1 Computational Thinking

1.0.1 Efficiency

Write a program that produces the following output

```
T
TI
TIN
TINK
TINKE
TINKER
TINKERB
TINKERBE
TINKERBEL
TINKERBELL
```

1.0.2 Problem solving

Write a program that produces the following output

```
1. TINKR
2. TINKR
3. TINKR
4. TINKR
5. TINKRBELL
6. TINKR
7. TINKR
8. TINKR
9. TINKR
10. TINKRBELL
etc.
```

1.1 Coding

Chose 1 of the following problems and try to solve it within 5 minutes (choose a program language of your liking)

1. Write a function that sorts an array of integers
2. Write a function that prints the triangle of pascal
3. Write a function that prints the Fibonacci sequence

2 JavaScript

2.0.1 Produce

Fill in the following function

```
/* makes a deep copy of the given object and returns it */
function deepCopy(obj){
  ...
    ...
}
```

2.0.2 Semantics *bonus

Explain the difference between the following 2 functions

```
/* Function example 1 */
<script>
var x = f1(2);

function f1(num){
   return num+2;
}
</script>
```

```
/* Function example 2 */
<script>
var x = f1(2);

var f1 = function(num){
  return num+2;
}
</script>
```

3 MEAN Production environment

3.0.1 callbacks

What is wrong with the following function, and how can you fix the problem

```
route.get("/", function(req, res, next){
  /* retrieve all user objects as an array of users */
  var users = getAllUsers();

  var allUserInfo = [];
  for (var i = 0; i < users.length; i++){

    /* asynchronously fetch more userinfo of an user */
    getUserInfoByUserId(users[i]._id, function(userInfo){
      allUserInfo.push(userInfo);
    }
  }
  res.send({success: true, allUserInfo: allUserInfo});
}
```

3.0.2 Back-end development

```
/****************** Models ******************/
var mongoose = require("mongoose");
var Schema = mongoose.Schema;

var carSchema = new Schema({
    licence: {type: String},
    description: {type: String}
});

var Car = mongoose.model("Cars", carSchema);

/****************** Routes ******************/

/**
```

```
  Fetch all cars from database and return them

  Return format
  {
    success: {type: Boolean},
    cars: [ {type: carObject} ],
    errors: {type: String/Object, default: null}
  }
*/
route.get("/cars", function(req, res, next){
  // TODO.
}
```

3.0.3 Front-end development

```
<script>
function carPageController($scope, $http){
  /* fetch carinfo from API */
}
</script>
```

```
<section ng-controller="carPageController">
  <!-- display all car information in a list -->
</section>
```