ПЕРЕЧИСЛЕНИЯ
ENUM

Перечисления - набор логически связанных констант.

Enum (java.lang.Enum) - родительский класс всех перечисляемых типов java.

Особенности перечислений

1. У перечислений есть методы, наследуемые от java.lang.Enum
2. У перечислений есть статические методы
3. Перечисления могут определять конструкторы, поля и методы
4. Перечисления не могут наследоваться от других классов, т.к. у них уже есть родитель
5. Перечисления могут реализовывать интерфейсы

Объявление перечисления происходит с помощью оператора enum, после которого идет название перечисления. Затем идет список элементов перечисления через запятую.

```
enum Season{
    // объекты
    WINTER,
    SPRING,
    SUMMER,
    AUTUMN
}
```

Объявление переменной перечисления и обращение к элементам перечисления

```
class Application {
    public static void main(String[] args) {
        // объявили переменную типа Season,
        // присвоили значение - элемент перечисления
        Season season = Season.WINTER;

        // сравнение перечислений осуществляется через ==
        if (season == Season.AUTUMN) System.out.println("Осень");
        else if (season == Season.SPRING) System.out.println("Весна")
        else if (season == Season.SUMMER) System.out.println("Лето");
        else System.out.println("Зима");
    }
}
```

Основные методы перечислений

- Статический метод values() возвращает массив всех констант перечисления
- Статический метод valueOf(String str) возвращает элемент перечисления, значение которой равно строке, переданной в качестве аргумента
- Метод ordinal() возвращает порядковый номер элемента перечисления

Основные методы перечислений

```
class Application {
    public static void main(String[] args) {
        /* методы перечислений: values, valueOf, ordinal, name */

        for (Season value : Season.values()) {
            System.out.println(value.ordinal() + " " + value.name());
        }
        System.out.println(Season.valueOf("SPRING"));
    }
}
```

Конструкторы, поля и методы перечисления

```
enum Color{
    // объекты
    ORANGE("#FFA500", "255,165,0"), YELLOW("#FFFF00", "255,255,0"),
    PURPLE("#800080", "128,0,128"), WHITE("#FFFFFF", "255,255,255");

    // свойства
    private String hex;
    private String rgb;
    // конструктор
    Color(String hex, String rgb) {
        setHex(hex);
        setRgb(rgb);
    }
    // методы
    public String getHex() {
        return hex;
    }
    public void setHex(String hex) {
        this.hex = hex;
    }
    // остальные методы ...
}
```

Конструкторы, поля и методы перечисления

```
1  class Application {
2      public static void main(String[] args) {
3          for (Color color : Color.values()) {
4              // обращение к методам
5              System.out.println(color.getHex());
6              System.out.println(color.getRgb());
7          }
8          Color oranged = Color.ORANGE;
9
10         // обращение к методам
11         oranged.setHex("#FF4500");
12         oranged.setRgb("255,69,0");
13
14         System.out.println(oranged.getHex());
15         System.out.println(oranged.getRgb());
16     }
17 }
```

Абстрактные методы

```
enum Degree{
    // объекты
    FAHRENHEIT {
        // реализация абстрактного метода
        @Override
        public double convert(double celsius) {
            return 1.8 * celsius + 32;
        }
    },
    KELVIN {
        // реализация абстрактного метода
        @Override
        public double convert(double celsius) {
            return celsius + 273.15;
        }
    };
    // абстрактный метод
    abstract public double convert(double celsius);
}
```

Абстрактные методы

```
 1 class Application {
 2     public static void main(String[] args) {
 3 
 4         for (Degree degree : Degree.values()) {
 5             // обращение к методам
 6             System.out.println(degree);
 7         }
 8 
 9         System.out.println(Degree.KELVIN.convert(34));
10         System.out.println(Degree.FAHRENHEIT.convert(23.5));
11 
12     }
13 }
```