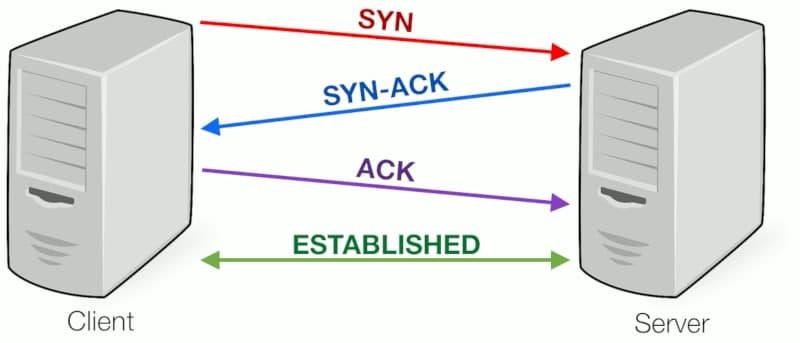
 

**Al-Imam Muhammad Bin Saud Islamic University College of Computer and Information Sciences, Department of Computer Science**

CS 330 Computer network TCP-based client server application

December 11, 2021



|  |  |  |
| --- | --- | --- |
| **Name** | **Email** | **ID** |
|  |  |  |
|  |  |  |
|  |  |  |
| **Hanan Owaidh Almutairi** |  |  |

**Index:**

1. [The goal 3](#_TOC_250013)
   1. [Setting up the Programming Environment 3](#_TOC_250012)
   2. [Steps for setting up the network 3](#_TOC_250011)
2. [Steps for socket programming 3](#_TOC_250010)
   1. [Steps to create a client using TCP/IP API 4](#_TOC_250009)
   2. [Steps to create a server using TCP/IP API 4](#_TOC_250008)
3. [project code and comments 6](#_TOC_250007)
   1. [Base64 encoding with an example 6](#_TOC_250006)
   2. Server source code 8
      1. [Images for the Server source code 11](#_TOC_250005)
   3. Client source code 12
      1. [Images for the Client source code 15](#_TOC_250004)
4. [Snapshots of the application outputs 16](#_TOC_250003)
5. [Demo of our program 19](#_TOC_250002)
6. [Problems 19](#_TOC_250001)
7. [References 20](#_TOC_250000)

# The goal

The aim of this project is to develop a simple TCP-based client server application. TCP client/server will communicate over the network and exchange data.

#### Setting up the Programming Environment

* + - We used java programming language because most of the students know the syntax of this language.
    - We did not use any environment. We used the text editor on our computers. We used the command line to compile and run the code.
    - We found all libraries that we need in the java application interface therefore we did not need to install any library.

#### Steps for setting up the network

We used wired technology by using:

* + - Ethernet wire



* + - Ethernet Adapter



* + - Plug the Caple into the adapter and then plug the adapter into the usb port in our computers.

# Steps for socket programming

We use java programming language socket programming to implement the client- server communication over TCP protocol [ <https://www.youtube.com/watch?v=xKgxqG411c&t=451s>].

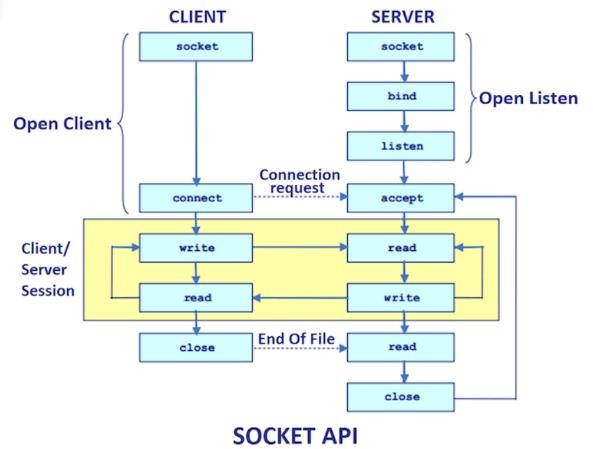
#### Steps to create a client using TCP/IP API

* + - Create a socket using the socket ().
    - Initialize the socket address structure as per the server and connect the socket to the address of the server using the connect ().
    - Read and write data by using the getInputStream() and getOutputStream() functions.
    - Close the connection by calling the close () function.

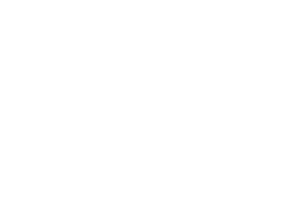
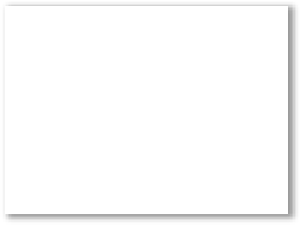
#### Steps to create a server using TCP/IP API

* + - Create a socket using the socket ().
    - Initialize the socket address structure and bind the socket to an address using the bind () function.
    - Listen for connections with the listen () function.
    - Accept a connection with the accept () function system call. This call typically blocks until a client connects to the server.
    - Read and write data by using the getInputStream () and getOutputStream() functions.
    - Close the connection by using the close () function.

This picture illustrates in general the mechanism of TCP socket programming.



# project code and comments



Note: Dear teacher, in section 3.1 we explain how the library base64 encrypts the word with a simple example to illustrate that. You can skip this section if would like to.

## Base64 encoding with an example

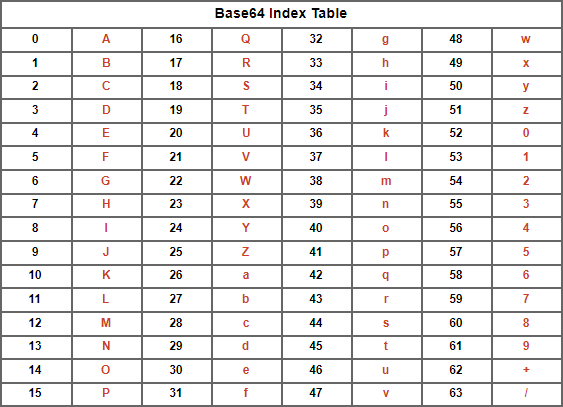
The Base64 encoding is used to convert bytes that have binary or text data into ASCII characters. Encoding prevents the data from getting corrupted when it is transferred or processed through a text-only system. In this article, we will discuss about Base64 encoding and decoding and its uses to encode and decode binary and text data.

Each Base64 character represents 6 bits of data. it is also important to note that it is not meant for encryption for obvious reasons.

To convert a string into a Base64 character the following steps should be followed:

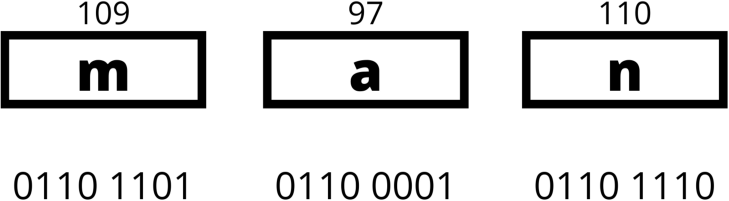
* + - Get the ASCII value of each character in the string.
    - Compute the 8-bit binary equivalent of the ASCII values
    - Convert the 8-bit characters chunk into chunks of 6 bits by re-grouping the digits
    - Convert the 6-bit binary groups to their respective decimal values.
    - Use the Base64 encoding table to align the respective Base64 values for each decimal value.

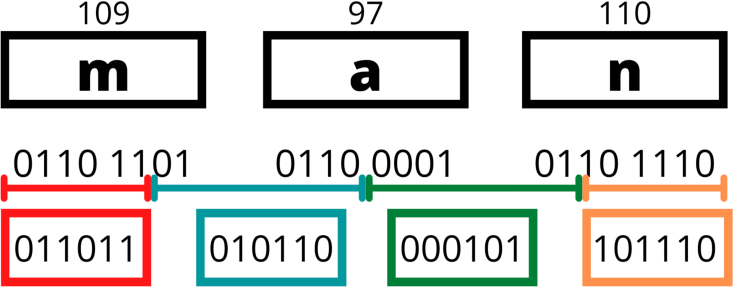
The below image provides us with a Base64 encoding table.



## Example:

We want to encode the word (man) based on base64.





011011 => 27 => b

010110 => 22 => W

000101 => 5 => F

101110 => 46 => u

#### man  bWFu

The decoding process is the opposite of this process.

## Server source code and comments

import java.net.\*;

import java.io.\*;

* + - **java.net:** Provides classes for networking applications.
    - **java.io:** provides for system input and output through data streams, serialization, and the file system.

public class Server { // open class Server.

public static void main(String[] args) throws IOException { // open main method.

ServerSocket ss = new ServerSocket(4999);

We created an object of class ServerSocket to receive any communication from any client.

Socket s = ss.accept();

We called accept method which returns an object of type Socket that can exchange data with any object of type Socket.

System.out.println("client connected");

print this statement after accept client request.

InputStreamReader in = new InputStreamReader(s.getInputStream());

read bytes that are written by the client.

System.out.println("client : " + Help.read\_word(in));

print client word.

Help.write\_word(s, "yes");

send a confirmation to the client that the server is running fine.

String choice= Help.read\_word(in);

read the mode selected by the client.

String message= Help.read\_word(in);

read the word written by the client.

int x = Integer.parseInt(choice);

while (choice.equals("1") || choice.equals("2")) { // open while loop.

System.out.println("client : " + message);

if the mode number is one (open mode) or two (secure mode), then we will print client word.

send back a copy of the word that was received from the client.

if (choice.equals("1")) { Help.send\_massege(message, s, choice, "Server"); }

else if (choice.equals("2")) {

1. message = Help.decode(message);
2. System.out.println("After decoding client message : " + message);
3. Help.send\_massege(message, s, choice, "Server");
4. Call decode method to decode the word and then save it in

the message object.

1. print decoded word.
2. send back a copy of the decoded word that was received from the client. The decoded prosses will be in send\_massege method.

4-

choice = Help.read\_word(in);

read the mode selected by the client.

if (choice.equals("3")) { s.close(); }

if the user wants to quit application the server will close the connection.

message = Help.read\_word(in);

read the word written by the client.

int x = Integer.parseInt(str1);

} // close while loop.

} // close main method.

} // close Server class.

### **3.2.1** Images for the Server source code



## Client source code and comments

import java.net.\*; import java.io.\*;

import java.util.Scanner;

* + - **java.net**: provides classes for networking applications.
    - **java.io:** provides for system input and output through data streams, serialization, and the file system.
    - **java.util.Scanner:** Contains the collections framework, legacy collection classes, event model, date and time facilities, internationalization, and miscellaneous utility classes (a string tokenizer, a random-number generator, and a bit array).

public class Client {// open Client class.

static Scanner input = new Scanner (System.in);

create a Scanner object to read from the user and make it as global attribute to use it in whole class.

public static void print\_choices() { // open print\_choices method.

System.out.println("choose your favorate option:\n 1) open mode\n 2) secure mode\n 3) quit app\n");

} // close print\_choices method.

this method will ask the user to select one option from a list of options.

public static void main(String[] args) throws IOException { // open main methode.

try { // open try.

Socket s = new Socket("localhost", 4999);

here we have created an object of class Socket to communicate with another program on a server that uses the port 4999.

Help.write\_word(s, "is it working");

send message to the server.

InputStreamReader in = new InputStreamReader(s.getInputStream());

read bytes that are written by the server.

System.out.println("server : " + Help.read\_word(in));

print server word.

boolean start = true;

while (start) { // open while loop. print\_choices();

enter the while loop if the value of start variable was true .After that we call print\_choices method to print the choices.

String choice = input.next();

read input from the user.

if (!choice.equals("1") && !choice.equals("2") &&

!choice.equals("3"))

{ System.out.println("WRONG CHOICE PLZ TRY AGAIN!"); continue; }

If the user's choice is not in the list of choices.

if (choice.equals("1") || choice.equals("2")) {

if the client choice was one (open mode) or two (secure mode).

System.out.println("What is your message?");

ask the user to write the message.

String message = input.next();

read the message written by the user and store it in message object.

Help.send\_massege(message, s, choice, "Client");

send a message that was written by the user to the server.

String Server\_message = Help.read\_word(in);

read the word written by the server.

System.out.println("Server : " + Server\_message);

print server word.

if (choice.equals("2")) {

1. Server\_message = Help.decode(Server\_message);
2. System.out.println("After decoding server message : " + Server\_message); }
3. Call decode method to decode the word and then save it in the Server\_message object.
4. print decoded word.

else if (choice.equals("3")) {

1. Help.write\_word(s, "3");
2. s.close();
3. start = false; }
   1. inform the server that the user chooses quit application.
   2. close the connection.
   3. change the value of start variable to false and stop while loop.

} // close while loop.

} // close try.

catch (IOException use) { // open catch.

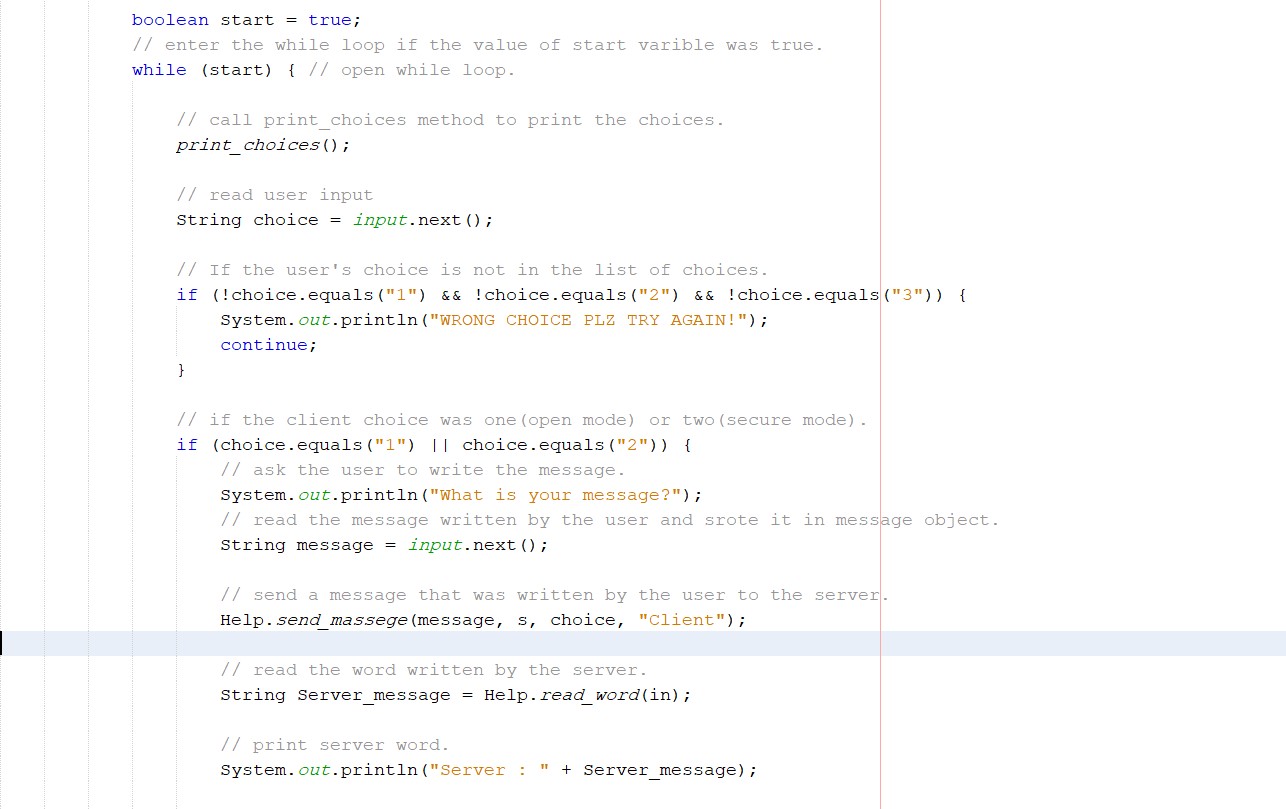
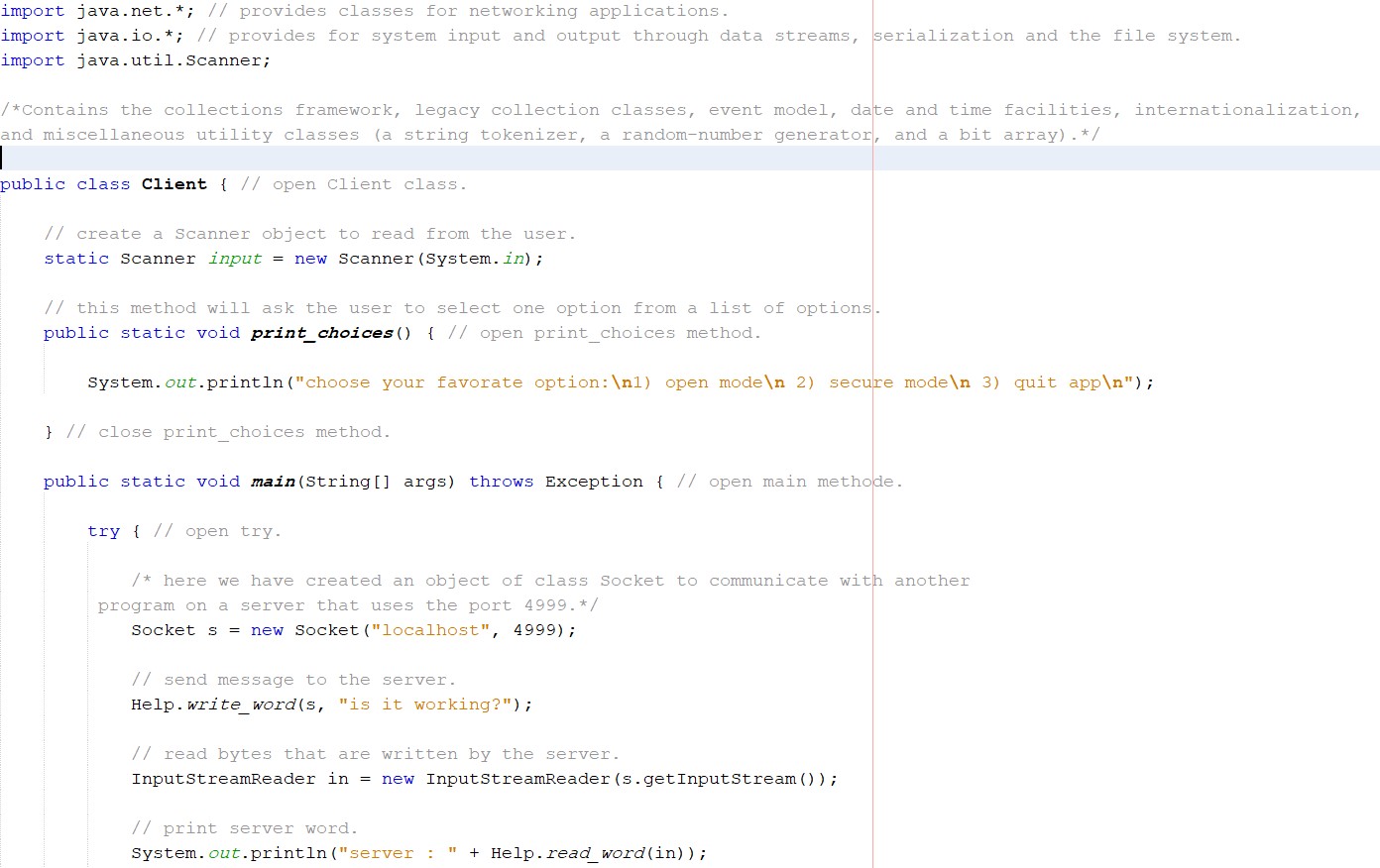
System.out.print("Exception:" + use.getMessage());

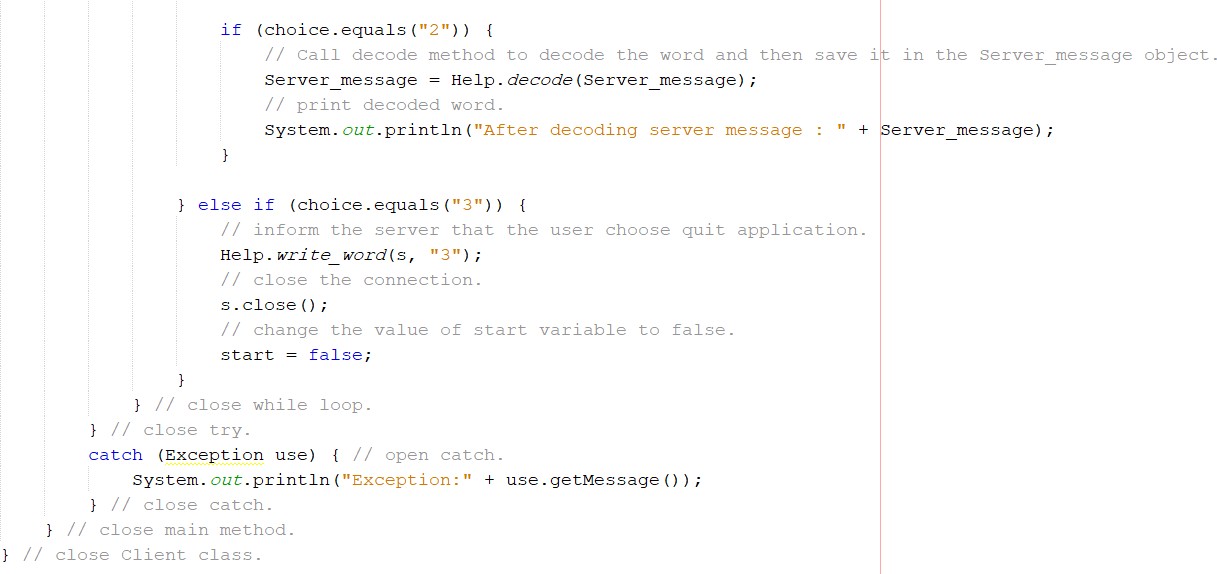
} // close catch.

} // close main method.

} // close Client class.

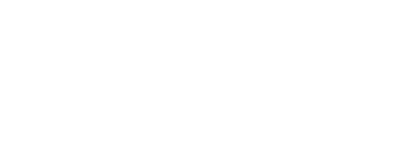
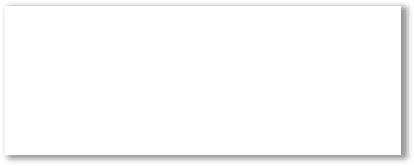
### Images for the Client source code





We have additional class called Help contains:

* + - * read\_word method.



Note: Dear teacher, we have attached the source code files with the delivery. If you would like to try the program.

* + - * write\_word method.
      * encode method.
      * decode method.

# Snapshots of the application outputs

#### Case 1: Start connection.

Server side:

صورة تحتوي على نص, إلكترونيات, لقطة شاشة, عرض  تم إنشاء الوصف تلقائياً

Client side:

صورة تحتوي على نص  تم إنشاء الوصف تلقائياً

#### Case 2: Open mode.

Client side:



صورة تحتوي على نص, إلكترونيات, لقطة شاشة  تم إنشاء الوصف تلقائياًServer side:

Client side:

صورة تحتوي على نص  تم إنشاء الوصف تلقائياً

#### Case 3: Secure mode.

Client side:



Server side:

صورة تحتوي على نص, إلكترونيات, لقطة شاشة, عرض  تم إنشاء الوصف تلقائياً

Client side:

صورة تحتوي على نص, إلكترونيات, لقطة شاشة, عرض  تم إنشاء الوصف تلقائياً

#### Case 4: quit application.

Client side:



#### Case 5: wrong chice.

Clint side:



#### Case 6: server is down.

Clint side:

Client side:

# Demo of our program

[https://drive.google.com/file/d/12fXQbkg4vpRw111TQoDmMHu-](https://drive.google.com/file/d/12fXQbkg4vpRw111TQoDmMHu-yhW7iig8/view?usp=sharing) [yhW7iig8/view?usp=sharing](https://drive.google.com/file/d/12fXQbkg4vpRw111TQoDmMHu-yhW7iig8/view?usp=sharing)

# Problems

**Problem 1:** Knowing whether the message arrived encrypted or not.

**Solution:** We printed it on the screen before decoding.

**Problem 2:** Problem with old JDK at compile time.

**Solution:** Install the JDK latest version to suit some functions in java.

**Problem 3:** Choosing the appropriate library to encrypt messages.

**Solution:** search on the internet.

**Problem 4:** Difficulty understanding the encryption and decryption mechanism of the library when sending messages.

**Solution:** see some tutorials on the internet.

**Problem 5:** Try to explain the encryption method simply in the report. Solution: insert some images in the report to help us to describe the encryption mechanism.

**Problem 6:** How can make two devices exchange messages? wired or wireless in both cases we encountered several problems, including the IP address.

**Solution:** We ask some other students and we found the solution.

# References

* Java Convert String to int**.** [Online]. Available: <https://www.javatpoint.com/java-string-to-int>
* Class PrintWriter**.** [Online]. Available: <https://docs.oracle.com/javase/7/docs/api/java/io/PrintWriter.html>
* Encoding and Decoding with Java**,** May 13,2020. [Online]. Available: <https://www.devx.com/tips/encoding-and-decoding-with-java.html>
* Java socket programming – Simple client server program, Nov 2,2018. [online]. Available:

<https://www.youtube.com/watch?v=xKgxqG411c&t=451s>

* Java base64 encoding and decoding, Nov 12,2021**.** [online]. Available: <https://www.baeldung.com/java-base64-encode-and-decode>

-Base64 Encoding Algorithm. [online]. Available: <https://medium.com/swlh/base64-encoding-algorithm-42abb929087d>

**-** Class Base64**.** [online]. Available: <https://docs.oracle.com/javase/8/docs/api/java/util/Base64.html>

**-**Java Socket Programming. [online]. Available: <https://www.javatpoint.com/socket-programming>