Кафедра комп’ютерної та програмної інженерії

УДУ імені М.П. Драгоманова

Факультет математики, інформатики та фізики

Кафедра комп’ютерної та програмної інженерії

ЛАБОРАТОРНА РОБОТА № 5

ТЕМА:Робота з функціями.

з курсу

«Програмування в інтернеті»

Студент Яворський Нікіта

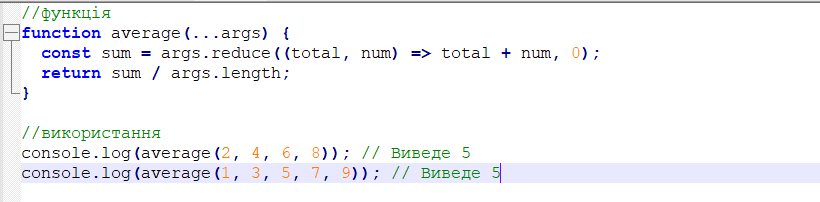
Група 31ІПЗ

Факультет МІФ

Викладач: Кархут В.Я.

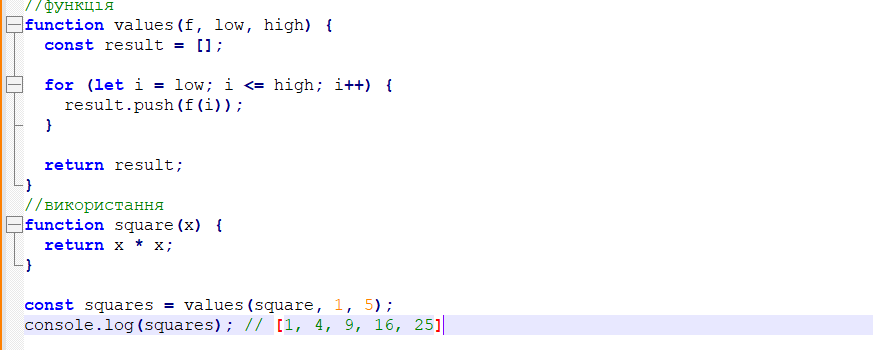
Київ 2023

1. **Створіть функцію average, яка знаходить середнє значення із довільного числа аргументів, використайте spread … оператор.**

****

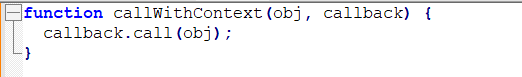
1. **Створіть функцію values(f, low, high), яка повертає масив значень [f(low), f(low + 1), ..., f(high)].**

Можна створити функцію values яка приймає функцію f, нижню межу low та верхню межу high. Використовуючи цикл for можна проітеруватись від low до high та викликати функцію f з кожним значенням та додавати результат у масив result. Після проходу по циклу масив result повертається з функції.

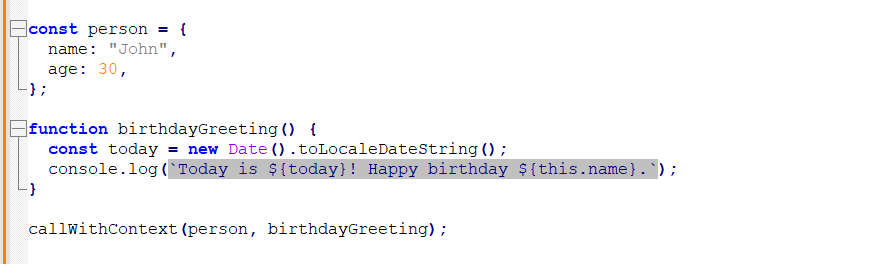


1. **Своріть функцію callWithContext як приймає обєкт та функцію коллбек яка викликається з контекстом пережаного обєкта. Викличте цю функцію з обєктом person з полями імя та вік та функцією яка виведе в консоль ‘Today is ${date }! Happy birthday ${name}.**

Можна написати функцію callWithContext наступним чином:



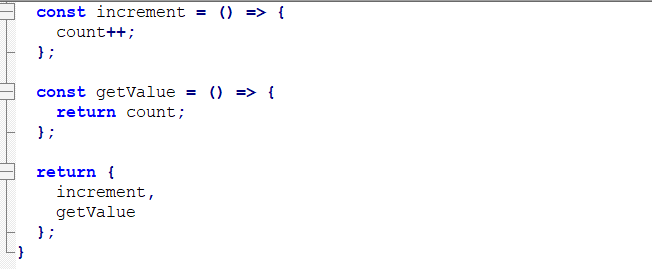
Тоді можна викликати цю функцію з об'єктом person і колбеком, який виводить привітання з його ім'ям і поточною датою, як це:



Цей код виведе щось на зразок Today is 04/10/2023! Happy birthday John. залежно від поточної дати. Функція callWithContext передає контекст person в якості this в колбек-функцію birthdayGreeting. Тому в колбек-функції ми можемо отримати доступ до поля name об'єкта person за допомогою this.name.

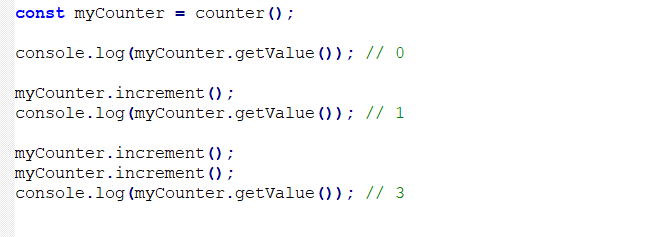
1. **Створіть функцію, яка повертає об’єкт з двома методами: increment і getValue. Метод increment має збільшувати значення, яке зберігається в замиканні, а метод getValue має повертати поточне значення.**

Можна створити функцію, що повертає об'єкт з приватними змінними (в замиканні) та публічними методами, що забезпечать інкапсуляцію даних:



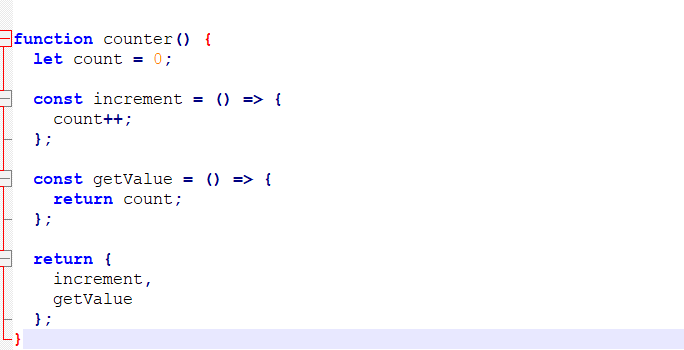
Ця функція повертає об'єкт з двома методами: increment, що збільшує значення локальної змінної count, та getValue, що повертає поточне значення count.

Тепер можна створити змінну, яка буде мати доступ до методів об'єкту, повернутого функцією counter, і використовувати ці методи для маніпулювання та отримання значення лічильника:

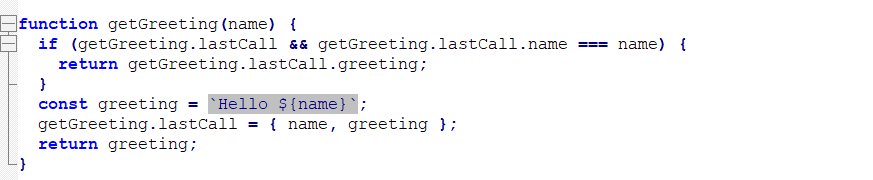


Також можна зробити лічильник приватним і повертати тільки методи increment та getValue, забезпечуючи повну інкапсуляцію даних і заборону на прямий доступ до лічильника:

javascript



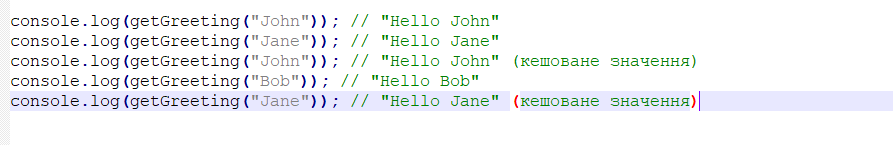
1. **Створіть функцію getGreeting яка приймає імя та повертає текстовий фрагмент типу ‘Hello name’. Зробіть щоб функція зберігала значення останнього виклику та якщо аикликана знову з таким же аргументом – повертала кешовне значення.**

****

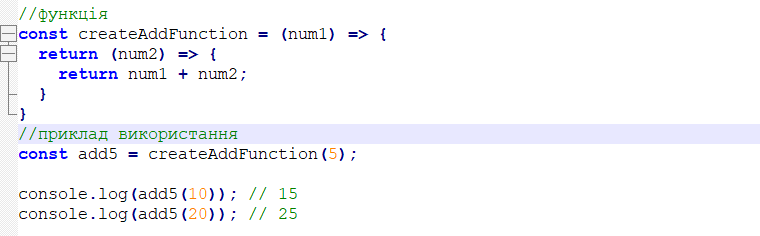
У цій функції є змінна getGreeting.lastCall, яка зберігає останнє викликання функції, щоб ми могли повернути кешоване значення, якщо функція була викликана з таким же іменем.

Якщо змінна getGreeting.lastCall містить кешоване значення для заданого імені, функція повертає кешоване значення. В іншому випадку функція створює нове вітання, зберігає його та повертає його.

Ось декілька прикладів використання функції:

****

1. **Створіть функцію, яка приймає число як аргумент і повертає функцію, яка приймає інше число як аргумент і повертає суму двох чисел. Перевірте функцію, викликавши її з різними номерами.**

****

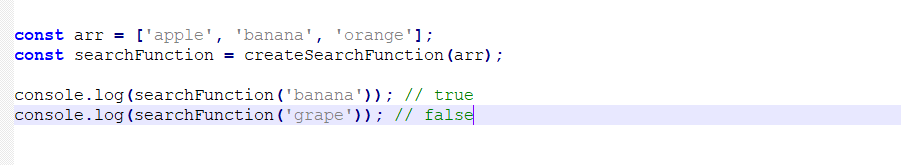
1. **Створіть функцію, яка приймає масив текстових фрагментів як аргумент і повертає нову функцію, яка приймає текстовий фрагмент як аргумент і повертає логічне значення, яке вказує, чи існує текстовий фрагмент у вихідному масиві.**

****

Ця функція приймає масив текстових фрагментів arr як параметр та повертає нову функцію. Отримана функція приймає текстовий фрагмент text як параметр та повертає true, якщо вхідний масив містить цей текст, та false, якщо ні.

У функції використовується замикання для збереження масиву arr в пам'яті та доступу до нього з внутрішньої функції.

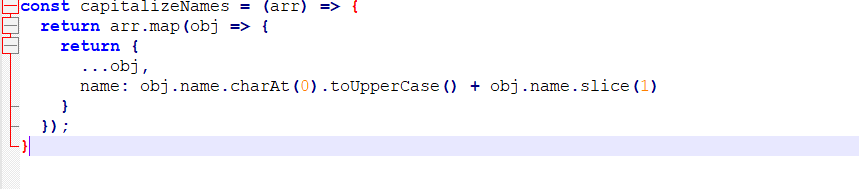
Ось приклад використання цієї функції:

****

Створено функцію searchFunction, викликавши createSearchFunction(arr) з вихідним масивом arr. Потім потрібно викликати searchFunction('banana'), що повертає true, оскільки масив містить текстовий фрагмент 'banana'. Далі викликаємо searchFunction('grape'), що повертає false, оскільки масив не містить текстовий фрагмент 'grape'.

1. **Створіть функцію, яка приймає масив об’єктів як аргумент і повертає новий масив об’єктів, де певна властивість написана з великої літери. Використовуйте стрілочну функцію.**

Ось приклад функції, яка приймає масив об'єктів та повертає новий масив об'єктів, де властивість name написана з великої літери:

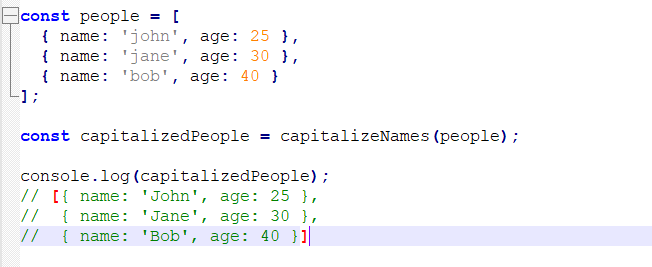
****

Ця функція використовує метод масиву map, щоб створити новий масив об'єктів, де кожен об'єкт має властивість name, яка написана з великої літери.

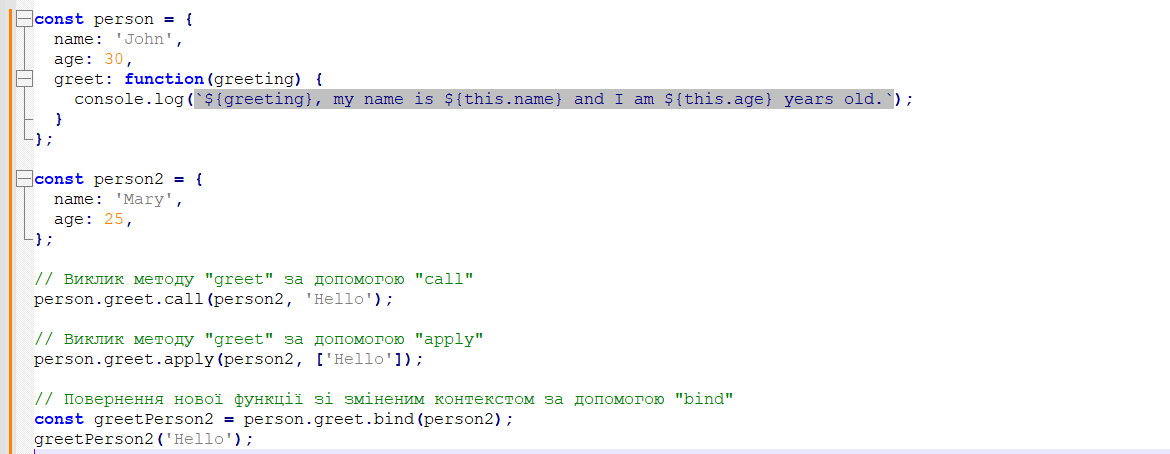
У функції використовується стрілочна функція, яка приймає масив об'єктів arr як параметр. В тілі функції використовується метод масиву map для створення нового масиву об'єктів, де кожен об'єкт має властивість name, яка написана з великої літери.

Для того, щоб змінити рядок name на велику літеру, функція використовує методи рядка charAt та slice. charAt(0) повертає перший символ рядка, а slice(1) повертає решту рядка, починаючи з другого символу. Таким чином, ми можемо збільшити перший символ на велику літеру та додати його до решти рядка, щоб отримати рядок з великої літери.

Ось приклад використання цієї функції:

****

1. **Напишіть приклад для демонстрації функцій call, apply, bind.**

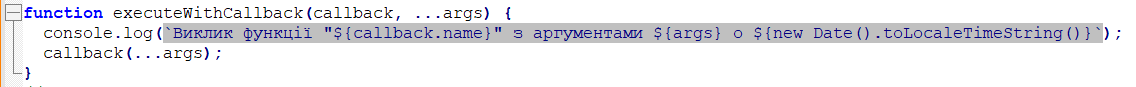
****

У цьому прикладі створено об'єкт person з властивостями name та age, а також методом greet, який виводить повідомлення з інформацією про особу. Також створено другий об'єкт person2 з властивостями name та age, але без методу greet.

Далі викликала метод greet об'єкта person за допомогою функцій call та apply та змінила контекст на об'єкт person2. Це означає, що метод буде виконуватися з властивостями об'єкта person2, але все ще використовуватиме код методу greet.

Використала функцію bind, щоб створити нову функцію greetPerson2, яка має той же код, що й метод greet, але контекст змінено на об'єкт person2. Далі потрібно було викликати цю нову функцію з аргументом 'Hello', і вона виведе те ж повідомлення, що і метод greet, але з властивостями об'єкта person2.

1. **Створіть функцію яка приймає коллбек – викликає його з переданими аргументами та виводить в консоль імя функції, аргументи та час коли функція викликана.**

****

Ця функція приймає перший параметр як колбек, а всі інші параметри передає як аргументи для виклику колбеку. Функція використовує оператор розширення ...args, щоб зібрати всі інші аргументи в масив args.

Потім функція виводить в консоль інформацію про виклик, включаючи ім'я функції, аргументи та поточний час. Нарешті, функція викликає переданий колбек з аргументами за допомогою callback(...args).

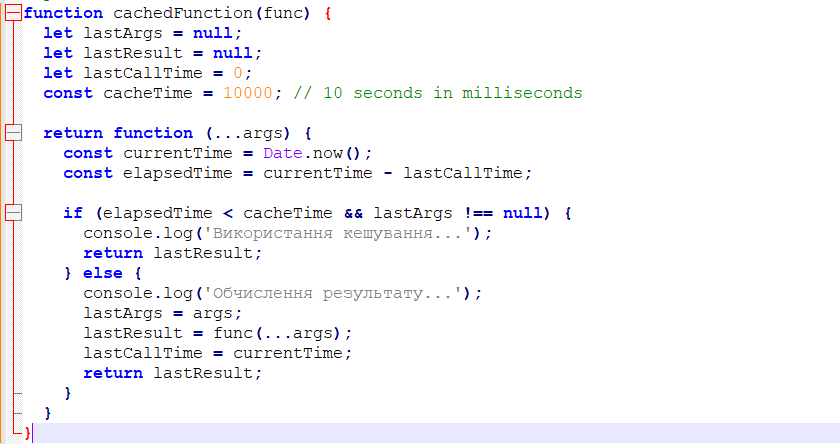
Ось приклад використання цієї функції з колбеком:

****

Цей код викличе функцію sayHello з аргументом 'John' та виведе наступне повідомлення в консоль:

**10.3.5**

1. **Створіть функцію яка кешує останній виклик на 10 секунд.**

****