Министерство образования и науки Российской Федерации

Севастопольский государственный университет

Институт информационных технологий и управления в технических системах

Кафедра ИС

ОТЧЕТ

по лабораторной работе №2

ОСНОВЫ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ НА ЯЗЫКЕ JAVA

Выполнил:

ст. гр. ИС/б-21-о

Куркчи А. Э.

Проверил:

Кузнецов А. С.

Севастополь

2016

1. ЦЕЛЬ РАБОТЫ

В ходе выполнения данной лабораторной работы необходимо ознакомиться с особенностями объектно-ориентированного программирования (ООП) на языке Java, приобрести практические навыки программирования на языке Java с использованием основных принципов ООП.

1. ПОСТАНОВКА ЗАДАЧИ

2.1. Описать абстрактный класс CBuffer, содержащий следующие поля:

* идентификатор буфера (**int** bufID) – уникальный идентификатор буфера;
* размер буфера (**int** bufSize) – максимальный размер буфера;
* количество созданных буферов **(int** BufCount).

Доступ к полям класса CBuffer должны иметь только методы этого класса и методы его потомков. Для организации доступа к этим полям из других классов необходимо реализовать общедоступные методы:

* **int** GetBufCount();
* **int** GetBufID().

Реализовать конструктор класса CBuffer(**int** count), выполняющий инициализацию идентификатора буфера(в качестве идентификатора использовать номер по порядку создаваемого буфера), размера буфера (значением count, передаваемым конструктору), увеличение количества созданных буферов.

В классе CBuffer описать абстрактный метод Generate().

2.2. В соответствии с вариантом задания реализовать дочерний класс для создания буфера, хранящего значения заданного типа T(см. таблицу 4.1). Для хранения значений реализовать поле – массив значений типа Т. В конструкторе класса использовать вызов конструктора родительского класса CBuffer, и кроме того создать массив значений типа Т(с использованием оператора **new)** и проинициализировать его с импользованием метода Generate().

Pеализовать метод Generate(), заполняющий массив случайными числами.

Для генерации случайных чисел необходимо, используя оператор import, подключить пакет java.util.Random. Для использования генератора случайный чисел сначала необходимо создать экземпляр класса Random:

*Random random = new Random();*

Генерация случайных чисел выполняется методами экземпляра класса Random. Например, для генерации случайного целого числа:

*random.nextInt();*

2.3. Описать интерфейсы:

1) IBufferPrintable – описывающий методы вывода на экран:

* **public** **void** PrintInfo() – выводит на экран идентификатор, тип и размер буфера.
* **public** **void** Print() – выводит на экран содержимое буфера.
* **public** **void** PrintFirstN(**int** n) – выводит на экран первые n элементов буфера.
* **public** **void** PrintLastN(**int** n) – выводит на экран последние n элементов буфера.

2) IBufferSortable – описывает метод для сортировки массива:

* **public** **void** Sort();

3) IBufferComputable – описывает методы для вычисления статистики значений буфера.

* **public void** Max() – вычисляет максимальный элемент буфера;
* **public void** Min() – вычисляет минимальный элемент буфера;
* **public void** Sum() – вычисляет сумму элементов буфера;

4) IBufferStorable – описывает методы для выгрузки буфера в текстовый файл.

* **public void** SaveOneLine(String filename) – сохраняет буфер в файл в одну строку;
* **public void** SaveSeparateLines (String filename) – сохраняет буфер в файл по одному элементу в строке;

2.4. Создать произвольный класс, унаследованный от класса, разработанного при выполнении п. 2.2, и реализующий методы интерфейсов из п. 2.3, необходимых для выполнения задания в соответствии с вариантом (см. таблицу 4.1).

2.5. Реализовать класс Lab2Java, в методе main которого в соответствии с вариантом задания (см. таблицу 4.1) реализовать работу с объектами класса из п. 2.4 с использованием их методов:

* Создать N буферов заданного типа T и размера L;
* Вывести на экран информацию o буферах;
* Вывести на экран первые 10 элементов буферов;
* Вычислить функцию F для каждого буфера и вывести результат на экран;
* Выполнить сортировку буферов методом S;
* Вывести на экран первые 10 элементов буферов;
* Сохранить буферы в файл с использованием метода O.

При написании программы допускается расширение классов необходимыми полями и методами.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 3 | Float | 80 | Выборки | Sum | SaveSeparateLines |

Вариант № 6

1. ТЕКСТ ПРОГРАММЫ

Файл Main.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab02**;**

public class Main **{**

public static void main**(**String**[]** args**)** **{**

int n **=** 3**;**

int l **=** 80**;**

CFloatBuffer bufs**[]** **=** **new** CFloatBuffer**[**n**];**

**for(**int i**=**0**;**i**<**n**;**i**++)** **{**

bufs**[**i**]** **=** **new** CFloatBuffer**(**l**);**

bufs**[**i**].**printInfo**();**

bufs**[**i**].**sum**();**

System**.**out**.**println**(**"First 10:"**);**

System**.**out**.**print**(**"Before: "**);**

bufs**[**i**].**printFirst**(**10**);**

bufs**[**i**].**sort**();**

System**.**out**.**print**(**"After: "**);**

bufs**[**i**].**printFirst**(**10**);**

bufs**[**i**].**printSeparateLines**(**"buffer"**+**String**.**valueOf**(**bufs**[**i**].**getBufID**())+**".txt"**);**

**}**

**}**

**}**

Файл IBufferStorable.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab02**;**

public interface IBufferStorable **{**

void printOneLine**(**String filename**);**

void printSeparateLines**(**String filename**);**

**}**

Файл IBufferSortable.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab02**;**

public interface IBufferSortable **{**

void sort**();**

**}**

Файл IBufferPrintable.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab02**;**

public interface IBufferPrintable **{**

void printInfo**();**

void print**();**

void printFirst**(**int n**);**

void printLast**(**int n**);**

**}**

Файл IBufferComputable.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab02**;**

public interface IBufferComputable **{**

void max**();**

void min**();**

void sum**();**

**}**

Файл CBuffer.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab02**;**

public abstract class CBuffer **{**

public static int bufCount **=** 0**;**

protected int bufID**;**

protected int bufSize**;**

public CBuffer**(**int count**)** **{**

bufID **=** **++**CBuffer**.**bufCount**;**

bufSize **=** count**;**

**}**

public abstract void generate**();**

public int getBufID**()** **{**

**return** bufID**;**

**}**

public int getBufSize**()** **{**

**return** bufSize**;**

**}**

**}**

Файл CFloatBuffer.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab02**;**

**import** java**.**io**.\*;**

**import** java**.**util**.**Random**;**

public class CFloatBuffer **extends** CBuffer **implements** IBufferComputable**,** IBufferSortable**,** IBufferStorable**,** IBufferPrintable **{**

protected float buf**[];**

public CFloatBuffer**(**int count**)** **{**

**super(**count**);**

buf **=** **new** float**[**count**];**

generate**();**

**}**

@Override

public void generate**()** **{**

Random rnd **=** **new** Random**();**

**for(**int i**=**0**;**i**<**bufSize**;**i**++)** **{**

buf**[**i**]** **=** rnd**.**nextFloat**();**

**}**

**}**

@Override

public void max**()** **{**

float max **=** buf**[**0**];**

**for(**int i**=**1**;**i**<**bufSize**;**i**++)** **{**

**if(**buf**[**i**]** **>** max**)** **{**

max **=** buf**[**i**];**

**}**

**}**

\_log**(**"Max: "**,**max**,**'\n'**);**

**}**

@Override

public void min**()** **{**

float min **=** buf**[**0**];**

**for(**int i**=**1**;**i**<**bufSize**;**i**++)** **{**

**if(**buf**[**i**]** **<** min**)** **{**

min **=** buf**[**i**];**

**}**

**}**

\_log**(**"Min: "**,**min**,**'\n'**);**

**}**

@Override

public void sum**()** **{**

float sum **=** 0**;**

**for(**int i**=**0**;**i**<**bufSize**;**i**++)** **{**

sum **+=** buf**[**i**];**

**}**

\_log**(**"Sum: "**,**sum**,**'\n'**);**

**}**

@Override

public void sort**()** **{**

int m**;**

float tmp**;**

**for(**int i**=**0**;**i**<**bufSize**-**1**;**i**++)** **{**

m **=** i**;**

**for(**int j**=**i**+**1**;**j**<**bufSize**;**j**++)** **{**

**if(**buf**[**j**]** **<** buf**[**m**])** **{**

m **=** j**;**

**}**

**}**

**if(**m **!=** i**)** **{**

tmp **=** buf**[**i**];**

buf**[**i**]** **=** buf**[**m**];**

buf**[**m**]** **=** tmp**;**

**}**

**}**

**}**

@Override

public void printOneLine**(**String filename**)** **{**

printToFile**(**filename**,**' '**);**

**}**

@Override

public void printSeparateLines**(**String filename**)** **{**

printToFile**(**filename**,**'\n'**);**

**}**

private void printToFile**(**String filename**,** char separator**)** **{**

**try(**PrintWriter out **=** **new** PrintWriter**(new** FileOutputStream**(**filename**,true)))** **{**

**for** **(**int i **=** 0**;** i **<** bufSize**;** i**++)** **{**

out**.**print**(**buf**[**i**]);**

out**.**print**(**separator**);**

**}**

out**.**print**(**'\n'**);**

out**.**close**();**

**}** **catch** **(**FileNotFoundException e**)** **{**

\_log**(**"File not found: "**,**filename**);**

e**.**printStackTrace**();**

**}**

**}**

@Override

public void printInfo**()** **{**

\_log**(**'\t'**,**"Buffer #"**,**bufID**,**" size: "**,**bufSize**,**'\n'**);**

**}**

@Override

public void print**()** **{**

**for(**int i**=**0**;**i**<**bufSize**;**i**++)** **{**

\_log**(**buf**[**i**],**' '**);**

**}**

\_log**(**'\n'**);**

**}**

@Override

public void printFirst**(**int n**)** **{**

**for(**int i**=**0**;**i**<**Math**.**min**(**bufSize**,**n**);**i**++)** **{**

\_log**(**buf**[**i**],**' '**);**

**}**

\_log**(**'\n'**);**

**}**

@Override

public void printLast**(**int n**)** **{**

**for** **(**int i **=** Math**.**max**(**0**,** bufSize **-** n**);** i **<** bufSize**;** i**++)** **{**

\_log**(**buf**[**i**],** ' '**);**

**}**

\_log**(**'\n'**);**

**}**

private static void \_log**(**Object**...** args**)** **{**

StringBuilder sb **=** **new** StringBuilder**();**

**for(**Object str **:** args**)** **{**

sb**.**append**(**str**);**

**}**

System**.**out**.**print**(**sb**.**toString**());**

**}**

**}**

Вывод

В ходе данной лабораторной работы были изучены особенности объектно-ориентированного программирования (ООП) на языке Java, приобретены практические навыки программирования на языке Java с использованием основных принципов ООП.