**Practical task**

1. Напишіть функцію propsCount(currentObject), яка приймає об’єкт і визначає кількість властивостей цього об’єкта.

Наприклад для об’єкта

let mentor = {

course: "JS fundamental",

duration: 3,

direction: "web-development"

};

Результат має бути 3.

propsCount(mentor); // 3

2. Створіть довільний об’єкт, який має 5 полів. Необхідно написати функцію showProps(obj), яка приймає даний об’єкт і виводить список його властивостей записаних в масив. Виведіть також масив значень властивостей об’єкта.

3. Створіть клас **Person,** в якого конструктор приймає параметри *name* і *surname*, а також міститься метод showFullName(), який виводить ім’я і прізвище особи.

Від класу **Person** наслідується клас **Student**, конструктор якого крім *name* і *surname*, приймає параметр *year* (рік вступу до університету).

В класі **Student** необхідно перевизначити метод showFullName(midleName), щоб виводилося не лише ім’я, прізвище, але і по-батькові (*midleName*) студента.

Також в класі **Student** необхідно реалізувати метод showCourse(), який виводитиме поточний курс студента (від 1 до 6). Значення курсу визначатиметься як різниця поточного року (визначити самостійно) і року вступу до ВУЗу *year*.

Приклад результату:

const stud1 = new Student("Petro", "Petrenko", 2015);

console.log(stud1.showFullName("Petrovych")); // Petrenko Petro Petrovych

console.log("Current course: " + stud1.showCourse()); //Current course: 6

4. Створіть клас **Worker** який буде мати конструктор, який приймає наступні властивості: **fullName** (ім’я і прізвище), **dayRate** (ставка за день роботи), **workingDays** (кількість відпрацьованих днів).

1) клас повинен мати метод **showSalary()**, який буде виводити зарплату працівника. Зарплата - це добуток ставки **dayRate** на кількість відпрацьованих днів **workingDays**.

2) додати приватне поле experience і присвоїти йому значення 1.2 і використовувати його як додатковий множник при визначенні зарплати – створити метод showSalaryWithExperience(). Вивести значення зарплати з цим коефіцієнтом.

3) додати гетери і сетери для поля experience. Встановити значення experience = 1.5 і вивести його на екран.

4) Вивести значення зарплати з новим experience.

5) Створити кілька екземплярів класу (працівників) з різними зарплатами, як показано в прикладі нижче. Посортувати зарплату працівників із найбільшим experience по зростанню і вивести результат в форматі: worker\_fullName: salary\_value

6) Реалізувати динамічне сортування для будь-кої кількості працівників-екземплярів класу Worker**.**

Example usage:

let worker1 = new Worker("John Johnson", 20, 23);

console.log(worker1.fullName);

worker1.showSalary();

console.log("New experience: " + worker1.showExp);

worker1.showSalaryWithExperience();

worker1.setExp = 1.5;

console.log("New experience: " + worker1.showExp);

worker1.showSalaryWithExperience();

let worker2 = new Worker("Tom Tomson", 48, 22);

. . . . . .

let worker3 = new Worker("Andy Ander", 29, 23);

. . . . . .

**Output example**:

John Johnson

John Johnson salary: 460

New experience: 1.2

John Johnson salary: 552

New experience: 1.5

John Johnson salary: 690

Tom Tomson

Tom Tomson salary: 1056

. . . . . .

Andy Ander

Andy Ander salary: 667

. . . . . .

Sorted salary:

John Johnson: 690

Andy Ander: 1000.5

Tom Tomson: 1584

5. Створіть батьківський клас **GeometricFigure**, який має порожній метод для визначення площі getArea() та метод toString() для виведення назви класу.

Створіть 3 класи нащадки***Triangle***, ***Square*** і ***Circle****,* які наслідуються від класу **GeometricFigure**. Кожен з дочірніх класів має свою реалізацію методу getArea(), для визначення площі фігури. В конструкторах дочірніх класів передбачити властивості необхідні для визначення площі фігури, наприклад для кола - радіус r.

Створіть зовнішню функцію handleFigures(figures), яка прийматиме масив об’єктів дочірніх класів figures, перевірятиме чи об’єкт належить батьківському класу з урахуванням наслідування, виводитиме назву створеного об’єкту відповідної фігури, розраховану площу фігури та сумарну площу всіх фігур. Для реалізації функції можете використати метод reduce().

class GeometricFigure {

getArea() {

return 0;

}

toString() {

     return Object.getPrototypeOf(this).constructor.name;

}

     }

Your code . . .

const figures = [new Triangle(4, 5), new Square(7), new Circle(5)];  
     console.log(handleFigures(figures));

Приклад результату:

Geometric figure: Triangle - area: 10

Geometric figure: Square - area: 49

Geometric figure: Circle - area: 78.53981633974483

137.53981633974485 // total area