МІНІСТЕРСТВО ОСВІТИ I НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ

КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**Лабораторна робота №5**

з дисципліни «Веб технології та веб-дизайн»

**Виконав:**

студент гр. БС-82

Нікіта Воронін

**Перевірив:**

Ас. каф БМК Матвійчук О.В.

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

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

(підпис викладача)

Київ-2020

**Завдання:**

1. **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).

1. **Виткані струни**

Створіть функцію, яка повертає масив заданого рядка, але зміщений пробілами. Ось кілька більш точних інструкцій:

Продовжуйте додавати пробіли зліва, поки у вас не буде однакової кількості пробілів, як довжина слова.

Потім продовжуйте видаляти пробіли, поки не досягнете початкового слова.

*Приклад:*

wiggleString("hello") ➞ [

"hello",

 " hello",

 "  hello",

 "   hello",

 "    hello",

 "     hello"

 "    hello",

 "   hello",

 "  hello",

 " hello",

 "hello"

 ]

1. **Наполегливість**

Аддитивна стійкість цілого числа, n - це кількість разів, яку вам доведеться замінити n сумою її цифр, поки n не стане одноцифровим цілим числом.

Мультиплікативна стійкість цілого числа, n - це кількість разів, коли вам доведеться замінити n на добуток його цифр, поки n не стане одноцифровим цілим числом.

Створіть дві функції, які беруть ціле число в якості аргументу і:

Поверніть її адитивну стійкість.

Поверніть його мультиплікативну стійкість.

*Приклади:*

additivePersistence(1679583) ➞ 3

// 1 + 6 + 7 + 9 + 5 + 8 + 3 = 39

// 3 + 9 = 12

// 1 + 2 = 3

// It takes 3 iterations to reach a single-digit number.

additivePersistence(123456) ➞ 2

// 1 + 2 + 3 + 4 + 5 + 6 = 21

// 2 + 1 = 3

additivePersistence(6) ➞ 0

// Because 6 is already a single-digit integer.

multiplicativePersistence(77) ➞ 4

// 7 x 7 = 49

// 4 x 9 = 36

// 3 x 6 = 18

// 1 x 8 = 8

// It takes 4 iterations to reach a single-digit number.

multiplicativePersistence(123456) ➞ 2

// 1 x 2 x 3 x 4 x 5 x 6 = 720

// 7 x 2 x 0 = 0

multiplicativePersistence(4) ➞ 0

// Because 4 is already a single-digit integer.

**Лістинг:**

**// fiscalCode.js**

class Person {

  constructor(name, surname, gender, birthdate) {

    this.name = name;

    this.surname = surname;

    this.gender = gender;

    this.birthdate = birthdate;

  }

}

function get\_surname(person, consonants, vowels) {

  let fc\_surname = "";

  person.surname = person.surname.toUpperCase();

  if (person.surname.length >= 3) {

    let consonantAmount = get\_consonants(person, consonants, person.surname);

    if (consonantAmount >= 3)

      fc\_surname = set\_fc\_letters(person.surname, consonants);

    else {

      fc\_surname = set\_fc\_letters(person.surname, consonants);

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

        for (let j = 0; j < vowels.length; j++) {

          if (person.surname[i] == vowels[j] && fc\_name.length < 3)

            fc\_name += person.surname[i];

        }

      }

    }

  }

  else {

    fc\_surname = person.surname;

    while (fc\_surname.length != 3)

      fc\_surname += "X";

  }

  return fc\_surname;

}

function get\_name(person, consonants, vowels) {

  fc\_name = "";

  person.name = person.name.toUpperCase();

  if (person.name.length >= 3) {

    let consonantAmount = get\_consonants(person, consonants, person.name);

    if (consonantAmount > 3) {

      for (let i = 0, count = 0; i < person.name.length; i++) {

        for (let j = 0; j < consonants.length; j++) {

          if (person.name[i] == consonants[j] && fc\_name.length < 3) {

            count++;

            if (count != 2)

              fc\_name += person.name[i];

          }

        }

      }

    }

    else if (consonantAmount == 3)

      fc\_name = set\_fc\_letters(person.name, consonants);

    else {

      fc\_name = set\_fc\_letters(person.name, consonants);

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

        for (let j = 0; j < vowels.length; j++) {

          if (person.name[i] == vowels[j] && fc\_name.length < 3)

            fc\_name += person.name[i];

        }

      }

    }

  }

  else {

    fc\_name = person.name;

    while (fc\_name.length != 3)

      fc\_name += "X";

  }

  return fc\_name;

}

function get\_birthdate(person, consonants, vowels) {

  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"};

  let birthdate\_array = person.birthdate.split("/");

  let fc\_year = birthdate\_array[2][2] + birthdate\_array[2][3];

  let fc\_month = months[parseInt(birthdate\_array[1])];

  let fc\_day = "";

  if (person.gender == "M")

    if (birthdate\_array[0].length == 1)

      fc\_day = "0" + birthdate\_array[0];

    else

      fc\_day = birthdate\_array[0];

  else

    fc\_day = parseInt(birthdate\_array[0]) + 40;

  return fc\_year + fc\_month + fc\_day;

}

function get\_consonants(person, consonants, obj) {

  let consonantAmount = 0;

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

    for (let j = 0; j < consonants.length; j++) {

      if (obj[i] == consonants[j])

        consonantAmount += 1;

    }

  }

  return consonantAmount;

}

function set\_fc\_letters(obj, letters) {

  fc\_name = "";

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

    for (let j = 0; j < letters.length; j++) {

      if (obj[i] == letters[j] && fc\_name.length < 3)

        fc\_name += obj[i];

    }

  }

  return fc\_name;

}

function fiscalCode(person, consonants, vowels) {

  fc\_surname = get\_surname(person, consonants, vowels);

  fc\_name = get\_name(person, consonants, vowels);

  fc\_birthdate = get\_birthdate(person, consonants, vowels);

  console.log(fc\_surname+fc\_name+fc\_birthdate);

}

const vowels = ['A','E','U','I','O'];

const consonants = ['Q','W','R','T','Y','P','S','D','F','G','H','J','K','L','Z','X','C','V','B','N','M'];

person = new Person("Dima", "Lobunko", "M", "1/1/1900");

fiscalCode(person, consonants, vowels);

**// functions.js**

function deleteFirstWord(string) {

  let array\_of\_string = string.split("");

  array\_of\_string.shift();

  string = array\_of\_string.join("");

  return string;

}

function stringWithSpaces(string) {

  const string\_array = [];

  const len = string.length;

  for (let i = 0; i < len; i++) {

    string\_array.push(string);

    string = " " + string;

  }

  string = deleteFirstWord(string);

  for (let i = 0; i < len - 1; i++) {

    string = deleteFirstWord(string);

    string\_array.push(string);

  }

  return string\_array;

}

function additivePersistance(number) {

  let count = 0;

  for (;number.toString().length > 1; count++) {

    let result = 0;

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

      result += Number(number.toString()[i]);

    }

    number = result;

  }

  return count;

}

function multiplicativePersistence(number) {

  let count = 0;

  for (;number.toString().length > 1; count++) {

    let result = 1;

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

      result \*= Number(number.toString()[i]);

    }

    number = result;

  }

  return count;

}