НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота № 1.2

з дисципліни

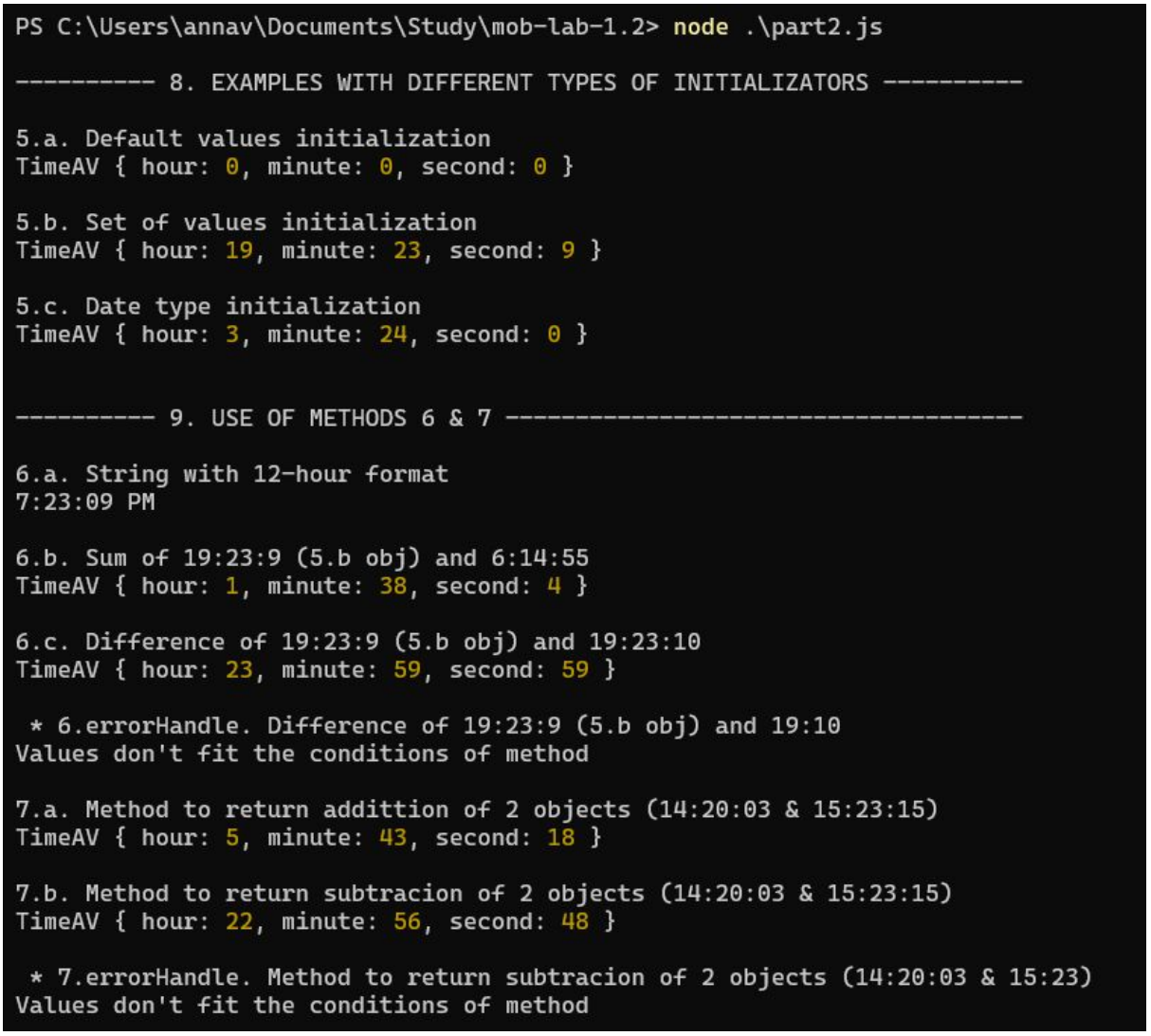
“Розроблення клієнтських додатків для мобільних платформ”

Виконала:
студентка групи ІП-84
ЗК ІП-8402
Анна Василашко

Київ 2021

Варіант 1

Скріншот роботи додатка



Лістинг коду

Частина 1

```
    // Частина 1


    // Дано рядок у форматі "Student1 - Group1; Student2 - Group2; ..."


    let studentsStr =
        "Дмитренко Олександр - ІП-84; Матвійчук Андрій - ІВ-83; Лесик
Сергій - ІО-82; Ткаченко Ярослав - ІВ-83; Аверкова Анастасія - ІО-83;
Соловйов Даніїл - ІО-83; Рахуба Вероніка - ІО-81; Кочерук Давид -
```

```
ІВ-83; Лихацька Юлія- ІВ-82; Головенець Руслан - ІВ-83; Ющенко Андрій - ІО-82; Мінченко Володимир - ІП-83; Мартинюк Назар - ІО-82; Базова Лідія - ІВ-81; Снігурець Олег - ІВ-81; Роман Олександр - ІО-82; Дудка Максим - ІО-81; Кулініч Віталій - ІВ-81; Жуков Михайло - ІП-83; Грабко Михайло - ІВ-81; Іванов Володимир - ІО-81; Востриков Нікіта - ІО-82; Бондаренко Максим - ІВ-83; Скрипченко Володимир - ІВ-82; Кобук Назар - ІО-81; Дровнін Павло - ІВ-83; Тарасенко Юлія - ІО-82; Дрозд Світлана - ІВ-81; Фещенко Кирил - ІО-82; Крамар Віктор - ІО-83; Іванов Дмитро - ІВ-82";


    // Завдання 1
    // Заповніть словник, де:
    // - ключ – назва групи
    // - значення – відсортований масив студентів, які відносяться до відповідної групи

    var studentsGroups = {};

    // Ваш код починається тут

    studentsStr.split(";").forEach((element) => {
      let components = element.split("- ");
      let name = components[0].trim();
      let group = components[1].trim();

      if (studentsGroups[group] === undefined) {
        studentsGroups[group] = [];
      }

      studentsGroups[group].push(name);
    });

    for (var key in studentsGroups) {
      if (studentsGroups.hasOwnProperty(key)) {
        studentsGroups[key].sort();
```

```
    }
  }

  // Ваш код закінчується тут

  console.log("Завдання 1");
  console.log(studentsGroups);
  console.log();

  // Дано масив з максимально можливими оцінками

  let points = [12, 12, 12, 12, 12, 12, 12, 16];

  // Завдання 2
  // Заповніть словник, де:
  // - ключ – назва групи
  // - значення – словник, де:
  //   - ключ – студент, який відносяться до відповідної групи
  //     - значення – масив з оцінками студента (заповніть масив випадковими значеннями, використовуючи функцію `randomValue(maxValue: Int) -> Int`)

  function randomValue(maxValue) {
    switch (Math.ceil(Math.random() * 6)) {
      case 1:
        return Math.ceil(maxValue * 0.7);
      case 2:
        return Math.ceil(maxValue * 0.9);
      case (3, 4, 5):
        return maxValue;
      default:
        return 0;
```

```
    }
  }

  var studentPoints = {};

  // Ваш код починається тут

  for (var key in studentsGroups) {
    if (studentsGroups.hasOwnProperty(key)) {
      let studentDict = {};
      studentsGroups[key].forEach((student) => {
                  let studentPoints = points.map((point) => randomValue(point));
        studentDict[student] = studentPoints;
      });

      studentPoints[key] = studentDict;
    }
  }

  // Ваш код закінчується тут

  console.log("Завдання 2");
  console.log(studentPoints);
  console.log();

  // Завдання 3
  // Заповніть словник, де:
  // - ключ – назва групи
  // - значення – словник, де:
  //   - ключ – студент, який відносяться до відповідної групи
  //   - значення – сума оцінок студента
```

```
var sumPoints = {};

// Ваш код починається тут

for (var group in studentPoints) {
  if (studentPoints.hasOwnProperty(group)) {
    let studentsDict = studentPoints[group];
    for (var student in studentsDict) {
      if (studentsDict.hasOwnProperty(student)) {
        let sum = studentsDict[student].reduce((p, c) => p + c, 0);
        studentsDict[student] = sum;
      }
    }
    sumPoints[group] = studentsDict;
  }
}

// Ваш код закінчується тут

console.log("Завдання 3");
console.log(sumPoints);
console.log();

// Завдання 4
// Заповніть словник, де:
// - ключ – назва групи
// - значення – середня оцінка всіх студентів групи

var groupAvg = {};
```

```
// Ваш код починається тут


for (var group in sumPoints) {
  if (sumPoints.hasOwnProperty(group)) {
    let studentsDict = studentPoints[group];
    let sum = 0;
    let count = 0;
    for (var student in studentsDict) {
      if (studentsDict.hasOwnProperty(student)) {
        sum += studentsDict[student];
        count++;
      }
    }
    groupAvg[group] = sum / count;
  }
}
// Ваш код закінчується тут


console.log("Завдання 4");
console.log(groupAvg);
console.log();


// Завдання 5
// Заповніть словник, де:
// - ключ - назва групи
// - значення - масив студентів, які мають >= 60 балів


var passedPerGroup = {};


// Ваш код починається тут
```

```
    for (var group in sumPoints) {
      if (sumPoints.hasOwnProperty(group)) {
        let passedStudents = [];
        let studentsDict = sumPoints[group];
        for (var student in studentsDict) {
          if (studentsDict.hasOwnProperty(student)) {
            if (studentsDict[student] >= 60) {
              passedStudents.push(student);
            }
          }
        }
        passedPerGroup[group] = passedStudents;
      }
    }

    // Ваш код закінчується тут

    console.log("Завдання 5");
    console.log(passedPerGroup);
```

Частина 2

```
    // Additional: Create custom error to handle errors in methods 6 &
7


    class ValueError extends Error {
      constructor(message) {
        super(message);
      }

      toString() {
        return `${this.name} ${this.message}`;
      }
```

```
}

// 3. Create class (name contains initials)

class TimeAV {
  // 4. Declare properties
  // 5.a. Default values initialization
  hour = 0;
  minute = 0;
  second = 0;

  constructor(...args) {
    if (args.length == 1) {
      //5.c. Date type initialization
      this.hour = args[0].getHours();
      this.minute = args[0].getMinutes();
      this.second = args[0].getSeconds();
    } else if (args.length == 3) {
      // 5.b. Set of values initialization
      const [hour, minute, second] = args;
      hour >= 0 && hour <= 23 ? (this.hour = hour) : 0;
      minute >= 0 && minute <= 59 ? (this.minute = minute) : 0;
      second >= 0 && second <= 59 ? (this.second = second) : 0;
    }
  }

  //6.a. Method to return string with 12-hour format values
  clock() {
    var ampm = this.hour >= 12 ? "PM" : "AM";

    let clockHour = this.hour % 12;
```

```
        clockHour = this.hour ? clockHour : 12;
          let clockMinute = this.minute < 10 ? "0" + this.minute :
this.minute;
          let clockSecond = this.second < 10 ? "0" + this.second :
this.second;

           var strTime = `${clockHour}:${clockMinute}:${clockSecond}
${ampm}`;

        console.log(strTime);
      }

      //6.b. Method to return addittion of 2 objects
      sumTime(...args) {
        try {
          if (args.length == 3) {
            let totalSeconds =
              this.hour * 3600 +
              args[0] * 3600 +
              this.minute * 60 +
              +args[1] * 60 +
              this.second +
              args[2];

            let newHour = Math.floor(totalSeconds / 3600);
            totalSeconds %= 3600;
            let newMinute = Math.floor(totalSeconds / 60);
            let newSeconds = totalSeconds % 60;

            if (newHour >= 24) {
                    console.log(new TimeAV(newHour - 24, newMinute,
newSeconds));
```

```
                } else console.log(new TimeAV(newHour, newMinute,
newSeconds));
            } else throw new ValueError("Values don't fit the conditions
of method");
        } catch (e) {
          console.log(e.message);
        }
      }

      // 6.c. Method to return subtracion of 2 objects
      diffTime(...args) {
        try {
          if (args.length == 3) {
            let totalSeconds =
              this.hour * 3600 +
              this.minute * 60 +
              this.second -
              (args[0] * 3600 + args[1] * 60 + args[2]);

            let newHour = Math.floor(totalSeconds / 3600);
            totalSeconds %= 3600;
            let newMinute = Math.floor(totalSeconds / 60);
            let newSeconds = totalSeconds % 60;

            if (newHour < 0 && newMinute < 0 && newSeconds < 0) {
              console.log(
                  new TimeAV(newHour + 24, newMinute + 60, newSeconds +
60)
              );
            } else if (newHour < 0 && newMinute < 0) {
                console.log(new TimeAV(newHour + 24, newMinute + 60,
newSeconds));
            } else if (newHour < 0) {
```

```
                    console.log(new TimeAV(newHour + 24, newMinute,
newSeconds));
                } else console.log(new TimeAV(newHour, newMinute,
newSeconds));
          } else throw new ValueError("Values don't fit the conditions
of method");
        } catch (e) {
          console.log(e.message);
        }
      }
    }

    const sumTwoObj = (...args) => {
      try {
        if (args.length == 6) {
          let totalSeconds =
            args[0] * 3600 +
            args[3] * 3600 +
            args[1] * 60 +
            args[4] * 60 +
            args[2] +
            args[5];

          let newHour = Math.floor(totalSeconds / 3600);
          totalSeconds %= 3600;
          let newMinute = Math.floor(totalSeconds / 60);
          let newSeconds = totalSeconds % 60;

          if (newHour >= 24) {
                    console.log(new TimeAV(newHour - 24, newMinute,
newSeconds));
                } else console.log(new TimeAV(newHour, newMinute,
newSeconds));
```

```
        } else throw new ValueError("Values don't fit the conditions of
method");
      } catch (e) {
        console.log(e.message);
      }
    };

    const diffTwoObj = (...args) => {
      try {
        if (args.length == 6) {
          let totalSeconds =
            args[0] * 3600 +
            args[1] * 60 +
            args[2] -
            (args[3] * 3600 + args[4] * 60 + args[5]);

          let newHour = Math.floor(totalSeconds / 3600);
          totalSeconds %= 3600;
          let newMinute = Math.floor(totalSeconds / 60);
          let newSeconds = totalSeconds % 60;

          if (newHour < 0 && newMinute < 0 && newSeconds < 0) {
                console.log(new TimeAV(newHour + 24, newMinute + 60,
newSeconds + 60));
          } else if (newHour < 0 && newMinute < 0) {
                console.log(new TimeAV(newHour + 24, newMinute + 60,
newSeconds));
          } else if (newHour < 0) {
                    console.log(new TimeAV(newHour + 24, newMinute,
newSeconds));
                } else console.log(new TimeAV(newHour, newMinute,
newSeconds));
```

```
        } else throw new ValueError("Values don't fit the conditions of
method");
      } catch (e) {
        console.log(e.message);
      }
    };

    // ---------- 8. EXAMPLES WITH DIFFERENT TYPES OF INITIALIZATORS
----------

    console.log();
    console.log(
      "---------- 8. EXAMPLES WITH DIFFERENT TYPES OF INITIALIZATORS
----------"
    );
    console.log();

    // 5.a. Default values initialization
    let defaultTime = new TimeAV();
    console.log("5.a. Default values initialization");
    console.log(defaultTime);
    console.log();

    // 5.b. Set of values initialization
    let setTime = new TimeAV(19, 23, 9);
    console.log("5.b. Set of values initialization");
    console.log(setTime);
    console.log();

    //5.c. Date type initialization
    let dateTime = new TimeAV(new Date("December 17, 1995 03:24:00"));
    console.log("5.c. Date type initialization");
```

```
    console.log(dateTime);
    console.log();

    //     ----------     9.     USE     OF     METHODS     6     &     7
------------------------------------


    console.log();
    console.log(
            "----------     9.     USE     OF     METHODS     6     &     7
------------------------------------"
    );
    console.log();

    //6.a. Method to return string with 12-hour format values
    console.log("6.a. String with 12-hour format");
    setTime.clock();
    console.log();

    //6.b. Method to return addittion of prev & new objects
    console.log(`6.b. Sum of 19:23:9 (5.b obj) and 6:14:55`);
    setTime.sumTime(6, 14, 55);
    console.log();

    // 6.c. Method to return subtracion of prev & new objects
    console.log(`6.c. Difference of 19:23:9 (5.b obj) and 19:23:10`);
    setTime.diffTime(19, 23, 10);
    console.log();

    // * 6.errorHandle. Method to return subtracion of prev & new
objects
    console.log(` * 6.errorHandle. Difference of 19:23:9 (5.b obj) and
19:10`);
    setTime.diffTime(19, 10);
```

```
    console.log();


    // 7.a. Method to return addittion of 2 objects
    console.log(
        "7.a. Method to return addittion of 2 objects (14:20:03 &
15:23:15)"
    );
    sumTwoObj(14, 20, 3, 15, 23, 15);
    console.log();


    // 7.b. Method to return subtracion of 2 objects
    console.log(
        "7.b. Method to return subtracion of 2 objects (14:20:03 &
15:23:15)"
    );
    diffTwoObj(14, 20, 3, 15, 23, 15);
    console.log();


    // * 7.errorHandle. Method to return subtracion of 2 objects
    console.log(
        " * 7.errorHandle. Method to return subtracion of 2 objects
(14:20:03 & 15:23)"
    );
    diffTwoObj(14, 20, 3, 15, 23);
    console.log();
```

Висновок

У ході виконання даної лабораторної роботи ми розглянули та навчились працювати з об’єктами і класами у мові програмування JavaScript. Окрім того ми навчились ініціалізувати об’єкти різними способами. Також ми розібрались як створювати методи в межах та поза межами класу. Додатково було розглянуто тему наслідування та обробку помилок.