|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

**ОТЧЕТ**

**по лабораторной работе № 3**

|  |  |
| --- | --- |
| **Название:** | Классы, наследование, полиморфизм |
| **Дисциплина:** | Языки программирования для работы с большими данными |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-22М |  | 17.03.2023 | И.Б. Нуриддинов |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П.В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

Цель работы: Изучение и практическое освоение концепций классов, наследования и полиморфизма в языке программирования Java.

*Вариант 1. Задание 1.* Определить класс Матрица размерности (n x n). Класс должен содержать несколько конструкторов. Реализовать методы для сложения, вычитания, умножения матриц. Объявить массив объектов. Создать методы, вычисляющие первую и вторую нормы матрицы



public class Matrix {

private int size;

private int[][] array;

public Matrix(int size) {

this.size = size;

array = new int[size][size];

Random rand = new Random();

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++) {

this.array[i][j] = rand.nextInt(10) - 5;

}

}

}

public Matrix(int size, int fill){

this.size = size;

array = new int[size][size];

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++) {

this.array[i][j] = fill;

}

}

}

public Matrix add (Matrix obj){

if (obj.size != this.size){

System.out.println("Matrices with difference size.");

return this;

}

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++) {

this.array[i][j] += obj.array[i][j];

}

}

return this;

}

public Matrix sub(Matrix obj){

if (obj.size != this.size){

System.out.println("Matrices with difference size.");

return this;

}

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++) {

this.array[i][j] -= obj.array[i][j];

}

}

return this;

}

public Matrix multiply(Matrix obj){

if (obj.size != this.size){

System.out.println("Matrices with difference size.");

return this;

}

int[][] tmp = new int[size][size];

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++) {

tmp[i][j] = this.array[i][j];

}

}

int sum = 0;

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++) {

sum = 0;

for (int k = 0; k < size; k++) {

sum += tmp[i][k] \* obj.array[k][j];

}

this.array[i][j] = sum;

}

}

return this;

}

public int normFirst(){

int max = 0;

int sum = 0;

for (int i = 0; i < size; i++) {

max += this.array[0][i];

}

for (int i = 1; i < size; i++) {

sum = 0;

for (int j = 0; j < size; j++) {

sum += Math.abs(this.array[i][j]);

}

max = (sum > max) ? sum : max;

}

return max;

}

public int normSecond(){

int max = 0;

int sum = 0;

for (int i = 0; i < size; i++) {

max += this.array[i][0];

}

for (int i = 1; i < size; i++) {

sum = 0;

for (int j = 0; j < size; j++) {

sum += Math.abs(this.array[j][i]);

}

max = (sum > max) ? sum : max;

}

return max;

}

public static int minNormFirst(Matrix[] objs){

int min = objs[0].normFirst();

int pos = 0;

int tmp;

for (int i = 1; i < objs.length; i++) {

if ((tmp = objs[i].normFirst()) < min){

min = tmp;

pos = i;

}

}

return pos;

}

public static int minNormSecond(Matrix[] objs){

int min = objs[0].normSecond();

int pos = 0;

int tmp;

for (int i = 1; i < objs.length; i++) {

if ((tmp = objs[i].normSecond()) < min){

min = tmp;

pos = i;

}

}

return pos;

}

public void print(){

for (int i = 0; i < size; i++) {

System.out.println(Arrays.toString(array[i]));

}

}

}

*Вариант 2. Задание 1.* Создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль.

Abiturient: id, Фамилия, Имя, Отчество, Адрес, Телефон, Оценки. Создать массив объектов. Вывести: a) список абитуриентов, имеющих неудовлетворительные оценки; b) список абитуриентов, средний балл у которых выше заданного; c) выбрать заданное число n абитуриентов, имеющих самый высокий средний балл (вывести также полный список абитуриентов, имеющих полупроходной балл).

public class Abiturient {

private long id;

private String name;

private String surname;

private String patronymic;

private String address;

private String phone;

private int[] marks;

private static long counter = 0;

public Abiturient(String name, String surname, String patronymic,

String address, String phone, int[] marks){

this.id = nextId();

this.name = name;

this.surname = surname;

this.patronymic = patronymic;

this.address = address;

this.phone = phone;

this.marks = new int[marks.length];

for (int i = 0; i < marks.length; i++) {

this.marks[i] = marks[i];

}

}

public Abiturient(String name, String surname, int marksCount){

this.id = nextId();

this.name = name;

this.surname = surname;

this.marks = new int[marksCount];

Random rand = new Random();

for (int i = 0; i < marksCount; i++) {

this.marks[i] = rand.nextInt(4) + 1;

}

}

private long nextId(){

return ++counter;

}

@Override

public String toString() {

return "Abiturient{" +

", name='" + name + '\'' +

", surname='" + surname + '\'' +

", patronymic='" + patronymic + '\'' +

", address='" + address + '\'' +

", phone='" + phone + '\'' +

", marks=" + Arrays.toString(marks) +

'}';

}

public long getId() {

return id;

}

public String getName() {

return name;

}

public String getSurname() {

return surname;

}

public String getPatronymic() {

return patronymic;

}

public String getFullName(){

return name + " " + surname + " " + patronymic;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public String getPhone() {

return phone;

}

public void setPhone(String phone) {

this.phone = phone;

}

public int[] getMarks() {

return marks;

}

public void setMarks(int[] marks) {

this.marks = marks;

}

public boolean hasPoorGrade(){

for (int i = 0; i < marks.length; i++) {

if (marks[i] < 4 && marks[i] > 0)

return true;

}

return false;

}

public float getGPA(){

float gpa = 0.0F;

for (int i = 0; i < marks.length; i++) {

gpa += marks[i];

}

gpa /= marks.length;

return (float)(Math.round(gpa \* 100)) / 100;

}

public boolean isGPAAbove(float gpa){

if (gpa <= 0){

System.out.println("Ошибка: Заданный средний балл ниже либо равно нулю.");

return false;

}

return (this.getGPA() >= gpa) ? true : false;

}

}

Вывод: Лабораторная работа позволила освоить ключевые концепции классов, наследования и полиморфизма в языке программирования Java. Правильное использование этих концепций позволяет создавать гибкие, модульные и легко расширяемые программы. Наследование позволяет строить иерархии классов, повторно использовать код и реализовывать специализированные классы на основе общих абстракций. Полиморфизм позволяет обрабатывать объекты разных классов с помощью общих интерфейсов и упрощает разработку кода.