Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №3

По дисциплине: «Современные платформы программирования»

Выполнила:
Студентка 3 курса
Группы ПО-6
Юсковец М.А.
Проверил:
Монтик Н.С.

Брест, 2022

Цель работы: научиться создавать и использовать классы в программах на языке программирования Java

Ход работы:

Вариант 25

Задание 1:

Реализовать простой класс.

Требования к выполнению
• Реализовать пользовательский класс по варианту.
• Создать другой класс с методом main, в котором будут находится примеры использования пользовательского класса.

Для каждого класса:
• Создать поля классов
• Создать методы классов
• Добавьте необходимые get и set методы (по необходимости)
• Укажите соответствующие модификаторы видимости
• Добавьте конструкторы
• Переопределить методы toString() и equals()

Множество целых чисел ограниченной мощности – Предусмотреть возможность объединения двух множеств, вывода на печать элементов множества, а так же метод, определяющий, принадлежит ли указанное значение множеству. Класс должен содержать методы, позволяющие добавлять и удалять элемент в/из множества. Конструктор должен позволить создавать объекты с начальной инициализацией. Мощность множества задается при создании объекта. Реализацию множества осуществить на базе одномерного массива. Реализовать метод equals, выполняющий сравнение объектов данного типа.

Текст программы:

Set.java

```
package setPackage;

public class Set {
    private Integer[] set;
    public Set(Integer[] set){
        this.set = new Integer[set.length];
        this.set = set;
    }
```

```
    public Set(int N){
        this.set = new Integer[N];
    }
    public Integer[] sort(Integer[] set){
        for (int i = 0; i < set.length; ++i) {
            for (int j = 0; j < set.length - 1; ++j) {
                if (set[j] > set[j + 1]){
                    Integer temp = set[j];
                    set[j] = set[j + 1];
                    set[j + 1] = temp;
                }
            }
        }
        return set;
    }
    public Set mergeSets(Set set2){
        Set mergedSet = new Set(this.set.length);
        for(int i = 0; i < this.set.length; ++i){
            mergedSet.set[i] = set[i];
        }
        for(int i = 0; i < set2.set.length; ++i){
            mergedSet.addElement(set2.set[i]);
        }
        mergedSet.set = sort(mergedSet.set);
        for(int i = 0; i < mergedSet.set.length; ++i){
            System.out.print(mergedSet.set[i]+" ");
        }
        System.out.println();
        return mergedSet;
    }
    public boolean isPartOfSet(int x){
        for(int i = 0; i < this.set.length; ++i){
            if(this.set[i].equals(x)) {
                return true;
            }
        }
        return false;
    }
    public void printSet(){
        System.out.println("Set:");
        for(int i = 0; i < this.set.length; ++i){
            System.out.print(this.set[i]+" ");
        }
        System.out.println();
    }
    public boolean addElement(int el){
        if (!this.isPartOfSet(el)){
            Integer[] newSet = new Integer[this.set.length + 1];

            for(int i = 0; i < this.set.length; ++i){
                newSet[i] = this.set[i];
            }
            newSet[this.set.length] = el;
            this.set = sort(newSet);
            return true;
        }
        return false;
    }
    public boolean deleteElement(int el){
        boolean isDeleted = this.isPartOfSet(el);
        Integer[] newSet = new Integer[this.set.length - 1];
        for(int i = 0, j = 0; i < this.set.length; ++i, ++j){
            if(this.set[i].equals(el)){
                ++i;
```

```
            }
            newSet[j] = this.set[i];
        }
        this.set = newSet;
        return isDeleted;
    }
    public boolean equals(Set set2){
        if(this.set.length != set2.set.length){
            return false;
        }
        for(int i = 0; i < this.set.length; ++i){
            if(!set2.set[i].equals(this.set[i])){
                return false;
            }
        }
        return true;
    }
    public String toString(){
        String result = new String();
        for(int i = 0; i < this.set.length; ++i){
            result += Integer.toString(this.set[i]);
        }
        return result;
    }
}
```

Main.java

```
import setPackage.Set;

public class Main {
    public static void main(String[] args) {
        Set set1 = new Set(new Integer[]{1, 3, 5});
        set1.printSet();
        set1 = set1.mergeSets(new Set(new Integer[] {3, 4, 8}));
        set1.printSet();

        System.out.println(set1.isPartOfSet(3));

        set1.addElement(9);
        set1.addElement(1);
        set1.deleteElement(5);
        set1.printSet();

        System.out.println(set1.equals(new Set(new Integer[] {1, 3, 4, 7,
8})));
        System.out.println(set1.equals(new Set(new Integer[] {1, 3, 4, 8,
9})));
        System.out.println(set1.toString());
    }
}
```

Результат программы:

```
Set:
1 3 5
1 3 4 5 8
Set:
1 3 4 5 8
true
Set:
1 3 4 8 9
false
true
13489

Process finished with exit code 0
```

Задание 2:

Разработать автоматизированную систему на основе некоторой структуры данных, манипулирующей объектами пользовательского класса. Реализовать требуемые функции обработки данных

Требования к выполнению

• Задание посвящено написанию классов, решающих определенную задачу автоматизации;

• Данные для программы загружаются из файла (формат произволен). Файл создать и написать вручную.

Моделирование файловой системы

Составить программу, которая моделирует заполнение гибкого диска (1440 Кб). В процессе работы файлы могут записываться на диск и удаляться с него.

С каждым файлом (File) ассоциированы следующие данные:

• Размер

• Расширение

• Имя файла

• Как файлы могут трактоваться и директории, которые в свою очередь содержат другие файлы и папки.

Если при удалении образовался свободный участок, то вновь записываемый файл помещается на этом свободном участке, либо, если он не помещается на этом участке, то его следует разместить после последнего записанного файла. Если файл превосходит длину самого большого участка, выдается аварийное сообщение. Рекомендуется создать список свободных участков и список занятых участков памяти на диске.

Текст программы:

Main.java

```
package package1;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.stream.IntStream;

public class Main {
    public static void GenerateFileSystem(List<File> file, int[] freeSpace,
int numFiles)
    {
        for (int i = 0; i < numFiles; i++)
        {
            File file1 = new File(i + 1, freeSpace);
            if (!file1.equals(null))
            {
                file.add(file1);
            }
        }
    }

    public static void DeleteFile(List<File> file, int[] freeSpace, String
fileName)
    {
        for(int i = 0; i < file.size(); i++)
        {
            if(file.get(i).getFileName() == fileName)
            {
                int temp1 = File.getArrayFirstOccurIndex(freeSpace,
file.get(i).ID);
                int temp2 = File.getArrayFirstOccurIndex(freeSpace,
file.get(i).ID) + file.get(i).getFileSize();
                file.remove(i);
                for(int j = temp1; j < temp2; j++)
                {
                    freeSpace[j] = 0;
                }
                i--;
                System.out.println("Файл удален!");
                //break; //Если нужно удалять только один файл
            }
        }
    }

    public static void AddFile(List<File> file, int[] freeSpace)
    {
        int[] tempArr = freeSpace;
//        IEnumerable<Integer> nums = IntStream.of(tempArr).distinct();
```

```
        tempArr = IntStream.of(tempArr).toArray();
        int j = 0;
        for(; j < tempArr.length + 1; j++)
        {
            if (!Arrays.asList(tempArr).contains(j))
            {
                break;
            }
        }

        File file1 = new File(j, freeSpace);
        if (!file1.Empty)
        {
            file.add(file1);
        }
    }

    public static void main(String[] args) {
        int[] FreeSpace = new int[1440];
        int NumFiles = 10;
        ArrayList<File> FileSystem = new ArrayList<File>();
        GenerateFileSystem(FileSystem, FreeSpace, NumFiles);
        for(int i = 0; i < FileSystem.size(); i++)
        {
            FileSystem.get(i).Print();
        }
        DeleteFile(FileSystem, FreeSpace, "da");

        AddFile(FileSystem, FreeSpace);

        for (int i = 0; i < FileSystem.size(); i++)
        {
            FileSystem.get(i).Print();
        }
    }
}
```

File.java

```
package package1;
import java.util.Random;
public class File {
    public static int FileSystemSize = 1440;
    private int FileSize;
    private String FileExtension;
    private String FileName;

    public int ID;
    public int NumOfFiles;
    public Boolean Empty;

    public int getFileSize() {
        return FileSize;
    }

    public void setFileSize(int value) {
        FileSize = value;
    }


    public String getFileExtension() {
        return FileExtension;
```

```
    }

    public void setFileExtension(String value) {
        FileExtension = value;
    }


    public String getFileName() {
        return FileName;
    }

    public void setFileName(String value) {
        FileName = value;
    }

    static String randomString(int lenght) {
        Random rand = new Random();
//          Console.OutputEncoding = System.Text.Encoding.UTF8;

        String a = "";

        for (int i = 0; i < lenght; i++) {
            a += (char)rand.nextInt(26) + 'a';
        }
        return a;
    }

    public void Print() {
        System.out.println("Название файла: "+getFileName() + "\t Расширение:
"+getFileExtension()+"\t Размер: "+getFileSize());
    }

    public void CheckFreeSpace() {
        for(int i = 0; i < FileSystemSize; i++) {}
    }

    public void FillingFreeSpace(int[] FreeSpace) {
        if (ID == 1) {
            for (int i = 0; i < getFileSize(); i++) {
                FreeSpace[i] = ID;
            }
        }
        else {
            int index = getArrayFirstOccurIndex(FreeSpace, 0);
            int total = 0;
            for (int i = index; i < FreeSpace.length - 1; i++) {
                if (FreeSpace[i] == 0) {
                    total++;
                    if (total == getFileSize()) {
                        for (int j = index; j < getFileSize() + index; j++) {
                            FreeSpace[j] = ID;
                        }
                        break;
                    }
                }
                else {
                    total = 0;
                    index = getArrayFirstOccurIndexFromCurrent(FreeSpace, 0,
i);
                }
            }
            if (total < getFileSize()) {
                System.out.println("Не хватает места для файла!");
                Empty = true;
```

```
            }
            else Empty = false;
        }
    }

    public static int getArrayFirstOccurIndex(int[] arr, int elem){
        for(int i = 0; i < arr.length; i++) {
            if (arr[i] == elem) {
                return i;
            }
        }
        return -1;
    }

    public static int getArrayFirstOccurIndexFromCurrent(int[] arr, int ind,
int elem){
        for(int i = 0; i < arr.length; i++) {
            if (arr[i] == elem) {
                for(int j = i; j < arr.length; j++) {
                    if (arr[j]==ind)
                        return j;
                }
            }
        }
        return -1;
    }

    public File(int id, int[] FreeSpace) {
        Random temp = new Random();
        String[] fe = new String[] { ".txt", ".rar", ".bat", ".dox", ".exe"
};
        String[] fn = new String[] { "tre", "da", "net", "ladno" };
        ID = id;
        setFileSize(temp.nextInt(150, 250));
        setFileExtension(fe[temp.nextInt(0, 5)]);
        setFileName(fn[temp.nextInt(0, 4)]); //randomString(temp.Next(3, 8));

        FillingFreeSpace(FreeSpace);
    }
}
```

Результат программы:

```
Не хватает места для файла!
Не хватает места для файла!
Не хватает места для файла!
Название файла: net  Расширение: .dox    Размер: 165
Название файла: ladno    Расширение: .dox    Размер: 245
Название файла: ladno    Расширение: .txt    Размер: 166
Название файла: ladno    Расширение: .txt    Размер: 223
Название файла: ladno    Расширение: .exe    Размер: 182
Название файла: ladno    Расширение: .rar    Размер: 172
Название файла: da   Расширение: .bat    Размер: 237
Название файла: ladno    Расширение: .txt    Размер: 226
Название файла: tre  Расширение: .txt    Размер: 232
Название файла: net  Расширение: .rar    Размер: 208
Файл удален!
Название файла: net  Расширение: .dox    Размер: 165
Название файла: ladno    Расширение: .dox    Размер: 245
Название файла: ladno    Расширение: .txt    Размер: 166
Название файла: ladno    Расширение: .txt    Размер: 223
Название файла: ladno    Расширение: .exe    Размер: 182
Название файла: ladno    Расширение: .rar    Размер: 172
Название файла: ladno    Расширение: .txt    Размер: 226
Название файла: tre  Расширение: .txt    Размер: 232
Название файла: net  Расширение: .rar    Размер: 208
Название файла: ladno    Расширение: .rar    Размер: 211

Process finished with exit code 0
```

Вывод: научились создавать и использовать классы в программах на языке программирования Java