**class Party:**

This class represents a participant in the protocol. All the communication between the parties is performed via this class. To execute the protocol, the following functions should be called: connectToAllParties, fInput, calcCircuit and reconstruct in the specified order.

After the main thread performs the connectToAllParties function the vector \_sockets contain all the sockets to the other parties through which the future messages of the protocol will be sent.

Moreover, the class contains a vector of shares, one share per input to the circuit. After the execution of fInput this vector holds the shares this party suppose to have as discussed in the paper.

Additionally, the party has a pointer to the arithmetic circuit which needs to be calculated.

Finally, the party holds a vector of Messages of unread messages this party received. A message marked as read if the main thread got the message.

**Class Share:**

This class represent a share as defined in the paper. The share object is implemented using an attribute of type pair. The first part and the second are pointers to Parts of the share.

**Class PartyShare:**

This class inherits from Share class. The purpose of this class is to allow communication during the calculation of the multiplication gate. In addition to the original share information PartyShare also contains a pointer to the party which possess the share, to perform communication.

**Class Part:**

This class represent the first and second parts in each share. It has value, index, and a single character to indicate what expression this parts construct together.

The addition of all three parts would give us the value of what they represent.

**Class circuit:**

This class represent the arithmetic circuit all 3 parties calculating together. The circuit contains layers of aggregation toward the output of the circuits. Each layer contains a pointer to an additionGate or a MultipicationGate which inherits from the abstract class Gate.

The circuit class also contains a pointer to the party. The reason for that is to forward this pointer to the PartyShare class at its construction. The circuit class creates PartyShare if the circuit contains a multiplication gate.

**Class Gate:**

An abstract class which represent a generic gate. It has a lest input and an output of type share.

**Class MultiplicationGate<operand>:**

A template class inherits from the abstract class Gate. This class represent a multiplication gate. It has an attribute that represent the right input to the gate. The type of this attribute can be a constant of type long or a share type.

**Class AdditionGate<operand>:**

A template class inherits from the abstract class Gate. This class represent an addition gate. It has an attribute that represent the right input to the gate. The type of this attribute can be a constant of type long or a share type.

**Class TcpSocket:**

This class represent a generic TCP socket. It has a static member WSAInitializer to allow communication via sockets. This class also contains a thread which waits for new messages to come from this socket.

Each party functions as both a server and a client. It is a server for Id-1 and c client for id+1 to keep it symmetric.

**Class TcpClient:**

This class represent a TCP client which inherits from TcpSocket class. This class initiate a TCP connection with the party with id+1.

**Class TcpServer:**

This class represent a TCP server which inherits from TcpSocket class. This class waits the party with id-1 to initiate a TCP connection with it.

**Class WSAInitializer:**

Initialize some dll files so the windows operation system will allow the program to open and use the sockets. This object should be constructed only once.