BTN415 Lab 6 – OO Sockets

In this lab, you will create classes to serve as blueprint to UDP/IP server and client objects.

### LEARNING OUTCOMES

Upon successful completion of this lab, you will have demonstrated the ability to:

* Create inherited classes
* Create constructors and destructors

For this lab, you should update two files, one called *socket.h* and other called *socket.cpp*. Then, you can test your results by running them with the main source code file which is provided in: <https://github.com/marceljar/BTN415_Labs/blob/main/lab6/main.cpp>. Note that this provided source code is already ready to test a UDP server against our basic UDP client that served as a starting point for lab 4 (<https://github.com/marceljar/BTN415_Labs/blob/main/lab6/udp_client.cpp>). You can test your UDP client, uncommenting and commenting the proper lines, and setting it against the UDP server in (<https://github.com/marceljar/BTN415_Labs/blob/main/lab6/udp_server.cpp>).

## PART A – [2.5 marks]

## Create the declaration of a Server\_UDP class, derived from Node, with the following properties and methods:

### Methods (all public):

* A base constructor
* A constructor that takes two arguments: a string (for the *ip*) and an int (for the *port*)
* A method called *bind\_socket* that takes no arguments and returns a Boolean. This method should bind the server’s socket (active) to the address provided in the constructor (this->ip and this->port). It should return true if the binding was successful.
* A method called *send\_message* that takes a string argument (for a message), another string with an IP address, an integer with a port number, and returns an integer. The message should be sent to the client with a matching IP address and port number.
* A method called *receive\_message* that takes a reference for a string argument (for a message), saves the received message in this referenced string, and returns an address struct (*struct sockaddr\_in*) containing info from whoever sent the message.

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

## PART B – [2.5 marks]

## Create the declaration of a Client\_UDP class, derived from Node, with the following methods:

### Methods (all public):

* A base constructor
* A method called *send\_message* that takes a string argument (for a message), another string with an IP address, an integer with a port number, and returns an integer. The message should be sent to the client with a matching IP address and port number.
* A method called *receive\_message* that takes a reference for a string argument (for a message), saves the received message in this referenced string, and returns an address struct (*struct sockaddr\_in*) containing info from whoever sent the message.

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

**Server + Client**

**Graphical user interface

Description automatically generated**

### SUBMISSION INSTRUCTIONS

*You only need to submit two source codes, a declarations socket.h file, and an implementation socket.cpp file.*