

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

**ТЕМА 4**

<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>Г</b>	<b>В</b>	<b>Г</b>	<b>В</b>	<b>В</b>	<b>А</b>	<b>А</b>	<b>Г</b>	<b>В</b>	<b>Г</b>	<b>Б</b>	<b>Б</b>	<b>А</b>	<b>В</b>	<b>Г</b>	<b>А</b>

17. 12

18. А

19.

<b>C#</b>
<pre> public class Person {     public string Name { get; set; }      public Person(string name)     {         Name = name;     }     ~Person()     {         Name = string.Empty;     } } </pre>
<b>Java</b>
<pre> public class Person {     private String name;      public Person(String name) {         this.name = name;     }      public void setName(String name) {         this.name = name;     }      public String getName() {         return name;     } } </pre>

20. – 3, 24, 47, 73, 87, 169

21.

<b>C#</b>
<pre>using System; class Program {     static void Main(string[] args)     {         string a, b, c;         Console.Write("Въведете a:");         a = Console.ReadLine();         Console.Write("Въведете b:");         b = Console.ReadLine();          c = a;         a = b;         b = c;          Console.WriteLine("a=" + a + " b=" + b);     } }</pre>
<b>Java</b>
<pre>import java.util.Scanner; public class SwapVariables {     public static void main(String[] args) {         String a, b, c;         Scanner input = new Scanner(System.in);          System.out.print("Въведете a: ");         a = input.nextLine();         System.out.print("Въведете b: ");         b = input.nextLine();          c = a;         a = b;         b = c;          System.out.println("a=" + a + " b=" + b);     } }</pre>

22.

<b>CTITLE</b>
DB2
SQL Workshop

23. Г

24.

(1) 5 операции.

(2) Стекът е празен след операциите.

(3) Елементите на опашката са 5, 4, 3, 2, 1 след операциите.

25.

**C#**

```
using System;
class Program
{
    static void Main(string[] args)
    {
        try
        {
            Console.Write("Въведете първо цяло число: ");
            int number1 = int.Parse(Console.ReadLine());
            Console.Write("Въведете второ цяло число: ");
            int number2 = int.Parse(Console.ReadLine());
            Console.WriteLine("Общите делители на {0} и {1} са:",
number1, number2);
            int minNumber = Math.Min(number1, number2);
            for (int i = 1; i <= minNumber; i++)
            {
                if (number1 % i == 0 && number2 % i == 0)
                {
                    Console.WriteLine(i);
                }
            }
        }
        catch (Exception)
        {
            Console.WriteLine("Something went wrong!");
        }
    }
}
```

**Java**

```
import java.util.Scanner;
public class CommonDivisors {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {
            System.out.print("Въведете първо цяло число: ");
            int number1 = Integer.parseInt(scanner.nextLine());
            System.out.print("Въведете второ цяло число: ");
            int number2 = Integer.parseInt(scanner.nextLine());
            System.out.println("Общите делители на " + number1 +
" и " + number2 + " са:");
            int minNumber = Math.min(number1, number2);
            for (int i = 1; i <= minNumber; i++) {
                if (number1 % i == 0 && number2 % i == 0) {
                    System.out.println(i);
                }
            }
        } catch (NumberFormatException e) {
            System.out.println("Something went wrong!");
        }
    }
}
```

**26.****C#**

```
using System;
using System.Collections.Generic;
using System.Linq;
class Program
{
    static void Main()
    {
        // Четем броя на студентите
        int N = int.Parse(Console.ReadLine());
        // Създаваме списък за резултатите на студентите
        List<double> results = new List<double>();
        // Четем резултатите и ги добавяме към списъка
        for (int i = 0; i < N; i++)
        {
            double result = double.Parse(Console.ReadLine());
            results.Add(result);
        }
        // Филтрираме студентите, които са се явили на изпита
        (резултат различен от -1)
```

```

        var presentStudents = results.Where(r => r != -1).
ToList();
        // Изчисляваме броя на студентите, явили се на изпита
        int presentCount = presentStudents.Count;
        // Изчисляваме средния резултат
        double averageResult = presentStudents.Average();
        // Намираме максималния резултат
        double maxResult = presentStudents.Max();
        // Намираме имената на студентите с най-висок резултат
        List<string> topStudents = new List<string>();
        for (int i = 0; i < N; i++)
        {
            if (results[i] == maxResult)
            {
                topStudents.Add($"Студент {i + 1}");
            }
        }
        // Извеждаме резултатите
        Console.WriteLine($"Брой на студентите, явили се на
изпита: {presentCount}");
        Console.WriteLine($"Среден резултат от изпита:
{averageResult:F2}");
        Console.WriteLine($"Студенти с най-висок резултат:
{string.Join(", ", topStudents)}");
    }
}

```

## Java

```

import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
public class PhysicsExam {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // Четем броя на студентите
        int N = Integer.parseInt(scanner.nextLine());
        // Създаваме списък за резултатите на студентите
        List<Double> results = new ArrayList<>();
        // Четем резултатите и ги добавяме към списъка
        for (int i = 0; i < N; i++) {
            double result = Double.parseDouble(scanner.nextLine());
            results.add(result);
        }
        // Филтрираме студентите, които са се явили на изпита
        (резултат различен от -1)
    }
}

```

```

List<Double> presentStudents = new ArrayList<>();
for (Double result : results) {
    if (result != -1) {
        presentStudents.add(result);
    }
}
// Изчисляваме броя на студентите, явили се на изпита
int presentCount = presentStudents.size();
// Изчисляваме средния резултат
double averageResult = 0.0;
if (presentCount > 0) {
    double sum = 0.0;
    for (Double result : presentStudents) {
        sum += result;
    }
    averageResult = sum / presentCount;
}
// Намираме максималния резултат
double maxResult = -1.0;
for (Double result : presentStudents) {
    if (result > maxResult) {
        maxResult = result;
    }
}
// Намираме имената на студентите с най-висок резултат
List<String> topStudents = new ArrayList<>();
for (int i = 0; i < results.size(); i++) {
    if (results.get(i).equals(maxResult)) {
        topStudents.add("Студент " + (i + 1));
    }
}
// Извеждаме резултатите
System.out.println("Брой на студентите, явили се на
изпита: " + presentCount);
System.out.println("Среден резултат от изпита: " +
String.format("%.2f", averageResult));
System.out.println("Студенти с най-висок резултат: " +
String.join(", ", topStudents));
}
}

```

## 27.

```

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);

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);

```

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

```

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

```

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

```

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

```

6. 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#
<pre> using System; using System.Collections.Generic;     // Клас за представяне на книга class Book {     public string Title { get; } </pre>

```

    public string Author { get; }
    public string ISBN { get; }
    public int PublicationYear { get; }
    public Book(string title, string author, string isbn, int
publicationYear)
    {
        Title = title;
        Author = author;
        ISBN = isbn;
        PublicationYear = publicationYear;
    }
    public override string ToString()
    {
        return $"Title: {Title}, Author: {Author}, ISBN: {ISBN},
Year: {PublicationYear}";
    }
}
// Клас за представяне на библиотека
class Library
{
    // Променете модификатора за достъп на ,books' на public
    public List<Book> books = new List<Book>();
    public void AddBook(Book book)
    {
        books.Add(book);
    }
    public void RemoveBook(string isbn)
    {
        books.RemoveAll(book => book.ISBN == isbn);
    }
    public void ListBooks()
    {
        foreach (var book in books)
        {
            Console.WriteLine(book);
        }
    }
}
// Клас за представяне на читател
class Reader
{
    public string Name { get; }
    private List<Book> borrowedBooks = new List<Book>();
    public Reader(string name)
    {
        Name = name;
    }
}

```



```

public void BorrowBook(Library library, string isbn)
{
    var book = library.books.Find(b => b.ISBN == isbn);
    if (book != null)
    {
        borrowedBooks.Add(book);
        library.RemoveBook(isbn);
        Console.WriteLine($"{Name} borrowed: {book}");
    }
    else
    {
        Console.WriteLine("Book not found or already
borrowed.");
    }
}

public void ReturnBook(Library library, Book book)
{
    borrowedBooks.Remove(book);
    library.AddBook(book);
    Console.WriteLine($"{Name} returned: {book}");
}

public void ListBorrowedBooks()
{
    Console.WriteLine($"{Name}'s borrowed books:");
    foreach (var book in borrowedBooks)
    {
        Console.WriteLine(book);
    }
}
}

class LibraryManagementSystem
{
    static void Main(string[] args)
    {
        // Създаване на библиотека и добавяне на книги
        Library library = new Library();
        library.AddBook(new Book("Хамлет", " Уилям Шекспир",
"9786192511067", 2021));
        library.AddBook(new Book("Те бяха десет", "
        library.AddBook(new Book("Брулени хълмове", "Емили Бронте",
"9789542843306", 2023));
        // Създаване на читатели и извършване на операции с книги
        Reader reader1 = new Reader("Петър");
        Reader reader2 = new Reader("Рая");
        reader1.BorrowBook(library, "9786192511067");
    }
}

```

```

        reader2.BorrowBook(library, "9789543897407");
        reader1.BorrowBook(library, "9789543897407");
        reader1.ListBorrowedBooks();
        reader2.ListBorrowedBooks();
        reader1.ReturnBook(library, new Book("Хамлет", "Уилям
Шекспир", " 9786192511067", 2021));
        reader1.ListBorrowedBooks();
        // Извеждане на списъка с книги в библиотеката
        Console.WriteLine("Library books:");
        library.ListBooks();
    }
}

```

## Java

```

import java.util.*;
public class LibraryManagementSystem {
    public static void main(String[] args) {
        // Клас за представяне на книга
        class Book {
            private String title;
            private String author;
            private String isbn;
            private int publicationYear;
            public Book(String title, String author, String isbn,
int publicationYear) {
                this.title = title;
                this.author = author;
                this.isbn = isbn;
                this.publicationYear = publicationYear;
            }
            @Override
            public String toString() {
                return "Title: " + title + ", Author: " + author
+ ", ISBN: " + isbn + ", Year: " + publicationYear;
            }
        }
        // Клас за представяне на библиотека
        class Library {
            private List<Book> books = new ArrayList<>();
            public void addBook(Book book) {
                books.add(book);
            }
            public void removeBook(String isbn) {
                books.removeIf(book -> book.toString().
contains(isbn));
            }
        }
    }
}

```

```

        public List<Book> getBooks() {
            return books;
        }
    }
    // Клас за представяне на читател
    class Reader {
        private String name;
        private Set<Book> borrowedBooks = new
HashSet<>();

        public Reader(String name) {
            this.name = name;
        }

        public void borrowBook(Library library, String isbn)
{
            List<Book> libraryBooks = library.getBooks();
            for (Book book : libraryBooks) {
                if (book.toString().contains(isbn)) {
                    borrowedBooks.add(book);
                    library.removeBook(isbn);
                    System.out.println(name + " borrowed: " +
book);
                }
            }
            return;
        }

        System.out.println("Book not found or already
borrowed.");
    }

    public void returnBook(Library library, Book book) {
        borrowedBooks.remove(book);
        library.addBook(book);
        System.out.println(name + " returned: " + book);
    }

    public void listBorrowedBooks() {
        System.out.println(name + "'s borrowed books:");
        for (Book book : borrowedBooks) {
            System.out.println(book);
        }
    }
}

// Създаване на библиотека и добавяне на книги
Library library = new Library();
library.addBook(new Book("Хамлет", "Уилям Шекспир",
"9786192511067", 2021));
library.addBook(new Book("Те бяха десет", "
library.addBook(new Book("Брулени хълмове", "Емили Бронте",
"9789542843306", 2023));

```

```

// Създаване на читатели и извършване на операции с книги
Reader reader1 = new Reader("Петър");
Reader reader2 = new Reader("Рая");
reader1.borrowBook(library, "9786192511067");
reader2.borrowBook(library, "9789543897407");
reader1.borrowBook(library, "9789543897407");
reader1.listBorrowedBooks();
reader2.listBorrowedBooks();
reader1.returnBook(library, new Book("Хамлет", "Уилям
Шекспир", "9786192511067", 2021));
reader1.listBorrowedBooks();
// Извеждане на списъка с книги в библиотеката
System.out.println("Library books:");
List<Book> libraryBooks = library.getBooks();
for (Book book : libraryBooks) {
    System.out.println(book);
}
}
}

```