Todos los ejercicios se programan usando la placa de Arduino ESP32. Para poder subir archivos y usarlos en el dispositivo se tiene que utilizar el IDE de VisualStudio con PlatformIO, puedes encontrar los detalles de instalación en https://www.youtube.com/watch?v=L9ZEKyLeSfU&t=6s

1. Diseñe una función que simule un canal ruidoso. La función admitirá un vector binario de longitud L bytes y devolverá un vector de la misma longitud donde la probabilidad de que un bit haya cambiado de valor será f. Ambos vectores binarios se pasarán empaquetados en un tipo binario mayor como por ejemplo el unsigned char
   La signatura de la función **void noisyChannel(unsigned char *in, unsigned char *out, int l, float f);**

2. Diseñar un codificador de repetición que grado Rn donde n es el número de bits que se repiten en cada bloque. A la función se le pasa como parámetros el mensaje a codificar en binario empaquetado en caracteres, la longitud l en bytes del mensaje, n el grado del codificador de repetición y un vector vacío suficientemente grande como para almacenar el resultado.
   La signatura de la función **void reptitionCoder(unsigned char *in, unsigned char *out, int l, int n);**

3. Diseñar un decodificador de repetición de grado Rn donde n es el número de bits que se repiten en cada bloque. A la función se le pasa como parámetros el mensaje a decodificar en binario empaquetado en caracteres, la longitud l en bytes del mensaje, n el grado del decodificador de repetición y un vector vacío suficientemente grande como para almacenar el resultado.
   La signatura de la función **void reptitionDecoder(unsigned char *in, unsigned char *out, int l, int n);**

4. Diseñar un codificador de bloque de código Hamming (7,4). A la función se le pasa como parámetros el mensaje a codificar en binario empaquetado en caracteres, la longitud l en bytes del mensaje y un vector vacío suficientemente grande como para almacenar el resultado. Tenga en cuenta que cada byte de entrada tiene dos paquetes de 4 bit a codificar
   La signatura de la función **void hammingCoder(unsigned char *in, unsigned char *out, int l);**

5. Diseñar un decodificador de bloque de código Hamming (7,4). A la función se le pasa como parámetros el mensaje a decodificar en binario empaquetado en caracteres, la longitud l en bytes del mensaje, n el grado del decodificador de repetición y un vector vacío suficientemente grande como para almacenar el resultado.
   La signatura de la función **void hammingDecoder(unsigned char *in, unsigned char *out, int l, int n);**

6. Diseñe una clase NoisyChannel que tenga los métodos necesarios para enviar un paquete y recibir un paquete a través del canal. El constructor indicará el porcentaje de ruido aleatorio que introducirá la clase al mensaje.

7. Diseñe una clase RepetitionCode que incluya los métodos usados para codificar y decodificar un mensaje utilizando este mecanismo de codificación.

8. Haga lo mismo para HammingCode

9. Analice las ventajas e inconvenientes de cada uno de los dos sistemas de codificación por bloques que hemos analizado hasta ahora. Tome fichero de texto plano no comprimido y hágalo pasar por el sistema codificador – canalRuidoso – decodificador de bloque con una probabilidad de error de 0.05. Utilice una codificación de Hamming y una codificación R4 y compare los resultados obtenidos entre ambos sistemas de codificación. https://es.wikipedia.org/wiki/Windows_bitmap en función de la tasa de transmisión conseguida versus el porcentaje de error de la imagen resultado.

10. Una forma de mejorar los sistemas de codificación es serializar dos códigos diferentes para minimizar la posibilidad de error. Diseñe una clase que aproveche las dos clases previas RepetitionCode y HammingCode para crear un nuevo código llamado HammingRepetition.

11. Escriba un programa que tome un fichero de texto plano de cualquier longitud y devuelva un fichero con el porcentaje relativo (en tanto por uno) de cada una de las letras del alfabeto dentro del fichero, normalizado a mayúsculas. (utilice un fichero de texto de al menos 1 millón de caracteres.