



# СОДЕРЖАНИЕ

<b>Предисловие</b> .....	<b>4</b>
<b>Инструкция по выполнению работы</b> .....	<b>6</b>
<b>ВАРИАНТ С ПОДРОБНЫМ РАЗБОРОМ РЕШЕНИЙ</b> .....	<b>7</b>
<b>ВАРИАНТЫ ЭКЗАМЕНАЦИОННЫХ РАБОТ</b>	
Вариант 1.....	37
Вариант 2.....	46
Вариант 3.....	55
Вариант 4.....	64
Вариант 5.....	73
Вариант 6.....	82
Вариант 7.....	91
Вариант 8.....	100
Вариант 9.....	109
Вариант 10.....	118
<b>Ответы на задания с кратким ответом</b> .....	<b>127</b>
<b>Ответы и критерии оценки к заданиям с развёрнутым ответом</b> .....	<b>128</b>

## ПРЕДИСЛОВИЕ

Уважаемые учащиеся 9-х классов, абитуриенты и учителя!

Вашему вниманию предлагается сборник тренировочных вариантов экзаменационных работ по информатике для подготовки к ОГЭ в 2024 году.

Содержание заданий разработано по основным темам курса информатики и ИКТ, объединенным в следующие тематические блоки: «Представление и передача информации», «Обработка информации», «Основные устройства ИКТ», «Запись средствами ИКТ информации об объектах и о процессах, создание и обработка информационных объектов», «Проектирование и моделирование», «Математические инструменты, электронные таблицы», «Организация информационной среды, поиск информации».

Данный сборник содержит 10 типовых вариантов экзаменационных работ, составленных в соответствии с демонстрационным вариантом и спецификацией. Перед тренировочными вариантами в сборнике представлен **вариант с подробным разбором решений каждого задания**. Изучив решения каждого экзаменационного задания, вам будет проще самостоятельно справиться с остальными вариантами.

При выполнении любого из заданий от экзаменуемого требуется решить какую-либо задачу: либо прямо использовать известное правило, алгоритм, умение; либо выбрать из общего количества изученных понятий и алгоритмов наиболее подходящее и применить его в известной либо новой ситуации.

Часть 2 работы содержит практические задания, проверяющие наиболее важные практические навыки курса информатики: умение обработать большой информационный массив данных, создать презентацию или текстовый документ, разработать и записать простой алгоритм. Практическая часть работы может быть выполнена с использованием различных операционных систем и различных прикладных программных продуктов.

В экзаменационном варианте используются задания двух типов: с кратким ответом и развёрнутым. В конце пособия представлены ответы и критерии оценки на все 10 вариантов, включённых в сборник.

Каждый вариант КИМ состоит из двух частей и включает в себя 15 заданий.

Часть 1 содержит 10 заданий с кратким ответом двух типов: на вычисление определённой величины и на установление правильной последовательности, представленной в виде строки символов по определённому алгоритму. Ответы на задания даются соответствующей записью в виде натурального числа или последовательности символов (букв или цифр), записанных без пробелов и других разделителей.

Часть 2 содержит 5 заданий, для выполнения которых необходим компьютер. Задания этой части направлены на проверку практических навыков использования информационных технологий. В этой части два задания с кратким ответом и три задания с развёрнутым ответом в виде файла. Объективность проверки заданий с развёрнутым ответом обеспечивается едиными критериями оценивания.

В КИМ представлены задания разных уровней сложности: базового, повышенного и высокого.

Задания базового уровня проверяют освоение базовых знаний и умений, без которых невозможно успешное продолжение обучения на следующей ступени. Задания повышенного уровня сложности проверяют способность экзаменуемых действовать в ситуациях, в которых нет явного указания на способ выполнения и необходимо выбрать этот способ из набора известных ему или сочетать два-три известных способа действий. Задания высокого уровня сложности проверяют способность экзаменуемых решать задачи, в которых нет явного указания на способ выполнения и необходимо сконструировать способ решения, комбинируя известные им способы.

Задания части 1 могут выполняться экзаменуемыми без использования компьютеров. Вычислительная сложность заданий не требует использования калькуляторов, поэтому в целях обеспечения равенства всех участников экзамена использование калькуляторов на экзаменах не разрешается.

Задания части 2 выполняются на компьютере. На компьютере должны быть установлены знакомые экзаменуемым программы.

Для выполнения задания 13.1 необходима программа для работы с презентациями.

Для выполнения задания 13.2 необходим текстовый процессор.

Для выполнения задания 14 необходима программа для работы с электронными таблицами.

**Для выполнения заданий 11, 12, 13.1 и 14 требуются дополнительные материалы. Их вы можете найти на странице авторского сайта по ссылке <https://dmushakov.ru/files/z1114.zip>**

Задание 15.1 предусматривает разработку алгоритма для исполнителя «Робот». Для выполнения задания 15.1 рекомендуется использование учебной среды исполнителя «Робот». В качестве такой среды может использоваться, например, учебная среда разработки «Кумир», разработанная в НИИСИ РАН (<http://www.niisi.ru/kumir>) или любая другая среда, позволяющая моделировать исполнителя «Ро-

бот». В случае, если синтаксис команд исполнителя в используемой среде отличается от того, который дан в задании, допускается внесение изменений в текст задания в части описания исполнителя «Робот». При отсутствии учебной среды исполнителя «Робот» решение задания 15.1 записывается в простом текстовом редакторе.

Задание 15.2 предусматривает запись алгоритма на универсальном языке программирования. В этом случае для выполнения задания необходима система программирования, используемая при обучении.

Решением каждого задания части 2 является отдельный файл, подготовленный в соответствующей программе (текстовом редакторе или электронной таблице). Экзаменуемые сохраняют данные файлы в каталог под именами, указанными техническим специалистом.

Верное выполнение каждого задания части 1 и заданий 11 и 12 части 2 оценивается 1 баллом. Эти задания считаются выполненными, если экзаменуемый дал ответ, соответствующий эталону верного ответа. Максимальное количество первичных баллов, которое можно получить за выполнение заданий с кратким ответом, равно 12.

Выполнение заданий 13 и 15 с развёрнутым ответом оценивается от 0 до 2 баллов, выполнение задания 14 — от 0 до 3 баллов. Максимальное количество баллов, которое можно получить за выполнение заданий с развёрнутым ответом, равно 7.

Максимальное количество баллов за выполнение всех заданий экзаменационной работы равно 19.

На основе баллов, выставленных за выполнение всех заданий работы, подсчитывается суммарный первичный балл, который переводится в отметку по пятибалльной шкале:

С 2020 г. в экзаменационном варианте расширен набор заданий, выполняемых на компьютере, за счёт включения трёх новых заданий, проверяющих умения и навыки практической работы с компьютером: поиск информации средствами текстового редактора или операционной системы (задание 11); анализ содержимого каталогов файловой системы (задание 12); создание презентации или текстового документа (задание 13).

В отличие от КИМ прошлых лет, в нынешних вариантах отсутствуют задания с выбором ответа из предложенных альтернатив, т.е. во всех заданиях предусмотрен либо краткий, либо развёрнутый ответ.

Если при решении заданий из данного пособия Вы обнаружите какие-то неточности или опечатки, то на странице авторского сайта [www.dmushakov.ru](http://www.dmushakov.ru) можно посмотреть самые последние сведения о замеченных ошибках. Там же можно задать вопрос автору сборника и посмотреть, какие дополнительные пособия Д. М. Ушакова по информатике, издаваемые в нашем издательстве, могут быть Вам полезны при подготовке к экзамену.

**В связи с возможными изменениями в формате и количестве заданий рекомендуем в процессе подготовки к экзамену обращаться к материалам сайта официального разработчика экзаменационных заданий — Федерального института педагогических измерений: [www.fipi.ru](http://www.fipi.ru).**

## ИНСТРУКЦИЯ ПО ВЫПОЛНЕНИЮ РАБОТЫ

Работа состоит из двух частей, включающих в себя 15 заданий. Часть 1 содержит 10 заданий с кратким ответом; часть 2 содержит 5 заданий, которые необходимо выполнить на компьютере.

На выполнение работы по информатике отводится 2 часа 30 минут (150 минут). Вы можете самостоятельно определять время, которое отводите на выполнение заданий, но рекомендуемое время на выполнение заданий части 1 — 30 минут, на выполнение заданий части 2 — 2 часа (120 минут).

Ответы к заданиям 1–12 записываются в виде числа, слова, последовательности букв или цифр. Ответ запишите в поле ответа в тексте работы, а затем перенесите в бланк ответов № 1.

Результатом выполнения каждого из заданий 13–15 является отдельный файл. Формат файла, его имя и каталог для сохранения Вам сообщат организаторы экзамена.

Все бланки заполняются яркими чёрными чернилами. Допускается использование гелевой или капиллярной ручки.

При выполнении заданий можно пользоваться черновиком. Записи в черновике, а также в тексте контрольных измерительных материалов не учитываются при оценивании работы.

Баллы, полученные Вами за выполненные задания, суммируются. Постарайтесь выполнить как можно больше заданий и набрать наибольшее количество баллов.

После завершения работы проверьте, чтобы ответ на каждое задание в бланке ответов № 1 был записан под правильным номером.

*Желаем успеха!*

# ВАРИАНТ С ПОДРОБНЫМ РАЗБОРОМ РЕШЕНИЙ

## Часть 1

Ответом к заданиям 1–6 является одна цифра, которая соответствует номеру правильного ответа. Запишите эту цифру в поле ответа в тексте работы, а затем перенесите в БЛАНК ОТВЕТОВ № 1 справа от номера соответствующего задания, начиная с первой клеточки.

1 В одной из кодировок Unicode каждый символ кодируется 16 битами.

Коля написал текст (в нём нет лишних пробелов):

«Як, лис, барс, жираф, гепард, медведь, россомаха — дикие животные».

Ученик вычеркнул из списка название одного из животных. Заодно он вычеркнул ставшие лишними запятые и пробелы — два пробела не должны идти подряд.

При этом размер нового предложения в данной кодировке оказался на 12 байт меньше, чем размер исходного предложения. Напишите в ответе вычеркнутое название животного.

Ответ: \_\_\_\_\_.

Решение.

Чтобы понять, какое животное было вычеркнуто, найдём, для начала, количество символов, которое было вычеркнуто. Объём сообщения уменьшился на 12 байт. При этом каждый символ кодируется 16 битами. Переведём биты в байты. В одном байте 8 бит. Поэтому в 16 битах 2 байта. Итак, объём уменьшился на 12 байт, а в каждом символе 2 байта. Значит, сообщение уменьшилось на  $12 : 2 = 6$  символов. При этом после названия каждого животного (кроме россомахи) написаны запятая и пробел. Это два символа. Значит, вычеркнутое слово состояло из  $6 - 2 = 4$  символов. Такое название животного одно — барс.

О т в е т: барс.

2 От разведчика было получено сообщение:

0011000001110010

В этом сообщении зашифрован пароль — последовательность русских букв.

В пароле использовались только буквы А, Г, Л, М, Н, О; каждая буква кодировалась двоичным словом по такой таблице:

А	Г	Л	М	Н	О
11	011	010	001	000	10

Расшифруйте сообщение. Запишите в ответе пароль.

Ответ: \_\_\_\_\_.

Решение.

Анализируем приведённый код. Замечаем, что кодовые слова имеют разную длину (2 и 3). То есть, это неравномерный код. Чтобы неравномерный код можно было удобно и однозначно декодировать, он должен быть либо префиксным, либо постфиксным. То есть, никакое кодовое слово не должно быть началом (или концом — для постфиксности) никакого другого кодового слова. Исследуем короткие кодовые слова (А — 11 и О — 10).

Сначала проверим префиксность. Будем проверять, не являются ли первые два символа остальных кодовых слов короткими кодовыми словами 11 или 10. Ни одно из кодовых слов не является. Значит, код префиксный и его можно удобно и быстро однозначно декодировать слева направо.

Не будем строить дерево декодирования (как это рекомендует теория кодирования). В данном случае у нас небольшой код и кодовые слова небольшой длины, достаточно будет просто перебирать кодовые слова по такому алгоритму:

1. Возьмём первые два символа сообщения.
2. Посмотрим, не совпадают ли они с кодовым словом длины 2 (11 или 10).
  - а. Если совпадают — считаем, что эти два символа задают соответствующее кодовое слово (для 11 — А, для 10 — О).
  - б. Если не совпадают — возьмем из сообщения еще один символ и найдем, какая буква имеет такое кодовое слово.
3. Отбросим рассмотренные 2 или 3 символа сообщения.
4. Если сообщение при этом не закончилось, возьмём из него следующие два символа сообщения.

Перейдём к пункту 2.

Выполним алгоритм для данного сообщения (0011000001110010):

Первые два символа (00) не совпадают с кодовыми словами длины 2 (11 или 10).

Возьмём ещё один символ (1). Получим кодовое слово 001.

Найдём такое кодовое слово в таблице. Это буква М.

Рассмотрим следующие два символа сообщения (001.1000001110010). Это 10.

Такое кодовое слово соответствует букве О.

Рассмотрим следующие два символа сообщения (001.10.00001110010). Это 00.

Оно не соответствует двухсимвольным кодовым словам.

Возьмём следующий символ (0). Получим кодовое слово 000.

Найдём такое кодовое слово в таблице. Это буква Н.

Рассмотрим следующие два символа сообщения (001.10.000.01110010). Это 01.

Оно не соответствует кодовым словам букв А и О.

Возьмём ещё один символ (1). Получим кодовое слово 011.

Такое кодовое слово соответствует букве Г.

Рассмотрим следующие два символа сообщения (001.10.000.011.10010). Это 10.

Это кодовое слово буквы О.

Рассмотрим следующие два символа сообщения (001.10.000.011.10.010). Это 01.

Такого кодового слова в таблице нет.

Возьмём следующий символ (0). Получим кодовое слово 010.

Найдём такое кодовое слово в таблице. Оно соответствует букве Л.

Сообщение на этом закончилось. Выписываем получившиеся буквы: МОНГОЛ.

**О т в е т:** МОНГОЛ.

*Замечание.* При выполнении задания удобнее отделять точками рассматриваемые кодовые слова и подписывать под кодовым словом соответствующую букву. В данном случае получилось бы:

001. 10. 000. 011. 10. 010  
М О Н Г О Л

**3**

Напишите наименьшее число  $x$ , для которого истинно высказывание:

$(x > 30) \text{ И НЕ } (x \text{ нечётное}).$

Ответ: \_\_\_\_\_.

**Р е ш е н и е.** Приведём имеющееся высказывание к виду, удобному для анализа с точки зрения математики. Для этого избавимся от тех логических операций, которые усложняют понимание высказывание (либо от всех, от которых можно избавиться). В данном случае можно избавиться от операции НЕ. Для этого к выражению, к которому она относится ( $x$  нечётное), применим отрицание. Отрицанием высказывания « $x$  нечётное» (противоположным высказыванием) является высказывание « $x$  чётное». Получаем высказывание:  $(x > 30) \text{ И } (x \text{ чётное})$ . То есть, необходимо найти наименьшее чётное число, которое больше 30. Это число 32.

**О т в е т:** 32.

4 Между населёнными пунктами А, В, С, D, Е построены дороги, протяжённость которых (в километрах) приведена в таблице.

	А	В	С	D	Е
А		3	7	2	2
В	3		4		1
С	7	4		3	6
D	2		3		5
Е	2	1	6	5	

Определите длину кратчайшего пути между пунктами А и Е, проходящего через пункт С. Передвигаться можно только по дорогам, протяжённость которых указана в таблице, два раза посещать один пункт нельзя.

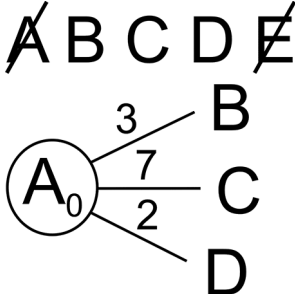
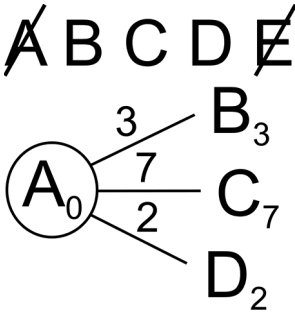
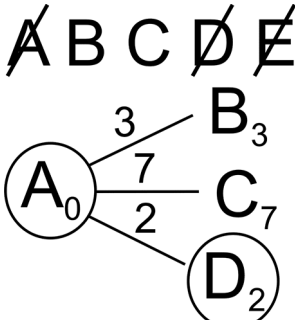
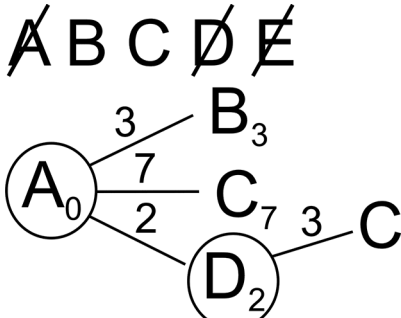
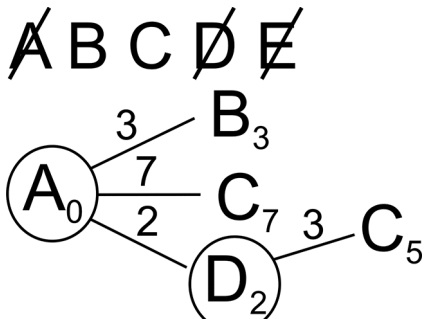
Ответ: \_\_\_\_\_.

**Решение.** Воспользуемся специальным алгоритмом, который эффективно решает именно такого вида задачу — находит кратчайшее расстояние на графе от одной вершины до другой. Этот алгоритм называется *алгоритмом Дейкстры*. Побочным эффектом алгоритма Дейкстры является нахождение кратчайшего расстояния от некоей стартовой вершины до всех остальных вершин графа. Но это никоим образом не ухудшает его качества — алгоритм Дейкстры ищет кратчайшее расстояние быстро, надёжно и эффективно.

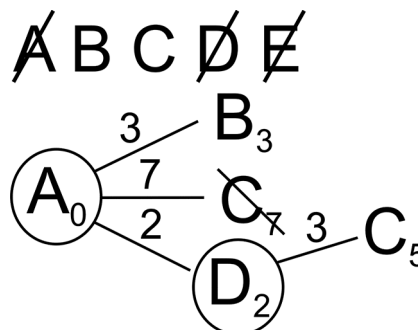
Чтобы найти при этом кратчайший путь, проходящий через пункт С, разобьём решение задачи на две части. На первом этапе найдём кратчайший путь из вершины А до вершины С. На втором этапе будем искать кратчайший путь из вершины С до вершины Е. Чтобы выполнить условие того, что нельзя посещать дважды один пункт, на втором этапе не будем рассматривать вершины, через которые прошёл путь на первом этапе.

0. Выпишем в отдельный список все возможные вершины графа.	A B C D E
1. Будем строить дерево. Нарисуем стартовую вершину (в нашем случае это точка А) и подпишем рядом с ней расстояние, которое нужно «проехать», чтобы добраться до неё из стартовой точки А. То есть, ноль. Эта вершина будет корнем нашего дерева. Так как через один и тот же пункт нельзя проезжать дважды, сразу вычеркнем из списка вершин, рассматриваемых на первом этапе, вершину Е. Потому что если мы проложим путь до вершины С через вершину Е, не сможем потом вернуться в вершину Е, т.к. получится, что мы посещаем вершину Е дважды.	<p>A B C D E</p> <p><del>E</del></p> <p>A<sub>0</sub></p>
2. Из всех возможных листовых вершин дерева найдём вершину с наименьшим числом (в данном случае, такая вершина только одна — А). Вычеркнем эту вершину из списка доступных вершин. В дальнейшем её мы больше не будем рассматривать в качестве возможной вершины дерева. То есть, вычеркнутые вершины в дерево больше не дорисовываются! Расстояния до них от стартовой вершины подсчитаны окончательно и лучше уже не будет. Следующее действие будем делать из этой вершины (А). Обведём её в кружок.	<p><del>A</del> B C D E</p> <p>(A<sub>0</sub>)</p>

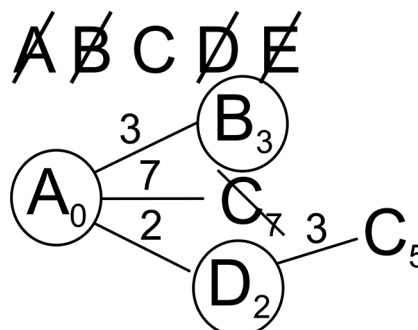


<p>3. По таблице расстояний найдём все вершины, до которых идёт ребро из текущей вершины (сейчас это вершина A) и которые ещё не вычеркнуты. Это вершины B, C, D. Нарисуем из текущей вершины (A) ветви дерева для каждой из найденных смежных вершин, нарисуем на концах ветвей названия этих вершин. Подпишем на ветвях длины рёбер по таблице расстояний.</p>	
<p>4. Для каждой полученной вершины посчитаем расстояние до неё как: расстояние до текущей вершины (подписано рядом с ней, сейчас для вершины A это 0) плюс расстояние (длина ребра) до каждой полученной вершины (подписано над ребром). Запишем эти расстояния рядом с каждой полученной вершиной.</p>	
<p>5. Среди листовых вершин найдём вершину с наименьшим расстоянием. В данном случае это вершина D<sub>2</sub>. Вычеркнем эту вершину из списка доступных вершин. Дальнейшие действия будем делать из неё. Обведём её в кружок. Заметим, это такое же действие, как действие 2.</p>	
<p>6. По таблице расстояний найдём все вершины, до которых идёт ребро из текущей вершины (сейчас это вершина D) и которые ещё не вычеркнуты. Это вершина C. Нарисуем из текущей вершины (D<sub>2</sub>) ветви дерева для каждой из найденных смежных вершин, нарисуем на концах ветвей названия этих вершин. Подпишем на ветвях длины рёбер по таблице расстояний. Сейчас такая вершина одна (C). Заметим, это такое же действие, как действие 3.</p>	
<p>7. Для каждой полученной вершины (сейчас она такая одна) посчитаем расстояние до неё как: расстояние до текущей вершины (подписано рядом с ней, сейчас для вершины D это 2) плюс расстояние (длина ребра) до полученной вершины (подписано над ребром, сейчас это 3). Запишем это расстояние рядом с полученной вершиной (5). Заметим, это такое же действие, как действие 4.</p>	

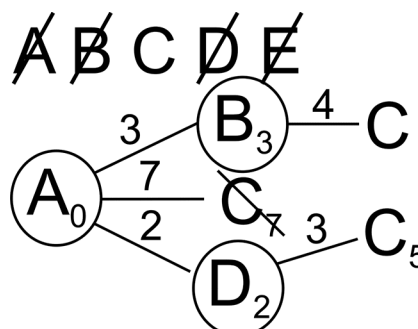
8. Среди всех листовых вершин будем искать одинаковые вершины. Сейчас это пара вершин  $C_5$  и  $C_7$ . Для каждой пары одинаковых вершин будем вычеркивать в дереве худшую вершину. То есть, вершину с бóльшим расстоянием до неё. Сейчас это  $C_7$ . Если в паре одинаковых вершин расстояния будут одинаковые, нужно вычеркнуть одну любую. Мы специально вычеркиваем вершины указанным образом (от левого верхнего к правому нижнему углу), чтобы перечеркнуть как название вершины, так и расстояние до неё. Это облегчит в дальнейшем поиск вершины с наименьшим расстоянием.  
Заметим, этого действия (вычеркивания) мы раньше не делали, потому что у нас ещё не встречались одинаковые вершины. В действительности, это часть циклического алгоритма. Действия 5–8 мы будем повторять, пока не вычеркнем конечную вершину (сейчас это  $C$ ) из списка доступных вершин.



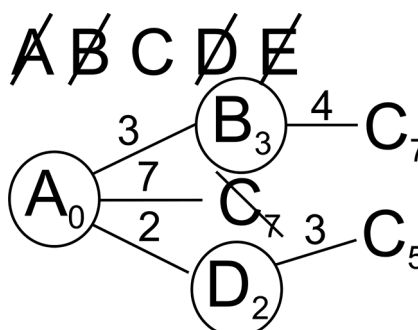
9. Среди листовых вершин найдём вершину с наименьшим расстоянием. В данном случае это вершина  $B$ . Вычеркнем эту вершину из списка доступных вершин. Обведём её в кружок. Дальнейшие действия будем делать из неё.



10. По таблице расстояний найдём все вершины, до которых идёт ребро из текущей вершины (сейчас это вершина  $B$ ) и которые ещё не вычеркнуты. Это вершина  $C$ . Нарисуем из текущей вершины ( $B_3$ ) ветви дерева для каждой из найденных смежных вершин, нарисуем на концах ветвей названия этих вершин. Подпишем на ветвях длины рёбер по таблице расстояний (сейчас это все делаем для одной вершины —  $C$ ).



11. Для полученной вершины посчитаем расстояние до неё как: расстояние до текущей вершины (подписано рядом с ней, сейчас для вершины  $B$  это 3) плюс расстояние (длины ребра) до полученной вершины (подписано над ребром (4)). Запишем это расстояние рядом с полученной вершиной. Сейчас это  $C_7$ .



<p>12. Среди всех листовых вершин будем искать одинаковые вершины. Сейчас это <math>C_5</math> и <math>C_7</math>. Для каждой пары одинаковых вершин будем вычеркивать в дереве худшую вершину. То есть, вершину с большим расстоянием до неё. Сейчас это <math>C_7</math>.</p>	
<p>13. Среди листовых вершин найдём вершину с наименьшим расстоянием. В данном случае это вершина <math>C</math>. Вычеркнем эту вершину из списка доступных вершин. Как только мы вычеркнули вершину <math>C</math> из списка доступных вершин, закончился первый этап алгоритма. Мы нашли кратчайший путь до вершины <math>C</math>. Это путь <math>A-D-C</math>. Его длина 5.</p>	
<p>14. Начинаем второй этап алгоритма. Выпишем список доступных вершин и сразу вычеркнем из него все вершины, которые используются в пути, найденном на первом этапе. Это вершины <math>A</math>, <math>C</math> и <math>E</math>. Будем рассматривать поиск из вершины <math>C</math>. Напишем рядом с ней расстояние, равное нулю. Обведём её в кружок.</p>	
<p>15. По таблице расстояний найдём все вершины, до которых идёт ребро из текущей вершины (сейчас это вершина <math>C</math>) и которые ещё не вычеркнуты. Это вершины <math>B</math> и <math>E</math>. Нарисуем из текущей вершины (<math>C</math>) ветви дерева для каждой из найденных смежных вершин, нарисуем на концах ветвей названия этих вершин. Подпишем на ветвях длины рёбер по таблице расстояний.</p>	
<p>16. Для каждой полученной вершины посчитаем расстояние до неё как: расстояние до текущей вершины (подписано рядом с ней, сейчас для вершины <math>C</math> это 0) плюс расстояние (длина ребра) до каждой полученной вершины (подписано над ребром). Запишем эти расстояния рядом с каждой полученной вершиной. Сейчас это <math>B_4</math> и <math>E_6</math>.</p>	
<p>17. Среди листовых вершин найдём вершину с наименьшим расстоянием. В данном случае это вершина <math>B_4</math>. Вычеркнем эту вершину из списка доступных вершин. Дальнейшие действия будем делать из неё. Обведём её в кружок.</p>	

<p>18. По таблице расстояний найдём все вершины, до которых идёт ребро из текущей вершины (сейчас это вершина <math>B_4</math>) и которые ещё не вычеркнуты. Это вершина <math>E</math>. Нарисуем из текущей вершины (<math>B_4</math>) ветви дерева для каждой из найденных смежных вершин, нарисует на концах ветвей названия этих вершин. Подпишем на ветвях длины рёбер по таблице расстояний. Сейчас это одно ребро и одна вершина (<math>E</math>).</p>	
<p>19. Посчитаем для найденной вершины (<math>E</math>) расстояние до неё и подпишем рядом с ней (<math>E_5</math>). Среди всех листовых вершин будем искать одинаковые вершины. Сейчас это пара вершин <math>E_5</math> и <math>E_6</math>. Для каждой пары одинаковых вершин будем вычеркивать в дереве худшую вершину. То есть, вершину с большим расстоянием до неё. Сейчас это <math>E_6</math>. Вычеркнем её. Среди листовых вершин найдём вершину с наименьшим расстоянием. В данном случае это вершина <math>E_5</math>. Это конечная вершина, расстояние до которой мы искали. Второй этап алгоритма закончен. Длина кратчайшего пути <math>C-E</math> равна 5. Складываем длины путей первого и второго этапа: <math>5 + 5</math>.</p>	

О т в е т: 10.

**5** У исполнителя Алго две команды, которым присвоены номера:

1. прибавь 1
2. умножь на  $b$

( $b$  — неизвестное натуральное число;  $b \geq 2$ ).

Выполняя первую из них, Алго увеличивает число на экране на 1, а выполняя вторую, умножает это число на  $b$ .

Программа для исполнителя Алго — это последовательность номеров команд.

Известно, что программа 11211 переводит число 8 в число 62.

Определите значение  $b$ .

Ответ: \_\_\_\_\_.

Р е ш е н и е.

Выпишем последовательность действий, которые получают из исходного числа 8 конечное число 62. То есть, вместо каждого номера команды (11211) напишем действие, которое этой команде соответствует. Например, после первого действия (номер 1, действие **прибавь 1**) из числа 8 получится число 9 ( $8 + 1 = 9$ ). Результат каждого действия будем подставлять в качестве аргумента следующего действия:

Исходное число	Номер команды	Действие	Вычисление	Результат
8	1	<b>прибавь 1</b>	$8 + 1$	9
9	1	<b>прибавь 1</b>	$9 + 1$	10
10	2	<b>умножь на <math>b</math></b>	$10 \cdot b$	$10b$
$10b$	1	<b>прибавь 1</b>	$10b + 1$	$10b + 1$
$10b + 1$	1	<b>прибавь 1</b>	$(10b + 1) + 1$	$10b + 2$

Получили, что в результате выполнения программы из числа 8 получилось число  $10b + 2$ . Что по условию задачи даёт число 62. Решим уравнение:  $10b + 2 = 62 \Rightarrow 10b = 60 \Rightarrow b = 6$ .

О т в е т: 6.

**6** Ниже приведена программа, записанная на пяти языках программирования.

Алгоритмический язык	Паскаль
<pre> алг нач цел s, t ввод s ввод t если s&gt;8 или t&gt;8   то вывод "YES"   иначе вывод "NO" все       кон           </pre>	<pre> var s,t: integer; begin   readln(s);   readln(t);   if (s&gt;8) or (t&gt;8)   then     writeln('YES')   else     writeln('NO')   end.           </pre>
Бейсик	C++
<pre> DIM s, t AS INTEGER INPUT s INPUT t IF s &gt; 8 OR t &gt; 8 THEN   PRINT "YES" ELSE   PRINT "NO" ENDIF           </pre>	<pre> #include &lt;iostream&gt; using namespace std; int main() {   int s,t;   cin &gt;&gt; s;   cin &gt;&gt; t;   if (s &gt; 8    t &gt; 8)     cout &lt;&lt; "YES";   else     cout &lt;&lt; "NO";   return 0; }           </pre>
Python	
<pre> s = int(input()) t = int(input()) if s &gt; 8 or t &gt; 8 :   print("YES") else:   print("NO")           </pre>	

Было проведено 9 запусков программы, при которых в качестве значений переменных вводились следующие пары чисел (s, t): (5, 6); (9, 5); (6, 8); (10, 9); (−9, −10); (−9, 10); (−10, 9); (8, 8); (8, 6).

Сколько было запусков, при которых программа напечатала «YES»?

Ответ: \_\_\_\_\_.

**Решение.**

Анализируем программу.

Понимаем, что программа просто проверяет для каждой введённой пары чисел выполнение логического выражения. И, в зависимости от полученного результата, выводит на экран «YES», если значение логического выражения истинно, и «NO», если оно ложно. Рассмотрим результаты проверки этого логического выражения для каждой введённой пары чисел.

Для удобства вычислений и подсчётов результатов будем выполнять вычисления в таблице.

При вычислении последнего выражения ( $s > 8$  or  $t > 8$ ) учитываем правило вычисления операции ИЛИ (or) — результат истина (true), если хотя бы один операнд истина.

Исходные числа		Проверяемое условие			На экране
s	t	$s > 8$	$t > 8$	$s > 8 \text{ or } t > 8$	
5	6	false	false	false	NO
9	5	<b>true</b>	false	<b>true</b>	<b>YES</b>
6	8	false	false	false	NO
10	9	<b>true</b>	<b>true</b>	<b>true</b>	<b>YES</b>
-9	-10	false	false	false	NO
-9	10	false	<b>true</b>	<b>true</b>	<b>YES</b>
-10	9	false	<b>true</b>	<b>true</b>	<b>YES</b>
8	8	false	false	false	NO
8	6	false	false	false	NO

Подсчитываем количество ответов «YES».

О т в е т : 4.

**7** Доступ к файлу **www.txt**, находящемуся на сервере **http.com**, осуществляется по протоколу **ftp**. В таблице фрагменты адреса файла закодированы цифрами от 1 до 7. Запишите последовательность этих цифр, кодирующую адрес указанного файла в сети Интернет.

<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>
.com	/	://	http	www	.txt	ftp

Ответ: \_\_\_\_\_.

**Р е ш е н и е.**

Структура адреса ресурса (файла) в сети Интернет такова:

Название протокола : // адрес сервера / имя файла на сервере

Выпишем последовательность частей адреса в указанном порядке:

Протокол		адрес сервера		имя файла на сервере
ftp	://	http.com	/	www.txt

Воспользуемся таблицей, приведённой в условии, для того, чтобы сопоставить каждому фрагменту его номер:

ftp	://	http	.com	/	www	.txt
7	3	4	1	2	5	6

О т в е т : 7341256.

**8** В языке запросов к поисковому серверу для обозначения логической операции «ИЛИ» используется символ « | », а для обозначения логической операции «И» — символ «&». В таблице приведены запросы и количество найденных по ним страниц некоторого сегмента сети Интернет.