

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**"МИРЭА - Российский технологический университет"**

**РТУ МИРЭА**

Институт информационных технологий(ИТ)

Кафедра практической прикладной информатики(ППИ)

**ОТЧЕТ**

**ПО ПРАКТИЧЕСКИМ РАБОТАМ №1-№7**

**по дисциплине**

**«Программирование на языке Java»**

Выполнил студент группы ИВБО-06-21 Черных А.В. Принял ассистент кафедры ИиППО Шошников И.К.

Практические работы выполнены « » 2022г.

«Зачтено» « » 2022 г.

Москва 2022

# Практическая работа №1

## Постановка задачи

Цель данной практической работы – освоить на практике работу с классами на Java.

## Программный код

*Листинг 1 – основной код*

class Book {

private String title;

private int date\_of\_publication;

public Book() {

title = "unknown";

date\_of\_publication = 0;

}

public Book(String title) {

this.title = title;

date\_of\_publication = 0;

}

public Book(String title, int date\_of\_publication) {

this.title = title;

this.date\_of\_publication = date\_of\_publication;

}

public void setTitle(String title) {

this.title = title;

}

public void setDate\_of\_publication(int date\_of\_publication) {

this.date\_of\_publication = date\_of\_publication;

}

public int getDate\_of\_publication() {

return date\_of\_publication;

}

public String getTitle() {

return title;

}

public String toString(){

return this.title+" "+this.date\_of\_publication;

}

}

public class TestBook {

public static void main(String[] args) {

Book book\_1 = new Book();

Book book\_2 = new Book("LOL");

Book book\_3 = new Book("NOT LOL", 1965);

System.out.println(book\_1);

System.out.println(book\_2);

System.out.println(book\_3);

}

}

## Вывод программы

*Листинг 2 — Вывод*

unknown 0

LOL 0

NOT LOL 1965

# Практическая работа №2

## Постановка задачи

Научиться писать программы по UML - диаграммам

## Программный код

*Листинг 3 – основной код*

class Author {

private String name;

private String email;

private char gender = 'U';

public Author() {}

public Author(String name, String email, char gender) {

this.name = name;

this.email = email;

this.gender = gender;

}

public String getName() {

return name;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public char getGender() {

return gender;

}

public String toString() {

return "Автор - "+ this.name + ", почта - " + this.email+ ", пол - "+ this.gender;

}

}

public class TestAuthor {

public static void main(String[] args) {

Author author\_1 = new Author("Gogol","gogol@gmail.com",'m');

Author author\_2 = new Author("Chekhov","chekhov@gmail.com",'m');

Author author\_3 = new Author("Tolstoy","tolostoy@gmail.com",'m');

System.out.println(author\_1);

System.out.println(author\_2);

System.out.println(author\_3);

}

}

## Вывод программы

*Листинг 4 — Вывод*

Автор - Gogol, почта - gogol@gmail.com, пол - m

Автор - Chekhov, почта - chekhov@gmail.com, пол - m

Автор - Tolstoy, почта - tolostoy@gmail.com, пол - m

# Практическая работа №3

## Постановка задачи

Напишите суперкласс с и его наследников так как показано на диаграмме в методичке к данному заданию.

## Программный код

*Листинг 5 – код абстрактного класса Shape*

import static java.lang.Math.PI;

abstract class Shape {

protected String color;

protected boolean filled;

public Shape(){};

public Shape(String color, boolean filled) {

this.color = color;

this.filled = filled;

};

public String getColor() {

return color;

}

public boolean isFilled() {

return filled;

}

public void setColor(String color) {

this.color = color;

}

public void setFilled(boolean filled) {

this.filled = filled;

}

public abstract double getArea();

public abstract double getPerimeter();

@Override

public String toString() {

return "Shape{" +

"color='" + color + '\'' +

", filled=" + filled +

'}';

}

}

*Листинг 6 – код наследника Rectangle абстрактного класса Shape*

class Rectangle extends Shape {

protected double width;

protected double length;

Rectangle(double width,double length){

this.length = length;

this.width = width;

};

Rectangle(double width,double length,String color, boolean filled){

super(color, filled);

this.length = length;

this.width = width;

};

public void setLength(double length) {

this.length = length;

};

public double getLength() {return length;}

public void setWidth(double width) {

this.width = width;

}

public double getWidth() {return width;}

@Override

public double getArea(){

return width\*length;

};

@Override

public double getPerimeter(){

return 2\*width+2\*length;

};

@Override

public String toString() {

return "Rectangle{" +

"color='" + color + '\'' +

", filled=" + filled +

", width=" + width +

", length=" + length +

'}';

}

}

*Листинг 7 – код наследников абстрактного класса Shape и класса Rectangle*

class Square extends Rectangle{

Square(double side){

super(side,side);

}

Square(double side,String color, boolean filled){

super(side,side,color, filled);

}

public double getSide(){return width;}

public void setSide(double side){

this.width = side;

this.length = side;

}

@Override

public String toString() {

return "Square{" +

"color='" + color + '\'' +

", filled=" + filled +

", side=" + width +

'}';

}

}

class Circle extends Shape {

protected double radius;

Circle(double radius){

this.radius = radius;

};

Circle(double radius,String color, boolean filled){

super(color, filled);

this.radius = radius;

};

public void setRadius(double radius) {

this.radius = radius;

}

public double getRadius() {return radius;}

@Override

public double getArea(){return PI\*radius\*radius;};

@Override

public double getPerimeter(){return 2\*PI\*radius;};

@Override

public String toString() {

return "Circle{color='" + color + '\'' +

", filled=" + filled + ", radius=" + radius +'}';

}

}

*Листинг 8 – основной алгоритм программы*

public class Main {

public static void main(String[] args) {

Circle circle = new Circle(14, "green",true);

Square square = new Square(13,"yellow", false);

System.out.println(circle);

System.out.println(square);

}

}

## Вывод программы

*Листинг 9 — Вывод*

Circle{color='green', filled=true, radius=14.0}

Square{color='yellow', filled=false, side=13.0}

# Практическая работа №4

## Постановка задачи

Напишите интерактивную программу с использованием GUI имитирует таблицу результатов матчей между командами Милан и Мадрид. Создайте JFrame приложение у которого есть следующие компоненты GUI:

• одна кнопка JButton labeled “AC Milan”

• другая JButton подписана “Real Madrid”

• надпись JLabel содержит текст “Result: 0 X 0”

• надпись JLabel содержит текст “Last Scorer: N/A”

• надпись Label содержит текст “Winner: DRAW”;

Всякий раз, когда пользователь нажимает на кнопку AC Milan, результат будет увеличиваться для Милана, сначала 1 X 0, затем 2 X 0 и так далее. Last Scorer означает последнюю забившую команду. В этом случае: AC Milan. Если пользователь нажимает кнопку для команды Мадрид, то счет приписывается ей. Победителем становится команда, которая имеет больше кликов кнопку на соответствующую, чем друга

## Программный код

*Листинг 10 – код абстрактного класса Shape*

import java.awt.\*;

import javax.swing.\*;

class TextAreaExample extends JFrame {

JTextArea jta1 = new JTextArea(10, 25);

JButton buttonACMilan = new JButton("AC Milan");

JButton buttonRealMadrid = new JButton("Real Madrid");

JLabel jLabelResult = new JLabel("Result: 0 X 0");

JLabel jLabelLastScorer = new JLabel("Last Scorer: N/A");

JLabel jLabelWinner = new JLabel("Winner: DRAW");

int scoreMadrid = 0, scoreMilan = 0;

public TextAreaExample() {

super("Example");

setSize(300, 300);

setLayout(new FlowLayout());

add(buttonACMilan);

add(buttonRealMadrid);

add(jLabelResult);

add(jLabelLastScorer);

add(jLabelWinner);

buttonACMilan.addActionListener(ae -> {

scoreMilan++;

jLabelResult.setText("Result: " + scoreMilan + " X "+scoreMadrid);

jLabelLastScorer.setText("Last Scorer: AC MILAN");

if (scoreMadrid == scoreMilan{jLabelWinner.setText("Winner: DRAW");

} elseif(scoreMadrid>scoreMilan{jLabelWinner.setText("Winner:Real Madrid");

} else {jLabelWinner.setText("Winner: AC Milan");}

});

buttonRealMadrid.addActionListener(ae -> {

scoreMadrid++;

jLabelResult.setText("Result: " + scoreMilan + " X " + scoreMadrid);

jLabelLastScorer.setText("Last Scorer: Real Madrid");

if (scoreMadrid == scoreMilan) {

jLabelWinner.setText("Winner: DRAW");

} else if (scoreMadrid > scoreMilan) {

jLabelWinner.setText("Winner: Real Madrid");

} else {

jLabelWinner.setText("Winner: AC Milan");

}

});

}

}

public class TestBall {

public static void main(String[] args) {

new TextAreaExample().setVisible(true);

}

}

*Листинг 11 – основной алгоритм программы*

public class Main {

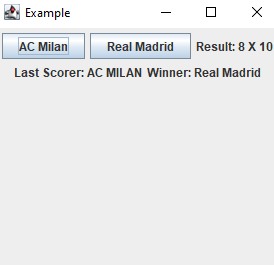
public static void main(String[] args) {

new TextAreaExample().setVisible(true)

}

}

## Вывод программы

**

**Рис. 1 – Программа имитирующая таблицу результатов**

# Практическая работа №5

## Постановка задачи

Решить задачи 13, 14, 15 пятой практической работы из методички.

## Программные коды

*Листинг 12 – код к 13 задаче*

import java.util.Scanner;

public class Main {

public static void lol() {

Scanner scanner = new Scanner(System.in);

int x = 1;

x = scanner.nextInt();

if (x == 0) return;

System.out.println(x);

x = scanner.nextInt();

if (x == 0) return;

lol();

}

public static void main(String[] args) {

lol();

}

}

*Листинг 13 – код к 14 задаче*

import java.util.Scanner;

public class Main {

static int num = 0;

public static void lol() {

if(num==0){return;}

int i = num;

num = num / 10;

lol();

System.out.print(i%10+" ");

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

num = scanner.nextInt();

lol();

}

}

*Листинг 14 – код к 15 задаче*

import java.util.Scanner;

public class Main {

static long num = 0;

public static void lol() {

if(num==0){return;}

System.out.print(num%10+" ");

num = num / 10;

lol();

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

num = scanner.nextInt();

lol();

}

}

## Ввод программ

*Листинг 15 — Ввод к 13 задаче*

1

2

3

4

5

6

*Листинг 16 — Ввод к 14 и 15 задачам*

123456

## Вывод программ

*Листинг 17 — Вывод в 13 задаче*

1

3

5

*Листинг 18 — Вывод в 14 задаче*

1 2 3 4 5 6

*Листинг 20 — Вывод в 15 задаче*

6 5 4 3 2 1

# Практическая работа №6

## Постановка задачи

Напишите класс SortingStudentsByGPA который реализует интерфейс Comparator таким образом, чтобы сортировать список студентов по их итоговым баллам в порядке убывания с использованием алгоритма быстрой сортировки.

## Программный код

*Листинг 21 – код базового класса*

import java.util.ArrayList;

import java.util.Comparator;

class Student {

private int id;

public Student(int id){

this.id = id;

}

public int getId() {return id;}

}

*Листинг 22 – основной алгоритм программы*

public class Main {

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

static SortingStudentsByGPA sortingStudentsByGPA=new SortingStudentsByGPA();

public static void main(String[] args) {

students.add(new Student(42));

students.add(new Student(76));

students.add(new Student(1001));

students.add(new Student(100));

students.add(new Student(129));

students.add(new Student(128));

sortingStudentsByGPA.quickSort(students, 0, students.size() - 1);

for (int i = students.size()-1 ;i>=0;i--) {

System.out.println(students.get(i).getId());

}

}

}

*Листинг 23 – код класса SortingStudenstByGBA*

class SortingStudentsByGPA implements Comparator<Student> {

@Override

public int compare(Student o1, Student o2) {

if(o1.getId()==o2.getId()){

return 0;

}else if(o1.getId()>o2.getId()){

return -1;

} else {return 1;}

}

public void quickSort(ArrayList<Student> students, int low, int high) {

if (students.size() == 0)

return;

if (low >= high)

return;

int middle = low + (high - low) / 2;

Student opora = students.get(middle);

int i = low, j = high;

while (i <= j) {

while (compare(students.get(i), opora) > 0) {i++;}

while (compare(students.get(j), opora) < 0) {j--;}

if (i <= j) {//меняем местами

Student temp = students.get(i);

students.set(i, students.get(j));

students.set(j, temp);

i++;

j--;

}

}

if (low < j)

quickSort(students, low, j);

if (high > i)

quickSort(students, i, high);

}

}

## Вывод программы

*Листинг 24 — Вывод*

1001

129

128

100

76

42

# Практическая работа №7

## Постановка задачи

Использовать для организации хранения структуру данных Stack

## Программный код

*Листинг 25 – основной код*

import java.util.Stack;

public class Main {

public static String work(int[] word) {

StringBuilder res = new StringBuilder();

Stack<Integer> wordArray = new Stack<>();

for (int j = 0; j<word.length; j++) {

if (word[j] > 0) {

wordArray.add(word[j]);

res.append("0");

} else if (word[j] == 0) {

if (wordArray.isEmpty()) {

res.append("-1");

} else {

res.append((int) wordArray.pop());

}

}

if (word.length!=1 && j!=word.length-1)

res.append(",");

}

return res.toString();

}

public static void main(String[] args) {

int[] word = new int[]{0, 1, 8, 4, 0, 3, 0, 8, 8, 3, 7};

System.out.println(work(word));

}

}

## Вывод программы

*Листинг 26 — Вывод*

-1,0,0,0,4,0,3,0,0,0,0

# Вывод

В ходе данных практических работ были получены практические навыки разработки программ, изучен синтаксис языка Java, были освоены основные конструкции языка Java и навыки осуществления стандартного потока ввода/вывода данных.