Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Российская академия народного хозяйства и государственной службы при Президенте Российской Федерации»

Нижегородский институт управления

Кафедра Информатики и информационных технологий

ОТЧЕТ

Задания лекции №12

Выполнила: студент группы:

ИБ-321

Кодоева Нели Тамазиевна

Нижний Новгород

2023 г.

Оглавление

[**Задание 1** 4](#_Toc132029576)

[**Текст программы (с ответами на вопросы в комментариях)** 4](#_Toc132029577)

[**Программа** 5](#_Toc132029578)

[**Реализация программы** 6](#_Toc132029579)

[**Объяснение работы программы** 6](#_Toc132029580)

[**Задание 2** 7](#_Toc132029581)

[**Пункт 1** 7](#_Toc132029582)

[**Текст программы** 7](#_Toc132029583)

[**Программа рисунка 12.3** 8](#_Toc132029584)

[**Реализация программы** 9](#_Toc132029585)

[**Объяснение работы программы** 9](#_Toc132029586)

[**Пункт 2** 10](#_Toc132029587)

[**Текст программы** 10](#_Toc132029588)

[**Программа** 11](#_Toc132029589)

[**Реализация программы** 12](#_Toc132029590)

[**Выявление ошибки** 12](#_Toc132029591)

[**Задание 3** 13](#_Toc132029592)

[**Текст программы** 13](#_Toc132029593)

[**Программа** 14](#_Toc132029594)

[**Реализация программы** 15](#_Toc132029595)

[**Особенности работы с массивами и объектами** 15](#_Toc132029596)

[**Описание программы** 15](#_Toc132029597)

[**Задание 4** 16](#_Toc132029598)

[**Текст программы** 16](#_Toc132029599)

[**Программа** 17](#_Toc132029600)

[**Реализация программы** 18](#_Toc132029601)

[**Пункт 1** 19](#_Toc132029602)

[**Программа** 19](#_Toc132029603)

[**Ответы на вопросы** 19](#_Toc132029604)

[**Пункт 2** 20](#_Toc132029605)

[**Программа** 20](#_Toc132029606)

[**Ответы на вопросы** 20](#_Toc132029607)

[**Пункт 3** 21](#_Toc132029608)

[**Программа** 21](#_Toc132029609)

[**Ответы на вопросы** 21](#_Toc132029610)

[**Задание 5** 22](#_Toc132029611)

[**Текст программы** 22](#_Toc132029612)

[**Программа** 23](#_Toc132029613)

[**Реализация программы** 24](#_Toc132029614)

[**Описание программы** 24](#_Toc132029615)

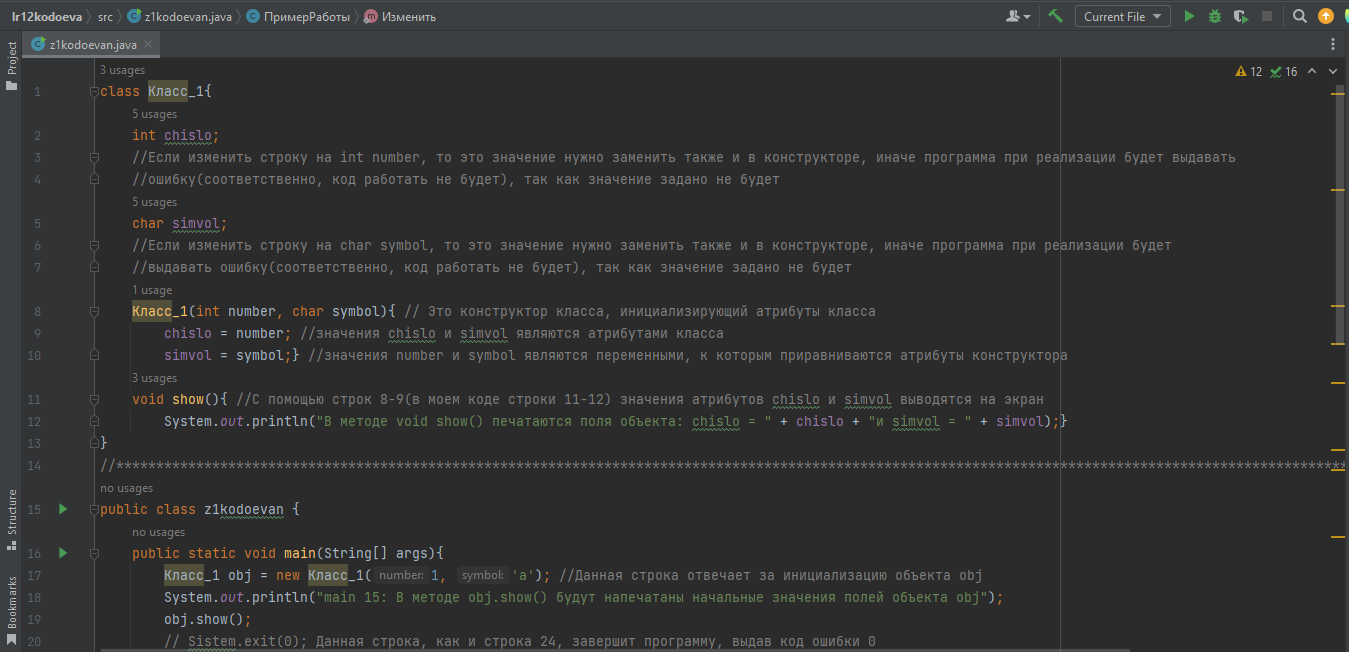
# **Задание 1**

**Реализовать код рисунка 12.2, ответить на имеющиеся в нём вопросы и объяснить работу кода**

## **Текст программы (с ответами на вопросы в комментариях)**

class Класс\_1{  
 int chislo;  
 //Если изменить строку на int number, то это значение нужно заменить также и в конструкторе, иначе программа при реализации будет выдавать  
 //ошибку(соответственно, код работать не будет), так как значение задано не будет  
 char simvol;  
 //Если изменить строку на char symbol, то это значение нужно заменить также и в конструкторе, иначе программа при реализации будет  
 //выдавать ошибку(соответственно, код работать не будет), так как значение задано не будет  
 Класс\_1(int number, char symbol){ // Это конструктор класса, инициализирующий атрибуты класса  
 chislo = number; //значения chislo и simvol являются атрибутами класса  
 simvol = symbol;} //значения number и symbol являются переменными, к которым приравниваются атрибуты конструктора  
 void show(){ //С помощью строк 8-9(в моем коде строки 11-12) значения атрибутов chislo и simvol выводятся на экран  
 System.*out*.println("В методе void show() печатаются поля объекта: chislo = " + chislo + "и simvol = " + simvol);}  
}  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
public class z1kodoevan {  
 public static void main(String[] args){  
 Класс\_1 obj = new Класс\_1(1, 'a'); //Данная строка отвечает за инициализацию объекта obj  
 System.*out*.println("main 15: В методе obj.show() будут напечатаны начальные значения полей объекта obj");  
 obj.show();  
 // Sistem.exit(0); Данная строка, как и строка 24, завершит программу, выдав код ошибки 0  
 ПримерРаботы.*Изменить*(obj); //В метод obj.show() передаётся объект obj как аргумент, в ходе чего атрибут изменится  
 System.*out*.println("main 19: В методе obj.show() будут напечатаны поля объекта obj после работы метода Изменить(obj)");  
 obj.show();  
 // Sistem.exit(0);  
 ПримерРаботы.*Изменить*(obj.chislo,obj.simvol);//В метод obj.show() атрибуты объекта передаются как аргументы  
 System.*out*.println("main 23: В методе obj.show() будут напечатаны поля объекта obj после работы метода Изменить(obj.chislo,obj.simvol)");  
 obj.show();  
 }  
}  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
class ПримерРаботы{  
 static void Изменить(Класс\_1 obj) {//Отличие строк 28 и 34(в моем коде 32 и 38) заключается в том, что в строке 28(32) метод Изменить() принимает  
 //экземпляр класса, а в строке 34(38) метод Изменить() принимает значения переменных number и symbol  
 obj.chislo++;//Данная строка увеличивает атрибут chislo на 1 с помощью ++  
 obj.simvol++;  
 System.*out*.println("В методе Изменить(Класс\_1 obj) печатаются Измененные поля объекта: поле chislo = " + obj.chislo + "и поле simvol = " +obj.simvol);  
 }  
 static void Изменить(int number, char symbol) {  
 number++;  
 symbol++;  
 System.*out*.println("39: В методе Изменить(int number, char symbol) печатаются не поля объекта, а изменённые базовые аргументы, : number = " + number + " symbol =" +symbol);  
 }  
}

## **Программа**



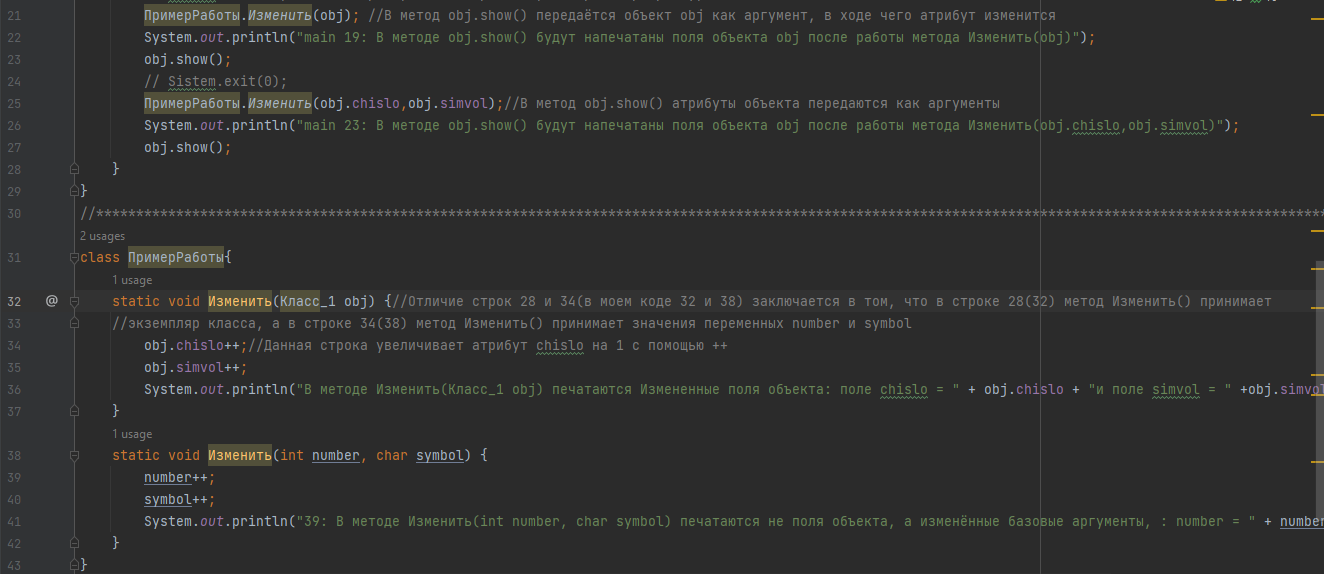


Рисунок 1-2 Программа для задания 1

## **Реализация программы**

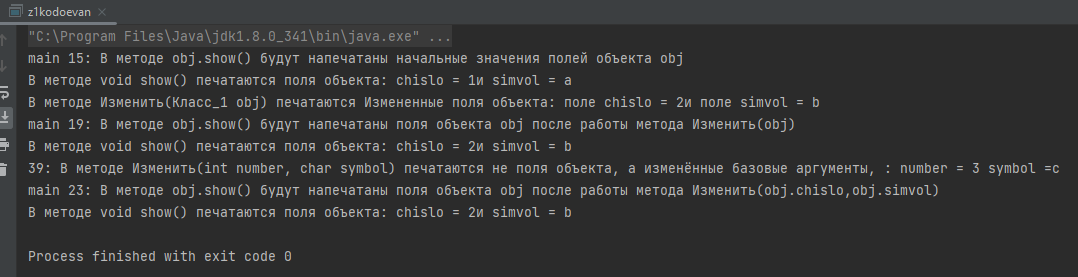


Рисунок 3 Результат программы для задания 1

## **Объяснение работы программы**

В данной программе есть 2 класса: «Класс\_1» и «ПримерРаботы».

Объекты класса «Класс\_1» имеют 2 атрибута: chislo и simvol.

С помощью класса «ПримерРаботы» изменяются атрибуты класса «Класс\_1»: исходное число увеличивается на единицу, а символ заменяется на следующий по алфавиту.

С помощью метода «ПримерРаботы.Изменить(obj)» изменяется объект полностью.

С помощью метода «ПримерРаботы.Изменить(obj.chislo,obj.simvol)» изменяются атрибуты объекта, сам объект остается неизменным.

# **Задание 2**

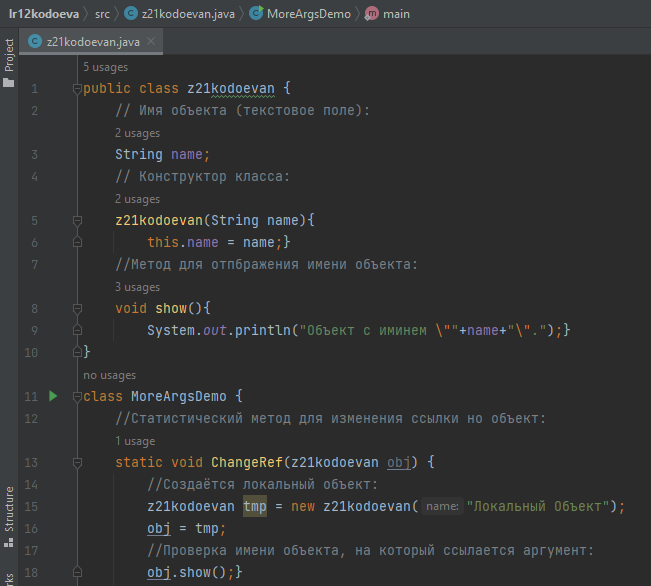
## **Пункт 1**

**Реализовать программу рисунка 12.3 и объяснить его работу**

### **Текст программы**

public class z21kodoevan {  
 // Имя объекта (текстовое поле):  
 String name;  
 // Конструктор класса:  
 z21kodoevan(String name){  
 this.name = name;}  
 //Метод для отпбражения имени объекта:  
 void show(){  
 System.*out*.println("Объект с иминем \""+name+"\".");}  
}  
class MoreArgsDemo {  
 //Статистический метод для изменения ссылки но объект:  
 static void ChangeRef(z21kodoevan obj) {  
 //Создаётся локальный объект:  
 z21kodoevan tmp = new z21kodoevan("Локальный Объект");  
 obj = tmp;  
 //Проверка имени объекта, на который ссылается аргумент:  
 obj.show();}  
 public static void main (String[] args){  
 // Создание объекта:  
 z21kodoevan obj = new z21kodoevan("Основной Объект");  
 //Поверка имени объекта:  
 obj.show();  
 //Попытка изменить ссылку на объекьт:  
 *ChangeRef*(obj);  
 //Проверка результата:  
 obj.show();  
 }  
}

### **Программа рисунка 12.3**



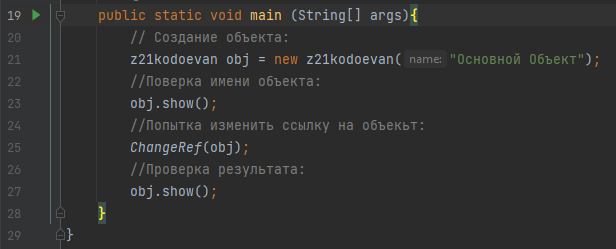


Рисунок 4-5 Программа для задания 2

### **Реализация программы**

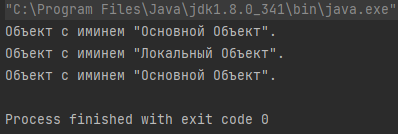


Рисунок 6 Результат программы для задания 2

### **Объяснение работы программы**

В программе присутствуют два класса: «z21kodoevan» и «MoreArgsDemo».

Класс «z21kodoevan» достаточно прост. У него всего одно текстовое поле name (отождествляем значение этого поля с именем объекта), конструктор с одним аргументом (значение, которое присваивается полю name) и метод show () для отображения значения текстового поля (имени объекта). В методе show () при выводе консольного сообщения в тексте отображаются двойные кавычки. Для отображения двойных кавычек в тексте перед ними указываем косую черту, то есть для вставки двойных кавычек в текст используем инструкцию \".

Класс «MoreArgsDemo» содержит главный метод программы и статический метод ChangeRef(), аргументом которому передается объект obj класса «z21kodoevan» (точнее, объектная переменная этого класса). В методе создается локальный объект tmp класса «z21kodoevan» (с именем "Локальный Объект") и ссылка на него присваивается аргументу метода (команда obj=tmp). После этого командой obj.show() проверяем, на какой объект ссылается переменная obj. В главном методе программы создаем объект obj с передачей конструктору класса «z21kodoevan» текстового аргумента "Основной Объект". Проверку имени объекта выполняем с помощью команды obj.show(). Затем пытаемся изменить значение объектной переменной, вызвав команду ChangeRef(obj) (предвосхищая результат, отметим, что попытка будет неудачной). Затем снова проверяем имя объекта, на который ссылается переменная obj, для чего используем команду obj.show()

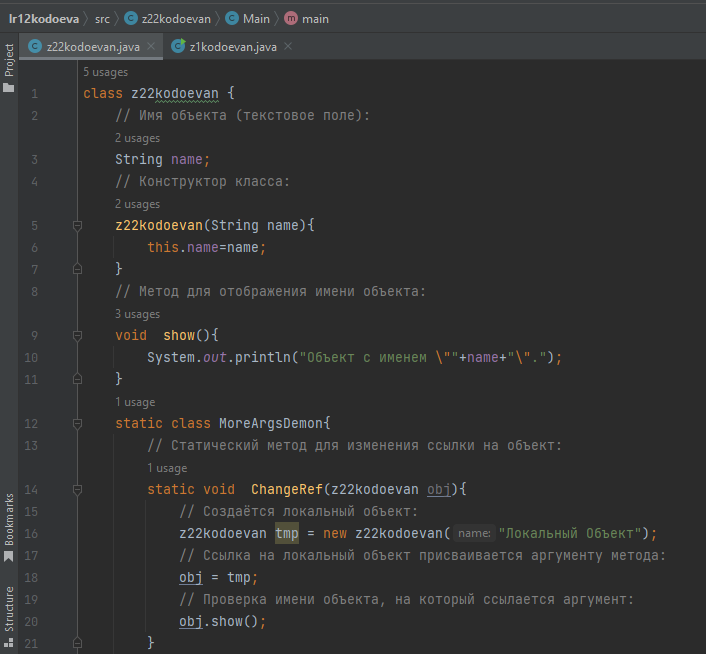
## **Пункт 2**

**Сравнить код рисунка 12.3 с кодом рисунка 12.4 и найти ошибку, и исправить её.**

### **Текст программы**

class z22kodoevan {  
 // Имя объекта (текстовое поле):  
 String name;  
 // Конструктор класса:  
 z22kodoevan(String name){  
 this.name=name;  
 }  
 // Метод для отображения имени объекта:  
 void show(){  
 System.*out*.println("Объект с именем \""+name+"\".");  
 }  
 static class MoreArgsDemon{  
 // Статический метод для изменения ссылки на объект:  
 static void ChangeRef(z22kodoevan obj){  
 // Создаётся локальный объект:  
 z22kodoevan tmp = new z22kodoevan("Локальный Объект");  
 // Ссылка на локальный объект присваивается аргументу метода:  
 obj = tmp;  
 // Проверка имени объекта, на который ссылается аргумент:  
 obj.show();  
 }  
 }  
 public class Main{  
 public void main(String[] args){  
 // Создание объекта:  
 z22kodoevan obj=new z22kodoevan("Основной Объект");  
 // Проверка имени объекта:  
 obj.show();  
 // Попытка изменить ссылку на объект:  
 MoreArgsDemon.*ChangeRef*(obj);  
 // Проверка результата:  
 obj.show();  
 }  
 }  
}

### **Программа**



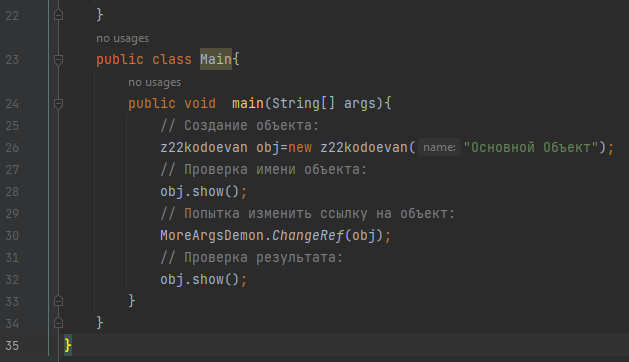


Рисунок 7-8 Программа для задания 2(рисунок 12.4)

### **Реализация программы**

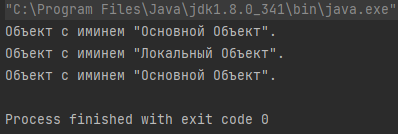


Рисунок 9 Результат программы

### **Выявление ошибки**

В строке public class MoreArgsDemon public заменяем на static (класс должен быть статическим), в противном случае программа работать не будет.

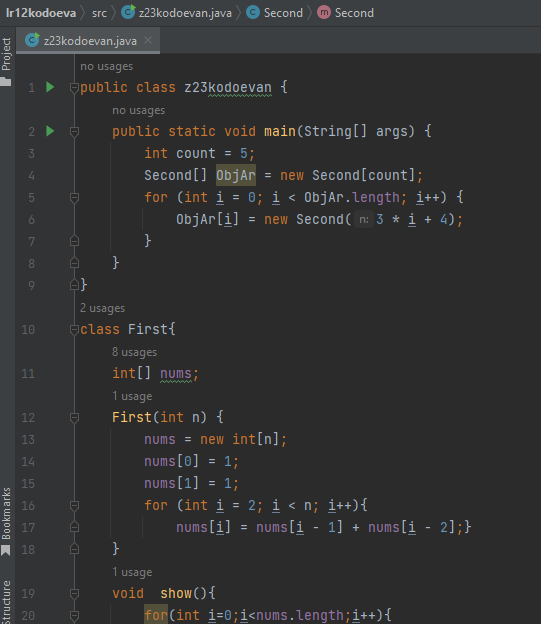
# **Задание 3**

**Разобрать работу программы на рисунке 12.5, определив особенности работы с массивами и объектами.**

## **Текст программы**

public class z23kodoevan {  
 public static void main(String[] args) {  
 int count = 5;  
 Second[] ObjAr = new Second[count];  
 for (int i = 0; i < ObjAr.length; i++) {  
 ObjAr[i] = new Second(3 \* i + 4);  
 }  
 }  
}  
class First{  
 int[] nums;  
 First(int n) {  
 nums = new int[n];  
 nums[0] = 1;  
 nums[1] = 1;  
 for (int i = 2; i < n; i++){  
 nums[i] = nums[i - 1] + nums[i - 2];}  
 }  
 void show(){  
 for(int i=0;i<nums.length;i++){  
 System.*out*.print(nums[i]+" ");}  
 System.*out*.println();}  
}  
class Second{  
 First obj;  
 Second(int n){  
 obj=new First(n);  
 obj.show();}  
}

## **Программа**



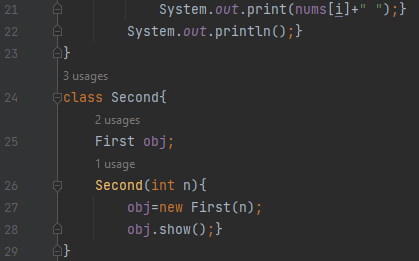


Рисунок 10-11 Программа для задания 3

## **Реализация программы**

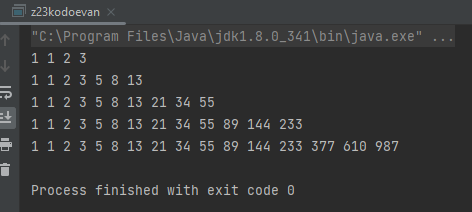


Рисунок 12 Результат программы

## **Особенности работы с массивами и объектами**

Особенности работы с массивами и объектами следуют одним и тем же правилам, так как и массивы, и объекты являются ссылочными сущностями.

## **Описание программы**

В программе присутствуют три класса: «z23kodoevan», «First» и «Second».

В классе «First» создается конструктор с массивом. Он заполняется числами Фибоначчи. С помощью метода void show() элементы полученного массива выводятся на экран.

Класс «Second» является массивом, заполненным объектами массива класса «First».

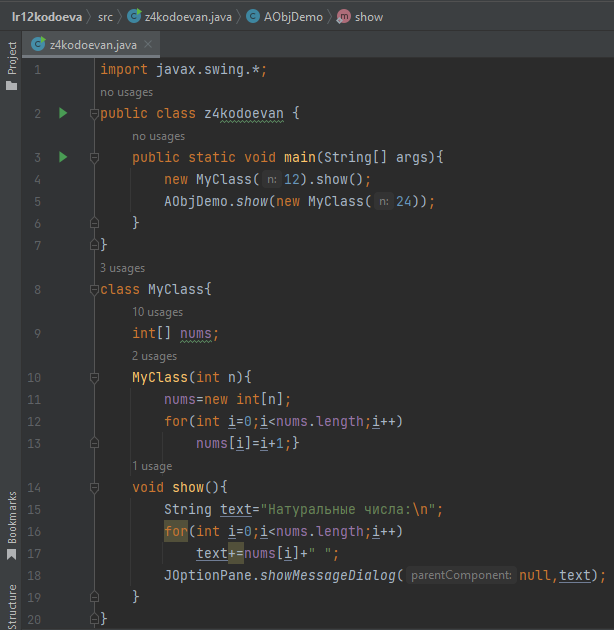
В классе «z23kodoevan» содержится экземпляр класса «Second», который заполняется объектами класса «First» и выводится на экран.

# **Задание 4**

## **Текст программы**

import javax.swing.\*;  
public class z4kodoevan {  
 public static void main(String[] args){  
 new MyClass(12).show();  
 AObjDemo.*show*(new MyClass(24));  
 }  
}  
class MyClass{  
 int[] nums;  
 MyClass(int n){  
 nums=new int[n];  
 for(int i=0;i<nums.length;i++)  
 nums[i]=i+1;}  
 void show(){  
 String text="Натуральные числа:\n";  
 for(int i=0;i<nums.length;i++)  
 text+=nums[i]+" ";  
 JOptionPane.*showMessageDialog*(null,text);  
 }  
}  
class AObjDemo {  
 static void show(MyClass obj) {  
 String text = "Элемента массива:\n";  
 int n = (int) Math.*ceil*(Math.*sqrt*(obj.nums.length));  
 for (int i = 0; i < obj.nums.length - 1; i++) {  
 text += obj.nums[i] + ((i + 1) % n == 0 ? ">> дальше\n" : " : ");  
 }  
 text += obj.nums[obj.nums.length - 1] + ". конец";  
 JOptionPane.*showMessageDialog*(null, text);  
 }  
}

## **Программа**



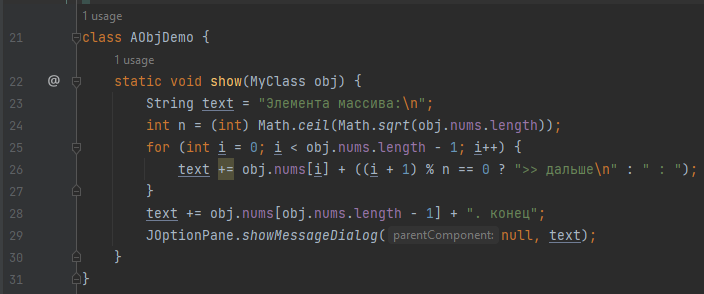
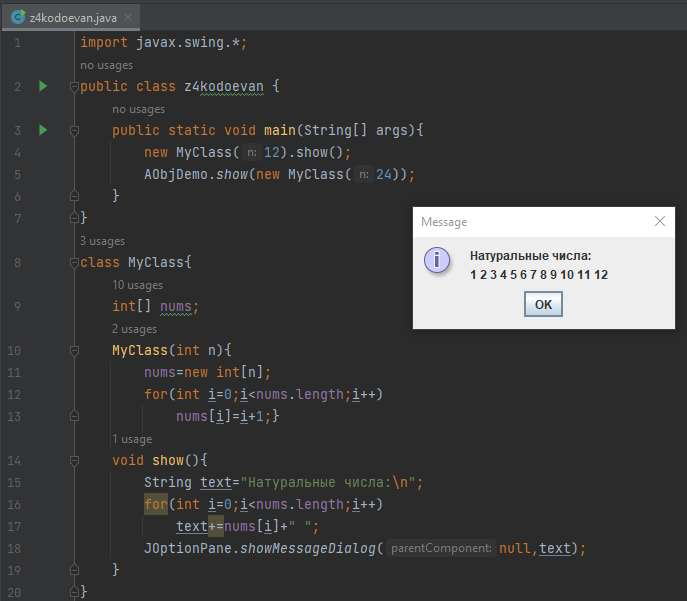


Рисунок 13-14 Программа для задания 4

## **Реализация программы**



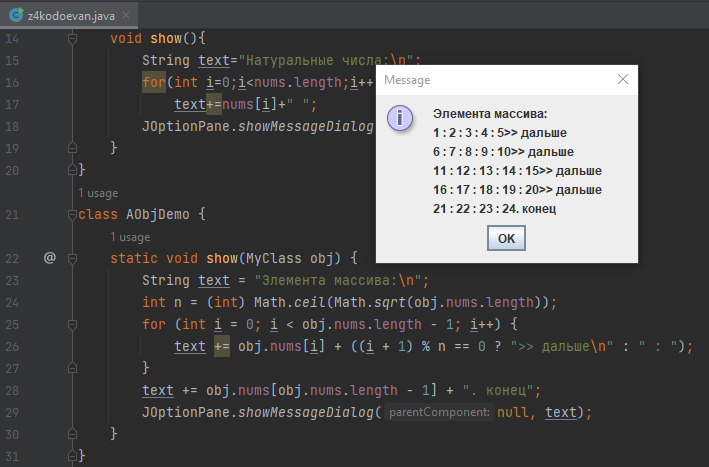


Рисунок 15-16 Результат программы

## **Пункт 1**

**Ответьте на следующие вопросы:**

**• В какой строчке создаётся анонимный объект?**

**• Какие ещё классы присутствуют в программе и что они делают?**

**• В каком виде будут выводится результаты работы всей программы?**

**• Что делает строка 4?**

**• Что делает строка 5 и что является аргументом в методе, используемым в этой строке?**

### **Программа**

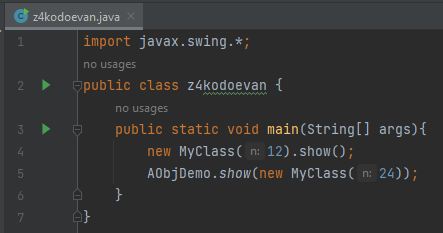


Рисунок 17 программа для задания 4.1

### **Ответы на вопросы**

1) В строке 4 мы создаем анонимный объект

2) В программе также присутствуют еще два класса: «MyClass» и «AobjDemo»

С помощью класса «MyClass» выводится массив, а с помощью класса «AobjDemo» элементы класса «MyClass»

3) Результаты работы всей программы выводятся в двух отдельных окнах (я продемонстрировала это в своем отчете на рисунках 15-16)

4) Строка 4 создает анонимный объект, также благодаря методу show() элементы класса выводятся в отдельное окно

5) В строке 5 используется метод show(), благодаря ему элементы объекта выводятся в отдельное окно.

Аргументом в методе является объект класса «MyClass»

## **Пункт 2**

**Ответьте на следующие вопросы:**

**• Зачем используется класс MyClass?**

**• Что создаётся в строке 12 кода на рисунке 12.11?**

**• Что создаётся в конструкторе класса MyClass?**

### **Программа**

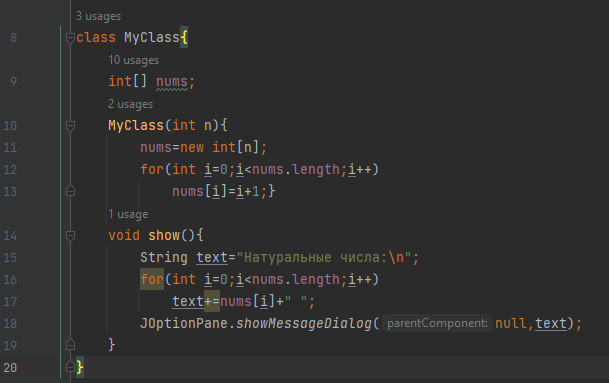


Рисунок 18 программа для задания 4.2

### **Ответы на вопросы**

1) В классе «MyClass» создается массив из натуральных чисел

2) В строке 12 происходит инициализация элемента nums массивом (в моем скриншоте это строка 11)

3) В конструкторе класса «MyClass» создается массив, которы содержит натуральные числа

## **Пункт 3**

**Ответьте на следующие вопросы:**

**• Как характеризовать метод в строке 24 рисунка 12.12? Отличается ли он от класса в строке 15 рисунка 12.11 и, если отличается, то чем?**

**• Что передаётся аргументом этого методу?**

**• Каковы особенности статического метода в Java?**

**• Как работает строка 26?**

**• Как работает строка 28?**

**• Сколько выводов будет выведено в результате работы программы в целом?**

### **Программа**

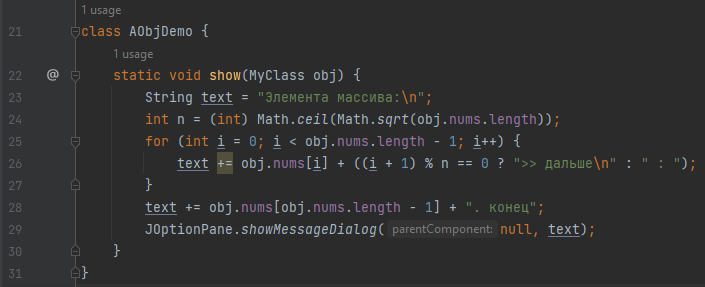


Рисунок 19 программа для задания 4.3

### **Ответы на вопросы**

1) В данной строке применяется метод show()(в моем скриншоте это строка 22). Он является статическим. Отличается он от метода в строке 15 (в моем скриншоте это строка 14) наличием дополнительных математических вычислений

2) Аргументом этому методу передается объект класса «MyClass»

3) Статический метод в Java может вызываться без создания экземпляра класса

4) Строка 26 (в моем скриншоте это строка 24) объявляет переменную n, ей присваивается метод ceil(), в котором в качестве значения возвращается ближайшее наименьшее целое число

5) В строке 28 (в моем скриншоте это строка 26) составляется строка, которая выводится на экран

6) В результате работы программы в целом будет выведено 2 окна (результат представлен в моем отчете на рисунках 15-16)

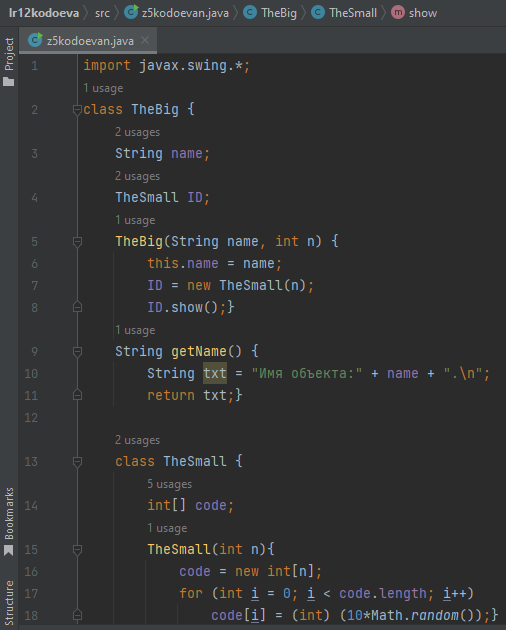
# **Задание 5**

**Разобрать работу программы на рисунке 12.6**

## **Текст программы**

import javax.swing.\*;  
class TheBig {  
 String name;  
 TheSmall ID;  
 TheBig(String name, int n) {  
 this.name = name;  
 ID = new TheSmall(n);  
 ID.show();}  
 String getName() {  
 String txt = "Имя объекта:" + name + ".\n";  
 return txt;}  
  
 class TheSmall {  
 int[] code;  
 TheSmall(int n){  
 code = new int[n];  
 for (int i = 0; i < code.length; i++)  
 code[i] = (int) (10\*Math.*random*());}  
 String getCode() {  
 String txt = "Код объекта: |";  
 //Формирование полного текста:  
 for (int i =0;i<code.length; i++)  
 txt+= code[i] +"|";  
 //Результат:  
 return txt;}  
 void show() {  
 String txt = getName() + getCode();  
 JOptionPane.*showMessageDialog*(null, txt);}  
 }  
}  
class z5kodoevan {  
 public static void main(String[] args){  
 new TheBig(" БОЛЬШОЙ",7);  
 }  
}

## **Программа**



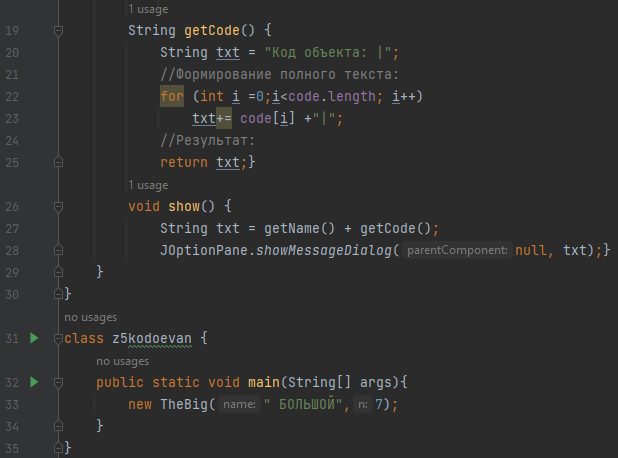


Рисунок 20-21 Программа для задания 5

## **Реализация программы**

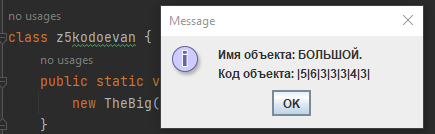


Рисунок 22 Результат программы

## **Описание программы**

В данной программе 3 класса: 2 внешних класса («z5kodoevan» и «TheBig») и внутренний класс «TheSmall».

Внутренний класс «TheSmall» создает массив и заполняет его рандомными числами. С помощью метода getCode() в качестве значения возвращается текстовая строка (тип String). В теле метода с помощью команды String txt="Код объекта: |" объявляется текстовая строка с начальным значением, к которому последовательно дописываются числа из массива code. Такж есть метод show(), который выводит методы на экран.

Внешний класс «TheBig» содержит внутренний класс «TheSmall», который описан в отчете выше. Также во внешнем классе содержится метод getName(), который возвращает имя экземпляра и конструктор, содержащий название экземпляра, массив класса TheSmall и метод show().

Внешний класс «z5kodoevan» создает анонимный объект внешнего класса «TheBig».