Министерство науки и высшего образования Российской Федерации

Новосибирский государственный технический университет

Лабораторная работа №5 по дисциплине  
«Технология программирования» на тему

**Потоки ввода-вывода.**

**Сериализация объектов в файл**

Факультет: АВТФ Преподаватель:

Группа: АВТ-810 Михайленко Д. А.

Студенты: Степаненко Владислав

Новосибирск

2020

#### Практические задания

1) добавить в главное меню команду «Консоль». По этой команде должно появляться немодальное диалоговое окно с многострочным текстовым полем, занимающим всю область окна. В это окно можно вводить команды по варианту. В это же окно выводится реакция программы на команду;

2) для передачи команды в обрабатывающий поток использовать каналы ввода-вывода;

3) создать конфигурационный файл для программы. В конфигурационный файл должны сохраняться все настройки симуляции, т.е. все данные и состояния, которые задаются в панели управления программы. Конфигурационный файл должен читаться при запуске программы и записываться при выходе. Формат файла текстовый;

4) добавить в главное меню пункты «Загрузить» и «Сохранить». Команда «Сохранить» вызывает сериализацию всех «живых» объектов в ней. Команда «Загрузить» останавливает текущую симуляцию (если симуляция запущена) и загружает объекты из выбранного файла. Не забудьте скорректировать время рождения объектов. После открытия симуляцию можно запустить, загруженные объекты должны вести себя естественно;

***Вариант 7***

Реализовать в консоли команды «Установить вероятность генерации капитальных домов» и «Получить вероятность генерации капитальных домов». Как параметр в команду установки должно передаваться значение вероятности. Полученная вероятность должна выводиться на консоль.

**Ход работы:**

**1)** Добавлена кнопка вызова консоли в главное меню, в консоли реализованы 2 команды:

getchance – возвращает значение вероятности появления капитальных домов

setchance x – устанавливает значение вероятности появления капитальных домов (значение х)

Консоль – окно JDialog с JTextArea внутри и слушателем на нажатие Enter

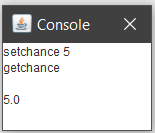


Рисунок - Окно консоли

**2)** Добавлены кнопки сохранения и загрузки объектов, текущие объекты сериализуются в файл Obj.out



**3)**Добавлены кнопки сохранения и загрузки параметров среды (также параметры сохраняются автоматически при выходе из программы, также автоматически загружаются при присутствии файла config.txt)



**Листинг программы(основные изменения):**

public void saveConfig() throws IOException {  
 FileWriter nFile = new FileWriter("config.txt");  
 nFile.write(t1 +"\n" );  
 nFile.write(t2 +"\n" );  
 nFile.write(p1 +"\n" );  
 nFile.write(p2 +"\n" );  
 nFile.write( stoneLifeTime +"\n" );  
 nFile.write( woodLifeTime +"\n" );  
 nFile.close();  
 }  
  
 public void loadConfig() throws IOException {  
 FileReader fr = new FileReader("config.txt");  
 Scanner scan = new Scanner(fr);  
  
 if (scan.hasNextLine()) {  
  
 this.t1 = Float.*parseFloat*(scan.nextLine());  
 this.t2 = Float.*parseFloat*(scan.nextLine());  
 this.p1 = Double.*parseDouble*(scan.nextLine());  
 this.p2 = Double.*parseDouble*(scan.nextLine());  
 this.stoneLifeTime = Integer.*parseInt*(scan.nextLine());  
 this.woodLifeTime = Integer.*parseInt*(scan.nextLine());  
 fr.close();  
 }  
 }  
}

public void showConsole() {  
 JDialog dialog = new JDialog();  
 dialog.setTitle("Console");  
 JTextArea console = new JTextArea();  
 dialog.add(console);  
 console.setEditable(true);  
 dialog.setSize(new Dimension(600, 400));  
 dialog.setVisible(true);  
 console.addKeyListener(new KeyAdapter() {  
 @Override  
 public void keyReleased(KeyEvent e) {  
 super.keyReleased(e);  
 if (e.getKeyCode() == e.*VK\_ENTER*) {  
 String[] input = console.getText().split("\n");  
  
 if(input[input.length-1].equals("getchance"))  
 {  
 console.append("\n" + p1 + "\n");  
 console.setCaretPosition(console.getText().length());  
 System.*out*.println(input[input.length-1].split(" "));  
 }  
 else  
 {  
 String[] newInput;  
 String com = "";double temp = 0;double temp2;  
 newInput=input[input.length-1].split(" ");  
 if (newInput.length==2)  
 {  
 if (newInput[0].equals("setchance"))  
 {  
 p1=Double.*parseDouble*(newInput[1]);  
 }  
 }  
 }  
 }  
  
 }  
 });  
  
}

public void saveObj() throws IOException {  
 FileOutputStream outFile =new FileOutputStream("Obj.out");  
 ObjectOutputStream outStream = new ObjectOutputStream(outFile);  
 outStream.writeObject(this);  
}

**Вывод:**

Были изучены особенности реализации и работы потоков ввода и вывода, ознакомился с сериализацией и десериализацией объектов.