CS453 Socket programming – UDP

Jaeseong Lee(29990425)

**How to run:**

1. Execute UDP_Server.py first to run the server.
2. Execute UDP_Client.py to run the client.

**Design:**

Both the server and client side were written in Python.

**Client Side:**

Server address and port number are defined. I used 127.0.0.1 which is local host and the 50005 for the port number which are same as the server side. The ‘d’ value for timeout test set as 0.1 at the beginning. The ‘d’ value will be doubled every request inside of a while loop and if the d value is greater than 2, throw a timeout error exception and notify the server is DEAD. In side of the while loop, a client side UDP socket is created and asking client users to enter the operator in between two values. The user can enter any string with/without space in between the digits and operator. However, invalid input values such as omitting operator or no values before/after the operator will receive a fail code which is 300. Client user will receive two values returned. Since the first return value is status code which is an integer, the value will be coded and converted into integer. The next value receiving is the result calculated. The client side will wait for ‘d’ seconds then increase the d value as double. If the state code is 200, decode the return value and display the result. Otherwise, the console will notify that the users entered invalid input values. Finally, since the client user received the status code with a result, the program will ask the client if they need to try another computation. If client enter ‘y’ or ‘Y’, it will stay in the loop otherwise it will jump out of the loop and close the socket.

**Server Side:**

Server address and port number are set as 127.0.0.1 which is local host and the 50005 for the port number which is same as the client side. Socket is created, and local host and port number are bind to ready to receive data from client side. There is a while loop which is always true so that the server socket is opened and always ready to receive the data. Inside the loop, it receives the input string and client address and display on a console. It will try to detect weather the operator (this program supports ‘+’,’-’,’*’,’/’ only) is exist in the input string or not. If there is no operator or more than one operator, it will return ‘-1’ and the fail code (300). If the input data is valid to operate, the data will be split into 2 values and be stored into the array for computation, so the length of the array must be 2. The result will be generated according to the operator and sent out to client side. Finally, the variables for storing the operator, array, first and second value are cleaned up the next request.

**Improvement**:

This program can be modified and improved to calculate with multiple operators in the input or other type of operators such as square root/modulo/exponent.