Homework -1

Create a simple Node script that converts 'www.mim.edu' domain name to the equivalent IP address. (Search and learn 'dns' module, resolve4)

Answer

```
const dns = require("dns");
const { promisify } = require("util");

const url = "www.mum.edu";
const promise = promisify(dns.resolve4);

async function hostToIpAddress() {
  await promise(url).then(ip =>
    console.log(`The IP address of ${url} is ${ip}`)
  );
}

hostToIpAddress();
```

Homework -2

2.Create a web server that's going to send a response of big image (bigger then 3MB) to any client that sends a request to your specified server:port. Use the best way for performance. (Try to solve this in many different ways and inspect the loading time in the browser and send many requests to see the performance differences)

Answer

```
const os = require("os");

const server = require("http").createServer();
const fs = require("fs");
const path = require("path");

const port = 5000;
const filePath = path.join(__dirname, "vid.mp4");

server
  .on("request", (req, res) => {
    var stat = fs.statSync(filePath);

    res.writeHead(200, {
      "Content-Type": "video/mp4",
```

```
      "Content-Length": stat.size
    });


    const data = fs.readFileSync(filePath, {
      "Content-Type": "video/mp4",
      "Content-Length": stat.size
    });
    res.write(data);

  })
  .listen(port, () => console.log(`Listening to port ${port}`));
```

```
    // Fix the slow function to be asynchronous/non-blocking
function slow(callback) {
  setTimeout(() => {
      for (let i = 0; i <= 5e8; i++) { }
  }, 0);

  if (Math.random() > 0.5) {
      return callback("Error", null)
  }

  return callback(null, { id: 12345 })
}

function exec(fn) {
  // Complete the code here to implement chaining with callback
  const callback = function (...restParam) {
      return restParam[0] === null ? restParam[1] : restParam[0];
  }

  const result = fn(callback);

  return {
      done: function (cb) {
          if (result !== "Error") {
              cb(result);
          }
          return this;
      },
      fail: function (cb) {
```

```
          if (result === "Error") {
              cb(result);
          }

          return this;
      }
  }
}

exec(slow).done(function (data) { console.log(data); })
  .fail(function (err) { console.log("Error: " + err); });


    const data = fs.readFileSync(filePath, {
      "Content-Type": "video/mp4",
      "Content-Length": stat.size
    });
    res.write(data);

  })
  .listen(port, () => console.log(`Listening to port ${port}`));
```

Homework -3

3.Using Node Stream API, create a script to unzip a file (after you zip it). (Use zlib.createGunzip() stream)

Answer

```
var fs = require("fs");
var zlib = require('zlib');
fs.createReadStream('./data.txt.gz')
    .pipe(zlib.createGunzip())
    .pipe(fs.createWriteStream('./input.txt'));
console.log("File Decompressed.")


/*Using Node Stream API, create a script to unzip a file.
(use zlib.createGunzip() stream)
*/

// var fs = require("fs");
// var zlib = require('zlib');
```

```
// fs.createReadStream('input.txt').pipe(zlib.createGzip())
// .pipe(fs.createWriteStream('input.txt.gz'));
// console.log("File Compressed.");
```