|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

**Отчет по лабораторной работе 8**

***по дисциплине «Языки программирования для работы с большими данными»***

Вариант 1

Студент ИУ6-21М **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Гришина А. М.**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Степанов П. В.**

(Подпись, дата) (И.О.Фамилия)

*2020 г.*

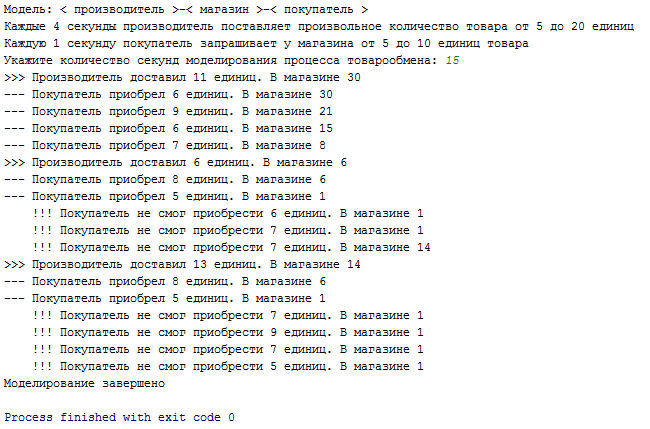
Вариант 1

3. Реализовать многопоточное приложение “Магазин”. Вся цепочка: производитель-магазин-покупатель. Пока производитель не поставит на склад продукт, покупатель не может его забрать. Реализовать приход товара от производителя в магазин случайным числом. В том случае, если товара в магазине не хватает– вывести сообщение.

**Код**:

**package** com.company;  
  
**import** java.util.Scanner;  
**import** java.util.Random;  
  
**public class** Main {  
  
 **static** Object *o* = **new** Object();  
  
 **public static void** main(String[] args) **throws** InterruptedException {  
  
 Scanner cmd = **new** Scanner(System.***in***);  
 System.***out***.print(**"Модель: < производитель >-< магазин >-< покупатель >\n"**);  
 System.***out***.print(**"Каждые 4 секунды производитель поставляет произвольное количество товара от 5 до 20 единиц\n"**);  
 System.***out***.print(**"Каждую 1 секунду покупатель запрашивает у магазина от 5 до 10 единиц товара\n"**);  
 System.***out***.print(**"Укажите количество секунд моделирования процесса товарообмена: "**);  
  
 **int** t = cmd.nextInt();  
  
 Shop sh = **new** Shop();  
 Manufacturer mn = **new** Manufacturer(t, sh);  
 Buyer br = **new** Buyer(t, sh);  
 mn.start();  
 br.start();  
 mn.join();  
 br.join();  
 System.***out***.println(**"Моделирование завершено"**);  
 }  
}  
  
**class** Shop{  
 **int Count** = 25;  
}  
  
**class** Manufacturer **extends** Thread{  
 **int Time**;  
 Shop **Sh**;  
 Manufacturer(**int** t, Shop sh){  
 **Time** = t / 4;  
 **Sh** = sh;  
 }  
 @Override  
 **public void** run() {  
 Random m = **new** Random();  
 **for**(**int** i = 0; i < **Time**; i++){  
 **int** k = m.nextInt(15) + 5;  
 **Sh**.**Count** += k;  
 System.***out***.printf(**">>> Производитель доставил %d единиц. В магазине %d\n"**, k, **Sh**.**Count**);  
 **try** {  
 Thread.*sleep*(4000);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}  
  
**class** Buyer **extends** Thread {  
 **int Time**;  
 Shop **Sh**;  
 Buyer(**int** t, Shop sh){  
 **Time** = t;  
 **Sh** = sh;  
 }  
 @Override  
 **public void** run() {  
 Random b = **new** Random();  
 **for** (**int** i = 0; i < **Time**; i++) {  
 **int** n = b.nextInt(5) + 5;  
 **if** (**Sh**.**Count** < n)  
 System.***out***.printf(**"\t!!! Покупатель не смог приобрести %d единиц. В магазине %d\n"**, n, **Sh**.**Count**);  
 **else** {  
 **Sh**.**Count** -= n;  
 System.***out***.printf(**"--- Покупатель приобрел %d единиц. В магазине %d\n"**, n, **Sh**.**Count**);  
 }  
 **try** {  
 Thread.*sleep*(1000);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}

**Проверка правильности работы:**



**Вывод:** Программа использует многопоточность для параллельного вывода в консоль. Пока производитель не поставит на склад продукт, покупатель не может его забрать. Реализован приход товара от производителя в магазин случайным числом. В том случае, если товара в магазине не хватает, то выводится сообщение. Программа работает корректно.