 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»

**Виконав:**

студент гр. БС-81

Сєров О. В.

**Перевірив:**

ас. Матвійчук О. В.

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

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

(підпис викладача)

Київ – 2020

**Завдання:**

В результаті у кожного мають бути функції для виконання 3х завдань,

1 - обов’язкове для всіх,

2 - variant mod 6

3 - (((group mod 10) + variant) mod 6) + 6

**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).

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).

*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"

Код:

let any\_name = 'Helen';

let any\_surname = 'Yu';

let any\_sex = 'F';

let any\_dob = '1/12/1950';

class Person

{

    constructor(Name, Surname, Sex, Dob)

    {

        this.name = Name;

        this.surname = Surname;

        this.sex = Sex;

        this.dob = Dob;

    }

    get\_fiscal\_code()

    {

        return (this.surname\_fiscal(this.surname) + this.name\_fiscal(this.name) +  this.date\_fiscal());

    }

    surname\_fiscal(Surname)

    {

        const vowels = "aeiou";

        Surname = Surname.toLowerCase();

        let code = '';

        let letters = 0;

        for (let i of Surname)

        {

            let isVowel = false;

            for (let j of vowels)

            {

                if (j == i)

                    isVowel = true;

            }

            if (!isVowel && letters < 3)

            {

                code += i;

                letters++;

            }

        }

        if (letters < 3)

        {

            for (let i of Surname)

            {

                for (let j of vowels)

                {

                    if (j == i)

                    {

                        code += i;

                        letters++;

                    }

                }

                if (letters == 3)

                    break;

            }

        }

        if (letters < 3)

        {

            for (let i = 0; i < (3 - Surname.length); i++)

                code += 'x';

        }

        return code.toUpperCase();

    }

    name\_fiscal(Name)

    {

        const vowels = "aeiou";

        Name = Name.toLowerCase();

        let code = '';

        let letters = 0;

        for (let i of Name)

        {

            let isVowel = false;

            for (let j of vowels)

            {

                if (j == i)

                    isVowel = true;

            }

            if (!isVowel && letters < 4)

            {

                code += i;

                letters++;

            }

        }

        if (letters == 4)

            code = code.replace(code[1], '');

        if (letters < 3)

        {

            for (let i of Name)

            {

                for (let j of vowels)

                {

                    if (j == i)

                    {

                        code += i;

                        letters++;

                    }

                }

                if (letters == 3)

                    break;

            }

        }

        if (letters < 3)

        {

            for (let i = 0; i < (3 - Name.length); i++)

                code += 'x';

        }

        return code.toUpperCase();

    }

    date\_fiscal() {

        const letter\_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" }

        var dobArray = this.dob.split('/');

        let lastChars = dobArray[2].slice(2);

        lastChars += letter\_months[Number(dobArray[1])];

        lastChars += (this.sex.toLowerCase() === "m") ? ((dobArray[0].length > 1) ? dobArray[0] : "0"+dobArray[0]) : (Number(dobArray[0])+40).toString();

        return lastChars;

    }

}

function output\_fiscal\_code()

{

    let human = new Person (any\_name, any\_surname, any\_sex, any\_dob);

    let code = human.get\_fiscal\_code();

    console.log('--- Output fiscal code here ---\n');

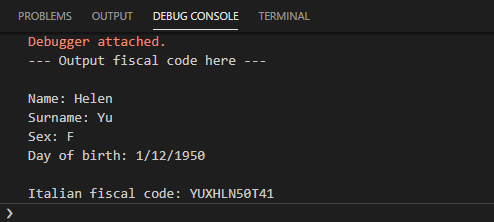
    console.log('Name:', any\_name, '\nSurname:', any\_surname, '\nSex:', any\_sex, '\nDay of birth:', any\_dob);

    console.log("\nItalian fiscal code:", code);

}

output\_fiscal\_code()

Результат:



**Завдання по варіантах:**

**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"

]

Код:

function wiggle\_string(wiggle\_it) {

    let arr = [];

    len = wiggle\_it.length;

    arr.push(wiggle\_it)

    for (let i = 0; i < len; i++)

    {

        wiggle\_it = ' ' + wiggle\_it;

        arr.push(wiggle\_it)

    }

    for (let i = 0; i < len; i++)

    {

        wiggle\_it = wiggle\_it.slice(1);

        arr.push(wiggle\_it)

    }

    return arr

}

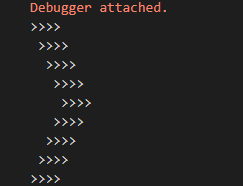
var X = wiggle\_string('>>>>');

for (var i of X) {

    console.log(i)

}

Результат:



**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.

Код:

var X = 1679583;

var Y = 77;

function AdditivePersistence(num) {

  var numbers = num.toString().split('');

  var count = 0;

  if (numbers.length <= 1) {

    return count;

  };

  do {

    var sum = 0;

    for (var i = 0; i<numbers.length; i++) {

      sum = sum + Number(numbers[i]);

    }

    numbers = sum.toString().split('');

    count = count + 1;

  } while (numbers.length > 1);

  return count;

};

function MuliplicativePersistence(num) {

  var times = 0;

  num = num.toString();

  while (num.length > 1) {

      times++;

      num = num.split('').map(Number).reduce((a, b) => a \* b).toString();

  }

  return times;

}

console.log('Additive persistence of', X, 'is', AdditivePersistence(X), '!');

console.log('Muliplicative persistence of', Y, 'is', MuliplicativePersistence(Y), '!');

Результат:

