Примерни казуси

1 – тема  
  
import java.util.Scanner;

public class Main {

public static double add(double num1, double num2) {

return num1 + num2;

}

public static double subtract(double num1, double num2) {

if(num1 > num2) {

return num1 - num2;

} else {

return num2 - num1;

}

}

public static double multiply(double num1, double num2) {

return num1 \* num2;

}

public static double divide(double num1, double num2) {

return num2 / num1;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

double num1 = sc.nextDouble();

double num2 = sc.nextDouble();

Тема – 2   
  
public class Main {

public static boolean isMagicSquare(int[][] matrix, int n) {

int sum1 = 0, sum2 = 0;

// finding the sum of first diagonal

for (int i = 0; i < n; i++) sum1 = sum1 + matrix[i][i];

// finding the sum of seciond diagonal

for (int i = 0; i < n; i++) sum2 = sum2 + matrix[i][n - 1 - i];

// Checking, whether the sum of two diagonal is same or not

if (sum1 != sum2) return false;

// finding the sum of each row

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

int rowSum = 0;

for (int j = 0; j < n; j++) rowSum = rowSum + matrix[i][j];

// checking, whether eachh row sum is equal to diagonal sum or not

if (rowSum != sum1) return false;

}

// finding the sum of each column

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

int columnSum = 0;

for (int j = 0; j < n; j++) columnSum += matrix[j][i];

// checking, whether each column sum is equal to diagonal sum or not

if (columnSum != sum1) return true;

}

return true;

}

public static void main(String[] args) {

// Test when the matrix IS a magical square

// Declare and initialize the 2d array

int[][] arr2d = new int[4][];

arr2d[0] = new int[] {16, 3, 2, 13};

arr2d[1] = new int[] {5, 10, 11, 8

Тема – 3

public class Student {

private String name;

private double physicsMarks;

private double chemistryMarks;

private double biologyMarks;

public Student(String name, double physicsMarks, double chemistryMarks, double biologyMarks) {

this.name = name;

this.physicsMarks = physicsMarks;

this.chemistryMarks = chemistryMarks;

this.biologyMarks = biologyMarks;

}

public String getName() {

return name;

}

public double getPhysicsMarks() {

return physicsMarks;

}

public double getChemistryMarks() {

return chemistryMarks;

}

public double getBiologyMarks() {

return biologyMarks;

}

Тема – 4

// Абстрактен клас Book

abstract class Book {

// Private полета

private String name;

private String author;

private String price;

protected String getName(){

return this.name;

}

protected String getAuthor(){

return this.author;

}

protected String getPrice(){

return this.price;

}

// Параметризиран конструктор

public Book(String name, String author, String price) {

this.name = name;

this.author = author;

this.price = price;

}

// Абстрактен метод

public abstract String getDetails();

}

// Класът MyBook наследява класа Book

class MyBook extends Book {

// Параметризиран конструктор

public MyBook(String name, String author, String price) {

super(name, author, price);

}

// Презаписване на абстрактния метод getDetails на базовия клас

public String getDetails() {

return getName() + ", " + getAuthor() + ", " + getPrice();

}

}

public class Demo {

public static void main(String[] args) {

Book myBook = new MyBook("Harry Potter", "J.k. Rowling", "100");

System.out.println(myBook.getDetails());

}

}

Тема – 5

import java.util.Scanner;

public class Main {

public static String bubbleSortString(String sentence) {

int n = sentence.length();

StringBuilder sb = new StringBuilder(sentence);

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

for (int j = 0; j < n - i - 1; j++) {

if (sb.charAt(j) > sb.charAt(j + 1)) {

char temp = sb.charAt(j);

sb.setCharAt(j, sb.charAt(j + 1));

sb.setCharAt(j + 1, temp);

}

}

}

return sb.toString();

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String sentence = sc.nextLine();

System.out.println(bubbleSortString(sentence));

}

}

Тема – 6

|  |
| --- |
| **Заявка 1** |
| SELECT TOP 5  e.FirstName, e.HireDate  FROM Employees AS e  WHERE e.HireDate > '06-02-2000'  ORDER BY e.FirstName ASC |

|  |
| --- |
| **Заявка 2** |
| SELECT CONCAT(FirstName,'.', LastName,'@bulgaria.bg') AS Emails  FROM Employees  ORDER BY FirstName ASC |

|  |
| --- |
| **Заявка 3** |
| SELECT TOP 10  e.[FirstName],  e.[LastName],  t.[Name] AS [Town],  a.[AddressText] AS [Address]  FROM Employees AS e  INNER JOIN Addresses AS a  ON e.AddressID = a.AddressID  INNER JOIN Towns AS t  ON a.TownID = t.TownID  ORDER BY e.FirstName ASC |

|  |
| --- |
| **Заявка 4** |
| CREATE PROCEDURE sp\_GetEmployeeName  (  @EmployeeId INT  )  AS  BEGIN  SELECT FirstName  FROM Employees  WHERE EmployeeID = @EmployeeId;  END; |

Тема – 7

|  |
| --- |
| Class Publisher |
| public class Publisher {  private String name;  private String address;  private String phone;  private String url;  public Publisher() {  }  public Publisher(String name, String address, String phone, String url) {  this.name = name;  this.address = address;  this.phone = phone;  this.url = url;  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  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 String getUrl() {  return url;  }  public void setUrl(String url) {  this.url = url;  }  } |

|  |
| --- |
| Class Book |
| public class Book {  private String ISBN;  private String publisherName;  private String authorName;  private String authorAddress;  private int year;  private String title;  private double price;  public Book() {  }  public Book(String ISBN, String publisherName, String authorName, String authorAddress, int year, String title, double price) {  this.ISBN = ISBN;  this.publisherName = publisherName;  this.authorName = authorName;  this.authorAddress = authorAddress;  this.year = year;  this.title = title;  this.price = price;  }  public String getISBN() {  return ISBN;  }  public void setISBN(String ISBN) {  this.ISBN = ISBN;  }  public String getPublisherName() {  return publisherName;  }  public void setPublisherName(String publisherName) {  this.publisherName = publisherName;  }  public String getAuthorName() {  return authorName;  }  public void setAuthorName(String authorName) {  this.authorName = authorName;  }  public String getAuthorAddress() {  return authorAddress;  }  public void setAuthorAddress(String authorAddress) {  this.authorAddress = authorAddress;  }  public int getYear() {  return year;  }  public void setYear(int year) {  this.year = year;  }  public String getTitle() {  return title;  }  public void setTitle(String title) {  this.title = title;  }  public double getPrice() {  return price;  }  public void setPrice(double price) {  this.price = price;  }  } |

|  |
| --- |
| Class ShoppingBasket\_Book |
| public class ShoppingBasket\_Book {  private int shoppingBasketId;  private String bookSBN;  private int count;  public ShoppingBasket\_Book() {  }  public ShoppingBasket\_Book(int shoppingBasketId, String bookSBN, int count) {  this.shoppingBasketId = shoppingBasketId;  this.bookSBN = bookSBN;  this.count = count;  }  public int getShoppingBasketId() {  return shoppingBasketId;  }  public void setShoppingBasketId(int shoppingBasketId) {  this.shoppingBasketId = shoppingBasketId;  }  public String getBookSBN() {  return bookSBN;  }  public void setBookSBN(String bookSBN) {  this.bookSBN = bookSBN;  }  public int getCount() {  return count;  }  public void setCount(int count) {  this.count = count;  }  } |

|  |
| --- |
| Class ShoppingBasket |
| public class ShoppingBasket {  private int id;  private String customerEmail;  public ShoppingBasket() {  }  public ShoppingBasket(int id, String customerEmail) {  this.id = id;  this.customerEmail = customerEmail;  }  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public String getCustomerEmail() {  return customerEmail;  }  public void setCustomerEmail(String customerEmail) {  this.customerEmail = customerEmail;  }  } |

|  |
| --- |
| Class Warehouse\_Book |
| public class Warehouse\_Book {  private String warehouseCode;  private String bookISBN;  private int count;  public Warehouse\_Book() {  }  public Warehouse\_Book(String warehouseCode, String bookISBN, int count) {  this.warehouseCode = warehouseCode;  this.bookISBN = bookISBN;  this.count = count;  }  public String getWarehouseCode() {  return warehouseCode;  }  public void setWarehouseCode(String warehouseCode) {  this.warehouseCode = warehouseCode;  }  public String getBookISBN() {  return bookISBN;  }  public void setBookISBN(String bookISBN) {  this.bookISBN = bookISBN;  }  public int getCount() {  return count;  }  public void setCount(int count) {  this.count = count;  }  } |

|  |
| --- |
| Class Customer |
| public class Customer {  private String email;  private String name;  private String phone;  private String address;  public Customer() {  }  public Customer(String email, String name, String phone, String address) {  this.email = email;  this.name = name;  this.phone = phone;  this.address = address;  }  public String getEmail() {  return email;  }  public void setEmail(String email) {  this.email = email;  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public String getPhone() {  return phone;  }  public void setPhone(String phone) {  this.phone = phone;  }  public String getAddress() {  return address;  }  public void setAddress(String address) {  this.address = address;  }  } |

|  |
| --- |
| Class Warehouse |
| public class Warehouse {  private String code;  private String phone;  private String address;  public Warehouse() {  }  public Warehouse(String code, String phone, String address) {  this.code = code;  this.phone = phone;  this.address = address;  }  public String getCode() {  return code;  }  public void setCode(String code) {  this.code = code;  }  public String getPhone() {  return phone;  }  public void setPhone(String phone) {  this.phone = phone;  }  public String getAddress() {  return address;  }  public void setAddress(String address) {  this.address = address;  }  } |

Тема – 8

import org.junit.Test;

import static org.junit.Assert.\*;

public class BankAccountTests {

@Test

public void testDeposit() {

BankAccount account = new BankAccount("John Doe", "1234567890", 1000.00);

double amount = 500.00;

account.deposit(amount);

assertEquals(1500.00, account.getBalance(), 0.001);

}

@Test

public void testWithdrawSufficientBalance() {

BankAccount account = new BankAccount("John Doe", "1234567890", 1000.00);

double amount = 500.00;

account.withdraw(amount);

assertEquals(500.00, account.getBalance(), 0.001);

}

@Test

public void testWithdrawInsufficientBalance() {

BankAccount account = new BankAccount("John Doe", "1234567890", 1000.00);

double amount = 1500.00;

account.withdraw(amount);

assertEquals(1000.00, account.getBalance(), 0.001);

}

@Test

public void testAccountHolderName() {

BankAccount account = new BankAccount("John Doe", "1234567890", 1000.00);

String newName = "Jane Smith";

account.setAccountHolderName(newName);

assertEquals("Jane Smith", account.getAccountHolderName());

}

@Test

public void testAccountNumber() {

BankAccount account = new BankAccount("John Doe", "1234567890", 1000.00);

String newNumber = "0987654321";

account.setAccountNumber(newNumber);

assertEquals("0987654321", account.getAccountNumber());

}

}

Тем – 8

import java.util.List;

import javax.persistence.\*;

public class Program {

public static void main(String[] args) {

// Create a new instance of the school database context

try (var db = new SchoolContext()) {

// Query the students table and print the results

List<Student> students = db.getStudents();

System.out.println("Students:");

for (Student student : students) {

System.out.println("- " + student.getFirstName() + " " + student.getLastName());

}

// Query the courses table and print the results

List<Course> courses = db.getCourses();

System.out.println("\nCourses:");

for (Course course : courses) {

System.out.println("- " + course.getCourseName() + " (" + course.getCourseCode() + ")");

}

}

System.out.println("\nPress any key to exit...");

}

}

// Define the school database context using Entity Framework

@Entity

public class SchoolContext {

@Id

private int Id;

@OneToMany

private List<Student> Students;

@OneToMany

private List<Course> Courses;

public List<Student> getStudents() {

return Students;

}

public List<Course> getCourses() {

return Courses;

}

public void setStudents(List<Student> students) {

Students = students;

}

public void setCourses(List<Course> courses) {

Courses = courses;

}

@PersistenceContext

protected void onConfiguring(EntityManagerFactory entityManagerFactory) {

EntityManager entityManager = entityManagerFactory.createEntityManager();

}

}

// Define the Student and Course models

@Entity

public class Student {

@Id

private int Id;

private String FirstName;

private String LastName;

private int Age;

public String getFirstName() {

return FirstName;

}

public String getLastName() {

return LastName;

}

public int getAge() {

return Age;

}

public void setFirstName(String firstName) {

FirstName = firstName;

}

public void setLastName(String lastName) {

LastName = lastName;

}

public void setAge(int age) {

Age = age;

}

}

@Entity

public class Course {

@Id

private int Id;

private String CourseName;

private String CourseCode;

private int CreditHours;

public String getCourseName() {

return CourseName;

}

public String getCourseCode() {

return CourseCode;

}

public int getCreditHours() {

return CreditHours;

}

public void setCourseName(String courseName) {

CourseName = courseName;

}

public void setCourseCode(String courseCode) {

CourseCode = courseCode;

}

public void setCreditHours(int creditHours) {

CreditHours = creditHours;

}

}

Тема – 9

import java.util.List;

import javax.persistence.\*;

public class Program {

public static void main(String[] args) {

// Create a new instance of the school database context

try (var db = new SchoolContext()) {

// Query the students table and print the results

List<Student> students = db.getStudents();

System.out.println("Students:");

for (Student student : students) {

System.out.println("- " + student.getFirstName() + " " + student.getLastName());

}

// Query the courses table and print the results

List<Course> courses = db.getCourses();

System.out.println("\nCourses:");

for (Course course : courses) {

System.out.println("- " + course.getCourseName() + " (" + course.getCourseCode() + ")");

}

}

System.out.println("\nPress any key to exit...");

}

}

// Define the school database context using Entity Framework

@Entity

public class SchoolContext {

@Id

private int Id;

@OneToMany

private List<Student> Students;

@OneToMany

private List<Course> Courses;

public List<Student> getStudents() {

return Students;

}

public List<Course> getCourses() {

return Courses;

}

public void setStudents(List<Student> students) {

Students = students;

}

public void setCourses(List<Course> courses) {

Courses = courses;

}

@PersistenceContext

protected void onConfiguring(EntityManagerFactory entityManagerFactory) {

EntityManager entityManager = entityManagerFactory.createEntityManager();

}

}

// Define the Student and Course models

@Entity

public class Student {

@Id

private int Id;

private String FirstName;

private String LastName;

private int Age;

public String getFirstName() {

return FirstName;

}

public String getLastName() {

return LastName;

}

public int getAge() {

return Age;

}

public void setFirstName(String firstName) {

FirstName = firstName;

}

public void setLastName(String lastName) {

LastName = lastName;

}

public void setAge(int age) {

Age = age;

}

}

@Entity

public class Course {

@Id

private int Id;

private String CourseName;

private String CourseCode;

private int CreditHours;

public String getCourseName() {

return CourseName;

}

public String getCourseCode() {

return CourseCode;

}

public int getCreditHours() {

return CreditHours;

}

public void setCourseName(String courseName) {

CourseName = courseName;

}

public void setCourseCode(String courseCode) {

CourseCode = courseCode;

}

public void setCreditHours(int creditHours) {

CreditHours = creditHours;

}

}

Тема – 10

Изведете информация от файла за памет с произволен достъп (RAM). (Ако използвате WSL система, командата няма да сработи, но ако отговорът ви е коректен, ще бъде зачетен.)

2. Инсталирайте apache2.

3. Покажете с кой PID e стартиран apache2

4. Актуализирайте всички пакети на вашата Ubuntu/Debian система.

5. Деинсталирайте от хранилището(package manager) apache2.

6. Покажете IP на вашата система.

7. Напишете скрипт който да извежда на конзолата “Hello, World!” след като стартирате файла sh start.sh

8. Добавете в скрипта от точка 7 променлива “ANIMAL” и я инициализирайте с “dog”.  
  
Тема – 11

import java.util.Arrays;

import java.util.List;

import java.util.Scanner;

import java.util.concurrent.\*;

import java.util.stream.Collectors;

public class Main {

public static void main(String[] args) throws InterruptedException, ExecutionException {

Scanner scn = new Scanner(System.in);

String line = "1 2 3 4 5 6 7 8 9 10"; // scn.nextLine();

List<Integer> array = Arrays.stream(line.split(" "))

.map(Integer::parseInt).toList();

final double[] results = new double[2];

var pool = Executors.newFixedThreadPool(2);

int middle = array.size() / 2;

var fu1 = pool.submit(() -> {

double sum = 0;

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

sum += array.get(i);

}

results[0] = sum / middle;

});

var fu2 = pool.submit(() -> {

double sum = 0;

for (int i = middle; i < array.size(); i++) {

sum += array.get(i);

}

results[1] = sum / (array.size() - middle);

});

fu1.get();

fu2.get();

double average = (results[0] + results[1]) / 2;

System.out.println("The average of the array is: " + String.format("%.2f", average));

}

}

Тема – 12

import java.util.Arrays;

import java.util.List;

import java.util.Scanner;

import java.util.concurrent.\*;

import java.util.stream.Collectors;

public class Main {

public static void main(String[] args) throws InterruptedException, ExecutionException {

Scanner scn = new Scanner(System.in);

String line = "1 2 3 4 5 6 7 8 9 10"; // scn.nextLine();

List<Integer> array = Arrays.stream(line.split(" "))

.map(Integer::parseInt).toList();

final double[] results = new double[2];

var pool = Executors.newFixedThreadPool(2);

int middle = array.size() / 2;

var fu1 = pool.submit(() -> {

double sum = 0;

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

sum += array.get(i);

}

results[0] = sum / middle;

});

var fu2 = pool.submit(() -> {

double sum = 0;

for (int i = middle; i < array.size(); i++) {

sum += array.get(i);

}

results[1] = sum / (array.size() - middle);

});

fu1.get();

fu2.get();

double average = (results[0] + results[1]) / 2;

System.out.println("The average of the array is: " + String.format("%.2f", average));

}

}

Тема – 13

import java.util.Scanner;

class Main {

  public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

    double[][] a = new double[3][4];

    double[] x = new double[3];

    double ratio;

    int i, j, k, n = 3;

    for (i = 0; i < n; i++) {

      for (j = 0; j < n + 1; j++) {

        a[i][j] = sc.nextDouble();

      }

    }

    for (i = 0; i < n - 1; i++) {

      if (a[i][i] == 0.0) {

        System.out.println("Mathematical Error!");

        return;

      } else {

        for (j = i + 1; j < n; j++) {

          ratio = a[j][i] / a[i][i];

          for (k = 0; k < n + 1; k++) {

            a[j][k] = a[j][k] - ratio \* a[i][k];

          }

        }

      }

    }

    // Obtaining Solution by Back Substitution Method

    x[n - 1] = a[n - 1][n] / a[n - 1][n - 1];

    for (i = n - 2; i >= 0; i--) {

      x[i] = a[i][n];

      for (j = i + 1; j < n; j++) {

        x[i] = x[i] - a[i][j] \* x[j];

      }

      x[i] = x[i] / a[i][i];

    }

    // Displaying Solution

    System.out.print("Solution: ");

    for (i = 0; i < n; i++) {

     System.out.printf("%.0f ", x[i]);

    }

  }

}

Тема – 14

|  |
| --- |
| <div id="myDiv">  <p>This is a paragraph.</p>  <p>This is another paragraph.</p>  </div> |

|  |
| --- |
| Action.js |
| let myDiv = document.getElementById("myDiv");  if(myDiv){  let paragraphs = myDiv.getElementsByTagName("p");  for (let i = 0; i < paragraphs.length; i++) {  paragraphs[i].style.color = "red";  }  }else{  console.error("Element with id 'myDiv' not found!");  } |

Тема – 15

@RestController

@RequestMapping("/api")

public class ExampleController {

@GetMapping("/greeting")

public String greeting() {

return "Hello World!";

}

@PostMapping("/add")

public int add(@RequestBody int a, @RequestBody int b) {

return a + b;

}

}

Тема – 16

|  |
| --- |
| Задача 1 |
| 1. ‘git unit’ 2. ‘git status’ 3. ‘git branch <branch>’ 4. ‘git marge <marge>’ 5. ‘git log’ 6. ‘git clone <repository>‘ |

|  |
| --- |
| Задача 2 |
| import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("Enter three numbers: ");  // Въвеждане на данни  int firstNum = scanner.nextInt();  int secondNum = scanner.nextInt();  int thirdNum = scanner.nextInt();  System.out.println("The smallest number is: " + findSmallestNumber(firstNum, secondNum, thirdNum));  }  // Метод намиращ най-малкото число  public static int findSmallestNumber(int a, int b, int c) {  int smallest = a;  // Проверка две числа  if(smallest > b) {  smallest = b;  }  if(smallest > c) {  smallest = c;  }  return smallest;    // Връща резултат с най-малкото число  }  } |

|  |
| --- |
| Задача 3 |
| import static org.junit.Assert.\*;  import org.junit.Test;  public class ProgramTest {  @Test  public void testAdd() {  int num1 = 3;  int num2 = 5;  int expectedResult = 8;  int actualResult = Program.add(num1, num2);  assertEquals(expectedResult, actualResult);  }  } |

Тема – 17

import Data.List

import Data.Array

import System.CPUTime

myArray = array (1, 3) [(1, "a"), (2, "b"), (3, "c")]

list = [1, 2, 3, 4, 5]

main = do

print cpuTimePrecision

time <- getCPUTime

putStrLn $ "CPU time: " ++ show time

print myArray

print $ myArray ! 2

putStrLn ""

print list

print $ head list

print $ tail list

print $ last list

print $ init list

Тема – 18

|  |
| --- |
| FractionalKnapsack.java |
| import java.util.Arrays;  public class FractionalKnapsack {    static class Item implements Comparable<Item> {  int value, weight;  // Constructor  Item(int value, int weight) {  this.value = value;  this.weight = weight;  }  // Compare Item according to val/weight ratio  @Override  public int compareTo(Item other) {  double ratio1 = (double) value / weight;  double ratio2 = (double) other.value / other.weight;  if (ratio1 > ratio2) {  return -1;  } else if (ratio1 < ratio2) {  return 1;  } else {  return 0;  }  }  }  public static double fractionalKnapsack(int knapsackWeight, Item[] itemArray, int n) {  Arrays.sort(itemArray); //sort based on value/weight ratio  int currentWeight = 0;  double finalValue = 0.0;  for (int i = 0; i < n; i++) {  if (currentWeight + itemArray[i].weight <= knapsackWeight) {  currentWeight += itemArray[i].weight;  finalValue += itemArray[i].value;  } else {  int remain = knapsackWeight - currentWeight;  finalValue += itemArray[i].value \* ((double) remain / itemArray[i].weight);  break;  }  }  return finalValue;  }  } |

|  |
| --- |
| For Student -18 – Main.java |
| public class Main {  public static void main(String[] args) {  int knapsackWeight = 50;  FractionalKnapsack.Item[] itemArray = {new FractionalKnapsack.Item(120, 30), new FractionalKnapsack.Item(100, 20), new FractionalKnapsack.Item(60, 10)};  int n = itemArray.length;  System.out.println("Maximum value we can obtain = " + FractionalKnapsack.fractionalKnapsack(knapsackWeight, itemArray, n));  }  } |