Backend-разработка на Java. Весна 2024

Оглавление

[**Вступительный контест Backend-разработка на Java, весна 2024** 2](#_Toc155807158)

[**1 задание** 3](#_Toc155807159)

[**2 задание** 4](#_Toc155807160)

[**3 задание** 5](#_Toc155807161)

[**4 задание** 7](#_Toc155807162)

[**5 задание** 9](#_Toc155807163)

[**Вступительный тест Backend-разработка на Java, весна 2024** 11](#_Toc155807164)

# **Вступительный контест Backend-разработка на Java, весна 2024**

Вступительный контест, состоящий из 5 задач. Задачи можно решать на любом языке программирования. Экзамен ограничен по времени — вам дается 4 часа. У вас есть только одна попытка написать контест.

# **1 задание**

**Ограничение времени**

1 секунда

**Ограничение памяти**

512 МБ

Львенок Максим использует в качестве паролей только последовательности из цифр.

Все пароли Максим хранит в одном текстовом файле в формате «code[DIGITS]», где вместо [DIGITS] подставляются цифры очередного пароля. Кроме записей паролей, в этом файле содержатся другие строчные латинские буквы, которые должны помогать Льву вспоминать, от чего тот или иной пароль.

Однажды Максим решил разобраться в своих паролях и распечатать свой файл. Он не хочет печатать этот файл в явном виде с паролями, так как это не безопасно.

Помогите Львенку и напишите программу, которая заменит все вхождения паролей в файле (вместе со словом code) на «???» (три символа вопроса).

**Формат входных данных**

В первой и единственной строке входных данных дано содержимое текстового файла. Текстовый файл содержит только строчные латинские буквы и цифры. Гарантируется, что текстовый файл содержит не больше ﻿105﻿ символов.

**Формат выходных данных**

Выведите содержимое файла с паролями, в котором вместо каждого пароля будут выведены три вопросительных знака.

Примеры данных

Пример 1

Ввод

a1b2c3

Вывод

a1b2c3

Пример 2

Ввод

code12345

Вывод

???

Пример 3

Ввод

aaacodebcode1code2code3bbb

Вывод

aaacodeb?????????bbb

Пример 4

Ввод

code999andcodec1andcocode888

Вывод

???andcodec1andco???

# **2 задание**

**Ограничение времени**

1 секунда

**Ограничение памяти**

512 МБ

Заяц Миша очень любит читать, поэтому на новый год ему подарили ﻿*n*﻿ книг.

Михаил хочет сложить компактно все подаренные книги в несколько стопок. Заяц считает, что книги сложены в стопки красиво, если в каждой стопке все книги имеют одинаковую высоту.

Миша хочет сложить книги в минимальное число стопок так, чтобы книги были сложены красиво. Помогите ему и найдите, как нужно сложить книги в стопки.

**Формат входных данных**

В первой строке входных данных дано целое число ﻿*n*﻿ — число книг, подаренных Зайцу(﻿1 ≤ n ≤ 105 1 ≤ *n* ≤ 105). Во второй строке дано ﻿*n*﻿ целых чисел ﻿*x*1​,…,*xn*​﻿ — высоты книг(﻿1≤xi≤ 105 1 ≤ *xi*​ ≤ 105﻿).

**Формат выходных данных**

В первой строке вывода напечатайте одно число ﻿*k*﻿ — минимальное число стопок книг. Во второй строке выведите высоты стопок в порядке не убывания.

Примеры данных

Пример 1

Ввод

2

1 2

Вывод

2

1 1

Пример 2

Ввод

4

1 2 2 3

Вывод

3

1 1 2

Пример 3

Ввод

5

5 4 4 5 5

Вывод

2

2 3

# **3 задание**

**Ограничение времени**

2 секунды

**Ограничение памяти**

512 МБ

Костя ответственно относится к подаркам для своих друзей. В этот раз он решил сделать ﻿*n*﻿ подарков своими руками для своих лучших друзей.

Для изготовления подарков он заказал материалы, причём материал для подарка ﻿*i*﻿ придёт в день ﻿*di*​﻿. Также Костя знает, что на изготовление подарка ﻿*i*﻿ нужно потратить ﻿*ci*​﻿ дней. Костя трудолюбивый, поэтому в один день может готовить любое число подарков (даже все).

Готовые подарки Костя планирует отправить почтой. Чтобы подарок ﻿*i*﻿ успел дойти до нового года, его нужно отправить не позднее дня ﻿*si*​﻿. Кроме того, на почте действует ограничение: можно отправлять не больше одной посылки в один день.

Помогите Косте понять, успеет ли он приготовить и отправить все подарки так, чтобы они дошли до получателей вовремя.

**Формат входных данных**

В первой строке входных данных дано число ﻿*n*﻿ — число подарков, которое планирует приготовить Костя (﻿1≤*n*≤105﻿).

В следующих ﻿*n*﻿ строках дана информация про каждый из подарков. Описание подарка ﻿*i*﻿ состоит из трех целых положительных чисел ﻿*di*​﻿, ﻿*ci*​﻿ и ﻿*si*​﻿ — день, в который доставят материалы для изготовления подарка, время, необходимое на приготовление подарка, и день, до которого должен быть отправлен подарок, чтобы он успел дойти (﻿1≤*di*​,*ci*​,*si*​≤109﻿).

**Формат выходных данных**

Выведите «YES» (без кавычек), если Костя успеет приготовить и отправить все подарки вовремя, или «NO» (без кавычек) в противном случае.

**Замечание**

В первом тесте из условия Костя может отправить первый подарок в третий день, второй — в шестой день, а третий — в пятый.

Во втором тесте из условия Костя никак не может успеть отправить все подарки, так как он должен за 3 дня (второй, третий и четвертый) отправить четыре подарка.

В третьем тесте из Костя может отправить первый подарок в третий день, второй — в шестой день, третий — в четвертый день, а четвертый — в пятый.

Примеры данных



# **4 задание**

**Ограничение времени**

1 секунда

**Ограничение памяти**

512 МБ

Дракоша Катя хочет провести большую вечеринку. Она уже определилась с тем, сколько и каких угощений она приготовит для гостей, однако не определилась, кого звать.

Про каждого друга-дракона Катя знает две характеристики ﻿*ai*​﻿ и ﻿*bi*​﻿ — интересность и прожорливость. Некоторые драконы могут дружить друг с другом. Также считается, что драконы ﻿*i*﻿ и ﻿*j*﻿ знакомы, если они дружат или дружат через одного или нескольких общих друзей. Более формально, дракон *i*﻿ знаком с драконом ﻿*j*﻿, если существует последовательность драконов ﻿*i*=*c*1​,*c*2​,…,*ck*​=*j*﻿ такая, что дракон ﻿*ct*−1​﻿ дружит с драконом ﻿*ct*​﻿.

Катя не может позвать дракона ﻿*i*﻿, если не позовет дракона ﻿*j*﻿, который знаком с драконом ﻿*i*﻿. Если она так сделает, то дракон ﻿*j*﻿ узнает о вечеринке и обидится на Катю.

Катя хочет позвать такое множество друзей, чтобы суммарная прожорливость всех гостей не превышала ﻿*g*﻿, а интересность приглашенных драконов была максимальной. Помогите Кате завершить приготовления к вечеринке и определиться со списком гостей.

**Формат входных данных**

Первая строка входных данных содержит три целых числа ﻿*n*﻿, ﻿*m*﻿ и ﻿*g*﻿﻿*w*﻿ — число друзей Кати, число пар драконов, которые дружат друг с другом, и максимальная суммарная прожорливость друзей, на которую хватит угощений, заготовленных Катей (﻿1≤*n*≤103,0≤*m*≤105,1≤*g*≤103﻿).

Вторая строка входных данных содержит ﻿*n*﻿ целых чисел ﻿1,2,…,*a*1​,*a*2​,…,*an*​﻿ (﻿1≤*ai*​ ≤106﻿) — интересность драконов-друзей Кати.

Третья строка содержит n целых чисел ﻿*b*1​,*b*2​,…,*bn*​﻿ (1≤*bi*​ ≤103﻿) — прожорливость драконов-друзей Кати.

В следующих ﻿*m*﻿ строках дано описание пар друзей-драконов, которые дружат друг с другом напрямую. Каждая пара друзей задается номерами драконов ﻿*ui*​﻿ и ﻿*vi*​﻿ (﻿1≤*ui*​ , *vi*​ ≤*n*,*ui*​ !=*vi*​﻿). При этом все пары номеров друзей (﻿*ui*​, *vi*​﻿) различны, а отношение дракон ﻿*ui*​﻿ дружит с драконом ﻿*vi*​﻿ является симметричным.

**Формат выходных данных**

В единственной строке выходного файла выведите наибольшую возможную суммарную интересность драконов, которых Екатерина может позвать на вечеринку, так чтобы соблюсти все условия.

Примеры данных



# **5 задание**

Ограничение времени

4 секунды

Ограничение памяти

512 МБ

Единорог Тимофей отправился в путешествие по волшебному лабиринту.

Лабиринт состоит из ﻿*n*﻿ комнат, которые соединены переходами так, что между двумя любыми комнатами единственный путь.

Тимофей боится заблудиться в лабиринте, поэтому написал программу, которая помогает выбрать правильное направление при движении по лабиринту. Программа по номеру комнаты, в которой сейчас находится Тимофей, и комнате, в которую он хочет попасть, находит номер комнаты, в которую нужно перейти из той комнаты, в которой он находится, чтобы за минимальное число переходов добраться до той комнаты, в которую он хочет попасть.

Единорог потратил на написание программы 30 минут, а сможете ли вы написать такую же быстрее?

**Формат входных данных**

В первой строке входных данных дано одно целое число ﻿*n*﻿ — число комнат в лабиринте (﻿2≤*n*≤2⋅105﻿).

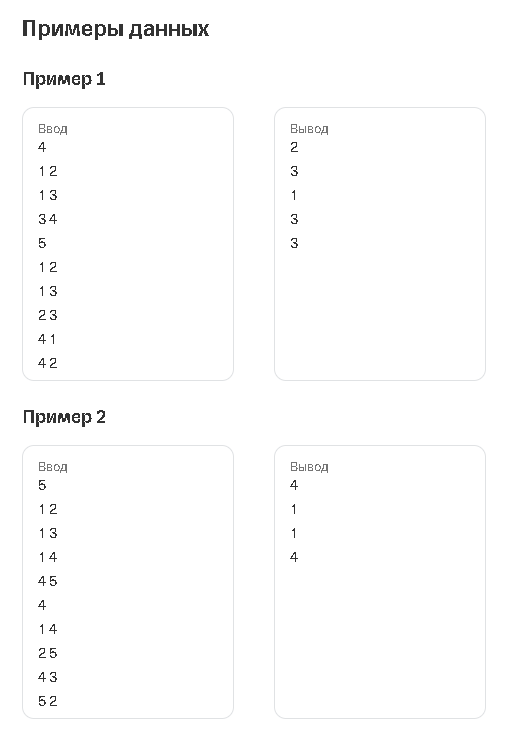
Далее дано ﻿*n*−1﻿ строк, в которых описаны переходы в лабиринте. Описание каждого перехода состоит из двух целых чисел — номеров узлов, которые он соединяет *ui*​﻿ и ﻿*vi*​﻿ (﻿1≤*ui*​ ,*vi*​ ≤*n*;*ui*​ != *vi*​﻿ ). Гарантируется, что между двумя любыми комнатами существует единственный путь.

В следующей строке дано одно целое число ﻿*q*﻿ — число вопросов о направлении, которые интересуют Тимофея (﻿1≤*q*≤2⋅105 ﻿).

В следующих ﻿*q*﻿ строках даны пары чисел ﻿*fromi*​﻿ , ﻿*toi*​﻿ — номера комнат, про направление движения между которыми хочет узнать Тимофей (1≤*fromi*​ ,*toi*​ ≤*n*;*fromi*​ != *toi*​﻿ ).

**Формат выходных данных**

Для каждого вопроса выведите номер комнаты, в которую нужно перейти из комнаты ﻿*fromi*​﻿, чтобы попасть в комнату ﻿*toi*​﻿ за минимальное число переходов.



# **Вступительный тест Backend-разработка на Java, весна 2024**

34 вопроса по основам Java. Экзамен ограничен по времени — на ответы на все вопросы вам дается **90 минут**. У вас есть только одна попытка пройти тест.

1. Какие особенности имеет платформа JVM?

1. Garbage collector

2. Кросс-платформенность

3. Оба варианта верны

2. Что такое boxing примитивных типов?

1. Преобразование примитивного типа в объект

2. Преобразование объекта в примитивный тип

3. Преобразование строки в примитивный тип

3. Что такое стирание типов в Java?

1. Процесс, при котором информация о типах становится недоступной во время выполнения программы

2. Процесс, при котором информация о типах становится доступной во время выполнения программы

3. Процесс, при котором информация о типах становится доступной только для компилятора

4. Что такое generics в Java?

1. Механизм, позволяющий создавать анонимные классы

2. Механизм, позволяющий создавать обобщенные типы данных

3. Механизм, позволяющий создавать вложенные классы

5. Выберите корректные лямбда-выражения

1. (Integer x, Integer y) -> x + y

2. int x, int y -> x + y

3. (int x y) -> { return x + y; }

6. Какие классы используются для работы с файлами в Java?

1. File

2. Path

3. Directory

4. Folder

7. Какой класс используется для обозначения потенциально отсутствующего значения в Java?

1. Optional

2. Nullable

3. Maybe

4. NullableOptional

8. Какое ключевое слово используется для блокировки доступа в многопоточном коде в Java?

1. volatile

2. atomic

3. synchronized

4. mutex

9. Чем отличается ключевое слово volatile от атомиков?

1. volatile гарантирует атомарность операций

2. volatile используется только для примитивных типов данных

3. volatile не гарантирует атомарность операций

4. volatile используется только для объектов

10. Какие ошибки могут возникать при разработке многопоточных программ?

1. race condition

2. deadlock

3. livelock12.

11. Что такое Java Memory Model?

1. Это способ описания расположения данных в памяти

2. Это документ, описывающий правила доступа к разделяемой памяти в многопоточных программах

3. Это набор библиотек для работы с памятью

12. Happens-Before в Java - это:

1. Механизм, который определяет порядок выполнения операций в многопоточном приложении

2. Способ гарантировать согласованность изменений, вносимых разными потоками в одни и те же данные

3. Правило, согласно которому операция записи в переменную будет видна всем последующим операциям чтения этой переменной

4. Механизм, который позволяет избежать гонок данных и других проблем, связанных с параллельным выполнением кода

13. Что такое байткод в контексте Java?

1. Промежуточный язык, на котором написана Java

2. Компилированный код Java, который исполняется JVM

3. Специальный формат для представления бинарных чисел

4. Тип данных для представления байтов

14. Что такое ClassLoader в Java?

1. Интерфейс для создания классов

2. Класс для загрузки байткода

3. Инструмент для работы с кэшем

4. Плагин для работы с базами данных

15. Что такое сборщик мусора (GC) в Java?

1. Класс для работы с многопоточными операциями

2. Интерфейс для создания классов

3. Инструмент для автоматического очищения памяти от неиспользуемых объектов

4. Все вышеперечисленное

16. Для чего используются аннотации в Java?

1. Для создания обобщенного кода

2. Для объявления констант

3. Для преобразования типов

4. Для дополнительной информации о коде, которая может быть использована во время выполнения или компиляции

17. Что такое JNI?

1. Механизм, используемый для вызова native-функций в Java

2. Механизм, используемый для оптимизации работы приложения

3. Механизм, используемый для защиты приложения от внешних атак

4. Механизм, используемый для работы с базами данных

18. Какие из следующих выражений не будут вычислены как true, если

boolean b1 = true, b2 = false;

int i1 = 1, i2 = 2;

1. (i1 | i2) == 3

2. i2 && b1

3. b1 || !b2

4. (i1 ^ i2) < 4

19. Следующий код выведет

String str = "abcde";

str.trim();

str.toUpperCase();

str.substring(3, 4);

System.out.println(str);

1. CD

2. CDE

3. D

4. аbcde

20. Какой из концептов ООП демонстрирует следующий код

List[] myLists = {

new ArrayList<>(),

new LinkedList<>(),

new Stack<>(),

new Vector<>(),

};

for(List list:myLists) {

list.clear();

}

1. Композиция

2. Параметризация (Generics)

3. Полиморфизм

4. Инкапсуляция

21. Какой из примеров эквивалентен коду

List<Integer> numbers = List.of(1, 2, 3, 4);

int total = 0;

for(Integer x:numbers) {

if(x % 2 == 0) total += x\*x;

}

1. int total = numbers.stream().transform(x -> x \* x).filter(x -> x % 2 == 0).sum();

2. int total = numbers.stream().filter(x -> x % 2 == 0).collect(Collectors.toInt());

3. int total = numbers.stream().mapToInt(x -> { if (x % 2 == 0) return x \* x }).sum();

4. int total = numbers.stream().mapToInt(x -> x % 2 == 0 ? x \* x : 0).sum();

22. Какие из утверждений верны в отношении аннотаций

1. @interface используется для создания новых аннотаций

2. @Override – встроенная в JDK аннотация

3. Аннотации нельзя применять к полям класса

4. @Retention это мета-аннотация

5. Аннотации удаляются после компиляции

23. Следующий код выведет

public class Test {

public static void main(String[] args) {

String s1 = new String("abc");

String s2 = new String("abc");

System.out.println("s1 == s2 is:" + (s1 == s2));

}}

1. s1 == s2 is:true

2. false

3. s1 == s2 is:false

4. true

24. Какой из классов не реализует интерфейс Collection

1. java.util.Vector

2. java.util.ArrayList

3. java.util.HashSet

4. java.util.HashMap

25. Какой из примитивов синхронизации подойдёт для многопоточного доступа к переменной

1. ReentrantLock

2. CountdownLatch

3. Future

4. Phaser

5. CyclicBarrier

26. Какое ключевое слово нужно добавить на метод, чтобы была гарантия, что два потока не смогут одновременно запустить его у одного и того же экземпляра?

1. native

3. synchronized

4. volatile

5. lock

27. Какой из классов является корневым для всех классов в Java

1. Any

2. Object

3. Nothing

4. Root

28. Какое описание больше всего соответствует понятию "unit-тесты"

1. Проверяют хорошо ли работают вместе различные модули и сервисы, используемые приложением

2. Копируют поведение пользователя при работе с ПО в контексте всего приложения

3. Оценивают работу системы при определенной рабочей нагрузке

4. Проверяют корректность работы отдельных, самых маленьких элементов системы - функции, методы

29. Выберите верные утверждения об интерфейсах

1. Интерфейс описывает контракт (функции)

2. Интерфейс хранит состояние (переменные)

3. В интерфейсе можно объявлять константы

4. Интерфейс может иметь имплементацию по умолчанию

5. Интерфейс не может быть пустым: должен быть описан хотя бы один метод

6. Методы интерфейса могут быть финальными

30. Какой из модификаторов запрещает редактирование переменной после инициализации

1. protected

2. static

3. final

4. private

31. Наследники класса создаются при помощи ключевого слова

1. abstracts

2. extends

3. implements

4. interfaces

32. Как сделать инициализацию переменной типа byte со значением 200

1. byte b = 200;

2. var b = 200;

3. byte b = (byte) 200;

4. Никак

33. Дан список сотрудников List<Employee> employees, при этом у класса Employee есть метод String getName(). Какой из вызовов вернет список имён сотрудников?

1. employees.collect(e → e.getName());

2.employees.filter(Employee::getName).collect(Collectors.toUnmodifiableList()

3.employees.stream().map(Employee::getName).collect(Collectors.toList());

4. employees.stream().collect(e → e.getName());

34. Какие из способов инициализации массива правильные

1. int arr[] = { 22, 2, 10 };

2. int[] arr = { 1, 11, 3 };

3. int arr[][] = { 1, 2, 3, 4 };

4. int[] arr = (5, 4, 3);

5. int[] arr = { "1", "2", "3" };

6. int[] arr = new int[] { 33, 41, 0 };