**PROGRAMACION ESTRUCTURADA**

**Trabajo práctico final 2021**

Se dispone de un archivo binario con información que se grabó a partir un canal de comunicación donde un emisor envía información a múltiples destinatarios en forma totalmente aleatoria. Se pide realizar un programa en lenguaje C que permita leer y analizar la información contenida en el archivo proporcionado por la cátedra de modo de recuperar la imagen bmp que se necesita enviar al usuario específico. La información contenida en el mismo está compuesta por un número desconocido de tramas (paquetes), cada una de longitud variable. Cada paquete se encuentra organizado de acuerdo al siguiente formato:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **H1** | **ID** | **DEST** | **SIZE** | **DATA** | **CRC** | **H2** |
| 1 byte | 1 byte | 1 byte | 1 byte | ···· N bytes ···· | 2 bytes | 1 byte |

**H1**: es un valor constante = 0x3C (‘<’) que indica el inicio de un paquete de información. Se debe tener en cuenta que este valor puede corresponder también a un dato válido por lo que se debe analizar detalladamente según se explica más adelante.

**ID**: es el identificador del orden del paquete. Dado que se trata de un canal de comunicación asincrónico, el orden en que llegan los paquetes NO necesariamente es el mismo en el que fueron generados. ID corresponde al índice en que se generaron.

**DEST**: es el identificador del destinatario de la información. Puede tomar los valores válidos del 1 al 8, ambos inclusive. A cada alumno se le asignará un único número que lo identifica y que será comunicado en el momento de enviarle el archivo binario correspondiente.

**SIZE**: es el tamaño (en bytes) del campo **DATA.**

**DATA**: corresponde a una secuencia de **SIZE** bytes que contiene la porción de información original incluida en esa trama.

**CRC**: corresponden al cálculo de un CRC de 16 bits. El CRC es un código de detección de errores usado frecuentemente en [redes](https://es.wikipedia.org/wiki/Internet) digitales y en [dispositivos de almacenamiento](https://es.wikipedia.org/wiki/Dispositivos_de_almacenamiento) para detectar cambios accidentales en los [datos](https://es.wikipedia.org/wiki/Datos). La función que implementa este cálculo es suministrada por la cátedra mediante 2 archivos: uno de cabecera, “crc\_16.h”, y el correspondiente código fuente “crc\_16.c”. Para chequear la integridad del paquete se debe llamar a la función “calculate\_checksum”, cuya firma se muestra a continuación:

***uint16\_t calculate\_checksum(uint8\_t[], uint16\_t);***

Si el valor de retorno de esta función es CERO significa que el paquete llegó sin modificaciones ni errores. Un valor distinto de CERO indica que se produjo un error y el paquete debe descartarse.

**H2**: es un valor constante = 0x3E (‘>’) que indica el fin de un paquete de información. Se debe tener en cuenta que este valor puede corresponder también a un dato válido por lo que se debe analizar detalladamente según se explica más adelante.

Comentarios:

1. El programa deberá contemplar la recepción de 3 parámetros de entrada que corresponden al identificador del destino (0x01 a 0x08), el nombre del archivo de entrada (\*.bin) y el nombre del archivo de salida (\*.bmp).
2. Una vez identificado el inicio de una trama se pueden presentar diferentes casos:
3. Que el campo DEST no corresponda con el asignado al destinatario, en cuyo caso se debe continuar la lectura hasta obtener el carácter H2 (final) sin almacenar ningún dato (se descarta la trama por completo).
4. Que el campo DEST corresponda con el destinatario asignado, en cuyo caso debe efectuarse la lectura de SIZE bytes y de 2 bytes adicionales correspondientes al CRC. Calcular el CRC con la función suministrada. Si la función retorna 0 indica que la trama es correcta por lo que debe almacenarse.
5. Si el resultado de la función que calcula el CRC es distinto de 0 la trama completa debe descartarse (aunque el DEST sea el adecuado).
6. Continuar la lectura de todas las tramas hasta que se alcance el final del archivo binario.
7. Al originarse la información se previó el hecho que los caracteres H1 y H2 pudieran representar también información válida. Al presentarse este caso se almacena ese carácter y se lo duplica para indicar que no es inicio ni fin de trama. Así cuando en el cuerpo de la misma un dato tiene el valor 0x3C/0x3E se lo almacena como <</>>.
8. Ordenar las tramas almacenadas de acuerdo con su campo ID.
9. Buscar en el cuerpo de cada paquete válido la aparición de los caracteres dobles (<</>>) y eliminar uno de ellos.
10. Guardar la información procesada en otro archivo binario con extensión bmp que al ser abierto con un programa de gráficos muestre la imagen correspondiente.
11. Se debe realizar el diagrama de flujo del programa completo y ser presentado en forma entendible y prolija.
12. Se evaluará la correcta modularización del programa, donde cada función debe realizar una tarea específica contando con los parámetros necesarios pasados por valor y/o referencia.
13. Recordar que la longitud de cada paquete es variable, por lo que se debe prestar ESPECIAL atención al dimensionamiento dinámico de la estructura donde se almacenará la información. Toda reserva de memoria que se realice debe ser liberada al finalizar. Toda apertura de archivo que se realice debe ser comprobada que no produjo error y debe cerrarse oportunamente.
14. El programa será probado desde la línea de comandos. Deben validarse TODOS los parámetros de entrada al main() de acuerdo a lo especificado en 1).