Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт информационных технологий и анализа данных

Центр программной инженерии

ОТЧЕТ

к лабораторной работе №4 по дисциплине:

|  |
| --- |
| «Объектно-ориентированное программирование» |
| Работа с потоками данных и обработка исключений  исключений |

наименование темы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы: | ИСТб-19-1 |  |  |  | Саранчук М.В |
|  | шифр группы |  | подпись |  | Фамилия ИО |
| Проверил: | доцент |  |  |  | Маланова Т.В. |
|  | должность |  | подпись |  | Фамилия ИО |

Иркутск. 2020 г.

Оглавление

[Оглавление 2](#_Toc59030370)

[1 Постановка задачи 3](#_Toc59030371)

[2 Иерархия наследования классов 4](#_Toc59030372)

[3 Класс Main 6](#_Toc59030373)

[Исходный код класса 6](#_Toc59030374)

[4 Класс Form 6](#_Toc59030375)

[Таблица спецификации 6](#_Toc59030376)

[Таблица спецификации методов 7](#_Toc59030377)

[Исходный код класса 7](#_Toc59030378)

[5 Класс Republic 8](#_Toc59030379)

[Таблица спецификации методов 8](#_Toc59030380)

[Исходный код класса 8](#_Toc59030381)

[6 Класс Monarchy 8](#_Toc59030382)

[Таблица спецификации 8](#_Toc59030383)

[Таблица спецификации методов 9](#_Toc59030384)

[Исходный код класса 9](#_Toc59030385)

[7 Класс Federation 9](#_Toc59030386)

[Таблица спецификации 9](#_Toc59030387)

[Таблица спецификации методов 10](#_Toc59030388)

[Исходный код класса 10](#_Toc59030389)

[8 Класс Menu 10](#_Toc59030390)

[Таблица спецификации 10](#_Toc59030391)

[Таблица спецификации методов 11](#_Toc59030392)

[Исходный код класса 11](#_Toc59030393)

[9 Класс Workwithdata 14](#_Toc59030394)

[Таблица спецификации 14](#_Toc59030395)

[Таблица спецификации методов 14](#_Toc59030396)

[Исходный код класса 14](#_Toc59030397)

[10 Таблица тестов 15](#_Toc59030398)

[10 Результаты тестирования 17](#_Toc59030399)

[Список использованных источников 20](#_Toc59030400)

1 Постановка задачи

В соответствии с индивидуальным заданием описать иерархию классов, для каждого класса описать поля и соответствующие методы доступа к ним. В зависимости от задания некоторые из этих классов (как минимум один) являются абстрактными и служат для выделения общих данных и поведения для других классов. Абстрактный класс должен содержать как минимум один абстрактный метод, реализация которого у его наследников должна различаться. Помимо этого, в общую часть задания входит разработка класса, группирующего объекты описанных в соответствии с заданием классов. Для первого варианта индивидуального задания это может быть класс «кафедра». Для выполнения задания необходимо создать некоторое количество объектов, добавить их в группу используя предусмотренные методы класса - «группы» и для каждого из них вызвать унаследованный метод.

Вариант №13

Республика, монархия, государство;

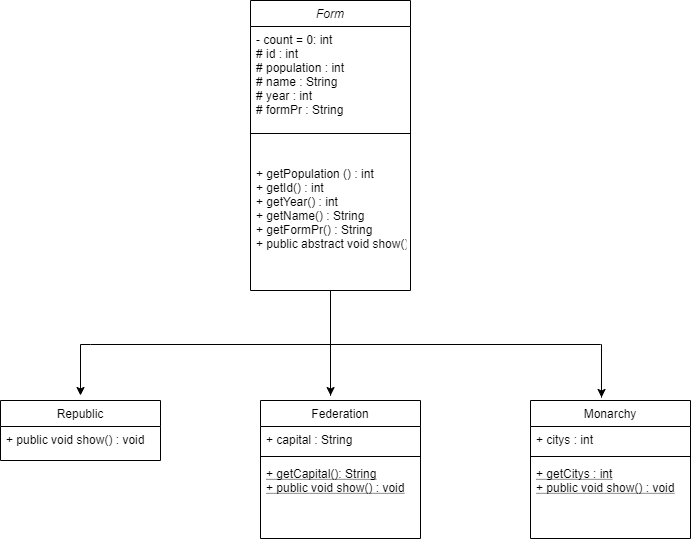
2 Иерархия наследования классов

Рисунок 1 – Схема наследования

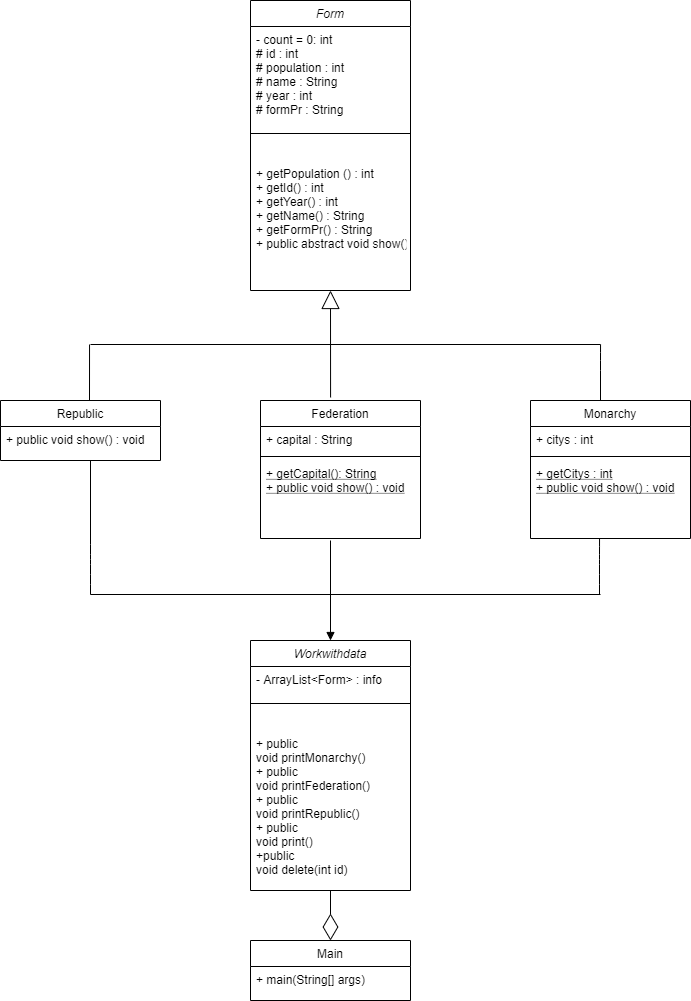


Рисунок 2 – UML схема алгоритма

3 Класс Main

Класс, содержащий метод main для входа в программу.

Исходный код класса

|  |
| --- |
| public class Main {  public static void main(String[] args) {  Menu.Menu();  }  } |

4 Класс Form

Абстрактный класс, от которого наследуют классы Republic, Monarchy, Federation

Таблица спецификации

Таблица 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя переменной | Модификатор доступа | Описание переменных | Тип данных |
| 1 | count | private | Счет кол-ва объектов | int |
| 2 | id | protected | Номер человека в базе данных | int |
| 3 | population | protected | Население (человек) | int |
| 4 | name | protected | Название страны | String |
| 5 | year | protected | Год создания страны | int |
| 6 | formPr | protected | Форма правления | String |

Таблица спецификации методов

Таблица 2

|  |  |  |  |
| --- | --- | --- | --- |
| № | Имя переменной | Описание переменных | Тип данных |
| public Form(int population, String name,int year,String formPr) | | | |
| 1 | id | Номер в базе данных | int |
| 2 | name | Название страны | String |
| 3 | population | Население (человек) | int |
| 4 | year | Год создания страны | int |
| 5 | formPr | Форма правления | String |
| public abstract void show(); | | | |
| public String getPopulation () | | | |
| public int getId() | | | |
| public int getYear() | | | |
| public int getName() | | | |
| public String getFormPr() | | | |

Исходный код класса

|  |
| --- |
| public abstract class Form {  private static int count = 0;  protected int id;  protected int population;  protected String name;  protected int year;  protected String formPr;  public Form(int population, String name,int year,String formPr ){  this.id = count +1 ;  count++;  this.name = name;  this.population = population;  this.year = year;  this.formPr= formPr;  }  public int getId(){  return id;  }  public int getPopulation(){  return population;  }  public String getName() {  return name;  }  public int getYear(){  return year;  }  public String getFormPr(){  return formPr;  }  public abstract void show();  } |

5 Класс Republic

Класс – потомок, наследующий методы и поля из класса Form.

Таблица спецификации методов

Таблица 3

|  |  |  |  |
| --- | --- | --- | --- |
| № | Имя переменной | Описание переменных | Тип данных |
| public Republic(String name, int population, int year, String formPr) | | | |
| 1 | name | Название страны | String |
| 2 | population | Население (человек) | int |
| 3 | year | Год создания страны | int |
| 4 | formPr | Форма правления | String |
| public void show () | | | |

Исходный код класса

|  |
| --- |
| public class Republic extends Form {   public Republic(String name, int population, int year, String formPr){  super(population, name, year, formPr);  }   @Override  public void show() {  StringBuilder stringBuilder = new StringBuilder("");  stringBuilder.append("Номер в базе данных: "+this.id + "\n");  stringBuilder.append("Название страны: " + this.name + "\n");  stringBuilder.append("Население (человек): " +this.population + "\n");  stringBuilder.append("Год создания страны: " +this.year + "\n");  stringBuilder.append("Форма правления: " + this.formPr + "\n");  System.*out*.println(stringBuilder.toString());  } } |

6 Класс Monarchy

Класс – потомок, наследующий методы и поля из класса Form.

Таблица спецификации

Таблица 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя переменной | Модификатор доступа | Описание переменных | Тип данных |
| 1 | citys | protected | Кол-во городов в стране | int |

Таблица спецификации методов

Таблица 5

|  |  |  |  |
| --- | --- | --- | --- |
| № | Имя переменной | Описание переменных | Тип данных |
| public Engineer(String name, int salary, int workingHourse, String function, String rank) | | | |
| 1 | name | Название страны | String |
| 2 | population | Население (человек) | int |
| 3 | year | Год создания страны | int |
| 4 | formPr | Форма правления | String |
| 5 | citys | Кол-во городов в стране | int |
| public String getCytys() | | | |
| public void show() | | | |

Исходный код класса

|  |
| --- |
| public class Monarchy extends Form {  protected int citys;  public Monarchy(String name, int population, int year, String formPr, int citys){  super(population, name, year, formPr);  this.citys = citys;  }  @Override  public void show() {  StringBuilder stringBuilder = new StringBuilder("");  stringBuilder.append("Номер в базе данных: "+this.id + "\n");  stringBuilder.append("Название страны: " + getName() + "\n");  stringBuilder.append("Население (человек): " +getPopulation() + "\n");  stringBuilder.append("Год создания страны: " +getYear() + "\n");  stringBuilder.append("Форма правления: " + getFormPr() + "\n");  stringBuilder.append("Кол-во городов в стране: "+ getCitys() + "\n");  System.out.println(stringBuilder.toString());  }  public int getCitys() {  return citys;  }  } |

7 Класс Federation

Класс – потомок, наследующий методы и поля из класса Form.

Таблица спецификации

Таблица 6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя переменной | Модификатор доступа | Описание переменных | Тип данных |
| 1 | capital | protected | Столица | int |

Таблица спецификации методов

Таблица 7

|  |  |  |  |
| --- | --- | --- | --- |
| № | Имя переменной | Описание переменных | Тип данных |
| public Federation(String name, int population, int year, String formPr, String capital) | | | |
| 1 | name | Название страны | String |
| 2 | population | Население (человек) | int |
| 3 | year | Год создания страны | int |
| 4 | formPr | Форма правления | String |
| 5 | capital | protected | Столица |
| public void show() | | | |
| public int getCapital() | | | |

Исходный код класса

|  |
| --- |
| public class Federation extends Form {  protected String capital;  public Federation(String name, int population, int year, String formPr, String capital) {  super(population, name, year, formPr);  this.capital = capital;  }  @Override  public void show() {  StringBuilder stringBuilder = new StringBuilder("");  stringBuilder.append("Номер в базе данных: " + this.id + "\n");  stringBuilder.append("Название страны: " + getName() + "\n");  stringBuilder.append("Население (человек): " + getPopulation() + "\n");  stringBuilder.append("Год создания страны: " + getPopulation() + "\n");  stringBuilder.append("Форма правления: " + getFormPr() + "\n");  stringBuilder.append("Столица: " + getCapital() + "\n");  System.out.println(stringBuilder.toString());  }  public String getCapital() {  return capital;  }  } |

8 Класс Menu

Статичный класс, реализующий функционал консольного меню.

Таблица спецификации

Таблица 8

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя переменной | Модификатор доступа | Описание переменных | Тип данных |
| 1 | workwithdata | private | Workwithdata | Database |

Таблица спецификации методов

Таблица 9

|  |  |  |  |
| --- | --- | --- | --- |
| № | Имя переменной | Описание переменных | Тип данных |
| public static void Menu() | | | |
| private static int check() | | | |
| private static int checkPopulation() | | | |
| private static int checkCyties() | | | |
| private static int borders(int min, int max) | | | |

Исходный код класса

|  |
| --- |
| public class Menu {  private static Scanner in = new Scanner(System.in);  private static Workwithdata workwithdata;  static {  workwithdata = new Workwithdata();  workwithdata.add(new Republic("Россия",146748590,1991,"Республика"));  workwithdata.add(new Republic("Беларусь",9408400,1991,"Республика"));  workwithdata.add(new Monarchy("Япония",125961625,660,"Монархия",780));  workwithdata.add(new Monarchy("ОАЭ",9992083,1971,"Монархия",7));  workwithdata.add(new Federation("Германия", 83149300,1871,"Федерация","Берлин"));  workwithdata.add(new Federation("США", 328239523,1492,"Федерация","Вашингтон"));  }  public static void Menu(){  int choice=-1;  while(choice!=0) {  System.out.println("Меню действий над базой данных");  System.out.println("1. Вывести базу данных о странах.");  System.out.println("2. Добавить страну с республиканской формой правления в базу данных.");  System.out.println("3. Добавить страну с федеративной формой правления в базу данных.");  System.out.println("4. Добавить страну с монархией, как формой правления в базу данных.");  System.out.println("5. Удалить страну из базы данных.");  System.out.println("6. Вывести страны только с монархической формой правления.");  System.out.println("7. Вывести страны только с республиканской формой правления.");  System.out.println("8. Вывести страны только с федеративной формой правления.");  System.out.println("0. Выход из базы данных.");  try {  choice = check();  } catch (Exception ex) {  System.out.println("Ошибка ввода, введите число:");  while (true) {  try {  choice = check();  break;  } catch (Exception e) {  System.out.println("Ошибка ввода, введите число:");  }  }  }  switch (choice){  case 1:  workwithdata.print();  System.out.print("---------------------------------------------"+"\n");  break;  case 2:  System.out.print("Название страны: ");  String name = in.nextLine();  System.out.print("Население (человек): ");  int population= checkPopulation();  System.out.print("Год создания: ");  int year = borders(1,2020);  String formPr = "Республика";  workwithdata.add(new Republic(name,population, year, formPr));  System.out.print("---------------------------------------------"+"\n");  break;  case 3:  System.out.print("Название страны (Федерация): ");  name = in.nextLine();  System.out.print("Население (человек): ");  population= checkPopulation();  System.out.print("Год создания: ");  year = borders(1,2020);  formPr = "Федерация";  System.out.print("Столица: ");  String capital = in.nextLine();  workwithdata.add(new Federation(name,population, year, formPr, capital));  System.out.print("---------------------------------------------"+"\n");  break;  case 4:  System.out.print("Название страны (Монархия): ");  name = in.nextLine();  System.out.print("Население (человек): ");  population = checkPopulation();  System.out.print("Год создания: ");  year = borders(1,2020);  formPr = "Монархия";  System.out.print("К-во городов: ");  int cityes = checkCyties();  workwithdata.add(new Monarchy(name,population, year, formPr, cityes));  System.out.print("---------------------------------------------"+"\n");  break;  case 5:  System.out.print ("Введите номер в базе данных: ");  int id = check();  workwithdata.delete(id);  System.out.print("---------------------------------------------"+"\n");  break;  case 6:  System.out.println("Вывод стран только с монархической формой правления.");  workwithdata.printMonarchy();  System.out.print("---------------------------------------------"+"\n");  break;  case 7:  System.out.println("Вывод стран только с республиканской формой правления.");  workwithdata.printRepublic();  System.out.print("---------------------------------------------"+"\n");  break;  case 8:  System.out.println("Вывод стран только с федеративной формой правления.");  workwithdata.printFederation();  System.out.print("---------------------------------------------"+"\n");  break;  case 0:  break;  default:  System.out.println("Такого пункта нет в меню.");  break;  }  }  }  private static int check() {  int reader = -1;  try{  reader = Integer.parseInt(in.nextLine());  }  catch (NumberFormatException ex){  System.out.print("Ошибка ввода, введите число:");  reader = check();  }  return reader;  }  private static int checkPopulation()  {  int population;  do {  population = check();  if(population < 1){  System.out.println("Население должно быть больше нуля.");  }  }while (population < 1);  return population;  }  private static int checkCyties()  {  int cityes;  do {  cityes = check();  if(cityes < 1){  System.out.println("Количество городов должно быть больше нуля.");  }  }while (cityes < 1);  return cityes;  }  private static int borders(int min, int max)  {  int value;  value= check();  if(value < min || value > max){  System.out.println("Значение не может быть меньше "+ min + " и больше "+ max);  }  while (value < min || value > max) {  value = check();  if (value < min || value > max) {  System.out.println("Значение не может быть меньше " + min + " и больше " + max);  }  }  return value;  }  } |

9 Класс Workwithdata

Класс для хранения информации о странах.

Таблица спецификации

Таблица 10

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя переменной | Модификатор доступа | Описание переменных | Тип данных |
| 1 | info | private | Лист для хранения информации о странах | Form |

Таблица спецификации методов

Таблица 11

|  |  |  |  |
| --- | --- | --- | --- |
| № | Имя переменной | Описание переменных | Тип данных |
| public void delete(int id) | | | |
| 1 | id | Номер страны в списке | int |
| public void printMonarchy() | | | |
| public void printFederation() | | | |
| public void printRepublic() | | | |
| public void print() | | | |
| public void delete(int id) | | | |

Исходный код класса

|  |
| --- |
| public class Workwithdata {  private ArrayList<Form> info = new ArrayList();  public void add(Form form){  this.info.add(form);  }  public void print(){  for (Form form : info) {  form.show();  }  }  public void printMonarchy() {  for (Form form : info) {  if(form instanceof Monarchy) {  form.show();  }  }  }  public void printFederation() {  for (Form form : info) {  if(form instanceof Federation) {  form.show();  }  }  }  public void printRepublic() {  for (Form form : info) {  if(form instanceof Republic) {  form.show();  }  }  }  public void delete(int id){  boolean bool = false;  for (Form form : info) {  if (id == form.getId()){  bool = true;  }  }  if (bool == false){  System.out.println("Такого id нет в базе данных.");  }else {  info.removeIf(form -> form.id == id);  }  }  } |

10 Таблица тестов

Таблица 12

|  |  |  |
| --- | --- | --- |
| **№** | **Действия** | **Результат** |
| 1 | Ошибка ввода в меню | Ошибка ввода, введите число: |
| 2 | Добавить страну с республиканской формой правления в базу данных | Страна добавлена |
| 3 | Добавить страну с федеративной формой правления в базу данных | Страна добавлена |
| 4 | Добавить страну с монархией, как формой правления в базу данных | Страна добавлена |
| 5 | Вывести страны только с монархической формой правления. | Будет показана информация только о странах с монархической формой правления. |
| 6 | Вывести страны только с республиканской формой правления | Будет показана информация только о странах с республиканской формой правления. |
| 7 | Вывести страны только с федеративной формой правления | Будет показана информация только о странах с федеративной формой правления. |

10 Результаты тестирования

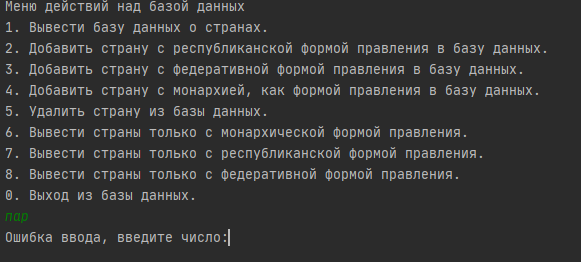


Рисунок 1 – Результат теста 1

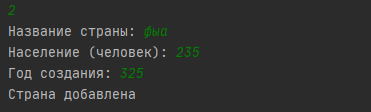


Рисунок 2 – Результат теста 2

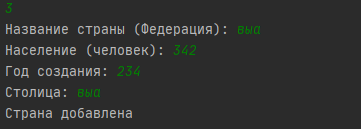


Рисунок 3 – Результат теста 3

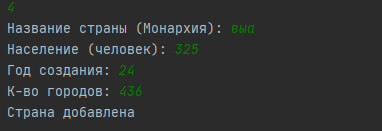
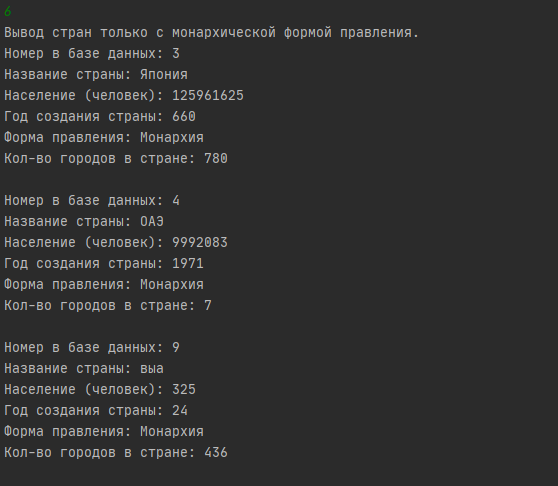
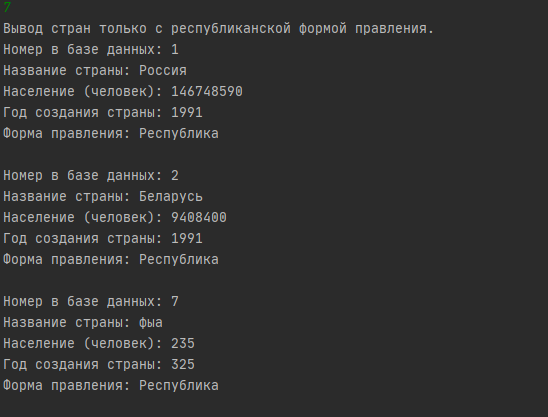


Рисунок 4 – Результат теста 4



Рисунок 5 – Результат теста 5

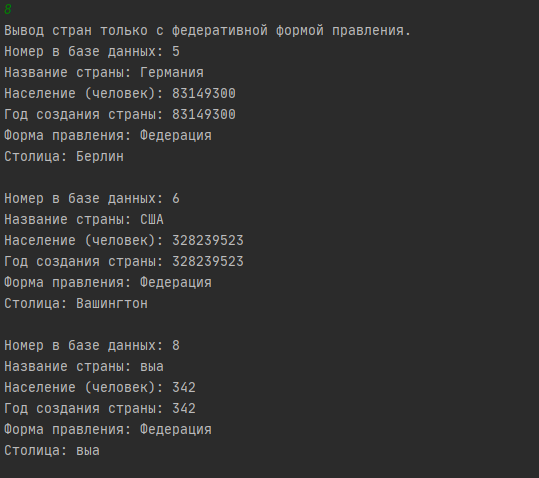
Рисунок 6 – Результат теста 6

Рисунок 7 – Результат теста 7

Список использованных источников

1. Объектно-ориентированное программирование. Метод. указания по

выполнению лабораторных работ / сост.: В.Л. Аршинский. – Иркутск : Изд-во

ИРНИТУ, 2017. – 24 c.