**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Заместитель руководителя департамента  Программной Инженерии  Факультета Компьютерных Наук  профессор  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.М. Гринкруг  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** | **RU.17701729.02.03-01 51 01-1** | | **ИССЛЕДОВАНИЕ И РЕАЛИЗАЦИЯ СЕРИАЛАЙЗЕРА-ДЕСЕРИАЛАЙЗЕРА АЦИКЛИЧЕСКОГО ГРАФА ИЗ ИНСТАНСОВ JAVA BEANS COMPONENT**  **Программа и методика испытаний**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.02.03-01 51 01-1-ЛУ** | | |
|  |  | |
| Исполнитель:  студент группы БПИ171  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Данилов А.А. /  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. | |
|  | | |
|  | |  |

**2019**

|  |  |  |
| --- | --- | --- |
| **УТВЕРЖДЕНО**  **RU.17701729.02.03-01 51 01-1-ЛУ** |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** | **RU.17701729.02.03-01 51 01-1** | | **ИССЛЕДОВАНИЕ И РЕАЛИЗАЦИЯ**  **СЕРИАЛАЙЗЕРА-ДЕСЕРИАЛАЙЗЕРА**  **АЦИКЛИЧЕСКОГО ГРАФА ИЗ**  **ИНСТАНСОВ JAVA BEANS COMPONENT**  **Программа и методика испытаний**  **RU.17701729.5071102-01 51 01-1**  **Листов 27** |
|
|  |  |
|  | **2019** |

Оглавление

[Аннотация 4](#_Toc1)

[**1. Объект испытаний** 6](#_Toc2)

[**2. Цель испытаний** 6](#_Toc3)

[**3. Требования к программе** 6](#_Toc4)

[2.1.1 **Основные функции:** 6](#_Toc5)

[2.1.2 **Дополнительные функции:** 6](#_Toc6)

[**4. Требования к программной документации** 7](#_Toc7)

[**5. Средства и порядок испытаний** 7](#_Toc8)

[**5.1 Технические средства**  7](#_Toc9)

[**5.2 Программные средства** 7](#_Toc10)

[**5.3 Порядок проведения испытаний** 8](#_Toc11)

[**6. Методы испытаний** 8](#_Toc12)

[**6.1 Испытания функциональности программы** 8](#_Toc13)

[**Основные функции:** 8](#_Toc14)

[**Дополнительные функции:** 13](#_Toc15)

[ПРИЛОЖЕНИЕ 1 14](#_Toc16)

[**ПРИЛОЖЕНИЕ 2** 15](#_Toc17)

[**ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ** 15](#_Toc18)

Аннотация

Программа и методика испытаний – это документ, в котором содержится информация о программном продукте, а также полное описание приемочных испытаний для данного программного продукта.

Настоящая Программа и методика испытаний для исследования и разработки нового сериалайзера-десериалайзера ациклического графа из инстансов java beans component содержит следующие разделы: «Объект испытаний», «Цель испытаний», «Требования к библиотеке», «Требования к программной документации», «Средства и порядок испытаний», «Приложения» [7].

В разделе «Объект испытаний» указано наименование и область применения библиотеки .

В разделе «Цель испытаний» указана цель проведения испытаний. Раздел «Требования к библиотеке» содержит основные требования к библиотеке , которые подлежат проверке во время испытаний.

Раздел «Требования к программным документам» содержит состав программной документации, которая представляется на испытания.

Раздел «Средства и порядок испытаний» содержит информацию о технических и программных средствах, которые следует использовать во время испытаний, а также порядок этих испытаний.

Раздел «Методы испытаний» содержит информацию об используемых методах испытаний.

Настоящий документ разработан в соответствии с требованиями:

1) ГОСТ 19.101-77 Виды программ и программных документов [1];

2) ГОСТ 19.102-77 Стадии разработки [2];

3) ГОСТ 19.103-77 Обозначения программ и программных документов [3];

4) ГОСТ 19.104-78 Основные надписи [4];

5) ГОСТ 19.105-78 Общие требования к программным документам [5];

6) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом [6];

7) ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению [7].

Изменения к данному документу оформляются согласно ГОСТ 19.603-78 [8], ГОСТ 19.604-78 [9].

Перед прочтением данного документа рекомендуется ознакомиться с терминологией, приведенной в Приложении 1 настоящей программы и методики испытаний.

# **1. Объект испытаний**

**Наименование:** «Исследование и реализация сериалайзера-десериалайзера ациклического графа из инстансов java beans component».

**Краткая характеристика и область назначения:** «Реализованный сериалайзер-десериалайзер будет способен записывать ациклический граф объекта класса в XML/JSON файл, а также наоборот: имея JSON/XML файл получить объект класса.

# **2. Цель испытаний**

Целью испытаний является проверка корректности выполнения программой функций, перечисленных в разделе «Требования к программе».

# **3. Требования к программе**

Библиотека должна соответствовать следующим функциональным требованиям, указанным в документе «Исследование и реализация сериалайзера-десериалайзера ациклического графа из инстансов java beans component. Техническое задание»:

* + 1. **Основные функции:**

1) Сериализация ациклического графа;

2) Десиализация его сериализованного представления в файле

* + 1. **Дополнительные функции:**

Дополнительные функции не требуются

# **4. Требования к программной документации**

На испытание должна быть представлена документация в следующем составе:

1. «Исследование и реализация сериалайзера-десериалайзера ациклического графа из инстансов java beans component». Техническое задание (ГОСТ 19.201-78);
2. «Исследование и реализация сериалайзера-десериалайзера ациклического графа из инстансов java beans component». Программа и методика испытаний (ГОСТ 19.301-78);
3. «Исследование и реализация сериалайзера-десериалайзера ациклического графа из инстансов java beans component». Пояснительная записка (ГОСТ 19.404-79);
4. «Исследование и реализация сериалайзера-десериалайзера ациклического графа из инстансов java beans component». Руководство оператора (ГОСТ 19.505-79).
5. «Исследование и реализация сериалайзера-десериалайзера ациклического графа из инстансов java beans component». Текст программы (ГОСТ 19.401-78);

# **5. Средства и порядок испытаний**

## **5.1 Технические средства**

Так как проектом является библиотека, то технические средства должны обеспечить работу JVM

## **5.2 Программные средства**

Во время испытаний должны быть использованы следующие программные средства:

1) JDK

2) IDE для Java

## **5.3 Порядок проведения испытаний**

Испытания должны проводиться в следующем порядке:

1. Выполнить стандартную процедуру начальной загрузки операционной системы.

2. Вставить в компьютер информационный накопитель, содержащий программный продукт.

3. Создать проект в любой IDE для java, написать java bean классы и объекты которые требуется сериализовать

4. Скопировать содержимое папки ”src” находящейся в папке “HSE-2-course-work-java-beans-serializer”

5. Провести испытания, описанные ниже в разделе «Методы испытаний».

6. Закрыть IDE.

# **6. Методы испытаний**

Испытания представляют собой процесс установления соответствия библиотеки заданным требованиям и программной документации.

## **6.1 Испытания функциональности программы**

Функционал программы проверяется в соответствии с указанными в документе «Программа для шифрования и дешифрирования с помощью решетки Кардано. Техническое задание»:

**Основные функции:**

В разработанном проекте в директории “Tests” приведены тесты покрывающие все основные функции.

Файл: JavaSerializerTest.java

package SerializerAndDeserializer;

import Syntacse.XML.XmlDeserializing;

import Syntacse.XML.XmlSerializingSyntacse;

import java.beans.IntrospectionException;

import java.beans.XMLDecoder;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.lang.reflect.InvocationTargetException;

import java.util.ArrayList;

import org.junit.jupiter.api.Assertions;

import org.junit.jupiter.api.Test;

import test1.PrimitiveExample;

import test2.Company;

import test2.Person;

import test3.Arrays;

import test4.ArraysInArrays;

import test5.CyclicGraph;

class JavaSerializerTest {

@Test

void primitiveTests() {

try {

PrimitiveExample primitiveExample = new PrimitiveExample();

JavaSerializer serializer = new JavaSerializer(primitiveExample);

serializer.Make(new FileOutputStream("test1.xml"), new XmlSerializingSyntacse());

primitiveExample.setaBoolean(true);

primitiveExample.setaByte((byte) 1);

primitiveExample.setaChar('2');

primitiveExample.setaDouble(3d);

primitiveExample.setaFloat(4f);

primitiveExample.setaLong(5L);

primitiveExample.setAnInt(6);

primitiveExample.setaShort((short) 7);

primitiveExample.setaString("8");

FileOutputStream fileWriter = new FileOutputStream("test1.xml");

serializer.Make(fileWriter, new XmlSerializingSyntacse());

fileWriter.close();

JavaDeserializer javaDeserializer = new JavaDeserializer();

PrimitiveExample pe = (PrimitiveExample) javaDeserializer

.Make("test1.xml", new XmlDeserializing());

Assertions.assertTrue(pe.eq(primitiveExample));

} catch (Exception e) {

e.printStackTrace();

}

}

@Test

void simpleExample()

throws IOException, IllegalAccessException, IntrospectionException, InvocationTargetException {

Company company = new Company("Yandex Taxi", 999999999);

ArrayList<Integer> p1rating = new ArrayList<>();

p1rating.add(5);

p1rating.add(4);

p1rating.add(4);

p1rating.add(5);

Person p1 = new Person(21, 2343, company, new String[]{"Боря", "Вася", "Федя"}, p1rating,

4.5f, "Маша");

ArrayList<Integer> p2rating = new ArrayList<>();

p2rating.add(2);

p2rating.add(4);

p2rating.add(4);

p2rating.add(2);

Person p2 = new Person(19, 7490, company, new String[0], p2rating, 3f, "Леша");

ArrayList<Integer> p3rating = new ArrayList<>();

p3rating.add(5);

p3rating.add(5);

p3rating.add(5);

p3rating.add(5);

Person p3 = new Person(43, 9842, company, new String[]{"Саша"}, p3rating, 5f, "Гоша");

company.getPersons().add(p1);

company.getPersons().add(p2);

company.getPersons().add(p3);

JavaSerializer javaSerializer = new JavaSerializer(company);

FileOutputStream fileWriter = new FileOutputStream("test2.xml");

javaSerializer.Make(fileWriter, new XmlSerializingSyntacse());

fileWriter.close();

FileInputStream fileInputStream = new FileInputStream("test2.xml");

XMLDecoder xmlDecoder = new XMLDecoder(fileInputStream);

Company company1 = (Company) xmlDecoder.readObject();

fileInputStream.close();

JavaDeserializer javaDeserializer = new JavaDeserializer();

Company company2 = (Company) javaDeserializer.Make("test2.xml", new XmlDeserializing());

Assertions.assertEquals(company.eq(company1), company.eq(company2));

}

@Test

void arraysTest()

throws IOException, IllegalAccessException, IntrospectionException, InvocationTargetException {

Arrays arrays = new Arrays();

JavaSerializer serializer = new JavaSerializer(arrays);

FileOutputStream fileWriter = new FileOutputStream("test3.xml");

serializer.Make(fileWriter, new XmlSerializingSyntacse());

fileWriter.close();

arrays.setaBoolean(new boolean[]{true, false});

arrays.setaByte(new byte[]{0, 1});

arrays.setaChar(new char[]{'a', 'b'});

arrays.setaDouble(new double[]{3.2, 3.3});

arrays.setaFloat(new float[]{3.2f, 3.3f});

arrays.setaLong(new long[]{5, 5});

arrays.setAnInt(new int[]{6, 6});

arrays.setaShort(new short[]{7, 7});

arrays.setaString(new String[]{"hello", "world"});

fileWriter = new FileOutputStream("test3.xml");

serializer.Make(fileWriter, new XmlSerializingSyntacse());

fileWriter.close();

JavaDeserializer javaDeserializer = new JavaDeserializer();

Arrays pe = (Arrays) javaDeserializer.Make("test3.xml", new XmlDeserializing());

Assertions.assertTrue(pe.eq(arrays));

}

@Test

void arraysInArraysTest()

throws IOException, IllegalAccessException, IntrospectionException, InvocationTargetException {

ArraysInArrays arrays = new ArraysInArrays();

JavaSerializer serializer = new JavaSerializer(arrays);

FileOutputStream fileWriter = new FileOutputStream("test4.xml");

serializer.Make(fileWriter, new XmlSerializingSyntacse());

fileWriter.close();

arrays.setaBoolean(new boolean[][]{new boolean[]{true, false}});

arrays.setaByte(new byte[][]{new byte[]{0, 1}});

arrays.setaChar(new char[][]{new char[]{'a', 'b'}});

arrays.setaDouble(new double[][]{new double[]{3.2, 3.3}});

arrays.setaFloat(new float[][]{new float[]{3.2f, 3.3f}});

arrays.setaLong(new long[][]{new long[]{5, 5}});

arrays.setAnInt(new int[][]{new int[]{6, 6}});

arrays.setaShort(new short[][]{new short[]{7, 7}});

arrays.setaString(new String[][]{new String[]{"hello", "world"}});

fileWriter = new FileOutputStream("test3.xml");

serializer.Make(fileWriter, new XmlSerializingSyntacse());

fileWriter.close();

JavaDeserializer javaDeserializer = new JavaDeserializer();

ArraysInArrays pe = (ArraysInArrays) javaDeserializer

.Make("test3.xml", new XmlDeserializing());

Assertions.assertTrue(pe.eq(arrays));

}

@Test

void cyclicGraphTest()

throws IOException, IllegalAccessException, IntrospectionException, InvocationTargetException {

CyclicGraph c1 = new CyclicGraph();

CyclicGraph c2 = new CyclicGraph();

c1.setCyclicGraph(c2);

c2.setCyclicGraph(c1);

JavaSerializer serializer = new JavaSerializer(c1);

FileOutputStream fileWriter = new FileOutputStream("test5.xml");

serializer.Make(fileWriter, new XmlSerializingSyntacse());

fileWriter.close();

JavaDeserializer javaDeserializer = new JavaDeserializer();

CyclicGraph pe = (CyclicGraph) javaDeserializer.Make("test5.xml", new XmlDeserializing());

Assertions.assertSame(pe, pe.getCyclicGraph().getCyclicGraph());

}

}

Файл: PrimitiveExample.java

package test1;

public class PrimitiveExample {

private int anInt;

private double aDouble;

private boolean aBoolean;

private String aString;

private short aShort;

private long aLong;

private float aFloat;

private char aChar;

private byte aByte;

public long getaLong() {

return aLong;

}

public byte getaByte() {

return aByte;

}

public char getaChar() {

return aChar;

}

public float getaFloat() {

return aFloat;

}

public short getaShort() {

return aShort;

}

public void setaShort(short aShort) {

this.aShort = aShort;

}

public void setaByte(byte aByte) {

this.aByte = aByte;

}

public void setaFloat(float aFloat) {

this.aFloat = aFloat;

}

public void setaChar(char aChar) {

this.aChar = aChar;

}

public void setaLong(long aLong) {

this.aLong = aLong;

}

public double getaDouble() {

return aDouble;

}

public int getAnInt() {

return anInt;

}

public String getaString() {

return aString;

}

public void setaString(String aString) {

this.aString = aString;

}

public void setaBoolean(boolean aBoolean) {

this.aBoolean = aBoolean;

}

public void setaDouble(double aDouble) {

this.aDouble = aDouble;

}

public void setAnInt(int anInt) {

this.anInt = anInt;

}

public boolean isaBoolean() {

return aBoolean;

}

public boolean eq(PrimitiveExample p) {

return

p.aBoolean == aBoolean &&

p.aByte == aByte &&

p.aChar == aChar &&

p.aDouble == aDouble &&

p.aFloat == aFloat &&

p.aLong==aLong &&

p.anInt==anInt &&

p.aShort==aShort &&

p.aString.equals(aString);

}

}

Файл: Company.java

package test2;

import java.util.ArrayList;

public class Company {

public Company(){}

private String companyName;

private ArrayList<Person> persons;

private long budget;

public boolean eq(Company o) {

boolean b = companyName.equals(o.companyName) && budget==o.budget;

for (int i=0;i<persons.size();i++)

b=b && persons.get(i).eq(o.persons.get(i));

return b;

}

public Company(String companyName, long budget) {

this.budget = budget;

this.companyName = companyName;

persons = new ArrayList<Person>();

}

public ArrayList<Person> getPersons() {

return persons;

}

public long getBudget() {

return budget;

}

public String getCompanyName() {

return companyName;

}

public void setBudget(long budget) {

this.budget = budget;

}

public void setCompanyName(String companyName) {

this.companyName = companyName;

}

public void setPersons(ArrayList<Person> persons) {

this.persons = persons;

}

}

Файл: Person.java

package test2;

import java.util.List;

public class Person {

public boolean eq(Person p){

boolean b=

age==p.age &&

childs==p.childs &&

telephone==p.telephone &&

rate==p.rate &&

name==p.name;

for(int i=0;i<ratingHistory.size();i++)

b=b && ratingHistory.get(i).equals(p.ratingHistory.get(i));

return b;

}

public Person(int age, int telephone, Company company, String[] childs,

List<Integer> ratingHistory, float rate, String name) {

this.age = age;

this.childs = childs;

this.company = company;

this.telephone = telephone;

this.ratingHistory = ratingHistory;

this.rate = rate;

this.name = name;

}

private String name;

private int age;

private int telephone;

private Company company;

private String[] childs;

private List<Integer> ratingHistory;

private float rate;

public Person() {

}

public void setName(String name) {

this.name = name;

}

public String getName() {

return name;

}

public int getAge() {

return age;

}

public Company getCompany() {

return company;

}

public float getRate() {

return rate;

}

public int getTelephone() {

return telephone;

}

public List<Integer> getRatingHistory() {

return ratingHistory;

}

public String[] getChilds() {

return childs;

}

public void setAge(int age) {

this.age = age;

}

public void setChilds(String[] childs) {

this.childs = childs;

}

public void setRate(float rate) {

this.rate = rate;

}

public void setCompany(Company company) {

this.company = company;

}

public void setRatingHistory(List<Integer> ratingHistory) {

this.ratingHistory = ratingHistory;

}

public void setTelephone(int telephone) {

this.telephone = telephone;

}

}

Файл: Arrays.java

package test3;

public class Arrays {

private int[] anInt;

private double[] aDouble;

private boolean[] aBoolean;

private String[] aString;

private short[] aShort;

private long[] aLong;

private float[] aFloat;

private char[] aChar;

private byte[] aByte;

public boolean eq(Arrays p) {

return

java.util.Arrays.equals(p.aBoolean, aBoolean) &&

java.util.Arrays.equals(p.aByte, aByte) &&

java.util.Arrays.equals(p.aChar, aChar) &&

java.util.Arrays.equals(p.aDouble, aDouble) &&

java.util.Arrays.equals(p.aFloat, aFloat) &&

java.util.Arrays.equals(p.aLong, aLong) &&

java.util.Arrays.equals(p.anInt, anInt) &&

java.util.Arrays.equals(p.aShort, aShort) &&

java.util.Arrays.equals(p.aString, aString);

}

public void setAnInt(int[] anInt) {

this.anInt = anInt;

}

public void setaDouble(double[] aDouble) {

this.aDouble = aDouble;

}

public void setaLong(long[] aLong) {

this.aLong = aLong;

}

public void setaShort(short[] aShort) {

this.aShort = aShort;

}

public void setaBoolean(boolean[] aBoolean) {

this.aBoolean = aBoolean;

}

public boolean[] getaBoolean() {

return aBoolean;

}

public void setaChar(char[] aChar) {

this.aChar = aChar;

}

public void setaString(String[] aString) {

this.aString = aString;

}

public double[] getaDouble() {

return aDouble;

}

public int[] getAnInt() {

return anInt;

}

public String[] getaString() {

return aString;

}

public void setaFloat(float[] aFloat) {

this.aFloat = aFloat;

}

public short[] getaShort() {

return aShort;

}

public void setaByte(byte[] aByte) {

this.aByte = aByte;

}

public byte[] getaByte() {

return aByte;

}

public char[] getaChar() {

return aChar;

}

public float[] getaFloat() {

return aFloat;

}

public long[] getaLong() {

return aLong;

}

}

Файл: ArraysInArrays.java

package test4;

import test3.Arrays;

public class ArraysInArrays {

private int[][] anInt;

private double[][] aDouble;

private boolean[][] aBoolean;

private String[][] aString;

private short[][] aShort;

private long[][] aLong;

private float[][] aFloat;

private char[][] aChar;

private byte[][] aByte;

public boolean eq(ArraysInArrays p) {

return

java.util.Arrays.deepEquals(p.aBoolean, aBoolean) &&

java.util.Arrays.deepEquals(p.aByte, aByte) &&

java.util.Arrays.deepEquals(p.aChar, aChar) &&

java.util.Arrays.deepEquals(p.aDouble, aDouble) &&

java.util.Arrays.deepEquals(p.aFloat, aFloat) &&

java.util.Arrays.deepEquals(p.aLong, aLong) &&

java.util.Arrays.deepEquals(p.anInt, anInt) &&

java.util.Arrays.deepEquals(p.aShort, aShort) &&

java.util.Arrays.deepEquals(p.aString, aString);

}

public void setaString(String[][] aString) {

this.aString = aString;

}

public void setaDouble(double[][] aDouble) {

this.aDouble = aDouble;

}

public void setaFloat(float[][] aFloat) {

this.aFloat = aFloat;

}

public void setaChar(char[][] aChar) {

this.aChar = aChar;

}

public void setaByte(byte[][] aByte) {

this.aByte = aByte;

}

public void setaBoolean(boolean[][] aBoolean) {

this.aBoolean = aBoolean;

}

public void setaShort(short[][] aShort) {

this.aShort = aShort;

}

public void setaLong(long[][] aLong) {

this.aLong = aLong;

}

public void setAnInt(int[][] anInt) {

this.anInt = anInt;

}

public boolean[][] getaBoolean() {

return aBoolean;

}

public byte[][] getaByte() {

return aByte;

}

public char[][] getaChar() {

return aChar;

}

public double[][] getaDouble() {

return aDouble;

}

public float[][] getaFloat() {

return aFloat;

}

public int[][] getAnInt() {

return anInt;

}

public long[][] getaLong() {

return aLong;

}

public short[][] getaShort() {

return aShort;

}

public String[][] getaString() {

return aString;

}

}

Файл CyclicGraph.java

package test5;

public class CyclicGraph {

private CyclicGraph cyclicGraph;

public CyclicGraph getCyclicGraph() {

return cyclicGraph;

}

public void setCyclicGraph(CyclicGraph cyclicGraph) {

this.cyclicGraph = cyclicGraph;

}

}

# **ПРИЛОЖЕНИЕ 1**

**ТЕРМИНОЛОГИЯ**

Таблица 1

|  |  |
| --- | --- |
| **Термин** | **Определение** |
| **Сериализация** | Процесс преобразования структур данных или состояния объекта в формат, который может быть сохранен (например, в файле или буфере памяти) |
| **Десериализация** | Обратный процесс сериализации |
| **XML** | Расширяемый язык разметки. Рекомендован Консорциумом Всемирной паутины (W3C). Спецификация XML описывает XML-документы и частично описывает поведение XML-процессоров (программ, читающих XML-документы и обеспечивающих доступ к их содержимому). |
| **JSON** | Текстовый формат обмена данными, основанный на JavaScript. Как и многие другие текстовые форматы, JSON легко читается людьми. |

**ПРИЛОЖЕНИЕ 2**

**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.104-78 Основные надписи. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.603-78 Общие правила внесения изменений. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 15150-69 Машины, приборы и другие технические изделия. Исполнения для различных климатических районов. Категории, условия эксплуатации, хранения и транспортирования в части воздействия климатических факторов внешней среды. – М.: Изд-во стандартов, 1997.
11. ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
12. Oracle // Java Platform Standard Edition 8 Documentation - https://docs.oracle.com/javase/8/docs/

# **ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированх |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |