МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ»
ФАКУЛЬТЕТ МАТЕМАТИКИ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
Кафедра системного аналізу та теорії оптимізації

А.Ю.Брила, Ю.В.Андрашко, М.М. Ломага

ФУНКЦІЇ У JAVASCRIPT

Методичні вказівки до лабораторних робіт з дисципліни «WEB програмування

Ужгород 2023

Функції у JavaScript. (Методичні вказівки до лабораторних робіт з дисципліни «WEB програмування»). / Укладачі: А.Ю. Брила, Ю.В. Андрашко, М.М. Ломага. Ужгород, 2023. 16 с.

Навчальний матеріал методичних вказівок призначений для аудиторної і самостійної підготовки студентів при вивченні дисципліни «WEB програмування».

Основна мета самостійної роботи студента – закріплення теоретичних відомостей, які викладаються на лекціях, та вміння їх застосувати, розв'язуючи задачі, що виникають на практиці. Завдання розроблених методичних матеріалів полягає в чіткій, цілеспрямованій допомозі студентам в організації самостійної підготовки до практичних занять з дисципліни «WEB програмування».

Методичні вказівки призначені для студентів різних напрямків підготовки.

.

Рецензенти:

к.ф.-м.н., доц. Погоріляк О.О.,

к.ф.-м.н., доц. Млавець Ю.Ю.

Рекомендовано до друку:

Кафедрою системного аналізу та теорії оптимізації (Протокол №11 від 16 червня 2023 року);

Науково-методичною комісією факультету математики та цифрових технологій, (Протокол № 10 від 20 червня 2023 року);

Вченою радою факультету математики та цифрових технологій ДВНЗ "Ужгородський національний університет", (Протокол №10 від 21 червня 2023 року).

ФУНКЦІЇ

Допоміжні алгоритми у JavaScript

Функції є реалізацією допоміжних алгоритмів. Вони є основою структурного підходу до розробки алгоритмів. При цьому складна задача розбивається на більш простіші підзадачі (принцип *декомпозиції*), а кожна із підзадач розв'язується за допомогою відповідної функції.

Розглянемо загальний вигляд опису функції

Загальний вигляд	`function <ім'я> (<список формальних параметрів>){`
	`  <тіло функції (оператори)>`
	`}`

Приклад	`function get_max(number_1, number_2){`
	`  if (number_1>number_2)`
	`  return number_1;`
	`  else`
	`  return number_2;`

Список формальних параметрів – це список позначень величини, які є необхідними для роботи функції. Так у наведеному прикладі для знаходження найбільшого серед двох чисел необхідно мати ці два числа `number_1`, `number_2`. Формально, ми могли б позначити ці величини довільними іменами, але на практиці потрібно намагатися позначати величини таким чином, щоб з назви було зрозуміло їх призначення. Зауважимо при цьому, що при розробці функції ми абстрагуємось від конкретних значень, для яких буде викликано дану функцію (застосовуємо принцип *абстракції*). Функцію потрібно розробляти так, щоб її можна було використати для довільних значень формальних параметрів.

У процесі свого виконання функція може обчислювати деякі величини, які є результатами роботи (у даному випадку найбільше серед двох чисел). Результати роботи функції можуть бути повернені у точку виклику за допомогою службового слова `return`

`return <результат>`

Якщо функція не повертає результату за допомогою `return`, то вважається, що функція повертає значення `undefined`, яке у логічному виразі приводиться до значення `false`.

Виклик функції здійснюється за її ім'ям. При цьому у дужках необхідно вказати список фактичних параметрів. Значення цих фактичних параметрів буде скопійовано у відповідні формальні параметри.

Загальний вигляд	<ім’я функції> (<список фактичних параметрів>)

Приклад	// ----- Опис функції -------
	function **get_max**(number_1, number_2){
	    if (number_1>number_2)
	        return number_1;
	    else
	        return number_2;
	// ------ Виклик фунції -------
	let result_1=**get_max(2,6);**
	alert("result_1="+result_1);
	let a=8;
	let b=7;
	let result_2=**get_max(a,b)**
	alert("result_2="+result_2)

Локальні і глобальні змінні

Область видимості змінної – це частина програми, де до цієї змінної можна звернутись. Блок – це послідовність операторів тіла функції або всього файлу. Область видимості змінної обмежується блоком, у якому цю змінну було ініціалізовано деяким значенням а також усіма вкладеними у даний блоками.

Значення, які ініціалізовано в тексті програми (за межами функції) є *глобальними* змінними і можуть бути використані у будь-якому вкладеному блоці (будь-якій функції). Змінні, які ініціалізовані в межах функції є *локальними* змінними. Їх областю видимості є функція, у якій вони були ініціалізовані.

Приклад.

```
let a=5;                ← глобальна змінна
function func(){
    alert(a);     ← звертання до глобальної змінної
    let b=3;            ← опис локальної змінної
    alert(b);     ← виведення значення локальної змінної (3)
}
func();      ← виклик функції
alert(b);     ← ПОМИЛКА !!! Змінної b не існує у даній області видимості
```

Слід також пам’ятати, що у функції можна перевизначити глобальну змінну (описати локальну з таким же іменем, як і глобальна). У цьому випадку значення глобальної змінної буде недоступним і без попереднього присвоєння локальній змінній значення цієї локальної змінної використовувати не можна.

Приклад.

```
let a=5;                ← глобальна змінна
function func(){
    alert(a);     ← Помилка!!! Звертання до локальної змінної,
                    яку ще не ініціалізовано
    let a=3;            ← опис локальної змінної, яка перевизначає глобальну
}
```

Слід відмітити, що не важливо, у якому саме місці функції міститься ініціалізація локальної змінної. Можливо ініціалізація знаходиться у фрагменті функції, який ніколи не буде виконано. Тут головним є фактор наявності ініціалізації новим значенням.

Значення за замовчуванням

При описі функції можна задавати так звані «значення за замовчуванням». Це значення, які використовуються у випадку, коли під час виклику не було вказано значення відповідного фактичного параметру. Формальні параметри зі значенням за замовчуванням обов'язково повинні бути в кінці списку формальних параметрів після параметрів без значень за замовчуванням (параметрів, які обов'язково потрібно вказати).

Загальний вигляд	function <ім'я> (<обов'язкові форм. парам.>,
	     <парам.1> = <знач.1>,
	     <парам.2> = <знач.2>, . . .){
	  <тіло функції (оператори)>
	}

Приклад	function func(a, **b=0, c=0**){
	  return a+b+2*c;
	}

Приклад виклику	alert (func(2)) ← Результат 2 (a=2, **b=0, c=0**) alert (func(2,7)) ← Результат 9 (a=2, **b=7, c=0**) alert (func(2,7,4)) ← Результат 17 (a=2, **b=7, c=4**)

Зауважимо, що значення параметрів можна вказувати і з використанням іменованого присвоєння формальним параметрам фактичних. Для цього під час виклику функції необхідно явно вказати формальний параметр, якому присвоюється значення.

Загальний вигляд	function <ім'я>(<форм.парам.1> = <факт.парам.1>, <форм.парам.2> = <факт.парам.2>, ...): <тіло функції (оператори)>

Приклад	//----- Опис функції ----
	function func(a, **b=0, c=0**){
	  return a+b+2*c;
	}
	//------ Виклик функції ----
	alert (func(2,c=4)) ← a=2, b=0, c=4
	alert (func(c=4,a=2)) ← a=2, b=0, c=4
	alert (func(b=8,a=2)) ← a=2, b=8, c=0

Передача довільної кількості параметрів

У JavaScript також є можливість описати функції з довільною кількістю параметрів (під час виклику можна передавати довільну кількість фактичних параметрів). Для цього необхідно описати формальний параметр, перед яким стоїть знак «…».

Загальний вигляд	function <ім'я>(. . . інші формальні параметри,
	…<формальний_параметр_кортеж>){
	<тіло функції (оператори)>
	}

Приклад.

```
//------ Опис функції з довільної кількістю параметрів -----
function get_sum(...args){
    let sum=0;
    for (let x of args)
        sum+=x;
    return sum;
}
//------ Виклик функції з довільної кількістю параметрів -----
alert (get_sum(2));                ← Результат 2
alert (get_sum(2,7));              ← Результат 9
alert (get_sum(2,3,67,5));         ← Результат 77
alert (get_sum(11,4,22,6,45,76)); ← Результат 164
```

Зауважимо, що після опису такого формального параметра не можуть бути описані інші формальні параметри.

ЛЯМБДА ВИРАЗИ

За допомогою лямбда виразів маємо можливість робити скорочені описи функцій. Особливо це є корисним при передачі функції як аргументу (наприклад при сортуванні списків).

Загальний вигляд	(<список аргументів>) => <результат функції>
Приклад	Функція для знаходження суми двох чисел //-------- Опис функції ---------- f= (a,b)=>a+b; //-------- Виклик функції ---------- alert(f(3,7)) //-------- Результат ---------- 10

Приклад 2	Упорядкувати елементи списку чисел за останньою цифрою
	//----- Використання функції як аргументу ----------
	let listV=[22,93,12,45,61];
	listV.sort(**(item1, item2)=>item1%10-item2%10)**;
	alert(listV);
	#-------- Результат ----------
	[6**1**, 2**2**, 1**2**, 9**3**, 4**5**]

РЕКУРСІЯ

Рекурсія − це такий спосіб організації обчислювального процесу, за якого функція звертається сама до себе. Такі звернення називаються *рекурсивними викликами*, а функція, що містить рекурсивні виклики, − *рекурсивною*.

Рекурсію використовують у ситуаціях, коли легко звести вихідну задачу до задачі того ж виду, але з іншими вихідними даними. Найпростішим прикладом такої задачі може стати обчислення факторіала. Так обчислення факторіала може бути здійснене у відповідності до наступного рекурентного правила:

$$n! = \begin{cases} (n-1)! \cdot n, \text{ якщо } n > 0; \\ 1, \text{ якщо } n = 0. \end{cases}$$

Реалізуємо це рекурентне правило у вигляді рекурсивної функції:

Приклад	Функція для знаходження суми двох чисел
	//-------- Опис функції ------------
	function **factorial**(n){
	  if (n==0)
	  return 1;
	  return **factorial**(n-1)*n;
	}
	//-------- Виклик функції ------------
	alert(factorial(5));
	//-------- Результат ------------
	120

Будь-яке рекурентне правило, яке визначає деяку рекурсію повинно містити умову зупинки, що називається *умовою завершення рекурсії*. Для випадку задачі обчислення факторіала *n*! умовою завершення рекурсії є умова *n*=0.

ЗАВДАННЯ ДЛЯ САМОСТІЙНОЇ РОБОТИ

Завдання 1.

1	За даними дійсними числами a і b обчислити $u = f(a,b) + f(2,a) + 2,$ де $f(x,y) = \begin{cases} x^3 + \sqrt{x^2 + y^4}, x > y, \\ \dfrac{x^2 - 2x + \sqrt{x}}{\sqrt[5]{x^3}}, \textit{інакше.} \end{cases}$
2	За даними дійсними числами a і b обчислити $u = f(a,b) + f(2,a) + 2,$ де $f(,y) = \begin{cases} x^3 + \sqrt{x^2 + y^4}, x > 0\, i\, y > 0, \\ \dfrac{x^2 - 2x + \sqrt{x}}{\sqrt[5]{x^3}}, x > 0\, i\, y < 0, \\ \sin(x * y), \textit{інакше.} \end{cases}$
3	За даними дійсними числами a і b обчислити $u = f(a) + f(2) + f(b),$ де $f(x) = \begin{cases} \sum_{i=1}^{5} x^i, x > 0, \\ \prod_{i=1}^{5} x^i, \textit{інакше.} \end{cases}$
4	За даними дійсними числами a і b обчислити $u = \min\{f(a), f(b)\},$ де $f(x) = \begin{cases} \sin x + \sin x^2 + ... + \sin x^6, x > 3, \\ -\cos x + \cos\cos x + ... + (-1)^5 \underbrace{\cos ... \cos}_{5} x, \textit{інакше.} \end{cases}$
5	За даними дійсними числами a і b обчислити $u = \max\{f(a), f(b)\},$ де $f(x) = \begin{cases} \sin x + \sin^3 x + \sin^5 x + ... + \sin^{17} x, x > 3, \\ 15 + tgx + tg\, tgx + ... + \underbrace{tg ... tg}_{5} x, \textit{інакше.} \end{cases}$
6	За даними дійсними числами a і b обчислити $u = \max\{f(a), f(b)\} - f(3),$ де $f(x) = \begin{cases} 1 + \dfrac{x}{2!} + \dfrac{x^2}{3!} + .. + \dfrac{x^8}{8!}, x < 0, \\ 15 + \sqrt{\cos^6 x}, \ 0 \le x \le 5, \\ \min\{1{,}2 * \sin(x)\}, \ x > 5. \end{cases}$

7	Дано дійсні числа x,y,z . Обчислити $u = \dfrac{\max\{x, z\} + \max\{x + y, xy\}}{\max^2\{x + y, xy\}}$.
8	Дано дійсні числа x,y,z . Обчислити $u = \dfrac{\max\{x, y, z\} + \max\{x + y, xy, 4z\}}{\max\{\max^2\{x + y, xy, x^2\}, 7, z^2\}}$.
9	Використовуючи підпрограму для знаходження коренів квадратного рівняння, знайти розв'язок наступної системи рівнянь $\begin{cases} ax^2 + 2x + 7 = 0, \\ bx^2 - 5x + 2 = 0. \end{cases}$ Числа $a, b \in R$.
10	Використовуючи підпрограму для знаходження найбільшого спільного дільника (НСД), знайти значення виразу S=(НСД(a,b)+ НСД(a,4))+ НСД(24,b)
11	Використовуючи підпрограму для знаходження коренів квадратного рівняння, знайти розв'язок наступної системи рівнянь $\begin{cases} ax^2 + 2x + 7 = 0, \\ bx^2 - 5x + 2 = 0. \end{cases}$ Числа $a, b \in R$.
12	Дано три дійсних числа: a, b, c. Використовуючи підпрограми для знаходження максимального та мінімального серед двох дійсних чисел знайти $\max(a,b) + (\max(a,b) + \min(b,c))^2$.
13	Обчислити значення виразу $f(2, a, 4) - 5 * f(a, b, -c)$, де $f(x, y, z) = \begin{cases} \lg(x + y - z), & \text{якщо } x + y - z > 10, \\ (\lvert x + y \rvert + 1)^z, & \text{якщо } z - 10 < x + y < z + 1, \\ x^2 + y^2 - z^3, & \text{якщо } x + y - z = 1, \\ \cos^2 x + \sin y - e^{2z+1}, & \text{в інших випадках.} \end{cases}$
14	Обчислити значення виразу $f(4) + 2f(a) - f(b)$, де $f(x) = \begin{cases} 0, & x < 23; \\ 1, & x = 23; \\ -1, & x > 23. \end{cases}$

Завдання 2.

1	Використовуючи підпрограму для знаходження скалярного добутку, обчислити значення виразу $s = 2\langle a,b\rangle$-$3\langle a,c\rangle$ де $a,b,c \in R^n$, $\langle x,y\rangle$ – скалярний добуток векторів.
2	Використовуючи підпрограму наближеного знаходження визначеного інтегралу за формулою лівих прямокутників, обчислити значення виразу $s = \int_a^3 \sqrt{4x + \sin\sqrt{x^3}}\,dx + \int_a^b \sqrt{4x + \sin\sqrt{x^3}}\,dx$
3	Використовуючи підпрограму наближеного знаходження визначеного інтегралу за формулою правих прямокутників, обчислити значення виразу $s = \int_a^3 \sqrt{4x + \sin\sqrt{x^3}}\,dx + \int_a^b \sqrt{4x + \sin\sqrt{x^3}}\,dx$
4	Використовуючи підпрограми для додавання векторів та множення вектора на число, знайти вектор $c = a - 3 * b + 2c$, де $a,b,c \in R^n$.
5	Використовуючи підпрограму визначення паралельності двох прямих на площині, визначити, скільки взаємно паралельних пар прямих є серед вказаних n прямих: $A_i x + B_i y + C_i = 0,\ i = 1,2,...,n$.
6	Використовуючи підпрограму визначення перпендикулярності двох прямих на площині, визначити, скільки взаємно перпендикулярних пар прямих є серед вказаних n прямих: $A_i x + B_i y + C_i = 0,\ i = 1,2,...,n$.
7	Два трикутники задано координатами вершин. Використовуючи підпрограму визначення належності точки внутрішності трикутника, з'ясувати, чи лежить один з трикутників у середині іншого.
8	Трикутник задано координатами своїх вершин на площині. Використовуючи підпрограму для знаходження кута між векторами на площині, встановити тип трикутника (гострокутний, прямокутний, тупокутний).
9	Дано послідовність натуральних чисель $a_1, a_2, ..., a_n$. Використовуючи підпрограму, яка дозволяє встановити, чи є послідовність із чотирьох чисел арифметичною прогресією, знайти кількість послідовно розміщених четвірок чисел, які утворюють арифметичну прогресію.
10	Дано послідовність натуральних чисель $a_1, a_2, ..., a_n$. Використовуючи підпрограму, яка дозволяє встановити, чи є послідовність із чотирьох чисел геометричною прогресією, знайти кількість послідовно розміщених четвірок чисел, які утворюють геометричну прогресію.
11	Використовуючи відповідні підпрограми, з'ясувати, що є більшим, середнє арифметичне чи середнє геометричне чисел $a_1, a_2, ..., a_n$.
12	Дано послідовність натуральних чисель $a_1, a_2, ..., a_n$. Використовуючи відповідні підпрограми знаходження суми та добутку цифр, знайти натуральне число, у якого найбільша сума цифр та найменший добуток цифр.
13	Дано послідовність натуральних чисель $a_1, a_2, ..., a_n$. Використовуючи підпрограму знаходження найбільшої та найменшої цифри, знайти число, у яке містить найбільшу цифру та число, яке містить найменшу цифру.
14	Використовуючи підпрограму для знаходження n-тового числа Фібоначчі. Обчислити значення виразу $s = f_3 + f_8$, де f_i – i-тове число Фібоначчі.

Завдання 3. Рекурсії

1	Використовуючи підпрограму для знаходження n-тового числа Фібоначчі. Обчислити значення виразу $s = f_3 + f_8$, де f_i – i-тове число Фібоначчі.
2	Використовуючи відповідну підпрограму знаходження g_i, обчислити значення виразу $$s = g_7 + g_9,$$ де $g_i = \sin(g_{i-1} + \cos(g_{i-2})),\ \ g_0 = 9, g_1 = 35.$
3	Нехай $x_0 = 1,\ x_i = x_{i-1} + 2i$, де $i = 1,2,\ldots$. Визначити x_{10}.
4	Нехай $x_0 = x_1 = 1,\ x_i = x_{i-1} + 2x_{i-2}$, де $i = 2,3,\ldots$. Визначити x_n.
5	Нехай $x_0 = x_1 = 1,\ x_i = x_{i-1} + x_{i-2}$. Визначити x_n.
6	Нехай $x_0 = 0, x_1 = 9,\ x_i = 2x_{i-1} + 3x_{i-2}$. Визначити x_n.
7	Нехай $x_0 = 0, x_1 = x_2 = 9,\ x_i = x_{i-1} + x_{i-2} + x_{i-3}$. Визначити x_n.
8	Нехай $x_0 = 0, x_1 = x_2 = 9,\ x_i = x_{i-1} + 4x_{i-3}$. Визначити x_n.
9	Нехай $x_0 = 0, x_1 = 7,\ x_i = x_{i-1}(1 + x_{i-2})$. Визначити x_n.
10	Нехай $x_0 = 0, x_1 = x_2 = x_3 = 7,\ x_i = \dfrac{x_{i-1}(1 + x_{i-2}) + x_{i-3}}{x_{i-4}}$. Визначити x_n.
11	Нехай $x_0 = 0, x_1 = x_2 = x_3 = 7,\ x_i = \dfrac{x_{i-1} + 4x_{i-2}(1 + x_{i-2}) + x_{i-3}}{\sqrt{x_{i-4}}} + x_{i-4}$. Визначити x_n.
12	Нехай $x_0 = x_1 = 1,\ x_i = \sin x_{i-1} + \dfrac{x_{i-2}}{\cos(x_{i-1})}$. Визначити x_n.
13	Нехай $x_0 = 2, x_1 = 1,\ x_i = \cos x_{i-1}^{\,x_{i-2}}$. Визначити x_n.
14	Нехай $x_0 = 2, x_1 = x_2 = 3,\ x_i = 7x_{i-1} - x_{i-2} * x_{i-3}$. Визначити x_n.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Сучасний підручник з JavaScript. URL: https://uk.javascript.info/

2. JavaScript. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript

3. JavaScript Підручник. Основи веб-програмування. URL: https://w3schoolsua.github.io/js/index.html#gsc.tab=0

4. Фрімен Е., Робсон Е. Книга Head First. Програмування на JavaScript. Київ: Фабула, 2022, 672 с.

ЗМІСТ

Укладачі: к. ф.-м. н., доц. Брила А.Ю.,
к. т. н., доц. Андрашко Ю.В.,
ст.викл., Ломага М.М.

Рецензенти: к.ф.-м.н., доц. Погоріляк О.О.,
к.ф.-м.н., доц. Млавець Ю.Ю.

ФУНКЦІЇ У JAVASCRIPT

Методичні вказівки до лабораторних робіт з дисципліни «WEB програмування»