

**ОТГОВОРИ, УПЪТВАНИЯ И ПРИМЕРНИ ИЗГЛЕДИ  
НА РЕШЕНИЯТА НА ПРАКТИЧЕСКИТЕ  
ЗАДАЧИ ОТ**

**ТЕМА 5**

<b>Въпрос</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>	<b>11</b>	<b>12</b>	<b>13</b>	<b>14</b>	<b>15</b>	<b>16</b>
<b>Отговор</b>	<b>В</b>	<b>Б</b>	<b>А</b>	<b>В</b>	<b>В</b>	<b>Б</b>	<b>А</b>	<b>Г</b>	<b>В</b>	<b>А</b>	<b>Б</b>	<b>Б</b>	<b>А</b>	<b>Б</b>	<b>А</b>	<b>А</b>

**17.**

<b>C#</b>
<pre>using System; using System.Collections.Generic; using System.Linq; class Program {     static void Main()     {         List&lt;int&gt; list = new List&lt;int&gt;() { 15, 8, 79, 3, 4, 9 };         list = list.OrderBy(x =&gt; -x).ToList();         Console.WriteLine(string.Join(", ", list));     } }</pre>
<b>Java</b>
<pre>import java.util.ArrayList; import java.util.Collections; import java.util.List; public class Main {     public static void main(String[] args) {         List&lt;Integer&gt; list = new ArrayList&lt;&gt;();         list.add(15);         list.add(8);         list.add(79);         list.add(3);         list.add(4);         list.add(9);         Collections.sort(list, Collections.reverseOrder());         System.out.println(String.join(", ", list.stream(). map(Object::toString).toArray(String[]::new)));     } }</pre>

**18.**

Колко е броят на операциите на изваждане и добавяне?

14

Колко са елементите в стека след изпълнението на операциите?

7 елемента

Кои са елементите на стека след изпълнението на операциите?

Елементите на стека след изпълнението на операциите са: 9, 6, 3, 0, -3, -6, -9.

19. Г

20.

CID	CTITLE	CDURATION
7890	DB2	5
8500	Oracle	5
8000	SQLServer	5
9000	SQL Workshop	3

21.

teacher	course	class
Иванов	информатика	11
Петрова	математика	10
Петрова	математика	11
Петрова	математика	12
Христова	Технологии	10
Каменова	литература	NULL

22.

<b>C#</b>
<pre>int n, tmp, reverse = 0; Console.Write("Въведете число: "); n = int.Parse(Console.ReadLine()); while (n != 0) {     tmp = n % 10;     reverse = reverse * 10 + tmp;     n = n/10; } Console.Write("Обърнато число: " + reverse);</pre>
<b>Java</b>
<pre>import java.util.Scanner; public class ReverseNumber {     public static void main(String[] args) {         int n, tmp, reverse = 0;         Scanner scanner = new Scanner(System.in);         System.out.print("въведете число: ");         n = scanner.nextInt();         while (n != 0) {             tmp = n % 10;             reverse = reverse * 10 + tmp;             n = n / 10;         }         System.out.println("обърнато число: " + reverse);         scanner.close();     } }</pre>

23.

**C#**

```
using System;
class Program
{
    static void Main()
    {
        int n;
        Console.Write("Въведете число: ");
        n = Convert.ToInt32(Console.ReadLine());
        for (int i = 1; i <= n; i++)
        {
            Console.WriteLine(i);
        }
    }
}
```

**Java**

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Число: ");
        int n = scanner.nextInt();
        for (int i = 1; i <= n; i++) {
            System.out.println(i);
        }
    }
}
```

24.

(1) 5 операции

(2) Стекът е празен след операциите.

(3) Елементите на опашката са 5, 6, 7, 8, 9 след операциите.

25.

Решение:

- Създаване на таблицата **Departments**

```
CREATE TABLE Departments (
    ID CHAR(3) PRIMARY KEY,
    Department_Name VARCHAR(100)
);
```

- Добавяне на записи в таблицата **Departments**

```
INSERT INTO Departments (ID, Department_Name) VALUES
('001', 'Компютърни системи и технологии'),
```

('002', 'Информационни и комуникационни технологии');

- Създаване на таблицата **Students**

```
CREATE TABLE Students (  
    Id CHAR(6) PRIMARY KEY,  
    Student_name VARCHAR(100),  
    Department_Id CHAR(3),  
    Results DECIMAL(5, 2),  
    Birth_date DATE,  
    FOREIGN KEY (Department_Id) REFERENCES Departments(ID)  
);
```

- Добавяне на записи в таблицата **Students**

```
INSERT INTO Students (Id, Student_name, Department_Id, Results, Birth_date)  
VALUES  
(,233023', 'Иван Петров Иванов', '001', 33.50, '2005-12-23'),  
(,233043', 'Алекс Драгомиров Колев', '001', 36.00, '2005-03-02'),  
(,233046', 'Ивайла Стоименова Николова', '002', 30.25, '2005-06-24');
```

- Информация за имената на студентите, специалността, за която кандидатстват, и бала им, сортирани по намаляващ ред на бала

```
SELECT s.Student_name, d.Department_Name, s.Results  
FROM Students s  
INNER JOIN Departments d ON s.Department_Id = d.ID  
ORDER BY s.Results DESC;
```

- Имената на студентите с най-висок бал

```
SELECT Student_name, Results  
FROM Students  
WHERE Results = (SELECT MAX(Results) FROM Students);
```

- Информация за имената на специалностите и броя на кандидатите във всяка от тях

```
SELECT d.Department_Name, COUNT(s.Id) AS Count_of_Students  
FROM Departments d  
LEFT JOIN Students s ON d.ID = s.Department_Id  
GROUP BY d.Department_Name;
```

## 26.

### C#

```
using System;
class Program
{
    static void Main()
    {
        try
        {
            Console.Write("Side a: ");
            double sideA = double.Parse(Console.ReadLine());

            Console.Write("Side b: ");
            double sideB = double.Parse(Console.ReadLine());

            double perimeter = 2 * (sideA + sideB);
            double area = sideA * sideB;

            Console.WriteLine($"P: {perimeter}");
            Console.WriteLine($"S: {area}");
        }
        catch (Exception)
        {
            Console.WriteLine("Something went wrong!");
        }
    }
}
```

### Java

```
import java.util.Scanner;
public class RectangleCalculator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        try {
            System.out.print("Side a: ");
            double sideA = scanner.nextDouble();

            System.out.print("Side b: ");
            double sideB = scanner.nextDouble();

            double perimeter = 2 * (sideA + sideB);
            double area = sideA * sideB;

            System.out.println("Perimeter: " + perimeter);
            System.out.println("Area: " + area);
        } catch (Exception e) {
            System.out.println("Something went wrong!");
        }
    }
}
```

**27. Примерно решение:**

```
1. CREATE TABLE Product_lines(  
Name VARCHAR(30) PRIMARY KEY,  
Description VARCHAR(256));
```

```
CREATE TABLE Products(  
Id INT PRIMARY KEY,  
Product_name VARCHAR(100) NOT NULL,  
Product_line_name VARCHAR(30) REFERENCES Product_lines(Name),  
Price DECIMAL(5,2),  
Quantity INT);
```

```
2. INSERT INTO Product_lines(Name,Description)  
VALUES (,Шоколадови вафли', 'В серията са включени вафли с шоколадова  
глазура'),  
(,Диетични вафли', 'Серия диетични вафли с натурални продукти');
```

```
INSERT INTO Products(Id, Product_name, Product_line_name, Price, Quantity)  
VALUES(23, 'Мура лешник', 'Шоколадови вафли', 0.95, 523),  
(24, 'Мура шоколад', 'Шоколадови вафли', 0.92, 600),  
(46, 'Лимецка', 'Диетични вафли', 1.25, 892);
```

3. Напишете заявка, която да изведе информация за имената на продуктите, продуктовата им серия и текстовото описание на продуктовата серия.

```
SELECT Product_name, Name, Description  
FROM Product_lines  
INNER JOIN Products  
ON Product_lines.Name=Products.Product_line_name;
```

4. Напишете заявка, която да изведе имената на продуктите с най-висока цена.

```
SELECT Product_name  
FROM Products  
WHERE Price=(SELECT max(Price) FROM Products);
```

5. Напишете заявка, която да отразява в тази база от данни продажбата на 200 броя от продукт с Id 24.

```
UPDATE Products  
SET Quantity=Quantity-200  
WHERE ID=24;
```

## 28.

**C#**

```
using System;
using System.Collections.Generic;
using System.Linq;

class Student
{
    public int StudentId { get; set; }
    public string Name { get; set; }
    public List<int> Scores { get; set; }

    public Student(int studentId, string name, List<int> scores)
    {
        StudentId = studentId;
        Name = name;
        Scores = scores;
    }
}

class StudentDatabase
{
    private List<Student> students;

    public StudentDatabase()
    {
        students = new List<Student>();
    }

    public void AddStudent(Student student)
    {
        students.Add(student);
    }

    public void RemoveStudent(int studentId)
    {
        students.RemoveAll(s => s.StudentId == studentId);
    }

    public string GetStudentInfo(int studentId)
    {
        var student = students.FirstOrDefault(s => s.StudentId == studentId);
        return student != null
            ? $"Student ID: {student.StudentId}, Name: {student.Name}, Scores: {string.Join(", ", student.Scores)}"
            : "Student not found.";
    }
}
```

```

        public string GetStudentAverageScore(int studentId)
        {
            var student = students.FirstOrDefault(s => s.StudentId ==
studentId);
            return student != null
                ? $"Average Score for {student.Name}: {student.
Scores.Average():F2}"
                : "Student not found.";
        }
    }
}
class Program
{
    static void Main()
    {
        var student1 = new Student(1, "John", new List<int> { 85,
90, 78 });
        var student2 = new Student(2, "Jane", new List<int> { 92,
88, 95 });

        var database = new StudentDatabase();

        database.AddStudent(student1);
        database.AddStudent(student2);

        Console.WriteLine(database.GetStudentInfo(1));
        Console.WriteLine(database.GetStudentInfo(2));

        Console.WriteLine(database.GetStudentAverageScore(1));
        Console.WriteLine(database.GetStudentAverageScore(2));

        database.RemoveStudent(1);

        Console.WriteLine(database.GetStudentInfo(1));
        Console.WriteLine(database.GetStudentInfo(2));
    }
}

```

#### Java

```

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
public class Main {
    public static void main(String[] args) {

```



```

class Student {
    private int student_id;
    private String name;
    private List<Integer> scores;
    public Student(int student_id, String name,
List<Integer> scores) {
        this.student_id = student_id;
        this.name = name;
        this.scores = scores;
    }
    public int getStudentId() {
        return student_id;
    }
    public String getName() {
        return name;
    }
    public List<Integer> getScores() {
        return scores;
    }
}
class StudentDatabase {
    private Map<Integer, Student> database;
    public StudentDatabase() {
        database = new HashMap<>();
    }
    public void addStudent(Student student) {
        database.put(student.getStudentId(), student);
    }
    public void removeStudent(int studentId) {
        database.remove(studentId);
    }
    public Student getStudentInfo(int studentId) {
        return database.get(studentId);
    }
    public double getStudentAverageScore(int studentId) {
        Student student = database.get(studentId);
        if (student == null) {
            return -1.0; // Връщаме -1, ако студентът не
съществува
        }
        List<Integer> scores = student.getScores();
        double sum = 0;
        for (int score : scores) {
            sum += score;
        }
        return sum / scores.size();
    }
}
Student student1 = new Student(1, "Иван", List.of(90, 85,
78));

```

```

        Student student2 = new Student(2, "Петя", List.of(95, 88,
92));
        Student student3 = new Student(3, "Георги", List.of(75,
60, 68));
        StudentDatabase database = new StudentDatabase();
        database.addStudent(student1);
        database.addStudent(student2);
        database.addStudent(student3);
        System.out.println("Информация за студентите:");
        System.out.println(database.getStudentInfo(1).getName() +
": " + database.getStudentInfo(1).getScores());
        System.out.println(database.getStudentInfo(2).getName() +
": " + database.getStudentInfo(2).getScores());
        System.out.println(database.getStudentInfo(3).getName() +
": " + database.getStudentInfo(3).getScores());
        System.out.println("\nСреден успех на студентите:");
        System.out.println(database.getStudentInfo(1).getName() +
": " + database.getStudentAverageScore(1));
        System.out.println(database.getStudentInfo(2).getName() +
": " + database.getStudentAverageScore(2));
        System.out.println(database.getStudentInfo(3).getName() +
": " + database.getStudentAverageScore(3));
        // Премахване на студент
        database.removeStudent(2);
        System.out.println("\nИнформация след премахване на
студент №2:");
        System.out.println(database.getStudentInfo(2)); // Трябва
да върне null
    }
}

```