Лабораторная работа **20** Перечисления

Цели и задачи

1. Научиться использовать перечисления (enum)
2. Узнать о применении перечислений

Начало работы

Убедитесь в работоспособности вашего ПК. При обнаружении неполадок, сообщите преподавателю.

Для работы вам потребуется:

1. Среда разработки **IntelliJ** (возможно так же использование другой IDE)
2. Java Development Kit 8 и выше.
3. Тетрадь и ручка для записи важных моментов

Теоретическая часть

Помните, как мы задавали переменной **gender** тип **byte**? Давайте представим ситуацию, что, например, разработчик Василий добавил этот функционал в наше приложение. Но по своей невнимательности, просто забыл правильно описать данное поле, и оставил его без комментирования. Поэтому, каждый понял, как мог. Кто-то женский пол обозначал 1, а мужской 0. Кто-то мужской 1, а женский 2. Все перепуталось. Василий запустил некую цепочку ошибок в конечном ПО. Не надо поступать как Василий. Воспользуемся механизмом перечислений.

Кроме отдельных примитивных типов данных и классов в Java есть такой тип как **enum** или перечисление. Перечисления представляют набор логически связанных констант. Объявление перечисления происходит с помощью оператора **enum**, после которого идет название перечисления. Затем идет список элементов перечисления через запятую:

```
enum Gender {
	MALE,
	FEMALE,
	NONE
}
```

Мы определили перечисление **Gender**. Теперь мы с легкостью воспользуемся данным перечислением:

```
public static void main(String[] args) {
	Gender gender = Gender.MALE;
	System.out.println(gender);    // MALE
}
```

Конечно, можем изменить наш класс **Human** (приведены не все поля):

```
class Human {
	private Gender gender;

	public Gender getGender() {
		return this.gender;
	}

	public void setGender(Gender gender) {
		this.gender = gender;
	}
}
```

Методы перечислений

Каждое перечисление имеет статический метод **`values()`**. Он возвращает массив всех констант перечисления.

Метод **`ordinal()`** возвращает порядковый номер определенной константы (нумерация начинается с 0).

Конструкторы, поля и методы перечисления

Перечисления, как и обычные классы, могут определять конструкторы, поля и методы. Например:

```
public class Main {

    public static void main(String[] args) {

        System.out.println(Color.RED.getCode());        // #FF0000
        System.out.println(Color.GREEN.getCode());      // #00FF00



    }
}
enum Color {
    RED("#FF0000"), BLUE("#0000FF"), GREEN("#00FF00");
    private String code;
    Color(String code){
        this.code = code;
    }
    public String getCode(){ return code;}
}
```

Перечисление **`Color`** определяет приватное поле **`code`** для хранения кода цвета, а с помощью метода **`getCode`** оно возвращается. Через конструктор передается для него значение. Следует отметить, что конструктор по умолчанию приватный, то есть имеет модификатор **`private`**. Любой другой модификатор будет считаться ошибкой. Поэтому создать константы перечисления с помощью конструктора мы можем только внутри перечисления.

Самостоятельная работа

Создайте класс Школа. Поместите в этот класс информацию о количестве учащихся, а также текущее время года (используя перечисление). Каждое время года должно иметь название на русском языке (используйте конструктор).

В зависимости от времени года, выведите информацию о школе следующим образом:

```
Школа №888, учащихся 666, сейчас мы учимся
// или если лето
Школа №888, учащихся 666, сейчас мы отдыхаем
```