**Introduction to Computer Communications – PA1**

**Yaniv Blum 312253586**

**Gai Greenberg 205431224**

**How to run:**

1. Run channel first in order to receive from it the ports for sender and receiver.

2. Give the ports to the sender and receiver accordingly.

3. Insert files name for both sender and receiver.

4. Outputs in terminal will be shown as required in the assignment.

**Code Structure**:

We devided the code into 4 projects under the same solution:

1. **Channel**:

Implemented as described in the assignment.

The channel creates a connection with the sender and the receiver, than getting a message from the sender. On that massage the channel adds noise according to the method given in the command line arg, and send the new noised massage to the receiver.

Channel project has:

* Channel.c
* ChannelHelper.c
* ChannelHelper.h

2. **Common**:

Common is a general name for the project that stores the utilities and sstructs used by the program. Mostly holding the socket function for establishing a connection between the different parts of the program. Also hold functions that were used by few projects.

Common project has:

* Utils.c
* Utils.h
* Definitions.h

3. **Sender**:

The sender is implemented as described in the assignment.

It reads a given file, codes it is using hamming code and sends it to the channel.

To help the receiver to allocate buffers in the right size, the sender first sends the file's size.

Sender project has:

* Sender.c
* SenderHelper.c
* SenderHelper.h

4. **receiver**:

The receiver is implemented as described in the assignment.

It gets a massage from the channel and decodes it using hamming decoder to restore the original message if hamming can restore in the given situation, again compatible to the instructions . At the end it writes to the requested file .

Receiver project has :

* Receiver.c
* ReceiverHelper.c
* ReceiverHelper.h

In our implementation we used several build in C function such as :

* Socket related function like :
  + WAS functions
  + Connect, bind, accept, etc..
  + Send, recv, etc…
* atoi, itoa, etc…

Libraries we used:

* math
* stdio
* stdlib
* string
* stdbool
* sys
* winsock2
* conio
* errno

All hamming code function (decode, encode) were written by us.

Bonus:

The program can handle any file size as long it can be divided to 26-bytes as required. The whole file is sent as a complete message once on the connection and the processing on the message is running internally.

Limitations & bugs:

* The program only works on files with size that can be divided to 26-byte .
* There is no handling for illegal arguments.
* Sometimes the connection establishment fails due to WSA problems, a WSA assertion will be printed on screen, trying to run again (once or twice) will result in a correct run.