



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИТ)

Кафедра инструментального и прикладного программного обеспечения (ИиППО)

ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №1
по дисциплине «Программирование на языке Джава»

Выполнил студент группы ИКБО-68-24

Данильченко Д.А

Принял
Старший преподаватель

Ермаков С.Р.

Москва 2025

Задача 1.1.1

Описание задачи, цель

Напишите программу, которая конвертирует сумму денег из китайских юаней в российские рубли по курсу покупки 11.91.

Ход работы

Константа задачи - final double ROUBLES_PER_YUAN = 11.91; (курс покупки)

Получаем значение юаней через консоль:

```
Scanner reader = new Scanner(System.in);  
int yuan = reader.nextInt();
```

Вычислим сумму рублей исходя из курса (константы), а после округлим используя функцию Math.floor

```
double roubles = yuan * ROUBLES_PER_YUAN;  
roubles = Math.floor(roubles*100)/100;
```

Выведем результат в консоль

```
System.out.println("RUB: "+ roubles);
```

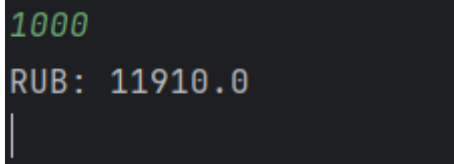
Итоговый код программы:

```
import java.util.Scanner;  
public class Main {  
    public static void main(String[] args) {  
        final double ROUBLES_PER_YUAN = 11.91;  
        Scanner reader = new Scanner(System.in);  
        int yuan = reader.nextInt();  
        reader.close();  
        double roubles = yuan * ROUBLES_PER_YUAN;  
        roubles = Math.floor(roubles*100)/100;  
        System.out.println("RUB: "+ roubles);  
    }  
}
```

Результаты и выводы

Ввод: 1000

Рисунок 1. Вывод результата консоли



```
1000
RUB: 11910.0
|
```

Разработанная программа успешно конвертирует юани в рубли по курсу 11.91. В ходе работы освоены основы Java: работа с консольным вводом/выводом и математические операции с округлением через `Math.floor`. Для 1000 юаней получен результат 11910.00 рублей.

Задача 1.1.2

Описание задачи, цель

Перепишите программу, которая конвертирует сумму денег из китайских юаней в российские рубли по курсу покупки 11.91, добавив структуру выбора для принятия решений об окончаниях входной валюты в зависимости от ее значения.

Ход работы

Добавим к коду предыдущей задачи переменную `digit` хранящую в себе значение последней цифры `yuan`

```
double digit = yuan % 10;
```

Исходя из знаний русского языка, окончания “китайский юань” меняется по правилу:

- ь - для чисел, оканчивающихся на 1 (кроме 11).
- я - для чисел, оканчивающихся на 2, 3, 4 (кроме 12, 13, 14).
- ей - для всех остальных случаев (5-20, 25-30 и т.д.).

Добавим проверки по этому правилу и выводы юаней с правильным окончанием

```
if ((10 < yuan ) && (yuan < 15)) {  
    System.out.println("CNY: " + yuan + " китайских юаней");  
} else if (digit == 1) {  
    System.out.println("CNY: " + yuan + " китайский юань");  
} else if ((1 < digit ) && (digit < 5) ) {  
    System.out.println("CNY: " + yuan + " китайских юаня");  
} else {  
    System.out.println("CNY: " + yuan + " китайских юаней");  
}
```

Итоговый код программы:

```
import java.util.Scanner;  
public class Main {  
    public static void main(String[] args) {  
        final double ROUBLES_PER_YUAN = 11.91;  
        Scanner reader = new Scanner(System.in);  
        int yuan = reader.nextInt();  
        reader.close();  
        double roubles = yuan * ROUBLES_PER_YUAN;  
        double digit = yuan % 10;  
        if ((10 < yuan ) && (yuan < 15)) {  
            System.out.println("CNY: " + yuan + " китайских юаней");  
        } else if (digit == 1) {  
            System.out.println("CNY: " + yuan + " китайский юань");  
        } else if ((1 < digit ) && (digit < 5) ) {  
            System.out.println("CNY: " + yuan + " китайских юаня");  
        }  
    }  
}
```

```

        } else {
            System.out.println("CNY: " + yuan + " китайских
юаней");
        }
        roubles = Math.floor(roubles*100)/100;
        System.out.println("RUB: "+ roubles);
    }
}

```

Результаты и выводы

Ввод: 1000

Рисунок 2. Вывод результата консоли

```

1000
CNY: 1000 китайских юаней
RUB: 11910.0

```

Ввод: 5631

Рисунок 3. Вывод результата консоли

```

5631
CNY: 5631 китайский юань
RUB: 67065.21

```

Программа успешно конвертирует юани в рубли по курсу 11.91 и теперь правильно склоняет название валюты в зависимости от количества. В ходе работы освоена работа с оператором % и конструкция if-else для проверки условий. Теперь программа определяет правильные окончания: "юань" для 1, "юаня" для 2-4, и "юаней" для остальных случаев, включая исключения.

Задача 1.2.1

Описание задачи, цель

Напишите программу, в которой создается класс Car. В данном классе должны быть обозначены следующие поля: String model, String license, String color, int year – модель автомобиля, номер автомобиля, цвет автомобиля и год выпуска соответственно. Класс должен содержать три конструктора, один конструктор, который включает в себя все поля класса, один конструктор по умолчанию, один включает поля по выбору студента.

Ход работы

Создаём файл car.java, в нём публичный класс car, объявляем поля String model, String license, String color, int year

```
public class car {  
    public String model;  
    public String license;  
    public String color;  
    public int year;
```

Создаём первый конструктор со всеми полями

```
public car(String model, String license, String color, int  
year) {  
    this.model = model;  
    this.license = license;  
    this.color = color;  
    this.year = year;  
}
```

Аналогично для конструктора с полями по желанию и по умолчанию. Для них меняем количество создаваемых полей

```
public car(String model, int year) { //Конструктор по желанию с  
выбранными полями  
    this.model = model;  
    this.year = year;  
}  
public car() { //Конструктор по умолчанию  
}
```

Итоговый код программы:

Main.java

```
import java.util.Scanner;  
public class Main {  
    public static void main(String[] args) {
```

```
}  
}
```

car.java

```
public class car {  
    public String model;  
    public String license;  
    public String color;  
    public int year;  
  
    public car(String model, String license, String color, int  
year) { //Конструктор со всеми полями  
        this.model = model;  
        this.license = license;  
        this.color = color;  
        this.year = year;  
    }  
    public car(String model, int year) { //Конструктор по  
умолчанию с выбранными полями  
        this.model = model;  
        this.year = year;  
    }  
    public car() { //Конструктор по умолчанию  
    }  
}
```

Результаты и выводы

Создан класс Car с полями: model, license, color, year. Реализовано три конструктора - полный со всеми полями, упрощенный двумя полями, и пустой (по умолчанию). Освоены навыки создания классов в Java, объявления полей разных типов данных и работа с конструкторами.

Задача 1.2.2

Описание задачи, цель

В отдельном классе Main создайте экземпляры классов (объекты), используя различные конструкторы, реализованные в задаче #1 (1.2.1). Создайте в классе метод To_String(), который будет выводить значения полей экземпляров класса. Проверьте работу созданного метода, вызвав его у объекта. Дополните класс методами для получения и установки значений для всех полей (геттерами и сеттерами). Создайте метод класса, который будет

возвращать возраст автомобиля, вычисляющийся от текущего года, значение текущего года допускается сделаться константным.

Ход работы

Добавим в Main.java экземпляры классов. car1, car2, car3

```
car car1 = new car();
car car2 = new car("CarModelTemplate", 1987);
car car3 = new car("AnotherCarModel", "b77op142", "Blue", 1966);
```

Создаем метод To_String() для класса car, выводящий значения полей объекта

```
public void To_String() {
    System.out.println( "Модель: "+this.model+ " Номер:
    "+this.license+ " Цвет: "+this.color+ " Год: "+this.year);
}
```

Создадим сеттеры для каждого из полей класса

```
public void setModel(String model) {
    this.model = model;
}
public void setLicense(String license) {
    this.license = license;
}
public void setColor(String color) {
    this.color = color;
}
public void setYear(int year) {
    this.year = year;
}
```

Создадим геттеры для каждого из полей класса

```
public String getModel() {
    return this.model;
}
public String geticense() {
    return this.license;
```



```

    }
    public String getColor() {
        return this.color;
    }
    public int getYear() {
        return this.year;
    }
}

```

Создаем метод возвращающий возраст автомобиля (для упрощения используем 2025 год)

```

    public int howOld() {
        return (2025-this.year);
    }
}

```

Добавим в Main.java исполнение созданных методов

```

        car3.To_String();
        System.out.println(car2.howOld());
        System.out.println(car1.getModel());
        car1.setModel("Audi m5-828");
        System.out.println(car1.getModel());
    }
}

```

Итоговый код программы:

Main.java

```

import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        car car1 = new car(); //Вызов конструктора по умолчанию
        car car2 = new car("CarModelTemplate", 1987); //Вызов
упрощенного конструктора
        car car3 = new car("AnotherCarModel",
        "b77op142", "Blue", 1966); //Вызов полного конструктора
        car3.To_String();
        System.out.println(car2.howOld());
        System.out.println(car1.getModel());
        car1.setModel("Audi m5-828");
        System.out.println(car1.getModel());
    }
}

```

car.java

```
public class car {
    public String model;
    public String license;
    public String color;
    public int year;

    public car(String model, String license, String color, int
year) { //Конструктор со всеми полями
        this.model = model;
        this.license = license;
        this.color = color;
        this.year = year;
    }
    public car(String model, int year) { //Конструктор по
умолчанию с выбранными полями
        this.model = model;
        this.year = year;
    }
    public car() { //Конструктор по умолчанию
    }

    //Метод вывода значений полей
    public void To_String() {
        System.out.println( "Модель: "+this.model+ " Номер:
"+this.license+ " Цвет: "+this.color+ " Год: "+this.year);
    }

    //Сеттеры
    public void setModel(String model) {
        this.model = model;
    }
    public void setLicense(String license) {
        this.license = license;
    }
    public void setColor(String color) {
        this.color = color;
    }
    public void setYear(int year) {
        this.year = year;
    }

    //Геттеры
    public String getModel() {
        return this.model;
    }
}
```

```
public String geticense() {  
    return this.license;  
}  
public String getColor() {  
    return this.color;  
}  
public int getYear() {  
    return this.year;  
}  
  
//Метод вычисления возраста автомобиля  
public int howOld() {  
    return (2025-this.year);  
}  
}
```

Результаты и выводы

Рисунок 4. Результат вызова программы

```
Модель: AnotherCarModel Номер: b77op142 Цвет: Blue Год: 1966  
38  
null  
Audi m5-828  
  
Process finished with exit code 0
```

Созданная программа успешно демонстрирует основные принципы объектно-ориентированного программирования в Java: создание классов, работу с конструкторами и методами класса, а также обращение через геттеры и сеттеры.