National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

**Department of Computing**

# SE-312: Software Construction

**Class: BESE 9AB**

**Lab 03: Intro to ES6**

**Date: 8th March 2021**

**Time: 09:00-11:50pm & 02:00-04:50pm**

**Instructor: Dr. Seema Jehan**

**Lab Engineer: Mr. Aftab Farooq**

**Muhammad Asim Khaskheli**

**BESE-9A**

**250319**

# Lab 03: Intro to ES6

## Objectives

The objective of this lab is helping students to familiarize themselves with basic concepts of the ES6 constructs. They will practice the concept of classes, subclasses, template strings, default parameters, maps, arrow functions and destructuring.

## Tools/Software Requirement

Notepad, browser

**Helping Material:**

File Uploaded on LMS

## Lab Tasks:

## TASK1:

Suppose that you're working in a small-town administration, and you're in charge of two town elements:

1. Parks
2. Streets

It's a very small town, so right now there are only 3 parks and 4 streets. All parks and streets have a name and a build year.

At an end-of-year meeting, your boss wants a final report with the following:

1. Tree density of each park in the town (formula: number of trees/park area)
2. Average age of each town's park (formula: sum of all ages/number of parks)
3. The name of the park that has more than 1000 trees
4. Total and average length of the town's streets
5. Size classification of all streets: tiny/small/normal/big/huge. If the size is unknown, the default is normal

All the report data should be printed to the console.
HINT: Use some of the ES6 features: classes, subclasses, template strings, default parameters, maps, arrow functions, destructuring, etc.

**Solution**

**Task 1 Code:**

```
// Parent class generalInfo that cotnains the general information of a
ll child classes
class GeneralInfo {
  constructor(name, yearBuild) {
    this.name = name;
    this.yearBuild = yearBuild;
  }
}

//Child class Park extending parent class GeneralInfo
class Park extends GeneralInfo {
  constructor(name, yearBuild, area, numOfTrees) {
    super(name, yearBuild);
    this.area = area;
    this.numOfTrees = numOfTrees;
  }

    //method treeDensity that returns the density of trees in a specif
ic park
  treeDensity() {
    const density = this.numOfTrees / this.area;
    console.log(
      `${this.name} has a tree density of ${density} trees per square
km.`
    );
  }
}

//Child class Street extending parent class GeneralInfo
class Street extends GeneralInfo {
  constructor(name, yearBuild, length, size) {
    super(name, yearBuild);
    this.length = length;
    this.size = size;
  }

//method StreetClassification that classifies sreets on the basis of t
heir sizes and then returns them
  StreetClassification() {
```

```
    const streetclass = new Map();
    streetclass.set(1, "tiny");
    streetclass.set(2, "small");
    streetclass.set(3, "normal");
    streetclass.set(4, "big");
    streetclass.set(5, "huge");
    console.log(
      `${this.name}, build in ${this.yearBuild}, is a ${streetclass.ge
t(
        this.size
      )} street.`
    );
  }
}

//Calculation function that calculates sum and avg
function Calculation(arr) {
  const sum = arr.reduce((prev, cur, index) => prev + cur, 0);
  return [sum, sum / arr.length];
}
//Report of Parks generated by reportStreets function
function reportParks(p) {
  console.log("!----PARKS REPORTS----!");
  // Density
  p.forEach((el) => el.treeDensity());

  // Average Age
  const ages = p.map((el) => new Date().getFullYear() - el.yearBuild);
  const [totalAge, avgAge] = Calculation(ages);
  console.log(`Our ${p.length} parks have an average of ${avgAge} year
s.`);

  // Which park has more than 1000 trees;
  const i = p.map((el) => el.numOfTrees).findIndex((el) => el >= 1000)
;
  console.log(`${p[i].name} has more than 1000 trees.`);
}

//Report of streets generated by reportStreets function
function reportStreets(s) {
  console.log("!---- STREETS REPORT ----!");

  //Total and average length of the town's streets
```

```
  const [totalLength, avgLength] = Calculation(s.map((el) => el.length
));
  console.log(
    `Our ${s.length} streets have a total length of ${totalLength} km,
 with an average of ${avgLength} km.`
  );

  //Classify sizes
  s.forEach((el) => el.StreetClassification());
}

const myParks = [
  new Park("Safari Park", 1987, 0.2, 215),
  new Park("F7 Park", 1894, 2.9, 3541),
  new Park("Lake Park", 1953, 0.4, 949),
];

const myStreets = [
  new Street("New Avenue", 1999, 1.1, 4),
  new Street("Evergreen Street", 2008, 2.7, 2),
  new Street("7th Street", 2015, 0.8),
  new Street("Sunset Boulevard", 1982, 2.5, 5),
];


reportParks(myParks);
reportStreets(myStreets);
```

**Task 1 Output Screenshot:**

```
F:\Software Construction\lab 3 - ES6>node Asim.js
!----PARKS REPORTS----!
Safari Park has a tree density of 1075 trees per square km.
F7 Park has a tree density of 1221.0344827586207 trees per square km.
Lake Park has a tree density of 2372.5 trees per square km.
Our 3 parks have an average of 76.33333333333333 years.
F7 Park has more than 1000 trees.
!---- STREETS REPORT ----!
Our 4 streets have a total length of 7.1000000000000005 km, with an average of 1.7750000000000001 km.
New Avenue, build in 1999, is a big street.
Evergreen Street, build in 2008, is a small street.
7th Street, build in 2015, is a undefined street.
Sunset Boulevard, build in 1982, is a huge street.

F:\Software Construction\lab 3 - ES6>
```

**Deliverables**

Compile a single word document by filling in the solution part and submit this Word file on LMS. This lab grading policy is as follows: The lab is graded between 0 to 10 marks. The submitted solution can get a maximum of 5 marks. At the end of each lab or in the next lab, there will be a viva related to the tasks. The viva has a weightage of 5 marks. Insert the solution/answer in this document. You must show the implementation of the tasks in the designing tool, along with your complete Word document to get your work graded. You must also submit this Word document on the LMS. In case of any problems with submissions on LMS, submit your Lab assignments by emailing it to Mr. Aftab Farooq: aftab.farooq@seecs.edu.pk.