Zhuban Marlen SIS-2302 IITU ID- 37681

My project consists of two main Java classes: *Main and Student, and an Excel file with students. Below is a detailed description of each file.*

	A	B	C
1	Name	Current Scholarship	New Scholarship
2	Zhuban Marlen	42795	47367
3	Asar Abunasir	42795	47367
4	Medetbokov Aldiyar	42795	47367

Main.java

```
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.util.List;
import java.util.ArrayList;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.apache.poi.ss.usermodel.*;
```

import java.io.File;

Import the File class for working with files, such as creating a file object.

import java.io.FileInputStream;

Import the FileInputStream class, which is used to read the contents of files.

import java.io.IOException;

Import a class for handling I/O errors.

import java.util.List;

Import the List interface for working with lists.

import java.util.ArrayList;

Import the ArrayList class, which implements the List interface.

import org.apache.poi.xssf.usermodel.XSSFWorkbook;

Import the XSSFWorkbook class for working with Excel files of the .xlsx format.

import org.apache.poi.ss.usermodel.*;

Import all classes from the org.apache.poi.ss.usermodel package that allow you to work with Excel sheets, rows, and cells.

```
public class Main {
```

public class Main

Declaring the main class of the program. This class contains the main method, which is the entry point to the program.

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

public static void main(String[] args)

Declares the main method of the program. It is the first method that is run when the program is executed.

```
String filePath = "C:\\Users\\marle\\IdeaProjects\\practice
12\\students.xlsx";
```

String filePath = ...;

A string variable is declared that stores the path to the Excel file from which the data will be read.

```
List<Student> students = readStudentsFromExcel(filePath);
```

List<Student> students = readStudentsFromExcel(filePath);

A list of Student objects is created and populated with data read from the Excel file using the readStudentsFromExcel method.

```
for (Student student : students) {
```

for (Student student : students)

Begins a for-each loop that iterates over each Student object in the students list.

```
for (Student student : students) {
    System.out.printf("Имя: %s, Текущая стипендия: %.2f, Новая
стипендия: %.2f, Увеличение: %.2f%n",
            student.getName(),
            student.getCurrentScholarship(),
            student.getNewScholarship(),
            student.getScholarshipIncrease());
```

System.out.printf(...)

Print information about each student to the console. Formatted output includes:

Student name (student.getName()).

Current scholarship (student.getCurrentScholarship()).

New scholarship (student.getNewScholarship()).

Scholarship increase (student.getScholarshipIncrease()).

```
public static List<Student> readStudentsFromExcel(String filePath) {
    List<Student> students = new ArrayList<>();
```

public static List<Student> readStudentsFromExcel(String filePath)

A method that reads data from an Excel file and returns a list of Student objects.

List<Student> students = new ArrayList<>();

An empty students list is created, where Student objects will be added.

```
try (FileInputStream fis = new FileInputStream(new File(filePath));
     XSSFWorkbook workbook = new XSSFWorkbook(fis)) {
```

try (...)

A try-with-resources block for safely closing resources.

FileInputStream fis = ...;

A stream is created to read the file.

XSSFWorkbook workbook = ...;

A workbook object is created to work with the Excel workbook.

```
Sheet sheet = workbook.getSheetAt(0);
```

Sheet sheet = workbook.getSheetAt(0);

Extracts the first sheet from the Excel file.

```
for (int i = 1; i <= sheet.getLastRowNum(); i++) {
```

for (int i = 1; i <= sheet.getLastRowNum(); i++)

A loop for processing all rows of a sheet, starting with the second row (the first row is the headers).

```
Row row = sheet.getRow(i);

String name = row.getCell(0).getStringCellValue();
double currentScholarship = row.getCell(1).getNumericCellValue();
double newScholarship = row.getCell(2).getNumericCellValue();
```

Row row = ...;

Get the current row.

String name = ...;

Get the student name from the first cell.

double currentScholarship = ...;

Get the current scholarship from the second cell.

double newScholarship = ...;

Get the new scholarship from the third cell.

```
students.add(new Student(name, currentScholarship, newScholarship));
```

students.add(...)

A Student object is created with the extracted data and added to the students list.

```
} catch (IOException e) {
    e.printStackTrace();
}
```

catch (IOException e)

Handling possible errors when working with files.

```
    return students;
     }
}
```

return students;

Returns the list of students from the method.

}

End of the method and class.

Student.java

```
public class Student {
    private String name;
    private double currentScholarship;
    private double newScholarship;
```

public class Student

Declaration of the Student class.

private String name;

Field for storing the student's name.

private double currentScholarship;

Field for the current scholarship.

private double newScholarship;

Field for the new scholarship.

```
public Student(String name, double currentScholarship, double newScholarship)
{
    this.name = name;
    this.currentScholarship = currentScholarship;
    this.newScholarship = newScholarship;
}
```

Class Constructor

Initializes all fields of the object upon creation.

```
public String getName() {
    return name;
}
```

getName Method

Returns the student's name.

```
public double getCurrentScholarship() {
    return currentScholarship;
}
```

getCurrentScholarship Method

Returns the current scholarship.

```
public double getNewScholarship() {
    return newScholarship;
}
```

getNewScholarship Method

Returns the new scholarship.

```
    public double getScholarshipIncrease() {
          return newScholarship - currentScholarship;
      }
}
```

getScholarshipIncrease Method

Returns the scholarship increase (*the difference between the new and current scholarships*).

}

End of class.