**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ**

**И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

(СПбГУТ)

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ (**ИТПИ**)

КАФЕДРА ПРОГРАММНОЙ ИНЖЕНЕРИИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ (ПИ И ВТ)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Лабораторная работа №3**

по дисциплине «Разработка фронтенд-приложений управления телекоммуникациями»

Тема «Моделирование экономического роста»

Вариант 24

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент гр. | ИКПИ-23 |  | 11.12.2024 | Харлова А.А. |
| Преподаватель |  |  |  | Белая Т.И. |

г. Санкт-Петербург

2024 г.

# ПОСТАНОВКА ЗАДАЧИ

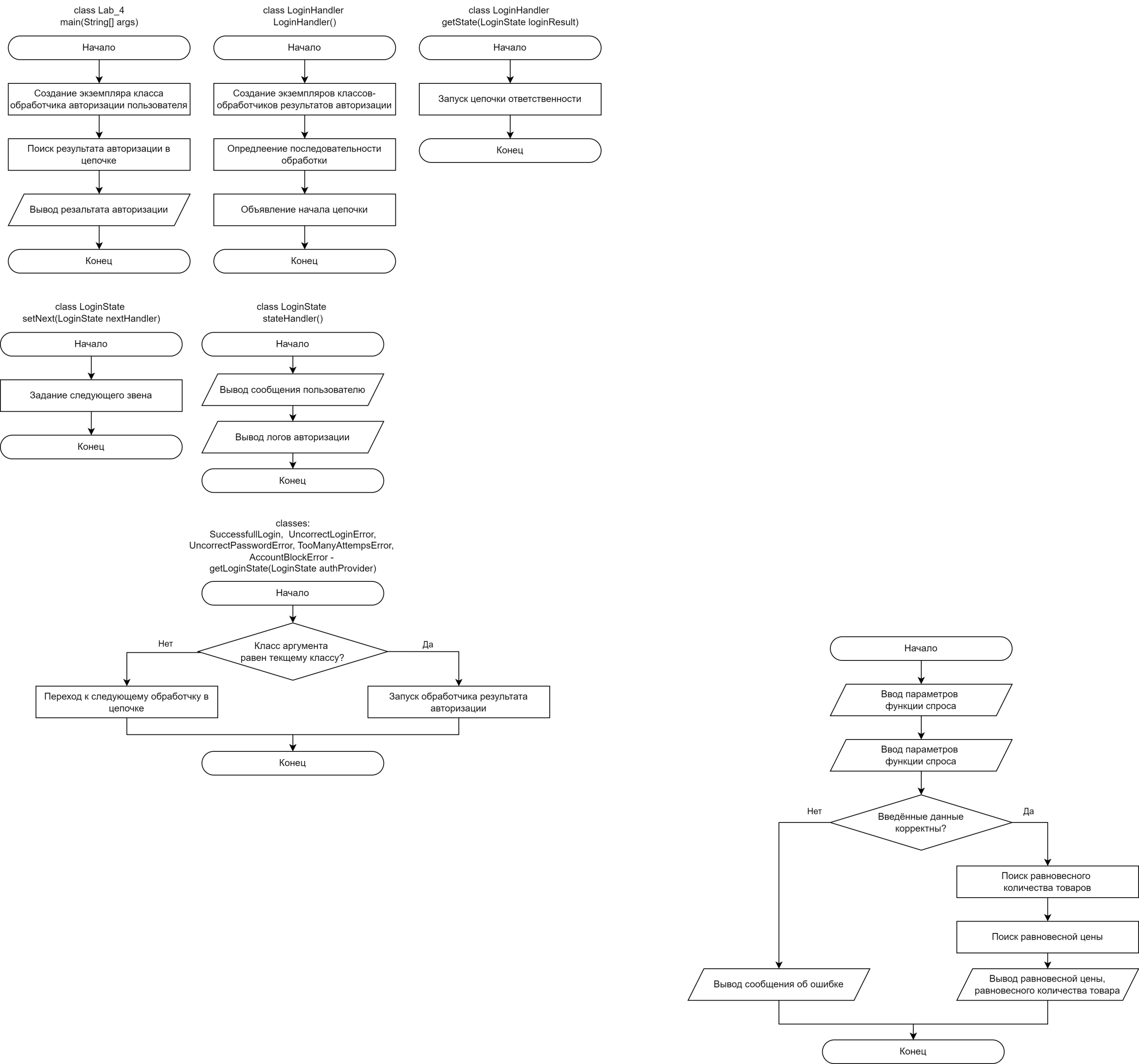
В настоящей лабораторной работе следует реализовать консольную программу, моделирующую экономический рост на основе заданной функции роста (например, модель Солоу или модель Харрода-Домара).

Необходимо:

1. Реализовать ввод начального значения ВВП и параметров функции роста.
2. Использовать методы численного моделирования для расчета и вывода ВВП в заданные моменты времени.

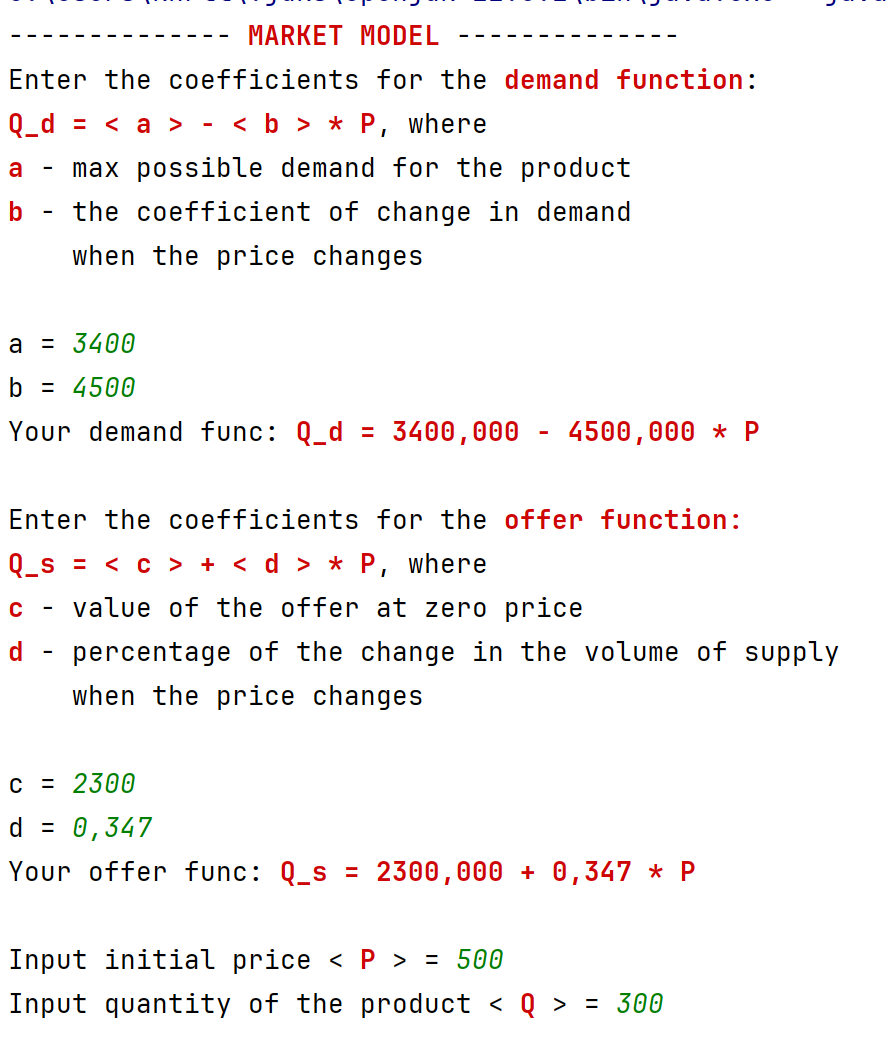
**РАЗРАБОТКА АЛГОРИТМА**

1. Создание точки входа в программу. Разработка класса "lab\_1" с функцией "main".
2. Создание класса "MarketModel", содержащего функции для работы с эконоческими данными.
   1. Написание функций ввода и вывода данных.
   2. Определение основных функций для расчёта равновесных экономических значений.

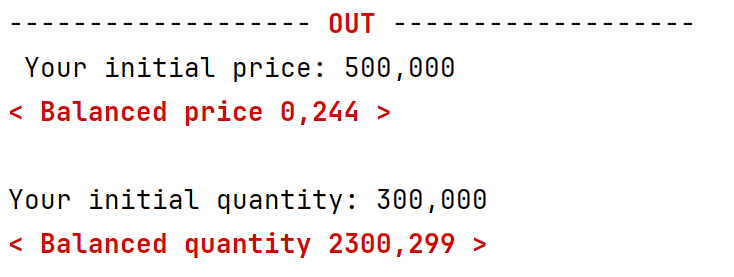


*Рисунок 1 - Алгоритм программы*

*Рисунок 2 - Алгоритм программы. Функции*



*Рисунок 3 - Пример ввода данных*



*Рисунок 4 - Пример вывода программы*

**ЛИСТИНГ ПРОГРАММЫ**

**«lab\_1.java»**

package lab\_1;  
  
public class lab\_1 {  
 public static void main(String[] args) {  
 MarketModel model = new MarketModel();  
 System.*out*.print("-------------- \033[1;31mMARKET MODEL\033[0m --------------\n");  
 if (!model.inputValues()){return;}  
 model.searchBalanceQuantity();  
 model.printValues();  
 }  
}

**«MarketModel.java»**

package lab\_1;  
  
import java.util.Scanner;  
  
public class MarketModel {  
 *// Коэфициенты функций спроса и предложения* float a, b, c, d;  
  
 *// Цены и количество товаров* float init\_price, balance\_price, balance\_quantity;  
 int init\_quantity;  
  
 *// Точность* static float *accuracy* = 0.001F;  
  
 *// Расчёт функции предложения* float clcSQuantity(float price) {  
 *// Q\_s = a - b \* P* return this.a - this.b \* price;  
 }  
  
 *// Расчёт функции для поиска равновесной цены* float clcBalanceQuantity(float price) {  
 *// f(p) = a - c + (d + b) \* P* return (this.a - this.c) - (this.d + this.b) \* price;  
 }  
  
 *// Поиск равновесного количества товара* void searchBalanceQuantity() {  
 *// Если не была найдена равновесная цена* if (this.balance\_price == 0) {  
 searchBalancePrice((this.a - this.c + this.d - this.b), -(this.a - this.c + this.d - this.b));  
 }  
 this.balance\_quantity = clcSQuantity(this.balance\_price);  
 }  
  
 *// Поиск равновесной цены* void searchBalancePrice(float a, float b) {  
 float price = (a + b) / 2;  
 *// Если выполнено условие точности -> ответ найден* if (Math.*abs*(b - a) < *accuracy*) {  
 this.balance\_price = price;  
 return;  
 }  
 *// Поиск нужного интервала по методу бисекции* if (clcBalanceQuantity(price) \* clcBalanceQuantity(b) <= 0) {  
 searchBalancePrice(price, b);  
 } else searchBalancePrice(a, price);  
 }  
  
 *// Форматированный вывод всех значений объекта* void printValues() {  
 System.*out*.print("\n------------------- \033[1;31mOUT\033[0m -------------------\n");  
 System.*out*.printf(" Your initial price: %.3f\n" +  
 "\033[1;31m< Balanced price %.3f >\033[0m\n\n" +  
 "Your initial quantity: %.3f\n" +  
 "\033[1;31m< Balanced quantity %.3f >\033[0m\n",  
 this.init\_price, this.balance\_price, this.init\_quantity, this.balance\_quantity);  
 }  
  
 *// Ввод данных и обработка ошибок* boolean inputValues() {  
 if (inputDemand()) {  
 if (inputOffer()) {  
 if (inputInitialData()) {  
 return true;  
 } else System.*out*.print("\033[1;31m[ Error: Invalid initial data ]\033[0m");  
 } else System.*out*.print("\033[1;31m[ Error: Invalid offer coefficient ]\033[0m");  
 } else System.*out*.print("\033[1;31m[ Error: Invalid demand coefficient ]\033[0m");  
 return false;  
 }  
  
 *// Ввод функции предложения* boolean inputDemand() {  
 Scanner in = new Scanner(System.*in*);  
 System.*out*.println("Enter the coefficients for the \033[1;31mdemand function\033[0m:\n" +  
 "\033[1;31mQ\_d = < a > - < b > \* P\033[0m, where\n" +  
 "\033[1;31ma\033[0m - max possible demand for the product\n" +  
 "\033[1;31mb\033[0m - the coefficient of change in demand\n" +  
 " when the price changes\n");  
 System.*out*.print("a = ");  
 this.a = in.nextFloat();  
 *// Обработка ввода некорректных данных* if (this.a <= 0) {  
 in.close();  
 return false;  
 }  
 System.*out*.print("b = ");  
 this.b = in.nextFloat();  
 System.*out*.printf("Your demand func: \033[1;31mQ\_d = %.3f - %.3f \* P\033[0m\n\n", a, b);  
 in.close();  
 return true;  
 }  
  
 *// Ввод функции спроса* boolean inputOffer() {  
 Scanner in = new Scanner(System.*in*);  
 System.*out*.println("Enter the coefficients for the \033[1;31moffer function:\033[0m\n" +  
 "\033[1;31mQ\_s = < c > + < d > \* P\033[0m, where\n" +  
 "\033[1;31mc\033[0m - value of the offer at zero price\n" +  
 "\033[1;31md\033[0m - percentage of the change in the volume of supply\n" +  
 " when the price changes\n");  
 System.*out*.print("c = ");  
 this.c = in.nextFloat();  
 *// Обработка ввода некорректных данных* if (this.c <= 0) {  
 in.close();  
 return false;  
 }  
 System.*out*.print("d = ");  
 this.d = in.nextFloat();  
 System.*out*.printf("Your offer func: \033[1;31mQ\_s = %.3f + %.3f \* P\033[0m\n\n", c, d);  
 in.close();  
 return true;  
 }  
  
 *// Ввод исходных значений цены и количества товара* boolean inputInitialData() {  
 Scanner in = new Scanner(System.*in*);  
 System.*out*.print("Input initial price < \033[1;31mP\033[0m > = ");  
 this.init\_price = in.nextFloat();  
 *// Обработка ввода некорректных данных* if (this.init\_price <= 0) {  
 in.close();  
 return false;  
 }  
 System.*out*.print("Input quantity of the product < \033[1;31mQ\033[0m > = ");  
 this.init\_quantity = in.nextInt();  
 in.close();  
 *// Обработка ввода некорректных данных* return !(this.init\_quantity <= 0);  
 }  
}