Inheritance (Miras)

Xatırlayırsınızsa, bizdə Person class vardı və ondan çoxlu person obyektləri qururduq. Lakin bütün obyektlərin xassələri eyni olurdu.
Misal. bir məktəb var və məktəbin direktoru, müəllimləri və şagirdləri var.
Bizə bunun üçün 3 class lazım olacaq.

```
class  Director{
    constructor(name, surname, age, location, specialty, salary){
        this.name = name
        this.surname = surname
        this.age = age
        this.location = location
        this.specialty = specialty
        this.salary = salary
    }
}
class  Teacher{
    constructor(name, surname, age, location, specialty, lessonCount, salary){
        this.name = name
        this.surname = surname
        this.age = age
        this.location = location
        this.specialty = specialty
        this.lessonCount= lessonCount
        this.salary = salary
    }
}
class  Student{
```

```
    constructor(name, surname, age, location, classNumber){
        this.name = name
        this.surname = surname
        this.age = age
        this.location = location
        this.classNumber = classNumber
    }
}
```

Diqqət etdinizsə 3 class-da eyni xassələr var. **name, surname, age, location**
Və 3 dəfə bu kodlar təkrarlanıb.
Keçən dərslərdən funksiya mövzusunda da demişdik ki, eyni kodları yazmaq yerinə bir funksiyaya yazıb, lazım olan yerdə onu çağıra bilərik. Bunu etmək həm sürət həm də kodu səliqəli göstərir.(Clean Code)

Və burada da bir ƏSAS CLASS yazıb digər class-larda o xassələri və ya metodları çağıra bilərik.
İnheritance - Bir class-ın digər class-dan **törəməsinə** deyilir. (Parent class, Child class)
Misal :

```
class  Person {
    constructor(name, surname, age, location){
        this.name = name
        this.surname = surname
        this.age = age
        this.location = location
    }
}

class  Director  extends  Person{
    constructor(name, surname, age, location, specialty, salary){
        super(name, surname, age, location)
```

```
        this.specialty = specialty
        this.salary = salary
    }
}

class  Teacher  extends  Person{
    constructor(name, surname, age, location, specialty, lessonCount, salary){
        super(name, surname, age, location)
        this.specialty = specialty
        this.specialty = lessonCount
        this.salary = salary
    }
}
class  Student  extends  Person{
    constructor(name, surname, age, location, classNumber){
        super(name, surname, age, location)
        this.classNumber = classNumber
    }
}
```

Misalda inheritance əlaqəsi quraraq həm daha sürətli həm də daha qısa kod yazmış olduq.
Bir Class digerinden miras alıbsa Class qarşısında extend açar sözündən istifadə edilir.
Constructorda gördüyünüz kimi super açar sözündən istifadə edərək Əsas Class-ın xassələrinə(property constructor ve metodlarina) çatmış olduruq.

Digər bir misala baxaq. Bu dəfə class-larda info() adlı metod yazaq.

```
class  Person  {
    constructor(name,  surname,  age,  location)  {
        this.name = name;
```

```
        this.surname = surname;
        this.age = age;
        this.location = location;
    }
    info()  {
        console.log(`--------- ${this.name} info ----------`)
        console.log(`Name: ${this.name}`);
        console.log(`Surname: ${this.surname}`);
        console.log(`Age: ${this.age}`);
        console.log(`Location: ${this.location}`);
    }
}

class  Director  extends  Person  {
    constructor(name,  surname,  age,  location,  specialty,  salary)  {
        super(name, surname, age, location);
        this.specialty = specialty;
        this.salary = salary;
    }
    info()  {
        console.log(`--------- ${this.name} info ----------`);
        console.log(`Name: ${this.name}`);
        console.log(`Surname: ${this.surname}`);
        console.log(`Age: ${this.age}`);
        console.log(`Location: ${this.location}`);
        console.log(`Specialty: ${this.specialty}`);
        console.log(`Salary: ${this.salary}`);
    }
}
```

```
class Teacher extends Person {
    constructor(name, surname, age, location, specialty, lessonCount, salary) {
        super(name, surname, age, location);
        this.specialty = specialty;
        this.specialty = lessonCount;
        this.salary = salary;
    }

    info() {
        console.log(`--------- ${this.name} info ----------`)
        console.log(`Name: ${this.name}`);
        console.log(`Surname: ${this.surname}`);
        console.log(`Age: ${this.age}`);
        console.log(`Location: ${this.location}`);
        console.log(`Specialty: ${this.specialty}`);
        console.log(`LessonCount: ${this.lessonCount}`);
        console.log(`Salary: ${this.salary}`);
    }
}

class Student extends Person {
    constructor(name, surname, age, location, classNumber) {
        super(name, surname, age, location);
        this.classNumber = classNumber;
    }
    info() {
        console.log(`--------- ${this.name} info ----------`)
        console.log(`Name: ${this.name}`);
        console.log(`Surname: ${this.surname}`);
        console.log(`Age: ${this.age}`);
```

```
        console.log(`Location: ${this.location}`);
        console.log(`Class Number: ${this.classNumber}`);
    }
}

const student1 = new Student("Zahid", "Vahabzade", 22, "Iceriseher", "Group Z");
const teacher1 = new Teacher("Rasim", "Memmedov", 34, "Razin", "Devoloper", 12, 4000);

student1.info()
teacher1.info();
```

Diqqət yetirdinizsə hər bir obyektdə info metodunda kodlar təkrarlanıb.
İndi isə gəlin obyektlərin info() metodunu super vasitəsi ilə daha qısa formada yazaq.

```
class Person {
    constructor(name, surname, age, location) {
        this.name = name;
        this.surname = surname;
        this.age = age;
        this.location = location;
    }
    info() {
        console.log(`--------- ${this.name} info -----------`)
        console.log(`Name: ${this.name}`);
        console.log(`Surname: ${this.surname}`);
        console.log(`Age: ${this.age}`);
        console.log(`Location: ${this.location}`);
    }
}

class Director extends Person {
```

```
    constructor(name, surname, age, location, specialty, salary) {
        super(name, surname, age, location);
        this.specialty = specialty;
        this.salary = salary;
    }
    info() {
        super.info();
        console.log(`Specialty: ${this.specialty}`);
        console.log(`Salary: ${this.salary}`);
    }
}



class Teacher extends Person {
    constructor(name, surname, age, location, specialty, lessonCount, salary) {
        super(name, surname, age, location);
        this.specialty = specialty;
        this.specialty = lessonCount;
        this.salary = salary;
    }
    info() {
        super.info();
        console.log(`Specialty: ${this.specialty}`);
        console.log(`LessonCount: ${this.lessonCount}`);
        console.log(`Salary: ${this.salary}`);
    }
}

class Student extends Person {
    constructor(name, surname, age, location, classNumber) {
```

```
        super(name, surname, age, location);
        this.classNumber = classNumber;
    }
    info()  {
        super.info();
        console.log(`Class Number: ${this.classNumber}`);
    }
}

const  student1  =  new  Student("Zahid",  "Vahabzade",  22,  "Iceriseher",  "Group Z");
const  teacher1  =  new  Teacher("Rasim",  "Memmedov",  34,  "Razin",  "Devoloper",  12,  4000);

student1.info()
teacher1.info();
```