МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

*Лабораторная работа №3 по дисциплине «Технология программирования»*

*Вариант 2*

Группа: АВТ-610

Студенты: Князев Р.И., Дунаев Н.Ю.

Преподаватель: Васюткина И. А.

Новосибирск 2018

**Темы:**

Классы-коллекции.

**Задание:**

1. Изучить особенности реализации классов-коллекций в Java.
2. Доработать программу, созданную в лабораторной работе № 2:
3. добавить генерируемым объектам понятия «время рождения» и «время жизни». Время рождения устанавливается в момент генерации объекта и по значению соответствует времени, прошедшему от начала симуляции. Время жизни – время, через которое объект должен исчезнуть, считая от времени рождения;
4. вынести установку параметров времени жизни объектов в пользовательский интерфейс. Для каждого типа объекта должно задаваться собственное время. Рекомендуется использовать текстовые поля, но следуют помнить о проверке на ввод некорректных данных;
5. добавить генерируемым объектам уникальные целочисленные идентификаторы (случайные числа), которые назначаются при генерации объекта. Для хранения сгенерированных идентификаторов используйте коллекцию удобную для поиска по варианту;
6. использовать коллекции по варианту. При генерации объекта происходит добавление его в коллекцию (в класс добавить поле идентификатора), а во вторую коллекцию: идентификаторы существующих объектов, в третью идентификатор + время рождения. При возникновении события по таймеру обойдите коллекцию и удалите все объекты, время жизни которых истекло, а также все данные во вспомогательных коллекциях;
7. добавьте в панель управления кпопку «Текущие объекты». По нажатию на эту кнопку появляется модальное диалоговое окно, содержащее список всех «живых» объектов на момент нажатия со временем их рождения (тип объекта, время рождения, идентификатор). В класс диалогового окна должна передаваться коллекция с хранением объектов по времени рождения. Типы коллекций задаются вариантом.

***Описание структуры классов:***

Для реализации программы были изменены следующие классы:

* DialogWind
* Habbit

**DialogWind** *– класс диалогового окна* :

DialogWind(ArrayList<Bee> buf) – конструктор класса диалог винд, принимает коллекцию, хранящую время жизни пчел

**Habbit** – добавлен слушатель для открытия диалогового окна, так же реализовано добавления в коллекции необходимых данных

***Описание функционала программы:***

Добавлено диалоговое окно, которое принимает в качестве параметра конструктора коллекцию, так же реализовано добавление в коллекции

**public class** DialogWind **extends** JDialog {  
 ArrayList<Bee> **mas**;  
  
 DialogWind(ArrayList<Bee> buf)  
 {  
 **mas** = buf;  
 String str = **new** String();  
 **for**(**int** i=0;i<Bee.*n*;i++)  
 {  
 **if**(**mas**.get(i) **instanceof** BeeWorker)  
 str += **"Рабочий "**;  
 **else** str += **"Трутень "**;  
 str += **mas**.get(i).**uniqueKey** + **"\n"**;  
 }  
 JTextArea beeLab = **new** JTextArea(str);  
 beeLab.setEditable(**false**);  
 add(beeLab);  
 setLocation(100,100);  
 setSize(300,500);  
 setVisible(**true**);  
  
 }  
}

**public class** Habitat {

**private** HashSet<Integer> **keys** = **new** HashSet<>();  
**private** TreeMap<Integer, Double> **timeMap** = **new** TreeMap<>();

public Habitat(Window myWind) {

myWind.**curBee**.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 DialogWind d = **new** DialogWind(**mas**);  
 }  
});

**if** (**secForTrut** > **timeSpawnTruten**) {  
 **if** ((**float**) BeeTruten.*trNumber* / Bee.*n* < **trutenPersent**) {  
 **int** key = (**int**)(Math.*random*()\* 100000);

**while** (**keys**.contains(key))  
 key = (**int**)(Math.*random*()\* 100000);

Bee tmpBee = **new** BeeTruten((**int**)(Math.*random*() \* 720), 55 + (**int**)(Math.*random*() \* 380), key , **time**, **timeTrutLiving**);  
 **mas**.add(tmpBee);  
 **keys**.add(key);  
 **timeMap**.put(tmpBee.**uniqueKey**, tmpBee.**liveTime**);  
 }  
 **secForTrut** -= **timeSpawnTruten**;  
}  
**if** (**secForWorker** > **timeSpawnWorker**) {  
 **if** (Math.*random*() >= (1 - **workerCreateChance**)) {  
 **int** key = (**int**)(Math.*random*()\* 100000);

**while** (**keys**.contains(key))  
 key = (**int**)(Math.*random*()\* 100000);

Bee tmpBee = **new** BeeWorker((**int**) (Math.*random*() \* 720),55 + (**int**)(Math.*random*() \* 380), key , **time**, **timeWorkLiving**);  
 **mas**.add(tmpBee);  
 **keys**.add(key);  
 **timeMap**.put(tmpBee.**uniqueKey**, tmpBee.**liveTime**);  
 }  
 **secForWorker** -= **timeSpawnWorker**;  
}

}

**Вывод:**

В ходе выполнения лабораторной работы мы познакомились с особенностями классов-коллекций, изучили различные классы-коллекции, узнали какие структуры данных они позволяют реализовать, попробовали применить их на практике.