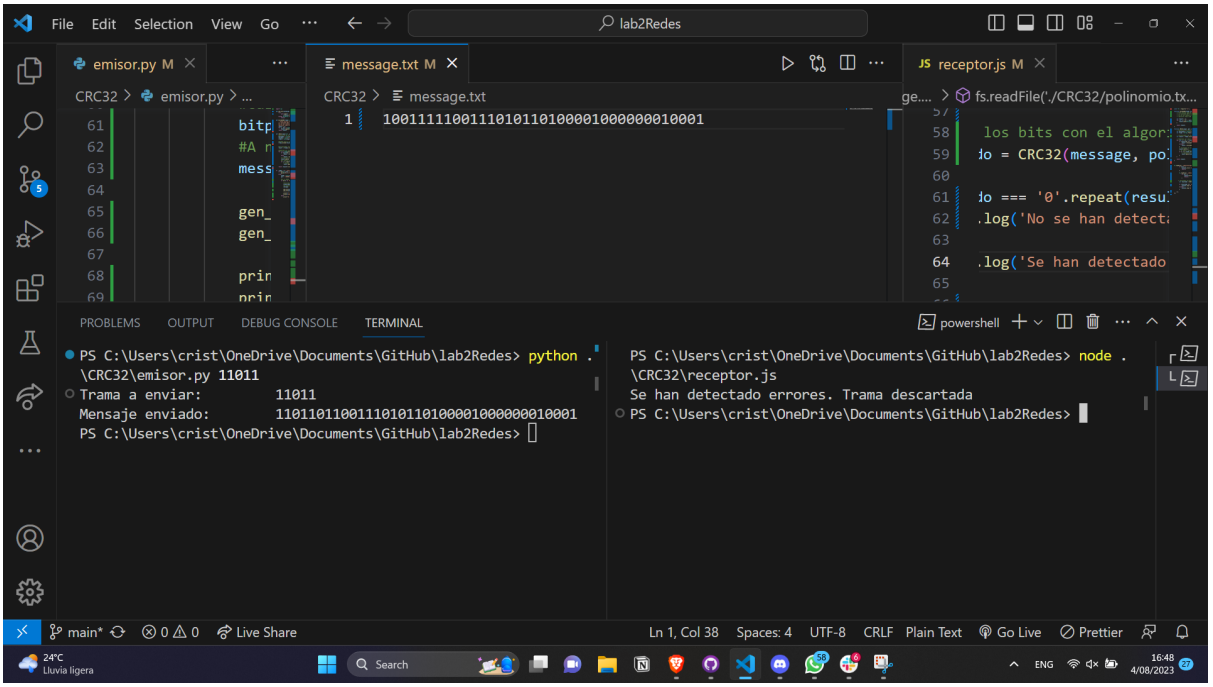


Tramas modificadas con dos bits para detectar errores.

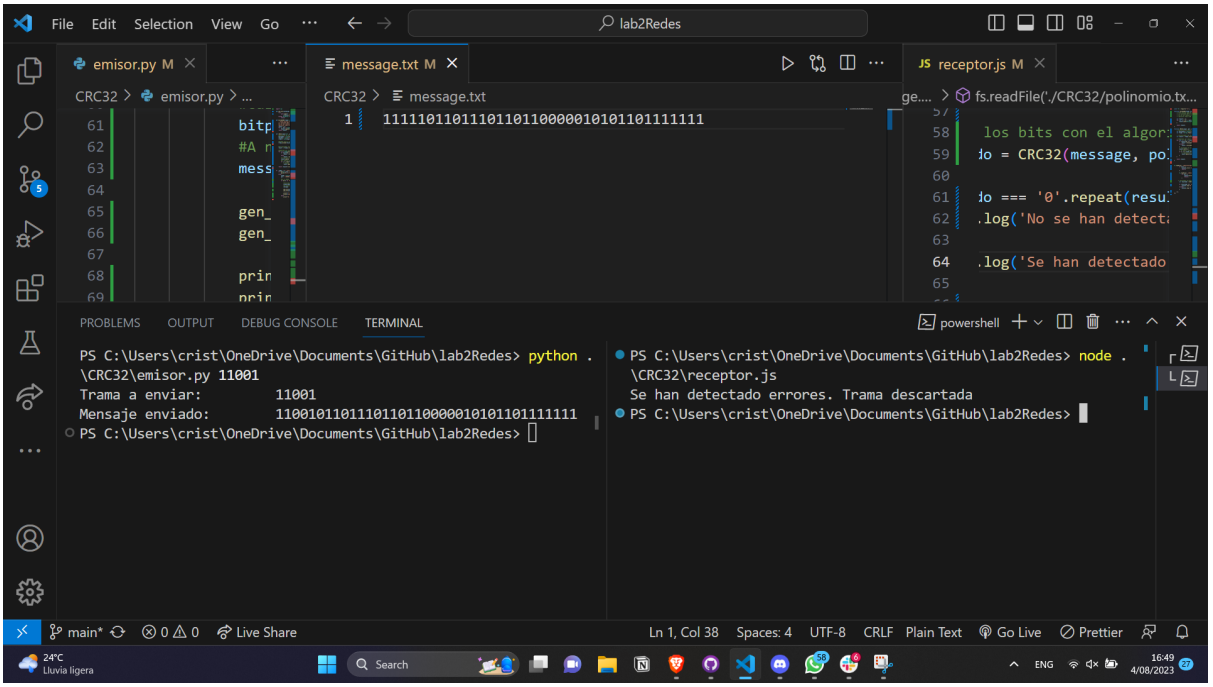
- **10011**
  - **Trama codificada generada:**
    10011010000010101001011111101101010 01
  - **Trama modificada a:**
    1 1 0 0 1 0 1 0 0 0 0 0 0 1 0 1 0 1 0 0 1 0 1 1 1 1 1 0 1 1 0 1 0 1 0 0 1



- **11011**
  - **Trama codificada generada:**
    1101101100111010110100001000000010001
  - **Trama modificada a:**
    1 0 0 1 1 1 1 1 0 0 1 1 1 0 1 0 1 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1



- **11001**
  - **Trama codificada generada:**
    11001011011101101100000101011011111111
  - **Trama modificada a:**
    11111011011101101100000101011011111111

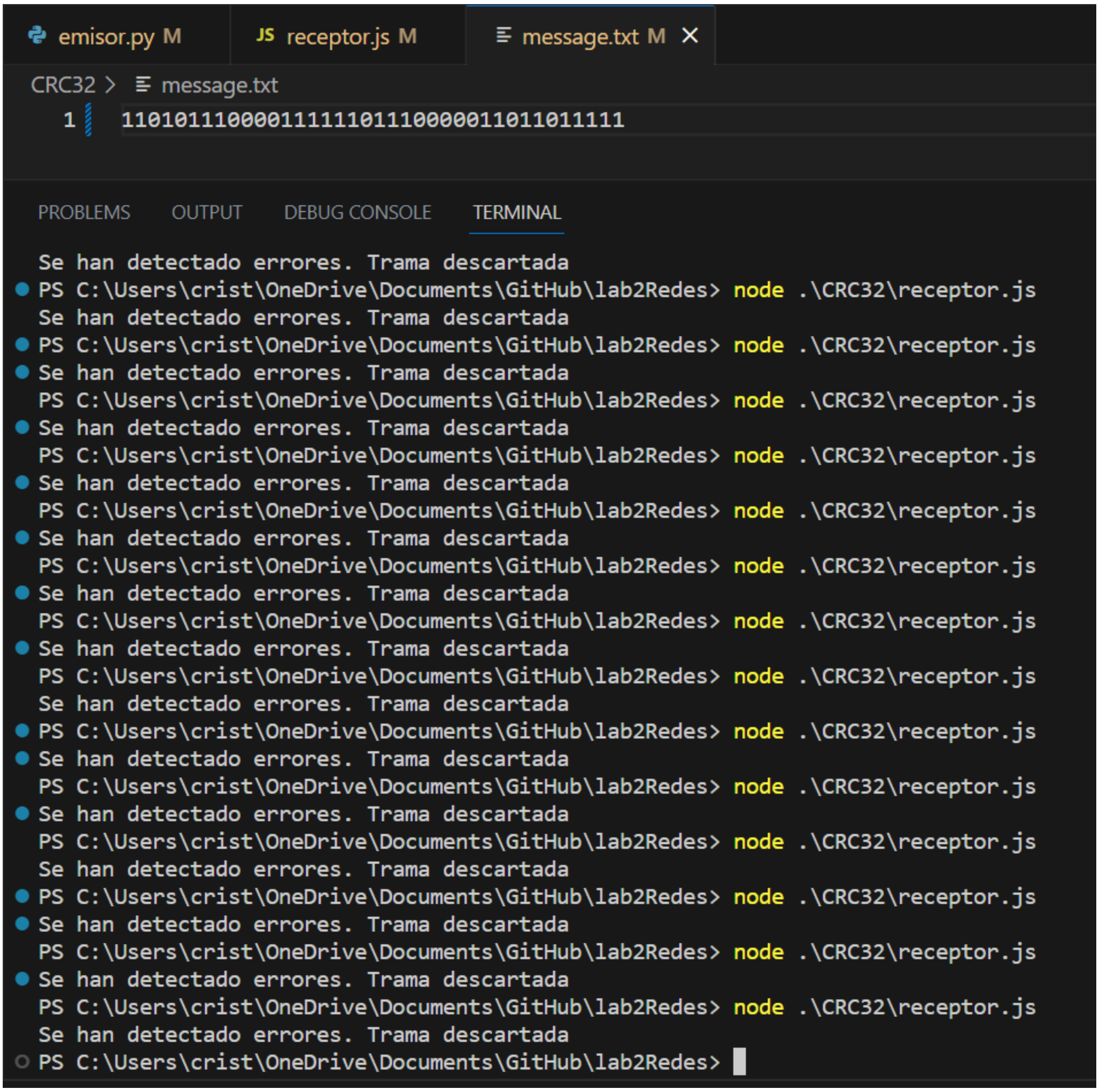


Trama modificada que el algoritmo no detecta como error.

- **No se encontró ninguna modificación en la trama que permitiera burlar al algoritmo.**
- Nota: Se utilizó un algoritmo que generaba todas las posibles cadenas binarias excepto la del mensaje recibido por el emisor, para que de este modo probara el

algoritmo CRC32 e identificara una cadena en la que no se detectara error a pesar de ser diferente al mensaje original. Cabe destacar que no fue posible encontrar una cadena válida además de la cadena conformada únicamente de 0s, lo cual por la teoría sabemos que no es posible válido. Algunas de las tramas utilizadas fueron:

- 11010111000011111101110000011011111010
- 11010111000011111101110000011011111101
- 11010111000011111101110000011011110111
- 11010111000011111101110000011011111000
- 11010111000011111101110000011011011011
- 11010111000011111101110000011011011110
- 11010111000011111101110000011011111111
- 11010111000011111101110000011011110011
- 11010111000011111101110000011011110100
- 11010111000011111101110000011011111001
- 11010111000011111101110000011011111111
- 11010111000011111101110000011011100111
- 11010111000011111101110000011011110011
- 11010111000011111101110000011011100111
- 11010111000011111101110000011011011111



Discusión

En el contexto de los Esquemas de detección y corrección de errores, los algoritmos de Hamming y CRC-32 presentan cierta incertidumbre en su capacidad para detectar y corregir errores. La versión básica de 4,7 bits del algoritmo de Hamming tiene una eficacia de detección de errores del 57%, mientras que una versión más ampliada de 11 bits aumenta teóricamente su capacidad de detección al 90%. En comparación, el algoritmo CRC-32 tiene una probabilidad mucho más alta de detectar errores, aunque no puede corregirlos directamente. La elección del algoritmo dependerá de las necesidades específicas de cada aplicación y la importancia de la integridad de los datos transmitidos. También se debe de tomar en cuenta el tiempo de procesamiento que tomará para estos algoritmos, pues debido al proceso de decodificación y cambio de bits de hamming el mismo algoritmo hace que sea más lento que CRC-32 por lo cual estariamos teniendo en cuenta el segundo es más rápido pero no presenta la capacidad de corrección.

Durante las pruebas con el algoritmo de Hamming, se determinó que tiene una limitación significativa pues este algoritmo es incapaz de cambiar y corregir más de un bit en una agrupación de 7 bits por una cadena de mensaje. Si se presentan dos o más errores, el algoritmo puede detectar la presencia de errores, pero no será capaz de identificar cuáles son los bits erróneos y, por lo tanto, no podrá corregirlos para hacer una cadena correcta. Esto se debe a la lógica de funcionamiento del algoritmo, que depende en gran medida de los bits de paridad y sus posiciones. En algunos casos, al cambiar ciertos bits en la trama, los nuevos valores podrían satisfacer las reglas de paridad y dar como resultado una respuesta "correcta" según el algoritmo, a pesar de que no es el correcto.

Esta incertidumbre en la corrección de errores con el algoritmo de Hamming es una consideración importante al implementarlo en sistemas críticos. En aplicaciones donde se requiere una alta confiabilidad en la corrección de errores, es posible que se necesitan algoritmos más avanzados, como códigos correctores de errores de mayor distancia, para garantizar una recuperación precisa de los datos originales. El algoritmo de Hamming sigue siendo útil en escenarios más simples, pero es esencial comprender sus limitaciones y considerar opciones más robustas cuando la precisión de la corrección es de suma importancia. Por ejemplo, usar el mismo algoritmo de hamming pero en la versión más robusta que aún así permite corregir errores es una buena opción sin embargo es mucho más demandante computacionalmente.

Regresando al algoritmo CRC32, se realizó una implementación relativamente sencilla, la cual pasó con éxito todas las pruebas realizadas. Sin embargo, es importante destacar que en la última prueba no fue posible encontrar una modificación en la cadena binaria que burlara al algoritmo, pues incluso se utilizó un algoritmo que generaba modificaciones en la trama e iterativamente iba procesando la trama generada con el algoritmo CRC32 y no fue posible encontrar una en la que no detectara error. Esto demostró la alta efectividad de dicho procedimiento en la detección de errores.

Comentario

- El algoritmo de Hamming a pesar de tener una explicación un poco más sencilla consta de muchas verificaciones de los bits de paridad lo cual lo hace más robusto pero no infalible. Por otra parte el algoritmo CRC32 a pesar de ser bastante simple, tiene un alto grado de asertividad al momento de detectar errores. Sin embargo, a pesar de que no es tan sencillo engañarlo, no se recomienda su uso como un proceso de seguridad en la integridad de los datos.

Conclusiones

1. Hamming es efectivo para detectar y corregir errores en la transmisión de datos, pero no es infalible pues podemos encontrar que al tener errores que satisfacen las reglas estos no son perceptibles por el algoritmo.
2. Hamming no es confiable cuando hay más de un bit con flip en una agrupación de 4 bits en una trama.
3. Es esencial no solo hacer uso de un algoritmo pues tenemos implementaciones de código más robustas como el crc-32 que llega a ser un poco más difícil de engañar con errores en las tramas.
4. El algoritmo CRC32 es relativamente simple de implementar y requiere recursos computacionales mínimos.
5. El algoritmo CRC32 es altamente efectivo, pues no es posible burlar el algoritmo modificando la trama.

Citas y referencias bibliográficas

Conway, J. H., & Sloane, N. J. A. (1998). Sphere Packings, Lattices and Groups (3rd ed.). New York: Springer-Verlag. ISBN 0-387-98585-9. (requiere registro).

History of Hamming Codes. (n.d.). Archivado desde el original el 25 de octubre de 2007. Consultado el 3 de abril de 2008.