 МІНІСТЕРСТВО ОСВІТИ I НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ

КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**Комп’ютерний практикум №5**

з дисципліни «Веб-технології та веб-дизайн-1»

**Виконав:**

студентка гр. БС-81

Кисельова І.Д.

**Перевірив:**

ас. каф. БМК Матвійчук О.В.

Зараховано від \_\_\_.\_\_\_.\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис викладача)

Київ-2020

**Завдання:**

**1. Завдання обов'язкове для всіх:**

Фіскальний кодекс

Кожна особа в Італії має унікальний ідентифікаційний ідентифікаційний код, виданий національною податковою службою після реєстрації народження: Фіскальний кодекс (Codice Fiscale). [https://en.wikipedia.org/wiki/Italian\_fiscal\_code\_card]

Враховуючи об'єкт, що містить особисті дані людини (ім'я, прізвище, стать та дата народження), поверніть 11 символів коду як рядок, виконуючи наступні кроки:

Створіть 3 символи прізвища, якщо воно має:

Принаймні 3 приголосні, тоді використовуються перші три приголосні. (Ньюмен -> NWM).

Менше ніж 3 приголосні, тоді голосні замінять відсутні літери в тому ж порядку, в якому вони відображаються (Fox -> FXO | Hope -> HPO).

Менше трьох літер тоді "X" займе третю позицію після приголосного та голосного (Yu -> YUX).

Створіть від імені 3 великі літери, якщо у них є:

Рівно 3 приголосних тоді приголосні вживаються в тому порядку, в якому вони з’являються (Метт -> МТТ).

Вживається більше 3 приголосних, тоді перший, третій та четвертий приголосні (Саманта -> СНТ | Томас -> ТМС).

Менше трьох приголосних, тоді голосні замінять відсутні літери в тому ж порядку, в якому вони відображаються (Bob -> BBO | Paula -> PLA).

Менше трьох літер тоді "X" займе третійю позицію після приголосного та голосного (Al -> LAX).

Утворіть 2 числа, 1 букву та 2 числа від дати народження та статі:

Візьміть дві останні цифри року народження (1985 -> 85).

Створіть лист, відповідний місяцю народження (січень -> A | грудень -> T), використовуючи таблицю для перерахунку, включену в код.

Для чоловіків приймають день народження, додаючи один нуль на початку, якщо він менше 10 (будь-який 9-й день -> 09 | будь-який 20-й день -> 20).

Для жінок беруть день народження і сумують до нього 40 (будь-який 9-й день -> 49 | будь-який 20-й день -> 60).

PS:

· Букви в відповіді мають бути в uppercase

· Дата народження в форматі D/M/YYYY

· Таблиця перетворення наведена в завданні

· Для людини створити клас

· Відображення місяця в літеру: // const months = { 1: "A", 2: "B", 3: "C", 4: "D", 5: "E", 6: "H", 7: "L", 8: "M", 9: "P", 10: "R", 11: "S", 12: "T" }

Приклад:

fiscalCode({

name: "Matt",

surname: "Edabit",

gender: "M",

dob: "1/1/1900"

}) ➞ "DBTMTT00A01"

fiscalCode({

name: "Helen",

surname: "Yu",

gender: "F",

dob: "1/12/1950"

}) ➞ "YUXHLN50T41"

Завдання по варіантах:

**2. Відстань до найближчого голосного**

Напишіть функцію, яка займає рядок і для кожного символу повертає відстань до найближчого голосного. Якщо символ сам є голосним, поверніть 0.

мова роботи функції англійська, тому голосними є “a, e, i, o, u”, також всі букви будуть в lowercase.

Приклад:

distanceToNearestVowel("abcdabcd") ➞ [0, 1, 2, 1, 0, 1, 2, 3]

distanceToNearestVowel("shopper") ➞ [2, 1, 0, 1, 1, 0, 1]

**9. Симетричні візерунки**

Кетлін володіє прекрасним магазином килимів. Їй подобається групувати килими в 4 взаємовиключні категорії.

· недосконалий => imperfect

· горизонтально симетричний => horizontally symmetric

· вертикально симетричний => vertically symetric

· досконалий => perfect

Недосконалий килим - це ні горизонтально, ні вертикально симетрично. Ось приклад недосконалого килима:

[

["a", "a", "a", "a"],

["a", "a", "a", "a"],

["a", "a", "b", "b"]

]

Далі - горизонтально симетричний килим. Ви можете "скласти" килим через гіпотетичну вісь x, і обидві сторони були б однаковими. Горизонтально симетричний килим не є вертикально симетричним (інакше цей килимок був би класифікований як ідеальний).

[

["c", "a", "a", "a"],

["b", "b", "b", "b"],

["c", "a", "a", "a"]

]

Далі - вертикально симетричний килим. Ви можете "скласти" килим через гіпотетичну вісь y, і обидві сторони були б однаковими. Вертикально симетричний не є горизонтально симетричним (інакше цей килимок був би класифікований як ідеальний).

[

["a", "b", "a"],

["b", "b", "b"],

["a", "b", "a"],

["a", "b", "a"]

]

Нарешті, ідеальний килим - це вертикально і горизонтально симетричний. Тобто, складені або по довжині, або по ширині дадуть дві однакові частини.

[

["a", "b", "b", "a"],

["b", "b", "b", "b"],

["a", "b", "b", "a"]

]

Давши килим розміром m x n, визначте, чи він недосконалий, горизонтально симетричний, вертикально симетричний чи ідеальний. Килимки представлені за допомогою двовимірного масиву.

Приклад:

classifyRug([

["a", "a"],

["a", "a"]

]) ➞ "perfect"

classifyRug([

["a", "a", "b"],

["a", "a", "a"],

["b", "a", "a"]

]) ➞ "imperfect"

classifyRug([

["b", "a"],

["b", "a"]

]) ➞ "horizontally symmetric"

classifyRug([

["a", "a"],

["b", "b"]

]) ➞ "vertically symmetric"

**Посилання на репозиторій github:**

<https://github.com/Irines/WEB_commits>

**Виконання роботи:**

**Task1**

class User {

    constructor(newName, newSurname, newGender, newBirthdate) {

*this*.name = newName;

*this*.surname = newSurname;

*this*.gender = newGender;

*this*.birthdate = newBirthdate;

    }

    set name (newName) { // TODO: move validation up

        if (*this*.\_validateInput(newName)) {

*this*.\_name = newName;

         }

    }

    set surname (newSurname) {

      if (*this*.\_validateInput(newSurname)) {

*this*.\_surname = newSurname;

       }

    }

    set birthdate (newBirthdate)

    {

      if (*this*.\_validate\_date(newBirthdate)) {

*this*.\_birthdate = newBirthdate;

      } else {

        alert("Введена некорректная дата!");

      }

    }

    \_validateInput(value)

    {

      if(value==="") {

        alert('Input at list 1 symbol in name or surname field');

      } else {

        return true;

      }

    }

    /\* Функция разбивает дату на составляющие (метод split()),

    а затем выполняет проверку составляющих при помощи объекта Date и методов getFullYear(), getMonth() и getDate(). \*/

    // arrD[1] -= 1 Потому что у объекта Date отсчет месяцев начинается с 0.

    \_validate\_date(value){

        let arrD = value.split("/");

        arrD[1] -= 1;

        let d = new Date(arrD[2], arrD[1], arrD[0]);

        if ((d.getFullYear() == arrD[2]) && (d.getMonth() == arrD[1]) && (d.getDate() == arrD[0])) {

          return true;

        }

        else {

          return false;

        }

    }

    surnameCode() {

      let letters = *this*.\_separateLetters(*this*.\_surname);

      let vowels = letters.get("vowels");

      let consonants =letters.get("consonants");

      return *this*.\_codeFromSurname(vowels,consonants);

    }

    nameCode() {

      let letters = *this*.\_separateLetters(*this*.\_name);

      let vowels = letters.get("vowels"), consonants = letters.get("consonants");

      return *this*.\_codeFromName(vowels,consonants);

    }

    \_codeFromSurname(vowels,consonants) {

        let code = '';

        for (let i = 0, j = 0; code.length < 3;) {

            if (typeof consonants[i] !== 'undefined') {

              code += consonants[i++];

            }

            else if (typeof vowels[j] !== 'undefined') {

              code += vowels[j++];

            }

            else {

              code += "X";

            }

        }

        return code;

    }

    \_codeFromName(vowels, consonants) {

        let code = '';

        if (consonants.length == 3)

        {

            code = consonants.slice(0,2).join('');

        }

        else if (consonants.length > 3)

        {

            code += consonants[0];

            code += consonants[2];

            code += consonants[3];

        }

        else if (consonants.length < 3)

        {

            for (let i = 0, j = 0; code.length < 3;) {

                if (typeof consonants[i] !== 'undefined') {

                  code += consonants[i++];

                }

                else if (typeof vowels[j] !== 'undefined') {

                  code += vowels[j++];

                }

                else {

                  code += "X";

                }

            }

        }

        return code;

    }

    // Наличие символа в массиве гласных можно проверить с помощью indexOf

    \_isVowel(char) {

        return ['a', 'e', 'i', 'o', 'u'].indexOf(char.toLowerCase()) !== -1; // todo: arr.includes(a);

    }

    \_separateLetters(someData){

        let vowels = [];

        let consonants = [];

        for (let i = 0; i < someData.length; i++) {

            // метод charAt() возвращает символ по заданному индексу внутри строки

          let char = someData.charAt(i);

          if (*this*.\_isVowel(char)) {

            vowels.push(char);

          } else {

            consonants.push(char);

          }

        }

        // Используется Map(), чтобы вернуть два массива.

        let letters = new Map();

        letters.set("vowels", vowels);

        letters.set("consonants", consonants);

        return letters;

    }

   /\*  Функция parseInt(строка, основание) выполняет синтаксический разбор строки начиная с первого символа,

    если первый символ является цифрой или знаком (- или +),

    то она переходит к обработке второго символа и так далее,

    пока не будет обнаружен символ, который не может быть преобразован в числовое значение,

    после этого она возвращает полученное целое число \*/

    \_parseDate(date) { // TODO: no variables from upper case

        const parsedDate = date.split("/").map(function (value) {

          return parseInt(value, 10);

        })

        return parsedDate;

      }

    \_codeFromBirthDate() {

        const months = {

          1: "A", 2: "B", 3: "C", 4: "D", 5: "E", 6: "F",

          7: "G", 8: "H", 9: "I", 10: "G", 11: "K", 12: "T"

        };

        let date = *this*.\_parseDate(*this*.\_birthdate);

        // adding '0' to yearCode/dayCode if there is only 1 number

        let yearCode = date[2] % 100;

        yearCode = yearCode < 10 ? '0' + yearCode : yearCode;

        let dayCode = *this*.gender === "Female" ? date[0] + 40 : date[0];

        dayCode = dayCode < 10 ? '0' + dayCode : dayCode;

        return yearCode + months[date[1]] + dayCode;

      }

      getCode() {

        return (*this*.surnameCode().toUpperCase() + *this*.nameCode().toUpperCase()

          + *this*.\_codeFromBirthDate()).toUpperCase();

      }

}

function makeCode() {

    let user = new User(document.getElementById('name').value,

    document.getElementById('surname').value,

    document.getElementById('gender').value,

    document.getElementById('bday').value);

    document.getElementById('fiscal-code').innerHTML = user.getCode();

}

**Task2**

function vowelLen() {

  const vowels = 'eyuioa';

  const vowelsIndex = [];

  const result = [];

  const input = document.getElementById('harsh').value;

  // const input = 'abcdefgaaaghjn';

  for (let i in input) {

    if (vowels.includes(input[i])) {

      vowelsIndex.push(i);

    }

  }

  for (let i in input) {

    if (vowelsIndex.includes(i)) {

      result.push(0);

    } else {

      let min = input.length;

      for (let vi of vowelsIndex) {

        min = Math.min(min, Math.abs(i - vi));

      }

      result.push(min);

    }

  }

 console.log(result);

 document.getElementById('length').innerHTML ="[ " + result + " ]";

}

**Task3**

function classifyrug()

{

  //let str = "aaaa,aaaa,aaaa";

  //input like aaaa,bbbb,cccc without spaces

  let  str= document.getElementById('rugInput').value;

  let rug=str.split(',').map( i=> i.split(""));

  //console.log(rug);

  function symmetryHorizontal()

  {

    for (let i=0; i<Math.floor(rug.length/2); i++)

    {

      if (rug.length%2 != 0)

      {

        if(i == Math.floor(rug.length/2))

          continue;

      }

      for (let j=0; j<rug[0].length; j++)

      {

        if (rug[i][j] !== rug[rug.length - 1 - i][j])

        {

          return false;

        }

      }

    }

    return true;

  }

  function symmetryVertical()

  {

    for (let j = 0; j < Math.floor(rug[0].length / 2); j++)

    {

      if (rug[0].length%2 != 0)

      {

        if(j == Math.floor(rug[0].length/2))

          continue;

      }

      for (let i=0; i<rug.length;i++)

      {

        if (rug[i][j] !== rug[i][rug[0].length- 1 - j])

          return false;

      }

    }

    return true;

  }

  if (symmetryHorizontal() && symmetryVertical())

    document.getElementById('rugRes').innerHTML = "perfect rug";

  else if(symmetryHorizontal())

    document.getElementById('rugRes').innerHTML ="horizontally symmetric";

  else if(symmetryVertical())

    document.getElementById('rugRes').innerHTML ="vertically symmetric";

  else  document.getElementById('rugRes').innerHTML ="imperfect";

}