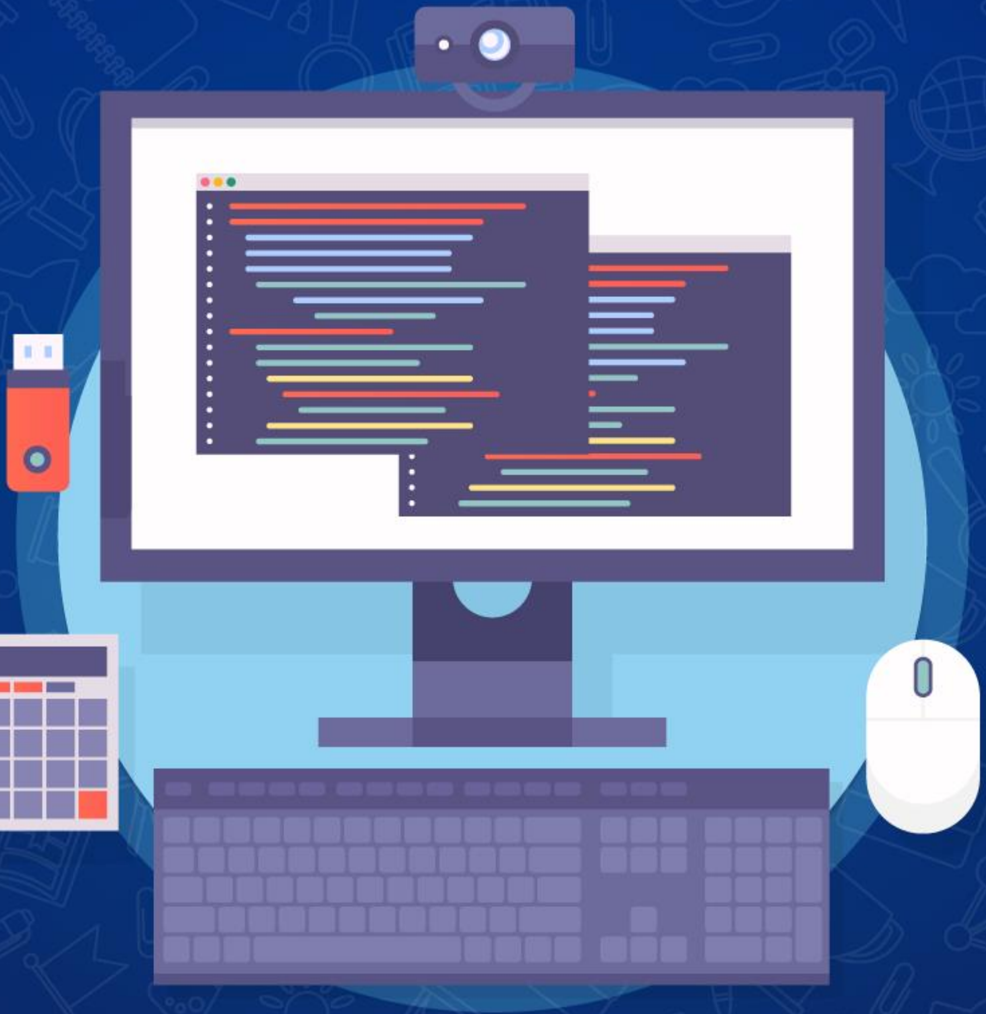
Фоксфорд



ГОТОВИМСЯ К ЕГЭ ПО ИНФОРМАТИКЕ

Пособие для школьников
от преподавателей Фоксфорда

Содержание

Автор: Тимофей Хирьянов

Преподаватель кафедры МФТИ,
эксперт ЕГЭ

О содержании экзамена

Задания ЕГЭ по информатике охватывают не всё содержание курса информатики и ИКТ, а лишь наиболее значимые его темы, которые удобно проверять в задачах, решаемых на бумаге. В частности, в рамках ЕГЭ невозможно проверить навыки пользователя, поэтому они на экзамене не пригодятся. С другой стороны, на некоторых темах составители заданий делают особенный акцент, это: **дискретная математика, алгебра логики, теория игр, программирование.**

Подготовка к экзамену по обычному школьному учебнику позволит вам заработать средние, но не высокие баллы, так как отвлечёт вас на то, что не проверяется на экзамене. Наилучший способ подготовки — это прорешивание задач из ЕГЭ и изучение методов их решения.

В оправдание учебников и школьной информатики скажу, что, будучи идеально готовы к ЕГЭ, вы не станете специалистом в информационных технологиях. За пределами ЕГЭ остаётся много тем, которые очень важны, но «не помещаются» на бумагу.

Экзаменационное задание состоит из двух частей. За каждую задачу первой части (независимо от сложности) можно получить не более 1 первичного балла. Всего их 23, соответственно, максимальный первичный балл за выполнение тестовой части — 23 балла.

Вторая, письменная часть состоит лишь из четырёх задач, но их стоимость 3, 2, 3 и 4 балла, что в сумме составляет 12 первичных баллов.

Решение заданий тестовой части

На выполнение экзаменационной работы отводится 3 часа 55 минут (235 минут), из которых на первую часть **желательно потратить не более полутора часов**, поскольку вторая часть сложнее первой и требует времени.

Навык решения заданий тестовой части стоит **довести до автоматизма,** замеряя при этом время выполнения. Возможно, вы осознаете, что некоторые задания (например, на алгебру логики) выгодней не выполнять совсем или выполнять только после решения задач второй части.

1. Двоичная система счисления и её «родственники»

В этом задании используются двоичная, восьмеричная и шестнадцатеричная системы счисления. Нужно уметь быстро переходить между ними, а также десятичной системой счисления.

Пример задания:

Сколько двоичных цифр содержит запись восьмеричного числа 23705_8?

Решение:

Поскольку восьмеричная и двоичная системы родственные, одна восьмеричная цифра соответствует ровно трём двоичным. Воспользуемся таблицей перевода.

x_{10}	*0*	***1***	***2***	***3***	***4***	***5***	***6***	***7***
x_8	***0***	***1***	***2***	***3***	***4***	***5***	***6***	***7***
x_2	***000***	***001***	***101***	***011***	***100***	***101***	***110***	***111***

23705_8= 010011111000101$_2$

Первый ноль — не значащий, его не нужно считать.

Итого, в двоичной записи данного числа четырнадцать цифр.

> **Замечание**
>
> Можно было не переводить из 8-й в 2-ю систему остальные цифры, кроме первой.
>
> 23705_8= 010 xxx xxx xxx xxx$_2$
>
> Ведь из этого уже ясно, что в пяти восьмеричных цифрах 15 двоичных, одна из которых — незначащий нуль.

2. Логические функции и таблицы истинности

Для решения нужно знать операции алгебры логики, их свойства, таблицы истинности, а также иметь опыт анализа логических выражений.

Пример задания:

Логическая функция F задаётся выражением $(x \land \neg y) \lor (y \equiv z) \lor \neg w$. На рисунке приведён фрагмент таблицы истинности функции F, содержащий все наборы аргументов, при которых функция F ложна. Определите, какому столбцу таблицы истинности функции F соответствует каждая из переменных w, x, y, z. Все строки в представленном фрагменте разные.

Перем.1	**Перем.2**	**Перем.3**	**Перем.4**
???	**???**	**???**	**???**
	0		
1	**0**		**0**
1		**0**	**0**

В ответе напишите буквы w, x, y, z в том порядке, в котором идут соответствующие им столбцы (без разделителей).

Решение:

Рассмотрим, когда именно «F» ложна: поскольку она является дизъюнкцией (логическим ИЛИ) трёх выражений, каждое из них должно быть ложно, иначе бы дизъюнкция была истинна.

Это значит, что $(x \wedge \neg y) = 0$, $(y \equiv z) = 0$, $\neg w = 0$.

Из последнего следует, что $w = 1$, а значит «Перем.1» — это w.

Из второго следует, что в строках, где «F» ложна, «y» не должен быть равен «z». Но во второй строке «Перем.2» = «Перем.4», а в третьей строке «Перем.3» = «Перем.4», откуда можно заключить, что «Перем.4» — это x, поскольку «y» и «z» исключаются.

Учитывая, что «y» и «z» не равны друг другу, а также, что все строки во фрагменте различны, мы можем теперь восстановить все ячейки таблицы:

Перем.1	**Перем.2**	**Перем.3**	**Перем.4**
w	**???**	**???**	**x**
1	**0**	**1**	**1**
1	**0**	**1**	**0**
1	**1**	**0**	**0**

Для определения где «y», а где «z», необходимо рассмотреть выражение $(x \land \neg y) = 0$. Его эквивалент, полученный отрицанием левой и правой части: $\neg x \lor y = 1$.

Интерпретируем это: либо x ложно, либо y истинно, либо и то и то сразу. Исключена ситуация, когда $x = 1$ и $y = 0$, а это значит, что «y» не может быть «Перем.2» (см. первую строку для «Перем.2» и «Перем.4»). Значит «Перем.2» — это «z», а «Перем.3» — это «y».

Ответ: wzyx.

> **Замечание:**
>
> Несмотря на то, что это задание находится в самом начале работы, оно относительно сложное, и его выполнение у многих учеников вызывает затруднения или по крайней мере требует значительного времени. Если сомневаетесь — пропускайте его и делайте в самом конце, т. к. его стоимость лишь 1 первичный балл.

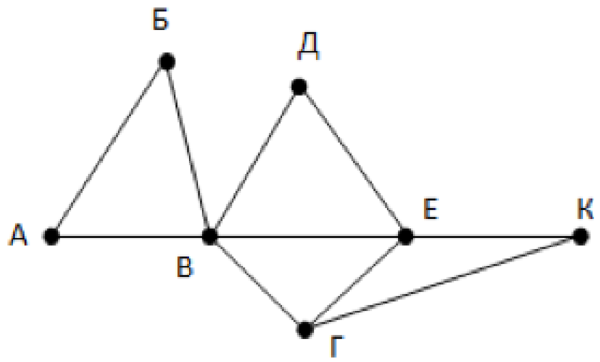
3. Графы и весовые матрицы

Нужно уметь переводить граф из табличной в графическую форму и наоборот. Полезно знать понятие *степень вершины* и как её найти в весовой матрице графа (таблице).

Пример задания:

На рисунке справа схема дорог Н-ского района изображена в виде графа, в таблице содержатся сведения о длинах этих дорог (в километрах). Так как таблицу и схему рисовали независимо друг от друга, то нумерация населённых пунктов в таблице никак не связана с буквенными обозначениями на графе. Определите, какова длина дороги из пункта В в пункт Е. В ответе запишите **целое число** — так, как оно указано в таблице.

	П1	П2	П3	П4	П5	П6	П7
П1		45		10			
П2	45			40		55	
П3					15	60	
П4	10	40				20	35
П5			15			55	
П6		55	60	20	55		45
П7				35		45	



Решение:

По рисунку находим, что степень вершины В равна 5, а степень вершины Е равна 4. В таблице же степень вершины — это количество
непустых клеток в соответствующей строке. Степени для вершин по таблице: П1 — 2, П2 — 3, П3 — 2, П4 — 4, П5 — 2, П6 — 5, П7 — 2.
Поскольку кроме П4 и П6 больше нет вершин со степенями 4 и 5, значит это искомые вершины Е и В.
Смотрим на длину ребра П4-П6 по таблице, и записываем её в **ответ: 20.**

Визуализация родословного дерева поможет не запутаться. Здесь мы её не проводим.

Пример задания:

Во фрагменте базы данных даны сведения о родственных отношениях. Определите на основании приведенных данных идентификатор бабушки Лобановой А.И.

ID	Фамилия_И.О.	Пол
17	Лобанов Т.М.	М
58	Макаревич И.Т.	М
31	Белых У.В.	Ж
42	Макаревич А.И.	Ж
23	Лобанова А.И.	Ж
96	Макаревич В.В.	Ж
82	Белых А.Н.	М
95	Рабинович Н.Т.	Ж
10	Рабинович Н.А.	М
	...	

ID_Родителя	ID_Ребенка
23	17
31	23
58	23
82	31
95	31
58	42
82	10
95	10

Решение:

По первой таблице определяем идентификатор Лобановой А.И. — 23. Далее, чтобы найти родителей работаем со второй таблицей: ищем записи, где код ребенка равен 23.

Итак, родители имеют коды 58 и 31.

Теперь во второй таблице ищем бабушек и дедушек, где идентификатор ребенка равен 58 или 31. Потенциально их может быть четыре, но в таблице находятся только два соответствующих идентификатора — это 82 и 95.

Расшифровываем чьи это ID по первой таблице: Белых А.Н. (мужского пола) и Рабинович Н.Т. (женского пола); последняя и является искомой бабушкой.

Ответ: 95.

5. Неравномерное и помехоустойчивое кодирование

Нужно знать условие Фано и обратное условие Фано однозначного декодирования для неравномерного кодирования. Также полезно уметь рисовать префиксное и суффиксное деревья для кодовой таблицы — это ускоряет решение задачи и делает ответ очевидным.

Пример задания:

По каналу связи передаются сообщения, содержащие только четыре буквы: П, О, С, Т; для передачи используется двоичный код, допускающий однозначное декодирование. Для букв Т, О, П используются такие кодовые слова: Т: 111, О: 0, П: 100.

Укажите кратчайшее кодовое слово для буквы С, при котором код будет допускать однозначное декодирование. Если таких кодов несколько, укажите код с наименьшим числовым значением.

Решение:

Заметим, что код однозначно декодируется, если выполняется условие Фано или обратное условие Фано. В нашем случае выполняется прямое условие, т.к. с кода буквы О (0) не начинается ни один из двух других кодов.

Будем подбирать код так, чтобы по-прежнему прямое условие Фано выполнялось. Заметим, что ни один из односимвольных или двухсимвольных кодов нам не подойдет (0 уже есть, с 1 начинается код буквы Т; 11 и 10 — начала кодов букв Т и П соответственно). Из трехсимвольных наименьших вариант 101.

> **Замечание:**
>
> Задачи на помехоустойчивое кодирование проще, но иногда требуют внимательного анализа вариантов ответа.

6. Анализ алгоритма, записанного на русском языке

Пример задания:

На вход алгоритма подаётся натуральное число N. Алгоритм строит по нему новое число R следующим образом.

1. Строится двоичная запись числа N.
2. К этой записи дописываются справа ещё два разряда по следующему правилу:
   a) складываются все цифры двоичной записи числа N, и остаток от деления суммы на 2 дописывается в конец числа (справа). Например, запись 11100 преобразуется в запись 111001;
   b) над этой записью производятся те же действия — справа дописывается остаток от деления суммы её цифр на 2.

Полученная таким образом запись (в ней на два разряда больше, чем в записи исходного числа N) является двоичной записью искомого числа R. Укажите минимальное число R, которое превышает число 83 и может являться результатом работы данного алгоритма. В ответе это число запишите в десятичной системе счисления.

Решение:

Цифры двоичной записи — это 0 и 1, поэтому «складываются все цифры» означает просто количество единиц. Остаток от деления на 2 — это признак чётности количества единиц.

Таким образом, если в исходном числе их было нечётное количество, то после выполнения пункта «a)» их количество *станет* чётным, а если было чётным, то оно *останется* чётным.

Значит количество единиц в итоговом числе *всегда окажется чётным*, а в пункте «b)» в конце всегда дописывается ноль.
Число R должно *превышать* 83, значит начинаем перебирать числа начиная от 84:
$84 = 64 + 20 = 64 + 16 + 4 = 1010100_2$ — три единицы,
$85 = 1010101_2$ — четыре единицы, но в конце не ноль,
$86 = 1010110_2$ — четыре единицы, в конце ноль.
Минимальное число, которое подходит — 86. **Это ответ.**

7. Адреса в электронных таблицах

Нужно знать как преобразуются ссылки в формулах в зависимости от места постановки значка $. Желательно поиграться с этими задачками в приложении Excel или LibreOffice.org Calc, чтобы увидеть глазами как именно это происходит, но не углубляйтесь в это: **для ЕГЭ прокачивать навыки пользователя табличным процессором** (как это нужно для ОГЭ в 9-м классе**) не имеет смысла**.

Пример задания:

В ячейке B4 электронной таблицы записана формула =$C3*2. Какой вид приобретет формула, после того, как ячейку B4 скопируют в ячейку B6? Примечание: знак $ используется для обозначения абсолютной адресации.

1. =$C5*4
2. =$C5*2
3. =$C3*4
4. =$C3*2

Решение:

Ссылка $C3 — это смешанная ссылка, в которой «заблокирован» только столбец C, а 3 — нет. После того, как ячейку B4 скопировали в B6, номер строки увеличился на 2, поэтому и в ссылке $C3 номер строки (относительная часть) также увеличится на 2, ссылка превратится в $C5 константы при копировании формул не меняются, поэтому получится =$C5*2.

Ответ: 2.

> **Замечание**
>
> Если ошибочно посчитать, что знак доллара защищает от изменений всю ссылку, получим неверный ответ.
>
> Иногда в этой задаче требуется найти неизвестное значение в одной из ячеек по остальным ячейкам и круговой диаграмме. В этом случае нужно уметь составлять и решать систему уравнений.

8. Анализ программы с циклами

Поскольку в задачах с циклами часто встречаются суммы арифметических прогрессий, нужно знать формулу для вычисления n-го члена арифметической прогрессии:

$$a_n = a_1 + d * (n - 1)$$

а также надо знать формулу для суммы первых n членов арифметической прогрессии:

$$S_n = \sum_{i=1}^{n} a_i = a_1 + a_1 + \ldots + a_n = \frac{a_1 + a_n}{2} * n$$

Бывает нужна формула суммы в такой форме:

$$S_n = \frac{2a_1 + d * (n - 1)}{2} * n$$

где a_i — i-ый элемент последовательности, d – шаг (разность) последовательности.

Пример задания:

Определите, что будет напечатано в результате работы следующего фрагмента программы:

```
s = 0
k = 0
while s < 1024:
  s = s + 10
  k = k + 1
print(k)
```

Решение:

Цикл заканчивается, когда нарушается условие s < 1024, то есть длительность цикла определяется переменной s.

В конце выводится значение переменной k.

s изменяется независимо от значений k, с каждым шагом цикла увеличиваясь на 10, а значение k – на единицу, так что фактически k – это счётчик шагов цикла

Начальные значения переменных k и s равны нулю.

Итого, нужно определить число шагов цикла, необходимое для того, чтобы значение s стало не меньше 1024.

Аналитически можно выписать формулу s = 0 + 10*k, где k — номер текущей итерации.

$$s_{fin} \geq 1024 \Rightarrow 10 * k_{fin} \geq 1024$$

$$k_{fin} \geq 103$$

Ответ: 103.

9. Кодирование информации

При выполнении задания часто допускаются ошибки в арифметических вычислениях.

Полезно уметь вычислять произведение и деление не в столбик, а через сокращение степеней у множителей числа. Естественно, нужно уметь раскладывать числа на

множители.
Байт — это 8 бит, то есть 2^3, а вот Кбайт — это не килобайт, а кибибайт (1024 байта), то есть 2^{10} байт или 2^{13} бит.
Степени числа 2 нужно знать наизусть до 10-й включительно.

Пример задания: <u>кодирование звуковой информации</u>

Производится одноканальная (моно) звукозапись с частотой дискретизации 16 кГц и глубиной кодирования 24 бита. Запись длится 1 минуту, её результаты записываются в файл, сжатие данных не производится. Какое из приведенных ниже чисел наиболее близко к размеру полученного файла?

1. 0,2 Мбайт
2. 2 Мбайт
3. 3 Мбайт
4. 4 Мбайт

Решение:

Частота дискретизации 16 кГц, значит за одну секунду запоминается 16000 значений сигнала.
Так как глубина кодирования – 24 бита = 3 байта, для хранения 1 секунды записи требуется 16000 * 3 байта = 48 000 байт (для стереозаписи было бы в 2 раза больше).
На 1 минуту = 60 секунд записи потребуется 60 * 48000 байта = 2 880 000 байт, то есть около 3 Мбайт.
Правильный ответ – 3.

Пример задания: <u>кодирование графической информации</u>

Рисунок размером 512 на 256 пикселей занимает в памяти 64 Кбайт (без учёта сжатия). Найдите максимально возможное количество цветов в палитре изображения.

Решение

Определим количество пикселей в рисунке:

$N = 512 * 256 = 2^9 * 2^8 = 2^{17}$.

Объём файла в Кбайтах

$64 = 2^6$,

объём файла в битах

$2^6 * 2^{13} = 2^{19}$.

Теперь можем определить глубину кодирования (количество битов, выделяемых на 1 пиксель):

$\frac{2^{19}}{2^{17}} = 2^2 = 4$ бита на пиксель. Из комбинаторики (или из формулы Хартли, см. ниже в задании 13), максимальное возможное количество цветов —2^4 = 16.

Ответ:16.

10. Комбинаторика.

Пример задания на перебор слов и системы счисления:

Все 5-буквенные слова, составленные из букв А, О, У, записаны в алфавитном порядке.

<u>Вот начало списка:</u>

1. ААААА
2. ААААО

3. АААAУ
4. АААОА
5. …

Запишите слово, которое стоит на **150-м месте** от начала списка.

Решение:

Если заменить букву **А** на 0, **О** на 1, а **У** на 2, очевидна параллель слов с числами в 3-й системе счисления:

1. 00000_3
2. 00001_3
3. 00002_3
4. 00010_3
5. ...

При этом заметно, что номер числа в списке не совпадает со значением самого числа на 1.

Таким образом на 150-м месте стоит число 149.

Для ответа на вопрос о слове нужно перевести 149 в троичную СС, и записать обратно буквами.

$149 = 12112_3$

Ответ: ОУООУ

11. Анализ программы с рекурсией

Пример задания:

Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:

F(1) = 1

F(n) = $(n - 1) * n$, при n > 1

Чему равно значение функции? F(5)?

В ответе запишите только натуральное число.

Решение

По рекуррентной формуле, находим, что F(5) = F(4) * 5

Применим её еще несколько раз, и получим:

F(5) = F(3) * 4 * 5 = F(2) * 3 * 4 * 5 = F(1) * 2 * 3 * 4 * 5

Мы дошли до крайнего случая, который останавливает рекурсию, так как определяет значение F(1) = 1.

Итого, F(5) = 1 * 2 * 3 * 4 * 5 = 120

Ответ: 120.

12. Сетевые адреса

IP-адрес IPv4 — это 32-разрядное двоичное число (в ЕГЭ встречается только эта версия протокола).

Маска локальной сети — это тоже 32-разрядное двоичное число, которое определяет, какая часть IP-адреса относится к адресу подсети, а какая — к номеру узла в этой подсети.
Старшие биты маски равны 1, а младшие биты — 0. Например, маска подсети может иметь вид:
11111111 11111111 11100000 00000000 (255.255.224.0)

Адрес сети можно узнать, применив «поразрядную конъюнкцию» к IP-адресу и маске (она же — «битовое И» или побитовое умножение). Это значит, что в данном случае 19 старших бит в IP-адресе — это адрес сети, а оставшиеся 13 младших бит — номер компьютера в сети. Маски и IP-адреса обычно записываются в виде четырёх десятичных чисел — побайтно переведённых в десятичную систему счисления.

Пример задания:

По заданным IP-адресу узла сети и маске определите значение третьего слева октета в адресе сети:
IP-адрес: 20.8.248.131
Маска: 255.255.224.0

Решение:

Все части IP-адреса, для которых маска равна 255, входят в IP-адрес сети без изменений, так как $255 = 11111111_2$.

Поскольку $0 = 00000000_2$, все части IP-адреса узла, для которых маска равна 0, в адресе подсети зануляются. Таким образом, адрес сети равен 20.8.X.0, где X как раз и нужно найти.

Переведем в двоичную систему третьи части IP-адреса и маски: $248 = 11111000_2$, $224 = 11100000_2$. Произведём поразрядную конъюнкцию этих чисел, получим, что $x = 11100000_2 = 224$.

13. Объём информации

Нужно знать формулу Хартли информационного объёма в форме $2^K = M$, где M — мощность кодируемого множества, а K — количество бит, необходимое для кодификации одного элемента из множества (например, символа). При этом идёт округление M в большую сторону до ближайшей степени двойки. А вот байты — это всего лишь «коробочки» по 8 бит, и количество бит округлять нужно вверх лишь до ближайшего числа, кратного 8.

Пример задания:

При регистрации в компьютерной системе каждому пользователю выдаётся пароль, состоящий из 10 символов и содержащий только символы из 26-символьного латинского алфавита. В базе данных для хранения сведений о каждом пользователе отведено одинаковое и минимально возможное целое число байт. При этом используют посимвольное кодирование паролей, все символы кодируют одинаковым и минимально возможным количеством бит. Кроме собственно пароля, для каждого пользователя в системе хранятся дополнительные сведения, для чего отведено 14 байт на одного пользователя. Определите объём памяти (в байтах), необходимый для хранения сведений о 5 пользователях.

Решение:

Каждый символ кодируется 5 битами, т. к. $26 < 32 = 2^5$.

10 символов пароля = 50 бит.

Вся информация о пользователе кратна целому количеству байт, доп. информация — тоже, значит пароль тоже занимает целое количество байт. Округляем 50 до 56, т. к. 56 = 7*8, и узнаём, что пароль занимает 7 байт.

Всего на одного пользователя уходит 14+7 = 21 байт.

На пять пользователей — 21*5 = 105 байт. Ответ: 105.

14. Исполнитель

Пример задания: исполнитель Черепашка

Исполнитель Черепашка перемещается на экране компьютера, оставляя след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существуют две команды:

Вперед n, где n – целое число, вызывающая передвижение черепашки на n шагов в направлении движения.

Направо m, где m – целое число, вызывающая изменение направления движения на m градусов по часовой стрелке.

Запись **Повтори 7 [Команда1 Команда2]** означает, что последовательность команд в скобках повторится 7 раз.

Черепашке был дан для исполнения следующий алгоритм:

Повтори 10 [Вперед 10 Направо 36]

Какая фигура появится на экране?

Решение:

Заметим, что сумма углов выпуклого десятиугольника равна $180^o * (10-2) = 180^o * 8 = 1440^o$.Следовательно, внутренний угол правильного десятиугольника равен $1440^o : 10 = 144^o$, а внешний угол равен $180^o - 144^o = 36^o$. Следовательно, исполнитель будет двигаться по контуру правильного десятиугольника.

Пример задания: исполнитель Робот

Система команд исполнителя РОБОТ, «живущего» в прямоугольном лабиринте на клетчатой плоскости:

Вверх-вниз, влево-вправо.

При выполнении любой из этих команд РОБОТ перемещается на одну клетку соответственно: вверх ↑, вниз ↓, влево ←, вправо →.

Четыре команды проверяют истинность условия отсутствия стены у каждой стороны той клетки, где находится РОБОТ:

сверху свободно снизу свободно

слева свободно справа свободно

Цикл

ПОКА < условие >

последовательность команд

КОНЕЦ ПОКА

выполняется, пока условие истинно. В конструкции

ЕСЛИ < условие >

 ТО команда1

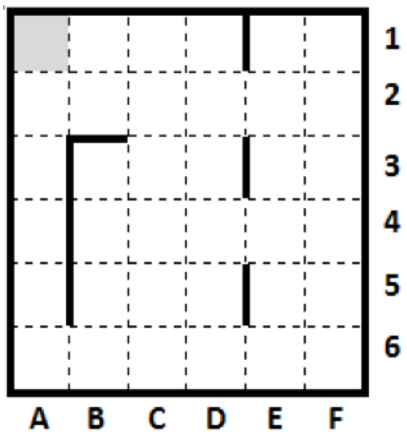
ИНАЧЕ команда2

КОНЕЦ ЕСЛИ

выполняется команда1 (если условие истинно) или команда2 (если условие ложно).

Если РОБОТ начнёт движение в сторону находящейся рядом с ним стены, то он разрушится и программа прервётся.

Сколько клеток лабиринта соответствуют требованию, что, начав движение в ней и выполнив предложенную программу, РОБОТ уцелеет и остановится в закрашенной клетке (клетка A1)?



ПОКА слева свободно ИЛИ сверху свободно
 ЕСЛИ слева свободно
  ТО влево
  ИНАЧЕ вверх
 КОНЕЦ ЕСЛИ
КОНЕЦ ПОКА

Решение:

Заметим, что в этой программе Робот не может разрушиться, так как возможность шага влево проверяется, а если влево ходить нельзя, то можно идти вверх, так как условие цикла "слева свободно ИЛИ сверху свободно" выполнено.

Робот останавливается в клетке, где нарушается условие "слева свободно ИЛИ сверху свободно", в этой клетке должны быть

стенки слева и сверху; таких клеток на поле всего три: A1, B3 и E1. Робот придет в клетку A1, если не попадет в клетку B3 или в клетку E1. Подсчитаем, из какого количества клеток он попадет в B3 или в E1 и вычтем это количество из 36.
Подсчитаем, сколько есть клеток, из которых Робот попадает в клетку B3. Робот сначала идет влево до упора, потом – вверх, пока не упрётся в стенку сверху или не откроется «окно» влево. Это клетки B3, B4, B5, C3, C4, C5, D3, D4, D5, E4, E5, F4, F5, всего 13.

Кроме того, есть две клетки, из которых Робот попадает в E1, это E1 и F1. Следовательно, нужных клеток 36 - 13 - 2 = 21.

Пример задания: исполнитель Чертежник.

Исполнитель Чертёжник перемещается на координатной плоскости, оставляя след в виде линии. Чертёжник может выполнять команду сместиться на (a, b), где a, b – целые числа. Эта команда перемещает Чертёжника из точки с координатами (x, y) в точку с координатами (x + a; y + b). Например, если Чертёжник находится в точке с координатами (4, 2), то команда сместиться на (2, −3) переместит Чертёжника в точку (6, −1).
Цикл
ПОВТОРИ число РАЗ
 последовательность команд
КОНЕЦ ПОВТОРИ
означает, что последовательность команд будет выполнена указанное число раз (число должно быть натуральным). Чертёжнику был дан для исполнения следующий алгоритм (буквами n, a, b обозначены неизвестные числа):
НАЧАЛО

сместиться на (–1, –2)
ПОВТОРИ n РАЗ
 сместиться на (a, b)
 сместиться на (-1, -2)
КОНЕЦ ПОВТОРИ
сместиться на (–24, -12)
КОНЕЦ
Укажите наибольшее возможное значение числа n, для которого найдутся такие значения чисел a и b, что после выполнения программы Чертёжник возвратится в исходную точку.

Решение

В результате выполнения этого алгоритма Чертежник сместился по X на $-1 + n \cdot (a - 1) - 24 = n \cdot (a - 1) - 25$, по Y на -2+ $n \cdot (b - 2)$-12= $n \cdot (b - 2) - 14$. Поскольку Чертёжник должен вернуться в исходную точку, эти величины должны быть равны нулю; следовательно, нужно найти наибольшее натуральное n, при котором система уравнений

$n \cdot (a - 1) = 25,$

$n \cdot (b - 2) = 14.$

разрешима в целых числах относительно a и b.

Для этого число n должно быть одновременно делителем чисел 14 и 25, наибольший общий делитель чисел 14 и 25 равен 1.

Ответ: 1.

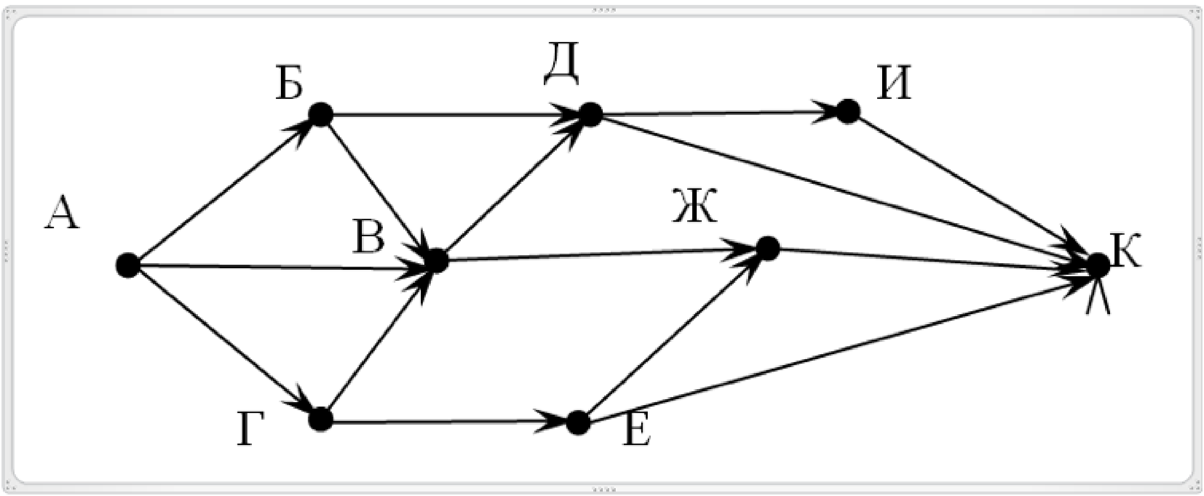
15. Количество траекторий в орграфе

Рассмотрим задачу, в которой дан ориентированный граф и идет речь о количестве путей из одного города в другой. Для решения задач такого типа используют следующее соображение.

Если в город Т можно приехать только из городов X, Y, и Z, то число различных путей из города А в город Т равно сумме числа различных путей проезда из А в X, из А в Y и из А в Z, то есть $N_{\mathrm{T}} = N_X + N_Y + N_Z$, где N_i обозначает количество путей из вершины А в некоторую вершину i. Если бы в графе дорог были замкнутые пути — циклы, то количество различных путей было бы бесконечно, поэтому циклов в этих задачах не бывает.

Пример задания:

На рисунке изображена схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, И, К. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько существует различных путей из города А в город К, если по пути обязательно нужно заехать в город В?



Решение:

Путь от А до К разбивается на два куска: путь от А до В и путь от В до К. Количество путей доехать из А в К равно произведению числа X — количества путей из А в В и числа Y — количества путей из В в К. Числа X и Y нужно найти отдельно.

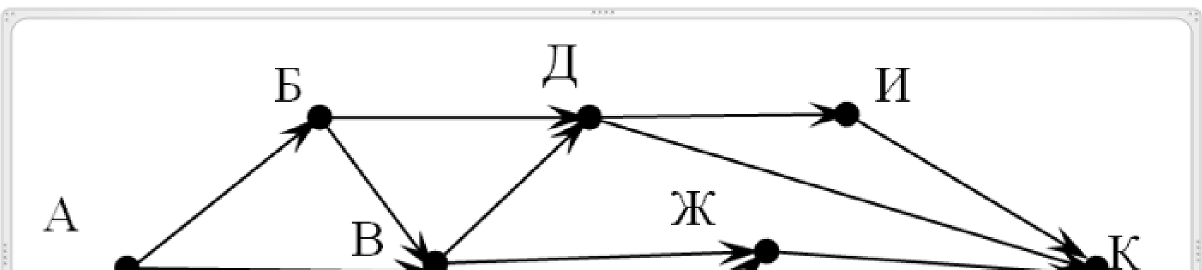
Найдём X: количество способов попасть из А в А равно 1, количество способов попасть из А в Б равно 1, количество способов попасть из А в Г равно 1, количество способов попасть из А в В равно 1 + 1 + 1 = 3.

Найдём Y. Теперь В считается начальной точкой и рассматриваются пути из В в К. Количество способов попасть из В в А, Б или Г равно 0. Количество способов из В попасть в В равно 1. Количество способов попасть из В в Д равно 1, количество способов попасть из В в Ж равно 1, количество способов попасть из В в Е равно 0, количество способов попасть из В в И равно 1. Количество способов попасть из В в К равно 1 + 1 + 1 + 0 = 3.

Следовательно, попасть из А в К, проездом через В можно 3* 3= 9 способами.

Пример задания:

На рисунке изображена схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, И, К. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько существует различных путей из города А в город К, если в город Ж заезжать не разрешается?



Решение:

Раз в город Ж заезжать не разрешается, удалим его вместе со всеми входящими и выходящими ребрами с рисунка. А затем подсчитаем количество путей из А в К.

Количество способов проехать из города А в город А равно 1 — есть только один способ, никуда не поехать. Количество способов попасть в город Б (и в город Г) равно 1. Количество способов попасть в город В равно 1 + 1 + 1 = 3. Количество способов попасть в город Д равно 1 + 3 = 4. Количество способов попасть в город Е равно 1.
Количество способов попасть в город И равно 4. Количество способов попасть в город К равно 4 + 4 + 1 = 9.

16. Позиционные системы счисления

Классификация задач

1. Сумма и разность чисел 2, 4 и 8 в больших степенях.
2. Связь 3-ой и 9-ой систем счисления.
3. Запись ответа уравнения в заданной системе счисления.

4. Поиск всех систем счисления, где число заканчивается на определённую цифру.
5. Поиск основания системы счисления по длине записи числа в ней.

Пример задания №1

Сколько единиц сдержится в двоичной записи значения выражения: $4^{2014} + 2^{2015} - 8$?

Решение

Данные числа настолько велики, что прямые вычисления можно осуществить только при помощи компьютера. Например, bin(4**2014 + 2**2015 - 8).count('1') в интерпретаторе Python 3.

Однако, в самом условии задачи есть подсказка: нужно использовать двоичную систему счисления **до того**, как осуществлять операции сложения и вычитания.

$4^{2014} = (2^2)^{2014} = 2^{(2*2014)} = 2^{4028} = 10\ldots0_2$, где после одной единицы 4028 нулей;
$2^{2015} = 10\ldots0_2$, где после одной единицы 2015 нулей;
$8 = 1000_2$.
Рассмотрим как вычитаются круглые числа в двоичной системе счисления:

$$\begin{array}{rrrrr} 1 & 0 & 0 & 0 & 0_2 \\ - & & & 1 & 0_2 \\ \hline & 1 & 1 & 1 & 0_2 \end{array}$$

У первого числа происходит заём из старшего разряда, и получается $10000_2 - 10_2 = 1110_2$.
Таким образом, $2^M - 2^N$, где M>N записывается в M разрядов, первые M-N которых единицы, а N — нули.
Итого, $2^{2015} - 2^3$ будет записываться 2015-3 = 2012 единицами и тремя нулями.
2^{4028} при сложении с этим числом не пересечётся с ним ни в одном разряде, и получится число, содержащее 2012+1 = 2013 единиц.
Ответ: 2013

Пример задания №2

Значение арифметического выражения: $9^8 + 3^5 - 9$ записали в системе счисления с основанием 3. Сколько цифр «2» содержится в этой записи?

Решение

Приведем все слагаемые к виду 3^x:
$9^8 + 3^5 - 9 = (3^2)^8 + 3^5 - 3^2 = 3^{16} + 3^5 - 3^2$.
Первое слагаемое при переводе в троичную систему даст одну единицу. То есть, нужно рассмотреть только выражение $3^5 - 3^2$ и узнать, сколько двоек в его записи в троичной системе счисления.
Можно непосредственно вычислить в десятичной системе счисления значение этого выражения и потом перевести его в троичную, однако в процессе вычислений нетрудно ошибиться в счёте.
Поэтому поступим иначе. $3^5 - 3^2 = 100000_3 - 100_3$. При вычитании два последних разряда не изменятся, там будут нули.

А заем будет происходить в трёх разрядах, то есть, в итоге мы получим три двойки.

Ответ: 3.

Пример задания №3

Решите уравнение $243_5 - x = 58_9$. Ответ запишите в четверичной системе счисления. Основание системы счисления указывать не нужно.

Решение:

Переведем все числа в десятичную систему счисления, решим уравнение и ответ переведем в четверичную систему.

$243_5 = 2 * 5^2 + 4 * 5^1 + 3 * 5^0 = 73,$

$58_9 = 5 * 9^1 + 8 * 9^0 = 53.$

Получили уравнение 73-x=53, откуда $x = 20_{10} = 110_4$.

Ответ: 110.

Пример задания №4

Укажите через запятую в порядке возрастания все основания систем счисления, в которых запись числа 31 оканчивается на 4.

Решение

Пусть N — основание искомой системы счисления. Тогда из алгоритма перевода числа из десятичной системы счисления в систему с основанием N следует, что 4 — это остаток от деления 31 на N и N > 4. Запишем 31 в виде 31 = k* N + 4, где k = 0, 1, 2, ... — любое целое неотрицательное число. Тогда 31-4=27=k*N, то есть, возможное значение N является одним из делителей числа 27. Все возможные делители: 1, 3, 9, 27. Из них нам подходят 9 и

27. Осталось не забыть записать их в ответе в порядке возрастания.

Пример задания №5

Запись числа 256 в системе счисления с основанием N содержит 3 цифры и оканчивается на 4. Чему равно минимально возможное основание системы счисления?

Решение

Из алгоритма перевода числа из десятичной системы счисления в систему с основанием N следует, что 4 — это остаток от деления 256 на N и N > 4. Запишем 256 в виде 256 = k* N + 4, где k = 0, 1, 2, ...— любое целое неотрицательное число. Тогда 256-4=252=k*N, то есть, возможное значение N является одним из делителей числа 252 = 2* 2* 3* 3* 7.

Кроме того, воспользуемся таким соображением: поскольку запись числа в новой системе счисления содержит три цифры, то $N^2 \leq 256 < N^3$.

При N=2, N=3, N=4, N = 6 это неравенство неверно, а при N=7 верно, следовательно, искомое значение N=7.

17. Поиск в сети и диаграммы Эйлера

Пример задания

Некоторый сегмент сети Интернет состоит из 1000 сайтов. Поисковый сервер в автоматическом режиме составил таблицу ключевых слов для сайтов этого сегмента. Вот ее фрагмент:

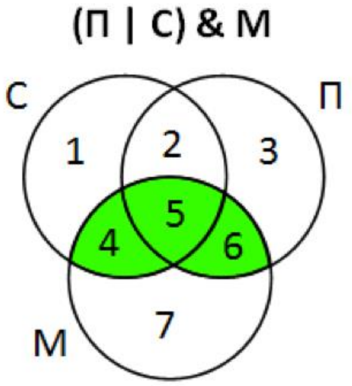
Ключевое слово	**Количество сайтов**
сканер	200
принтер	250
монитор	450

Сколько сайтов будет найдено по запросу

(принтер | сканер) & монитор, если по запросу

принтер | сканер было найдено 450 сайтов, по запросу *принтер & монитор* – 40, а по запросу *сканер & монитор* – 50.

Решение



Обозначим высказывания через С, П, М и нарисуем эти области виде кругов Эйлера.

Запросу (П | С) & М соответствует объединение областей 4, 5 и 6.

Количество сайтов, удовлетворяющих запросу в области i, будем обозначать через N_i

Уравнения, которые определяют запросы:

- сканер $N_1 + N_2 + N_4 + N_5 = 200$
- принтер $N_2 + N_3 + N_5 + N_6 = 250$
- принтер | сканер $N_1 + N_2 + N_4 + N_5 + N_3 + N_6 = 450$

Из первого и третьего уравнений сразу следует

$200 + N_3 + N_6 = 450$ => $N_3 + N_6 = 250$

Из второго уравнения

$N_2 + N_5 + 250 = 250$ => $N_2 + N_5 = 0$

Поскольку количество сайтов не может быть отрицательной величиной, $N_2 = N_5 = 0$

Учитываем, что $N_5 = 0$:

принтер & монитор $N_5 + N_6 = 40$ => $N_6 = 40$

сканер & монитор $N_4 + N_5 = 50$ => $N_4 = 50$

В результате:

(принтер | сканер) & монитор $N_4 + N_5 + N_6 = N_4 + N_6 = 40 + 50 = 90$.

Ответ: 90.

18. Сложное логическое выражение

Эта задача требует знаний не только логики, но и математики, так как часто сводится к системам и совокупностям неравенств. Нужно понимать связь операций между операциями алгебры логики и операциями над множествами, являющимися областями истинности выражений.

Ранее в задании 18 встречалась «поразрядная конъюнкция», но из-за чрезмерной сложности таких задач, их в ЕГЭ более не используют. Задачи с делимостью и множествами по-прежнему возможны.

Пример задания:

Укажите наименьшее целое число A, при котором выражение

$$(y+3x < A) \lor (x > 20) \lor (y > 40)$$

истинно для любых целых положительных значений x и y.

Решение:

Данное логическое выражение истинно для любых x, больших 20, а также для любых y, больших 40. Это значит, что область ложности выражения $(x > 20) \lor (y > 40)$ — это левая нижняя четверть декартовой плоскости от точки M = (20, 40).

Область истинности y+3x < A — это полуплоскость под наклонной прямой y = A-3x.

Для того, чтобы область истинности исходного выражения покрывала всю декартову плоскость (а именно это и означает фраза истинно для любых целых положительных значений x и y) Нужно найти такое число А, чтобы точка М была не просто на этой прямой, а ниже её. То есть: 40 < A-3*20.
Отсюда A > 40+3*20, значит A > 100.

И **ответ:** 101.

19. Обработка массива

Пример задания:

В программе используется одномерный целочисленный массив A с индексами от 0 до 9. Ниже представлен фрагмент программы, записанный на разных языках программирования, в котором значения элементов сначала задаются, а затем меняются.

Python	**Pascal**

for i in range(10):
  A[i] = 9 - i
for i in range(4):
  k = A[i]
  A[i] = A[9 - i]
A[9 - i] = k
	for i := 0 to 9 do
	  A[i] := 9 - i;
	for i := 0 to 4 do begin
	  k := A[i];
	  A[i] := A[9 - i];
	  A[9 - i] := k;
	end;

Чему будут равны элементы этого массива после выполнения фрагмента программы?

1. 9 8 7 6 5 4 3 2 1 0
2. 0 1 2 3 4 5 6 7 8 9
3. 9 8 7 6 5 5 6 7 8 9
4. 0 1 2 3 4 4 3 2 1 0

Решение

выясним, как заполняется массив в первом цикле

for i:=0 to 9 do

A[i]:=9-i;

здесь элемент A[0] равен 9, элемент A[1] = 8, ..., A[9] = 0.

Во втором цикле операторы

k := A[i];

A[i] := A[9-i];

A[9-i] := k;

меняют местами элементы A[i] и A[9-i]. Второй цикл выполняется всего 5 раз, то есть останавливается ровно на половине массива.

for i := 0 to 4 do begin

...

end;

таким образом в нем меняются элементы A[0] и A[9], A[1] и A[8], A[2] и A[7], A[3] и A[6] и A[4] и A[5]. В результате массив оказывается "развернут" наоборот, элемент A[0] (он был равен 9) стал последним, следующий (A[1] = 8) – предпоследним и так далее, то есть получили

0 1 2 3 4 5 6 7 8 9

Ответ:2.

20. Анализ программы с циклами

Пример задания:

Получив на вход число x, эта программа печатает два числа, L и M. Укажите наибольшее из таких чисел x, при вводе которых алгоритм печатает сначала 3, а потом 7.

Python	**Pascal**

<pre>x = int(input())
L = 0
M = 0
while x > 0:
        L = L+1
        if M < x % 10:
        M = x%10
        x = x//10
print(L, M)</pre>
	<pre>var x, L, M: integer;
	begin
	  readln(x);
	  L:=0; M:=0;
	  while x > 0 do begin
	        L:=L+1;
	        if M < (x mod 10) then begin
	        M:=x mod 10;
	        end;
	        x:= x div 10;
	  end;
	  writeln(L);
	  write(M);
	end.</pre>

Решение

В данном цикле while у числа x постепенно «откусывают» младшую цифру операцией x:= x div 10, пока число не занулится.

При этом с каждой итерацией каждая цифра (x mod 10) участвует в алгоритме поиска максимума if M < (x mod 10) then M:=x mod 10, а значит в переменной M окажется наибольшая цифра числа x.

Переменная L просто считает количество итераций, на каждой из которых «откусывается» цифра. Итого в ней окажется количество цифр числа.

Итак, нам известно, что цифр у числа — три, а максимальная цифра — 7. Наибольшее из таких чисел — это 777.
Ответ: 777.

21. Анализ программы с циклами, условиями и подпрограммами

Научимся искать максимума и минимум функции на отрезке. Следующая программа ищет наименьшее значение функции F(x) на интервале [a,b], просматривая значения от a до b с шагом 1:

Python	**Pascal**

M = a
R = F(a)
for t in range(a, b+1):
    if F(t) < R:
        R = F(t)
        M = t
	M := a;
	R : = F(a);
	for t := a to b do
	 if F(t) < R then begin
	   R := F(t);
	   M := t;
	  end;

Цикл для поиска наибольшего значения выглядит точно так же, только знак < нужно заменить на знак >.

Если функция представляет собой квадратный трехчлен вида $F(x) = ax^2 + bx + c$, то положение точки минимума по оси x вычисляется по формуле

$$x_{min} = \frac{-b}{2a}$$

Если парабола задана в виде $F(x) = a(x - p)(x - q)$,

то по формуле $x_{min} = \frac{p+q}{2}$.

Кроме точки экстремума на максимум/минимум нужно проверить крайние точки промежутка.

Пример задания

Напишите в ответе количество различных значений входной переменной a из интервала от 1 до 100 (включая границы), при которых программа выдаёт тот же ответ, что и при входном значении a = 20. Значение a = 20 также включается в подсчёт различных значений a.

Python	Pascal

<pre>def f(x):
  if x > 1:
    return x % 2 * f(x // 2)
  else:
    return x
k = 0
a = int(input())
for i in range(1, a + 1):
  if f(i) == 1:
    k = k + 1
print(k)</pre>
	<pre>var i, k,a: integer;
	function f(x: integer): integer;
	begin
	  if x > 1 then
	     f := x mod 2 * f(x div 2)
	  else
	    f := x;
	end;
	begin
	 k := 0;
	 readln(a);
	 for i := 1 to a do
	      if f(i) = 1 then
	          k := k+1;
	 writeln(k);
	end.</pre>

Решение:

Рассмотрим, как работает функция, приведенная в программе. Заметим, что x mod 2 (на языке Python: x % 2) – последняя (младшая) цифра двоичного представления числа x в двоичной системе счисления, x div 2 (на языке Python: x // 2) – число x без последней цифры в двоичном представлении.

Таким образом, функция находит произведение цифр числа в двоичном представлении. Программа в целом находит количество чисел, произведение цифр в двоичной записи которых равно 1, то есть таких чисел, двоичная запись которых не содержит нулей.

В общем виде такие числа можно представить, как 2^n-1, где n – натуральное число. В диапазоне от 1 до 20 таких чисел четыре: 1, 3, 7, 15. А в диапазоне от 1 до 100 —шесть: 1, 3, 7, 15, 31, 63. Таким образом, искомые числа — это числа от 15 до 30.

Ответ:16.

22. Динамическое программирование

Задача про исполнитель Калькулятор очень похожа своей логикой на задачу 15 про количество траекторий между двумя городами на орграфе. И там, и тут правильно двигаться от начала к концу, расширяя наше знание о количестве допустимых траекторий в заданную точку.

Пример:

Исполнитель Калькулятор преобразует число на экране. У исполнителя есть две команды, которым присвоены номера:

1. Прибавить 1
2. Умножить на 2

Программа для исполнителя Калькулятор — это последовательность команд. Сколько существует программ, для которых при исходном числе 1 результатом является число 21, при этом траектория вычислений содержит число 10 и не содержит число 17?

Решение

Составим таблицу для чисел N от 1 до 10, в которой буквой K назовём количество «траекторий» (различных программ) из точки 1 в точку N. Для чётных «городов» есть «входящая дорога» не только из предыдущего числа (командой 1), но и из «города», вдвое меньшего по значению (командой 2). В этом случае количества «траекторий» будем складывать.

N	1	2	3	4	5	6	7	8	9	10
$K_{1 \to N}$	1	2	2	4	4	6	6	10	10	14

Запомним число $K_{1 \to 10}$=14, ведь именно 10 город обязательно требуется посетить.

Составим вторую таблицу для чисел N от 10 до 21, вписав ноль в $K_{1 \to 17}$, ведь «город» 17 недоступен для посещения. Заметим, что команда 2 вплоть до числа 20 не может быть использована, поскольку мы начинаем от обязательного «города»-числа 10.

N	10	11	12	13	14	15	16	17	18	19	20	21
$K_{1 \to N}$	14	14	14	14	14	14	14	0	0	0	14	14

Ответ: 14.

23. Системы логических уравнений

Эта задача стоит 1 первичный балл, при этом её сложность выше, чем у большинства задач второй части. Если вы не планируете набрать более 90 баллов на ЕГЭ, к ней можно даже не готовиться, а сразу пропускать. Рекомендую по крайней мере отложить её решение на самый конец экзамена.

Нужно знать: свойства логических операций и их таблицы истинности, логические тождества, метод битовых цепочек, метод отображений (это динамическое программирование в применении к количеству решений подсистемы).

Также нужно уметь осуществлять замену логических переменных и аккуратно возвращаться из неё.

Критически важны правильные арифметические вычисления.

Нужно пользоваться правилами комбинаторики сложения и умножения.

Пример задания:

Сколько существует различных наборов значений логических переменных x1, x2, ... x8, которые удовлетворяют всем перечисленным ниже условиям?

$(x_1 \vee \neg x_2) \rightarrow (x_3 \vee \neg x_4) = 1$

$(x_3 \lor \neg x_4) \rightarrow (x_5 \lor \neg x_6) = 1$
$(x_5 \lor \neg x_6) \rightarrow (x_7 \lor \neg x_8) = 1$

Решение:

Для данной конкретной задачи подходит замена переменных. Пусть $y_1 = (x_1 \lor \neg x_2)$, $y_2 = (x_3 \lor \neg x_4)$, $y_3 = (x_5 \lor \neg x_6)$, $y_4 = (x_7 \lor \neg x_8)$. Система сводится к:

$y_1 \rightarrow y_2 = 1$
$y_2 \rightarrow y_3 = 1$
$y_3 \rightarrow y_4 = 1$

Из истины может следовать только истина, поэтому из предположения, что $y_1 = 1$, следует, что $y_2 = 1$, что в свою очередь требует $y_3 = 1$ и $y_4 = 1$. Одно решение найдено.

Рассматривая противоположное предположение, что $y_1 = 0$, мы получаем, что y_2допустимо равняться как 0, так и 1.

Получается, что в рамках варианта $y_1 = 0$ требуется рассмотреть про y_2оба предположения. Вариант $y_2 = 1$ как следствие приводит к $y_3 = 1$ и $y_4 = 1$. Вариант $y_2 = 0$ оставляет y_3свободу. Рассмотрим варианты для y_3: $y_3 = 1$ приводит к $y_4 = 1$. А вариант $y_3 = 0$ оставляет y_4возможность равняться 0 или 1.

y_1	y_2	y_3	y_4
1	1	1	1
0	1	1	1
0	0	1	1
0	0	0	1
0	0	0	0

Итого, по Y получилось 5 решений.

Но для X решений будет больше, ведь $y_1 = (x_1 \lor \neg x_2)$ может быть приведён к истине тремя комбинациями x_2и x_2, и то же самое для других y_i и соответствующих им иксов. Ложности y_i можно достичь лишь одной комбинацией двух соответствующих ему иксов.
Количества комбинаций соответствующих иксов:

y_1	y_2	y_3	y_4
3	3	3	3
1	3	3	3
1	1	3	3
1	1	1	3
1	1	1	1

По правилам комбинаторики количество решений по X будет 3*3*3*3 + 1*3*3*3 + 1*1*3*3 + 1*1*1*3 + 1*1*1*1.

Ответ: 121.

Решение заданий второй части

24. Исправление ошибок в программе

Эта задача стоит целых 3 первичных балла! Она очень проста в оформлении: в идеальном ответе может не быть ни одного русского слова. Обязательно попытаться решить!

Ошибок всегда ровно две, и они всегда логические, а не синтаксические. Есть особенность критериев проверки: за указание верной строки в качестве ошибочной (в п. 3) снимают балл. Лучше указать только одну ошибку, чем наугад «найти» вторую.

Пример задания:

На вход программы поступают 4 неотрицательных целых числа, не превышающие 1000, среди которых могут быть одинаковые. Нужно написать программу, которая выводит количество чисел, не кратных 3, и максимальное из этих чисел. Если среди входных данных нет чисел, не кратных трёх, программа должна вывести слово «NO».

Программист написал программу неправильно.

Последовательно выполните следующее:

1. Напишите, что выведет эта программа при вводе чисел 7 15 8 21.

2. Приведите пример такой последовательности, содержащей число, кратное 3, при вводе которой программа выведет правильный ответ.
3. Найдите все ошибки в этой программе (их может быть одна или несколько). Известно, что каждая ошибка затрагивает только одну строку и может быть исправлена без изменения других строк. Для каждой ошибки:
   1) выпишите строку, в которой сделана ошибка;
   2) укажите, как исправить ошибку, т.е. приведите правильный вариант строки.

<table>
<tr><th>Python</th><th>Pascal</th></tr>
<tr><td>

```
count = 0
maximum = 1000
for i in range(4):
    x = int(input())
    if x % 3 != 0:
        count = count + 1
        if x > maximum:
            maximum = i
if count > 0:
    print(count)
    print(maximum)
else:
    print("NO")
```

</td><td>

```
var i, x: integer;
var maximum, count: integer;
begin
 count := 0;
 maximum := 1000;
 for i:=1 to 4 do begin
   read( x );
   if x mod 3 <> 0 then
   begin
     count := count + 1;
     if x > maximum then
       maximum := i;
   end;
 end;
 if count > 0 then begin
   writeln(count);
   writeln(maximum);
 end
 else
   writeln('NO');
end.
```

</td></tr>
</table>

Решение:

Удобно вначале найти ошибки и интерпретировать что делает эта «неправильная программа», и только затем отвечать на вопросы 1 и 2.

Видим, что в переменной maximum изначально записано максимальное значение, а значит данный алгоритм никогда не зайдет в условие изменения максимума. Если же исправить эту инициализацию, то в maximum будет хранится не само число, а его номер при считывании.

Итак, программа сначала выводит количество чисел, не кратных трём, потом *всегда* выводит 1000. Однако, при отсутствии в ряду чисел, не кратных трём, программа работает корректно.

Запись решения в бланке:

1. 2
   1000
2. 2 3 4 1000
3.

	неверно	**верно**
1)	maximum := 1000;	maximum := 0;
2)	maximum := i;	maximum := x;

25. Написать фрагмент программы

Это задание стоит 2 первичных балла. Обратите внимание, что не нужно переписывать всю программу в бланк — вы пишете только то, что должно оказаться на месте многоточия. По-русски ничего в бланке писать не нужно.
В этой задаче синтаксические ошибки вам простят в любом количестве, но за каждую логическую будут отнимать балл.
Самые частые ошибки: нет инициализации переменной-счётчика, нет вывода ответа на экран.

Пример задания:
Дан целочисленный массив из 40 элементов. Элементы массива могут принимать целые значения от 0 до 10 000 включительно. Опишите на естественном языке или на одном из языков программирования алгоритм, позволяющий найти и вывести количество пар элементов массива, в которых десятичная запись хотя бы одного числа оканчивается на 2.

В данной задаче под парой подразумевается два подряд идущих элемента массива. Например, для массива из пяти элементов: 16 3 142 55 22 – ответ: 3.

Исходные данные объявлены так, как показано ниже на примерах для некоторых языков программирования и естественного языка. Запрещается использовать переменные, не описанные ниже, но разрешается не использовать некоторые из описанных переменных.

Python	Pascal

a = []
n = 40
for i in range(0, n):
    a.append(int(input()))
...
	const N=40;
	var a: array [1..N] of integer;
	   i, j, k: integer;
	begin
	 for i:=1 to N do
	readln(a[i]);
	  ...
	end.

Решение:

Записываем в переменную K начальное значение, равное 0. В цикле от первого элемента до предпоследнего находим остаток от деления текущего и следующего элементов массива на 10. Если первый или второй из полученных остатков равен 2, увеличиваем переменную K на единицу. После завершения цикла выводим значение переменной K.

Запись решения на Pascal

```
k := 0;
for i := 1 to N - 1 do
   if (a[i] mod 10 = 2) or (a[i + 1] mod 10 = 2) then
      inc(k);
writeln(k);
```

Запись решение на Python

```
k = 0
for i in range(0, n – 1):
   if (a[i] % 10 == 2 or a[i + 1] % 10 == 2):
      k += 1
print(k)
```

26. Теория игр

Задача стоит 3 первичных балла.

Выигрышной позицией является такая, из которой можно реализовать выигрышную стратегию.

Проигрышной позицией — такая, из которой выигрышную стратегию реализовать невозможно (если противник не дурак, то есть, играет наилучшим для себя образом).

Рассуждения о проигрышных и выигрышных позициях удобнее, чем разговоры о выигрышных и проигрышных ходах.

Как анализировать позицию:

1. Если все возможные ходы из неё ведут *только* в выигрышные позиции, значит данная позиция — *проигрышная*.
   (Что ты ни делай, как ни ходи — ты подыграешь противнику, дашь ему возможность реализовать выигрышную стратегию.)
2. Если есть *хотя бы один* ход, который ведёт в проигрышную позицию, то данная позиция — *выигрышная*.

Классические задачи в ЕГЭ на теорию игр — про кучу или две кучи камней. Задача про одну кучу проще (в ней меньше позиций), про две — сложнее. В 2017 году провели эксперимент с задачей на игру в слова, но она оказалась слишком простой, и от неё отказались. В 2018 году вернулись к задаче с двумя кучами камней.

Пример задания:

Два игрока, Петя и Ваня, играют в следующую игру. Перед игроками лежат две кучи камней. Игроки ходят по очереди, первый ход делает Петя. За один ход игрок может добавить в одну из куч (по своему выбору) один камень или увеличить количество камней в куче в два раза. Например, пусть в одной куче 10 камней, а в другой 7 камней; такую позицию в игре будем обозначать (10, 7). Тогда за один ход можно получить любую из четырёх позиций: (11, 7), (20, 7), (10, 8), (10, 14). Для того чтобы делать ходы, у каждого игрока есть неограниченное количество камней.

Игра завершается в тот момент, когда суммарное количество камней в кучах становится не менее 73. Победителем считается игрок, сделавший последний ход, т.е. первым получивший такую позицию, что в кучах всего будет 73 камня или больше.

Будем говорить, что игрок имеет выигрышную стратегию, если он может выиграть при любых ходах противника. Описать стратегию игрока – значит описать, какой ход он должен сделать в любой ситуации, которая ему может встретиться при различной игре противника. Например, при начальных позициях (6, 34), (7, 33), (9, 32) выигрышная стратегия есть у Пети. Чтобы выиграть, ему достаточно удвоить количество камней во второй куче.

Задание 1. Для каждой из начальных позиций (6, 33), (8, 32) укажите, кто из игроков имеет выигрышную стратегию. В каждом случае опишите выигрышную стратегию; объясните, почему эта стратегия ведёт к выигрышу, и укажите, какое наибольшее количество ходов может потребоваться победителю для выигрыша при этой стратегии.

Задание 2. Для каждой из начальных позиций (6, 32), (7, 32), (8, 31) укажите, кто из игроков имеет выигрышную стратегию. В каждом случае опишите выигрышную стратегию; объясните, почему эта стратегия ведёт к выигрышу, и укажите, какое наибольшее количество ходов может потребоваться победителю для выигрыша при этой стратегии.

Задание 3. Для начальной позиции (7, 31) укажите, кто из игроков имеет выигрышную стратегию. Опишите выигрышную стратегию; объясните, почему эта стратегия ведёт к выигрышу, и укажите, какое наибольшее количество ходов может потребоваться победителю для выигрыша при этой стратегии. Постройте дерево всех партий, возможных при указанной Вами выигрышной стратегии. Представьте дерево в виде рисунка или таблицы.

Решение:

Задание 1

В начальных позициях (6, 33), (8, 32) выигрышная стратегия есть у Вани. При начальной позиции (6, 33) после первого хода Пети может получиться одна из следующих четырёх позиций: (7, 33), (12, 33), (6, 34), (6, 66). Каждая из этих позиций содержит менее 73 камней. При этом из любой из этих позиций Ваня может получить позицию, содержащую не менее 73 камней, удвоив количество камней во второй куче. Для позиции (8, 32) после первого хода Пети может получиться одна из следующих четырёх позиций: (9, 32), (16, 32), (8, 33), (8, 64). Каждая из этих позиций содержит менее 73 камней. При этом из любой из этих позиций Ваня может получить позицию, содержащую не менее 73 камней, удвоив количество камней во второй куче. Таким образом, Ваня при любом ходе Пети выигрывает своим первым ходом.

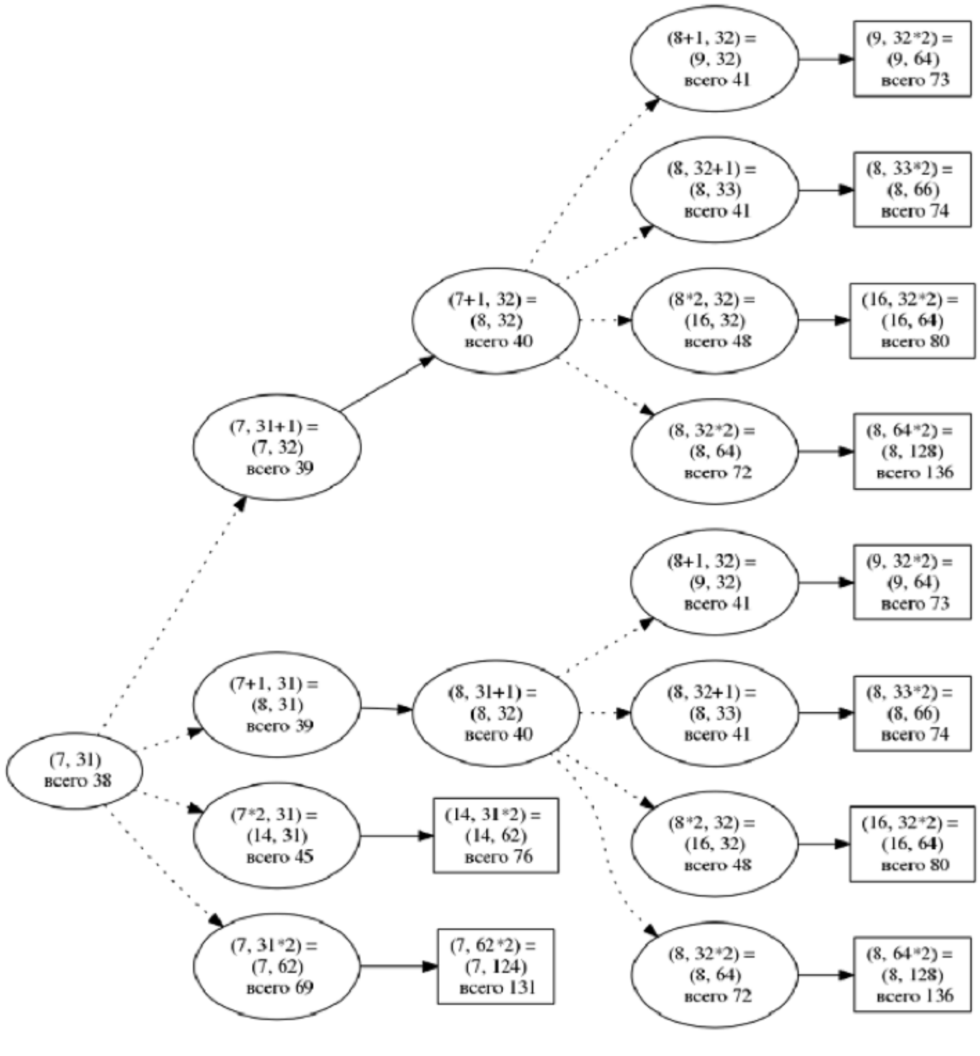
Задание 2

В начальных позициях (6, 32), (7, 32) и (8, 31) выигрышная стратегия есть у Пети. При начальной позиции (6, 32) он должен первым ходом получить позицию (6, 33), из начальных позиций (7, 32) и (8, 31). Петя после первого хода должен получить позицию (8, 32). Позиции (6, 33) и (8, 32) рассмотрены при разборе задания 1. В этих позициях выигрышная стратегия есть у игрока, который будет ходить вторым (теперь это Петя). Эта стратегия описана при разборе задания 1. Таким образом, Петя при любой игре Вани выигрывает своим вторым ходом.

Задание 3

В начальной позиции (7, 31) выигрышная стратегия есть у Вани. После первого хода Пети может возникнуть одна из четырёх позиций: (8, 31), (7, 32), (14, 31) и (7, 62). В позициях (14, 31) и

(7, 62) Ваня может выиграть одним ходом, удвоив количество камней во второй куче. Позиции (8, 31) и (7, 32) были рассмотрены при разборе задания 2. В этих позициях у игрока, который должен сделать ход (теперь это Ваня), есть выигрышная стратегия. Эта стратегия описана при разборе задания 2. Таким образом, в зависимости от игры Пети Ваня выигрывает на первом или втором ходу.



В решении, которое приведено выше, слишком много слов. Запись такого длинного текста в бланке утомит как вас, так и эксперта, проверяющего работу. Постарайтесь как можно больше рисовать.

27. Создать программу, оптимальную по памяти и по времени

Задача стоит 4 балла, что составляет более 11% от возможных 35 первичных баллов за всю работу.

Но научить решать эту задачу в рамках такого краткого пособия решительно невозможно. Это единственная задача ЕГЭ по информатике, которая требует реально высокого уровня подготовки.

Обязательно пишите вначале неэффективное полнопереборное решение: запоминайте все числа в массив и обрабатывайте их вложенными циклами. Сложность такого решения чуть выше сложности задачи 25, и у вас будет «в кармане» 2 первичных балла

Пример задания:

На спутнике «Фотон» установлен прибор, предназначенный для измерения энергии космических лучей. Каждую минуту прибор передаёт по каналу связи неотрицательное вещественное число — количество энергии, полученной за последнюю минуту, измеренное в условных единицах. Временем, в течение которого происходит передача, можно пренебречь. Необходимо найти в заданной серии показаний прибора минимальное произведение двух показаний, между моментами передачи которых прошло не менее 6 минут. Количество энергии, получаемое прибором за минуту, не превышает 1000 условных единиц. Общее количество показаний прибора в серии не превышает 10 000.

Вам предлагаются два задания, связанные с этой задачей: задание А и задание Б. Вы можете решать оба задания А и Б или одно из них по своему выбору.

Итоговая оценка выставляется как максимальная из оценок за задания А и Б. Если решение одного из заданий не представлено, то считается, что оценка за это задание составляет 0 баллов.

Задача А. Напишите программу для решения поставленной задачи, в которой входные данные будут запоминаться в массиве, после чего будут проверены все возможные пары элементов. Максимальная оценка за выполнение задания А — 2 балла.
Задача Б. Напишите программу для решения поставленной задачи, которая будет эффективна как по времени, так и по памяти (или хотя бы по одной из этих характеристик).

Входные данные представлены следующим образом. В первой строке задаётся число N – общее количество показаний прибора. Гарантируется, что N > 6. В каждой из следующих N строк задаётся одно положительное целое число – очередное показание прибора.

Пример входных данных:
11

12

45

5

4

25

23

21

20

10

12

26

Пример выходных данных для приведённого выше примера входных данных:

48

Программа считается эффективной по времени, если время работы программы пропорционально количеству элементов последовательности N, т.е. при увеличении N в k раз время работы программы должно увеличиваться не более чем в k раз. Обязательно укажите, что программа является решением задания Б.

Перед программой укажите версию языка и кратко опишите использованный алгоритм. В первой строке задаётся число N — общее количество показаний прибора. Гарантируется, что N > 6. В каждой из следующих N строк задаётся одно неотрицательное вещественное число — очередное показание прибора.

Решение:

Чтобы программа была эффективной по времени, нужно для каждого элемента входных данных, начиная с седьмого, вычислять его произведение на значение минимума от самого начала данных до элемента, находящегося на шесть элементов раньше текущего. Среди таких произведений нужно выбрать наименьшее.

```
N = int(input())
R_min = 1000**2 + 1
m = 1000 + 1
# очередь длины 6
Q = [0]*6
# первые шесть чисел просто уходят в очередь
for i in range(6):
   Q[i] = int(input())
for i in range(N-6):
   x = int(input())
   # подновление наименьшего среди чисел,
   # находящихся левее текущего не менее,
   # чем на 6 позиций
   if Q[0] < m:
      m = Q[0]
   for j in range(5):
      Q[j] = Q[j+1]
   # вставляем само число x в эту очередь в конец
   Q[5] = x
   # m - наилучшая пара к текущему x
   # ищем минимальное произведение
   if x*m < R_min:
      R_min = x*m

print(R_min)
```

Замечание

Поскольку задача бывает сложна для восприятия экспертом, полезно писать комментарии и словесное описание решения, хотя они формально и не оцениваются.

Будьте внимательны и к логическим, и к синтаксическим ошибкам: за каждую логическую — минус балл, за семь синтаксических — тоже.