# 1 (базовый уровень, время – 2 мин)

**Тема**: Кодирование и декодирование информации.

**Что нужно знать**:

* кодирование – это перевод информации с одного языка на другой (запись в другой системе символов, в другом алфавите)
* обычно кодированием называют перевод информации с «человеческого» языка на формальный, например, в двоичный код, а декодированием – обратный переход
* один символ исходного сообщения может заменяться одним символом нового кода или несколькими символами, а может быть и наоборот – несколько символов исходного сообщения заменяются одним символом в новом коде (китайские иероглифы обозначают целые слова и понятия)
* кодирование может быть *равномерное* и *неравномерное*;при равномерном кодировании все символы кодируются кодами равной длины;  
  при неравномерном кодировании разные символы могут кодироваться кодами разной длины, это затрудняет декодирование
* закодированное сообщение можно однозначно декодировать с начала, если выполняется *условие Фано*: никакое кодовое слово не является началом другого кодового слова;
* закодированное сообщение можно однозначно декодировать с конца, если выполняется *обратное условие Фано*: никакое кодовое слово не является окончанием другого кодового слова;
* условие Фано – это достаточное, но не необходимое условие однозначного декодирования.

### Пример задания

**Р-12.** *Для кодирования некоторой последовательности, состоящей из букв А, Б, В, Г, решили использовать неравномерный двоичный код, удовлетворяющий условию Фано. Для буквы А использовали кодовое слово 0, для буквы Б – кодовое слово 110. Какова наименьшая возможная суммарная длина всех четырёх кодовых слов?*

1) 7 2) 8 3) 9 4) 10

**Решение**:

1. условие Фано означает, что ни одно кодовое слово не совпадает с началом другого кодового слова
2. поскольку уже есть кодовое слово 0, ни одно другое кодовое слово не может начинаться с 0
3. поскольку есть код 110, запрещены кодовые слова 1, 11; кроме того, ни одно другое кодовое слово не может начинаться с 110
4. таким образом, нужно выбрать еще два кодовых слова, для которых выполняются эти ограничения
5. есть одно допустимое кодовое слово из двух символов: 10
6. если выбрать кодовое слово 10 для буквы В, то остаётся одно допустимое трёхсимвольное кодовое слово – 111, которое можно выбрать для буквы Г
7. таким образом, выбрав кодовые слова А – 0, Б – 110, В – 10, Г – 111, получаем суммарную длину кодовых слов 9 символов
8. если же не выбрать В – 10, то есть три допустимых трёхсимвольных кодовых слова: 100, 101 и 110; при выборе любых двух их них для букв В и Г получаем суммарную длину кодовых слов 10, что больше 9; поэтому выбираем вариант 3 (9 символов)
9. Ответ: 3.

### Ещё пример задания

**Р-11.** *По каналу связи передаются сообщения, содержащие только 5 букв А, И, К, О, Т. Для кодирования букв используется неравномерный двоичный код с такими кодовыми словами:*

*А — 0, И — 00, К — 10, О — 110, Т — 111.*

*Среди приведённых ниже слов укажите такое, код которого можно декодировать только одним способом. Если таких слов несколько, укажите первое по алфавиту.*

1) КАА 2) ИКОТА 3) КОТ 4) ни одно из со­об­ще­ний не под­хо­дит

**Решение**:

1. прежде всего заметим, что для заданного кода не выполняется ни прямое, ни обратное условие Фано; «виновата» в этом пара А – И: код буквы А совпадает как с началом, так и с окончанием кода буквы И; больше ни для одной пары кодовых слов прямое условие Фано не нарушено
2. это означает, что не все сообщения могут быть декодированы однозначно
3. теперь нужно понять, какие последовательности могут быть декодированы неоднозначно; в данном случае очевидно, что сообщения АА и И кодируются одинаково: 00, поэтому все слова, где есть АА или И, не могут быть декодированы однозначно
4. поэтому варианты 1 (КАА) и 2 (ИКОТА) отпадают
5. на всякий случай проверим вариант 3: КОТ = 10110111; первой буквой может быть только К (по-другому сочетание 10 получить нельзя), аналогично вторая буква – только О, а третья – только Т
6. Ответ: 3.

### Ещё пример задания

**Р-10.** *По каналу связи передаются сообщения, содержащие только 4 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**:

1. код однозначно декодируется, если выполняется условие Фано или обратное условие Фано; в данном случае «прямое» условие Фано выполняется: с кода буквы О (0) не начинается ни один из двух других кодов;
2. новый код не может начинаться с нуля (иначе нарушится условие Фано)
3. начнём проверку с кодов длиной 1; единственный код, не начинающийся с нуля – 1 – не подходит, потому что с 1 начинаются два других кода: Т (111) и П (100
4. кодов длиной 2, начинающихся с 1, всего 2: 10 и 11, но их использовать нельзя, потому что с 10 начинается код буквы П, а с 11 – код буквы Т
5. рассматриваем коды длиной 3, начинающиеся с 1; коды 100 и 111 уже заняты, а ещё два – 101 и 110 – свободны и их можно использовать, причём условие Фано выполняется в обоих случаях;
6. поскольку нужно выбрать код с минимальным значением, выбираем 101
7. Ответ: 101.

### Ещё пример задания

**Р-09.** *Для кодирования некоторой последовательности, состоящей из букв А, Б, В, Г и Д, используется неравномерный двоичный код, позволяющий однозначно декодировать полученную двоичную последовательность. Вот этот код: А – 0; Б – 100; В – 1010; Г – 111; Д – 110. Требуется сократить для одной из букв длину кодового слова так, чтобы код по-прежнему можно было декодировать однозначно. Коды остальных букв меняться не должны.*

*Каким из указанных способов это можно сделать?*

1) для буквы В – 101 2) это невозможно

3) для буквы В – 010 4) для буквы Б – 10

**Решение**:

1. код однозначно декодируется, если выполняется условие Фано или обратное условие Фано; в данном случае «прямое» условие Фано выполняется: с кода буквы А (0) не начинается ни один другой код, оставшиеся короткие коды (Б, Г и Д) не совпадают с началом длинного кода буквы В; таким образом, при сокращении нужно сохранить выполнение условия Фано
2. вариант 3 не подходит, потому что новый код буквы В начинается с 0 (кода А), поэтому условие Фано нарушено
3. вариант 4 не подходит, потому что код буквы В начинается с 10 (нового кода б), поэтому условие Фано нарушено
4. вариант 1 подходит, условие Фано сохраняется (все трёхбитные коды различны, ни один не начинается с 0)
5. Ответ: 1.

### Ещё пример задания

**Р-08.** *По каналу связи передаются сообщения, содержащие только 4 буквы: А, И, С, Т.*

*В любом сообщении больше всего букв А, следующая по частоте буква – С, затем – И. Буква Т встречается реже, чем любая другая. Для передачи сообщений нужно использовать неравномерный двоичный код, допускающий однозначное декодирование; при этом сообщения должны быть как можно короче. Шифровальщик может использовать один из перечисленных ниже кодов. Какой код ему следует выбрать?*

1) А – 0, И – 1, С – 00, Т – 11 2) С – 1, И – 0, А – 01, Т – 10

3) А – 1, И – 01, С – 001, Т – 000 4) С – 0, И – 11, А – 101, Т – 100

**Решение**:

1. сначала выберем коды, допускающие однозначное декодирование: это коды 3 и 4 (для них выполняется условие Фано), коды 1 и 2 не подходят
2. для того, чтобы длина сообщения была как можно короче, должно выполнять правило: «чем чаще встречается буква, тем короче её код»;
3. к сожалению, правило, приведённое выше, не совсем «хорошо» выполняется для кодов 3 и 4: в коде 3 длина кодового слова для буквы С больше, чем длина кодового слова буквы И (а хочется наоборот); для кода 4 длина кодового слова для буквы А – не самая маленькая из всех
4. сравним коды 3 и 4, предполагая, что в сообщении буква А встречается α раз, буква С – β раз, буква И – γ раз и буква Т – δ раз; причём по условию задачи α > β > γ > δ
5. при кодировании кодом 3 получаем сообщение длиной

L3  = α + 3β + 2γ +3 δ

1. при кодировании кодом 3 получаем сообщение длиной

L4  = 3α + β + 2γ +3 δ

1. находим разность: L4 ­– L3 = (3α + β + 2γ +3 δ) – (α + 3β + 2γ +3 δ) = 2α – 2β
2. поскольку α > β, получаем L4 ­– L3 > 0, то есть код 3 более экономичный
3. Ответ: 3.

### Ещё пример задания

**Р-07.** *По каналу связи передаются сообщения, содержащие только 4 буквы: Е, Н, О, Т. Для кодирования букв Е, Н, О используются 5-битовые кодовые слова: Е - 00000, Н - 00111, О - 11011. Для этого набора кодовых слов выполнено такое свойство: любые два слова из набора отличаются не менее чем в трёх позициях. Это свойство важно для расшифровки сообщений при наличии помех. Какое из перечисленных ниже кодовых слов можно использовать для буквы Т, чтобы указанное свойство выполнялось для всех четырёх кодовых слов?*

1) 11111 2) 11100 3) 00011 4) не подходит ни одно из указанных выше слов

**Решение**:

1. код, рассмотренный в условии задачи, относится к помехоустойчивым кодам, которые позволяют обнаружить и исправить определенное количество ошибок, вызванных помехами при передаче данных;
2. количество позиций, в которых отличаются два кодовых слова одинаковой длины, называется расстоянием Хэмминга
3. код, в котором расстояние Хэмминга между каждой парой кодовых слов равно *d*, позволяет обнаружить до *d-1* ошибок; для исправления *r* ошибок требуется выполнение условия

*d ≥ 2r + 1*

поэтому код с *d = 3* позволяет обнаружить одну или две ошибки, и исправить одну ошибку.

1. легко проверить, что для заданного кода (Е - 00000, Н - 00111, О - 11011) расстояние Хэмминга равно 3; в таблице выделены отличающиеся биты, их по три в парах Е-Н и Н-О и четыре в паре Е-О:

**Е – 00000 Е – 00000 Н – 00111**

**Н – 00111 О – 11011 О – 11011**

1. теперь проверяем расстояние между известными кодами и вариантами ответа; для первого ответа 11111 получаем минимальное расстояние 1 (в паре О-Т), этот вариант не подходит:

**Е – 00000 Н – 00111 О – 11011**

**Т - 11111 Т - 11111 Т - 11111**

1. для второго ответа 11100 получаем минимальное расстояние 3 (в парах Е-Т и О-Т):

**Е – 00000 Н – 00111 О – 11011**

**Т - 11100 Т - 11100 Т - 11100**

1. для третьего ответа 00011 получаем минимальное расстояние 1 (в паре Н-Т) , этот вариант не подходит:

**Е – 00000 Н – 00111 О – 11011**

**Т - 00011 Т - 00011 Т - 00011**

1. таким образом, расстояние Хэмминга, равное 3, сохраняется только для ответа 2
2. Ответ: 2.

### Ещё пример задания:

**Р-06.** *Для кодирования некоторой последовательности, состоящей из букв А, Б, В, Г и Д, используется неравномерный двоичный код, позволяющий однозначно декодировать полученную двоичную последовательность. Вот этот код: А–00, Б–010, В–011, Г–101, Д–111. Можно ли сократить для одной из букв длину кодового слова так, чтобы код по-прежнему можно было декодировать однозначно? Коды остальных букв меняться не должны. Выберите правильный вариант ответа.*

1) для буквы Б – 01 2) это невозможно

3) для буквы В – 01 4) для буквы Г – 01

**Решение (1 способ, проверка условий Фано)**:

1. для однозначного декодирования достаточно, чтобы выполнялось условие Фано или обратное условие Фано;
2. проверяем последовательно варианты 1, 3 и 4; если ни один из них не подойдет, придется выбрать вариант 2 («это невозможно»);
3. проверяем вариант 1: *А–00, Б–01, В–011, Г–101, Д–111.*

«прямое» условие Фано не выполняется (код буквы Б совпадает с началом кода буквы В);

«обратное» условие Фано не выполняется (код буквы Б совпадает с окончанием кода буквы Г); поэтому этот вариант не подходит;

1. проверяем вариант 3: *А–00, Б–010, В–01, Г–101, Д–111.*

«прямое» условие Фано не выполняется (код буквы В совпадает с началом кода буквы Б);

«обратное» условие Фано не выполняется (код буквы В совпадает с окончанием кода буквы Г); поэтому этот вариант не подходит;

1. проверяем вариант 4: *А–00, Б–010, В–011, Г–01, Д–111.*

«прямое» условие Фано не выполняется (код буквы Г совпадает с началом кодов букв Б и В); но **«обратное» условие Фано выполняется** (код буквы Г не совпадает с окончанием кодов остальных буквы); поэтому этот вариант подходит;

1. правильный ответ – 4.

**Решение (2 способ, дерево)**:

1. построим двоичное дерево, в котором от каждого узла отходит две ветки, соответствующие выбору следующей цифры кода – 0 или 1; разместим на этом дереве буквы А, Б, В, Г и Д так, чтобы их код получался как последовательность чисел на рёбрах, составляющих путь от корня до данной буквы (красным цветом выделен код буквы В – 011):

**0**

0

0

0

0

1

0

1

1

**1**

**1**

0

1

1

корень

1. здесь однозначность декодирования получается за счёт того, что при движении от корня к любой букве в середине пути не встречается других букв (выполняется условие Фано);
2. теперь проверим варианты ответа: предлагается перенести одну из букв, Б, В или Г, в узел с кодом 01, выделенный синим цветом
3. видим, что при переносе любой из этих букв нарушится условие Фано; например, при переносе буквы Б в синий узел она оказывается на пути от корня до В, и т.д.; это значит, что предлагаемые варианты не позволяют выполнить прямое условие Фано
4. хочется уже выбрать вариант 2 («это невозможно»), но у нас есть еще обратное условие Фано, для которого тоже можно построить аналогичное дерево, в котором движение от корня к букве дает её код **с конца** (красным цветом выделен код буквы В – 011, записанный с конца):

0

0

0

0

0

1

**0**

1

1

1

1

0

**1**

**1**

корень

видно, что обратное условие Фано также выполняется, потому что на пути от корня к любой букве нет других букв

1. в заданных вариантах ответа предлагается переместить букву Б, В или Г в синий узел; понятно, что Б или В туда перемещать нельзя – перемещённая буква отказывается на пути от корня к букве Г; а вот букву Г переместить можно, при этом обратное условие Фано сохранится
2. правильный ответ – 4.

### Ещё пример задания:

**Р-05.** *Для кодирования некоторой последовательности, состоящей из букв А, Б, В, Г и Д, решили использовать неравномерный двоичный код, позволяющий однозначно декодировать двоичную последовательность, появляющуюся на приёмной стороне канала связи. Использовали код:   
А–1, Б–000, В–001, Г–011. Укажите, каким кодовым словом должна быть закодирована буква Д. Длина этого кодового слова должна быть наименьшей из всех возможных. Код должен удовлетворять свойству однозначного декодирования.*

1) 00 2) 01 3)11 4) 010

**Решение**:

1. заметим, что для известной части кода выполняется условие Фано – никакое кодовое слово не является началом другого кодового слова
2. если Д = 00, такая кодовая цепочка совпадает с началом Б = 000 и В = 001, невозможно однозначно раскодировать цепочку 000000: это может быть ДДД или ББ; поэтому первый вариант не подходит
3. если Д = 01, такая кодовая цепочка совпадает с началом Г = 011, невозможно однозначно раскодировать цепочку 011: это может быть ДА или Г; поэтому второй вариант тоже не подходит
4. если Д = 11, условие Фано тоже нарушено: кодовое слово А = 1 совпадает с началом кода буквы Д, невозможно однозначно раскодировать цепочку 111: это может быть ДА или ААА; третий вариант не подходит
5. для четвертого варианта, Д = 010, условие Фано не нарушено;
6. правильный ответ – 4.

|  |
| --- |
| **Возможные ловушки**:   * условие Фано – это **достаточное**, но не необходимое условие однозначного декодирования, поэтому для уверенности полезно найти для всех «неправильных» вариантов контрпримеры: цепочки, для которых однозначное декодирование невозможно |

### Еще пример задания:

**Р-04.** *Для кодирования букв А, Б, В, Г решили использовать двухразрядные последовательные двоичные числа (от 00 до 11, соответственно). Если таким способом закодировать последовательность символов БАВГ и записать результат шестнадцатеричным кодом, то получится*

1) 4B16 2) 41116 3)BACD16 4) 102316

**Решение**:

1. из условия коды букв такие: A – 00, Б –01, В – 10 и Г – 11, код равномерный
2. последовательность БАВГ кодируется так: 01 00 10 11 = 1001011
3. разобьем такую запись на тетрады справа налево и каждую тетраду переведем в шестнадцатеричную систему (то есть, сначала в десятичную, а потом заменим все числа от 10 до 15 на буквы A, B, C, D, E, F); получаем

1001011 = 0100 10112 = 4B16

1. правильный ответ – 1.

|  |
| --- |
| **Возможные ловушки**:   * расчет на то, что при переводе тетрад в шестнадцатеричную систему можно забыть заменить большие числа (10–15) на буквы (10112 = 11, получаем неверный ответ 41116) * может быть дан неверный ответ, в котором нужные цифры поменяли местами (расчет на невнимательность), например, B416 * в ответах дана последовательность, напоминающая исходную (неверный ответ BACD16), чтобы сбить случайное угадывание |

### Еще пример задания:

**Р-03.** *Для 5 букв латинского алфавита заданы их двоичные коды (для некоторых букв – из двух бит, для некоторых – из трех). Эти коды представлены в таблице:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| 000 | 01 | 100 | 10 | 011 |

Определить, какой набор букв закодирован двоичной строкой 0110100011000

1) EBCEA 2) BDDEA 3) BDCEA 4) EBAEA

**Решение (вариант 1, декодирование с начала)**:

1. здесь используется неравномерное кодирование, при котором декодирование может быть неоднозначным, то есть, заданному коду может соответствовать несколько разных исходных сообщений
2. попробуем декодировать с начала цепочки, первой буквой может быть B или E, эти случаи нужно рассматривать отдельно
3. пусть первая буква – E с кодом 011, тогда остается цепочка 0100011000

* для кода 0100011000 первой буквой может быть только B с кодом 01, тогда остается 00011000 ( начало исходной цепочки – EB?)
* для кода 00011000 первой буквой может быть только A с кодом 000, тогда остается 11000, а эта цепочка не может быть разложена на заданные коды букв
* поэтому наше предположение о том, что первая буква – E, неверно

1. пусть первая буква – B с кодом 01, тогда остается цепочка 10100011000

* для кода 10100011000 первой буквой может быть только D с кодом 10, тогда остается 100011000 (можно полагать, что начало исходной цепочки – BD?)
* для кода 100011000 первой буквой может быть только С с кодом 100, тогда остается 011000 (начало исходной цепочки – BDC?)

Несмотря на то, что среди ответов есть единственная цепочка, которая начинается с BDC, здесь нельзя останавливаться, потому что «хвост» цепочки может «не сойтись»

* для кода 011000 на первом месте может быть B (код 01) или E (011); в первом случае «хвост» 1000 нельзя разбить на заданные коды букв, а во втором – остается код 000 (буква А), поэтому исходная цепочка может быть декодирована как BDCEA

1. правильный ответ – 3

|  |
| --- |
| **Возможные ловушки и проблемы**:   * при декодировании неравномерных кодов может быть очень много вариантов, их нужно рассмотреть все; это требует серьезных усилий и можно легко запутаться * нельзя останавливаться, не закончив декодирование до конца и не убедившись, что все «сходится», на это обычно и рассчитаны неверные ответы |

**Решение (вариант 2, декодирование с конца)**:

1. для кода 0110100011000 последней буквой может быть только А (код 000), тогда остается цепочка 0110100011
2. для 0110100011 последней может быть только буква E (011), тогда остается цепочка 0110100
3. для 0110100 последней может быть только буква C (100), тогда остается цепочка 0110
4. для 0110 последней может быть только буква D (10), тогда остается 01 – это код буквы B
5. таким образом, получилась цепочка BDCEA
6. правильный ответ – 3

|  |
| --- |
| **Возможные ловушки и проблемы**:   * при декодировании неравномерных кодов может быть очень много вариантов (здесь *случайно* получилась единственно возможная цепочка), их нужно рассмотреть все; это требует серьезных усилий и можно легко запутаться * нельзя останавливаться, не закончив декодирование до конца и не убедившись, что все «сходится», на это обычно и рассчитаны неверные ответы |

**Решение (вариант 3, кодирование ответов)**:

1. в данном случае самое простое и надежное – просто закодировать все ответы, используя приведенную таблицу кодов, а затем сравнить результаты с заданной цепочкой
2. получим

1) EBCEA – 01101100011000 2) BDDEA – 011010011000

3) BDCEA – 0110100011000 4) EBAEA – 01101000011000

1. сравнивая эти цепочки с заданной, находим, что правильный ответ – 3.

|  |
| --- |
| **Возможные проблемы**:   * сложно сравнивать длинные двоичные последовательности, поскольку они однородны, содержат много одинаковых нулей и единиц |

### Еще пример задания:

**Р-02.** *Для передачи по каналу связи сообщения, состоящего только из букв А, Б, В, Г, решили использовать неравномерный по длине код: A=0, Б=10, В=110. Как нужно закодировать букву Г, чтобы длина кода была минимальной и допускалось однозначное разбиение кодированного сообщения на буквы?*

1) 1 2) 1110 3) 111 4) 11

**Решение (вариант 1, метод подбора)**:

1. рассмотрим все варианты в порядке увеличения длины кода буквы Г
2. начнем с Г=1; при этом получается, что сообщение «10» может быть раскодировано двояко: как ГА или Б, поэтому этот вариант не подходит
3. следующий по длине вариант – Г=11; в этом случае сообщение «110» может быть раскодировано как ГА или В, поэтому этот вариант тоже не подходит
4. третий вариант, Г=111, дает однозначное раскодирование во всех сочетаниях букв, поэтому…
5. … правильный ответ – 3.

|  |
| --- |
| **Возможные проблемы**:   * при переборе можно ошибиться и «просмотреть» какой-нибудь вариант |

**Решение (вариант 2, «умный» метод)**:

1. для того, чтобы сообщение, записанное с помощью неравномерного по длине кода, однозначно раскодировалось, требуется, чтобы никакой код не был началом другого (более длинного) кода; это условие называют *условием Фано*
2. как и в первом решении, рассматриваем варианты, начиная с самого короткого кода для буквы Г; в нашем случае код Г=1 является началом кодов букв Б и В, поэтому условие Фано не выполняется, такой код не подходит
3. код Г=11 также является началом другого кода (кода буквы В), поэтому это тоже ошибочный вариант
4. третий вариант кода, Г=111, не является началом никакого уже известного кода; кроме того, ни один уже имеющийся код не является началом кода 111; таким образом, условие Фано выполняется
5. поэтому правильный ответ – 3.

|  |
| --- |
| **Возможные проблемы**:   * нужно знать условие Фано |

### Еще пример задания[[1]](#footnote-1):

**Р-01.** *Черно-белое растровое изображение кодируется построчно, начиная с левого верхнего угла и заканчивая в правом нижнем углу. При кодировании 1 обозначает черный цвет, а 0 – белый.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

*Для компактности результат записали в шестнадцатеричной системе счисления. Выберите правильную запись кода.*

1) BD9AA5 2) BDA9B5 3) BDA9D5 4) DB9DAB

**Решение**:

1. «вытянем» растровое изображение в цепочку: сначала первая (верхняя) строка, потом – вторая, и т.д.:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| *1 строка* | | | | | | *2 строка* | | | | | | *3 строка* | | | | | | *4 строка* | | | | | |

1. в этой полоске 24 ячейки, черные заполним единицами, а белые – нулями:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *1* | *0* | *1* | *1* | *1* | *1* | *0* | ***1*** | ***1*** | *0* | ***1*** | *0* | ***1*** | *0* | *0* | ***1*** | ***1*** | ***1*** | *0* | ***1*** | *0* | ***1*** | *0* | ***1*** |
| *1 строка* | | | | | | *2 строка* | | | | | | *3 строка* | | | | | | *4 строка* | | | | | |

1. поскольку каждая цифра в шестнадцатеричной системе раскладывается ровно в 4 двоичных цифры, разобьем полоску на **тетрады** – группы из четырех ячеек (в данном случае все равно, откуда начинать разбивку, поскольку в полоске целое число тетрад – 6):

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *1* | *0* | *1* | *1* | *1* | *1* | *0* | ***1*** | ***1*** | *0* | ***1*** | *0* | ***1*** | *0* | *0* | ***1*** | ***1*** | ***1*** | *0* | ***1*** | *0* | ***1*** | *0* | ***1*** |

1. переводя тетрады в шестнадцатеричную систему, получаем последовательно цифры B (11), D(13), A(10), 9, D(13) и 5, то есть, цепочку BDA9D5
2. поэтому правильный ответ – 3.

|  |
| --- |
| **Возможные проблемы**:   * нужно уметь быстро переводить тетрады в шестнадцатеричные цифры (в крайнем случае, это можно сделать через десятичную систему) |

### Еще пример задания:

**Р-00.** *Для передачи чисел по каналу с помехами используется код проверки четности. Каждая его цифра записывается в двоичном представлении, с добавлением ведущих нулей до длины 4, и к получившейся последовательности дописывается сумма её элементов по модулю* **2** *(например, если передаём* **23***, то получим последовательность* **0010100110***). Определите, какое число передавалось по каналу в виде* **01010100100111100011***?*

1) 59143 2) 5971 3) 102153 4) 10273

**Решение**:

1. сначала разберемся, как закодированы числа в примере; очевидно, что используется код равномерной длины; поскольку 2 знака кодируются 10 двоичными разрядами (битами), на каждую цифру отводится 5 бит, то есть

2 → 00101 и 3 → 00110

1. как следует из условия, четыре первых бита в каждой последовательности – это двоичный код цифры, а пятый бит (бит четности) используется для проверки и рассчитывается как «сумма по модулю два», то есть остаток от деления суммы битов на 2; тогда

2 = 00102, бит четности (0 + 0 + 1 + 0) mod 2 = 1

3 = 00112, бит четности (0 + 0 + 1 + 1) mod 2 = 0

1. но бит четности нам совсем **не нужен**, важно другое: пятый бит в каждой пятерке **можно отбросить**!
2. разобъем заданную последовательность на группы по 5 бит в каждой:

01010, 10010, 01111, 00011.

1. отбросим пятый (последний) бит в каждой группе:

0101, 1001, 0111, 0001.  
это и есть двоичные коды передаваемых чисел:

01012 = 5, 10012 = 9, 01112 = 7, 00012 = 1.

1. таким образом, были переданы числа 5, 9, 7, 1 или число 5971.
2. Ответ: 2.

# 2 (базовый уровень, время – 3 мин)

**Тема**: Построение и анализ таблиц истинности логических выражений.

**Про обозначения**

К сожалению, обозначения логических операций И, ИЛИ и НЕ, принятые в «серьезной» математической логике (**∧**,**∨**,**¬**), неудобны, интуитивно непонятны и никак не проявляют аналогии с обычной алгеброй. Автор, к своему стыду, до сих пор иногда путает **∧** и **∨**. Поэтому на его уроках операция «НЕ» обозначается чертой сверху, «И» – знаком умножения (поскольку это все же логическое умножение), а «ИЛИ» – знаком «+» (логическое сложение).   
В разных учебниках используют разные обозначения. К счастью, в начале задания ЕГЭ приводится расшифровка закорючек (**∧**,**∨**,**¬**), что еще раз подчеркивает проблему.

**Что нужно знать**:

* условные обозначения логических операций

**¬ A, **  не A (отрицание, инверсия)

**A ∧ B, ** A и B (логическое умножение, конъюнкция)

**A ∨ B, **  A или B (логическое сложение, дизъюнкция)

**A** → **B**  импликация (следование)

**A** ≡ **B**  эквивалентность (равносильность)

* операцию «импликация» можно выразить через «ИЛИ» и «НЕ»:

**A** → **B = ¬ A ∨ B** или в других обозначениях **A** → **B = **

* иногда для упрощения выражений полезны формулы де Моргана:

**¬ (A ∧ B) = ¬ A ∨ ¬ B **

**¬ (A ∨ B) = ¬ A ∧ ¬ B **

* если в выражении нет скобок, сначала выполняются все операции «НЕ», затем – «И», затем – «ИЛИ», «импликация», и самая последняя – «эквивалентность»
* таблица истинности выражения определяет его значения при всех возможных комбинациях исходных данных
* если известна только часть таблицы истинности, соответствующее логическое выражение однозначно определить нельзя, поскольку частичной таблице могут соответствовать несколько *разных* логических выражений (не совпадающих для других вариантов входных данных);
* количество *разных* логических выражений, удовлетворяющих неполной таблице истинности, равно , где  – число *отсутствующих* строк; например, полная таблица истинности выражения с тремя переменными содержит 23=8 строчек, если заданы только 6 из них, то можно найти 28-6=22=4 *разных* логических выражения, удовлетворяющие этим 6 строчкам (но отличающиеся в двух оставшихся)
* логическая сумма A + B + C + … равна 0 (выражение ложно) тогда и только тогда, когда все слагаемые одновременно равны нулю, а в остальных случаях равна 1 (выражение истинно)
* логическое произведение A · B · C · … равно 1 (выражение истинно) тогда и только тогда, когда все сомножители одновременно равны единице, а в остальных случаях равно 0 (выражение ложно)
* логическое следование (импликация) А→В равна 0 тогда и только тогда, когда из A (посылка) истинна, а B (следствие) ложно
* эквивалентность А≡B равна 1 тогда и только тогда, когда оба значения одновременно равны 0 или одновременно равны 1

### Пример задания:

**Р-13**. Каждое логическое выражение A и B зависит от одного и того же набора из 5 переменных. В таблицах истинности каждого из этих выражений в столбце значений стоит ровно по 4 единицы. Каково минимально возможное число единиц в столбце значений таблицы истинности выражения A ∨ ¬B?

**Решение:**

1. полная таблица истинности каждого выражения с пятью переменными содержит 25 = 32 строки
2. в каждой таблице по 4 единицы и по 28 (= 32 – 4) нуля
3. выражение A ∨ ¬B равно нулю тогда и только тогда, когда A = 0 или B = 1
4. минимальное количество единиц в таблице истинности выражения A ∨ ¬B будет тогда, когда там будет наибольшее число нулей, то есть в наибольшем количество строк одновременно A = 0 и B = 1
5. по условию A = 0 в 28 строках, и B = 1 в 4 строках, поэтому выражение A ∨ ¬B может быть равно нулю не более чем в 4 строках, оставшиеся 32 – 4 = 28 могут быть равны 1
6. Ответ: 28.

### Ещё пример задания:

**Р-12**. Дан фрагмент таблицы истинности для выражения F:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *x1* | *x2* | *x3* | *x4* | *x5* | *F* |
| **0** | **0** | **1** | **0** | **0** | **0** |
| **1** | **0** | **1** | **0** | **1** | **1** |
| **0** | **1** | **1** | **1** | **0** | **1** |

Укажите максимально возможное число различных строк полной таблицы истинности этого выражения, в которых значение x1 не совпадает с F.

**Решение:**

1. полная таблица истинности выражения с пятью переменными содержит 25 = 32 строки
2. в приведённой части таблицы в двух строках значение *x1* совпадает с *F*, а в одной – не совпадает
3. во всех оставшихся (неизвестных) 32 – 3 = 29 строках значения *x1* и *F* могут не совпадать
4. всего несовпадающих строк может быть 1 + 29 = 30.
5. Ответ: 30.

### Ещё пример задания:

**Р-11**. Александра заполняла таблицу истинности для выражения F. Она успела заполнить лишь небольшой фрагмент таблицы:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *x1* | *x2* | *x3* | *x4* | *x5* | *x6* | *x7* | *x8* | *F* |
|  | **0** |  |  |  |  |  | **1** | **0** |
| **1** |  |  | **0** |  |  |  |  | **1** |
|  |  |  | **1** |  |  |  | **1** | **1** |

Каким выражением может быть F?

1) **x1 ∧ ¬x2 ∧ x3 ∧ ¬x4 ∧ x5 ∧ x6 ∧ ¬x7 ∧ ¬x8**

2) **x1 ∨ x2 ∨ x3 ∨ ¬x4 ∨ ¬x5 ∨ ¬x6 ∨ ¬x7 ∨ ¬x8**

3) **¬x1 ∧ x2 ∧ ¬x3 ∧ x4 ∧ x5 ∧ ¬x6 ∧ ¬x7 ∧ ¬x8**

4) **x1 ∨ ¬x2 ∨ x3 ∨ ¬x4 ∨ ¬x5 ∨ ¬x6 ∨ ¬x7 ∨ ¬x8**

**Решение:**

1. перепишем выражения в более простой форме, заменив «И» (**∧**) на умножение и «ИЛИ» (**∨**) на сложение:

1)**

2) **

3) **

4) **

1. в последнем столбце таблицы истинности видим две единицы, откуда сразу следует, что это не может быть цепочка операций «И» (конъюнкций), которая даёт только одну единицу; поэтому ответы 1 и 3 заведомо неверные
2. анализируем первую строку таблицы истинности; мы знаем в ней только два значения - и 
3. для того, чтобы в результате в первой строке получить 0, необходимо, чтобы переменная  входила в сумму с инверсией (тогда из 1 получится 0!), это условие выполняется для обоих оставшихся вариантов, 2 и 4
4. кроме того, переменная  должна входить в выражение без инверсии (иначе соответствующее слагаемое в первой строке равно 1, и это даст в результате 1); этому условию не удовлетворяет выражение 4; остается один возможный вариант – выражение 2
5. Ответ: 2.

### Ещё пример задания:

**Р-10**. Александра заполняла таблицу истинности для выражения F. Она успела заполнить лишь небольшой фрагмент таблицы:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *x1* | *x2* | *x3* | *x4* | *x5* | *x6* | *x7* | *x8* | *F* |
|  | **0** |  |  |  |  |  | **1** | **1** |
| **1** |  |  | **0** |  |  |  |  | **0** |
|  |  |  | **1** |  |  |  | **1** | **0** |

Каким выражением может быть F?

1) **x1 ∧ ¬x2 ∧ x3 ∧ ¬x4 ∧ x5 ∧ x6 ∧ ¬x7 ∧ ¬x8**

2) **x1 ∨ x2 ∨ x3 ∨ ¬x4 ∨ ¬x5 ∨ ¬x6 ∨ ¬x7 ∨ ¬x8**

3) **x1 ∧ ¬x2 ∧ ¬x3 ∧ x4 ∧ x5 ∧ ¬x6 ∧ ¬x7 ∧ x8**

4) **x1 ∨ ¬x2 ∨ x3 ∨ ¬x4 ∨ ¬x5 ∨ ¬x6 ∨ ¬x7 ∨ ¬x8**

1. перепишем выражения в более простой форме, заменив «И» (**∧**) на умножение и «ИЛИ» (**∨**) на сложение:

1)**

2) **

3) **

4) **

1. в последнем столбце в таблице видим одну единицу и два нуля, поэтому это не может быть дизъюнкция, которая даёт ноль только при одном наборе значений переменных; таким образом, варианты 2 и 4 заведомо неверные, нужно сделать выбор между ответами 1 и 3
2. рассматриваем «особую» строчку таблице, в которой функция равна 1;
3. поскольку мы говорим о конъюнкции, переменная  должна входить в неё с инверсией (это выполняется для обоих оставшихся вариантов), а переменная – без инверсии; последнее из этих двух условий верно только для варианта 3, это и есть правильный ответ.
4. Ответ: 3.

### Ещё пример задания:

**Р-09**. Александра заполняла таблицу истинности для выражения F. Она успела заполнить лишь небольшой фрагмент таблицы:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *x1* | *x2* | *x3* | *x4* | *x5* | *x6* | *x7* | *x8* | *F* |
|  | **0** |  |  |  |  |  | **1** | **1** |
| **1** |  |  | **0** |  |  |  |  | **0** |
|  |  |  | **1** |  |  |  | **1** | **0** |

Каким выражением может быть F?

1) **¬x1 ∧ x2 ∨ x2 ∧ ¬x3 ∧ ¬x4 ∨ x2 ∧ ¬x5 ∨ x5 ∧ x6 ∧ ¬x7 ∧ ¬x8**

2) **(x1 ∧ ¬x2 ∨ ¬x3 ∨ x4) ∧ (x5 ∨ x6 ∨ ¬x7 ∨ x8)**

3) **x1 ∧ ¬x8 ∨ ¬x3 ∧ x4 ∧ x5 ∨ ¬x6 ∧ ¬x7 ∧ x8**

4) **x1 ∧ ¬x4 ∨ x2 ∧ x3 ∧ ¬x4 ∨ ¬x5 ∨ ¬x6 ∨ ¬x7 ∨ ¬x8**

**Решение:**

1. перепишем выражения в более простой форме, заменив «И» (**∧**) на умножение и «ИЛИ» (**∨**) на сложение:

1) ****

2) ****

3) ****

4) ****

1. cреди заданных вариантов ответа нет «чистых» конъюнкций и дизъюнкций, поэтому мы должны проверить возможные значения всех выражений для каждой строки таблицы
2. подставим в эти выражения известные значения переменных из первой строчке таблицы, и :

1) ****

2) ****

3) ****

4) ****

1. видим, что первое выражение при и  всегда равно нулю, поэтому вариант 1 не подходит; остальные выражения вычислимы, то есть, могут быть равны как 0, так и 1
2. подставляем в оставшиеся три выражения известные данные из второй строчки таблицы, и :

2) ****

3) ****

4) ****

1. видим, что выражение 4 при этих данных всегда равно 1, поэтому получить F=0, как задано в таблице, невозможно; этот вариант не подходит
2. остаются выражения 2 и 3; подставляем в них известные данные из третьей строчки таблицы, и :

2) ****

3) ****

1. Выражение 2 в этом случае всегда равно 1, поэтому оно не подходит (по таблице истинности оно должно быть равно 0); выражение 3 вычислимо, это и есть правильный ответ
2. Ответ: 3.

### Ещё пример задания:

**Р-08**. Дан фрагмент таблицы истинности выражения F.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *x1* | *x2* | *x3* | *x4* | *x5* | *x6* | *x7* | *x8* | *F* |
| **1** | **0** | **1** | **0** | **1** | **1** | **1** | **0** | **0** |
| **0** | **1** | **0** | **1** | **1** | **0** | **0** | **1** | **0** |
| **0** | **1** | **1** | **0** | **1** | **0** | **1** | **0** | **1** |

Какое выражение соответствует F?

1) **(x2 → x1) ∧ ¬x3 ∧ x4 ∧ ¬x5 ∧ x6 ∧ ¬x7 ∧ x8**

2) **(x2 → x1) ∨ ¬x3 ∨ x4 ∨ ¬x5 ∨ x6 ∨ ¬x7 ∨ x8**

3) **¬(x2 → x1) ∧ x3 ∧ ¬x4 ∧ x5 ∧ ¬x6 ∧ x7 ∧ ¬x8**

4) **(x2 → x1) ∨ x3 ∨ ¬x4 ∨ x5 ∨ ¬x6 ∨ x7 ∨ ¬x8**

**Решение:**

1. перепишем выражение в более простой форме, заменив «И» (**∧**) на умножение и «ИЛИ» (**∨**) на сложение:



1. в этом задании среди значений функции только одна единица, как у операции «И», это намекает на то, что нужно искать правильный ответ среди вариантов, содержащих «И», «НЕ» и импликацию (это варианты 1 и 3)
2. действительно, вариант 2 исключён, потому что при *x­4=*1 во второй строке получаем 1, а не 0
3. аналогично, вариант 4 исключён, потому что при *x­5=*1 в первой строке получаем 1, а не 0
4. итак, остаются варианты 1 и 3; вариант 1 не подходит, потому что при *x­6=*0 в третьей строке получаем 0, а не 1
5. проверяем подробно вариант 3, он подходит во всех строчках
6. Ответ: 3.

### Ещё пример задания:

**Р-07**. Дан фрагмент таблицы истинности выражения F.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *x1* | *x2* | *x3* | *x4* | *x5* | *x6* | *F* |
| **0** | **1** | **0** | **0** | **1** | **1** | **0** |
| **0** | **0** | **1** | **0** | **0** | **1** | **0** |
| **0** | **1** | **0** | **1** | **0** | **1** | **0** |

Какое выражение соответствует F?

1) **(x1 ∧ x2) ∨ (x3 ∧ x4) ∨ (x5 ∧ x6)**

2) **(x1 ∧ x3) ∨ (x3 ∧ x5) ∨ (x5 ∧ x1)**

3) **(x2 ∧ x4) ∨ (x4 ∧ x6) ∨ (x6 ∧ x2)**

4) **(x1 ∧ x4) ∨ (x2 ∧ x5) ∨ (x3 ∧ x6)**

**Решение:**

1. во-первых, обратим внимание, что в столбце F – все нули, то есть, при всех рассмотренных наборах x1, …, x6 функция ложна
2. перепишем предложенные варианты в более простых обозначениях:

*x1⋅x2 + x3⋅x4 + x5⋅x6*

*x1⋅x3 + x3⋅x5 + x5⋅x1*

*x2⋅x4 + x4⋅x5 + x6⋅x2*

*x1⋅x4 + x2⋅x5 + x3⋅x6*

1. это суммы произведений, поэтому для того, чтобы функция была равна 0, необходимо, чтобы все произведения были равны 0
2. по таблице смотрим, какие произведения равны 1:

1-я строка: *x2⋅x5*, *x2⋅x6* и *x5⋅x6*

2-я строка: *x3⋅x6*

3-я строка: *x2⋅x4*, *x2⋅x6* и *x4⋅x6*

1. таким образом, нужно выбрать функцию, где эти произведения не встречаются; отметим их:

*x1⋅x2 + x3⋅x4 + x5⋅x6*

*x1⋅x3 + x3⋅x5 + x5⋅x1*

*x2⋅x4 + x4⋅x5 + x6⋅x2*

*x1⋅x4 + x2⋅x5 + x3⋅x6*

1. единственная функция, где нет ни одного «запрещённого» произведения – это функция 2
2. Ответ: 2.

### Ещё пример задания:

**Р-06**. (<http://ege.yandex.ru>) *Дан фрагмент таблицы истинности выражения F.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *x1* | *x2* | *x3* | *x4* | *x5* | *F* |
| **1** | **1** | **1** | **0** | **0** | **1** |
| **1** | **1** | **0** | **1** | **1** | **0** |
| **0** | **0** | **1** | **1** | **1** | **1** |

Одно из приведенных ниже выражений истинно при любых значениях переменных *x1, x2,x3, x4, x5*. Укажите это выражение.

1) **F(x1,x2,x3,x4,x5)→x1**

2) **F(x1,x2,x3,x4,x5)→x2**

3) **F(x1,x2,x3,x4,x5)→x3**

4) **F(x1,x2,x3,x4,x5)→x4**

**Решение:**

1. во всех заданных вариантах ответа записана импликация, она ложна только тогда, когда левая часть (значение функции F) истинна, а правая – ложна.
2. выражение 1 ложно для набора переменных в третьей строке таблицы истинности, где *F*(*…*) *=*1 и , оно не подходит
3. выражение 2 ложно для набора переменных в третьей строке таблицы истинности, где *F*(*…*) *=*1 и , оно не подходит
4. выражение 3 истинно для всех наборов переменных, заданных в таблице истинности
5. выражение 4 ложно для набора переменных в первой строке таблицы истинности, где *F*(*…*) *=*1 и , оно не подходит
6. ответ: 3.

### Ещё пример задания:

**Р-05**. *Дано логическое выражение, зависящее от 5 логических переменных:*

**z1 ∧ ¬z2 ∧ ¬z3 ∧ ¬z4 ∧ z5**

*Сколько существует различных наборов значений переменных, при которых выражение ложно?*

1) **1** 2) **2** 3) **31** 4) **32**

**Решение:**

1. задано выражение с пятью переменными, которые могут принимать 25 = 32 различных комбинаций значений
2. операция **∧** – это логическое умножение, поэтому заданное выражение истинно только тогда, когда все сомножитель истинны, то есть в одном единственном случае
3. тогда остается 32 – 1 = 31 вариант, когда выражение ложно
4. ответ: 3.

### Ещё пример задания:

**Р-04**. *Дан фрагмент таблицы истинности выражения F. Какое выражение соответствует F?*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x1* | *x2* | *x3* | *x4* | *x5* | *x6* | *x7* | *F* |
| **0** | **1** | **0** | **1** | **1** | **1** | **0** | **0** |
| **1** | **1** | **0** | **1** | **0** | **1** | **0** | **1** |
| **0** | **1** | **0** | **1** | **1** | **0** | **1** | **0** |

1) (**x1 ∨ x2) ∧ ¬x3 ∧ x4 ∧ ¬x5 ∧ x6 ∧ ¬x7**

2) (**x1 ∧ x2) ∨ ¬x3 ∨ x4 ∨ ¬x5 ∨ x6 ∨ x7**

3) (**x1 ∧ ¬x2) ∧ x3 ∧ ¬x4 ∧ ¬x5 ∧ x6 ∧ ¬x7**

4) (**¬x1 ∧ ¬x2) ∧ x3 ∧ ¬x4 ∧ x5 ∧ ¬x6 ∧ x7**

**Решение:**

1. в последнем столбце таблицы всего одна единица, поэтому стоит попробовать использовать функцию, состоящую из цепочки операций «И» (ответы 1, 3 или 4);
2. для этой «единичной» строчки получаем, что инверсия (операция «НЕ») должна быть применена к переменным x3, x5  и x7, которые равны нулю:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x1* | *x2* | *x3* | *x4* | *x5* | *x6* | *x7* | *F* |
| **1** | **1** | **0** | **1** | **0** | **1** | **0** | **1** |

таким образом, остается только вариант ответа 1 (в ответах 3 и 4 переменная x3 указана без инверсии)

1. проверяем скобку (**x1 ∨ x2)**: в данном случае она равна 1, что соответствует условию
2. ответ: 1.

### Ещё пример задания:

|  |  |  |  |
| --- | --- | --- | --- |
| *X* | *Y* | *Z* | *F* |
| 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

**Р-03**. *Символом F обозначено одно из указанных ниже логических выражений от трех аргументов: X, Y, Z. Дан фрагмент таблицы истинности выражения F. Какое выражение соответствует F?*

1) **¬X ∧ ¬Y ∧ ¬Z** 2) **X ∧ Y ∧ Z** 3) **X ∨ Y ∨ Z** 4) **¬X ∨ ¬Y ∨ ¬Z**

**Решение (основной вариант):**

1. нужно для каждой строчки подставить заданные значения X, Y и Z во все функции, заданные в ответах, и сравнить результаты с соответствующими значениями F для этих данных
2. если для какой-нибудь комбинации X, Y и Z результат не совпадает с соответствующим значением F, оставшиеся строчки можно не рассматривать, поскольку для правильного ответа все три результата должны совпасть со значениями функции F
3. перепишем ответы в других обозначениях:  
    1)  2)  3)  4) 
4. первое выражение, , равно 1 только при , поэтому это неверный ответ (первая строка таблицы не подходит)
5. второе выражение, , равно 1 только при , поэтому это неверный ответ (первая и вторая строки таблицы не подходят)
6. третье выражение,, равно нулю при , поэтому это неверный ответ (вторая строка таблицы не подходит)
7. наконец, четвертое выражение,  равно нулю только тогда, когда , а в остальных случаях равно 1, что совпадает с приведенной частью таблицы истинности
8. таким образом, правильный ответ – 4 ; частичная таблица истинности для всех выражений имеет следующий вид:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *X* | *Y* | *Z* | *F* |  |  |  |  |
| 1 | 0 | 0 | 1 | 0 **×** | 0 **×** | 1 | 1 |
| 0 | 0 | 0 | 1 | – | – | 0 **×** | 1 |
| 1 | 1 | 1 | 0 | – | – | – | 0 |

(красный крестик показывает, что значение функции не совпадает с F, а знак «–» означает, что вычислять оставшиеся значения не обязательно).

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + серьезные сложности представляет применяемая в заданиях ЕГЭ форма записи логических выражений с «закорючками», поэтому рекомендуется сначала *внимательно* перевести их в «удобоваримый» вид;   + расчет на то, что ученик перепутает значки **∧** и **∨** (неверный ответ 1)   + в некоторых случаях заданные выражения-ответы лучше сначала упростить, особенно если они содержат импликацию или инверсию сложных выражений (как упрощать – см. разбор задачи А10) |

**Решение (вариант 2):**

1. часто правильный ответ – это самая простая функция, удовлетворяющая частичной таблице истинности, то есть, имеющая единственный нуль или единственную единицу в полной таблице истинности
2. в этом случае можно найти такую функцию и проверить, есть ли она среди данных ответов
3. в приведенной задаче в столбце F есть единственный нуль для комбинации 
4. выражение, которое имеет единственный нуль для этой комбинации, это , оно есть среди приведенных ответов (ответ 4)
5. таким образом, правильный ответ – 4

|  |
| --- |
| **Возможные проблемы**:   * + метод применим не всегда, то есть, найденная в п. 4 функция может отсутствовать среди ответов |

### Еще пример задания:

|  |  |  |  |
| --- | --- | --- | --- |
| *X* | *Y* | *Z* | *F* |
| 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 |

**Р-02**. *Символом F обозначено одно из указанных ниже логических выражений от трех аргументов: X, Y, Z. Дан фрагмент таблицы истинности выражения F:*

*Какое выражение соответствует F?*

1) **¬X ∧ ¬Y ∧ ¬Z** 2) **X ∧ Y ∧ Z** 3) **X ∧ ¬Y ∧ ¬Z** 4) **X ∨ ¬Y ∨ ¬Z**

**Решение (вариант 2):**

1. перепишем ответы в других обозначениях:   
    1)  2)  3)  4) 
2. в столбце F есть единственная единица для комбинации , простейшая функция, истинная (только) для этого случая, имеет вид , она есть среди приведенных ответов (ответ 3)
3. таким образом, правильный ответ – 3.

### Еще пример задания:

**Р-01**. *Дано логическое выражение, зависящее от 5 логических переменных:*

**X1 ∧ ¬X2 ∧ X3 ∧ ¬X4 ∧ X5**

*Сколько существует различных наборов значений переменных, при которых выражение ложно?*

1) 1 2) 2 3) 31 4) 32

**Решение (вариант 2):**

1. перепишем выражение в других обозначениях:   
    
2. таблица истинности для выражения с пятью переменными содержит 25 = 32 строки (различные комбинации значений этих переменных)
3. логическое произведение истинно в том и только в том случае, когда все сомножители равны 1, поэтому только один из этих вариантов даст истинное значение выражения, а остальные 32 – 1 = 31 вариант дают ложное значение.
4. таким образом, правильный ответ – 3.

### Ещё пример задания:

**Р-00**. Дан фрагмент таблицы истинности выражения F.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x1* | *x2* | *x3* | *x4* | *x5* | *x6* | *x7* | *F* |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| **1** | **0** | **1** | **0** | **1** | **1** | **0** | **0** |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |

Какое выражение соответствует F?

1) **¬x1 ∧ x2 ∧ ¬x3 ∧ x4 ∧ x5 ∧ ¬x6 ∧ ¬x7**

2) **¬x1 ∨ x2 ∨ ¬x3 ∨ x4 ∨ ¬x5 ∨ ¬x6 ∨ x7**

3) **x1 ∧ ¬x2 ∧ x3 ∧ ¬x4 ∧ x5 ∧ x6 ∧ ¬x7**

4) **x1 ∨ ¬x2 ∨ x3 ∨ ¬x4 ∨ ¬x5 ∨ x6 ∨ ¬x7**

**Решение (вариант 2):**

1. перепишем выражения 1-4 в других обозначениях:
2. 
3. 
4. 
5. 
6. поскольку в столбце F есть два нуля, это не может быть выражение, включающее только операции «ИЛИ» (логическое сложение), потому что в этом случае в таблице был бы только один ноль, поэтому варианты 2 и 4 отпадают:
7. 
8. 

аналогично, если бы в таблице был один ноль и две единицы, это не могла бы быть цепочка операций «И», которая всегда дает только одну единицу;

1. для того, чтобы в последней строке таблицы получилась единица, нужно применить операцию «НЕ» (инверсию) к переменным, значения которых в этой строке равны нулю, то есть к  и ; остальные переменные инвертировать не нужно, так как они равны 1; видим, что эти условия в точности совпадают с выражением 1, это и есть правильный ответ
2. Ответ: 1.

## 

|  |  |  |  |
| --- | --- | --- | --- |
| *X* | *Y* | *Z* | *F* |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 |

# 3-1 (базовый уровень, время – 3 мин)

**Тема**: Файловая система.

**Что нужно знать**:

* данные на дисках хранятся в виде файлов (наборов данных, имеющих имя)
* чтобы было удобнее разбираться с множеством файлов, их объединяют в каталоги (в *Windows* каталоги называются «папками»)
* каталог можно воспринимать как контейнер, в котором размещаются файлы и другие каталоги, которые называются подкаталогами или *вложенными* каталогами (они находятся внутри другого каталога, вложены в него)
* каталоги организованы в многоуровневую (иерархическую) структуру, которая называется «деревом каталогов»
* главный каталог диска (который пользователь видит, «открыв» диск, например, в Проводнике *Windows* или аналогичной программе) называется *корневым* каталогом или «корнем» диска, он обозначается буквой логического диска, за которой следует двоеточие и знак «\» (обратный слэш[[2]](#footnote-2)); например, **A:\** – это обозначение корневого каталога диска А
* каждый каталог (кроме корневого) имеет (один единственный!) «родительский» каталог – этот тот каталог, внутри которого находится данный каталог
* полный адрес каталога – это перечисление каталогов, в которые нужно войти, чтобы попасть в этот каталог (начиная с корневого каталога диска); например  
   **С:\USER\BIN\SCHOOL**
* полный адрес файла состоит из адреса каталога, в котором он находится, символа «\» и имени файла, например  
   **С:\USER\BIN\SCHOOL\Вася.txt**
* маска служит для обозначения (выделения) группы файлов, имена которых имеют общие свойства, например, общее расширение
* в масках, кроме «обычных» символов (допустимых в именах файлов) используются два специальных символа: звездочка «\*» и знак вопроса «?»;
* звездочка «\*» обозначает любой количество любых символов, в том числе, может обозначать пустую последовательность;
* знак вопроса «?» обозначает ровно один любой символ
* при выводе списка имен файлов они могут быть отсортированы по имени, типу (расширению), дате последнего изменения, размеру; это не меняет их размещения на диске;
* если установлена сортировка по имени или типу, сравнение идет по кодам символов, входящих в имя или в расширение

### Пример задания:

*В каталоге находится 6 файлов:*

**maveric.map**

**maveric.mp3**

**taverna.mp4**

**revolver.mp4**

**vera.mp3**

**zveri.mp3**

*Ниже представлено восемь масок. Сколько из них таких, которым соответствуют ровно четыре файла из данного каталога?*

**\*ver\*.mp\***

**\*?ver?\*.mp?**

**?\*ver\*.mp?\***

**\*v\*r\*?.m?p\***

**???\*???.mp\***

**???\*???.m\***

**\*a\*.\*a\***

**\*a\*.\*p\***

**Решение:**

1. фактически нужно проверить каждое имя файла на соответствие каждой маске
2. сформулируем по-русски правила, которые задаёт каждая маска:

|  |  |
| --- | --- |
| **\*ver\*.mp\*** | в имени содержится «ver», расширение начинается с «.mp» |
| **\*?ver?\*.mp?** | в имени содержится «ver», слева и справа от «ver» есть, по крайней мере, по одному символу; расширение начинается с «.mp» и состоит из 3 символов |
| **?\*ver\*.mp?\*** | в имени содержится «ver», слева от «ver» есть, по крайней мере, один символ; расширение начинается с «.mp» и состоит не менее, чем из 3 символов |
| **\*v\*r\*?.m?p\*** | в имени содержится буква «v» и после неё – «r», справа от «r» есть, по крайней мере, один символ; расширение начинается с «m», потом идёт ещё один какой-то символ, за ним – «p» |
| **???\*???.mp\*** | имя состоит не менее, чем из 6 символов, расширение начинается с «.mp» |
| **???\*???.m\*** | имя состоит не менее, чем из 6 символов, расширение начинается с «.m» |
| **\*a\*.\*a\*** | в имени есть буква «a», в расширении тоже есть буква «a» |
| **\*a\*.\*p\*** | в имени есть буква «a», в расширении – буква «p» |

1. составляем таблицу проверки, выделяем красным фоном символы, не совпадающие с маской; если все совпало, ячейку выделяем зеленым фоном:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **\*ver\*.mp\*** | **maveric.map** | **maveric.mp3** | **taverna.mp4** | **revolver.mp4** | **vera.mp3** | **zveri.mp3** |
| **\*?ver?\*.mp?** | **maveric.map** | **maveric.mp3** | **taverna.mp4** | **revolver.mp4** | **vera.mp3** | **zveri.mp3** |
| **?\*ver\*.mp?\*** | **maveric.map** | **maveric.mp3** | **taverna.mp4** | **revolver.mp4** | **vera.mp3** | **zveri.mp3** |
| **\*v\*r\*?.m?p\*** | **maveric.map** | **maveric.mp3** | **taverna.mp4** | **revolver.mp4** | **vera.mp3** | **zveri.mp3** |
| **???\*???.mp\*** | **maveric.map** | **maveric.mp3** | **taverna.mp4** | **revolver.mp4** | **vera.mp3** | **zveri.mp3** |
| **???\*???.m\*** | **maveric.map** | **maveric.mp3** | **taverna.mp4** | **revolver.mp4** | **vera.mp3** | **zveri.mp3** |
| **\*a\*.\*a\*** | **maveric.map** | **maveric.mp3** | **taverna.mp4** | **revolver.mp4** | **vera.mp3** | **zveri.mp3** |
| **\*a\*.\*p\*** | **maveric.map** | **maveric.mp3** | **taverna.mp4** | **revolver.mp4** | **vera.mp3** | **zveri.mp3** |

1. видим, что трём маскам: **?\*ver\*.mp?\***, **???\*???.m\*** и **\*a\*.\*p\***, соответствует ровно по 4 файла.
2. Ответ: 3.

### Ещё пример задания:

*В каталоге находится 7 файлов:*

**carga.mp3**

**cascad.mpeg**

**cassa.mp3**

**cassandra.mp4**

**castrol.mp4**

**picasa.map**

**picasa.mp4**

*Определите, по какой из перечисленных масок из этих 7 файлов будет*

*отобрана указанная группа файлов:*

**cascad.mpeg**

**cassa.mp3**

**cassandra.mp4**

**picasa.mp4**

1) **\*cas\*a\*.mp\*** 2) **\*ca\*a\*.mp\*** 3) **\*cas\*.mp\*** 4) **\*cas\*a\*.mp?**

**Решение:**

1. в этом задании две части: во-первых, нужно проверить, чтобы все отобранные имена файлов удовлетворяли выбранной маске; во-вторых, этой же маске НЕ должны соответствовать все отброшенные имена файлов, а именно:

**carga.mp3**

**castrol.mp4**

**picasa.map**

1. обратим внимание на маску 4: **\*cas\*a\*.mp?**, которая предполагает, что расширение состоит ровно из трёх символов; ей не соответствует имя выбранного файла **cascad.mpeg** (4 символа в расширении), поэтому эта маска не подходит
2. всем остальным маскам (1-3) имена выбранных файлов удовлетворяют (жёлтым маркером выделено совпадение с частями шаблона, звёздочка в маске обозначает любые символы, в том числе и отсутствие символов):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Маска** | **Совпадение с шаблоном** | | | |
| **\*cas\*a\*.mp\*** | **cascad.mpeg** | **cassa.mp3** | **cassandra.mp4** | **picasa.mp4** |
| **\*ca\*a\*.mp\*** | **cascad.mpeg** | **cassa.mp3** | **cassandra.mp4** | **picasa.mp4** |
| **\*cas\*.mp\*** | **cascad.mpeg** | **cassa.mp3** | **cassandra.mp4** | **picasa.mp4** |

1. теперь так же проверим на соответствие маскам имена невыбранных файлов (красным маркером отмечен первый символ или блок, не совпадающий с маской):

|  |  |  |  |
| --- | --- | --- | --- |
| **Маска** | **Совпадение с шаблоном** | | |
| **\*cas\*a\*.mp\*** | **carga.mp3** | **castrol.mp4** | **picasa.map** |
| **\*ca\*a\*.mp\*** | **carga.mp3** | **castrol.mp4** | **picasa.map** |
| **\*cas\*.mp\*** | **carga.mp3** | **castrol.mp4** | **picasa.map** |

из этой таблицы видим, что маска **\*ca\*a\*.mp\*** «пропускает» имя файла **carga.mp3**, а маска **\*cas\*.mp\*** «пропускает» имя файла **castrol.mp4** (эти имена выделены зелёным фоном), поэтому маски 2 и 3 не подходят

1. первая маска, **\*cas\*a\*.mp\***, отсекает все нужные файлы, и ей соответствуют все выбранные файлы, это и есть правильный ответ.
2. Ответ: 1.

### Ещё пример задания:

*Определите, какое из указанных имен файлов удовлетворяет маске:* **?hel\*lo.c?\***

1) **hello.c** 2) **hello.cpp** 3) **hhelolo.cpp** 4) **hhelolo.c**

**Решение:**

1. будем проверять соответствие файлов маске по частям, записывая результаты в таблицу
2. начнем с первой части маски, «**?hel**»; эта часть означает, что перед сочетанием «**hel**» в начале имени стоит один любой символ;
3. сразу видим, что первые два имени не подходят (начинаются прямо с «**hel**», без стартового символа), отмечаем их крестиком в таблице и больше не рассматриваем:

|  |  |
| --- | --- |
|  | **?hel** |
| **hello.cp** | **×** |
| **hello.cpp** | **×** |
| **hhelolo.cpp** | **hhelolo.cpp** |
| **hhelolo.c** | **hhelolo.c** |

желтым и фиолетовым маркером в таблице выделены соответствующие части маски и имен файлов (где есть совпадение)

1. для двух последних имен проверяем второй блок маски: после «**hel**» должна быть цепочка «**lo.c**», или вплотную (и это возможно!) или через произвольную «вставку», на которую указывает звездочка в маске; видим, что оба имени прошли проверку:

|  |  |  |
| --- | --- | --- |
|  | **?hel** | **?hel\*lo.c** |
| **hello.cp** | **×** |  |
| **hello.cpp** | **×** |  |
| **hhelolo.cpp** | **hhelolo.cpp** | **hhelolo.cpp** |
| **hhelolo.c** | **hhelolo.c** | **hhelolo.c** |

1. последняя часть маски, «**?\***», означает, что после «**lo.c**» должен стоять по крайне мере один любой символ (на это указывает знак «?»); проверяя это правило, обнаруживаем, что для последнего имени, «**hhelolo.c**», маска не подходит, поскольку после «**lo.c**» ни одного символа нет:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **?hel** | **?hel\*lo.c** | **?hel\*lo.c?\*** |
| **hello.cp** | **×** |  |  |
| **hello.cpp** | **×** |  |  |
| **hhelolo.cpp** | **hhelolo.cpp** | **hhelolo.cpp** | **hhelolo.cpp** |
| **hhelolo.c** | **hhelolo.c** | **hhelolo.c** | **×** |

1. таким образом, правильный ответ – 3.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + можно забыть, что звездочка «\*» может соответствовать и пустой последовательности; например, в рассмотренной задаче имя «**hhelolo.cp**» также соответствует маске   + можно забыть, что знак «?» НЕ может соответствовать пустой последовательности, а заменяет ровно 1 символ |

### Еще пример задания:

*Перемещаясь из одного каталога в другой, пользователь последовательно посетил каталоги* **DOC***,* **USER***,* **SCHOOL***,* **A:\***,* **LETTER***,* **INBOX***. При каждом перемещении пользователь либо спускался в каталог на уровень ниже, либо поднимался на уровень выше. Каково полное имя каталога, из которого начал перемещение пользователь?*

1) **A:\DOC**

2) **A:\LETTER\INBOX**

3) **А:\SCHOOL\USER\DOC**

4) **А:\DOC\USER\SCHOOL**

**Решение:**

1. сразу отметим, что здесь предполагается, что пользователь не переходил на другой диск;
2. в задачах, где нужно определить полный адрес файла или каталога, нужно начинать с поиска имени диска, в данном случае каталог находится на диске **A:\** (так обозначается корневой каталог)
3. поскольку в списке посещенных каталогов перед **A:\** стоит **SCHOOL**, пользователь мог попасть в корень диска **A:\** только через каталог **SCHOOL**, поэтому адрес стартового каталога начинается с **A:\SCHOOL**
4. так как среди предложенных вариантов только один удовлетворяет этому условию, можно остановиться (правильный ответ – 3), однако, давайте все же доведем процедуру до конца, это позволит выяснить некоторые интересные моменты, которые не сработали здесь, но могут сработать в других задачах
5. как пользователь попал в каталог **SCHOOL**? по условию он мог как подниматься, так и спускаться по дереву каталогов
6. явно он не мог спуститься из **USER** в **SCHOOL**, потому что **SCHOOL** находится в корневом каталоге и каждый каталог имеет только одного «родителя»; значит, пользователь поднялся из **USER** в **SCHOOL**, и начальная часть полного адреса **А:\SCHOOL\USER**
7. аналогично можно доказать, что пользователь поднялся в каталог **USER** из каталога **DOC**
8. таким образом, мы доказали, что правильный ответ – 3.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + можно забыть, что названия каталогов в полном имени перечисляются в порядке входа в них, начиная от корневого; пользователь *выходил* из каталогов, поднимаясь к корню диска **A:\**, поэтому проходил каталоги в обратном порядке (на эту ошибку рассчитан неверный ответ **А:\DOC\USER\SCHOOL**)   + можно перепутать каталог, из которого вышел пользователь, и каталог, где он в конечном счете оказался (на эту ошибку рассчитан неверный ответ **А:\LETTER\INBOX**)   + в условии есть лишние данные, которые только запутывают дело; например, имена каталогов **LETTER***,* **INBOX** никак не влияют на ответ, потому что пользователь попал в них уже *после* выхода в корневой каталог диска **A:\**, то есть, пройдя весь путь исходного каталога в обратном порядке |

### Еще пример задания:

*Каталог содержит файлы с именами*

а) **q.c**

б) **qq.cpp**

в) **qq.c**

г) **q1.c1**

д) **qaa.cmd**

е) **q12.cpp**

*Определите, в каком порядке будут показаны файлы, если выбрана сортировка по типу (по возрастанию).*

1) **авгдбе** 2) **авгдеб** 3) **абвгде** 4) **авдбег**

**Решение:**

1. при сортировке по типу сравниваются расширения имен файлов
2. при сравнении используют коды символов
3. отсутствие символа (когда расширение закончилось) считается «меньше» любого символа, то есть, файл с расширением **.с** будет находиться в списке выше, чем файлы с расширениями **.c1** и **.cmd**
4. коды цифр размещаются в таблице символов раньше, чем коды букв, то есть, файл с расширением **.с1** будет находиться в списке выше, чем файл с расширением **.сmd**
5. теперь можно распределить имена файлов по расширениям

а) **q.c**

в) **qq.c**

г) **q1.c1**

д) **qaa.cmd**

б) **qq.cpp**

е) **q12.cpp**

1. осталась еще одна проблема – решить, что делать, если расширения совпадают; в этом случае в большинстве программ для определенности используется дополнительная сортировка по имени, поэтому файл с именем **q12.cpp** будет стоять в списке выше, чем файл **qq.cpp** (код цифры ‘1' меньше, чем код буквы ‘q’)
2. в итоге получаем

а) **q.c**

в) **qq.c**

г) **q1.c1**

д) **qaa.cmd**

е) **q12.cpp**

б) **qq.cpp**

1. таким образом, мы доказали, что правильный ответ – 2.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + можно забыть правильно расставить имена файлов с одинаковыми расширениями (неверный ответ 1)   + нельзя сравнивать числовые значения: например, интуитивно кажется, что файл с расширением **.c10** «больше», чем файл с расширением **.c2**, однако это неверно, потому что код цифры '2' больше, чем код цифры '1’; поэтому файл с расширением **.c10** будет стоять в списке выше файла с расширением **.c2** (при сортировке по типу в порядке возрастания)   + можно забыть, что отсутствие кода (имя или расширение закончилось) «меньше» любого кода   + можно забыть, что коды цифр меньше, чем коды букв   + очень легко по невнимательности выбрать не тот ответ |

### Еще пример задания:

*Для групповых операций с файлами используются маски имен файлов. Маска представляет собой последовательность букв, цифр и прочих допустимых в именах файлов символов, в которых также могут встречаться следующие символы: Символ «?» (вопросительный знак) означает ровно один произвольный символ. Символ «\*» (звездочка) означает любую последовательность символов произвольной длины, в том числе «\*» может задавать и пустую последовательность. Определите, по какой из масок будет выбрана указанная группа файлов:*

**1234.xls**

**23.xml**

**234.xls**

**23.xml**

1) **\*23\*.?x\***

2) **?23?.x??**

3) **?23?.x\***

4) **\*23\*.???**

**Решение:**

1. рассмотрим первую маску **\*23\*.?x\***: она требует, чтобы в имени файла были стоящие подряд цифры 23 (это есть у всех заданных файлов) и в расширении слева от буквы «**x**» был один обязательный символ (ни одно имя файла не подходит); поэтому вариант 1 неверный
2. рассмотрим вторую маску **?23?.x??**: она требует, чтобы в имени файла слева и справа от цифр 23 стояло по одному символу (подходит только для первого имени файла, для остальных не подходит); поэтому вариант 2 неверный
3. рассмотрим третью маску **?23?.x\***: она не подходит по тем же причинам, что и вариант 2
4. рассмотрим последнюю маску **\*23\*.???**: она требует, чтобы в имени файла были стоящие подряд цифры 23 (это есть у всех заданных файлов) и в расширении было ровно три обязательных символа (это тоже верно для всех имен файлов); поэтому вариант 4 верный
5. таким образом, мы доказали, что правильный ответ – 4.

### Еще пример задания:

*Для групповых операций с файлами используются маски имен файлов. Маска представляет собой последовательность букв, цифр и прочих допустимых в именах файлов символов, в которых также могут встречаться следующие символы: Символ «?» (вопросительный знак) означает ровно один произвольный символ. Символ «\*» (звездочка) означает любую последовательность символов произвольной длины, в том числе «\*» может задавать и пустую последовательность.*

*В каталоге находятся пять файлов:*

**fort.docx**

**ford.docx**

**lord.doc**

**orsk.dat**

**port.doc**

*Определите, по какой из масок будет выбрана указанная группа файлов:*

**fort.docx**

**ford.docx**

**lord.doc**

**port.doc**

1) **\*o?\*.d?\***

2) **?o\*?.d\***

3) **\*or\*.doc?**

4) **?or?.doc?**

**Решение:**

1. авторы привели список файлов в каталоге, что наводит на мысль о каком-то подвохе, и он действительно есть
2. сформулируем правила, соответствующие каждой маске:
3. в имени файла есть буква «о», за которой следует еще хотя бы один символ; расширение начинается с буквы «d», за которой следует не менее 1 символа
4. в имени файла есть буква «о», перед ней стоит один символ, а за ней следует еще хотя бы один символ; расширение начинается с буквы «d»
5. в имени файла есть «or», расширение из 4-х символов, начинается с «doc»
6. в имени файла 4 символа, причем в середине – «or»; расширение из 4-х символов, начинается с «doc»
7. анализируя маску 1, с удивлением обнаруживаем, что под нее подходят ВСЕ исходные файлы, находящиеся в каталоге, поэтому этот ответ **неверный**
8. маске 2 соответствуют только нужные нам 4 файла, файл **orsk.dat** отсекается, потому что в маске перед «о» должен быть еще один символ; скорее всего, это и есть правильный ответ
9. маски 3 и 4 не соответствуют последним двум файлам, у которых расширение **doc** состоит из трех символов, поэтому это неверные варианты
10. таким образом, мы доказали, что правильный ответ – 2.

# 3-2 (базовый уровень, время – 3 мин)

**Тема**: Поиск и сортировка информации в базах данных.

**Что нужно знать**:

* при составлении условия отбора можно использовать знаки отношений <, <= (меньше или равно), >, >= (больше или равно), = (равно), <> (не равно)
* последовательность выполнения логических операций в сложных запросах: сначала выполняются отношения, затем – «И», потом – «ИЛИ»
* для изменения порядка выполнения операции используют скобки

### Пример задания:

*В фрагменте базы данных представлены сведения о родственных отношениях. Определите на основании приведенных данных фамилию и инициалы дяди Леоненко В.С. Пояснение: дядей считается брат отца или матери.*

**Таблица 1**

|  |  |  |
| --- | --- | --- |
| **ID** | **Фамилия\_И.О.** | **Пол** |
| 14 | Леоненко Н.А. | Ж |
| 23 | Геладзе И.П. | М |
| 24 | Геладзе П.И. | М |
| 25 | Геладзе П.П. | М |
| 34 | Леоненко А.И. | Ж |
| 35 | Леоненко В.С. | Ж |
| 33 | Леоненко С.С. | М |
| 42 | Вильямс О.С. | Ж |
| 44 | Гнейс А.С. | Ж |
| 45 | Гнейс В.А. | М |
| 47 | Вильямс П.О. | М |
| 57 | Паоло А.П. | Ж |
| 64 | Моор П.А. | Ж |

**Таблица 2**

|  |  |
| --- | --- |
| **ID\_Родителя** | **ID\_Ребенка** |
| 23 | 24 |
| 44 | 24 |
| 24 | 25 |
| 64 | 25 |
| 23 | 34 |
| 44 | 34 |
| 34 | 35 |
| 33 | 35 |
| 14 | 33 |
| 34 | 42 |
| 33 | 42 |
| 24 | 57 |
| 64 | 57 |

1) Геладзе И.П. 2) Геладзе П.И. 3) Гнейс А.С. 4) Леоненко Н.А.

**Решение:**

1. лицо мужского пола не может быть бабушкой, поэтому ответы 3 и 4 неверны
2. ищем в первой таблице *Леоненко В.С.*, определяем, что ее код 35
3. чтобы найти родителей *Леоненко В.С.*, ищем во второй таблице записи, где код ребенка равен 35: таким образом, её родители имеют коды 33 и 34
4. ищем бабушек и дедушек, то есть, записи во второй таблице, где код ребенка равен 33 или 34: соответствующие коды бабушек и дедушки Леоненко В.С. – это 14, 44 и 23
5. ищем детей персон с кодами 14, 44 и 23 – это братья и сестры родителей Леоненко В.С., то есть, её дяди и тёти; находим, что это человек с кодом 24, Геладзе П.И.
6. Ответ: 2

### Ещё пример задания:

*В фрагменте базы данных представлены сведения о родственных отношениях. Определите на основании приведенных данных фамилию и инициалы бабушки Ивановой А.И.*

**Таблица 1**

|  |  |  |
| --- | --- | --- |
| **ID** | **Фамилия\_И.О.** | **Пол** |
| 71 | Иванов Т.М. | М |
| 85 | Петренко И.Т. | М |
| 13 | Черных И.А. | Ж |
| 42 | Петренко А.И. | Ж |
| 23 | Иванова А.И. | Ж |
| 96 | Петренко Н.Н. | Ж |
| 82 | Черных А.Н. | M |
| 95 | Цейс Т.Н. | Ж |
| 10 | Цейс Н.А. | М |
|  | ... |  |

**Таблица 2**

|  |  |
| --- | --- |
| **ID\_Родителя** | **ID\_Ребенка** |
| 23 | 71 |
| **13** | **23** |
| **85** | **23** |
| 82 | 13 |
| 95 | 13 |
| 85 | 42 |
| 82 | 10 |
| 95 | 10 |
| ... | ... |

1) Иванов Т.М. 2) Черных И.А. 3) Цейс Т.Н. 4) Петренко Н.Н.

**Решение:**

1. лицо мужского пола не может быть бабушкой, поэтому ответ 1 неверен
2. ищем в первой таблице Иванову А.И., определяем, что ее код 23
3. чтобы найти родителей Ивановой А.И., ищем во второй таблице записи, где код ребенка равен 23: таким образом, её родители имеют коды 85 и 13
4. теперь ищем бабушек и дедушек, то есть, записи во второй таблице, где код ребенка равен 85 или 13: соответствующие коды бабушки и дедушки нашей Ивановой – это 82 и 95
5. в таблице 1 смотрим, кто имеет коды 82 и 95: это Черных А.Н. (мужского пола) и Цейс Т.Н (женского пола); последняя явно является бабушкой
6. ответ: 3

### Ещё пример задания:

*Результаты тестирования представлены в таблице:*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Фамилия | Пол | Математика | Русский язык | Химия | Информатика | Биология |
| Аганян | ж | 82 | 56 | 46 | 32 | 70 |
| Воронин | м | 43 | 62 | 45 | 74 | 23 |
| Григорчук | м | 54 | 74 | 68 | 75 | 83 |
| Роднина | ж | 71 | 63 | 56 | 82 | 79 |
| Сергеенко | ж | 33 | 25 | 74 | 38 | 46 |
| Черепанова | ж | 18 | 92 | 83 | 28 | 61 |

*Сколько записей в ней удовлетворяют условию «***Пол =’ж’ ИЛИ Химия > Биология***»?*

1) 5 2) 2 3) 3 4) 4

**Решение:**

1. заданное сложное условие отбора состоит из двух простых

**У1: Пол =’ж’**

**У2: Химия > Биология**

которые связаны с помощью логической операции «ИЛИ»

1. заметим, что столбцы «Фамилия», «Математика», «Русский язык» и «Информатика» никак не влияют на результат; уберем их из таблицы и добавим два новых столбца, в которых будем отмечать, выполняются ли условия У1 и У2 для каждой строчки

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Пол | Химия | Биология | **Пол =’ж’** | **Химия > Биология** |
| ж | 46 | 70 | **+** |  |
| м | 45 | 23 |  | **+** |
| м | 68 | 83 |  |  |
| ж | 56 | 79 | **+** |  |
| ж | 74 | 46 | **+** | **+** |
| ж | 83 | 61 | **+** | **+** |

1. логическая операция «ИЛИ» означает выполнение хотя бы одного из двух условия (или обоих одновременно), поэтому заданному сложному условию удовлетворяют все строки, где есть хотя бы один плюс; таких строк пять, они выделены зеленым фоном:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Пол | Химия | Биология | **Пол =’ж’** | **Химия > Биология** |
| ж | 46 | 70 | **+** |  |
| м | 45 | 23 |  | **+** |
| м | 68 | 83 |  |  |
| ж | 56 | 79 | **+** |  |
| ж | 74 | 46 | **+** | **+** |
| ж | 83 | 61 | **+** | **+** |

1. таким образом, правильный ответ – 1.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + можно перепутать действие операций «И» и «ИЛИ» (неверный ответ 2)   + можно перепутать порядок выполнения операций «И» и «ИЛИ», если они обе используются в сложном условии   + помните, что в бланк нужно вписать не количество записей, удовлетворяющих условию, а номер ответа из предложенных |

### Еще пример задания:

*Из правил соревнования по тяжелой атлетике: Тяжелая атлетика – это прямое соревнование, когда каждый атлет имеет три попытки в рывке и три попытки в толчке. Самый тяжелый вес поднятой штанги в каждом упражнении суммируется в общем зачете. Если спортсмен потерпел неудачу во всех трех попытках в рывке, он может продолжить соревнование в толчке, но уже не сможет занять какое-либо место по сумме 2-х упражнений. Если два спортсмена заканчивают состязание с одинаковым итоговым результатом, высшее место присуждается спортсмену с меньшим весом. Если же вес спортсменов одинаков, преимущество отдается тому, кто первым поднял победный вес. Таблица результатов соревнований по тяжелой атлетике:*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Фамилия И.О.* | *Вес спортсмена* | *Взято в рывке* | *Рывок с попытки* | *Взято в толчке* | *Толчок с попытки* |
| *Айвазян Г.С.* | *77,1* | *150,0* | *3* | *200,0* | *2* |
| *Викторов М.П.* | *79,1* | *147,5* | *1* | *202,5* | *1* |
| *Гордезиани Б.Ш.* | *78,2* | *147,5* | *2* | *200,0* | *1* |
| *Михальчук М.С.* | *78,2* | *147,5* | *2* | *202,5* | *3* |
| *Пай С.В.* | *79,5* | *150,0* | *1* | *200,0* | *1* |
| *Шапсугов М.Х.* | *77,1* | *147,5* | *1* | *200,0* | *1* |

*Кто победил в общем зачете (по сумме двух упражнений)?*

1) Айвазян Г.С. 2) Викторов М.П. 3) Михальчук М.С. 4) Пай С.В.

**Решение:**

1. основная сложность этой задачи (особенно для тех, кто не увлекается тяжелой атлетикой) состоит в том, что бы внимательно прочитать и понять достаточно запутанные условия соревнований
2. можно убрать из таблицы всех участников, кроме тех, которые упомянуты в ответах
3. в условии читаем первое правило для определения победителя: «*Самый тяжелый вес поднятой штанги в каждом упражнении суммируется в общем зачете*», поэтому добавим в таблицу столбец «Общий зачет», в котором для каждого спортсмена сложим веса, взятые в рывке и в толчке

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *Фамилия И.О.* | *Вес спортсмена* | *Взято в рывке* | *Рывок с попытки* | *Взято в толчке* | *Толчок с попытки* | *Общий зачет* |
| *Айвазян Г.С.* | *77,1* | *150,0* | *3* | *200,0* | *2* | *350,0* |
| *Викторов М.П.* | *79,1* | *147,5* | *1* | *202,5* | *1* | *350,0* |
| *Михальчук М.С.* | *78,2* | *147,5* | *2* | *202,5* | *3* | *350,0* |
| *Пай С.В.* | *79,5* | *150,0* | *1* | *200,0* | *1* | *350,0* |

1. все интересующие нас участники набрали одинаковый результат, поэтому по этому критерию выявить победителя не удалось; читаем далее: «*Если два спортсмена заканчивают состязание с одинаковым итоговым результатом, высшее место присуждается спортсмену с меньшим весом*»; отсюда сразу следует, что победитель – Айвазян Г.С., поскольку его вес – наименьший среди всех участников
2. таким образом, правильный ответ – 1.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + длинное и запутанное условие, которое нужно формализовать   + можно перепутать порядок применения условий; например, если сначала учесть количество попыток, то победителем будет Викторов   + лишняя информация, которая не влияет на решение задачи, но осложняет восприятие длинного условия и выделение действительно значимой информации |

### Еще пример задания:

*В таблице представлены несколько записей из базы данных «Расписание»:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *№* | *Учитель* | *День\_недели* | *Номер\_урока* | *Класс* |
| *1* | *Айвазян Г.С.* | *понедельник* | *3* | *8А* |
| *2* | *Айвазян Г.С.* | *понедельник* | *4* | *9Б* |
| *3* | *Айвазян Г.С.* | *вторник* | *2* | *10Б* |
| *4* | *Михальчук М.С.* | *вторник* | *2* | *9А* |
| *5* | *Пай С.В.* | *вторник* | *3* | *10Б* |
| *6* | *Пай С.В.* | *среда* | *5* | *8Б* |

*Укажите номера записей, которые удовлетворяют условию***Номер\_урока > 2 И Класс > '8А'**

1) 1, 6 2) 2, 6 3) 2, 5, 6 4) 1, 2, 5, 6

**Решение:**

1. уберем из таблицы всю лишнюю информацию, оставив только номер записи, номер урока и класс:

|  |  |  |
| --- | --- | --- |
| *№* | *Номер\_урока* | *Класс* |
| *1* | *3* | *8А* |
| *2* | *4* | *9Б* |
| *3* | *2* | *10Б* |
| *4* | *2* | *9А* |
| *5* | *3* | *10Б* |
| *6* | *5* | *8Б* |

1. логическая связка **И** означает одновременное выполнение двух условий; оставим в таблице только те строки, для которых выполняется первое из двух условий, **Номер\_урока > 2**

|  |  |  |
| --- | --- | --- |
| *№* | *Номер\_урока* | *Класс* |
| *1* | *3* | *8А* |
| *2* | *4* | *9Б* |
| *5* | *3* | *10Б* |
| *6* | *5* | *8Б* |

1. теперь нужно из оставшихся строк отобрать те, для которых **Класс > '8А'**; на взгляд «нормального» человека, этому условию удовлетворяют последние 3 строчки, однако это неправильный ответ
2. дело в том, что в данном случае поле **Класс** имеет тип «символьная строка», поэтому сравнение будет **Класс > '8А'** выполняться **по кодам** символов, начиная с первого
3. цифры во всех кодовых таблицах располагаются последовательно, одна за другой, от 0 до 9
4. поэтому код цифры «1» меньше, чем код цифры «8», и строка 5 не удовлетворяет условию **Класс > '8А'**
5. к счастью, русские буквы **А** и **Б** во всех кодовых таблицах расположены друг за другом[[3]](#footnote-3), поэтому сравнение пройдет «нормально», условие **Класс > '8А'** для записи № 6 будет истинно
6. в результате после применения условия **Класс > '8А'** остаются две записи

|  |  |  |
| --- | --- | --- |
| *№* | *Номер\_урока* | *Класс* |
| *2* | *4* | *9Б* |
| *6* | *5* | *8Б* |

1. таким образом, правильный ответ – 2.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + помните, что символьные строки сравниваются по кодам символов   + цифры в таблице кодов стоят подряд от 0 до 9 (коды 48-57)   + в кодировке *Windows* русские буквы стоят по алфавиту |

### Еще пример задания:

*База данных о торговых операциях дистрибутора состоит из трех связанных таблиц. Ниже даны фрагменты этих таблиц.*

*Таблица зарегистрированных дилеров*

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование организации** | **ID дилера** | **Регион** | **Адрес** |
| *ООО «Вектор»* | *D01* | *Башкортостан* | *г. Уфа, ул. Школьная, 15* |
| *АО «Луч»* | *D02* | *Татарстан* | *г. Казань, ул. Прямая, 17* |
| *АОЗТ «Прямая»* | *D03* | *Адыгея* | *г. Майкоп, просп. Мира, 8* |
| *ООО «Окружность»* | *D04* | *Дагестан* | *г. Дербент, ул. Замковая, 6* |
| *ИЧП Скаляр* | *D05* | *Дагестан* | *г. Махачкала, ул. Широкая, 28* |
| *АО «Ромб»* | *D06* | *Татарстан* | *г. Набережные Челны, ул. Заводская, 4* |

*Таблица отгрузки товара*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Номер накладной** | **Отгружено дилеру** | **Артикул товара** | **Отгружено упаковок** | **Дата отгрузки** |
| *001* | *D01* | *01002* | *300* | *5/01/2009 г.* |
| *002* | *D02* | *01002* | *100* | *5/01/2009 г.* |
| *003* | *D06* | *01002* | *200* | *5/01/2009 г.* |
| *004* | *D01* | *02002* | *20* | *5/01/2009 г.* |
| *005* | *D02* | *02002* | *30* | *5/01/2009 г.* |
| *006* | *D02* | *01003* | *20* | *6/01/2009 г.* |

*Таблица товаров*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование товара | Артикул | Отдел | Количество единиц в упаковке | Брутто вес упаковки |
| Фломастеры, пачка 24 шт. | 01001 | Канцтовары | 24 | 5 |
| Бумага А4,пачка 500 листов | 01002 | Канцтовары | 5 | 10 |
| Скрепки металлические, 1000 шт. | 01003 | Канцтовары | 48 | 20 |
| Розетки трехфазные | 02001 | Электротовары | 12 | 2 |
| Лампа накаливания, 60 Вт | 02002 | Электротовары | 100 | 8 |
| Выключатель 2-клавишный | 02003 | Электротовары | 48 | 7 |

Сколько пачек бумаги было отгружено в Татарстан 5 января 2009 года?

1) 100 2) 200 3) 300 4) 1500

**Решение:**

1. уберем из таблиц всю лишнюю информацию; во-первых, нас интересует только бумага и только количество пачек, поэтому таблица товаров сводится к одной строчке

|  |  |  |
| --- | --- | --- |
| Наименование товара | Артикул | Количество единиц в упаковке |
| Бумага А4,пачка 500 листов | 01002 | 5 |

1. во-вторых, нас интересуют только дилеры из Татарстана, причем их названия и адреса не дают полезной информации, нужен только код; вот что остается от таблицы дилеров:

|  |  |
| --- | --- |
| **ID дилера** | **Регион** |
| *D02* | *Татарстан* |
| *D06* | *Татарстан* |

1. из таблицы отгрузки товара выбираем только информацию о поставках этим дилерам:

|  |  |  |  |
| --- | --- | --- | --- |
| **Отгружено дилеру** | **Артикул товара** | **Отгружено упаковок** | **Дата отгрузки** |
| *D02* | *01002* | *100* | *5/01/2009 г.* |
| *D06* | *01002* | *200* | *5/01/2009 г.* |
| *D02* | *02002* | *30* | *5/01/2009 г.* |
| *D02* | *01003* | *20* | *6/01/2009 г.* |

1. в последней таблице отмечаем строчки, которые относятся к бумаге (артикул 01002) и дате 5/01/2009:

|  |  |  |  |
| --- | --- | --- | --- |
| **Отгружено дилеру** | **Артикул товара** | **Отгружено упаковок** | **Дата отгрузки** |
| *D02* | *01002* | *100* | *5/01/2009 г.* |
| *D06* | *01002* | *200* | *5/01/2009 г.* |

1. таким образом, в 5/01/2009 в Татарстан было отгружено 300 упаковок бумаги
2. теперь вспоминаем, что в таблице товаров сказано, что в каждой упаковке 5 пачек, поэтому всего отгружено 1500 пачек
3. таким образом, правильный ответ – 4.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + обратите внимание, что спрашивается количество пачек, а не количество упаковок; среди ответов есть «отвлекающий» вариант 300 – после выполнения шага 5 появляется соблазн выбрать именно его |

# 4 (базовый уровень, время – 1 мин)

**Тема**: Системы счисления и двоичное представление информации в памяти компьютера.

**Что нужно знать**:

* перевод чисел между десятичной, двоичной, восьмеричной и шестнадцатеричной системами счисления (см. презентацию «Системы счисления»)

|  |
| --- |
| **Полезно помнить, что в двоичной системе:**   * четные числа оканчиваются на 0, нечетные – на 1; * числа, которые делятся на 4, оканчиваются на 00, и т.д.; числа, которые делятся на 2k, оканчиваются на *k* нулей * если число N принадлежит интервалу 2k-1 ≤ N < 2k, в его двоичной записи будет всего *k* цифр, например, для числа **125**:   26 = 64 ≤ **125** < 128 = 27, 125 = 11111012 (7 цифр)   * числа вида 2k записываются в двоичной системе как единица и *k* нулей, например:   16 = 24 = 100002   * числа вида 2k-1записываются в двоичной системе *k* единиц, например:   15 = 24-1 = 11112   * если известна двоичная запись числа N, то двоичную запись числа 2·N можно легко получить, приписав в конец ноль, например:  15 = 11112, 30 = 111102, 60 = 1111002, 120 = 11110002 |

* отрицательные целые числа хранятся в памяти в двоичном дополнительном коде (подробнее см. презентацию «Компьютер изнутри»)
* для перевода отрицательного числа **(-a)** в двоичный дополнительный код нужно сделать следующие операции:
  + перевести число **a-1** в двоичную систему счисления
  + сделать инверсию битов: заменить все нули на единицы и единицы на нули в пределах разрядной сетки (см. пример далее)

### Пример задания:

*Укажите наименьшее четырёхзначное восьмеричное число, двоичная запись которого содержит 5 единиц. В ответе запишите только само восьмеричное число, основание системы счисления указывать не нужно.*

**Решение:**

1. вообще, минимальное двоичное число, содержащее 5 единиц – это 111112, но в восьмеричной системе оно записывается как 37 – двухзначное число
2. минимальное четырёхзначное восьмеричное число – 10008 = 1 000 000 0002, для решения задачи в конце этого числа нужно заменить четыре нуля на единицы:

1 000 001 1112  = 10178

1. Ответ: 1017

### Ещё пример задания:

*Сколько единиц в двоичной записи десятичного числа 519?*

**Решение:**

1. проще всего представить заданное число в виде суммы степеней числа 2:

519 = 512 + 7 = 29 + 4 + 3 = 29 + 22 + 2 + 1 = 29 + 22 + 21 + 20

1. количество единиц в двоичной записи числа равно количеству слагаемых в таком разложении
2. Ответ: 4

### Ещё пример задания:

*Даны 4 числа, они записаны с использованием различных систем счисления. Укажите среди этих чисел то, в двоичной записи которого содержится ровно 6 единиц. Если таких чисел несколько, укажите наибольшее из них.*

1) 6310 \* 410 2) F816 + 110 3) 3338 4) 111001112

**Решение:**

1. нужно перевести все заданные числа в двоичную систему, подсчитать число единиц и выбрать наибольшее из чисел, в которых ровно 6 единиц;
2. для первого варианта переведем оба сомножителя в двоичную систему:

63­10 = 111111­2 410 = 100­2

в первом числе ровно 6 единиц, умножение на второе добавляет в конец два нуля:

63­10 \* 410 = 111111­2 \* 100­2 = 111111­002

то есть в этом числе 6 единиц

1. для второго варианта воспользуемся связью между шестнадцатеричной и двоичной системами счисления: каждую цифру шестнадцатеричного числа можно переводить отдельно в тетраду (4 двоичных цифры):

F­16 = 1111­2 816 = 100­02 F816 = 1111 10002

после добавления единицы F816 + 1 = 1111 10012 также получаем число, содержащее ровно 6 единиц, но оно меньше, чем число в первом варианте ответа

1. для третьего варианта используем связь между восьмеричной и двоичной системами: каждую цифру восьмеричного числа переводим отдельно в триаду (группу из трёх) двоичных цифр:

3338 = 011 011 011­2 = 110110112

это число тоже содержит 6 единиц, но меньше, чем число в первом варианте ответа

1. последнее число 111001112 уже записано в двоичной системе, оно тоже содержит ровно 6 единиц, но меньше первого числа
2. таким образом, все 4 числа, указанные в вариантах ответов содержат ровно 6 единиц, но наибольшее из них – первое
3. Ответ: 1.

### Ещё пример задания:

*Сколько единиц в двоичной записи числа 1025?*

1) 1 2) 2 3) 10 4) 11

**Решение (вариант 1, прямой перевод):**

1. переводим число 1025 в двоичную систему: 1025 = 10000000001­2
2. считаем единицы, их две
3. Ответ: 2

|  |
| --- |
| **Возможные проблемы**:  легко запутаться при переводе больших чисел. |

**Решение (вариант 2, разложение на сумму степеней двойки):**

1. тут очень полезно знать наизусть таблицу степеней двойки, где 1024 = 210 и 1 = 20
2. таким образом, 1025= 1024 + 1 = 210 + 20
3. вспоминая, как переводится число из двоичной системы в десятичную (значение каждой цифры умножается на 2 в степени, равной её разряду), понимаем, что в двоичной записи числа ровно столько единиц, сколько в приведенной сумме различных степеней двойки, то есть, 2
4. Ответ: 2

|  |
| --- |
| **Возможные проблемы**:  нужно помнить таблицу степеней двойки. |
| **Когда удобно использовать:**   * когда число чуть больше какой-то степени двойки |

### Ещё пример задания:

*Дано: *и **. *Какое из чисел с*, записанных в двоичной системе счисления, удовлетворяет  *неравенству a < c < b?*

1) 110110012 2) 110111002 3) 110101112 4) 110110002

**Общий подход:**

перевести все числа (и исходные данные, и ответы) в одну (любую!) систему счисления и сравнить.

**Решение (вариант 1, через десятичную систему):**

1. **
2. **
3. переводим в десятичную систему все ответы:

110110012 = 217, 11011100 2= 220, 110101112 = 215, 110110002=216

1. очевидно, что между числами 215 и 217 может быть только 216
2. таким образом, верный ответ – 4 .

|  |
| --- |
| **Возможные проблемы**:  арифметические ошибки при переводе из других систем в десятичную. |

**Решение (вариант 2, через двоичную систему):**

1. ** (каждая цифра шестнадцатеричной системы *отдельно* переводится в четыре двоичных – *тетраду*);
2. ** (каждая цифра восьмеричной системы *отдельно* переводится в три двоичных – *триаду*, старшие нули можно не писать);
3. теперь нужно сообразить, что между этими числами находится только двоичное число 110110002 – это ответ 4.

|  |
| --- |
| **Возможные проблемы**:  запись двоичных чисел однородна, содержит много одинаковых символов – нулей и единиц, поэтому легко запутаться и сделать ошибку. |

**Решение (вариант 3, через восьмеричную систему):**

1. **(сначала перевели в двоичную систему, потом двоичную запись числа разбили на триады **справа налево**, каждую триаду перевели *отдельно* в десятичную систему, так как для чисел от 0 до 7 их восьмеричная запись совпадает с десятичной);
2. **, никуда переводить не нужно;
3. переводим в восьмеричную систему все ответы:

110110012 = 011 011 0012 = 3318 (разбили на триады **справа налево**, каждую триаду перевели *отдельно* в десятичную систему, как в п. 1)

11011100 2= 3348, 110101112 = 3278, 110110002=3308

1. в восьмеричной системе между числами 3278 и 3318 может быть только 3308
2. таким образом, верный ответ – 4 .

|  |
| --- |
| **Возможные проблемы**:  нужно помнить двоичную запись чисел от 0 до 7 (или переводить эти числа в двоичную систему при решении). |

**Решение (вариант 4, через шестнадцатеричную систему):**

1. ** никуда переводить не нужно;
2. ** (сначала перевели в двоичную систему, потом двоичную запись числа разбили на тетрады **справа налево**, каждую тетраду перевели в шестнадцатеричную систему; при этом тетрады можно переводить из двоичной системы в *десятичную,* а затем заменить все числа, большие 9, на буквы – A, B, C, D, E, F);
3. переводим в шестнадцатеричную систему все ответы:

110110012 = 1101 10012 = D916 (разбили на тетрады **справа налево**, каждую тетраду перевели *отдельно* в десятичную систему, все числа, большие 9, заменили на буквы – A, B, C, D, E, F, как в п. 1)

11011100 2= DC16, 110101112 = D716, 110110002=D816

1. в шестнадцатеричной системе между числами D716 и D916 может быть только D816
2. таким образом, верный ответ – 4 .

|  |
| --- |
| **Возможные проблемы**:  нужно помнить двоичную запись чисел от 0 до 15 (или переводить эти числа в двоичную систему при решении). |

**Выводы:**

* есть несколько способов решения, «каждый выбирает для себя»;
* наиболее сложные вычисления – при переводе всех чисел в десятичную систему, можно легко ошибиться;
* сравнивать числа в двоичной системе сложно, также легко ошибиться;
* видимо, *в этой задаче* наиболее простой вариант – использовать восьмеричную систему, нужно просто запомнить двоичные записи чисел от 0 до 7 и аккуратно все сделать;
* в других задачах может быть так, что выгоднее переводить все в десятичную или шестнадцатеричную систему счисления.

### Еще пример задания:

*Для хранения целого числа со знаком используется один байт. Сколько единиц содержит внутреннее представление числа* **(-78)***?*

1) 3 2) 4 3) 5 4) 6

**Решение (вариант 1, классический):**

1. переводим число 78 в двоичную систему счисления:

78 = 64 + 8 + 4 + 2 = 26 + 23 + 22 + 21 = 10011102

1. по условию число занимает в памяти 1 байт = 8 бит, поэтому нужно представить число с помощью 8 разрядов
2. чтобы получилось всего 8 разрядов (бит), добавляем впереди один ноль:

78 = 010011102

1. делаем инверсию битов (заменяем везде 0 на 1 и 1 на 0):

010011102 → 101100012

1. добавляем к результату единицу

101100012 + 1 = 101100102

это и есть число **(-78)** в двоичном дополнительно коде

1. в записи этого числа 4 единицы
2. таким образом, верный ответ – 2 .

|  |
| --- |
| **Возможные ловушки и проблемы**:   * нужно не забыть в конце добавить единицу, причем это может быть не так тривиально, если будут переносы в следующий разряд – тут тоже есть шанс ошибиться из-за невнимательности |

**Решение (вариант 2, неклассический):**

1. переводим число 78 – 1=77 в двоичную систему счисления:

77 = 64 + 8 + 4 + 1 = 26 + 23 + 22 + 20 = 10011012

1. по условию число занимает в памяти 1 байт = 8 бит, поэтому нужно представить число с помощью 8 разрядов
2. чтобы получилось всего 8 разрядов (бит), добавляем впереди один ноль:

77 = 010011012

1. делаем инверсию битов (заменяем везде 0 на 1 и 1 на 0):

010011012 → 101100102

это и есть число **(-78)** в двоичном дополнительно коде

1. в записи этого числа 4 единицы
2. таким образом, верный ответ – 2 .

|  |
| --- |
| **Возможные ловушки и проблемы**:   * нужно помнить, что в этом способе в двоичную систему переводится не число **a**, а число  **a-1**; именно этот прием позволяет избежать добавления единицы в конце (легче вычесть в десятичной системе, чем добавить в двоичной) |

**Решение (вариант 3, неклассический):**

1. переводим число 78 в двоичную систему счисления:

78 = 64 + 8 + 4 + 2 = 26 + 23 + 22 + 21 = 10011102

1. по условию число занимает в памяти 1 байт = 8 бит, поэтому нужно представить число с помощью 8 разрядов
2. чтобы получилось всего 8 разрядов (бит), добавляем впереди один ноль:

78 = 010011102

1. для всех битов, которые стоят **слева от младшей единицы**, делаем инверсию битов (заменяем везде 0 на 1 и 1 на 0):

010011102 → 101100102

это и есть число **(-78)** в двоичном дополнительно коде

1. в записи этого числа 4 единицы
2. таким образом, верный ответ – 2 .

|  |
| --- |
| **Возможные ловушки и проблемы**:   * нужно помнить, что при инверсии младшая единица и все нули после нее не меняются |

# 5 (базовый уровень, время – 3 мин)

**Тема**: Использование информационных моделей (таблицы, диаграммы, графики).  
 Перебор вариантов, выбор лучшего по какому-то признаку.

**Что нужно знать**:

* в принципе, особых дополнительных знаний, кроме здравого смысла и умения перебирать варианты (не пропустив ни одного!) здесь, как правило, не требуется
* полезно знать, что такое *граф* (это набор вершин и соединяющих их ребер) и как он описывается в виде таблицы, хотя, как правило, все необходимые объяснения даны в формулировке задания
* чаще всего используется *взвешенный граф*, где с каждым ребром связано некоторое число (вес), оно может обозначать, например, расстояние между городами или стоимость перевозки
* рассмотрим граф (рисунок слева), в котором 5 вершин (A, B, C, D и E); он описывается таблицей, расположенной в центре; в ней, например, число 4 на пересечении строки В и столбца С означает, что, во-первых, есть ребро, соединяющее В и С, и во-вторых, вес этого ребра равен 4; пустая клетка на пересечении строки А и столбца В означает, что ребра из А в В нет

1

2

4

2

3

1

2

4

2

3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | Е |
| A |  |  | 3 | 1 |  |
| B |  |  | 4 |  | 2 |
| C | 3 | 4 |  |  | 2 |
| D | 1 |  |  |  |  |
| Е |  | 2 | 2 |  |  |

* обратите внимание, что граф по заданной таблице (она еще называется *весовой матрицей*) может быть нарисован по-разному; например, той же таблице соответствует граф, показанный на рисунке справа от нее
* в приведенном примере матрица симметрична относительно главной диагонали; это может означать, например, что стоимости перевозки из В в С и обратно равны (это не всегда так)
* желательно научиться быстро (и правильно) строить граф по весовой матрице и наоборот

### Пример задания:

**Р-06.** *Между населёнными пунктами A, B, C, D, E, F построены дороги, протяжённость которых приведена в таблице. (Отсутствие числа в таблице означает, что прямой дороги между пунктами нет.)*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F |
| A |  | 2 | 4 | 8 |  | 16 |
| B | 2 |  |  | 3 |  |  |
| C | 4 |  |  | 3 |  |  |
| D | 8 | 3 | 3 |  | 5 | 3 |
| E |  |  |  | 5 |  | 5 |
| F | 16 |  |  | 3 | 5 |  |

*Определите длину кратчайшего пути между пунктами A и F, проходящего через пункт E и не проходящего через пункт B. Передвигаться можно только по указанным дорогам.*

**Решение:**

1. поскольку нас интересуют только маршруты, НЕ проходящие через пункт В, столбец и строку, соответствующие этому пункту, можно удалить из таблицы:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | C | D | E | F |
| A |  | 4 | 8 |  | 16 |
| C | 4 |  | 3 |  |  |
| D | 8 | 3 |  | 5 | 3 |
| E |  |  | 5 |  | 5 |
| F | 16 |  | 3 | 5 |  |

1. дальше действуем так же, как показано при решении следующих далее разобранных задач; причем из всех маршрутов нужно оставить только те, которые проходят через пункт Е
2. первый шаг от А (в скобках указаны длины маршрутов):

АС (4), AD (8)

прямой маршрут AF не рассматриваем, потому что он не проходит через пункт E

1. второй шаг

ACD (7), ADC (11), ADE (13)

маршрут ADF не рассматриваем, потому что он не проходит через пункт E

1. третий шаг:

ACDE (12), ADEF (18)

маршрут ADEF дошел до пункта назначения;

маршрут ADC продолжать не имеет смысла, потому что из C можно проехать только в пункты A и D, где мы уже были;

маршрут ACDF не рассматриваем, потому что он не проходит через пункт E

1. четвертый шаг:

ACDEF(17)

1. этот маршрут тоже дошел до пункта назначения, его длина меньше, чем для предыдущего, его и выбираем
2. Ответ: 17.

### Ещё пример задания:

**Р-05.** *Между населёнными пунктами A, B, C, D, E, F, Z построены дороги с односторонним движением. В таблице указана протяжённость каждой дороги. Отсутствие числа в таблице означает, что прямой дороги между пунктами нет. Например, из A в B есть дорога длиной 4 км, а из B в A дороги нет.*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | Z |
| A |  | 4 | 6 |  |  |  | 30 |
| B |  |  | 3 |  |  |  |  |
| C |  |  |  | 11 |  |  | 27 |
| D |  |  |  |  | 4 | 7 | 10 |
| E |  |  |  |  |  | 4 | 8 |
| F |  |  |  |  | 5 |  | 2 |
| Z | 29 |  |  |  |  |  |  |

*Сколько существует таких маршрутов из A в Z, которые проходят через 6 и более населенных пунктов? Пункты A и Z при подсчете учитывать. Два раза проходить через один пункт нельзя.*

**Решение:**

1. обратим внимание, что числа в таблице нас совсем не интересуют – достаточно знать, что между данными пунктами есть дорога
2. нам нужно найти все пути, которые проходят через 6 и более пунктов, считая начальный и конечный; то есть между A и Z должно быть не менее 4 промежуточных пункта
3. начнем с перечисления всех маршрутов из А, которые проходят через 2 пункта; по таблице видим, что из A можно ехать в B, C и Z; количество пунктов на маршруте будем записывать сверху:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | 3 | 4 | 5 | 6 | 7 |
| AB |  |  |  |  |  |
| AC |  |  |  |  |  |
| AZ |  |  |  |  |  |

1. маршрут AZ нас не интересует, хотя он и пришел в конечный пункт, он проходит меньше, чем через 6 пунктов (только через 2!); здесь и далее такие «неинтересные» маршруты из A в Z будем выделять серым фоном
2. теперь ищем все маршруты, проходящие через 3 пункта; из B можно ехать только в C, а из С – в D и Z:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | 3 | 4 | 5 | 6 | 7 |
| AB | ABC |  |  |  |  |
| AC | ACD |  |  |  |  |
| ACZ |  |  |  |  |
| AZ |  |  |  |  |  |

1. далее из C едем в D и Z, а из D – в E, F и Z:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | 3 | 4 | 5 | 6 | 7 |
| AB | ABC | ABCD |  |  |  |
| ABCZ |  |  |  |
| AC | ACD | ACDE |  |  |  |
| ACDF |  |  |  |
| ACDZ |  |  |  |
| ACZ |  |  |  |  |
| AZ |  |  |  |  |  |

1. строим следующий уровень только для тех маршрутов, которые ещё не пришли в Z:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | 3 | 4 | 5 | 6 | 7 |
| AB | ABC | ABCD | ABCDE |  |  |
| ABCDF |  |  |
| ABCDZ |  |  |
| ABCZ |  |  |  |
| AC | ACD | ACDE | ACDEF |  |  |
| ACDEZ |  |  |
| ACDF | ACDFE |  |  |
| ACDFZ |  |  |
| ACDZ |  |  |  |
| ACZ |  |  |  |  |
| AZ |  |  |  |  |  |

1. следущие два уровня дают «интересные» маршруты, проходящие через 6 или 7 пунктов:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | 3 | 4 | 5 | 6 | 7 |
| AB | ABC | ABCD | ABCDE | ABCDEF | ABCDEFZ |
| ABCDEZ |  |
| ABCDF | ABCDFE | ABCDFEZ |
| ABCDFZ |  |
| ABCDZ |  |  |
| ABCZ |  |  |  |
| AC | ACD | ACDE | ACDEF | ACDEFE |  |
| ACDEFZ |  |
| ACDEZ |  |  |
| ACDF | ACDFE | ACDFEF |  |
| ACDFEZ |  |
| ACDFZ |  |  |
| ACDZ |  |  |  |
| ACZ |  |  |  |  |
| AZ |  |  |  |  |  |

1. на последней схеме зелёным фоном выделены «интересные» маршруты, их всего 6; красным фоном отмечены маршруты, в которых получился цикл – они дважды проходят через один и тот же пункт; такие маршруты запрещены и мы далее их не рассматриваем
2. Ответ: 6.
3. можно было нарисовать схему возможных маршрутов в виде дерева:

A

AB

AС

AZ

ABC

ACD

ACZ

ABCD

ABCZ

ACDE

ACDF

ACDZ

ABCDE

ABCDZ

ABCDF

ACDEF

ACDEZ

ACDFE

ACDFZ

ABCDEF

ABCDEZ

ABCDFE

ABCDFZ

ACDEFE

ACDEFZ

ACDFEF

ACDFEZ

ABCDEFZ

ABCDFEZ

1

2

3

4

5

6

7

### Ещё пример задания:

**Р-04.** *Между населёнными пунктами A, B, C, D, E, F, G построены дороги, протяжённость которых приведена в таблице. (Отсутствие числа в таблице означает, что прямой дороги между пунктами нет.)*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G |
| A |  | 5 |  | 12 |  |  | 25 |
| B | 5 |  |  | 8 |  |  |  |
| C |  |  |  | 2 | 4 | 5 | 10 |
| D | 12 | 8 | 2 |  |  |  |  |
| E |  |  | 4 |  |  |  | 5 |
| F |  |  | 5 |  |  |  | 5 |
| G | 25 |  | 10 |  | 5 | 5 |  |

*Определите длину кратчайшего пути между пунктами A и G (при условии, что передвигаться можно только по построенным дорогам).*

**Решение:**

1. начнём строить возможные маршруты из пункта A; за 1 шаг можно приехать в B, C или сразу в G (в скобках показаны длины маршрутов):

AB(5), AD(12), AG(25)

заметим, что G – это целевая точка (конечный пункт), поэтому мы уже имеем один полный маршрут длиной 25

1. строим двух шаговые маршруты: из B дальше можно ехать в D (возврат в А неинтересен!)

ABD (5 + 8 = 13)

этот маршрут нет смысла продолжать, поскольку в D можно приехать быстрее: длина уже найденного маршрута AD равна 12

1. из D можно ехать в B и C:

ADB (12 + 8 = 20)

ADC (12 + 2 = 14)

1. **третий шаг**: маршрут ADB продолжать бессмысленно: из B можно вернуться только в A и D
2. продолжаем маршрут ADC (14):

ADCE (14 + 4 = 18)

ADCF (14 + 5 = 19)

ADCG (14 + 10 = 24)

в последнем варианте мы приехали в конечный пункт, причем новый маршрут имеет длину 24 < 25, то есть, он короче найденного ранее

1. **четвёртый шаг**: продолжаем маршрут ADCE:

ADCEG (18 + 5 = 23)

и маршрут ADCF:

ADCFG (19 + 5 = 24)

1. других продолжений (без возврата в уже посещённые пункты) нет, поэтому кратчайший маршрут – ADCEG, он имеет длину 23.
2. Ответ: 23.
3. Заметим, что эти рассуждения можно зарисовать в виде дерева возможных маршрутов. После первого шага:

A

B

D

5

12

+5

+12

G

25

+25

После второго шага:

A

B

D

5

12

G

25

B

С

14

20

+5

+12

+25

+2

+8

После третьего шага:

A

B

D

5

12

G

24

B

С

14

20

+5

+12

+25

+2

+8

F

E

G

+4

+5

+10

25

18

19

После четвёртого шага:

A

B

D

5

12

G

24

B

С

14

20

+5

+12

+25

+2

+8

F

E

G

+4

+5

+10

25

18

19

G

G

+5

+5

24

23

### Ещё пример задания:

**Р-03.** *Между населёнными пунктами A, B, C, D, E, F построены дороги, протяжённость которых приведена в таблице. (Отсутствие числа в таблице означает, что прямой дороги между пунктами нет.)*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F |
| A |  | 2 | 4 |  |  |  |
| B | 2 |  | 1 |  | 7 |  |
| C | 4 | 1 |  | 3 | 4 |  |
| D |  |  | 3 |  | 3 |  |
| E |  | 7 | 4 | 3 |  | 2 |
| F |  |  |  |  | 2 |  |

*Определите длину кратчайшего пути между пунктами A и F (при условии, что передвигаться можно только по построенным дорогам).*

**Решение (вариант 1, использование схемы):**

1. построим граф – схему, соответствующую этой весовой матрице; из вершины А можно проехать в вершины B и C (длины путей соответственно 2 и 4):

A

B

C

2

4

1. для остальных вершин можно рассматривать только часть таблицы над главной диагональю, которая выделена серым цветом; все остальные рёбра уже были рассмотрены ранее
2. например, из вершины В можно проехать в вершины C и E (длины путей соответственно 1 и 7):

A

B

C

E

2

4

7

1

1. новые маршруты из С – в D и E (длины путей соответственно 3 и 4):

D

A

B

C

E

2

4

7

1

3

4

1. новый маршрут из D – в E (длина пути 3):

D

A

B

C

E

2

4

7

1

3

4

3

1. новый маршрут из E – в F (длина пути 2):

D

F

A

B

C

E

2

4

7

1

3

4

3

2

1. нужно проехать из А в F, по схеме видим, что в любой из таких маршрутов входит ребро EF длиной 2; таким образом, остается найти оптимальный маршрут из A в E
2. попробуем перечислить возможные маршруты из А в Е:

А – В – Е длина 9

А – В – С – Е длина 7

А – В – C – D – Е длина 9

А –C – Е длина 8

А –C – B – Е длина 12

А –C – D – Е длина 10

1. из перечисленных маршрутов кратчайший – A-B-C-E – имеет длину 7, таким образов общая длина кратчайшего маршрута A-B-C-E-F равна 7 + 2 = 9
2. таким образом, правильный ответ – 9.

**Решение (вариант 2, с начала маршрута):**

1. составим граф, который показывает, куда (и как) можно ехать из пункта А, рядом с дугами будем записывать увеличение пути, а рядом с названиями пунктов – общую длину пути от пункта A:

A

B

C

2

4

+2

+4

1. видно, что напрямую в пункт F из A не доехать
2. строим граф возможных путей дальше: определяем, куда можно ехать из B и C (конечно, не возвращаясь обратно); из B можно ехать только в A (обратно), в C и в E;
3. узел C уже есть на схеме, и оказывается, что короче ехать в него по маршруту A-B-C, чем напрямую A-C, длина «окольного» пути составляет 3 вместо 4 для «прямого»;  
   при движении по дороге B-E длина увеличивается на 7:

A

B

C

2

4 3

+2

+4

+1

E

+7

9

1. строим маршруты из пункта C; кроме A и B, из пункта C можно ехать в D (длина 3) и E (длина 4), причем кратчайший маршрут из A в E оказывается A-B-C-E (длина 7); «невыгодные» маршруты на схеме показывать не будем:

A

B

C

2

3

+2

+1

E

7 9

+4

D

+3

6

1. из пункта D, кроме как в С и E, ехать некуда; путь D-C – это возврат назад (нас не интересует), путь D-E тоже не интересует, поскольку он дает длину 6 + 3 = 9, а мы уже нашли, что в E из A можно доехать по маршруту длины 7
2. из пункта E можно ехать в F, длина полного маршрута 7 + 2 = 9

A

B

C

2

3

+2

+1

E

7

+4

D

+3

6

F

+2

9

1. Ответ: 9

**Решение (вариант 3, с конца маршрута):**

1. можно точно так же начинать с пункта F и искать кратчайший маршрут до A; судя по таблице, из F можно ехать только в E:

E

2

F

+2

1. из E ведут дороги в B, C и D

B

C

9

6

E

2

+4

D

5

F

+2

+7

+3

1. из B можно сразу попасть в A, длина пути будет равна 11:

B

C

9

6

E

2

+4

D

5

F

+2

+7

+3

A

+2

11

1. из пункта C есть прямая дорога в A длиной 4, таким образом, существует маршрут длиной   
   6 + 4 = 10

B

C

9

6

E

2

+4

D

5

F

+2

+7

+3

A

+2

11 10

+4

1. кроме того, есть дорога C-B, которая дает маршрут F-E-C-B-A длиной 9

B

C

7

6

E

2

+4

D

5

F

+2

+3

A

+2

10 9

+4

+1

1. рассмотрение пути C-D не позволяет улучшить результат: оптимальный маршрут имеет длину 9
2. Ответ: 9

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + можно не заметить, что маршруты, проходящие через большее число пунктов, оказываются короче (A-B-C короче, чем A-C, A-B-C-E короче, чем A-B-E) |

### Ещё пример задания:

**Р-02.** *Между четырьмя местными аэропортами: ОКТЯБРЬ, БЕРЕГ, КРАСНЫЙ и СОСНОВО, ежедневно выполняются авиарейсы. Приведён фрагмент расписания перелётов между ними:*

*Аэропорт вылета Аэропорт прилета Время вылета Время прилета*

*СОСНОВО КРАСНЫЙ 06:20 08:35*

*КРАСНЫЙ ОКТЯБРЬ 10:25 12:35*

*ОКТЯБРЬ КРАСНЫЙ 11:45 13:30*

*БЕРЕГ СОСНОВО 12:15 14:25*

*СОСНОВО ОКТЯБРЬ 12:45 16:35*

*КРАСНЫЙ СОСНОВО 13:15 15:40*

*ОКТЯБРЬ СОСНОВО 13:40 17:25*

*ОКТЯБРЬ БЕРЕГ 15:30 17:15*

*СОСНОВО БЕРЕГ 17:35 19:30*

*БЕРЕГ ОКТЯБРЬ 19:40 21:55*

*Путешественник оказался в аэропорту ОКТЯБРЬ в полночь (0:00). Определите самое раннее время, когда он может попасть в аэропорт СОСНОВО.*

1) 15:40 2) 16:35 3)17:15 4) 17:25

**Решение:**

1. сначала заметим, что есть прямой рейс из аэропорта ОКТЯБРЬ в СОСНОВО с прибытием в 17:25:

*ОКТЯБРЬ СОСНОВО 13:40 17:25*

1. посмотрим, сможет ли путешественник оказаться в СОСНОВО раньше этого времени, если полетит через другой аэропорт, с пересадкой
2. можно лететь, через КРАСНЫЙ, но, как следует из расписания,

*ОКТЯБРЬ КРАСНЫЙ 11:45 13:30*

*…*

*КРАСНЫЙ СОСНОВО 13:15 15:40*

путешественник не успеет на рейс КРАСНЫЙ – СОСНОВО, который улетает в 13:15, то есть на 15 минут раньше, чем в КРАСНЫЙ прилетает самолет ОКТЯБРЬ – КРАСНЫЙ

1. можно лететь через БЕРЕГ,

*БЕРЕГ СОСНОВО 12:15 14:25*

*…*

*ОКТЯБРЬ БЕРЕГ 15:30 17:15*

но рейс БЕРЕГ – СОСНОВО вылетает даже раньше, чем рейс ОКТЯБРЬ – БЕРЕГ, то есть, пересадка не получится

1. поскольку даже перелеты с одной пересадкой не стыкуются по времени, проверять варианты с двумя пересадками в данной задаче бессмысленно (хотя в других задачах они теоретически могут дать правильное решение)
2. таким образом, правильный ответ – 4 (прямой рейс).

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + можно не заметить, что путешественник не успеет на пересадку в КРАСНОМ (неверный ответ 15:40)   + можно перепутать аэропорты вылета и прилета (неверный ответ 16:35) |

**Решение (вариант 2, граф):**

1. для решения можно построить граф, показывающий, куда может попасть путешественник из аэропорта ОКТЯБРЬ
2. из аэропорта ОКТЯБРЬ есть три рейса:

*ОКТЯБРЬ СОСНОВО 13:40 17:25*

*ОКТЯБРЬ КРАСНЫЙ 11:45 13:30*

*ОКТЯБРЬ БЕРЕГ 15:30 17:15*

1. построим граф, около каждого пункта запишем время прибытия

17:25

ОКТЯБРЬ

СОСНОВОО

БЕРЕГ

КРАСНЫЙ

13:30

17:15

1. проверим, не будет ли быстрее лететь с пересадкой: рейс «КРАСНЫЙ-СОСНОВО» вылетает в 13:15, то есть, путешественник на него не успевает; он не успеет также и на рейс «БЕРЕГ-СОСНОВО», вылетающий в 12:15
2. таким образом, правильный ответ – 4 (прямой рейс).

### Еще пример задания:

*Грунтовая дорога проходит последовательно через населенные пункты А, B, С и D. При этом длина дороги между А и В равна 80 км, между В и С – 50 км, и между С и D – 10 км. Между А и С построили новое асфальтовое шоссе длиной 40 км. Оцените минимально возможное время движения велосипедиста из пункта А в пункт В, если его скорость по грунтовой дороге – 20 км/час, по шоссе – 40 км/час.*

1) 1 час 2) 1,5 часа 3)3,5 часа 4) 4 часа

**Решение:**

1. нарисуем схему дорог, обозначив данные в виде дроби (расстояние в числителе, скорость движения по дороге – в знаменателе):









1. разделив числитель на знаменатель, получим время движения по каждой дороге









1. ехать из А в B можно
   * напрямую, это займет **4** часа, или …
   * через пункт C, это займет 1 час по шоссе (из А в С) и 2,5 часа по грунтовой дороге   
     (из В в С), всего 1 + 2,5 = **3,5** часа
2. таким образом, правильный ответ – 3.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + можно не заметить, что требуется найти минимальное время поездки именно в В, а не в С (неверный ответ 1 час)   + можно ограничиться рассмотрением только прямого пути из А в В и таким образом получить неверный ответ 4 часа   + можно неправильно нарисовать схему |

### Еще пример задания:

**Р-01.** *Таблица стоимости перевозок устроена следующим образом: числа, стоящие на пересечениях строк и столбцов таблиц, означают стоимость проезда между соответствующими соседними станциями. Если пересечение строки и столбца пусто, то станции не являются соседними. Укажите таблицу, для которой выполняется условие: «Минимальная стоимость проезда из А в B не больше 6». Стоимость проезда по маршруту складывается из стоимостей проезда между соответствующими соседними станциями.*

|  |  |  |  |
| --- | --- | --- | --- |
| 1) | 2) | 3) | 4) |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | A | B | C | D | Е | | A |  |  | 3 | 1 |  | | B |  |  | 4 |  | 2 | | C | 3 | 4 |  |  | 2 | | D | 1 |  |  |  |  | | Е |  | 2 | 2 |  |  | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | A | B | C | D | Е | | A |  |  | 3 | 1 | 1 | | B |  |  | 4 |  |  | | C | 3 | 4 |  |  | 2 | | D | 1 |  |  |  |  | | Е | 1 |  | 2 |  |  | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | A | B | C | D | Е | | A |  |  | 3 | 1 | 4 | | B |  |  | 4 |  | 2 | | C | 3 | 4 |  |  | 2 | | D | 1 |  |  |  |  | | Е | 4 | 2 | 2 |  |  | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | A | B | C | D | Е | | A |  |  |  | 1 |  | | B |  |  | 4 |  | 1 | | C |  | 4 |  | 4 | 2 | | D | 1 |  | 4 |  |  | | Е |  | 1 | 2 |  |  | |

**Решение (вариант 1):**

1. нужно рассматривать все маршруты из А в В, как напрямую, так и через другие станции
2. рассмотрим таблицу 1:

* из верхней строки таблицы следует, что из А в В напрямую везти нельзя, только через C (стоимость перевозки А-С равна 3) или через D (стоимость перевозки из А в D равна 1)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | Е |
| A |  |  | 3 | 1 |  |

* предположим, что мы повезли через C; тогда из третьей строки видим, что из C можно ехать в В, и стоимость равна 4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | Е |
| C | 3 | 4 |  |  | 2 |

* таким образом общая стоимость перевозки из А через С в В равна 3 + 4 = 7
* кроме того, из С можно ехать не сразу в В, а сначала в Е:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | Е |
| C | 3 | 4 |  |  | 2 |

а затем из Е – в В (стоимость также 2),

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | Е |
| Е |  | 2 | 2 |  |  |

так что общая стоимость этого маршрута равна 3  + 2 + 2 = 7

* теперь предположим, что мы поехали из А в D (стоимость 1); из четвертой строки таблицы видим, что из D можно ехать только обратно в А, поэтому этим путем в В никак не попасть:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | Е |
| D | 1 |  |  |  |  |

* таким образом, для первой таблицы минимальная стоимость перевозки между А и В равна 7; заданное условие «*не больше 6»* **не выполняется**

1. аналогично рассмотрим вторую схему; возможные маршруты из А в В:

* , стоимость 7
* , стоимость 7
* таким образом, минимальная стоимость 7, условие **не выполняется**

1. для третьей таблицы:

* , стоимость 7
* , стоимость **6**
* , стоимость 7
* таким образом, минимальная стоимость 6, условие **выполняется**

1. для четвертой:

* , стоимость 9
* , стоимость 8
* минимальная стоимость 8, условие **не выполняется**

1. условие *«не больше 6»* выполняется только для таблицы 3
2. таким образом, правильный ответ – 3.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + метод ненагляден, легко запутаться и пропустить решение с минимальной стоимостью |

**Решение (вариант 2, с рисованием схемы):**

1. для каждой таблицы нарисуем соответствующую ей схему дорог, обозначив стоимость перевозки рядом с линиями, соединяющими соседние станции:

|  |  |  |  |
| --- | --- | --- | --- |
| 1) | 2) | 3) | 4) |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | A | B | C | D | Е | | A |  |  | 3 | 1 |  | | B |  |  | 4 |  | 2 | | C | 3 | 4 |  |  | 2 | | D | 1 |  |  |  |  | | Е |  | 2 | 2 |  |  | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | A | B | C | D | Е | | A |  |  | 3 | 1 | 1 | | B |  |  | 4 |  |  | | C | 3 | 4 |  |  | 2 | | D | 1 |  |  |  |  | | Е | 1 |  | 2 |  |  | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | A | B | C | D | Е | | A |  |  | 3 | 1 | 4 | | B |  |  | 4 |  | 2 | | C | 3 | 4 |  |  | 2 | | D | 1 |  |  |  |  | | Е | 4 | 2 | 2 |  |  | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | A | B | C | D | Е | | A |  |  |  | 1 |  | | B |  |  | 4 |  | 1 | | C |  | 4 |  | 4 | 2 | | D | 1 |  | 4 |  |  | | Е |  | 1 | 2 |  |  | |

1

2

4

2

3

1

2

4

1

3

1

2

4

4

3

2

1

2

4

4

1

1. теперь по схемам определяем кратчайшие маршруты для каждой таблицы:

1:  или , стоимость 7

2:  или , стоимость 7

3: , стоимость **6**

4: , стоимость 8

1. условие *«не больше 6»* выполняется только для таблицы 3
2. таким образом, правильный ответ – 3.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + нужно внимательно строить схемы по таблицам, этот дополнительный переход (от табличных моделей к графическим) повышает наглядность, но добавляет еще одну возможность для ошибки   + наглядность схемы зависит от того, как удачно вы выберете расположение ее узлов; один из подходов – сначала расставить все узлы равномерно на окружности, нарисовать все связи и посмотреть, как можно расположить узлы более удобно   + по невнимательности можно пропустить решение с минимальной стоимостью |

### Еще пример задания[[4]](#footnote-4):

**Р-00.** *Между четырьмя местными аэропортами: ВОСТОРГ, ЗАРЯ, ОЗЕРНЫЙ и ГОРКА, ежедневно выполняются авиарейсы. Приведён фрагмент расписания перелётов между ними:*

*Аэропорт вылета Аэропорт прилета Время вылета Время прилета*

*ВОСТОРГ ГОРКА 16:15 18:30*

*ОЗЕРНЫЙ ЗАРЯ 13:40 15:50*

*ОЗЕРНЫЙ ВОСТОРГ 14:10 16:20*

*ГОРКА ОЗЕРНЫЙ 17:05 19:20*

*ВОСТОРГ ОЗЕРНЫЙ 11:15 13:20*

*ЗАРЯ ОЗЕРНЫЙ 16:20 18:25*

*ВОСТОРГ ЗАРЯ 14:00 16:15*

*ЗАРЯ ГОРКА 16:05 18:15*

*ГОРКА ЗАРЯ 14:10 16:25*

*ОЗЕРНЫЙ ГОРКА 18:35 19:50*

*Путешественник оказался в аэропорту ВОСТОРГ в полночь (0:00). Определите самое раннее время, когда он может попасть в аэропорт ГОРКА.*

1) 16:15 2) 18:15 3)18:30 4) 19:50

**Решение («обратный ход»):**

1. сначала заметим, что есть прямой рейс из аэропорта ВОСТОРГ в ГОРКУ с прибытием в 18:30:

*ВОСТОРГ ГОРКА 16:15 18:30*

1. посмотрим, сможет ли путешественник оказаться в ГОРКЕ раньше этого времени, если полетит через другой аэропорт, с пересадкой; рассмотрим все остальные рейсы, который **прибывают** в аэропорт ГОРКА:

*ЗАРЯ ГОРКА 16:05 18:15*

*ОЗЕРНЫЙ ГОРКА 18:35 19:50*

1. это значит, что имеет смысл проверить только возможность перелета через аэропорт ЗАРЯ (через ОЗЕРНЫЙ явно не получится раньше, чем прямым рейсом); для этого нужно быть в ЗАРЕ не позже, чем в 16:05
2. смотрим, какие рейсы прибывают в аэропорт ЗАРЯ раньше, чем в 16:05:

*ОЗЕРНЫЙ ЗАРЯ 13:40 15:50*

1. дальше проверяем рейсы, который приходят в ОЗЕРНЫЙ раньше, чем в 13:40

*ВОСТОРГ ОЗЕРНЫЙ 11:15 13:20*

1. таким образом, мы «пришли» от конечного пункта к начальному, в обратном направлении
2. поэтому оптимальный маршрут

18:15

ВОСТОРГ

ГОРКА

ОЗЕРНЫЙ

ЗАРЯ

15:50

13:20

1. и правильный ответ – 2.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + «напрашивается» ошибочный ответ 18:30 (прямой рейс)   + при решении задачи «прямым ходом», с начального пункта, легко пропустить вариант с двумя пересадками |

# 6-1 (базовый уровень, время – 4 мин)

**Тема**: Выполнение и анализ простых алгоритмов.

**Что нужно знать**:

* в общем-то, никаких знаний из курса информатики здесь не требуется, эту задачу можно давать детям начальной школы для развития логического мышления
* в некоторых задачах нужно иметь представление о системах счисления (могут использоваться цифры восьмеричной и шестнадцатеричной систем счисления)

### Пример задания:

*Автомат получает на вход четырёхзначное число. По этому числу строится новое число по следующим правилам.*

*1. Складываются первая и вторая, а также третья и четвёртая цифры исходного числа.*

*2. Полученные два числа записываются друг за другом в порядке убывания (без разделителей).*

*Пример. Исходное число: 3165. Суммы: 3 + 1 = 4; 6 + 5 = 11. Результат: 114.*

*Укажите наименьшее число, в результате обработки которого, автомат выдаст число 1311.*

**Решение:**

1. единственный способ разбить запись 1311 на два числа – это 13 и 11 (числа 131 и 311 не могут образоваться в результате сложения значений двух десятичных цифр)
2. сумма первой и второй цифр должна быть наименьшей (тогда и число будет меньше!), она равна 11; тогда сумма значений двух последних цифр равна 13
3. для того, чтобы всё число было минимально, числа, составленные из первых двух и последних двух цифр должны быть минимальными соответственно для сумм 11 и 13
4. минимальное двузначное число, у которого сумма значений цифр равна 11, - это 29, с этих двух цифр начинается исходное четырёхзначное число
5. сумма двух последних цифр – 13, минимальное двузначное число с такой суммой цифр – 49.
6. Ответ: 2949.

### Ещё пример задания:

*В некоторой информационной системе информация кодируется двоичными шестиразрядными словами. При передаче данных возможны их искажения, поэтому в конец каждого слова добавляется седьмой (контрольный) разряд таким образом, чтобы сумма разрядов нового слова, считая контрольный, была чётной. Например, к слову 110011 справа будет добавлен 0, а к слову 101100 – 1.*

*После приёма слова производится его обработка. При этом проверяется сумма его разрядов, включая контрольный. Если она нечётна, это означает, что при передаче этого слова произошёл сбой, и оно автоматически заменяется на зарезервированное слово 0000000. Если она чётна, это означает, что сбоя не было или сбоев было больше одного. В этом случае*

*принятое слово не изменяется.*

*Исходное сообщение*

1100101 1001011 0011000

*было принято в виде*

1100111 1001110 0011000.

*Как будет выглядеть принятое сообщение после обработки?*

1) 1100111 1001011 0011000

2) 1100111 1001110 0000000

3) 0000000 0000000 0011000

4) 0000000 1001110 0011000

**Решение:**

1. по условию в правильно принятом блоке число единиц должно быть чётное
2. в принятом сообщении 1100111 1001110 0011000 нечётное число единиц (5) только в первом блоке, поэтому он будет заменён на нули
3. ответ: 4.

|  |
| --- |
| **Возможные ловушки проблемы**:   * + не нужно сравнивать полученное сообщение с исходным; если при передаче блока произошло чётное число ошибок, то приёмник не сможет обнаружить ошибку и будет считать этот блок правильным |

### Ещё пример задания:

*Учитель предлагает детям три цифры. Ученики должны сначала найти сумму*

*первой и второй цифр, потом – сумму второй и третьей цифр. Затем полученные числа записываются друг за другом в порядке невозрастания (правое число меньше или равно левому).*

*Пример. Исходные цифры: 6, 3, 9. Суммы: 6 + 3 = 9; 3 + 9 = 12. Результат: 129.*

*Укажите, какая из следующих последовательностей символов может быть получена в результате.*

1) 1915 2) 1815 3) 188 4) 1518

**Решение:**

1. число записано в десятичной системе счисления, поэтому все цифры меньше или равны 9, так что при сложении двух таких чисел может получиться сумма от 0 до 18
2. в первом варианте ответа 4 цифры, это два двузначных числа, записанные подряд; заметим, что первое число – 19, такая сумма не могла получиться, поэтому это неправильный ответ
3. в ответе 4 тоже две суммы, 15 и 18, но они стоят в порядке возрастания, поэтому это тоже неверный ответ
4. в ответах 2 и 3 два числа стоят в порядке убывания (18 и 15 в ответе 2, 18 и 8 в ответе 3), это соответствует условию
5. чтобы выбрать между ответами 2 и 3, нужно вспомнить, что вторая цифра по условию входит в обе суммы
6. заметим, что если сумма равна 18, то обе цифры (в том числе вторая) равны 9, поэтому другая сумма не может получиться меньше 9; это означает, что ответ 3 (188) неверный
7. Ответ: 2.

### Ещё пример задания:

*Автомат получает на вход два двузначных шестнадцатеричных числа. В этих*

*числах все цифры не превосходят цифру 6 (если в числе есть цифра больше 6, автомат отказывается работать). По этим числам строится новое*

*шестнадцатеричное число по следующим правилам.*

1. *Вычисляются два шестнадцатеричных числа – сумма старших разрядов полученных чисел и сумма младших разрядов этих чисел.*
2. *Полученные два шестнадцатеричных числа записываются друг за другом в порядке возрастания (без разделителей).*

***Пример.*** *Исходные числа: 66, 43. Поразрядные суммы: A, 9. Результат: 9A.*

*Определите, какое из следующих чисел может быть результатом работы автомата.*

1) 9F 2) 911 3) 42 4) 7A

**Решение:**

1. по условию обе цифры числа меньше или равны 6, поэтому при сложении двух таких чисел может получиться сумма от 0 до 12 = C­16
2. из п. 1 сразу делаем вывод, что цифры F в записи числа быть не может, вариант 1 не подходит
3. каждая из двух сумм находится в интервале 0..12, поэтому записывается одной шестнадцатеричной цифрой, так что результат работы автомата всегда состоит ровно из двух цифр
4. из п. 2 следует, что вариант 2, состоящий из трех цифр, не подходит
5. по условию цифры записаны в порядке возрастания, поэтому вариант 3 не подходит
6. остается вариант 4, в котором все условия соблюдаются
7. Ответ: 4.

### Ещё пример задания:

*Автомат получает на вход два трехзначных числа. По этим числам строится новое число по следующим правилам. Вычисляются три числа – сумма старших разрядов заданных трехзначных чисел, сумма средних разрядов этих чисел, сумма младших разрядов.*

*Полученные три числа записываются друг за другом в порядке убывания (без разделителей).*

***Пример.***

*Исходные трехзначные числа: 835, 196. Поразрядные суммы: 9, 12, 11. Результат: 12119*

*Определите, какое из следующих чисел может быть результатом работы автомата.*

1) 151303 2) 161410 3) 191615 4) 121613

**Решение:**

1. итак, число строится из трех чисел, каждое из которых может быть однозначным (от 0 до 9) или двузначным (от 10 до 9 + 9 = 18)
2. если в числе 6 цифр, значит соединены три двузначных числа; в первом числе одно из них записывается как «03», что недопустимо (в этом случае правильное число было бы записано как 15133)
3. в третьем числе тоже 6 цифр: три двузначных числа, первое из которых равно 19, чего не может быть (никакие два однозначных числа не могут дать такую сумму)
4. в четвертом числе тоже 6 цифр: три числа 12, 16 и 13 расположены НЕ в порядке убывания, поэтому этот вариант неверен
5. во втором варианте никаких противоречий с условием нет
6. таким образом, ответ: 2.

### Еще пример задания:

*Предлагается некоторая операция над двумя произвольными трехзначными десятичными числами:*

1. *Записывается результат сложения старших разрядов этих чисел.*
2. *К нему дописывается результат сложения средних разрядов по такому правилу: если он меньше первой суммы, то полученное число приписывается к первому слева, иначе – справа.*
3. *Итоговое число получают приписыванием справа к числу, полученному после второго шага, сумму значений младших разрядов исходных чисел.*

*Какое из перечисленных чисел могло быть построено по этому правилу?*

1) 141819 2) 171418 3) 141802 4) 171814

**Решение:**

1. заметим, что сумма двух однозначных чисел – это число от 0 до 18 включительно
2. все предложенные числа шестизначные, поэтому все суммы, из которых составлены числа, должны быть двузначными

1) 141819 2) 171418 3) 141802 4) 171814

1. поскольку числа 19 быть не может (его не получить суммой двух однозначных чисел), этот вариант не подходит
2. из условия (2) следует, что первые два двузначных числа должны быть расположены по возрастанию (неубыванию), поэтому вариант 2 не подходит
3. при записи числа 2 ноль впереди не добавляется (в условии про это ничего не сказано), поэтому третий вариант тоже не подходит
4. вариант 4 удовлетворяет всем условиям.
5. таким образом, ответ: 4.

### Ещё пример задания:

*Цепочка из трех бусин, помеченных латинскими буквами, формируется по следующему правилу. В конце цепочки стоит одна из бусин A, B, C. На первом месте – одна из бусин B, D, C, которой нет на третьем месте. В середине – одна из бусин А, C, E, B, не стоящая на первом месте. Какая из перечисленных цепочек создана по этому правилу?*

1) CBB 2) EAC 3)BCD 4) BCB

**Решение (краткий вариант):**

1. проверяем первое условие: «*В конце цепочки стоит одна из бусин A, B, C».* Ему не удовлетворяет цепочка BCD, ее можно вычеркнуть:

1) CBB 2) EAC ~~3)BCD~~ 4) BCB

1. проверяем второе условие: «*На первом месте – одна из бусин B, D, C, которой нет на третьем месте*». Ему не удовлетворяют цепочки EAC (на первом месте – E) и BCB (на первом и третьем местах стоит буква B), поэтому остается только вариант CBB:

1) CBB ~~2) EAC~~ ~~4) BCB~~

1. проверяем третье условие: «*В середине – одна из бусин А, C, E, B, не стоящая на первом месте*». К счастью, оставшаяся цепочка CBB ему удовлетворяет.
2. таким образом, правильный ответ – 1.

|  |
| --- |
| **Возможные проблемы**:   * + не все могут сделать подобный анализ в уме |

**Решение (подробный вариант):**

1. правило содержит три условия, обозначим их так:

**У1**: третья бусина – A, B или C

**У2-3**: первая бусина – B, D или C, не совпадающая с третьей

**У4-5**: вторая бусина – A, B, C или E, не совпадающая с первой

1. фактически условия У2-3 и У4-5 сложные, их можно разбить на два, так что получится всего пять условий

**У1**: третья бусина – A, B или C

**У2**: первая бусина – B, D или C

**У3**: первая и третья бусины – разные

**У4**: вторая бусина – A, B, C или E

**У5**: первая и вторая бусины – разные

1. теперь для каждого из ответов проверим выполнение всех условий; в таблице красный крестик обозначает, что условие не выполняется для данного варианта; зеленым цветом выделена строка, где нет ни одного крестика, то есть все условия выполняются:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | У1 | У2 | У3 | У4 | У5 |
| 1) CBB |  | **×** |  |  |  |
| 2) EAC | **×** |  |  |  |  |
| 3) BCD |  |  | **×** |  |  |
| 4) BCB |  |  |  |  |  |

1. таким образом, правильный ответ – 1.

# 6-2 (базовый уровень, время – 4 мин)

**Тема**: Поиск алгоритма минимальной длины для исполнителя.

**Что нужно знать**:

* каких-либо особых знаний из курса информатики не требуется, задача решаема на уровне 6-7 класса простым перебором вариантов, просто его нужно организовать оптимальным образом
* *исполнитель* – это человек, группа людей, животное, машина или другой объект, который может понимать и выполнять некоторые команды

### Пример задания:

*У исполнителя Аккорд две команды, которым присвоены номера:*

**1. отними 1**

**2. умножь на x**

*где x – неизвестное положительное число. Выполняя первую из них, Аккорд отнимает от числа на экране 1, а выполняя вторую, умножает это число на x.*

*Программа для исполнителя Аккорд – это последовательность номеров команд.*

*Известно, что программа 12121 переводит число 4 в число 23. Определите значение x.*

**Решение (составление уравнения)**:

1. проблема здесь в том, что мы не знаем значения *x*, поэтому выполним программу, используя *x* как переменную:

Вход: 4

1: 4 – 1 = 3

2: 3·*x* = 3*x*

1: 3·*x* – 1

2: (3·*x* – 1) ·*x* = 3*x*2– *x*

1: 3*x*2– *x –* 1 = 23

1. остаётся решить уравнение  или 
2. это уравнение имеет 2 корня, *x*1*=* 3 и *x*2*=* – 2,666
3. нас интересует только целое положительное решение, поэтому ответ – 3
4. Ответ: 3.

**Решение (метод перебора)**:

1. можно использовать метод подбора, учитывая, что нас интересует только натуральное число, большее, чем 1
2. пусть *x* = 2, тогда при выполнении программы 12121 для числа 4 получаем

4 → 3 → 6 → 5 → 10 → 9

что не совпадает с заданным значением 23

1. берём следующее значение, пусть *x* = 3, тогда при выполнении программы 12121 для числа 4 получаем

4 → 3 → 9 → 8 → 24 → 23

что совпадает с заданным результатом.

1. Ответ: 3.

### Ещё пример задания:

*У исполнителя Удвоитель две команды, которым присвоены номера:*

**1. прибавь 1**

**2. умножь на 2**

*Выполняя первую из них, Удвоитель прибавляет к числу на экране 1, а выполняя вторую, умножает его на 2. Запишите порядок команд в программе получения из числа 3 числа 63, содержащей не более 8 команд, указывая лишь номера команд.*

**Решение («обратный ход»)**:

1. такие задачи проще решать, если переформулировать их для обратного исполнителя, которого можно назвать Раздвоителем; его команды
2. вычти 1
3. раздели на 2 (только для чётных чисел)
4. получим с помощью Раздвоителя число 3 из 63 (идём в обратную сторону)
5. будем использовать следующий (в данном случае – оптимальный) алгоритм: если число нечётное, вычитаем единицу (команда 1), потому что делить его на 2 нельзя; если число чётное, делим его на два; сверху записаны номера выполняемых команд:

1 2 1 2 1 2 1 2

63 → 62 → 31 → 30 → 15 → 14 → 7 → 6 → 3

таким образом, выполняя программу 12121212, Раздвоитель получает число 3 из 63

1. программу для Удвоителя (выполняющего обратную цепочку действий) запишем в обратном порядке: 21212121
2. Ответ: 21212121

### Ещё пример задания:

*У исполнителя Калькулятор две команды, которым присвоены номера:*

**1. прибавь 3**

**2. умножь на 4**

*Выполняя первую из них, Калькулятор прибавляет к числу на экране 3, а выполняя вторую, умножает его на 4. Запишите порядок команд в программе получения из числа 3 числа 57, содержащей не более 6 команд, указывая лишь номера команд.*

*(Например, программа 21211 это программа*

**умножь на 4**

**прибавь 3**

**умножь на 4**

**прибавь 3**

**прибавь 3**

*которая преобразует число 2 в 50.)*

**Решение (вариант 1, «прямой ход»)**:

1. обратим внимание, что в условии ограничено число команд, поэтому неявно ставится задача написать самую короткую программу для решения задачи
2. начнем решать задачу, «отталкиваясь» от начального числа
3. на первом шаге с помощью имеющихся команд из числа 3 можно получить 6 или 12;
4. на втором шаге из 6 можно получить 9 и 24, а из 12 – 15 и 48, и т.д., получается такая схема (структура «дерево»), цифры около стрелок показывает номер выполненной команды:
5. уже чувствуется, что дерево сильно разрастается, на следующем уровне будет уже 8 вариантов, потом – 16 и т.д. (на каждом следующем уровне – в 2 раза большем, чем на предыдущем)
6. нужно выбрать такой план дальнейшего перебора вариантов, который может быстрее всего привести к цели (числу 57)
7. видим, что после второй операции ближе всего к результату оказалось число 48, попробуем начать анализ с этой ветки; если не получится – возьмем число 24 и т.д.
8. ветка дерева, начиная от числа 48, построена на рисунке справа; красный крестик показывает, что полученное значение превышает 57
9. итак, мы вышли на число 57 в результате такой последовательности команд: 22111, ее длина равна 5, что удовлетворяет условию задачи.
10. таким образом, правильный ответ – 22111.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * большую схему неудобно рисовать, в ней легко запутаться * не всегда можно сразу угадать нужную ветку «дерева», то есть, ту, которая быстрее всего приведет к успеху |

**Решение (вариант 2, «обратный ход»)**:

1. нам нужно увеличить число (с 3 до 57), для этого в большинстве случаев умножение эффективнее сложения, поэтому нужно постараться максимально использовать умножение, а сложение – только в крайних случаях
2. попробуем решить задачу «обратным ходом», начав с числа 57;
3. очевидно, что последней командой не может быть умножение на 4 (57 на 4 не делится), поэтому последняя команда – сложение (**прибавь 3**), над стрелкой записан номер команды:



1. число 54 также не делится на 4, поэтому предыдущая команда – тоже сложение:



1. аналогично для числа 51:



1. число 48 делится на 4, поэтому используем умножение:



1. наконец, добавив в начало программы еще одно умножение, получаем полную цепочку:



1. таким образом, правильный ответ – 22111, эта программа состоит из 5 команд.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * иногда может потребоваться «откат» назад, например, если исходное число – 6, то применив деление на 4 для 12 мы «проскакиваем» его (получаем **12/4=3<6**), поэтому нужно возвращаться обратно к 12 и дважды применять сложение; в этом случае ответ будет такой: |

|  |
| --- |
| **Почему здесь «обратный ход» лучше?**:   * обратим внимание, что когда мы «шли» в обратном направлении, от конечного числа к начальному, часто очередную операцию удавалось определить однозначно (когда число не делилось на 4) * это связано с тем, что среди допустимых команд есть «не всегда обратимая» операция – умножение: умножить целое число на 4 можно всегда, а разделить нацело – нет; в подобных случаях результат быстрее получается именно «обратным ходом», во время которого сразу отбрасываются невозможные варианты |

### Еще пример задания:

*У исполнителя, который работает с положительными однобайтовыми двоичными числами, две команды, которым присвоены номера:*

**1. сдвинь влево**

**2. вычти 1**

*Выполняя первую из них, исполнитель сдвигает число на один двоичный разряд влево, а выполняя вторую, вычитает из него 1. Исполнитель начал вычисления с числа 104 и выполнил цепочку команд 11221. Запишите результат в десятичной системе.*

**Решение**:

1. важно, что числа однобайтовые – на число отводится 1 байт или 8 бит
2. главная проблема в этой задаче – разобраться, что такое «сдвиг влево»; так называется операция, при которой все биты числа в ячейке (регистре) сдвигаются на 1 бит влево, в младший бит записывается нуль, а старший бит попадает в специальную ячейку – *бит переноса*:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |
| ? |  | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | = 45 |
|  |  |  |  |  |  |  |  |  |  | 0 |
| 0 |  | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | = 90 |

бит   
 переноса

можно доказать, что в большинстве случаев результат этой операции – умножение числа на 2, однако есть исключение: если в старшем (7-ом) бите исходного числа *x* была 1, она будет «выдавлена» в бит переноса, то есть потеряна[[5]](#footnote-5), поэтому мы получим остаток от деления числа *2x* на 28=256

1. попутно заметим, что при сдвиге вправо[[6]](#footnote-6) в старший бит записывается 0, а младший «уходит» в бит переноса; это равносильно делению на 2 и отбрасыванию остатка
2. таким образом, фактически команда **сдвинь влево** означает **умножь на 2**
3. поэтому последовательность команд 11221 выполняется следующим образом

|  |  |  |  |
| --- | --- | --- | --- |
| **Код команды** | **Действие** | **Результат** | **Примечание** |
|  |  | 104 |  |
| 1 | **умножь на 2** | 208 |  |
| 1 | **умножь на 2** | 160 | остаток от деления 208\*2 на 256 |
| 2 | **вычти 1** | 159 |  |
| 2 | **вычти 1** | 158 |  |
| 1 | **умножь на 2** | 60 | остаток от деления 158\*2 на 256 |

1. правильный ответ – 60.

### Еще пример задания:

*Исполнитель Робот действует на клетчатой доске, между соседними клетками которой могут стоять стены. Робот передвигается по клеткам доски и может выполнять команды 1 (вверх), 2 (вниз), 3 (вправо) и 4 (влево), переходя на соседнюю клетку в направлении, указанном в скобках. Если в этом направлении между клетками стоит стена, то Робот разрушается. Робот успешно выполнил программу*

**3233241**

*Какую последовательность из трех команд должен выполнить Робот, чтобы вернуться в ту клетку, где он был перед началом выполнения программы, и не разрушиться вне зависимости от того, какие стены стоят на поле?*

**Решение**:

1. фактически заданная программа движения Робота, которую он успешно выполнил, показывает нам свободный путь, на котором стенок нет
2. поэтому для того, чтобы не разрушиться на обратном пути, Робот должен идти точно по тому же пути в обратном направлении
3. нарисуем путь Робота, который выполнил программу 3233241:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ? | ? | ? | ? | ? | ? |
| ? |  |  | ? | ? | ? |
| ? | ? |  |  |  | ? |
| ? | ? | ? |  |  | ? |
| ? | ? | ? | ? | ? | ? |

Робот начал движение из клетки, отмеченной красной точкой, и закончил в клетке, где стоит синяя точка

1. чтобы вернуться в исходную клетку (с красной точкой) по пройденному пути, Роботу нужно сделать шаг влево (команда 4), затем шаг вверх (команда 1) и еще один шаг влево (команда 4)
2. таким образом, ответ – 414.

### Еще пример задания:

*Исполнитель Робот ходит по клеткам бесконечной вертикальной клетчатой доски, переходя по одной из команд вверх, вниз, вправо, влево в соседнюю клетку в указанном направлении. Робот выполнил следующую программу:*

**вправо**

**вверх**

**влево**

**влево**

**вниз**

**вниз**

**вправо**

**вправо**

**вправо**

**вниз**

**влево**

*Укажите наименьшее возможное число команд в программе, переводящей Робота из той же начальной клетки в ту же конечную.*

**Решение (способ 1, моделирование движения Робота)**:

1. отметим, что в условии ничего не говорится о стенках, то есть, молчаливо предполагаем, что их нет
2. можно повторить все движения Робота на бумажке и посмотреть, куда он уйдет; на схеме исходная точка обозначена красной точкой, а конечная – синей, синяя линия показывает путь Робота:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

1. поскольку Робот не может ходить по диагонали, для перехода из начальной точки в конечную кратчайшим путем ему нужно выполнить, например, такую программу (см. штриховые линии на рисунке):

|  |
| --- |
| **вниз**  **вниз**  **вправо** |

1. есть и другие варианты (попробуйте их найти!), но все они содержат 3 команды: одну команду **вправо** и две команды **вниз**
2. таким образом, ответ – 3.

**Решение (способ 2, анализ программы)**:

1. можно решить задачу без повторения движений Робота
2. обратим внимание, что пары команд «вперед-назад» и «влево-вправо» дают нулевой эффект, то есть, не перемещают Робота, поэтому все такие пары можно выкинуть из программы
3. поскольку стенок нет, все равно где стоят парные команды в программе, вычеркиваем их:

**вправо**

**вверх**

**влево**

**влево**

**вниз**

**вниз**

**вправо**

**вправо**

**вправо**

**вниз**

**влево**

1. смотрим, какие команды остались (они отмечены желтым маркером), их всего 3
2. таким образом, ответ – 3.

### Еще пример задания:

*Исполнитель КУЗНЕЧИК живёт на числовой оси. Начальное положение КУЗНЕЧИКА – точка 0. Система команд Кузнечика:*

**Вперед 4** *– Кузнечик прыгает вперед на 4 единицы,*

**Назад 3** *– Кузнечик прыгает назад на 3 единицы.*

*Какое наименьшее количество раз должна встретиться в программе команда «Назад 3», чтобы Кузнечик оказался в точке 27?*

**Решение (составление уравнения, подбор решения)**:

1. обозначим через  количество команд «Вперед 4» в программе, а через  – количество команд «Назад 3»
2. для того, чтобы КУЗНЕЧИК попал в точку 27 из точки 0, должно выполняться условие



1. это уравнение называется *диофантовым*; поскольку числа 4 и 3 – взамнопростые (их наибольший общий делитель равен 1), оно имеет бесконечно много решений
2. из всех решений нас интересует такое, при котором  – наименьшее возможное неотрицательное (!) число
3. представим уравнение в виде



нужно подобрать минимальное неотрицательное , при котором правая часть делится на 4

1. дальше используем метод подбора (или перебора), начиная от 1; получаем



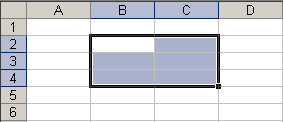
1. видим, что первое , при котором  делится на 4, это  (при этом ).
2. таким образом, ответ – 3.

# 7-1 (базовый уровень, время – 3 мин)

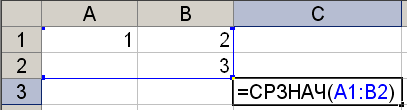
**Тема**: Электронные таблицы.

**Что нужно знать**:

* адрес ячейки в электронных таблицах состоит из имени столбца и следующего за ним номера строки, например, C15
* формулы в электронных таблицах начинаются знаком = («равно»)
* знаки +, –, \*, / и ^ в формулах означают соответственно сложение, вычитание, умножение, деление и возведение в степень
* запись B2:C4 означает диапазон, то есть, все ячейки внутри прямоугольника, ограниченного ячейками B2 и C4:

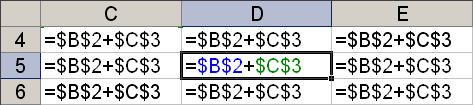


* например, по формуле =СУММ(B2:C4) вычисляется сумма значений ячеек B2, B3, B4, C2, C3 и C4
* в заданиях ЕГЭ могут использоваться стандартные функции СЧЕТ (количество непустых ячеек), СУММ (сумма), СРЗНАЧ (среднее значение), МИН (минимальное значение), МАКС (максимальное значение)
* функция СРЗНАЧ при вычислении среднего арифметического не учитывает пустые ячейки и ячейки, заполненные текстом; например, после ввода формулы в C2 появится значение 2 (ячейка А2 – пустая):



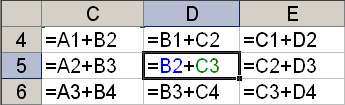
функция СЧЕТ(A1:B2) в этом случае выдаст значение 3 (а не 4).

* адреса ячеек (или ссылки на ячейки) бывают относительные, абсолютные и смешанные, вся разница между ними проявляется при копировании формулы в другую ячейку:
  + в *абсолютных* адресах перед именем столбца и перед номером строки ставится знак доллара $, такие адреса не изменяются при копировании; вот что будет, если формулу **=$B$2+$C$3** скопировать из D5 во все соседние ячейки

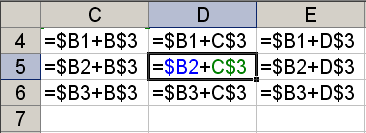


знак $ как бы «фиксирует» значение: в абсолютных адресах и имя столбца, и номер строки зафиксированы

* + в *относительных* адресах знаков доллара нет, такие адреса при копировании изменяются: номер столбца (строки) изменяется на столько, на сколько отличается номер столбца (строки), где оказалась скопированная формула, от номера столбца (строки) исходной ячейки; вот что будет, если формулу **=B2+C3** (в ней оба адреса – относительные) скопировать из D5 во все соседние ячейки:



* + в *смешанных* адресах часть адреса (строка или столбец) – абсолютная, она «зафиксирована» знаком $, а вторая часть – относительная; относительная часть изменится при копировании так же, как и для относительной ссылки:



### Пример задания:

Дан фрагмент электронной таблицы. Из ячейки D2 в одну из ячеек диапазона E1:E4 была скопирована формула. При копировании адреса ячеек в формуле автоматически изменились, и значение формулы стало равным 8. В какую ячейку была скопирована формула? В ответе укажите только одно число – номер строки, в которой расположена ячейка.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | А | В | С | D | Е |
| 1 | 1 | 2 | 3 | 4 |  |
| 2 | 2 | 3 | 4 | =B$3+$C2 |  |
| 3 | 3 | 4 | 5 | 6 |  |
| 4 | 4 | 5 | 6 | 7 |  |

**Решение:**

1. поскольку в формуле, которая записана в ячейку D2, две смешанных ссылки, в первой заблокирована строка 3, а во второй – столбец C
2. формула перемещается в столбец E (на 1 столбец вправо), поэтому в первой ссылке адрес столбца будет C, так что формула примет вид:

=C$3 + $C?

где вместо знака вопроса будет некоторый номер строки – той строки, в которую скопируют формулу

1. значение ячейки C3 равно 5, для того, чтобы получить в сумме 8, нужно добавить к нему число 3 – в столбце С оно находится в ячейке C1; поэтому формулу нужно скопировать в первую строку (в ячейку E1).
2. ответ: 1.

### Ещё пример задания:

Коле нужно с помощью электронных таблиц построить таблицу квадратов двузначных чисел от 20 до 59. Для этого сначала в диапазоне В1:К1 он записал числа от 0 до 9, и в диапазоне А2:А5 он записал числа от 2 до 5. Затем в ячейку В5 записал формулу квадрата двузначного числа (А5 – число десятков; В1 – число единиц), после чего скопировал её во все ячейки диапазона B2:К5. В итоге получил таблицу квадратов двузначных чисел. На рисунке ниже представлен фрагмент этой таблицы.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | А | В | С | D | Е |
| 1 |  | 0 | 1 | 2 | 3 |
| 2 | 2 | 400 | 441 | 484 | 529 |
| 3 | 3 | 900 | 961 | 1024 | 1089 |
| 4 | 4 | 1600 | 1681 | 1764 | 1849 |
| 5 | 5 | 2500 | 1601 | 2704 | 2809 |

Какая формула была записана в ячейку В5?

1) =(B1+10\*А5)^2 2) =($B1+10\*$А5)^2 3) =(B$1+10\*$А5)^2 4) =($B1+10\*А$5)^2

**Решение:**

1. посмотрим, куда ссылаются правильные формулы в B5 и в какой-нибудь другой ячейке, которая отличается от B5 и строкой, и столбцом, например, в D3:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | А | В | С | D | Е |
| 1 |  | 0 | 1 | 2 | 3 |
| 2 | 2 |  |  |  |  |
| 3 | 3 |  |  | =(D1+10\*A3)^2 |  |
| 4 | 4 |  |  |  |  |
| 5 | 5 | =(B1+10\*A5)^2 |  |  |  |

1. смотрим, что в тих формулах меняется, а что не меняется; видим, что в первой ссылке не меняется строка 1, а во второй – столбец А, их и нужно сделать абсолютными, заблокировать с помощью знака $
2. поэтому в B5 нужно ввести формулу **=(B$1+10\*$A5)^2**
3. Ответ: 3.

**Решение** (частный случай, *А.Н. Носкин*)**:**

1. проанализируем предлагаемые ответы на наличие «конфликта» при копировании в другие ячейки.
2. в трёх вариантах ответа, а именно в 1, 2 и 4 есть ссылки B1 или $B1, в которых не заблокирована первая строка; это значит, что при копировании такой формулы «вверх» номер строки станет нулевым или отрицательным, а нулевых или отрицательных строк (столбцов) в Excel не существует.
3. поэтому в B5 нужно ввести формулу **=(B$1+10\*$A5)^2**
4. Ответ: 3.

### Ещё пример задания:

Нужно с помощью электронных таблиц построить таблицу значений формулы 2*х*+3*у* для значений *х* и *у* от 4 до 7. Для этого сначала в диапазонах В1:Е1 и А2:А5 записали числа от 4 до 7. Затем в ячейку В5 записали формулу (А5 - значение *х*, В1 – значение *y*), после чего эта формула была скопирована во все ячейки диапазона В2:Е5. В итоге на экране получился фрагмент таблицы.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | А | В | С | D | Е |
| 1 |  | 4 | 5 | 6 | 7 |
| 2 | 4 | 20 | 23 | 26 | 29 |
| 3 | 5 | 22 | 25 | 28 | 31 |
| 4 | 6 | 24 | 27 | 30 | 33 |
| 5 | 7 | 26 | 29 | 32 | 35 |

Какая формула была записана в ячейку В5?

1) =$А5\*2+В$1\*3 2) =А5\*2+В1\*3 3) =$А5\*2+$В1\*3 4) =А$5\*2+$В1\*3

**Решение:**

1. посмотрим, куда ссылаются правильные формулы в B5 и в какой-нибудь другой ячейке, которая отличается от B5 и строкой, и столбцом, например, в D3:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | А | В | С | D | Е |
| 1 |  | 4 | 5 | 6 | 7 |
| 2 | 4 |  |  |  |  |
| 3 | 5 |  |  | =A3\*2+D1\*3 |  |
| 4 | 6 |  |  |  |  |
| 5 | 7 | =A5\*2+B1\*3 |  |  |  |

1. смотрим, что в тих формулах меняется, а что не меняется; видим, что в первой ссылке не меняется столбец А, а во второй – строка 1, их и нужно сделать абсолютными, заблокировать с помощью знака $
2. поэтому в B5 нужно ввести формулу **=$A5\*2+B$1\*3**
3. Ответ: 1.

**Решение (частный случай, А.Н. Носкин):**

1. проанализируем предлагаемые формулы

1) =$А5\*2+В$1\*3 2) =А5\*2+В1\*3 3) =$А5\*2+$В1\*3 4) =А$5\*2+$В1\*3

1. формулы 2, 3 и 4 содержат ссылки на B1, в которых номер строки 1 не закреплён абсолютной ссылкой, то есть будет изменяться при копировании
2. поэтому при копировании формул 2, 3 и 4 из В5 вверх (в строку с меньшим номером) номер строки должен получиться меньше 1, что приведет к ошибочной ссылке
3. следовательно, варианты 2, 3 и 4 не подходят
4. Ответ: 1.

### Ещё пример задания:

В ячейке E15 электронной таблицы записана формула. Эту формулу скопировали в ячейки D17 и C18. В соответствии с формулой, полученной в ячейке D17, значение в этой ячейке равно разности значений в ячейках D32 и C32; в соответствии с формулой, полученной в ячейке C18, значение в этой ячейке равно разности значений в ячейках D33 и B32. Укажите, какая формула могла быть написана в ячейке E15.

1) =E$32-D$30 2) =$D$32-$B$32 3) =$D$30-$C$32 4) =$D30-D$32

**Решение:**

1. одну и ту же формулу скопировали в две ячейки и получили:

в E15 → ?

? → в D17 → D32 – C32

? → в C18 → D33 – B32

1. видим, что обе целевые ячейки, D17 и C18, относятся к разным столбцам и разным строкам, в то же время в обеих формулах в первой ссылке – столбец D, а во второй – строка 32
2. следовательно, эти части ссылок абсолютные, они заблокированы от изменений знаком $
3. номера строк в первой ссылке и имена столбцов во второй – разные, они относительные
4. таким образом, получаем формулу =$D30 – D$32
5. Ответ: 4.

### Ещё пример задания:

В ячейке X15 электронной таблицы записана формула. Эту формулу скопи­ровали в ячейку Z13. В соответствии с формулой, полученной в ячейке Z13, значение в этой ячейке равно произведению значений в ячейках D20 и E25. Напишите, сколько из следующих четырёх утверждений не противоречат этим данным.

1. Значение в ячейке X15 равно *х\*у*, где *х* - значение в ячейке D20, а *у* - значение в ячейке C27.

2. Значение в ячейке X15 равно *х\*у*, где *х* - значение в ячейке B20, а *у* - значение в ячейке E25.

3. Значение в ячейке X15 вычисляется по формуле *х\*у*, где *х* - значение в ячейке D22, а *у* - значение в ячейке C25.

4. Значение в ячейке X15 равно *х2*, где *х* - значение в ячейке E27.

1) 1 2) 2 3) 3 4) 4

**Решение:**

1. по условию мы знаем, что в ячейке Z13 записана формула =D20\*E25, в которой каждая ссылка может быть абсолютной, относительной и смешанной, то есть возможны, например, такие варианты =$D$20\*$E$25, =D$20\*$E25 и т.д.
2. для того, чтобы определить, какая формула была в X15, нужно скопировать формулу из Z13 в X15, поменяв соответствующим образом ссылки, тип которых мы не знаем
3. начнём с варианта с относительными ссылками: при копировании формулы из Z13 в X15 номер столбца уменьшается на 2 (Z→X), а номер строки – увеличивается на 2 (13→15), поэтому формула с относительными ссылками изменится так:

Z13: =D20\*E25 → X15: =B22\*C27

1. кроме того, каждая часть ссылки может быть защищена от изменений знаком $; например, для первой ссылки получаем такие варианты преобразования

D20 → B22, $D20 → $D22, D$20 → B$20, $D$20 → $D$20,

то есть первая ссылка может превратиться в B20, B22, D20 и D22

1. аналогично вторая ссылка (E25) при копировании может превратиться в C25, C27, E25 и E27
2. при проверке утверждений 1, 2 и 3 выясняется, что все адреса ячеек допустимые, то есть входят в перечисленные в пп. 4 и 5, поэтому эти утверждения не противоречат исходным данным.
3. в утверждении 4 обе ссылки должны стать равны E27, это возможно для второй ссылки, но не для первой (см. п. 4), поэтому это утверждение не может быть верным.
4. Ответ: 3.

### Ещё пример задания:

*Дан фрагмент электронной таблицы.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **A** | **B** | **C** | **D** |
| **1** | 1 | 2 | 3 |  |
| **2** | 5 | 4 | =$A$2+B$3 |  |
| **3** | 6 | 7 | =A3+B3 |  |

*Чему станет равным значение ячейки D1, если в неё скопировать формулу из ячейки С2?*

*Примечание: знак $ обозначает абсолютную адресацию.*

1) 18 2) 12 3) 14 4) 17

**Решение:**

1. при копировании формулы в другую ячейку все абсолютные ссылки на строки и столбцы (перед которыми стоит знак $) сохраняются, а все относительные – изменяются в соответствии со сдвигом формулы: если, например, формулу скопировали на 3 столбца вправо и на одну строку вверх, все «незаблокированные» адреса столбцов увеличиваются на 3, а все номера строк, перед которыми нет знака $, уменьшаются на 1
2. формула в ячейке С3 (=$A$2+B$3) содержит одну абсолютную ссылку ($A$2), которая при копировании не меняется (и строка, и столбец заблокированы) и одну смешанную (B$3), в которой столбец B будет изменяться, а строка 3 – нет
3. при копировании из C2 в D1 столбец увеличивается на 1, поэтому вместо B будет C, так что окончательный вид формулы в ячейке D1 после копирования – «=$A$2+C$3»
4. вычисление этого выражения дает 5 + (6 + 7) = 18, это вариант 1
5. Ответ: 1.

### Ещё 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:**

1. ссылка $C3 – это смешанная ссылка, в которой «заблокирован» столбец C, а строка 3 – это относительный адрес;
2. 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 $C5
3. константы при копировании формул не меняются, поэтому получится =$C5\*2
4. таким образом, правильный ответ – 2.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + если ошибочно посчитать, что знак $ защищает от изменений всю ссылку, получим неверный ответ 4 |

### Ещё пример задания:

*Три страны: Королевство Бельгия, Королевство Нидерланды и Великое Герцогство Люксембург образуют экономико-политический союз, который носит название Бенилюкс. Ниже приведен фрагмент электронной таблицы, характеризующий каждую из стран союза и союз в целом:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | А | B | C | D |
| 1 | Страна | Население (тыс. чел) | Площадь  (кв. км) | Плотность населения (чел / кв.км) |
| 2 | Бельгия | 10 415 | 30 528 | 341 |
| 3 | Нидерланды | 16 357 | 41 526 | 394 |
| 4 | Люксембург | 502 | 2 586 | 194 |
| 5 | Бенилюкс в целом | 27 274 | 74 640 |  |

Какое значение должно стоять в ячейке D5?

1) 365 2) 929 3) 310 4) 2,74

**Решение:**

1. нужно не забыть, что плотность населения вычисляется как отношение населения к площади (не наоборот!);
2. население не забываем перевести из тысяч человек в единицы: 27 274 000 чел
3. поэтому для всего Бенилюкса получаем 27 274 000 / 74 640 ≈ 365
4. таким образом, правильный ответ – 1.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + в такой простой задаче есть сильная ловушка: ответ 4 (2,74) получается при «обратном» делении, то есть 74 640 / 27 274 ≈ 2,74 |

### Еще пример задания:

*В электронной таблице значение формулы* **=СУММ(B1:B2)** *равно 5. Чему равно значение ячейки B3, если значение формулы* **=СРЗНАЧ(B1:B3)** *равно 3?*

1) 8 2) 2 3) 3 4) 4

**Решение:**

1. функция **СУММ(B1:B2)** считает сумму значений ячеек B1 и B2, поэтому B1 + B2 = 5
2. функция **СРЗНАЧ(B1:B3)** считает среднее арифметическое диапазона B1:B3
3. строго говоря, такие задачи некорректны, потому что
   1. функция СРЗНАЧ учитывает только числовые данные (числа или формулы, при вычислении которых получается число), то есть возможны варианты:

**СРЗНАЧ(B1:B3)=СУММ(B1:B3)**, если есть только одна числовая ячейка

**СРЗНАЧ(B1:B3)=СУММ(B1:B3)/2**, если есть две числовых ячейки

**СРЗНАЧ(B1:B3)=СУММ(B1:B3)/3**, если все три ячейки – числовые

* 1. в условии не задано, сколько числовых ячеек в диапазоне **B1:B3**

1. в такой ситуации логичнее всего считать, что все три ячейки содержат числовые данные (во всех известных автору задачах такого типа используется именно это допущение)
2. итак, в диапазон **B1:B3** входят три ячейки; предполагаем, что все они содержат числовые данные, тогда среднее арифметическое – это сумма их значений, деленная на 3; таким образом B1 + B2 + B3 = 3 · 3 = 9
3. поскольку B1 + B2 = 5, сразу получаем B3 = 9 – 5 = 4
4. таким образом, правильный ответ – 4.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + чтоб сбить угадывание, среди ответов приведены сумма исходных данных (8) и их разность (2) , это неверные ответы |

### Еще пример задания:

*Дан фрагмент электронной таблицы:*

|  |  |  |  |
| --- | --- | --- | --- |
|  | А | В | С |
| 1 | 10 | 20 | = A1+B$1 |
| 2 | 30 | 40 |  |

*Чему станет равным значение ячейки* С2*, если в нее скопировать формулу из ячейки* С1*? Знак* $ *обозначает абсолютную адресацию.*

1) 40 2) 50 3)60 4) 70

**Решение:**

1. это задача на использование абсолютных и относительных адресов в электронных таблицах
2. вспомним, что при копировании все относительные адреса меняются (согласно направлению перемещения формулы), а абсолютные – нет
3. в формуле, которая находится в C1, используются два адреса: A1 и B$1
4. адрес A1 – относительный, он может изменяться полностью (и строка, и столбец)
5. адрес B$1 – смешанный, в нем номер строки «зафиксирован» знаком доллара, а имя столбца – нет, поэтому при копировании может измениться только имя столбца
6. при копировании из C1 в C2 столбец не изменяется, а номер строки увеличивается на 1, поэтому в C2 получим формулу **=A2+B$1** (здесь учтено, что у второго адреса номер строки «зафиксирован»)
7. сумма ячеек A2 и B1 равна 30 + 20 = 50
8. таким образом, правильный ответ – 2.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + расчет на то, что ученик забудет, что абсолютная ссылка не меняется (тогда получится формула **=A2+B$2**, на этот случай дан неверный ответ 70) |

### Еще пример задания:

*Дан фрагмент электронной таблицы:*

|  |  |  |  |
| --- | --- | --- | --- |
|  | А | В | С |
| 1 | 1 | 2 |  |
| 2 | 2 | 6 | =СЧЁТ(A1:B2) |
| 3 |  |  | =СРЗНАЧ(A1:C2) |

*Как изменится значение ячейки* С3*, если после ввода формул переместить содержимое ячейки* В2 в В3?(«+1» означает увеличение на 1, а «–1» – уменьшение на 1)

1) –2 2) –1 3) 0 4) +1

**Решение:**

1. это задача на знание особенностей функций СЧЕТ и СРЗНАЧ, которые не учитывают пустые ячейки
2. после ввода формул в С2 окажется количество непустых ячеек диапазона А1:В2, равное 4
3. в С3 будет выведено среднее значение диапазона А1:С2 равное

(1+2+2+6+4)/5 = 3

1. после *перемещения* (не копирования!) содержимого ячейки В2 в В3 ячейка В2 окажется пустой, поэтому в С2 выводится число 3 – количество *непустых* ячеек диапазона А1:В2
2. в С3 будет выведено среднее значение диапазона А1:С2 равное

(1+2+2+3)/4 = 2,

то есть значение С3 уменьшится на 1

1. таким образом, правильный ответ – 2.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + нужно помнить, что при перемещении содержимого ячейки в другое место она становится пустой   + нужно помнить, что функции СЧЕТ и СРЗНАЧ не учитывают пустые ячейки |

# 7-2 (базовый уровень, время – 3 мин)

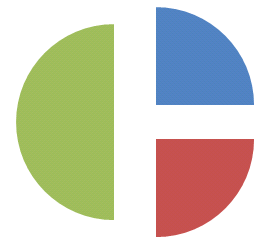
**Тема**: Представление данных в электронных таблицах в виде диаграмм и графиков.

**Что нужно знать**:

* что такое столбчатая, линейчатая и круговая диаграмма, какую информацию можно получить с каждой из них
* адрес ячейки в электронных таблицах состоит из имени столбца и следующего за ним номера строки, например, C15
* формулы в электронных таблицах начинаются знаком = («равно»)
* знаки +, –, \*, / и ^ в формулах означают соответственно сложение, вычитание, умножение, деление и возведение в степень
* в заданиях ЕГЭ могут использоваться стандартные функции СУММ (сумма), СРЗНАЧ (среднее значение), МИН (минимальное значение), МАКС (максимальное значение)
* запись B2:C4 означает диапазон, то есть, все ячейки внутри прямоугольника, ограниченного ячейками B2 и C4; например, с помощью формулы =СУММ(B2:C4) вычисляется сумма значений ячеек B2, B3, B4, C2, C3 и C4

### Ещё пример задания:

*Дан фрагмент электронной таблицы:*



|  |  |  |  |
| --- | --- | --- | --- |
|  | **A** | **B** | **C** |
| **1** | **???** | **4** | **6** |
| **2** | **=(A1–2)/(B1–1)** | **=C1\*B1/(4\*A1+4)** | **=C1/(A1–2)** |

*Какое целое число должно быть записано в ячейке A1, чтобы диаграмма, построенная по значениям ячеек диапазона A2:С2, соответствовала рисунку? Известно, что все значения ячеек из рассматриваемого диапазона неотрицательны.*

**Решение:**

1. сначала предполагаем, что диаграмма не повернута, то есть первый сектор начинается с направления «на север» (вверх от центра)
2. по диаграмме определяем, что третий сектор в два раза больше остальных двух, то есть A2 = B2 = C2/2
3. обозначив значение A1 за *x*, записываем значения ячеек второй строки:



1. чтобы найти *x*, можно решить одно из трёх уравнений:

A2 = B2, B2 = C2/2, A2 = C2/2

причём проще решать уравнение B2=C2/2, поскольку оно линейное, а остальные два сводятся к квадратным уравнениям

1. решим уравнение B2 = C2/2:



1. проверяем условие A2=B2 при *x = 5*:

 - истинно

1. ответ: 5.

### Ещё пример задания:

*Дан фрагмент электронной таблицы:*



|  |  |  |  |
| --- | --- | --- | --- |
|  | **A** | **B** | **C** |
| **1** | **6** | **1** | **???** |
| **2** | **=(C1+3)/(A1+6)** | **=(4+B1)/(C1-1)** | **=(A1-1)/(C1-B1)** |

*Какое число должно быть записано в ячейке C1, чтобы построенная после выполнения вычислений диаграмма по значениям диапазона ячеек A2:C2 соответствовала рисунку справа? Все значения в ячейках диапазона A1:C1 имеют одинаковый знак.*

**Решение:**

1. по диаграмме определяем, что все секторы равны, то есть A2 = B2 = C2
2. обозначив значение C1 за *x*, записываем значения ячеек второй строки:



1. очевидно, что B2 = C2 при любом *x*, поэтому остаётся обеспечить условие A2 = B2:



1. полагая, что *x* не равен 1, получаем квадратное уравнение:



1. это уравнение имеет два решения: –9 и 7; поскольку по условию нас интересуют только положительные решения (все ячейки диапазона A1:C2 имеют один знак, положительны), подходит только второе из решений
2. ответ: 7.

### Еще пример задания:

*Дан фрагмент электронной таблицы:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **A** | **B** | **C** | **D** |
| **1** | **3** |  | **3** | **2** |
| **2** | **=(C1+A1)/2** | **=C1–D1** | **=A1–D1** | **=B1/2** |

*Какое число должно быть записано в ячейке B1, чтобы построенная после выполнения вычислений диаграмма по значениям диапазона ячеек A2:D2 соответствовала рисунку:*



**Решение:**

1. прежде всего, нужно понять, что мы видим круговую диаграмму, которая строится по одному ряду данных и показывает доли частей в чем-то целом
2. по диаграмме находим, что первая часть составляет половину целого, а остальные три равны, каждая составляет по одной шестой (в 3 раза меньше, чем первая).
3. вычислим значения во второй строке, которые уже можно найти по исходным данным:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **A** | **B** | **C** | **D** |
| **1** | **3** |  | **3** | **2** |
| **2** | **3** | **1** | **1** | **=B1/2** |

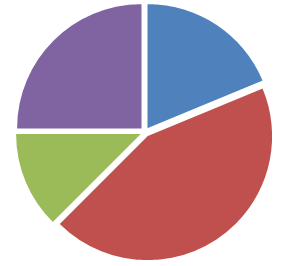
1. единственная неизвестная ячейка (зависящая от **B1**) – это **D2**, содержащая формулу **B1/2**
2. как мы узнали из диаграммы (п. 2), значение одной (первой) ячейки должно быть в 3 раза больше каждой из оставшихся, поэтому в **D2** должно быть число 1; это возможно только при **B1 = 2**
3. ответ: 2.

### Еще пример задания:

*Дан фрагмент электронной таблицы:*

|  |  |  |
| --- | --- | --- |
|  | **A** | **B** |
| **1** | **6** | **=A1-A2** |
| **2** |  | **=A3-A2** |
| **3** | **10** | **=A1/B1** |
| **4** | **18** | **=B2-B1** |

*Какое число должно быть записано в ячейке A2, чтобы построенная после выполнения вычислений диаграмма по значениям диапазона ячеек B1:B4 соответствовала рисунку:*



**Решение:**

1. это круговая диаграмма, которая строится по одному ряду данных и показывает доли частей в чем-то целом
2. по диаграмме четко видно, что

* одно из значений равно четверти от общего количества (фиолетовый сектор)
* предыдущее значение примерно в 2 раза меньше
* красный сектор (напротив фиолетового больше всех)

1. обозначив значение A2 через  и подставляя известные данные, находим, что диаграмма строится по значениям , ,  и 4
2. предполагаем, что диаграмма не повернута, то есть, фиолетовому сектору соответствует значение 4, и оно составляет одну четверть от общей суммы
3. тогда сумма равна 16, получаем уравнение:



1. отсюда следует







1. это уравнение имеет два решения,  и 
2. при  получаем в ячейках B1:B4 значения 3, 7, 2 и 4, что соответствует диаграмме
3. при  получаем в ячейках B1:B4 значения 1, 5, 6 и 4, что НЕ соответствует диаграмме (значение для предпоследнего сектора должно быть 2)
4. заметим, что можно было немного ускорить решение, используя условие , откуда сразу следует, что ; при этом квадратное уравнение решать не нужно
5. ответ: 3.

### Еще пример задания:



*Дан фрагмент электронной таблицы:*

|  |  |  |  |
| --- | --- | --- | --- |
|  | **A** | **B** | **C** |
| **1** | **6** |  | **=A1/2** |
| **2** | **=B1-4** | **=(B1–C1)/2** | **=B2+C1** |

*Какое число должно быть записано в ячейке B1, чтобы построенная после выполнения вычислений диаграмма по значениям диапазона ячеек A2:C2 соответствовала рисунку:*

**Решение:**

1. прежде всего, нужно понять, что мы видим круговую диаграмму, которая строится по одному ряду данных и показывает доли частей в чем-то целом
2. по диаграмме видим, что два сектора одинаковые, а оставшийся значительно больше (в 4-5 раз)
3. обозначим значение ячейки B1 через  и подставим все известные данные:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **A** | **B** | **C** |
| **1** | **6** | **x** | **3** |
| **2** | **=x-4** | **=(x-3)/2** | **=(x+3)/2** |

1. предположив, что диаграмма не повернута (начало «раскрутки» совпадает с направлением на север – вертикально вверх), попытаемся приравнять первый и последний сектора;
2. решение уравнения  дает , при этом получаем

|  |  |  |  |
| --- | --- | --- | --- |
|  | **A** | **B** | **C** |
| **1** | **6** | **11** | **3** |
| **2** | **7** | **4** | **7** |

здесь третий (оставшийся) сектор (B2 = 4) меньше, чем найденные 2 (A2 = C2 = 7), этот вариант не подходит, потому что не соответствует диаграмме; значит, диаграмма повернута;

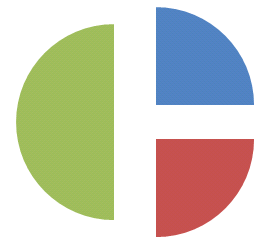
1. рассмотрим следующий вариант, приравняв ячейки A2 и B2;
2. решение уравнения  дает , при этом получаем

|  |  |  |  |
| --- | --- | --- | --- |
|  | **A** | **B** | **C** |
| **1** | **6** | **5** | **3** |
| **2** | **1** | **1** | **4** |

1. этот вариант соответствует диаграмме (третий сектор больше остальных двух в 4 раза); диаграмма повернута на 60 градусов
2. ответ: 5.

### Еще пример задания (ege.yandex.ru):

*Дан фрагмент электронной таблицы:*



|  |  |  |  |
| --- | --- | --- | --- |
|  | **A** | **B** | **C** |
| **1** | **4** | **???** | **???** |
| **2** | **=4\*C1** | **=B1–C1** | **=B2+A1** |

*Какое число должно быть записано в ячейке B1, чтобы построенная после выполнения вычислений диаграмма по значениям диапазона ячеек A2:C2 соответствовала рисунку:*

**Решение:**

1. обозначим значения ячеек B1 и C1 соответственно через  и , и вычислим значения остальных ячеек через эти переменные:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **A** | **B** | **C** |
| **1** | **4** | **???** | **???** |
| **2** | **4\*y** | **x–y** | **x-y+4** |

1. по диаграмме видно, что два сектора имеют одинаковый размер, а оставшийся в два раза больше; в принципе диаграмма может быть повернута, поэтому любые два сектора могут быть равны
2. в нашем случае по формулам видим, что C2 = B2 + 4, поэтому C2 – самый большой сектор, и



1. получаем систему уравнений:



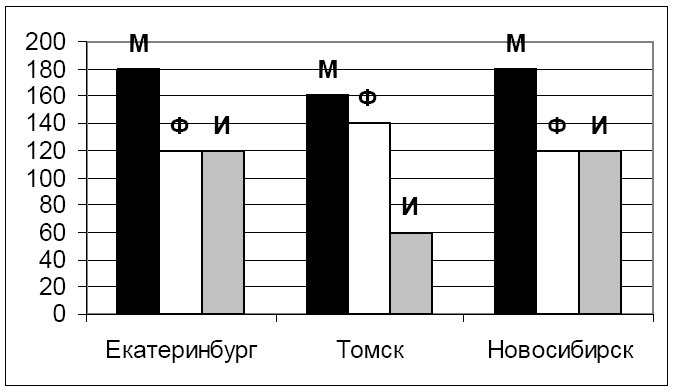
1. из первого уравнения получаем , подставляем этот результат во второе:



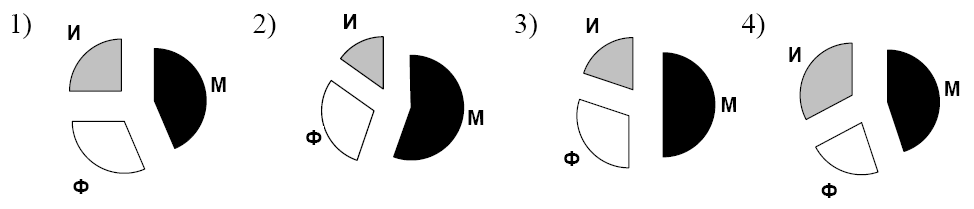
1. тогда , и значения ячеек A2 = B2 = 4, C2 = 8; все они положительны и удовлетворяют условию задачи
2. ответ: 5.

### Еще пример задания:

*На диаграмме показано количество призеров олимпиады по информатике (И), математике (М), физике (Ф) в трех городах России.*



*Какая из диаграмм правильно отражает соотношение общего числа призеров по каждому предмету для всех городов вместе?*



**Решение:**

1. в условии дана столбчатая диаграмма, по которой можно определить все числовые данные
2. в ответах все диаграммы – круговые, по ним можно определить только доли отдельных составляющих в общей сумме
3. при анализе диаграмм-ответов нужно «вылавливать» их характерные черты (половину или четверть круга, одинаковые значения, соотношения между секторами), именно они позволяют определить верный ответ
4. попробуем сначала проанализировать круговые диаграммы (ответы)

* наибольшая доля (на всех диаграммах) приходится на математику
* самый меньший сектор на диаграммах 1-3 – информатика, а на 4-ой – физика
* на 1-ой диаграмме информатика составляет четверть от общей суммы
* на 3-ей диаграмме математика составляет половину от общей суммы

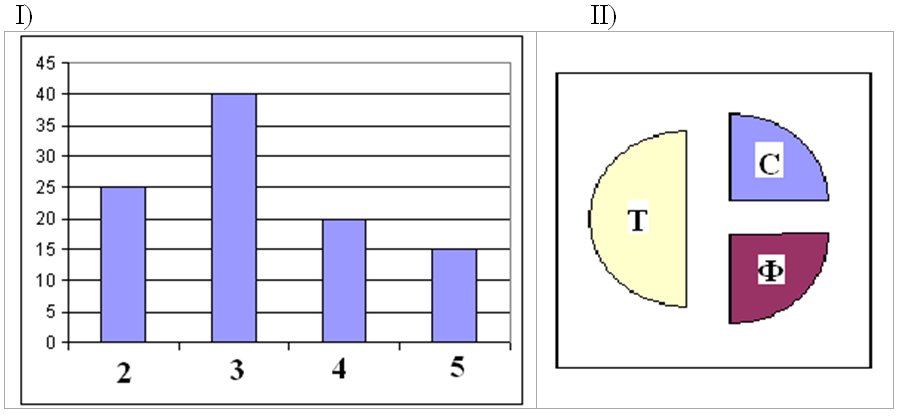
1. теперь снимем данные с заданной столбчатой диаграммы и подсчитаем сумму призеров по каждому предмету:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | М | Ф | И | **Всего** |
| Екатеринбург | 180 | 120 | 120 |  |
| Томск | 160 | 140 | 60 |  |
| Новосибирск | 180 | 120 | 120 |  |
| Всего | **520** | **380** | **300** | **1200** |

1. по условию для построения круговой диаграммы использовалась нижняя строка таблицы
2. общее количество призеров ­ – 1200, информатика составляет ровно **четверть** от этого числа
3. таким образом, правильный ответ – 1.

### Еще пример задания:

*В цехе трудятся рабочие трех специальностей – токари (Т), слесари (С) и фрезеровщики (Ф). Каждый рабочий имеет разряд не меньший второго и не больший пятого. На диаграмме I отражено количество рабочих с различными разрядами, а на диаграмме II – распределение рабочих по специальностям. Каждый рабочий имеет только одну специальность и один разряд.*



Имеются четыре утверждения:

А) Все рабочие третьего разряда могут быть токарями

Б) Все рабочие третьего разряда могут быть фрезеровщиками

В) Все слесари могут быть пятого разряда

Г) Все токари могут быть четвертого разряда

Какое из этих утверждений следует из анализа обеих диаграмм?

1) А 2) Б 3) В 4) Г

**Решение:**

1. в условии даны столбчатая диаграмма, по которой можно определить все числовые данные, и круговая диаграмма, по которой можно определить только доли отдельных составляющих в общей сумме
2. по данным столбчатой диаграммы определим, сколько рабочих имеют 2-ой, 3-й, 4-й и 5-й разряды:

2-ой разряд: 25 чел. 3-й разряд: 40 чел.

4-й разряд: 20 чел. 5-й разряд: 15 чел.

1. сложив все эти числа, определим, что всего в цехе 25 + 40 + 20 + 15 = 100 рабочих
2. по круговой диаграмме видим, что половина из них – токари (значит их 50 человек), четверть – слесари (25 чел.) и еще четверть – фрезеровщики (25 чел.)
3. теперь последовательно рассмотрим все утверждения-ответы:

А: Все рабочие третьего разряда (их 40 чел.) **МОГУТ** быть токарями, потому в цеху 50 токарей

Б: Все рабочие третьего разряда (их 40 чел.) **НЕ** могут быть фрезеровщиками, потому в цеху всего 25 фрезеровщиков

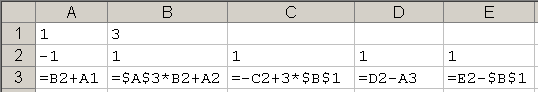
В: Все слесари (их 25 чел.) **НЕ** могут быть 5-ого разряда, потому в цеху только 15 рабочих имеют 5-й разряд

Г: Все токари (их 50 чел.) **НЕ** могут быть четвертого разряда, потому в цеху только 20 рабочих имеют 4-й разряд

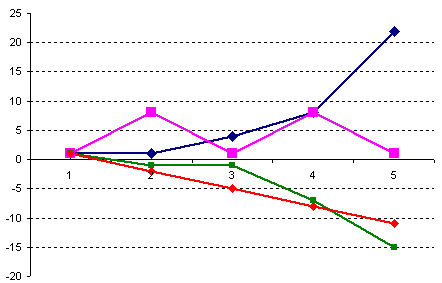
1. таким образом, правильный ответ – 1.

### Еще пример задания:

*Дан фрагмент электронной таблицы в режиме отображения формул.*



*После копирования диапазона ячеек АЗ:ЕЗ в диапазон А4:Е6 была построена диаграмма (график) по значениям столбцов диапазона ячеек В2:Е6.*

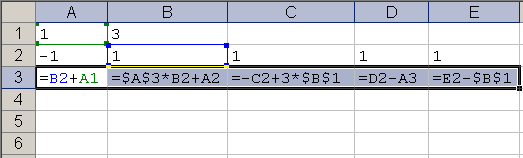


*Значениям С2:С6 соответствует график*

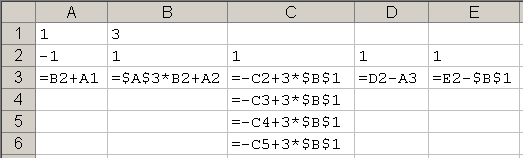
1) А 2) Б 3) В 4) Г

**Решение:**

1. прежде всего разберемся, что значит фраза «*После копирования диапазона ячеек АЗ:ЕЗ в диапазон А4:Е6*»; очевидно, что размеры диапазонов АЗ:ЕЗ и А4:Е6 разные, поэтому авторы задачи имели ввиду следующее: выделяется диапазон АЗ:ЕЗ и «растягивается» вниз за маркер заполнения до строки 6:



1. при этом формула, находящаяся в А3, скопируется в ячейки А4:А6, формула из В3 – в ячейки В4:В6 и т.д.
2. по условию нас в конечном счете интересует только столбец С, посмотрим, что получится при копировании формулы из С3 (**=-C2+3\*$B$1**) в ячейки С4:С6
3. в этой формуле есть ссылки на две ячейки – одна относительная, на С2 (при копировании она будет меняться (на С3, С4 и т.д.), а вторая – абсолютная, на В1, она при копировании не изменится:



1. видим, что формулы в столбце С зависят только от В1 и ячеек этого же столбца, поэтому не нужно рассчитывать все остальные ячейки
2. последовательно найдем все числа в диапазоне С3:С6:

**С3=-С2+3\*В1=-1+3\*3=8**

**С4=-С3+3\*В1=-8+3\*3=1**

**С5=-С4+3\*В1=-1+3\*3=8**

**С6=-С5+3\*В1=-8+3\*3=1**

1. посмотрев на график, видим, что именно так меняются данные на графике Б
2. таким образом, правильный ответ – 2.

### 

# 8 (базовый уровень, время – 3 мин)

**Тема**: Анализ программы.

**Что нужно знать**:

* основные конструкции языка программирования:
  + объявление переменных
  + оператор присваивания
  + оператор вывода
  + циклы
* уметь выполнять ручную прокрутку программы
* уметь выделять переменную цикла, от изменения которой зависит количество шагов цикла
* уметь определять количество шагов цикла
* уметь определять переменную, которая выводится на экран
* формулу для вычисления -ого элемента арифметической прогрессии:



* формулу для вычисления суммы первых  членов арифметической прогрессии:



где  – -ый элемент последовательности,  – шаг (разность) последовательности

### Пример задания:

*При каком наибольшем введенном числе d после выполнения программы будет напечатано 55?*

**var n, s, d: integer;**

**begin**

**readln(d);**

**n := 0;**

**s := 0;**

**while s <= 365 do begin**

**s := s + d;**

**n := n + 5**

**end;**

**write(n)**

**end.**

**Решение:**

1. из программы видно, что начальные значения переменных **s** и **n** равны нулю
2. шаг изменения переменной **n** равен 5, а шаг изменения переменной **s** равен неизвестному значению **d**
3. для того, чтобы значение **n** стало равно 55, нужно увеличить его на 5 (с нуля) ровно 11 раз, поэтому цикл выполнится ровно 11 раз
4. следовательно, s увеличится на d тоже 11 раз и станет равно 0 + 11·d = 11·d
5. чтобы цикл остановился на 11-м шаге, нужно выполнить условие 11·d > 365, при этом он не должен остановиться на 10-м шаге, то есть, 10·d ≤ 365, поэтому получаем два неравенства:





1. в итоге значение d – целое число – ограничено отрезком [34; 36], наибольшее из подходящих чисел равно 36
2. Ответ: 36.

### Ещё пример задания:

*Запишите число, которое будет напечатано в результате выполнения программы.*

**var s, n: integer;**

**begin**

**s := 33;**

**n := 1;**

**while s > 0 do begin**

**s := s – 7;**

**n := n \* 3**

**end;**

**writeln(n)**

**end.**

**Решение:**

1. из программы видно, что начальные значения переменных **s** и **n** равны соответственно 33 и 1
2. цикл заканчивается, когда нарушается условие **s > 0**, то есть количество шагов цикла определяется изменением переменной **s**
3. после окончания цикла выводится значение переменной **n**
4. таким образом, задача сводится к тому, чтобы определить число шагов цикла, необходимое для того, чтобы значение **s** стало меньше или равно 0
5. с каждым шагом цикла значение **s** уменьшается на 7, а значение **n** увеличивается в 3 раза, так что **n=3k**, где **k** – это число шагов цикла
6. поскольку **s** уменьшается на 7, конечное значение **s** должно быть равно **33-7\*k** , причём первое значение, меньшее или равное 0, достигается при **k=5** (и **s=33–7\*5=-2**)
7. тогда **n=3k=35=243**
8. Ответ: 243.

### Ещё пример задания:

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

**var k, s: integer;**

**begin**

**s:=0;**

**k:=0;**

**while s < 1024 do begin**

**s:=s+10;**

**k:=k+1;**

**end;**

**write(k);**

**end.**

**Решение:**

1. из программы видно, что начальные значения переменных **k** и **s** равны нулю
2. цикл заканчивается, когда нарушается условие **s < 1024**, то есть количество шагов цикла определяется изменением переменной **s**
3. после окончания цикла выводится значение переменной **k**
4. таким образом, задача сводится к тому, чтобы определить число шагов цикла, необходимое для того, чтобы значение **s** стало не меньше 1024
5. с каждым шагом цикла значение **s** увеличивается на 10, а значение **k** – на единицу, так что фактически **k** – это счётчик шагов цикла
6. поскольку **s** увеличивается на 10, конечное значение **s** должно быть кратно 10, то есть это 1030 > 1024
7. для достижения этого значения переменную **s** нужно 103 раза увеличить на 10, поэтому цикл выполнится 103 раза
8. так как **k** – это счётчик шагов цикла, конечное значение **k** будет равно 103
9. Ответ: 103.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * можно перепутать переменную, которая выводится на экран (внимательно смотрим на оператор вывода) |

### Ещё пример задания:

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

**var k, s: integer;**

**begin**

**k:=5;**

**s:=2;**

**while k < 120 do begin**

**s:=s+k;**

**k:=k+2;**

**end;**

**write(s);**

**end.**

**Решение:**

1. начальные значения переменных **k** и **s** равны соответственно 5 и 2
2. цикл заканчивается, когда нарушается условие **k** **<** **120**, то есть количество шагов цикла определяется изменением переменной **k**
3. после окончания цикла выводится значение переменной **s**
4. с каждым шагом цикла значение **s** увеличивается на **k**, а затем значение **k** – на 2, так что к начальному значению **s** добавляется сумма членов арифметической прогрессии с начальным значением  и разностью 
5. поскольку начальное значение **k** равно 5 и с каждым шагом оно увеличивается на 2, переменная **k** принимает последовательно нечётные значения: 5, 7, 9, …
6. цикл заканчивается, когда значение **k** становится не меньше 120; поскольку **k** всегда нечётное, конечное значение **k** равно 121
7. поскольку значение **k** увеличивается после того, как увеличивается значение s, значение 121 уже не входит в сумму, то есть последний элемент последовательности :



1. количество  членов последовательности, которые входят в сумму, можно вычислить: чтобы из 5 получить 119 нужно 57 раз добавить шаг 2, поэтому общее число элементов последовательности равно (на один больше)
2. теперь используем формулу для вычисления суммы членов арифметической прогрессии:



1. к этой сумме нужно добавить начальное значение переменной **s**, равное 2:



1. Ответ: 3598.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * попытка делать ручную трассировку, скорее всего, приведет к вычислительной ошибке, потому что число шагов слишком велико * легко забыть, что начальные значения переменных **s** и **k** не равны нулю * нужно помнить, что количество членов арифметической прогрессии на 1 больше, чем количество шагов, которые необходимы для перехода от первого значения к последнему |

# 9-1 (базовый уровень, время – 5 мин)

**Тема**: Кодирование звука.

**Что нужно знать**:

* при оцифровке звука в памяти запоминаются только отдельные значения сигнала, который нужно выдать на динамик или наушники
* частота дискретизации определяет количество отсчетов, запоминаемых за 1 секунду; 1 Гц (один герц) – это один отсчет в секунду, а 8 кГц – это 8000 отсчетов в секунду
* глубина кодирования – это количество бит, которые выделяются на один отсчет
* для хранения информации о звуке длительностью  секунд, закодированном с частотой дискретизации Гц и глубиной кодирования бит требуется  бит памяти; например, при кГц, глубине кодирования 16 бит на отсчёт и длительности звука 128 секунд требуется

бит

байт

Кбайт

Мбайт

* при двухканальной записи (стерео) объем памяти, необходимый для хранения данных одного канала, умножается на 2
* для упрощения ручных расчетов можно использовать приближённые равенства

1 мин = 60 сек ≈ 64 сек = 26 сек

1000 ≈ 1024 = 210

* нужно помнить, что

1 Мбайт = 220 байт = 223 бит,

1 Кбайт = 210 байт = 213 бит

### Пример задания:

*Производилась двухканальная (стерео) звукозапись с частотой дискретизации 64 кГц и 24-битным разрешением. В результате был получен файл размером 120 Мбайт, сжатие данных не производилось. Определите приблизительно, сколько времени (в минутах) производилась запись. В качестве ответа укажите ближайшее к времени записи целое число, кратное 5.*

**Решение (через степени двойки):**

1. так как частота дискретизации 64 кГц, за одну секунду запоминается 64000 значений сигнала
2. так как глубина кодирования – 24 бита = 3 байта, для хранения 1 секунды записи требуется

2 × 64000 × 3 байта

(коэффициент 2 – для стерео записи)

1. на 1 минуту = 60 секунд записи потребуется

60 × 2 × 64000 × 3 байта

1. переходим к степеням двойки, заменяя 60 ← 64 = 26; 1000 ← 1024 = 210:

26 × 21 × 26 × 210 × 3 байта = 26 × 21 × 26 × 3 Кбайта

= 22 × 21 × 3 Мбайта = 24 Мбайта

1. тогда время записи файла объёмом 120 Мбайт равно 120 / 24 = 5 минут
2. таким образом, правильный ответ – 5.

### Ещё пример задания:

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

1) 0,2 2) 2 3) 3 4) 4

**Решение (вариант 1, «в лоб»):**

1. так как частота дискретизации 16 кГц, за одну секунду запоминается 16000 значений сигнала
2. так как глубина кодирования – 24 бита = 3 байта, для хранения 1 секунды записи требуется

16000 × 3 байта = 48 000 байт

(для стерео записи – в 2 раза больше)

1. на 1 минуту = 60 секунд записи потребуется

60 × 48000 байта = 2 880 000 байт,

то есть около 3 Мбайт

1. таким образом, правильный ответ – 3.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + если указано, что выполняется двухканальная (стерео) запись, нужно не забыть в конце умножить результат на 2   + могут получиться довольно большие числа, к тому же «некруглые» (к сожалению, использовать калькулятор по-прежнему запрещено) |

**Решение (вариант 2, через степени двойки, с сайта ege-go.ru):**

1. обратите внимание, что в этой задаче не требуется ТОЧНО вычислять размер файла, нужно только выполнить прикидочные расчеты
2. в этом случае, если нет калькулятора (а на ЕГЭ его нет) удобно привести все числа к ближайшим степеням двойки, например,

1 мин = 60 сек ≈ 64 сек = 26 сек

1000 ≈ 1024 = 210

1. так как частота дискретизации 16 кГц, за одну секунду запоминается 16000 значений сигнала, что примерно равно

16 × 1000 ≈ 16 × 1024 = 24 × 210 = 214 Гц

1. так как глубина кодирования – 24 бита = 3 байта, для хранения 1 секунды записи требуется

16000 × 3 байта ≈ 214 × 3 байт

(для стерео записи – в 2 раза больше)

1. на 1 минуту = 60 сек ≈ 64 сек = 26 сек записи потребуется примерно

64 × 214 × 3 байта = 26 × 214 × 3 байта = 3 × 220 байта

1. переводит эту величину в Мбайты:

(3 × 220 байта) / 220 = 3 Мбайт

1. таким образом, правильный ответ – 3.

### Еще пример задания:

*Производится одноканальная (моно) звукозапись с частотой дискретизации 64Гц. При записи использовались 32 уровня дискретизации. Запись длится 4 минуты 16 секунд, её результаты записываются в файл, причём каждый сигнал кодируется минимально возможным и одинаковым количеством битов. Какое из приведённых ниже чисел наиболее близко к размеру полученного файла, выраженному в килобайтах?*

1) 10 2) 64 3) 80 4) 512

**Решение:**

1. так как частота дискретизации 64 Гц, за одну секунду запоминается 64 значения сигнала
2. глубина кодирования не задана!
3. используется 32 = 25уровня дискретизации значения сигнала, поэтому на один отсчет приходится 5 бит
4. время записи 4 мин 16 с = 4 × 60 + 16 = 256 с
5. за это время нужно сохранить

256 × 5 × 64 бит = 256 × 5 × 8 байт = 5 × 2 Кбайт = 10 Кбайт

1. таким образом, правильный ответ – 1.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + если указано, что выполняется двухканальная (стерео) запись, нужно не забыть в конце умножить результат на 2   + если «по инерции» считать, что 32 – это глубина кодирования звука в битах, то получим неверный ответ 64 Кбайта |

# 9-2 (повышенный уровень, время – 5 мин)

**Тема**: Определение скорости передачи информации при заданной пропускной способности канала.

**Что нужно знать**:

* «физический» аналог задачи:

лимонад

*пропускная способность – 10 л/мин*

лимонад

сколько лимонада перекачается по трубе за 1 час?  
ответ: 10 л/мин · 60 мин = 600 л

* любой канал связи имеет ограниченную пропускную способность (скорость передачи информации), это число ограничивается свойствами аппаратуры и самой линии (кабеля)
* объем переданной информации  вычисляется по формуле , где  – пропускная способность канала (в битах в секунду или подобных единицах), а  – время передачи

### Пример задания:

*Документ объёмом 40 Мбайт можно передать с одного компьютера на другой двумя способами.*

*А. Сжать архиватором, передать архив по каналу связи, распаковать.*

*Б. Передать по каналу связи без использования архиватора.*

*Какой способ быстрее и насколько, если:*

*• средняя скорость передачи данных по каналу связи составляет 223 бит в секунду;*

*• объём сжатого архиватором документа равен 90% исходного;*

*• время, требуемое на сжатие документа, – 16 секунд, на распаковку – 2 секунды?*

*В ответе напишите букву А, если быстрее способ А, или Б, если быстрее способ Б. Сразу после буквы напишите число, обозначающее, на сколько секунд один способ быстрее другого?*

**Решение:**

1. вспомним, что 1 Мбайт = 210 Кбайт = 220 байт = 223 бит
2. время передачи несжатого файла (по варианту Б): 40 × 223 / 223 = 40 с
3. время передачи файла по варианту А: 16 + 0,9 × 40 + 2 = 18 + 36 = 54 с
4. таким образом, быстрее вариант Б на 54 – 40 = 14 с
5. Ответ: Б14.

### Ещё пример задания:

*Документ (без упаковки) можно передать по каналу связи с одного компьютера на другой за 75 секунд. Если предварительно упаковать документ архиватором, передать упакованный документ, а потом распаковать на компьютере получателя, то общее время передачи (включая упаковку и распаковку) составит 30 секунд. При этом на упаковку и распаковку данных всего ушло 15 секунд. Размер исходного документа 20 Мбайт. Чему равен размер упакованного документа (в Мбайт)?.*

**Решение:**

1. определяем скорость передачи данных по каналу связи:

***v*** = 20 Мбайт / 75 c

1. тогда время передачи упакованного файла размером ***x*** Мбайт равно

***x*** Мбайт ***/ v*** = ***x*** Мбайт / (20 Мбайт / 75 c) = (75 · ***x*** / 20) с

1. по условию это время равно 30 – 15 = 15 с
2. решаем уравнение (75 · ***x*** / 20) с = 15 с, получаем ***x*** = 4 Мбайт
3. Ответ: 4.

**Решение (А.Н. Носкин):**

1. определяем скорость передачи данных по каналу связи:

***v*** = 20 Мбайт / 75 c

1. тогда размер упакованного файла равен



где *t* = 30 – 15 = 15 с – время передачи упакованного файла

1. тогда сразу получаем *V* = (20 / 75) ⋅ 15 = 4 Мбайт
2. Ответ: 4.

### Ещё пример задания:

*Документ объёмом 40 Мбайт можно передать с одного компьютера на другой двумя способами:*

*А. Сжать архиватором, передать архив по каналу связи, распаковать.*

*Б. Передать по каналу связи без использования архиватора.*

*Какой способ быстрее и насколько, если:*

* *средняя скорость передачи данных по каналу связи составляет 220 бит в секунду;*
* *объём сжатого архиватором документа равен 40% исходного;*
* *время, требуемое на сжатие документа, – 10 секунд, на распаковку – 2 секунды?*

*В ответе напишите букву А, если быстрее способ А, или Б, если быстрее способ Б. Сразу после буквы напишите число, обозначающее, на сколько секунд один способ быстрее другого.*

*Так, например, если способ Б быстрее способа А на 50 секунд, в ответе нужно написать Б50.*

*Единицы измерения «секунд», «сек.», «с.» к ответу добавлять не нужно.*

**Решение:**

1. переводим количество информации из Мбайтов в биты

40 Мбайт = 40 · 223 бит

1. определяем время передачи несжатого файла



1. определяем время передачи сжатого файла, которое составляет 40% или 0,4 от времени передачи несжатого файла:

0,4 · 320 с = 128 с

1. определяем полное время передачи сжатого файла с учетом 10 секунд на упаковку и 2 секунд на распаковку:



1. видим, что передача документа способом А (с упаковкой) быстрее на

320 – 140 = 180 с

1. таким образом, ответ – А180.

### Ещё пример задания:

*Скорость передачи данных через ADSL-соединение равна 128000 бит/c. Через данное соединение передают файл размером 625 Кбайт. Определите время передачи файла в секундах.*

|  |
| --- |
| **Большие числа. Что делать?**  Обычно (хотя и не всегда) задачи, в условии которых даны большие числа, решаются достаточно просто, если выделить в этих числах степени двойки. На эту мысль должны сразу наталкивать такие числа как  128 = 27, 256 = 28, 512 = 29 , 1024 = 210,  2048 = 211, 4096 = 212 , 8192 = 213, 16384 = 214, 65536 = 216 и т.п.  Нужно помнить, что соотношение между единицами измерения количества информации также представляют собой степени двойки:  1 байт = 8 бит = 23 бит,  1 Кбайт = 1024 байта = 210 байта  = 210 · 23 бит = 213 бит,  1 Мбайт = 1024 Кбайта = 210 Кбайта  = 210 · 210 байта = 220 байта  = 220 · 23 бит = 223 бит.  Правила выполнения операций со степенями:   * при умножении степени при одинаковых основаниях складываются      * … а при делении – вычитаются: |

**Решение:**

1. выделим в заданных больших числах степени двойки и переведем размер файла в биты, чтобы «согласовать» единицы измерения:

 128000 бит/c = 128 · 1000 бит/с = 27 · 125 · 8 бит/с = 27 · 53 · 23 бит/с = 210 · 53 бит/с

625 Кбайт = 54Кбайт = 54 · 213 бит

1. чтобы найти время передачи в секундах, нужно разделить размер файла на скорость передачи:



1. таким образом, ответ – 40 с .

|  |
| --- |
| **Возможные проблемы**:   * + вычисления с большими числами (лучше делать через степени двойки)   + несогласованность единиц измерения, например, скорость в битах/с, а размер файла в байтах или Кбайтах; согласованные единицы измерения:   биты/с – биты, байты/с – байты, Кбайты/с – Кбайты   + чтобы не перепутать, где нужно делить, а где умножать, проверяйте размерность полученной величины |

### Еще пример задания:

*Скорость передачи данных через ADSL-соединение равна 512 000 бит/c. Передача файла через это соединение заняла 1 минуту. Определить размер файла в килобайтах.*

**Решение:**

1. выделим в заданных больших числах степени двойки; переведем время в секунды (чтобы «согласовать» единицы измерения), а скорость передачи – в Кбайты/с, поскольку ответ нужно получить в Кбайтах:

1 мин = 60с = 4 · 15 с = 22  · 15 с

 512000 бит/c = 512 · 1000 бит/с = 29 · 125 · 8 бит/с = 29 · 53 · 23 бит/с   
 = 212 · 53 бит/с = 29 · 53 байт/с =  Кбайт/с =  Кбайт/с

1. чтобы найти время объем файла, нужно умножить время передачи на скорость передачи:

Кбайт/с КбайтКбайт

1. таким образом, ответ – 3750 Кбайт.

### Еще пример задания:

*У Васи есть доступ к Интернет по высокоскоростному одностороннему радиоканалу, обеспечивающему скорость получения им информации 256 Кбит[[7]](#footnote-7) в секунду. У Пети нет скоростного доступа в Интернет, но есть возможность получать информацию от Васи по низкоскоростному телефонному каналу со средней скоростью 32 Кбит в секунду. Петя договорился с Васей, что тот будет скачивать для него данные объемом 5 Мбайт по высокоскоростному каналу и ретранслировать их Пете по низкоскоростному каналу. Компьютер Васи может начать ретрансляцию данных не раньше, чем им будут получены первые 512 Кбайт этих данных. Каков минимально возможный промежуток времени (в секундах), с момента начала скачивания Васей данных, до полного их получения Петей? В ответе укажите только число, слово «секунд» или букву «с» добавлять не нужно.*

**Решение:**

1. сначала нарисуем схему:

32 Кбит/с



256 Кбит/с



1. фактически нужно определить, сколько времени будет передаваться файл объемом 5 Мбайт по каналу со скоростью передачи данные 32 Кбит/с; к этому времени нужно добавить задержку файла у Васи (пока он не получит 512 Кбайт данных по каналу со скоростью 256 Кбит/с); можно построить такую диаграмму Ганта, где на горизонтальной оси откладывается время[[8]](#footnote-8):

время

Вася скачивает первые 512 Кбайт

Вася скачивает весь файл

Петя получает весь файл от Васи

Полное время передачи

1. согласовываем единицы измерения, находим объем файла в Кбитах:

Кбайт Кбит

1. время «чистой» передачи файла от Васи к Пете со скоростью Кбит/с:

с

1. определяем, сколько Кбит должен скачать Вася до начала передачи Пете:

Кбайт Кбит

1. задержка файла у Васи = время скачивания файла объемом 512 Кбайт со скоростью Кбит/с:

с

1. общее время с
2. таким образом, ответ – 1296 с.

|  |
| --- |
| **Возможные проблемы и ловушки**:   * + длинное и запутанное условие, сложная словесная формулировка   + несогласованность единиц измерения, например, скорость в битах/с, а размер файла в байтах или Кбайтах; согласованные единицы измерения:   биты/с – биты, байты/с – байты, Кбайты/с – Кбайты |

### Еще пример задания:

*Каково время (в минутах) передачи полного объема данных по каналу связи, если известно, что передано 150 Мбайт данных, причем первую половину времени передача шла со скоростью 2 Мбит в секунду, а остальное время – со скоростью 6 Мбит в секунду?*

**Решение (метод 1, с переменной):**

1. обозначим неизвестное время (в секундах) за X, тогда…
2. за первый период, равный X/2, передано 2 Мбит/с·X/2 = X Мбит данных
3. за вторую половину передано 6 Мбит/с·X/2 = 3·X Мбит данных
4. объем переданной информации нужно перевести из Мбайт в Мбиты:

150 Мбайт = 150·8 Мбит = 1200 Мбит

1. получаем уравнение X + 3·X = 1200 Мбит, откуда X = 300 секунд
2. переводим время из секунд в минуты (1 минута = 60 с), получаем 300/60 = 5 минут
3. таким образом, ответ – 5.

|  |
| --- |
| **Возможные проблемы и ловушки**:   * + несогласованность единиц измерения: скорость в Мбитах/с, а размер файла в Мбайтах или Кбайтах   + можно забыть перевести время из секунд в минуты |

**Решение (метод 2, А.Н. Носкин):**

1. Передача идет общее время Т.
2. Передача идет на разных скоростях 2 Мбит/с и 6 Мбит/с.
3. Отношение скоростей 2 / 6 = 1 / 3, то есть 1 часть информации передается на одной скорости (2 Мбит/с), а три части информации на другой (6 Мбит/с). Итого 1+3 = 4 части информации.
4. Переведем Мбайт в Мбиты:

150 Мбайт = 150·8 Мбит = 1200 Мбит

1. Узнаем время передачи всех частей информации 1200 / 4 = 300 секунд
2. переводим время из секунд в минуты (1 минута = 60 с), получаем 300/60 = 5 минут
3. таким образом, ответ – 5.

### Еще пример задания (ege.yandex.ru):

*Данные объемом 100 Мбайт передаются из пункта А в пункт Б по каналу связи, обеспечивающему скорость передачи данных 220 бит в секунду, а затем из пункта Б в пункт В по каналу связи, обеспечивающему скорость передачи данных 222 бит в секунду. Задержка в пункте Б (время между окончанием приема данных из пункта А и началом передачи в пункт В) составляет 24 секунды. Сколько времени (в секундах) прошло с момента начала передачи данных из пункта А до их полного получения в пункте В? В ответе укажите только число, слово «секунд» или букву «с» добавлять не нужно.*

1. построим диаграмму Ганта, которая показывает все этапы передачи данных из пункта А в пункт В:

время

передача из А в Б

передача из Б в В

Полное время передачи

задержка в Б

1. переводим количество информации в биты:

100 Мбайт = 100·223 бит

1. вычисляем время передачи данных из пункта А в пункт Б:

100·223 бит/ (220 бит/с) = 100·23 с = 800 с

1. вычисляем время передачи данных из пункта Б в пункт В:

100·223 бит/ (222 бит/с) = 100·21 с = 200 с

1. общее время передачи с учетом задержки 24 с:



1. таким образом, ответ – 1024.

# 10 (базовый уровень, время – 4 мин)

**Тема**: Анализ последовательностей, системы счисления.

**Что нужно знать**:

* русский алфавит
* принципы работы с числами, записанными в позиционных системах счисления

### Пример задания:

*Сколько существует различных символьных последовательностей длины 5 в четырёхбуквенном алфавите {A, C, G, T}, которые содержат ровно две буквы A?*

**Решение (вариант 1, перебор):**

1. рассмотрим различные варианты слов из 5 букв, которые содержат две буквы А и начинаются с А:

АА\*\*\* А\*А\*\* А\*\*А\* А\*\*\*А

Здесь звёздочка обозначает любой символ из набора {C, G, T}, то есть один из трёх символов.

1. итак, в каждом шаблоне есть 3 позиции, каждую из которых можно заполнить тремя способами, поэтому общее число комбинаций (для каждого шаблона!) равно 33 = 27
2. всего 4 шаблона, они дают 4 · 27 = 108 комбинаций
3. теперь рассматриваем шаблоны, где первая по счёту буква А стоит на второй позиции, их всего три:

\*АА\*\* \*А\*А\* \*А\*\*А

они дают 3 · 27 = 81 комбинацию

1. два шаблона, где первая по счёту буква А стоит на третьей позиции:

\*\*АА\* \*\*А\*А

они дают 2 · 27 = 54 комбинации

1. и один шаблон, где сочетание АА стоит в конце

\*\*\*АА

они дают 27 комбинаций

1. всего получаем (4 + 3 + 2 + 1) · 27 = 270 комбинаций
2. ответ: 270.

**Решение (вариант 2, использование формул комбинаторики):**

1. в последовательности из 5 символов нужно использовать ровно две буквы А и три символа, не совпадающих с А, которые обозначим звездочкой
2. сначала найдём количество перестановок из двух букв А и трёх звёздочек
3. используем формулу для вычисления числа перестановок с повторениями; для двух разных символов она выглядит так:



Здесь – количество букв А, – количество звёздочек и восклицательный знак обозначает *факториал* натурального числа, то есть произведение всех натуральных чисел от 1 до : 

1. в нашем случае  и , так что получаем



1. теперь разберёмся со звёздочками: вместо каждой из них может стоять любой из трёх символов (кроме А), то есть на каждую из 10 перестановок мы имеем 33 = 27 вариантов распределения остальных символов на месте звёздочек
2. таким образом, получаем всего 10 · 27 = 270 вариантов.
3. ответ: 270.

### Ещё пример задания:

*Сколько слов длины 5, начинающихся с гласной буквы, можно составить из букв Е, Г, Э? Каждая буква может входить в слово несколько раз. Слова не обязательно должны быть осмысленными словами русского языка.*

**Решение:**

1. первая буква слова может быть выбрана двумя способами (Е или Э), остальные – тремя
2. общее число различных слов равно 2\*3\*3\*3\*3 = 162
3. ответ: 162.

**Решение (через формулы, А.Н. Носкин):**

1. Дано слово длиной 5 символов типа \*\*\*\*\*, где красная звездочка – гласная буква (Е или Э), а черная буква любая из трёх заданных.
2. Общая формула количества вариантов:

*N = M L*, где *М* – мощность алфавита, а *L* – длина кода.

1. Так как положение одной из букв строго регламентировано (знак умножения в зависимых событиях), то формула всех вариантов примет вид: *N = M*1*L*1 *∙ M*2*L2*,
2. Тогда *M*1 *=* 2 (алфавит гласных букв), а *L*1 *=* 1 (только 1 позиция в слове).

*M*2 *=* 3 (алфавит всех букв), а *L*2 *=* 4 (оставшиеся 4 позиции в слове).

1. В итоге получаем: *N =* 21 ∙ 34 = 2 ∙ 81 = 162.
2. ответ: 162.

### Ещё пример задания:

*Все 4-буквенные слова, составленные из букв К, Л, Р, Т, записаны в алфавитном порядке и пронумерованы. Вот начало списка:*

**1. КККК**

**2. КККЛ**

**3. КККР**

**4. КККТ**

**……**

*Запишите слово, которое стоит на 67-м месте от начала списка.*

**Решение:**

1. самый простой вариант решения этой задачи – использование систем счисления; действительно, здесь расстановка слов в алфавитном порядке равносильна расстановке по возрастанию чисел, записанных в четверичной системе счисления (основание системы счисления равно количеству используемых букв)
2. выполним замену К→0, Л→1, Р→2, Т→3; поскольку нумерация слов начинается с единицы, а первое число КККК→0000 равно 0, под номером 67 будет стоять число 66, которое нужно перевести в четверичную систему: 66 = 10024
3. Выполнив обратную замену (цифр на буквы), получаем слово ЛККР.
4. Ответ: ЛККР.

### Ещё пример задания:

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

*Вот начало списка:*

**1. ААААА**

**2. ААААО**

**3. ААААУ**

**4. АААОА**

**……**

*Запишите слово, которое стоит на 240-м месте от начала списка.*

**Решение (1 способ, перебор с конца):**

1. подсчитаем, сколько всего 5-буквенных слов можно составить из трех букв;
2. очевидно, что есть всего 3 однобуквенных слова (А, О, У); двух буквенных слов уже 3×3=9 (АА, АО, АУ, ОА, ОО, ОУ, УА, УО и УУ)
3. аналогично можно показать, что есть всего 35 = 243 слова из 5 букв
4. очевидно, что последнее, 243-е слово – это УУУУУ
5. далее идём назад: предпоследнее слово УУУУО (242-е), затем идет УУУУА (241-е) и, наконец, УУУОУ (240-е)
6. Ответ: УУУОУ.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + хорошо, что требовалось найти слово, которое стоит близко к концу списка; если бы было нужно, скажем, 123-е слово, работы было бы значительно больше |

**Решение (2 способ, троичная система, идея М. Густокашина):**

1. по условию задачи важно только то, что используется набор из трех разных символов, для которых задан порядок (алфавитный); поэтому для вычислений можно использовать три любые символа, например, цифры 0, 1 и 2 (для них порядок очевиден – по возрастанию)
2. выпишем начало списка, заменив буквы на цифры:

**1. 00000**

**2. 00001**

**3. 00002**

**4. 00010**

**……**

1. это напоминает (в самом деле, так оно и есть!) числа, записанные в троичной системе счисления в порядке возрастания: на первом месте стоит число 0, на втором – 1 и т.д.
2. тогда легко понять, что 240-м месте стоит число 239, записанное в троичной системе счисления
3. переведем 239 в троичную систему: 239 = 222123
4. заменяем обратно цифры на буквы: 22212 → УУУОУ
5. Ответ: УУУОУ.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + нужно помнить, что нумерация в задаче начинается с 1, а числа в троичной системе – с нуля, поэтому для получения 240-го элемента списка нужно переводить в троичную систему число 240-1 = 239. |

**Решение (3 способ, закономерности в чередовании букв, И.Б. Курбанова):**

1. подсчитаем, сколько всего 5-буквенных слов можно составить из трех букв:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **1** | А | А | А | А | А |
| **2** | А | А | А | А | О |
| **3** | А | А | А | А | У |
| **4** | А | А | А | О | А |
| **…** | … | … | … | … | … |
| **...** |  |  |  |  |  |
| **…** |  |  |  |  |  |
| **240** | **У** | **У** | **У** | **О** | **У** |
| **241** | У | У | У | У | А |
| **242** | У | У | У | У | О |
| **243** | У | У | У | У | У |

35 = 243 слова; 240-ое место – четвертое с конца;

1. так как слова стоят в алфавитном порядке, то первая треть (81 шт) начинаются с «А», вторая треть (тоже 81) – с «О», а последняя треть – с «У», то есть первая буква меняется через 81 слово
2. аналогично:

* 2-я буква меняется через 81/3 = 27 слов;
* 3-я буква – через 27/3 = 9 слов;
* 4-я буква – через 9/3 = 3 слова и
* 5-я буква меняется в каждой строке.

1. из этой закономерности ясно, что
   * на первой позиции в искомом слове будет буква «У» (последние 81 букв);
   * на второй – тоже буква «У» (последние 27 букв);
   * на третьей – тоже буква «У» (последние 9 букв);
   * на четвертой – буква «О» (т.к. последние три буквы «У», а перед ними 3 буквы «О»)%
   * на пятой – буква «У» (т.к. последние 3 буквы чередуются «А», «О», «У», а перед ними такая же последовательность).
2. Ответ: УУУОУ.

### Еще пример задания (автор – В.В. Путилов):

*Все 5-буквенные слова, составленные из 5 букв А, К, Л, О, Ш, записаны в алфавитном порядке.*

*Вот начало списка:*

**1. ААААА**

**2. ААААК**

**3. ААААЛ**

**4. ААААО**

**5. ААААШ**

**6. АААКА**

**……**

*На каком месте от начала списка стоит слово ШКОЛА?*

**Решение:**

1. по аналогии с предыдущим решением будем использовать пятеричную систему счисления с заменой А → 0, К → 1, Л → 2, О → 3 и Ш → 4
2. слово ШКОЛА запишется в новом коде так: 413205
3. переводим это число в десятичную систему:

413205 = 4⋅54 + 1⋅53 + 3⋅52 + 2⋅51 = 2710

1. поскольку нумерация элементов списка начинается с 1, а числа в пятеричной системе – с нуля, к полученному результату нужно прибавить 1, тогда…
2. Ответ: 2711.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + нужно помнить, что список в задании начинается с 1, а числа в троичной системе – с нуля, поэтому для получения N-ой по счёту цепочки нужно переводить в троичную систему число N-1. |

### Еще пример задания:

*Все 5-буквенные слова, составленные из букв А, О, У, записаны в* ***обратном*** *алфавитном порядке. Вот начало списка:*

**1. УУУУУ**

**2. УУУУО**

**3. УУУУА**

**4. УУУОУ**

**……**

*Запишите слово, которое стоит на 240-м месте от начала списка.*

**Решение (2 способ, троичная система, идея М. Густокашина):**

1. по условию задачи важно только то, что используется набор из трех разных символов, для которых задан порядок (алфавитный); поэтому для вычислений можно использовать три любые символа, например, цифры 0, 1 и 2 (для них порядок очевиден – по возрастанию)
2. выпишем начало списка, заменив буквы на цифры так, чтобы **порядок символов был обратный алфавитный** (У → 0, О → 1, А → 2):

**1. 00000**

**2. 00001**

**3. 00002**

**4. 00010**

**……**

1. это напоминает (в самом деле, так оно и есть!) числа, записанные в троичной системе счисления в порядке возрастания: на первом месте стоит число 0, на втором – 1 и т.д.
2. тогда легко понять, что 240-м месте стоит число 239, записанное в троичной системе счисления
3. переведем 239 в троичную систему: 239 = 222123
4. заменяем обратно цифры на буквы, **учитывая обратный алфавитный порядок** (0 → У, 1 → О, 2 → А): 22212 → АААОА
5. Ответ: АААОА.

# 11 (базовый уровень, время – 5 мин)

**Тема**: рекурсивные алгоритмы.

**Что нужно знать**:

* рекурсия – это приём, позволяющий свести исходную задачу к одной или нескольким более простым задачам того же типа
* чтобы определить рекурсию, нужно задать
  + условие остановки рекурсии (базовый случай или несколько базовых случаев)
  + рекуррентную формулу
* любую рекурсивную процедуру можно запрограммировать с помощью цикла
* рекурсия позволяет заменить цикл и в некоторых сложных задачах делает решение более понятным, хотя часто менее эффективным
* существуют языки программирования, в которых рекурсия используется как один из основных приемов обработки данных (Lisp, Haskell)

### Пример задания:

*Дан рекурсивный алгоритм:*

**procedure F(n: integer);**

**begin**

**writeln(n);**

**if n < 5 then begin**

**F(n + 1);**

**F(n + 3)**

**end**

**end;**

*Найдите сумму чисел, которые будут выведены при вызове F(1).*

**Решение (вариант 1, построение дерева вызовов):**

1. поскольку в начале каждого вызова на экран выводится значение единственного параметра функции, достаточно определить порядок рекурсивных вызовов и сложить значения параметров
2. поскольку при  выполняется два рекурсивных вызова, решать такую задачу «на бумажке» удобно в виде двоичного дерева (в узлах записаны значения параметров при вызове функции):

1

2

4

5

3

4

5

7

6

5

7

1. складывая все эти числа, получаем 49
2. ответ: 49.

**Решение (вариант 2, подстановка):**

1. можно обойтись и без дерева, учитывая, что при каждом вызове с n < 5 происходит два рекурсивных вызова; сумму чисел, полученных при вызове , обозначим через :



1. выполняем вычисления:



1. теперь остаётся вычислить ответ «обратным ходом»:
2. 
3. Ответ: 49.

### Ещё пример задания:

*Дан рекурсивный алгоритм:*

**procedure F(n: integer);**

**begin**

**writeln(n);**

**if n < 6 then begin**

**F(n+2);**

**F(n\*3)**

**end**

**end;**

*Найдите сумму чисел, которые будут выведены при вызове F(1).*

**Решение (вариант 1, метод подстановки):**

1. сначала определим рекуррентную формулу; обозначим через G(n) сумму чисел, которая выводится при вызове F(n)
2. при n >= 6 процедура выводит число n и заканчивает работу без рекурсивных вызовов:

G(n) = n при n >= 6

1. при n < 6 процедура выводит число n и дважды вызывает сама себя:

G(n) = n + G(n+2) + G(3n) при n < 6

1. в результате вызова F(1) получаем

G(1) = 1 + G(3) + G(3)

G(3) = 3 + G(5) + G(9) = 3 + G(5) + 9

G(5) = 5 + G(7) + G(15) = 5 + 7 + 15 = 27

1. используем обратную подстановку:

G(3) = 3 + G(5) + 9 = 3 + 27 + 9 = 39

G(1) = 1 + 2\*G(3) = 79

1. Ответ: 79.

**Решение (вариант 2, динамическое программирование):**

1. п. 1-3 такие же, как в первом варианте решения
2. заполняем таблицу G(n) при n >= 6 (где G(n) = n)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| n | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| G(n) |  |  |  |  |  | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

1. остальные ячейки заполняем, начиная с n = 5 справа налево, используя формулу :

G(n) = n + G(n+2) + G(3n)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| n | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| G(n) | 79 | 30 | 39 | 22 | 27 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

1. ответ читаем в самой левой ячейке
2. Ответ: 79.

### Пример задания:

*Дан рекурсивный алгоритм:*

**procedure F(n: integer);**

**begin**

**writeln('\*');**

**if n > 0 then begin**

**F(n-2);**

**F(n div 2)**

**end**

**end;**

*Сколько символов "звездочка" будет напечатано на экране при выполнении вызова F(7)?*

**Решение (вариант 1, составление полной таблицы):**

1. сначала определим рекуррентную формулу; обозначим через G(n) количество звездочек, которые выводит программа при вызове F(n)
2. из программы видим, что

G(n) = 1 при всех n <= 0

G(n) = 1 + G(n-2) + G(n div 2) при n > 0

1. вспомним, что **n div 2** – это частное от деления n на 2
2. по этим формулам заполняем таблицу, начиная с нуля:

G(0) = 1

G(1) = 1 + G(-1) + G(0) = 1 + 1 + 1 = 3

G(2) = 1 + G(0) + G(1) = 1 + 1 + 3 = 5

G(3) = 1 + G(1) + G(1) = 1 + 3 + 3 = 7

G(4) = 1 + G(2) + G(2) = 1 + 5 + 5 = 11

G(5) = 1 + G(3) + G(2) = 1 + 7 + 5 = 13

G(6) = 1 + G(4) + G(3) = 1 + 11 + 7 = 19

G(7) = 1 + G(5) + G(3) = 1 + 13 + 7 = 21

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| n | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| G(n) | 1 | 3 | 5 | 7 | 11 | 13 | 19 | 21 |

1. Ответ: 21.

**Решение (вариант 2, «с конца»):**

1. пп. 1-3 – как в варианте 1
2. по формулам G(7) = 1+ G(5) + G(3), поэтому нужно найти G(5) и G(3)
3. G(5) = 1 + G(3) + G(2), нужны G(3) и G(2)
4. G(3) = 1 + G(1) + G(1), нужно G(1)
5. G(2) = 1 + G(0) + G(1) = 2 + G(1), нужно G(1)
6. G(1) = 1 + G(-1) + G(0) = 1 + 1 + 1 = 3
7. теперь идем «обратным ходом»:

G(2) = 2 + G(1) = 5

G(3) = 1 + G(1) + G(1) = 1 + 3 + 3 = 7

G(5) = 1 + G(3) + G(2) = 1 + 7 + 5 = 13

G(7) = 1 + G(5) + G(3) = 1 + 13 + 7 = 21

1. Ответ: 21.

### Пример задания:

*Алгоритм вычисления значений функций F(n) и G(n), где n – натуральное число, задан следующими соотношениями:*

*F(1) = 1; G(1) = 1;*

*F(n) = F(n – 1) – G(n – 1),*

*G(n) = F(n–1) + G(n – 1), при n >=2*

*Чему равно значение величины F(5)/G(5)?*

*В ответе запишите только целое число.*

**Решение:**

1. фактически рекуррентная формула задана для пары (F(n); G(n))
2. замечаем, что F(n) – это разность предыдущей пары, а G(n) – сумма тех же значений
3. заполняем таблицу, начиная с известной первой пары

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| n | 1 | 2 | 3 | 4 | 5 |
| F(n) | 1 | 0 | –2 | –4 | –4 |
| G(n) | 1 | 2 | 2 | 0 | –4 |

1. искомое значение F(5)/G(5) равно 1
2. ответ: 1.

### Ещё пример задания:

*Алгоритм вычисления значения функции F(n), где n – натуральное число,*

*задан следующими соотношениями:*

**F(1) = 1**

**F(n) = F(n–1) \* n, при n > 1**

*Чему равно значение функции F(5)?*

*В ответе запишите только целое число.*

**Решение:**

1. используя заданную рекуррентную формулу, находим, что

F(5) = F(4) \* 5

1. применив формулу еще несколько раз, получаем

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

1. мы дошли до базового случая, который останавливает рекурсию, так как определяет значение F(1) = 1
2. окончательно F(5) = 1 \* 2 \* 3 \* 4 \* 5 = 120
3. ответ: 120.

### Ещё пример задания:

*Процедура F(n), где n – натуральное число, задана следующим образом (язык Паскаль):*

**procedure F(n: integer);**

**begin**

**if n < 3 then**

**write('\*')**

**else begin**

**F(n-1);**

**F(n-2);**

**F(n-2)**

**end;**

**end;**

*Сколько звездочек напечатает эта процедура при вызове F(6)? В ответе запишите только целое число.*

**Решение:**

1. эта задача по сути такая же, как и предыдущая, но «завёрнута» в другой фантик: для **n < 3** (то есть, для 1 и 2) функция выводит одну звездочку

F(1) = F(2) = 1

а для бóльших n имеем рекуррентную формулу

F(n) = F(n-1) + F(n-2) + F(n-2)

= F(n-1) + 2\*F(n-2)

1. запишем в таблицу базовые случаи

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **n** | **1** | **2** | **3** | **4** | **5** | **6** |
| **F(n)** | **1** | **1** |  |  |  |  |

1. заполняем таблицу, используя рекуррентную формулу:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **n** | **1** | **2** | **3** | **4** | **5** | **6** |
| **F(n)** | **1** | **1** | **3** | **5** | **11** | **21** |

F(3) = F(2) + 2\*F(1) = 3

F(4) = F(3) + 2\*F(2) = 5

F(5) = F(4) + 2\*F(3) = 11

F(6) = F(5) + 2\*F(4) = 21

1. ответ: 21.

# 11 (базовый уровень, время – 5 мин)

**Тема**: рекурсивные алгоритмы.

**Что нужно знать**:

* рекурсия – это приём, позволяющий свести исходную задачу к одной или нескольким более простым задачам того же типа
* чтобы определить рекурсию, нужно задать
  + условие остановки рекурсии (базовый случай или несколько базовых случаев)
  + рекуррентную формулу
* любую рекурсивную процедуру можно запрограммировать с помощью цикла
* рекурсия позволяет заменить цикл и в некоторых сложных задачах делает решение более понятным, хотя часто менее эффективным
* существуют языки программирования, в которых рекурсия используется как один из основных приемов обработки данных (Lisp, Haskell)

### Пример задания:

*Дан рекурсивный алгоритм:*

**void F(int n)**

**{**

**printf(″%d\n″,n);**

**if (n < 5){**

**F(n + 1);**

**F(n + 3);**

**}**

**}**

*Найдите сумму чисел, которые будут выведены при вызове F(1).*

**Решение (вариант 1, построение дерева вызовов):**

1. поскольку в начале каждого вызова на экран выводится значение единственного параметра функции, достаточно определить порядок рекурсивных вызовов и сложить значения параметров
2. поскольку при  выполняется два рекурсивных вызова, решать такую задачу ″на бумажке″ удобно в виде двоичного дерева (в узлах записаны значения параметров при вызове функции):

1

2

4

5

3

4

5

7

6

5

7

1. складывая все эти числа, получаем 49
2. ответ: 49.

**Решение (вариант 2, подстановка):**

1. можно обойтись и без дерева, учитывая, что при каждом вызове с n < 5 происходит два рекурсивных вызова; сумму чисел, полученных при вызове , обозначим через :



1. выполняем вычисления:



1. теперь остаётся вычислить ответ ″обратным ходом″:
2. 
3. Ответ: 49.

### Ещё пример задания:

*Дан рекурсивный алгоритм:*

**void F(int n)**

**{**

**printf(″%d\n″,n);**

**if (n < 6 ){**

**F(n+2);**

**F(n\*3);**

**}**

**}**

*Найдите сумму чисел, которые будут выведены при вызове F(1).*

**Решение (вариант 1, метод подстановки):**

1. сначала определим рекуррентную формулу; обозначим через G(n) сумму чисел, которая выводится при вызове F(n)
2. при n >= 6 процедура выводит число n и заканчивает работу без рекурсивных вызовов:

G(n) = n при n >= 6

1. при n < 6 процедура выводит число n и дважды вызывает сама себя:

G(n) = n + G(n+2) + G(3n) при n < 6

1. в результате вызова F(1) получаем

G(1) = 1 + G(3) + G(3)

G(3) = 3 + G(5) + G(9) = 3 + G(5) + 9

G(5) = 5 + G(7) + G(15) = 5 + 7 + 15 = 27

1. используем обратную подстановку:

G(3) = 3 + G(5) + 9 = 3 + 27 + 9 = 39

G(1) = 1 + 2\*G(3) = 79

1. Ответ: 79.

**Решение (вариант 2, динамическое программирование):**

1. п. 1-3 такие же, как в первом варианте решения
2. заполняем таблицу G(n) при n >= 6 (где G(n) = n)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| n | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| G(n) |  |  |  |  |  | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

1. остальные ячейки заполняем, начиная с n = 5 справа налево, используя формулу :

G(n) = n + G(n+2) + G(3n)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| n | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| G(n) | 79 | 30 | 39 | 22 | 27 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

1. ответ читаем в самой левой ячейке
2. Ответ: 79.

### Пример задания:

*Дан рекурсивный алгоритм:*

**void F(int n)**

**{**

**printf(″\*″);**

**if (n > 0) {**

**F(n-2);**

**F(n / 2);**

**}**

**}**

*Сколько символов "звездочка" будет напечатано на экране при выполнении вызова F(7)?*

**Решение (вариант 1, составление полной таблицы):**

1. сначала определим рекуррентную формулу; обозначим через G(n) количество звездочек, которые выводит программа при вызове F(n)
2. из программы видим, что

G(n) = 1 при всех n <= 0

G(n) = 1 + G(n-2) + G(n / 2) при n > 0

1. вспомним, что **n / 2** – это частное от деления n на 2
2. по этим формулам заполняем таблицу, начиная с нуля:

G(0) = 1

G(1) = 1 + G(-1) + G(0) = 1 + 1 + 1 = 3

G(2) = 1 + G(0) + G(1) = 1 + 1 + 3 = 5

G(3) = 1 + G(1) + G(1) = 1 + 3 + 3 = 7

G(4) = 1 + G(2) + G(2) = 1 + 5 + 5 = 11

G(5) = 1 + G(3) + G(2) = 1 + 7 + 5 = 13

G(6) = 1 + G(4) + G(3) = 1 + 11 + 7 = 19

G(7) = 1 + G(5) + G(3) = 1 + 13 + 7 = 21

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| n | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| G(n) | 1 | 3 | 5 | 7 | 11 | 13 | 19 | 21 |

1. Ответ: 21.

**Решение (вариант 2, ″с конца″):**

1. пп. 1-3 – как в варианте 1
2. по формулам G(7) = 1+ G(5) + G(3), поэтому нужно найти G(5) и G(3)
3. G(5) = 1 + G(3) + G(2), нужны G(3) и G(2)
4. G(3) = 1 + G(1) + G(1), нужно G(1)
5. G(2) = 1 + G(0) + G(1) = 2 + G(1), нужно G(1)
6. G(1) = 1 + G(-1) + G(0) = 1 + 1 + 1 = 3
7. теперь идем ″обратным ходом″:

G(2) = 2 + G(1) = 5

G(3) = 1 + G(1) + G(1) = 1 + 3 + 3 = 7

G(5) = 1 + G(3) + G(2) = 1 + 7 + 5 = 13

G(7) = 1 + G(5) + G(3) = 1 + 13 + 7 = 21

1. Ответ: 21.

### Пример задания:

*Алгоритм вычисления значений функций F(n) и G(n), где n – натуральное число, задан следующими соотношениями:*

*F(1) = 1; G(1) = 1;*

*F(n) = F(n – 1) – G(n – 1),*

*G(n) = F(n–1) + G(n – 1), при n >=2*

*Чему равно значение величины F(5)/G(5)?*

*В ответе запишите только целое число.*

**Решение:**

1. фактически рекуррентная формула задана для пары (F(n); G(n))
2. замечаем, что F(n) – это разность предыдущей пары, а G(n) – сумма тех же значений
3. заполняем таблицу, начиная с известной первой пары

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| n | 1 | 2 | 3 | 4 | 5 |
| F(n) | 1 | 0 | –2 | –4 | –4 |
| G(n) | 1 | 2 | 2 | 0 | –4 |

1. искомое значение F(5)/G(5) равно 1
2. ответ: 1.

### Ещё пример задания:

*Алгоритм вычисления значения функции F(n), где n – натуральное число,*

*задан следующими соотношениями:*

**F(1) = 1**

**F(n) = F(n–1) \* n, при n > 1**

*Чему равно значение функции F(5)?*

*В ответе запишите только целое число.*

**Решение:**

1. используя заданную рекуррентную формулу, находим, что

F(5) = F(4) \* 5

1. применив формулу еще несколько раз, получаем

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

1. мы дошли до базового случая, который останавливает рекурсию, так как определяет значение F(1) = 1
2. окончательно F(5) = 1 \* 2 \* 3 \* 4 \* 5 = 120
3. ответ: 120.

### Ещё пример задания:

*Процедура F(n), где n – натуральное число, задана следующим образом (язык Паскаль):*

**void F(int n)**

**{**

**if (n < 3 )**

**printf(″\*″);**

**else {**

**F(n-1);**

**F(n-2);**

**F(n-2);**

**}**

**}**

*Сколько звездочек напечатает эта процедура при вызове F(6)? В ответе запишите только целое число.*

**Решение:**

1. эта задача по сути такая же, как и предыдущая, но ″завёрнута″ в другой фантик: для **n < 3** (то есть, для 1 и 2) функция выводит одну звездочку

F(1) = F(2) = 1

а для бóльших n имеем рекуррентную формулу

F(n) = F(n-1) + F(n-2) + F(n-2)

= F(n-1) + 2\*F(n-2)

1. запишем в таблицу базовые случаи

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **n** | **1** | **2** | **3** | **4** | **5** | **6** |
| **F(n)** | **1** | **1** |  |  |  |  |

1. заполняем таблицу, используя рекуррентную формулу:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **n** | **1** | **2** | **3** | **4** | **5** | **6** |
| **F(n)** | **1** | **1** | **3** | **5** | **11** | **21** |

F(3) = F(2) + 2\*F(1) = 3

F(4) = F(3) + 2\*F(2) = 5

F(5) = F(4) + 2\*F(3) = 11

F(6) = F(5) + 2\*F(4) = 21

1. ответ: 21.

# 12 (базовый уровень, время – 2 мин)

**Тема**: Компьютерные сети. Адресация в Интернете.

**Что нужно знать**:

* каждый компьютер, подключенный к сети Интернет, должен иметь собственный адрес, который называют IP-адресом (IP = *Internet Protocol*)
* IP-адрес состоит из четырех чисел, разделенных точками; каждое из этих чисел находится в интервале 0…255, например: **192.168.85.210**
* адрес документа в Интернете (URL = *Uniform Resource Locator*) состоит из следующих частей:
  + протокол, чаще всего **http** (для Web-страниц) или **ftp** (для файловых архивов)
  + знаки **://**, отделяющие протокол от остальной части адреса
  + доменное имя (или IP-адрес) сайта
  + каталог на сервере, где находится файл
  + имя файла
* принято разделять каталоги не обратным слэшем «\» (как в *Windows*), а прямым «/», как в системе *UNIX* и ее «родственниках», например, в *Linux*
* пример адреса (URL)

http://www.vasya.ru/home/user/vasya/qu-qu.zip

здесь желтым маркером выделен протокол, фиолетовым – доменное имя сайта, голубым – каталог на сайте и серым – имя файла

### Пример задания:

*В терминологии сетей TCP/IP маска сети – это двоичное число, меньшее* 232*; в маске сначала (в старших разрядах) стоят единицы, а затем с некоторого места нули. Маска определяет, какая часть IP-адреса узла сети относится к адресу сети, а какая – к адресу самого узла в этой сети. Обычно маска записывается по тем же правилам, что и IP-адрес – в виде четырёх байт, причём каждый байт записывается в виде десятичного числа. Адрес сети получается в результате применения поразрядной конъюнкции к заданному IP-адресу узла и маске.*

*Например, если IP-адрес узла равен 231.32.255.131, а маска равна 255.255.240.0, то адрес сети равен 231.32. 240.0.*

*Для узла с IP-адресом 224.128.112.142 адрес сети равен 224.128.64.0. Чему равен третий слева байт маски? Ответ запишите в виде десятичного числа.*

**Решение:**

1. вспомним, что в маске сначала стоят все единицы (они выделяют часть IP-адреса, которая соответствует номеру подсети), а затем – все нули (они соответствуют части, в которой записан номер компьютера)
2. для того, чтобы получить номер подсети, нужно выполнить поразрядную логическую операцию «И» между маской и IP-адресом (конечно, их нужно сначала перевести в двоичную систему счисления)

IP-адрес: 224.128.112.142 = 11100000.10000000.01110000.10001110

Маска: ???.???.???.??? = ????????.????????.????????.????????

Подсеть: 224.128. 64. 0 = 11100000.10000000.01000000.00000000

1. Биты, которые выделены жёлтым фоном, изменились (обнулились!), для этого соответствующие биты маски должны быть равны нулю (помним, что X и 1 = X, а X и 0 = 0)
2. С другой стороны, слева от самого крайнего выделенного бита стоит 1, поэтому этот бит в маске должен быть равен 1
3. Поскольку в маске сначала идет все единицы, а потом все нули, маска готова, остаётся перевести все числа из двоичной системы в десятичную:

Подсеть: 224.128. 64. 0 = 11100000.10000000.01000000.00000000

Маска: 255.255.192.000 = 11111111.11111111.11000000.00000000

1. Нам нужно только третье число, оно равно 192 (кстати, первое и второе всегда равны 255).
2. Ответ: 192.

### Ещё пример задания:

*В терминологии сетей TCP/IP маской сети называют двоичное число, которое показывает, какая часть IP-адреса узла сети относится к адресу сети, а какая – к адресу узла в этой сети. Адрес сети получается в результате применения поразрядной конъюнкции к заданному адресу узла и его маске. По заданным IP-адресу узла сети и маске определите адрес сети:*

*IP-адрес: 217.8.244.3 Маска: 255.255.252.0*

*При записи ответа выберите из приведенных в таблице чисел 4 фрагмента четыре элемента IP-адреса и запишите в нужном порядке соответствующие им буквы без точек.*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | F | G | H |
| 0 | 3 | 8 | 217 | 224 | 244 | 252 | 255 |

*Пример. Пусть искомый адрес сети 192.168.128.0 и дана таблица*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | F | G | H |
| 128 | 168 | 255 | 8 | 127 | 0 | 17 | 192 |

*В этом случае правильный ответ будет HBAF.*

**Решение (1 способ, логическое «И» маски и номера узла):**

1. нужно помнить, что каждая часть в IP-адресе (и в маске) – восьмибитное двоичное число, то есть десятичное число от 0 до 255 (поэтому каждую часть адреса и маски называют *октетом*)
2. поскольку 255 = 111111112, все части IP-адреса узла, для которых маска равна 255, входят в IP-адрес сети без изменений (они полностью относятся к номеру сети)
3. поскольку 0 = 000000002, все части IP-адреса узла, для которых маска равна 0, в IP-адресе сети заменяются нулями (они полностью относятся к номеру узла в сети)
4. таким образом, мы почти определили адрес сети, он равен 217.8.X.0, где X придется определять дополнительно
5. переведем в двоичную систему третью часть IP-адреса и маски

Адрес: 244 = 111101002

Маска: 252 = 111111002

1. заметим, что в маске сначала идет цепочка единиц, а потом до конца – цепочка нулей; это правильно, число где цепочка единиц начинается не с левого края (не со старшего, 8-ого бита) или внутри встречаются нули, не может быть маской; поэтому есть всего несколько допустимых чисел для последней части маски (все предыдущие должны быть равны 255):

100000002 = 128

110000002 = 192

111000002 = 224

111100002 = 240

111110002 = 248

111111002 = 252

111111102 = 254

111111112 = 255

1. выполним между этими числами поразрядную конъюнкцию – логическую операцию «И»; маска 252 = 111111002 говорит о том, что первые 6 битов соответствующего числа в IP-адресе относятся к номеру сети, а оставшиеся 2 – к адресу узла:

244 = **111101**002

252 = **111111**002

поэтому часть номера сети – это 244 = 111101002.

1. таким образом, полный адрес сети – 217.8.244.0
2. по таблице находим ответ: DCFA (D=217, C=8, F=244, A=0)

### Ещё пример задания:

*В терминологии сетей TCP/IP маской сети называют двоичное число, которое показывает, какая часть IP-адреса узла сети относится к адресу сети, а какая – к адресу узла в этой сети. Адрес сети получается в результате применения поразрядной конъюнкции к заданному адресу узла и его маске. По заданным IP-адресу узла сети и маске определите адрес сети:*

*IP-адрес: 10.8.248.131 Маска: 255.255.224.0*

*При записи ответа выберите из приведенных в таблице чисел 4 фрагмента четыре элемента IP-адреса и запишите в нужном порядке соответствующие им буквы без точек.*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | F | G | H |
| 8 | 131 | 255 | 224 | 0 | 10 | 248 | 92 |

*Пример. Пусть искомый адрес сети 192.168.128.0 и дана таблица*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | F | G | H |
| 128 | 168 | 255 | 8 | 127 | 0 | 17 | 192 |

*В этом случае правильный ответ будет HBAF.*

**Решение (1 способ, логическое «И» маски и номера узла):**

1. нужно помнить, что каждая часть в IP-адресе (и в маске) – восьмибитное двоичное число, то есть десятичное число от 0 до 255 (поэтому каждую часть адреса и маски называют *октетом*)
2. поскольку 255 = 111111112, все части IP-адреса узла, для которых маска равна 255, входят в IP-адрес сети без изменений (они полностью относятся к номеру сети)
3. поскольку 0 = 000000002, все части IP-адреса узла, для которых маска равна 0, в IP-адресе сети заменяются нулями (они полностью относятся к номеру узла в сети)
4. таким образом, мы почти определили адрес сети, он равен 10.8.X.0, где X придется определять дополнительно
5. переведем в двоичную систему третью часть IP-адреса и маски

248 = 111110002

224 = 111000002

1. заметим, что в маске сначала идет цепочка единиц, а потом до конца – цепочка нулей; это правильно, число где цепочка единиц начинается не с левого края (не со старшего, 8-ого бита) или внутри встречаются нули, не может быть маской; поэтому есть всего несколько допустимых чисел для последней части маски (все предыдущие должны быть равны 255):

100000002 = 128

110000002 = 192

111000002 = 224

111100002 = 240

111110002 = 248

111111002 = 252

111111102 = 254

111111112 = 255

1. выполним между этими числами поразрядную конъюнкцию – логическую операцию «И»; маска 224 = 111000002 говорит о том, что первые три бита соответствующего числа в IP-адресе относятся к номеру сети, а оставшиеся 5 – к адресу узла:

248 = **111**110002

224 = **111**000002

поэтому часть номера сети – это 224 = 111000002, а номер узла – это 110002 = 24.

1. таким образом, полный адрес сети – 10.8.224.0
2. по таблице находим ответ: FADE (F=10, A=8, D=224, E=0)

**Решение (2 способ, использование размера подсети, М. Савоськин):**

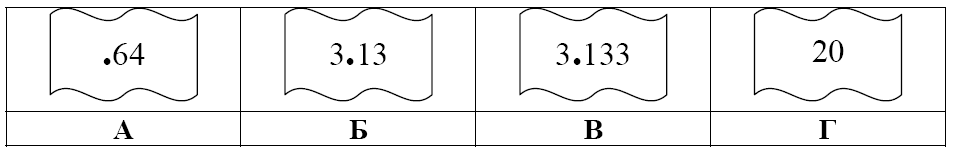
1. п. 1-4 – так же, как и в способе 1; в результате находим, что адрес сети имеет вид 10.8.X.0
2. третье число в маске (соответствующее неизвестному X) – 224; в такую подсеть входят адреса, в которых третий октет (третье число IP-адреса) может принимать 256 – 224 = 32 разных значений
3. выпишем адреса, принадлежащие всем возможным подсетям такого вида (третий октет изменяется от 0 с шагом 32):

|  |  |
| --- | --- |
| Начальный IP-адрес (адрес сети) | Конечный IP-адрес (широковещательный) |
| 10.8.0.0 | 10.8.31.255 |
| 10.8.32.0 | 10.8.63. 255 |
| 10.8.64.0 | 10.8.95. 255 |
| 10.8.96.0 | 10.8.127. 255 |
| 10.8.128.0 | 10.8.159. 255 |
| 10.8.160.0 | 10.8.191. 255 |
| 10.8.192.0 | 10.8.223. 255 |
| 10.8.224.0 | 10.8.255. 255 |

1. смотрим, что нужный нам адрес 10.8.248.131 оказывается в подсети с адресом 10.8.224.0; в данном случае можно было быстрее получить ответ, если бы мы строили таблицу с конца, т.е. с последней подсети
2. по таблице находим ответ: FADE (F=10, A=8, D=224, E=0)

### Ещё пример задания:

*Петя записал IP-адрес школьного сервера на листке бумаги и положил его в карман куртки. Петина мама случайно постирала куртку вместе с запиской. После стирки Петя обнаружил в кармане четыре обрывка с фрагментами IP-адреса. Эти фрагменты обозначены буквами А, Б, В и Г. Восстановите IP-адрес. В ответе укажите последовательность букв, обозначающих фрагменты, в порядке, соответствующем IP-адресу.*



**Решение:**

1. самое главное – вспомнить, что каждое из 4-х чисел в IP-адресе должно быть в интервале от 0 до 255
2. поэтому сразу определяем, что фрагмент А – самый последний, так как в противном случае одно из чисел получается больше 255 (643 или 6420)
3. фрагмент Г (число 20) может быть только первым, поскольку варианты 3.1320 и 3.13320 дают число, большее 255
4. из фрагментов Б и В первым должен быть Б, иначе получим 3.1333.13 (1333 > 255)
5. таким образом, верный ответ – ГБВА.

|  |
| --- |
| **Возможные проблемы**:   * + если забыть про допустимый диапазон 0..255, то может быть несколько «решений» (все, кроме одного – неправильные) |

### Еще пример задания:

|  |  |
| --- | --- |
| A | / |
| Б | com |
| В | .edu |
| Г | :// |
| Д | .net |
| Е | htm |
| Ж | ftp |

*Доступ к файлу* **htm.net***, находящемуся на сервере* **com.edu***, осуществляется по протоколу* **ftp***. В таблице фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующую адрес указанного файла в сети Интернет.*

**Решение:**

1. адрес файла начинается с протокола, после этого ставятся знаки «**://**», имя сервера, каталог и имя файла
2. каталог здесь не указан, поэтому сразу получаем

**ftp://com.edu/htm.net**

1. такой адрес можно собрать из приведенных в таблице «кусков»

**ftp://com.edu/htm.net**

1. таким образом, верный ответ – ЖГБВАЕД.

|  |
| --- |
| **Возможные проблемы**:   * + существуют домены первого уровня **com** и **net**, а здесь **com** – это домен второго уровня, а **net** – расширение имени файла, все это сделано специально, чтобы запутать отвечающего   + **htm** – это обычно расширение файла (*Web-*страницы), а здесь оно используется как первая часть имени файла   + поскольку в ответе требуется написать не адрес файла, а последовательность букв, есть риск ошибиться при таком кодировании |

### Еще пример задания:

*Маской подсети называется 32-разрядное двоичное число, которое определяет, какая часть IP-адреса компьютера относится к адресу сети, а какая часть IP-адреса определяет адрес компьютера в подсети. В маске подсети старшие биты, отведенные в IP-адресе компьютера для адреса сети, имеют значение 1;младшие биты, отведенные в IP-адресе компьютера для адреса компьютера в подсети, имеют значение 0.Например,маска подсети может иметь вид:*

*11111111 11111111 11100000 00000000 (255.255.224.0)*

*Это значит, что 19 старших бит в IP-адресе содержит адрес сети, оставшиеся 13 младших бит содержат адрес компьютера в сети. Если маска подсети 255.255.255.240 и IP-адрес компьютера в сети 162.198.0.44, то порядковый номер компьютера в сети равен\_\_\_\_\_*

**Решение (1 способ):**

1. эта задача аналогична предыдущей с той разницей, что требуется определить не номер сети, а номер компьютера (узла) в этой сети
2. нужно помнить, что каждая часть в IP-адресе (и в маске) – восьмибитное двоичное число, то есть десятичное число от 0 до 255 (поэтому каждую часть адреса и маски называют *октетом*)
3. первые три числа в маске равны 255, в двоичной системе это 8 единиц, поэтому первые три числа IP-адреса компьютера целиком относятся к номеру сети
4. для последнего числа (октета) маска и соответствующая ей последняя часть IP-адреса равны

**240 = 111100002**

**44 = 001011002**

1. выше голубым цветом выделены нулевые биты маски и соответствующие им биты IP-адреса, определяющие номер компьютера в сети: 11002 = 12
2. Ответ: 12.

**Решение (2 способ, использование размера подсети, М. Савоськин):**

1. п. 1-3 – так же, как и в способе 1;
2. последнее число в маске – 240; в такую подсеть входят адреса, в которых четвертый октет может принимать 256 – 240 = 16 разных значений
3. выпишем адреса, принадлежащие всем возможным подсетям такого вида (четвертый октет изменяется от 0 с шагом 16):

|  |  |
| --- | --- |
| Начальный IP-адрес (адрес сети) | Конечный IP-адрес (широковещательный) |
| 162.198.0.0 | 162.198.0.15 |
| 162.198.0.16 | 162.198.0.31 |
| 162.198.0.32 | 162.198.0.47 |
| ... |  |

1. смотрим, что нужный нам адрес 162.198.0.44 оказывается в подсети с адресом 162.198.0.32
2. номер компьютера 162.198.0.44 в сети 162.198.0.32 находим как 44 – 32 = 12
3. таким образом, ответ: 12

### Еще пример задания:

*Маской подсети называется 32-разрядное двоичное число, которое определяет, какая часть IP-адреса компьютера относится к адресу сети, а какая часть IP-адреса определяет адрес компьютера в подсети. В маске подсети старшие биты, отведенные в IP-адресе компьютера для адреса сети, имеют значение 1;младшие биты, отведенные в IP-адресе компьютера для адреса компьютера в подсети, имеют значение 0.Например, маска подсети может иметь вид:*

*11111111 11111111 11100000 00000000 (255.255.224.0)*

*Это значит, что 19 старших бит в IP-адресе содержит адрес сети, оставшиеся 13 младших бит содержат адрес компьютера в сети. Если маска подсети 255.255.240.0 и IP-адрес компьютера в сети 162.198.75.44, то порядковый номер компьютера в сети равен\_\_\_\_\_*

**Решение (1 способ):**

1. первые два числа в маске равны 255, в двоичной системе это 8 единиц, поэтому первые два числа IP-адреса компьютера целиком относятся к номеру сети и про них (в этой задаче) можно забыть
2. последнее число в маске – 0, поэтому последнее число IP-адреса целиком относится к номеру узла
3. третье число маски – **240** = **111100002**, это значит, что первые 4 бита третьей части адреса (75) относятся к адресу сети, а последние 4 бита – к номеру узла:

**240 = 111100002**

**75 = 010010112**

1. выше голубым цветом выделены нулевые биты маски и соответствующие им биты IP-адреса, определяющие старшую часть номера компьютера в сети: 10112 = 11
2. кроме того, нужно учесть еще и последнее число IP-адреса (44 = 001011002), таким образом, полный номер компьютера (узла) в двоичной и десятичной системах имеет вид

**1011.001011002 = 11.44**

1. для получения полного номера узла нужно перевести число **1011001011002** в десятичную систему: **1011001011002** **=** **2860** или, что значительно удобнее, выполнить все вычисления в десятичной системе: первое число в полученном двухкомпонентном адресе 11.44 умножается на 28 = 256 (сдвигается на 8 битов влево), а второе просто добавляется к сумме:

11·256 + 44 = 2860

1. Ответ: 2860.

**Решение (2 способ, использование размера подсети, М. Савоськин):**

1. п. 1-2 – так же, как и в способе 1;
2. третье число в маске (соответствующее неизвестному X) – 240; в такую подсеть входят адреса, в которых третий октет (третье число IP-адреса) может принимать 256 – 240 = 16 разных значений
3. выпишем адреса, принадлежащие всем возможным подсетям такого вида (третий октет изменяется от 0 с шагом 32):

|  |  |
| --- | --- |
| Начальный IP-адрес (адрес сети) | Конечный IP-адрес (широковещательный) |
| 162.198.0. 0 | 162.198.15.255 |
| 162.198.16. 0 | 162.198.31.255 |
| 162.198.32. 0 | 162.198.47.255 |
| 162.198.48. 0 | 162.198.63.255 |
| 162.198.64. 0 | 162.198.79.255 |
| ... |  |

1. смотрим, что нужный нам адрес 162.198.75.44 оказывается в сети с адресом 162.198.64.0
2. номер компьютера 162.198.75.44 в сети 162.198.64.0 находим как

256\*(75 – 64) + 44 = 2860

1. таким образом, ответ: 2860

### Еще пример задания:

*В терминологии сетей TCP/IP маской подсети называется 32-разрядное двоичное число, определяющее, какие именно разряды IP-адреса компьютера являются общими для всей подсети - в этих разрядах маски стоит 1. Обычно маски записываются в виде четверки десятичных чисел - по тем же правилам, что и IP-адреса.*

*Для некоторой подсети используется маска 255.255.252.0. Сколько различных адресов компьютеров допускает эта маска?*

*Примечание. На практике два из возможных адресов не используются для адресации узлов сети: адрес сети, в котором все биты, отсекаемые маской, равны 0, и широковещательный адрес, в котором все эти биты равны 1.*

**Решение (1 способ):**

1. фактически тут нужно найти какое количество N бит в маске нулевое, и тогда количество вариантов, которые можно закодировать с помощью N бит равно 2N
2. каждая часть IP-адреса (всего 4 части) занимает 8 бит
3. поскольку младшая часть маски 255.255.252.0 нулевая, 8 бит уже свободны
4. третья часть маски 252 = 255 – 3 = 111111002 содержит 2 нулевых бита
5. общее число нулевых битов N = 10, число свободных адресов 2NNN= 1024
6. поскольку из них 2 адреса не используются (адрес сети и широковещательный адрес) для узлов сети остается 1024 – 2 = 1022 адреса
7. Ответ: 1022.

**Решение (2 способ, использование размера подсети, М. Савоськин):**

1. найдём количество адресов соответствующих маске 255.255.252.0:

256\*(256 – 252) = 1024

1. поскольку из них 2 адреса не используются (адрес сети и широковещательный адрес) для узлов сети остается 1024 – 2 = 1022 адреса
2. Ответ: 1022.

# 13 (повышенный уровень, время – 3 мин)

**Тема**: Вычисление информационного объема сообщения.

**Что нужно знать**:

* с помощью *K* бит можно закодировать различных вариантов (чисел)
* таблица степеней двойки, она же показывает, сколько вариантов *Q* можно закодировать с помощью *K*  бит:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *K*, бит | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| *Q*, вариантов | 2 | 4 | 8 | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |

* при измерении количества информации принимается, что в одном байте 8 бит, а в одном килобайте (1 Кбайт) – 1024 байта, в мегабайте (1 Мбайт) – 1024 Кбайта[[9]](#footnote-9)
* чтобы найти информационный объем сообщения (текста) *I*, нужно умножить количество символов (отсчетов) *N* на число бит на символ (отсчет) *K*: 
* две строчки текста не могут занимать 100 Кбайт в памяти
* мощность алфавита *M*  – это количество символов в этом алфавите
* если алфавит имеет мощность *M*, то количество всех возможных «слов» (символьных цепочек) длиной *N* (без учета смысла) равно ; для двоичного кодирования (мощность алфавита *M* – 2 символа) получаем известную формулу: 

### Пример задания:

*При регистрации в компьютерной системе каждому пользователю выдаётся пароль, состоящий из 15 символов и содержащий только символы Ш, К, О, Л, А (таким образом, используется 5 различных символов). Каждый такой пароль в компьютерной системе записывается минимально возможным и одинаковым целым количеством байт (при этом используют посимвольное кодирование и все символы кодируются одинаковым и минимально*

*возможным количеством бит). Укажите объём памяти в байтах, отводимый этой системой для записи 30 паролей. В ответе запишите только число, слово «байт» писать не нужно.*

**Решение**:

1. согласно условию, в пароле можно использовать 5 символов
2. для кодирования номера одного из 5 символов нужно выделить 3 бита памяти (они позволяют закодировать 23 = 8 вариантов)
3. для хранения всех 15 символов пароля нужно 15 ⋅ 3 = 45 бит
4. поскольку пароль должен занимать целое число байт, берем ближайшее большее (точнее, не меньшее) значение, которое кратно 8: это 48 = 6 ⋅ 8; то есть один пароль занимает 6 байт
5. тогда 30 паролей занимают 6 ⋅ 30 = 180 байт
6. ответ: 180.

### Ещё пример задания:

*Для регистрации на сайте некоторой страны пользователю требуется придумать пароль. Длина пароля – ровно 11 символов. В качестве символов используются десятичные цифры и 12 различных букв местного алфавита, причём все буквы используются в двух начертаниях: как строчные, так и заглавные (регистр буквы имеет значение!).*

*Под хранение каждого такого пароля на компьютере отводится минимально возможное и одинаковое целое количество байтов, при этом используется посимвольное кодирование и все символы кодируются одинаковым и минимально возможным количеством битов.*

*Определите объём памяти в байтах, который занимает хранение 60 паролей.*

**Решение**:

1. согласно условию, в пароле можно использовать 10 цифр (0..9) + 12 заглавных букв местного алфавита + 12 строчных букв, всего 10 + 12 + 12 = 34 символа
2. для кодирования номера одного из 34 символов нужно выделить 6 бит памяти (5 бит не хватает, они позволяют закодировать только 25 = 32 варианта)
3. для хранения всех 11 символов пароля нужно 11 ⋅ 6 = 66 бит
4. поскольку пароль должен занимать целое число байт, берем ближайшее большее (точнее, не меньшее) значение, которое кратно 8: это 72 = 9 ⋅ 8; то есть один пароль занимает 9 байт
5. тогда 60 паролей занимают 9 ⋅ 60 = 540 байт
6. ответ: 540.

|  |
| --- |
| **Возможные ловушки**:   * часто забывают, что пароль должен занимать ЦЕЛОЕ число байт |

### Ещё пример задания:

*В велокроссе участвуют 119 спортсменов. Специальное устройство регистрирует прохождение каждым из участников промежуточного финиша, записывая его номер с использованием минимально возможного количества бит, одинакового для каждого спортсмена. Каков информационный объем в битах сообщения, записанного устройством, после того как промежуточный финиш прошли 70 велосипедистов?*

**Решение**:

1. велосипедистов было 119, у них 119 разных номеров, то есть, нам нужно закодировать 119 вариантов
2. по таблице степеней двойки находим, что для этого нужно минимум 7 бит (при этом можно закодировать 128 вариантов, то есть, еще есть запас); итак, 7 бит на один отсчет
3. когда 70 велосипедистов прошли промежуточный финиш, в память устройства записано 70 отсчетов
4. поэтому в сообщении 70\*7 = 490 бит информации.

|  |
| --- |
| **Возможные ловушки**:   * дано число, которое есть в условии (неверные ответы 70 бит, 70 байт, 119 байт), чтобы сбить случайное угадывание * указано правильное число, но другие единицы измерения (мог быть вариант 490 байт) * расчет на невнимательное чтение условия: можно не заметить, что требуется определить объем только 70 отсчетов, а не всех 119 (мог быть вариант 119\*7=833 бита) |

### Еще пример задания:

*Объем сообщения, содержащего 4096 символов, равен 1/512 части Мбайта. Какова мощность алфавита, с помощью которого записано это сообщение?*

|  |
| --- |
| **Большие числа. Что делать?**  Обычно (хотя и не всегда) задачи, в условии которых даны большие числа, решаются достаточно просто, если выделить в этих числах степени двойки. На эту мысль должны сразу наталкивать такие числа как  128 = 27, 256 = 28, 512 = 29 , 1024 = 210,  2048 = 211, 4096 = 212 , 8192 = 213, 16384 = 214, 65536 = 216 и т.п.  Нужно помнить, что соотношение между единицами измерения количества информации также представляют собой степени двойки:  1 байт = 8 бит = 23 бит,  1 Кбайт = 1024 байта = 210 байта  = 210 · 23 бит = 213 бит,  1 Мбайт = 1024 Кбайта = 210 Кбайта  = 210 · 210 байта = 220 байта  = 220 · 23 бит = 223 бит.  Правила выполнения операций со степенями:   * при умножении степени при одинаковых основаниях складываются      * … а при делении – вычитаются: |

**Решение (вариант 1)**:

1. в сообщении было 4096 = 212 символов
2. объем сообщения

1/512 Мбайта = 223 / 512 бита = 223 / 29 бита = 214 бита (= 16384 бита!)

1. место, отведенное на 1 символ:

214 бита / 212 символов = 22 бита на символ = 4 бита на символ

1. 4 бита на символ позволяют закодировать 24 = 16 разных символов
2. поэтому мощность алфавита – 16 символов

|  |
| --- |
| **Возможные ловушки**:   * легко запутаться, если выполнять вычисления «в лоб», не через степени двойки |

**Решение (вариант 2, предложен В.Я. Лаздиным)**:

1. объем сообщения

1/512 Мбайт = 1024/512 Кбайт = 2 Кбайт = 2048 байт

1. на 1 символ приходится 2048 байт / 4096 = 1/2 байта = 4 бита
2. 4 бита на символ позволяют закодировать 24 = 16 разных символов
3. поэтому мощность алфавита – 16 символов

|  |
| --- |
| **Возможные проблемы**:   * не всегда удобно работать с дробными числами (1/2 байта) * метод разработан специально для этой задачи, где он хорошо работает; в других задачах может быть не так гладко |

### Еще пример задания:

*В зоопарке 32 обезьяны живут в двух вольерах, А и Б. Одна из обезьян заболела. Сообщение «Заболевшая обезьяна живет в вольере А» содержит 4 бита информации. Сколько обезьян живут в вольере Б?*

**Решение (вариант 1)**:

1. информация в 4 бита соответствует выбору одного из 16 вариантов, …
2. … поэтому в вольере А живет 1/16 часть всех обезьян (это **самый важный момент**!)
3. всего обезьян – 32, поэтому в вольере А живет

32/16 = 2 обезьяны

1. поэтому в вольере Б живут все оставшиеся

32 – 2 = 30 обезьян

1. ответ – 30.

|  |
| --- |
| **Возможные ловушки**:   * можно сделать неверный вывод о том, что в вольере А живет 4 обезьяны (столько же, сколько бит информации мы получили), следовательно, в вольере Б живут оставшиеся 28 обезьян (неверный ответ 3) * после п. 1 можно сделать (неверный) вывод о том, что в вольере А живет 16 обезьян, следовательно, в вольере Б – тоже 16 (неверный ответ 2) |

**Решение (вариант 2, использование формулы Шеннона[[10]](#footnote-10))**:

1. заболевшая обезьяна может жить в вольере А (событие 1) или в вольере Б (событие 2)
2. количество информации в сообщении о произошедшем событии с номером  равно , где  – вероятность этого события; таким образом, получаем вероятность того, что заболевшая обезьяна живет в вольере А:

.

1. у нас не было никакой предварительной информации о том, где живет заболевшая обезьяна, поэтому можно считать, что вероятность определяется количеством обезьян в вольере – если вероятность равна 1/16, то в вольере живет 1/16 часть всех обезьян:

32/16 = 2 обезьяны

1. поэтому в вольере Б живут все оставшиеся

32 – 2 = 30 обезьян

1. ответ – 30.

### Еще пример задания:

*В корзине лежат 32 клубка шерсти, из них 4 красных. Сколько бит информации несет сообщение о том, что достали клубок красной шерсти?*

**Решение (вариант 1)**:

1. красные клубки шерсти составляют 1/8 от всех, …
2. поэтому сообщение о том, что первый вынутый клубок шерсти – красный, соответствует выбору одного из 8 вариантов
3. выбор 1 из 8 вариантов – это информация в 3 бита (по таблице степеней двойки)
4. ответ – 3.

**Решение (вариант 2, использование формулы Шеннона)**:

1. красные клубки шерсти составляют 1/8 от всех, поэтому вероятность  того, что первый вынутый клубок шерсти – красный, равна 1/8
2. по формуле Шеннона находим количество информации в битах:

бита.

1. ответ – 3.

### Еще пример задания:

*В некоторой стране автомобильный номер длиной 7 символов составляется из заглавных букв (всего используется 26 букв) и десятичных цифр в любом порядке. Каждый символ кодируется одинаковым и минимально возможным количеством бит, а каждый номер – одинаковым и минимально возможным целым количеством байт. Определите объем памяти, необходимый для хранения 20 автомобильных номеров.*

**Решение**:

1. всего используется 26 букв + 10 цифр = 36 символов
2. для кодирования 36 вариантов необходимо использовать 6 бит, так как , т.е. пяти бит не хватит (они позволяют кодировать только 32 варианта), а шести уже достаточно
3. таким образом, на каждый символ нужно 6 бит (минимально возможное количество бит)
4. полный номер содержит 7 символов, каждый по 6 бит, поэтому на номер требуется бита
5. по условию каждый номер кодируется целым числом байт (в каждом байте – 8 бит), поэтому требуется 6 байт на номер (), пяти байтов не хватает, а шесть – минимально возможное количество
6. на 20 номеров нужно выделить байт
7. ответ – 120.

|  |
| --- |
| **Возможные ловушки**:   * если не обратить внимание на то, что каждый номер кодируется целым числом БАЙТ, получаем неверный ответ 2 (бит = 105 байт) * если по невнимательности считать, что каждый СИМВОЛ кодируется целым числом байт, получаем 7 байт на символ и всего 140 байт (неверный ответ 4) * если «забыть» про цифры, получим всего 26 символов, 5 бит на символ, 35 бит (5 полных байт) на каждый номер и неверный ответ 100 байт (на 20 номеров) |

**Еще пример задания:**

*В школьной базе данных хранятся записи, содержащие информацию об учениках:*

*<Фамилия> – 16 символов: русские буквы (первая прописная, остальные строчные),*

*<Имя> – 12 символов: русские буквы (первая прописная, остальные строчные),*

*<Отчество> – 16 символов: русские буквы (первая прописная, остальные строчные),*

*<Год рождения> – числа от 1992 до 2003.*

*Каждое поле записывается с использованием минимально возможного количества бит. Определите минимальное количество байт, необходимое для кодирования одной записи, если буквы е и ё считаются совпадающими.*

**Решение**:

1. очевидно, что нужно определить минимально возможные размеры в битах для каждого из четырех полей и сложить их;
2. важно! известно, что первые буквы имени, отчества и фамилии – всегда заглавные, поэтому можно хранить их в виде строчных и делать заглавными только при выводе на экран (но нас это уже не волнует)
3. таким образом, для символьных полей достаточно использовать алфавит из 32 символов (русские строчные буквы, «е» и «ё» совпадают, пробелы не нужны)
4. для кодирования каждого символа 32-символьного алфавита нужно 5 бит (32 = 25555), поэтому для хранения имени, отчества и фамилии нужно (16 + 12 + 16)•5=220 бит
5. для года рождения есть 12 вариантов, поэтому для него нужно отвести 4 бита (24 = 16 ≥ 12)
6. таким образом, всего требуется 224 бита или 28 байт
7. правильный ответ – 28 байт.

# 14 (повышенный уровень, время – 6 мин)

**Тема**: Выполнение алгоритмов для исполнителя.

**Что нужно знать**:

* правила выполнения линейных, разветвляющихся и циклических алгоритмов
* основные операции с символьными строками (определение длины, выделение подстроки, удаление и вставка символов, «сцепка» двух строк в одну)
* *исполнитель* – это человек, группа людей, животное, машина или другой объект, который может понимать и выполнять некоторые команды
* в школьном алгоритмическом языке **нц** обозначает «начало цикла», а **кц** – «конец цикла»; все команды между **нц** и **кц** – это тело цикла, они выполняются несколько раз
* запись **нц для i от 1 до n** обозначает начало цикла, в котором переменная **i** (она называется переменной цикла) принимает последовательно все значения от 1 до **n** с шагом 1

### Пример задания:

*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:**

1. запишем общее изменение координат Чертёжника в результате выполнения этого алгоритма:

****

1. поскольку Чертёжник должен вернуться в исходную точку, эти величины должны быть равны нулю; следовательно, нужно найти набольшее натуральное *n*, при котором система уравнений



разрешима в целых числах относительно *a* и *b*

1. несложно заметить, что для этого число *n* должно быть одновременно делителем чисел 14 и 25
2. наибольший общий делитель чисел 14 и 25 равен 1
3. ответ – 1.

### Ещё пример задания:

*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, при этом n > 1):*

НАЧАЛО

**сместиться на (–3, –3)**

ПОВТОРИ n РАЗ

**сместиться на (a, b)**

**сместиться на (27, 12)**

КОНЕЦ ПОВТОРИ

**сместиться на (–22, -7)**

КОНЕЦ

*Укажите наименьшее возможное значение числа n* (*n* > 1)*, для которого найдутся такие значения чисел a и b, что после выполнения программы Чертёжник возвратится в исходную точку*.

**Решение:**

1. запишем общее изменение координат Чертёжника в результате выполнения этого алгоритма:

****

1. поскольку Чертёжник должен вернуться в исходную точку, эти величины должны быть равны нулю; следовательно, нужно найти наименьшее натуральное *n >* 1, при котором система уравнений

****

разрешима в целых числах относительно *a* и *b*

1. несложно заметить, что для этого число *n* должно быть одновременно делителем чисел 10 и 25
2. наименьший общий делитель чисел 10 и 25 равен 5
3. ответ – 5.

### Ещё пример задания:

*Исполнитель Чертёжник перемещается на координатной плоскости, оставляя след в виде линии. Чертёжник может выполнять команду* **Сместиться на (a, b)** *(где a, b – целые числа), перемещающую Чертёжника из точки с координатами (x, y) в точку с координатами (x + a, y + b). Если числа a, b положительные, значение соответствующей координаты увеличивается, если отрицательные – уменьшается. Например, если Чертёжник находится в точке с координатами (2, 4), то команда* **Сместиться на (1, –5)** *переместит Чертёжника в точку (3, –1).*

*Запись*

Повтори k раз

Команда1

Команда2

Команда3

конец

*означает, что последовательность команд Команда1 Команда2 Команда3 повторится k раз.*

*Чертёжнику был дан для исполнения следующий алгоритм:*

Сместиться на (3,-3)

Повтори N раз

Сместиться на (27, 17)

Сместиться на (a, b)

конец

Сместиться на (–27, –17)

*Чему должно равняться N, чтобы Чертежник смог вернуться в исходную точку, из которой он начал движение?*

1) 4 2) 5 3) 6 4) 7

**Решение:**

1. обратим внимание, что в этой задаче значения a и b неизвестны
2. вычислим итоговое смещение Чертёжника (общее изменение каждой координаты должно быть равно 0):

(3, -3)

N раз (27+a, 17+b)

(-27, 17)

общее изменение x-координаты:

3+N(27+a)-27 = 0

общее изменение y-координаты:

-3+N(17+b)-17 = 0

1. упрощаем оба уравнения:

N(27+a) = 24

N(17+b) = 20

1. таким образом, N – общий делитель чисел 24 и 20, это может быть 2 или 4; из вариантов ответа, приведённых в задаче, подходит только 4 (ответ 1)
2. Ответ: 1.

### Ещё пример задания:

*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**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **1** |
|  |  |  |  |  |  | **2** |
|  |  |  |  |  |  | **3** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **5** |
|  |  |  |  |  |  | **6** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

*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 А1)?*

1) 8 2) 12 3) 17 4) 21

**ПОКА слева свободно ИЛИ сверху свободно**

**ЕСЛИ слева свободно**

**ТО влево**

**ИНАЧЕ вверх**

**КОНЕЦ ЕСЛИ**

**КОНЕЦ ПОКА**

**Решение:**

1. в программе один цикл со сложным условием, внутри которого расположен условный оператор «если»
2. 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
3. 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: конечная цель маршрута А1 и две «ложные цели» в В3 и Е1:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **1** |
|  |  |  |  |  |  | **2** |
|  |  |  |  |  |  | **3** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **5** |
|  |  |  |  |  |  | **6** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

1. из п. 2 и 3 следует, что Робот успешно придет в клетку А1, если только он не попадёт в клетки В3 и Е1
2. подсчитаем, сколько есть клеток, из которых Робот попадает в клетку В3; Робот сначала идет влево до упора, потом – вверх, пока не упрётся в стенку сверху или не откроется «окно» влево; отметим голубым цветом все клетки, из которых Робот попадает в В3, их всего 13

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **1** |
|  |  |  |  |  |  | **2** |
|  | **•** |  |  |  |  | **3** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **5** |
|  |  |  |  |  |  | **6** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

1. кроме того, есть две клетки, из которых Робот попадает в Е1, они показаны фиолетовым цветом:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **•** |  | **1** |
|  |  |  |  |  |  | **2** |
|  | **•** |  |  |  |  | **3** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **5** |
|  |  |  |  |  |  | **6** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

1. таким образом, на поле есть всего 15 клеток, из которых Робот при выполнении заданной программы не попадает в клетку А1
2. следовательно, «нужных» клеток 36 – 15 = 21
3. Ответ: 4.

### Ещё пример задания:

*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*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **1** |
|  |  |  |  |  |  | **2** |
|  |  |  |  |  |  | **3** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **5** |
|  |  |  |  |  |  | **6** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

*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 F6)?*

1) 8 2) 15 3) 24 4) 27

**НАЧАЛО**

**ПОКА < справа свободно ИЛИ снизу свободно >**

**ПОКА < справа свободно >**

**вправо**

**КОНЕЦ ПОКА**

**ПОКА < снизу свободно >**

**вниз**

**КОНЕЦ ПОКА**

**КОНЕЦ ПОКА**

**КОНЕЦ**

**Решение:**

1. обратим внимание, что в программе три цикла, причем два внутренних цикла вложены в один внешний
2. цикл

**ПОКА < справа свободно >**

**вправо**

**КОНЕЦ ПОКА**

означает «двигаться вправо до упора», а цикл

**ПОКА < снизу свободно >**

**вниз**

**КОНЕЦ ПОКА**

означает «двигаться вниз до упора»

1. тогда программу можно записать в свободном стиле так:

**ПОКА не пришли в угол**

**двигаться вправо до упора**

**двигаться вниз до упора**

**КОНЕЦ ПОКА**

где угол – это клетка, в которой есть стенки снизу и справа

1. за каждый шаг внешнего цикла Робот проходит путь в виде «сапога», двигаясь сначала вправо до упора, а затем – вниз до упора:

|  |  |  |  |
| --- | --- | --- | --- |
| **→** | **→** | **↓** |  |
|  |  | **↓** |  |
|  |  | **↓** |  |

клетка, выделенная красным фоном особая – в ней заканчивается один шаг внешнего цикла и начинается следующий:

1. Робот может попасть в эту клетку, двигаясь вниз из клетки, где справа – стенка
2. снизу есть стенка;
3. снизу стенка есть, справа – нет, поэтому будет выполнен еще один шаг внешнего цикла.
4. в клетку F6 (это угол, где Робот остановился), Робот мог придти за один шаг внешнего цикла (за один «сапог») только из отмеченных клеток:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **1** |
|  |  |  |  |  |  | **2** |
|  |  |  |  |  |  | **3** |
| **→** | **→** | **→** | **→** | **→** | **↓** | **4** |
|  |  |  | **→** | **→** | **↓** | **5** |
| **→** | **→** | **→** | **→** | **→** |  | **6** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

1. теперь отметим красным фоном особые клетки, которые удовлетворяют условиям а-в пункта 4 (см. выше), их всего 2:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **1** |
|  |  |  |  |  |  | **2** |
|  |  |  |  |  |  | **3** |
| **→** | **→** | **→** | **→** | **→** | **↓** | **4** |
|  |  |  | **→** | **→** | **↓** | **5** |
| **→** | **→** | **→** | **→** | **→** |  | **6** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

1. отметим все пути в форме «сапога», которые приводят в особые клетки:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **1** |
| **→** | **→** | **↓** |  |  |  | **2** |
| **→** | **→** | **↓** |  |  |  | **3** |
| **→** | **→** | **→** | **→** | **→** | **↓** | **4** |
| **→** | **→** | **↓** | **→** | **→** | **↓** | **5** |
| **→** | **→** | **→** | **→** | **→** |  | **6** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

1. больше особых клеток (см. пункт 4) нет; всего отмечено 24 клетки (считая конечную клетку F6)
2. таким образом, правильный ответ – 3.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + нужно помнить, что внешний цикл может выполняться более одного раза; неучет этого обстоятельства приводит к неверному ответу 2 (15 клеток)   + важен порядок выполнения внутренних циклов (в данном случае сначала Робот идет вправо, а затем – вниз); при изменении этого порядка изменится и результат, в частности, изменятся условия, определяющие особую клетку |

### Еще пример задания:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **6** |
|  |  |  |  |  |  | **5** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **3** |
|  |  |  |  |  |  | **2** |
|  |  |  |  |  |  | **1** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

*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> команда** *выполняется, пока условие истинно, иначе происходит переход на следующую строку. Сколько клеток приведенного лабиринта соответствуют требованию, что, выполнив предложенную ниже программу, РОБОТ остановится в той же клетке, с которой он начал движение?*

1) 1 2) 2 3) 3 4) 0

**НАЧАЛО**

**ПОКА <снизу свободно> вниз**

**ПОКА <слева свободно> влево**

**ПОКА <сверху свободно> вверх**

**ПОКА <справа свободно> вправо**

**КОНЕЦ**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

**Решение:**

1. легко понять, что для того, чтобы исполнитель вернулся обратно в ту клетку, откуда он начал движения, четыре стенки должны быть расставлены так, чтобы он упирался в них сначала при движении вниз, затем – влево, вверх и, наконец, вправо:

на рисунке красная точка обозначает клетку, начав с которой РОБОТ вернется обратно;

1. кроме этих четырех стенок, необходимо, чтобы коридор, выделенный на рисунке справа зеленым фоном, был свободен для прохода
2. обратим внимание, что возможны еще «вырожденные» варианты, вроде таких:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

1. итак, мы выяснили, что нужно рассматривать лишь те клетки, где есть стенка справа; отметим на исходной карте клетки-кандидаты:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **•** |  |  | **•** | **6** |
|  |  |  |  | **•** | **•** | **5** |
|  |  |  |  |  | **•** | **4** |
|  |  |  |  |  | **•** | **3** |
|  | **•** |  |  |  | **•** | **2** |
| **•** |  |  |  |  | **•** | **1** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | **•** | **6** |
|  |  |  |  |  | **•** | **5** |
|  |  |  |  |  | **•** | **4** |
|  |  |  |  |  | **•** | **3** |
|  |  |  |  |  | **•** | **2** |
|  |  |  |  |  | **•** | **1** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

1. этих «подозрительных» клеток не так много, но можно еще сократить количество рассматриваемых вариантов: если РОБОТ начинает движение с *любой* клетки на вертикали F, он все равно приходит в клетку F4, которая удовлетворяет заданному условию, таким образом, **одну клетку мы нашли**, а остальные клетки вертикали F условию не удовлетворяют:
2. проверяем оставшиеся четыре клетки-кандидаты, но для каждой из них после выполнения алгоритма РОБОТ не приходит в ту клетку, откуда он стартовал:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **•** |  |  |  | **6** |
|  |  |  |  |  |  | **5** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **3** |
|  |  |  |  |  |  | **2** |
|  |  |  |  |  |  | **1** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **6** |
|  |  |  |  |  |  | **5** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **3** |
|  | **•** |  |  |  |  | **2** |
|  |  |  |  |  |  | **1** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **6** |
|  |  |  |  | **•** |  | **5** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **3** |
|  |  |  |  |  |  | **2** |
|  |  |  |  |  |  | **1** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **6** |
|  |  |  |  |  |  | **5** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **3** |
|  |  |  |  |  |  | **2** |
| **•** |  |  |  |  |  | **1** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

1. итак, условию удовлетворяет только одна клетка – F4
2. таким образом, правильный ответ – 1.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + вариантов может быть достаточно много, важно не пропустить ни один из них   + можно попытаться выполнить алгоритм для *каждой* клетки лабиринта, но это займет много времени; поэтому лучше ограничиться только клетками-кандидатами   + нужно правильно определить свойства, по которым клетку можно считать «кандидатом»   + можно не заметить стенку и таким образом получить лишнее решение |

### Еще пример задания:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **6** |
|  |  |  |  |  |  | **5** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **3** |
|  |  |  |  |  |  | **2** |
|  |  |  |  |  |  | **1** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

*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> команда** *выполняется, пока условие истинно, иначе происходит переход на следующую строку. Сколько клеток приведенного лабиринта соответствуют требованию, что, выполнив предложенную ниже программу, РОБОТ уцелеет (не врежется в стену) и остановится в той же клетке, с которой он начал движение?*

1) 1 2) 2 3) 3 4) 0

**НАЧАЛО**

**ПОКА <слева свободно> вверх**

**ПОКА <сверху свободно> вправо**

**ПОКА <справа свободно> вниз**

**ПОКА <снизу свободно> влево**

**КОНЕЦ**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| 2  1 |  |  |  |  | 3 |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  | 4 |  |  |  |  |
|  |  |  |  |  |  |  |

**Решение:**

1. особенность этой задач в том, что РОБОТ проверяет стенку в одном направлении, а движется в другом
2. рассмотрим первый цикл:

**ПОКА <слева свободно> вверх**

понятно, что при движении вверх РОБОТ остановится в первой же клетке, где слева будет стена

1. рассуждая аналогично, находим, что во втором цикле при движении вправо РОБОТ останавливается в клетке, где есть стена сверху; в третьем цикле (движение вниз) РОБОТ останавливается в клетке, где есть стена справа;
2. наконец, в четвертом цикле РОБОТ останавливается в клетке, где есть стена снизу; при этом он должен попасть обратно в исходную клетку, обозначенную на рисунке красной точкой;
3. кроме этих четырех стенок, необходимо, чтобы коридор, выделенный на рисунке зеленым фоном, был свободен для прохода, иначе РОБОТ врежется в стенку
4. теперь отметим на карте все клетки-кандидаты, где снизу есть стена:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **•** |  | **6** |
|  | **•** |  |  |  |  | **5** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  | **•** | **3** |
|  |  | **•** |  |  |  | **2** |
| **•** | **•** | **•** | **•** | **•** | **•** | **1** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

1. при движении из клеток B5, D1, E1, E6, F1 и F3 РОБОТ врежется в стенку, потому что слева стены нет и условие «*слева свободно»* всегда истинно:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **•** |  | **6** |
|  | **•** |  |  |  |  | **5** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  | **•** | **3** |
|  |  | **•** |  |  |  | **2** |
| **•** | **•** | **•** | **•** | **•** | **•** | **1** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

1. начав движение с клетки A1, C1 или C2, РОБОТ также врезается в стенку и разрушается:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **6** |
|  |  |  |  |  |  | **5** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **3** |
|  |  | **•** |  |  |  | **2** |
| **•** | **•** | **•** |  |  |  | **1** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

1. и только путь, начатый в клетке B1, приводит РОБОТА обратно в точку старта:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **6** |
|  |  |  |  |  |  | **5** |
|  |  |  |  |  |  | **4** |
|  |  |  |  |  |  | **3** |
|  |  |  |  |  |  | **2** |
|  | **•** |  |  |  |  | **1** |
| **A** | **B** | **C** | **D** | **E** | **F** |  |

1. таким образом, только клетка B1 удовлетворяет условию задачи, поэтому …
2. правильный ответ – 1.

### Еще пример задания:

*В приведенном ниже фрагменте алгоритма, записанном на алгоритмическом языке, переменные* **a***,* **b***,* **c** *имеют тип «строка», а переменные* **i***,* **k** *– тип «целое». Используются следующие функции:*

**Длина(a)** *– возвращает количество символов в строке* **a***. (Тип «целое»)*

**Извлечь(a,i)** *– возвращает* **i***-тый (слева) символ в строке* **a***. (Тип «строка»)*

**Склеить(a,b)** *– возвращает строку, в которой записаны сначала все символы   
 строки* **a***, а затем все символы строки* **b***. (Тип «строка»)*

*Значения строк записываются в одинарных кавычках (Например,* **a:='дом'***). Фрагмент алгоритма:*

**i := Длина(a)**

**k := 2**

**b := 'А'**

**пока i > 0**

**нц**

**c := Извлечь(a,i)**

**b := Склеить(b,c)**

**i := i – k**

**кц**

**b := Склеить(b,'Т')**

*Какое значение будет у переменной* **b** *после выполнения вышеприведенного фрагмента алгоритма, если значение переменной* **a** *было* ‘ПОЕЗД’*?*

1) ‘АДЕПТ’ 2) ‘АДЗЕОП’ 3) ‘АДТЕТПТ’ 4) ‘АДЗОТ’

**Решение:**

1. эта задача более близка к классическому программированию, здесь выполняется обработка символьных строк; вся информация для успешного решения, вообще говоря, содержится в условии, но желательно иметь хотя бы небольшой опыт работы с символьными строками на Паскале (или другом языке)
2. заметим, что последняя команда алгоритма, **b:=Склеить(b,'Т')**, добавляет букву 'Т' в конец строки **b**, поэтому ответ 2 – явно неверный (строка должна оканчиваться на букву 'Т', а не на 'П')
3. для решения будем использовать ручную прокрутку; здесь пять переменных: **a**, **b**, **c**, **i**, **k**, для каждой из них выделим столбец, где будем записывать изменение ее значения
4. перед выполнением заданного фрагмента мы знаем только значение **a**, остальные неизвестны (обозначим их знаком вопроса):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **a** | **b** | **c** | **i** | **k** |
|  | 'ПОЕЗД' | ? | ? | ? | ? |

1. в первой команде длина строки **a** (она равна 5 символам) записывается в переменную **i**:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **a** | **b** | **c** | **i** | **k** |
|  | 'ПОЕЗД' | ? | ? | ? | ? |
| **i:=Длина(a)** |  |  |  | 5 |  |

1. следующие два оператора записывают начальные значения в **k** и **b**:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **a** | **b** | **c** | **i** | **k** |
|  | 'ПОЕЗД' | ? | ? | ? | ? |
| **i:=Длина(a)** |  |  |  | 5 |  |
| **k:=2** |  |  |  |  | 2 |
| **b:='А'** |  | 'A' |  |  |  |

1. далее следует цикл **пока** с проверкой условия **i>0** в начале цикла; сейчас **i=5>0**, то есть, условие выполняется, цикл начинает работать и выполняются все операторы в теле цикла:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **a** | **b** | **c** | **i** | **k** |
|  | 'ПОЕЗД' | ? | ? | ? | ? |
| **i:=Длина(a)** |  |  |  | 5 |  |
| **k:=2** |  |  |  |  | 2 |
| **b:='А'** |  | 'A' |  |  |  |
| **i > 0?** | **да** | | | | |
| **c:=Извлечь(a,i)** | **i:=Длина(a)** |  |  |  | 5 |
| **b:=Cклеить(b,c)** | **k:=2** |  |  |  |  |
| **i:=i–k** |  |  |  | 3 |  |

* поскольку **i=5**, вызов функции **Извлечь(a,i)** выделяет из строки **a** символ с номером 5, это 'Д';
* следующей командой этот символ приписывается в «хвост» строки **b**, теперь в ней хранится цепочка 'АД';
* в команде **i:=i-k** значение переменной **i** уменьшается на **k** (то есть, на 2)

1. далее нужно перейти в начало цикла и снова проверить условие **i>0**, оно опять истинно, поэтому выполняется следующий шаг цикла, в котором к строке **b** добавляется 3-й символ строки **a**, то есть 'Е':

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **a** | **b** | **c** | **i** | **k** |
| **...** | 'ПОЕЗД' | 'АД' | … | 3 | 2 |
| **i > 0?** | **да** | | | | |
| **c:=Извлечь(a,i)** |  |  | 'Е' |  |  |
| **b:=Cклеить(b,c)** |  | 'АДЕ' |  |  |  |
| **i:=i–k** |  |  |  | 1 |  |

1. условие **i>0** истинно, поэтому тело цикла выполняется еще один раз, к строке **b** добавляется 1-й символ строки **a**, то есть 'П':

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **a** | **b** | **c** | **i** | **k** |
| **...** | 'ПОЕЗД' | 'АДЕ' | … | 1 | 2 |
| **i > 0?** | **да** | | | | |
| **c:=Извлечь(a,i)** |  |  | 'П' |  |  |
| **b:=Cклеить(b,c)** |  | 'АДЕП' |  |  |  |
| **i:=i–k** |  |  |  | –1 |  |

1. теперь **i=-1**, поэтому при очередной проверке условие **i>0** в начале цикла оказывается ложным, выполнение цикла заканчивается, и исполнителю остается выполнить единственную строчку после цикла, которая дописывает в конец строки **b** букву 'Т':

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **a** | **b** | **c** | **i** | **k** |
| **...** | 'ПОЕЗД' | 'АДЕП' | … | –1 | 2 |
| **i > 0?** | **нет** | | | | |
| **b:=Склеить(b,'Т')** |  | 'АДЕПТ' |  |  |  |

1. у нас получилось, что в конце выполнения фрагмента алгоритма в переменной **b** будет записана последовательность символов 'АДЕПТ'
2. таким образом, правильный ответ – 1.

|  |
| --- |
| **Возможные проблемы**:   * + таблица получилась достаточно громоздкая, однако она позволяет наиболее наглядно решить задачу |

### Еще пример задания[[11]](#footnote-11):

*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> команда** *выполняется, пока условие истинно, иначе происходит переход на следующую строку.*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

*Сколько клеток приведенного лабиринта соответствуют требованию, что, выполнив предложенную ниже программу, РОБОТ уцелеет (не врежется в стену)?*

1) 1 2) 13 3) 21 4) 39

**НАЧАЛО**

**ПОКА <снизу свободно> вниз**

**ПОКА <слева свободно> влево**

**вверх**

**вправо**

**КОНЕЦ**

**Решение:**

1. нарисуем примерный путь Робота в соответствии с программой; вот три варианта, когда Робот не разбивается:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1) | ? | ? | ? | ? |  | 2) | ? | ? |  | ? | 3) | ? |  | ? | ? |
|  | ? | ? | ? | ? |  |  | ? | ? |  | ? |  | ? |  | ? | ? |
|  | ? |  |  | ? |  |  | ? |  |  | ? |  | ? |  |  | ? |
|  |  |  |  |  |  |  |  |  |  | ? |  |  |  | ? | ? |
|  | ? | ? | ? | ? |  |  | ? | ? |  | ? |  | ? |  | ? | ? |

здесь ключевые клетки – две стенки (слева и снизу) и три ярко-зеленых клетки, которые должны быть свободны

1. теперь ищем на карте участки, где есть все ключевые клетки (они выделены на рисунке):

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

обратите внимание, что в двух случаях нижняя «ключевая» стенка имеет длину больше 1 (темно-коричневый цвет), то есть Робот может спускаться по разным линиям.

1. теперь осталось подсчитать все клетки, спускаясь из которых Робот упирается в темно-коричневые стенки:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

1. подсчет показывает, что их 39 штук;
2. поэтому правильный ответ – 4.

# 15 (повышенный уровень, время – 3 мин)

**Тема**: Графы. Поиск количества путей

**Что нужно знать**:

* если в город R можно приехать только из городов X, Y, и Z, то число различных путей из города A в город R равно сумме числа различных путей проезда из A в X, из A в Y и из A в Z, то есть

,

где  обозначает число путей из вершины A в некоторую вершину Q

* число путей конечно, если в графе нет циклов – замкнутых путей

### Пример задания:

*На рисунке – схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, И, К, М. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько существует различных путей, ведущих из города А в город М и проходящих через город В?*

А

Б

Г

Д

B

Ж

Е

И

К

М

**Решение:**

1. для того, чтобы оставить только маршруты, проходящие через вершину В, нужно представить граф в таком виде, «собрав его в пучок» около вершины В:

А

М

В

1. проведём сечение графа через вершину В:

А

Б

Г

Д

B

Ж

Е

И

К

М

1. обратим внимание на такой факт: если мы перешли через линию сечения из левой части в правую по ребру ГЕ или через вершину Ж, мы уже никак не попадём в вершину В (нет рёбер с «обратным направлением», поэтому эти маршруты запрещены; для более сложных случаев, когда такие рёбра с «обратным направлением» есть, нужно перерисовать граф (или провести сечение иначе) так, чтобы все вершины, ИЗ которых можно попасть в В, оказались слева от линии сечения
2. в данном случае выбрасывается вершина Ж, все связанные с ней рёбра, и ребро ГЕ:

А

Б

Г

Д

B

Е

И

К

М

1. дальше используем стандартный метод (см. разбор следующей задачи)
2. покажем только окончательный результат:

А

Б

Г

Д

B

Е

И

К

М

1

1

1

3

4

4

8

4

16

1. Ответ: 16.

### Ещё пример задания:

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

А

Б

В

Г

Д

Е

Ж

И

К

Л

**Решение:**

1. будем обозначать через NX количество различных путей из города А в город X
2. для города А есть только один маршрут – никуда не двигаться, поэтому NA = 1
3. для любого города X количество маршрутов NX можно вычислить как

Nx = Ny + … + Nz

где сумма взята по всем вершинам, из которых есть прямой путь в вершину X; например,

NЛ = NИ + NЖ + NК

1. около каждого города будем записывать количество маршрутов из А в этот город
2. начнем считать количество путей с начала маршрута – с города А:

А

Б

В

Г

Д

Е

Ж

И

К

Л

1

1. теперь находим те вершины, в которые можно попасть напрямую из уже рассмотренных вершин (пока – только из А), это Б и Г, для них тоже количество путей равно 1:

А

Б

В

Г

Д

Е

Ж

И

К

Л

1

1

1

1. теперь можно определить количество путей для В и Е; в В можно приехать только из А, Б и Г, а в Е – только из Г:

NВ = NА + NБ + NГ = 1 + 1 + 1 = 3

NЕ = NГ = 1

А

Б

В

Г

Д

Е

Ж

И

К

Л

1

1

1

3

1

1. теперь можно определить количество путей для Д, Ж и К; в Д можно приехать только из Б и В, в Ж – из В и Е, а в Е – только из Г:

NД = NБ + NВ = 1 + 3 = 4

NЖ = NВ + NЕ = 3 + 1 = 4

NК = NЕ­ = 1

А

Б

В

Г

Д

Е

Ж

И

К

Л

1

1

1

3

1

4

1

4

1. теперь можно определить количество путей для И, куда можно приехать только из Д (NИ = NД) и, наконец, для Л:

NЛ = NД + NИ + NЖ + NК = 13

А

Б

В

Г

Д

Е

Ж

И

К

Л

1

1

1

3

1

4

1

4

4

13

1. Ответ: 13.

### Ещё пример задания:

*На карту нанесены 4 города (A, B, C и D)*. Известно, что

*между городами A и С – три дороги*

*между городами C и B – две дороги*

*между городами A и B – две дороги*

*между городами C и D – две дороги*

*между городами B и D – четыре дороги*

*По каждой из этих дорог можно ехать в обе стороны. Сколькими различными способами можно проехать из города А в город D, посещая каждый город не более одного раза?*

**Решение:**

1. нарисуем граф, в котором множественные дороги из одного города в другой будем обозначать одной дугой и подписывать около неё количество дорог:

А

B

D

С

3

2

4

2

2

1. выпишем все маршруты, по которым можно ехать из A в D так, чтобы дважды не проезжать один и тот же город:

|  |  |  |  |
| --- | --- | --- | --- |
| 2 4 | 3 2 | 2 2 2 | 3 2 4 |
| A → B → D | A → С → D | A → B → С → D | A → C → B → D |

1. теперь рассмотрим маршрут A → B → D; сначала можно двумя путями приехать из A в B, а затем – 4-мя путями из B в D; поэтому общее количество различных маршрутов равно произведению этих чисел: 2\*4 = 8
2. аналогично находит количество различных путей по другим маршрутам

A → С → D: 3\*2 = 6

A → B → С → D: 2\*2\*2 = 8

A → C → B → D: 3\*2\*4 = 24

1. всего получается 8 + 6 + 8 + 24 = 46.
2. Ответ: 46.

### Еще пример задания:

*На рисунке – 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?*

Е

Г

В

А

К

Б

Д

Ж

И

**Решение (1 вариант, подстановки):**

1. начнем считать количество путей с конца маршрута – с города К
2. будем обозначать через NX количество различных путей из города А в город X
3. общее число путей обозначим через N
4. по схеме видно, что NБ = NГ = 1
5. очевидно, что если в город X можно приехать только из Y, Z, то NX = NY + N­Z, то есть нужно сложить число путей, ведущих из A во все города, откуда можно приехать в город X
6. поскольку в K можно приехать из Е, Д, Ж или И, поэтому

N = N­К = NД + NЕ + NЖ + NИ

1. в город И можно приехать только из Д, поэтому NИ = NД
2. в город Ж можно приехать только из Е и В, поэтому

N­Ж = NЕ + NВ

1. подставляем результаты пп. 6 и 7 в формулу п. 5:

N = NВ + 2NЕ + 2NД

1. в город Д можно приехать только из Б и В, поэтому

N­Д = NБ + NВ

так что

N = 2NБ + 3NВ + 2NЕ

1. в город Е можно приехать только из Г, поэтому N­Е = NГ так что

N = 2NБ + 3NВ + 2NГ

1. по схеме видно, что NБ = NГ = 1, кроме того, NВ = 1 + N­Б + NГ = 3
2. окончательно N = 2NБ + 3NВ + 2NГ  = 2·1 + 3·3 + 2·1 = 13
3. Ответ: 13.

**Решение (2 вариант, удобная форма записи):**

1. начнем считать количество путей с конца маршрута – с города К
2. записываем для каждой вершины, из каких вершин можно в нее попасть

|  |  |
| --- | --- |
| вершина | откуда? |
| К | ИДЖЕ |
| И | Д |
| Ж | ВЕ |
| Е | Г |
| Д | БВ |
| Г | А |
| В | АБГ |
| Б | А |

К ← ИДЖЕ

И ← Д

Ж ← ВЕ

Е ← Г

Д ← БВ

Г ← А

В ← АБГ

Б ← А

1. теперь для удобства «обратного хода» вершины можно отсортировать так[[12]](#footnote-12), чтобы сначала шли все вершины, в которые можно доехать только из начальной точки А:

Б ← А

Г ← А

затем на каждом шаге добавляем те вершины, в которые можно доехать из уже добавленных в список (и из исходной точки):

В ← АБГ

Е ← Г

|  |  |  |
| --- | --- | --- |
| вершина | откуда? | N |
| Б | А | 1 |
| Г | А | 1 |
| В | АБГ | 3 |
| Е | Г | 1 |
| Д | БВ | 4 |
| Ж | ВЕ | 4 |
| И | Д | 4 |
| К | ИДЖЕ | 13 |

далее добавляем все вершины, куда можно доехать из А, Б, Г, В и Е:

Д ← БВ

Ж ← ВЕ

на следующем шаге добавляем вершину И

И ← Д

и, наконец, конечную. вершину

К ← ИДЖЕ

именно в таком порядке мы и будем вычислять количество путей для каждой вершины

1. теперь идем по полученному списку вершин, полагая, что количество вариантов попасть в вершину равно суммарному количеству вариантов попасть в ее непосредственных предшественников.

NБ = 1, NГ = 1

NВ = 1+1+1 = 3, NЕ = 1

NД = 1+3 = 4, NЖ = 3 + 1 = 4

NИ = 4,

N = NК = 4 + 4 + 4 + 1 = 13

1. заметим, что вершины можно и не сортировать специально, а просто выбирать возможный порядок вычисления: проверять, какие значения известны и какие можно рассчитать с их помощью на следующем шаге
2. Ответ: 13.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + очень важна аккуратность и последовательность; сначала идем от конечной точки к начальной, выписывая все вершины, из которых можно приехать в данную; затем идем обратно, определяя числовые значения   + построение полного дерева маршрутов – занятие трудоемкое и достаточно бесперспективное, даже грамотные учителя информатики здесь в большинстве случаев что-то забывают и ошибаются |

**Решение (3 вариант, перебор вершин по алфавиту):**

|  |  |
| --- | --- |
| вершина | откуда? |
| Б | А |
| В | АБГ |
| Г | А |
| Д | БВ |
| Е | Г |
| Ж | ВЕ |
| И | Д |
| К | ИДЖЕ |

1. Запишем вершины в алфавитном порядке и для каждой из них определим, из каких вершин можно в нее попасть

Б ← А

В ← АБГ

Г ← А

Д ← БВ

Е ← Г

Ж ← ВЕ

И ← Д

К ← ИДЖЕ

1. теперь определяем количество путей; сначала ставим 1 для тех вершин, в которые можно проехать только из начальной (А):

|  |  |  |
| --- | --- | --- |
| вершина | откуда? | N |
| Б | А | 1 |
| В | АБГ |  |
| Г | А | 1 |
| Д | БВ |  |
| Е | Г |  |
| Ж | ВЕ |  |
| И | Д |  |
| К | ИДЖЕ |  |

1. затем на каждом шаге добавляем те вершины, в которые можно доехать из уже добавленных в список (и из исходной точки):

|  |  |  |
| --- | --- | --- |
| вершина | откуда? | N |
| Б | А | 1 |
| В | АБГ | 3 |
| Г | А | 1 |
| Д | БВ |  |
| Е | Г | 1 |
| Ж | ВЕ |  |
| И | Д |  |
| К | ИДЖЕ |  |

1. следующий шаг

|  |  |  |
| --- | --- | --- |
| вершина | откуда? | N |
| Б | А | 1 |
| В | АБГ | 3 |
| Г | А | 1 |
| Д | БВ | 4 |
| Е | Г | 1 |
| Ж | ВЕ | 4 |
| И | Д |  |
| К | ИДЖЕ |  |

1. и последние 2 шага

|  |  |  |
| --- | --- | --- |
| вершина | откуда? | N |
| Б | А | 1 |
| В | АБГ | 3 |
| Г | А | 1 |
| Д | БВ | 4 |
| Е | Г | 1 |
| Ж | ВЕ | 4 |
| И | Д | 4 |
| К | ИДЖЕ | 13 |

1. Ответ: 13.

**Решение (4 вариант, перебор всех путей с начала, А. Яфарова):**

1. запишем все вершины, в которые есть прямой путь из вершины A: Б, В и Г; получается три начальных отрезка:

АБ, АВ, АГ

1. рассмотрим маршрут АБ: из Б можно ехать в В и Д, поэтому получаем два маршрута:

АБ***В***, АБ***Д***

1. рассматриваем конечные точки этих маршрутов: из В можно ехать в Д и Ж, а из Д – в И и К:

АБВ***Д***, АБВ***Ж***, АБД***И***, АБД***К***

1. снова смотрим на конечные точки: из Д едем в И и К, из Ж и И – только в К:

АБВД***И***, АБВД***К***, АБВЖ***К***, АБДИ***К***, АБДК

1. из И едем только в К, таким образом, все возможные маршруты, содержащие участок АБ, доведены до конечной точки К, всего **5 таких маршрутов**:

АБВДИ***К***, АБВДК, АБВЖК, АБДИК, АБДК

1. затем аналогично рассматриваем маршруты, которые начинаются с АВ:

АВД, АВЖ

АВДИ, АВДК, АВЖК

АВДИК, АВДК, АВЖК

всего **3 маршрута**

1. наконец, остается рассмотреть маршруты, которые начинаются с АГ:

АГВ, АГЕ

АГВД, АГВЖ, АГЕЖ, АГЕК

АГВДИ, АГВДК, АГВЖК, АГЕЖК, АГЕК

АГВДИК, АГВДК, АГВЖК, АГЕЖК, АГЕК

всего **5 маршрутов**

1. складываем количество маршрутов для всех начальных участков: 5 + 3 + 5 = 13
2. Ответ: 13.

|  |
| --- |
| **Возможные проблемы**:   * + при большом количестве маршрутов легко запутаться и что-то пропустить |

**Решение (5 вариант, графический, О.О. Грущак, КузГПА):**

1. Главную идею решения: (число дорог в город N есть сумма дорог, приводящих в города, из которых есть прямой проезд в город N), отразим на самой схеме, показывая на ней ЧИСЛО ДОРОГ, приводящих в каждый город.
2. Последовательность очевидна: начинаем с Б и Г (городов, куда есть по 1-й дороге из А)

В

А

К

Е

Д

Ж

И

1. Посчитаем дороги в В: 1 (из A)+ 1(дороги города Б)+ 1(дороги города В)= 3

А

К

Е

Д

Ж

И

1. Аналогично посчитаем дороги в Д, И, Е, Ж:

А

1. Определяем число дорог в город К, как сумму дорог в города, с которыми он связан: Д, И, Ж, Е.

А

1. Ответ: 13.

# 16 (повышенный уровень, время – 2 мин)

**Тема**: Кодирование чисел. Системы счисления.

**Что нужно знать**:

* принципы кодирования чисел в позиционных системах счисления
* чтобы перевести число, скажем, 12345N, из системы счисления с основанием  в десятичную систему, нужно умножить значение каждой цифры на  в степени, равной ее разряду:

**N0 = 1**

4 3 2 1 0 ← разряды

**1 2 3 4 5N = 1·N4 + 2·N3 + 3·N2 + 4·N1 + 5·N0**

* последняя цифра записи числа в системе счисления с основанием  – это остаток от деления этого числа на 
* две последние цифры – это остаток от деления на , и т.д.
* число 2N в двоичной системе записывается как единица и N нулей:
* число 2N-1в двоичной системе записывается как *N* единиц:
* число 2*N–*2*K* при *K* < *N* в двоичной системе записывается как *N–K* единиц и *K* нулей: 
* поскольку , получаем , откуда следует, что 

### Пример задания:

**Р-21**. Сколько значащих нулей в двоичной записи числа 4512 + 8512 – 2128 – 250

**Решение (способ Е.А. Смирнова, Нижегородская область):**

1. *Общая идея*: количество значащих нулей равно количеству всех знаков в двоичной записи числа (его длине!) минус количество единиц
2. приведём все числа к степеням двойки, учитывая, что 250 = 256 – 4 – 2 = 28 – 22 – 21:

4512 + 8512 – 2128 – 250 = (22)512 + (23)512 – 2128 – 28 + 22 + 21 =

= 21536 + 21024 – 2128 – 28 + 22 + 21

1. старшая степень двойки – 21536, двоичная запись этого числа представляет собой единицу и 1536 нулей, то есть, состоит из 1537 знаков; таким образом, остаётся найти количество единиц
2. вспомним, число 2*N–*2*K* при *K* < *N* записывается как *N–K* единиц и *K* нулей: 
3. для того чтобы использовать это свойство, нам нужно представить заданное выражение в виде пар вида 2*N–*2*K*, причём в этой цепочке степени двойки нужно выстроить по убыванию
4. в нашем случае вы выражении

21536 + 21024 – 2128 – 28 + 22 + 21

стоит два знака «минус» подряд, это не позволяет сразу использовать формулу

1. используем теперь равенство , так что – 2128 = – 2129 + 2128; получаем

21536 + 21024 – 2129 + 2128 – 28 + 22 + 21

здесь две пары 2*N–*2*K* , а остальные слагаемые дают по одной единице

1. общее число единиц равно 1 + (1024 – 129) + (128 – 8) + 1 + 1 = 1018
2. таким образом, количество значащих нулей равно 1537 – 1018 = 519
3. ответ: 519.

### Ещё пример задания:

**Р-20**. Сколько единиц в двоичной записи числа 42015 + 8405 – 2150 – 122

**Решение (способ Е.А. Смирнова, Нижегородская область):**

1. приведём все числа к степеням двойки, учитывая, что 122 = 128 – 4 – 2 = 27 – 22 – 21:

42015 + 8405 – 2150 – 122 = (22)2015 + (23)405 – 2150 – 27 + 22 + 21 =

= 24030 + 21215 – 2150 – 27 + 22 + 21

1. вспомним, число 2*N–*2*K* при *K* < *N* записывается как *N–K* единиц и *K* нулей: 
2. для того чтобы использовать это свойство, нам нужно представить заданное выражение в виде пар вида 2*N–*2*K*, причём в этой цепочке степени двойки нужно выстроить по убыванию
3. в нашем случае вы выражении

24030 + 21215 – 2150 – 27 + 22 + 21

стоит два знака «минус» подряд, это не позволяет сразу использовать формулу

1. используем теперь равенство , так что – 2150 = – 2151 + 2150; получаем

24030 + 21215 – 2151 + 2150 – 27 + 22 + 21

здесь две пары 2*N–*2*K* , а остальные слагаемые дают по одной единице

1. общее число единиц равно 1 + (1215 – 151) + (150 – 7) + 1 + 1 = 1210
2. ответ: 1210.

**Решение (С.О. Куров, Москва):**

1. приведём все числа к степеням двойки, учитывая, что 122 = 128 – 4 – 2 = 27 – 22 – 21:

42015 + 8405 – 2150 – 122 = (22)2015 + (23)405 – 2150 – 27 + 22 + 21 =

= 24030 + 21215 – 2150 – 27 + 22 + 21

1. ищем в **разности** крайнюю левую степень двойки и крайнюю правую 21215 – 27, при этом 2150 на время «теряем»
2. определяем количество единиц в разности 21215 – 27, получаем 1215 – 7 = 1208 единиц
3. так как «внутри» этой разности есть еще 2150, то просто вычитаем одну единицу: 1208 – 1 = 1207; итого в разности 21215 – 2150 – 27 ровно 1207 единиц
4. осталось прибавить по одной единицы от чисел 24030, 22, 21
5. Ответ: 1210

### Ещё пример задания:

**Р-19**. Решите уравнение .

Ответ запишите в троичной системе счисления. Основание системы счисления указывать не нужно.

**Решение:**

1. переведём все числа в десятичную систему счисления:



1. собирая всё в одно уравнение получаем



1. это уравнение имеет два решения, 6 и -8; основание системы счисления – натуральное число, поэтому ответ – 6
2. переводим ответ в троичную систему: 6 = 2∙31 = 203.
3. ответ: 20.

### Ещё пример задания:

**Р-18**. Сколько единиц в двоичной записи числа 42014 + 22015 – 8

**Решение:**

1. приведём все числа к степеням двойки:

42014 + 22015 – 8 = (22)2014 + 22015 – 23 = 24028 + 22015 – 23

1. вспомним, что число 2N-1в двоичной системе записывается как *N* единиц:,   
   а число 2*N–*2*K* при *K* < *N* записывается как *N–K* единиц и *K* нулей: 
2. согласно п. 2, число 22015 – 23 запишется как 2012 единиц и 3 нуля
3. прибавление 24028 даст ещё одну единицу, всего получается 2012 + 1 = 2013 единиц
4. ответ: 2013.

### Ещё пример задания:

**Р-17**. Сколько единиц в двоичной записи числа 42016 + 22018 – 8600 + 6

**Решение:**

1. приведём все числа к степеням двойки, разложив 6 как 22+21

42016 + 22018 – 8600 + 6 = (22)2016 + 22018 - (23)600 + 22 + 21 = 24032 + 22018 – 21800 + 22 + 21

1. вспомним, что число 2N-1в двоичной системе записывается как *N* единиц:,   
   а число 2*N–*2*K* при *K* < *N* записывается как *N–K* единиц и *K* нулей: 
2. согласно п. 2, число 22018 – 21800 запишется как 218 единиц и 1800 нулей
3. прибавление 24032 даст ещё одну единицу, а прибавление 22 + 21 – ещё две, всего получается 218 + 3 = 221 единица
4. ответ: 221.

### Ещё пример задания:

**Р-16**. Сколько единиц в двоичной записи числа 42016 – 22018 + 8800 – 80

**Решение:**

1. приведём все числа к степеням двойки, разложив 80 как 26+24

42016 – 22018 + 8800 – 80 = (22)2016 – 22018 + (23)800 – 22 – 21 = 24032 – 22018 + 22400 – 26 – 24

1. перестроим слагаемые в порядке уменьшения степеней двойки

24032 + 22400 – 22018 – 26 – 24

1. вспомним, что число 2N-1в двоичной системе записывается как *N* единиц:,   
   а число 2*N–*2*K* при *K* < *N* записывается как *N–K* единиц и *K* нулей: 
2. согласно п. 2, число 22400 – 22018 запишется как 382 единицы и 2018 нулей
3. добавляем старшее слагаемое 24032, получаем число 24032 + 22400 – 22018, в котором 383 единицы и в конце (после последней единицы) – 2018 нулей:



1. выделим из этого значения последнюю единицу со следующими 2018 нулями как отдельное слагаемое (число 22018):

,

где число *K* содержит 382 единицы в старших разрядах; таки образом, интересующее нас число равно 

1. согласно п. 2, число 22018 – 26 запишется как 2012 единиц и 6 нулей; также выделим последнюю единицу с последующими нулями как отдельное слагаемое:



где число *L* содержит 2011 единиц

1. теперь остаётся найти, сколько единиц будет в двоичной записи числа 26 – 24, согласно п. 2 находим, что оно содержит 2 единицы
2. таким образом, общее число единиц равно 382 + 2011 + 2 = 2395
3. ответ: 2395.

**Решение (способ 2, Е.А. Смирнов, Нижегородская область):**

1. приведём все числа к степеням двойки, разложив 80 как 26+24

42016 – 22018 + 8800 – 80 = (22)2016 – 22018 + (23)800 – 22 – 21 = 24032 – 22018 + 22400 – 26 – 24

1. перестроим слагаемые в порядке уменьшения степеней двойки

24032 + 22400 – 22018 – 26 – 24

1. представим – 22018 = – 22019 + 22018  и – 26 = – 27 + 26

24032 + 22400 – 22019 + 22018  – 27 + 26– 24

1. слагаемое 24032 в двоичной записи содержит 1 единицу
2. слагаемое 22400 – 22019содержит 381 единицу (число 2*N–*2*K* при *K* < *N* в двоичной системе записывается как *N–K* единиц и *K* нулей: )
3. слагаемое 22018  – 27 содержит 2011 единиц, слагаемое 26– 24содержит 2 единицы
4. позиции единиц во всех этих слагаемых не совпадают, поэтому общее количество единиц равно 1 + 381 + 2011 + 2 = 2395

ответ: 2395

**Решение (способ 3, А.И. Козлов, г. Северобайкальск):**

1. приведём все числа к степеням двойки, разложив 80 как 26+24

42016 – 22018 + 8800 – 80 = (22)2016 – 22018 + (23)800 – 22 – 21 = 24032 – 22018 + 22400 – 26 – 24

1. перестроим слагаемые в порядке уменьшения степеней двойки

24032 + 22400 – 22018 – 26 – 24

1. выражение 22400–24 дает 2396 единиц и 4 нолика в конце, откуда вычеркиваем (заменяем на ноль) единичку, стоящую на седьмом месте справа (26) и, соответственно на 2019 месте справа (22018). Следовательно, остается 2394 единички.
2. С учетом того, что 24032  дает нам одну единицу, в итоге получаем 2395 единиц
3. Ответ: 2395

### Ещё пример задания:

**Р-15**. *Решите уравнение .  
Ответ запишите в шестеричной системе счисления. Основание системы счисления указывать не нужно.*

**Решение:**

1. удобнее всего перевести все числа в десятичную систему, решить уравнение и результат перевести в шестеричную систему
2. получаем **
3. уравнение приобретает вид **, откуда получаем **
4. переводим 15 в шестеричную систему счисления: **
5. ответ: 23.

### Ещё пример задания:

**Р-14**. *Запись десятичного числа в системах счисления с основаниями 3 и 5 в обоих случаях имеет последней цифрой 0. Какое минимальное натуральное десятичное число удовлетворяет этому требованию?*

**Решение:**

1. если запись числа в системе счисления с основанием N заканчивается на 0, то это число делится на N нацело
2. поэтому в данной задаче требуется найти наименьшее натуральное число, которое делится одновременно на 3 и на 5, то есть, делится на 15
3. очевидно, что это число 15.

### Ещё пример задания:

**Р-13**. *Запись числа 6710 в системе счисления с основанием N оканчивается на 1 и содержит 4 цифры. Укажите основание этой системы счисления N.*

**Решение:**

1. поскольку запись в системе счисления с основанием N заканчивается на 1, то остаток от деления числа 67 на N равен 1, то есть при некотором целом  имеем



1. следовательно, основание N – это делитель числа 66
2. с другой стороны, запись числа содержит 4 цифры, то есть 
3. выпишем кубы и четвертые степени первых натуральных чисел, которые являются делителями числа 66:



1. видим, что из этого списка только для числа N = 3 выполняется условие 
2. таким образом, верный ответ – 3.
3. можно сделать проверку, переведя число 67 в троичную систему **6710 = 21113**

### Еще пример задания:

**Р-12**. *Запись числа 38110 в системе счисления с основанием N оканчивается на 3 и содержит 3 цифры. Укажите наибольшее возможное основание этой системы счисления N.*

**Решение:**

1. поскольку запись в системе счисления с основанием N заканчивается на 3, то остаток от деления числа 381 на N равен 3, то есть при некотором целом  имеем



1. следовательно, основание N – это делитель числа 
2. с другой стороны, запись числа содержит 3 цифры, то есть 
3. неравенство  дает (так как )
4. неравенство  дает (так как )
5. таким образом, ; в этом диапазоне делителями числа 378 являются числа
   * 9, при  получаем запись числа 
   * 14, при  получаем запись числа 
   * 18, при  получаем запись числа 
6. наибольшим из приведенных чисел – это 18 (можно было сразу искать подбором наибольший делитель числа 378, начиная с 19 «вниз», на уменьшение)
7. таким образом, верный ответ – 18.

### Еще пример задания:

**Р-11**. *Укажите через запятую в порядке возрастания все десятичные числа, не превосходящие* 25*, запись которых в системе счисления с основанием четыре оканчивается на* 11*?*

**Общий подход:**

* вспомним алгоритм перевода числа из десятичной системы в систему с основанием  (см. презентацию), из него следует, что младшая цифра результата – это остаток от деления исходного числа на , а две младших цифры – это остаток от деления на  и т.д.
* в данном случае , остаток от деления числа на  должен быть равен 114 = 5
* потому задача сводится к тому, чтобы определить все числа, которые меньше или равны 25 и дают остаток 5 при делении на 16

**Решение (вариант 1, через десятичную систему):**

1. общий вид чисел, которые дают остаток 5 при делении на 16:



где  – целое неотрицательное число (0, 1, 2, …)

1. среди всех таких чисел нужно выбрать те, что меньше или равны 25 («не превосходят 25»); их всего два: 5 (при ) и 21 (при )
2. таким образом, верный ответ – 5, 21 .

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + выражение «не превосходящие » означает «меньшие или равные », а не строго меньшие   + остаток, состоящий из нескольких цифр (здесь – 114), нужно не забыть перевести в десятичную систему   + найденные числа нужно записать именно в порядке возрастания, как требуется |

**Решение (вариант 2, через четверичную систему, предложен О.А. Тузовой):**

1. переведем 25 в четверичную систему счисления: 25 = 1214, все интересующие нас числа не больше этого значения
2. из этих чисел выделим только те, которые заканчиваются на 11, таких чисел всего два:   
   это 114 = 5 и 1114 = 21
3. таким образом, верный ответ – 5, 21 .

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + есть риск случайно «забыть» какое-то число или найти «лишнее» (в данном случае – большее 25)   + можно сделать ошибки при переводе чисел из четверичной системы в десятичную или вообще «забыть» перевести |

### Еще пример задания:

**Р-10**. *Укажите через запятую в порядке возрастания все основания систем счисления, в которых запись числа* 23 *оканчивается на* 2*.*

**Общий подход:**

* здесь обратная задача – неизвестно основание системы счисления, мы обозначим его через 
* поскольку последняя цифра числа – 2, основание должно быть больше 2, то есть 
* вспомним алгоритм перевода числа из десятичной системы в систему с основанием  (см. презентацию), из него следует, что младшая цифра результата – это остаток от деления исходного числа на 

**Решение:**

1. итак, нужно найти все целые числа , такие что остаток от деления 23 на  равен 2, или (что то же самое)

 (\*)

где  – целое неотрицательное число (0, 1, 2, …);

1. сложность в том, что и , и  неизвестны, однако здесь нужно «играть» на том, что это *натуральные числа*
2. из формулы (\*) получаем , так что задача сводится к тому, чтобы найти все делители числа 21, которые больше 2
3. в этой задаче есть только три таких делителя:  и 
4. таким образом, верный ответ – 3, 7, 21 .

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + нужно учесть, что основание системы счисления должно быть *больше* любой цифры числа, поэтому делитель  не подходит (должно быть )   + числа нужно записывать в ответе в порядке возрастания, как требуется по условию |

### Еще пример задания:

**Р-9**. *Укажите через запятую в порядке возрастания все основания систем счисления, в которых запись числа* 31 *оканчивается на* 11*.*

**Общий подход:**

* неизвестно основание системы счисления, мы обозначим его через 
* пока будем считать, что запись числа 31 в системе с основанием  состоит из трех цифр, причем две младшие (11) нам даны, а одну (обозначим ее через ) нужно найти:

2 1 0 ← разряды

**31 = k 1 1N = k·N2 + N1 + N0 = k·N2 + N + 1**

* можно показать, что при большем количестве разрядов эта формула также верна, то есть, число 31 можно представить как  при некотором целом ; например, для числа с пятью разрядами получаем:

4 3 2 1 0 ← разряды

**31 = k4 k3 k2 1 1N = k4·N4 + k3·N3 + k2·N2 + N1 + N0**

**= k·N2 + N + 1**

для  (из первых трех слагаемых вынесли общий множитель )

**Решение:**

1. итак, нужно найти все целые числа , такие что

 (\*\*)

где  – целое неотрицательное число (0, 1, 2, …);

1. сложность в том, что и , и  неизвестны, однако здесь нужно «играть» на том, что это *натуральные числа*
2. из формулы (\*\*) получаем , так что задача сводится к тому, чтобы найти все делители  числа 30 и отобрать только те из них, для которых уравнение (\*\*) разрешимо при целом , то есть,  – целое число
3. выпишем все делители числа 30, большие или равные 2: 2, 3, 5, 6, 10, 15, 30
4. из всех этих делителей только для 2, 3, 5 и 30 значение  – целое число (оно равно соответственно 7, 3, 1 и 0)
5. таким образом, верный ответ – 2, 3, 5, 30.

### Еще пример задания:

**Р-8**. *Укажите, сколько всего раз встречается цифра 2 в записи чисел 10, 11, 12, …, 17 в системе счисления с основанием 5.*

**Решение (вариант 1):**

1. запишем первое и последнее число в заданном диапазоне в системе счисления с основанием 5:

10 = 205, 17 = 325 .

1. заметим, что оба они содержат цифру 2, так что, 2 цифры мы уже нашли
2. между 205 и 325 есть еще числа

215, 225, 235, 245, 305, 315.

1. в них 5 цифр 2 (в числе 225 – сразу две двойки), поэтому всего цифра 2 встречается 7 раз
2. таким образом, верный ответ – 7.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + нужно не забыть, что в системе счисления с основанием 5 старшая цифра – 4, то есть, вслед за 245 следует 305   + помните, что нужно определить не количество чисел, в которых есть двойка, а количество самих двоек   + можно не обратить внимание на то, что в числе 225 цифра 2 встречается 2 раза |

**Решение (вариант 2):**

1. переведем все указанные числа в систему счисления с основанием 5:

10 = 205, 11 = 215, 12 = 225, 13 = 235, 14 = 245, 15 = 305, 16 = 315, 17 = 325 .

1. считаем цифры 2 – получается 7 штук
2. таким образом, верный ответ – 7 .

### Еще пример задания:

**Р-7**. *Укажите наименьшее основание системы счисления, в которой запись числа 30 трехзначна.*

**Решение:**

1. обозначим через  неизвестное основание системы счисления, тогда запись числа 30 в этой системе имеет вид



1. вспомним алгоритм перевода числа из системы счисления с основанием в десятичную систему: расставляем сверху номера разрядов и умножаем каждую цифру на основание в степени, равной разряду:



1. поскольку запись трехзначная, , поэтому 
2. с другой стороны, четвертой цифры нет, то есть, в третьем разряде – ноль, поэтому 
3. объединяя последние два условия, получаем, что искомое основание  удовлетворяет двойному неравенству



1. учитывая, что  – целое число, методом подбора находим целые решения этого неравенства; их два – 4 и 5:





1. минимальное из этих значений – 4
2. таким образом, верный ответ – 4 .

**Решение (без подбора):**

1. выполним п.1-4 так же, как и в предыдущем варианте решения
2. найдем первое целое число, куб которого больше 30; это 4, так как



1. проверяем второе неравенство: , поэтому в системе счисления с основанием 4 запись числа 30 трехзначна
2. таким образом, верный ответ – 4 .

### Еще пример задания:

**Р-6**. *Укажите через запятую в порядке возрастания все десятичные числа, не превосходящие* 30*, запись которых в системе счисления с основанием 5 начинается на 3?*

**Решение (вариант 1):**

1. нас интересуют числа от 1 до 30
2. сначала определим, сколько цифр может быть в этих числах, записанных в системе счисления с основанием 5
3. поскольку , в интересующих нас числах может быть от 1 до 3 цифр
4. рассмотрим трехзначные числа, начинающиеся на 3 в системе с основанием 5:



все они заведомо не меньше , поэтому в наш диапазон не попадают;

1. таким образом, остается рассмотреть только однозначные и двухзначные числа
2. есть всего одно однозначное число, начинающееся на 3, это 3
3. общий вид всех двузначных чисел, начинающихся на 3 в системе с основанием 5:



где  – целое число из множества {0, 1, 2,3,4} (поскольку система счисления имеет основание 5 и цифр, больших 4, в записи числа быть не может)

1. используя эту формулу, находим интересующие нас двузначные числа – 15, 16, 17, 18 и 19
2. таким образом, верный ответ – 3, 15, 16, 17, 18, 19 .

**Решение (вариант 2, предложен Сенькиной Т.С., г. Комсомольск-на-Амуре ):**

1. нас интересуют числа от 1 до 30; сначала определим, сколько цифр может быть в пятеричной записи эти чисел
2. поскольку , в интересующих нас числах может быть не более 2 цифр (все трехзначные пятеричные числа, начинающиеся с 3, больше 30)
3. есть всего одно однозначное число, начинающееся на 3, это 3
4. выпишем все пятеричные двузначные числа, которые начинаются с 3, и переведем их в десятичную систему: 305 = 15, 315 = 16, 325 = 17, 335 = 18 и 345 = 19
5. таким образом, верный ответ – 3, 15, 16, 17, 18, 19 .

### Еще пример задания:

**Р-5**. *Укажите через запятую в порядке возрастания все основания систем счисления, в которых запись числа 71 оканчивается на 13.*

**Решение (1 способ):**

1. Если число в системе с основанием  оканчивается на 13, то
   1. , потому что в системах с меньшим основанием нет цифры 3
   2. это число можно представить в виде , где – целое неотрицательное число
2. определим наибольшее возможное  с учетом условия . Из уравнения  следует .
3. очевидно, что чем меньше , тем больше , поэтому значение  не превышает 

здесь мы подставили  – наименьшее допустимое значение 

1. остается перебрать все допустимые значения  (от 0 до ), решая для каждого из них уравнение

 или равносильное 

относительно , причем нас интересуют только натуральные числа 

1. получаем
   1. при : 
   2. при : решения – не целые числа
   3. при :  и , второе решение не подходит
2. таким образом, верный ответ: 4, 68.

**Решение (2 способ, М.В. Кузнецова и её ученики):**

1. запись числа71 в системе с основанием  оканчивается на 13, т.е. в разряде единиц – 3, это значит, что остаток от деления 71 на  равен 3, то есть для некоторого целогоимеем



1. таким образом, искомые основания – делители числа 68; остается выбрать из них те, которые соответствуют другим условиям задачи
2. среди чисел, оканчивающихся на 13 в системе счисления с основанием ,минимальное – это само число ; отсюда найдем максимальное основание:



так что первый ответ: 68.

1. остальные числа, окачивающиеся в этой системе на 13, имеют не менее 3-х знаков (,…), т.е. все они больше 
2. поэтому , следовательно, 
3. по условию в записи числа есть цифра 3, поэтому  (в системах с основанием ≤ 3 цифры 3 нет)
4. итак: , и при этом  – делитель 68; единственное возможное значение  (на 5,6,7 и 8 число 68 не делится)
5. таким образом, верный ответ: 4, 68.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + на шаге 1 нужно вычесть из числа только число единиц, то есть младшую из двух заданных цифр (в примере – 3)   + можно забыть рассмотреть двузначное число, записанное заданными в условии цифрами (в примере – 13x ), и пропустить максимальное основание   + нужно помнить, что  1. максимальная цифра на 1 меньше основания системы счисления 2. 100 в системе с основанием ***p*** равно ***p2*** |

### Еще пример задания:

**Р-4**. *Укажите через запятую в порядке возрастания все основания систем счисления, в которых запись числа 86 оканчивается на 22.*

**Решение (1 способ):**

1. Если число в системе с основанием  оканчивается на 22, то
   1. , потому что в системах с меньшим основанием нет цифры 2
   2. это число можно представить в виде , где – целое неотрицательное число
2. определим наибольшее возможное  с учетом условия . Из уравнения  следует .
3. очевидно, что чем меньше , тем больше , поэтому значение  не превышает 

здесь мы подставили  – наименьшее допустимое значение 

1. остается перебрать все допустимые значения  (от 0 до ), решая для каждого из них уравнение

 или равносильное 

относительно , причем нас интересуют только натуральные числа 

1. получаем
   1. при : 
   2. при : решения – не целые числа
   3. при :  и , второе решение не подходит
   4. при : решения – не целые числа
2. таким образом, верный ответ: 6, 42.

**Решение (2 способ, М.В. Кузнецова и её ученики):**

1. запись числа 86 в системе с основанием  оканчивается на 22, т.е. в разряде единиц – 2, это значит, что остаток от деления 86 на  равен 2, то есть для некоторого целогоимеем



1. таким образом, искомые основания – делители числа 84; остается выбрать из них те, которые соответствуют другим условиям задачи
2. среди чисел, оканчивающихся на 22 в системе счисления с основанием ,минимальное – это само число ; отсюда найдем максимальное основание:



так что первый ответ: 42.

1. остальные числа, окачивающиеся в этой системе на 22, имеют не менее 3-х знаков (,…), т.е. все они больше 
2. поэтому , следовательно, 
3. по условию в записи числа есть цифра 2, поэтому 
4. итак: , и при этом  – делитель 84; возможные значения  (на 5,8 и 9 число 84 не делится)
5. переводя число 86 в системы счисления с основаниями , находим, что только для основания 6 запись числа оканчивается на 22 (при делении на 3, 4 и 7 «вторые» остатки не равны 2):

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 6 |  | 3 |  |  |  | *Дальше делить  нет смысла* | 8 | 6 |  | 4 |  |  |  |  |  |  | **8** | **6** |  | **6** |  |  |  |  | 8 | 6 |  | 7 |  |  |  |  |
| 8 | 4 |  | 2 | 8 |  | 3 | 8 | 4 |  | 2 | 1 |  |  | 4 |  |  | **8** | **4** |  | **1** | **4** |  | **6** |  | 8 | 4 |  | 1 | 2 |  | 7 |  |
|  | 2 |  | 2 | 7 |  | 9… |  | 2 |  | 2 | 0 |  |  | 5… |  |  |  | **2** |  | **1** | **2** |  | **2** |  |  | 2 |  |  | 7 |  | 1 |  |
|  |  |  |  | **1** |  |  |  |  |  |  | **1** |  |  |  |  |  |  |  |  |  | **2** |  |  |  |  |  |  |  | **5** |  |  |  |

1. таким образом, верный ответ: 6, 42.

### Еще пример задания:

**Р-3**. *Укажите через запятую в порядке возрастания все основания систем счисления, в которых запись числа 94 начинается на 23.*

**Решение:**

1. Из условия сразу видно, что искомое основание не меньше 4 (в записи есть цифра 3).
2. Если запись числа 94 в некоторой системе счисления с основанием  двузначна (94 = 23x), то справедливо равенство ; нас интересуют натуральные решения этого уравнения, такие что , таких решений нет.
3. Предположим, что число четырехзначное. Минимальное допустимое четырехзначное число – 2300x, где . При минимальном основании () оно равно, поэтому запись нужного нам числа имеет не больше трех знаков.
4. На основании (2) и (3) делаем вывод, что число трехзначное, то есть , где – целое неотрицательное число, такое что .
5. Максимальное  можно определить как решение уравнения  (при ); получаем одно из решений – 6,15; поэтому 
6. Если мы знаем , то определится как ; пробуем подставлять в эту формулу , пытаясь получить 
7. Минимальное  будет при : , а при  получается 
8. Таким образом, верный ответ: 6.

### Еще пример задания:

**Р-2**. *Найти сумму восьмеричных чисел 178 +1708 +17008 +...+17000008, перевести в 16-ую систему счисления. Найдите в записи числа, равного этой сумме, третью цифру слева.*

**Решение:**

1. Несложно выполнить прямое сложение восьмеричных чисел, там быстро обнаруживается закономерность:

*178 + 1708 = 2078*

*178 + 1708 + 17008 = 21078*

*178 + 1708 + 17008 + 170008 = 211078*

*178 + 1708 + 17008 + 170008 + 1700008 = 2111078*

*178 + 1708 + 17008 + 170008 + 1700008 + 17000008 = 21111078*

1. Переведем последнюю сумму через триады в двоичный код (заменяем каждую восьмеричную цифру на 3 двоичных):

100010010010010001112

1. Теперь разбиваем цепочку на тетрады (группы из 4-х двоичных цифр), начиная справа, и каждую тетраду представляем в виде шестнадцатеричной цифры

100010010010010001112

8 9 2 4 7

1. Таким образом, верный ответ (третья цифра слева): 2.

### Еще пример задания:

**Р-1**. *Чему равно наименьшее основание позиционной системы счисления* *, при котором 225x = 405y? Ответ записать в виде целого числа.*

**Решение:**

1. Поскольку в левой и в правой частях есть цифра 5, оба основания больше 5, то есть перебор имеет смысл начинать с .
2. Очевидно, что , однако это не очень нам поможет.
3. Для каждого «подозреваемого»  вычисляем значение и решаем уравнение , причем нас интересуют только натуральные .
4. Для  и  нужных решений нет, а для  получаем



так что.

1. Таким образом, верный ответ (минимальное значение ): 8.

### Еще пример задания:

**Р-0**. *Запись числа 3010 в системе счисления с основанием N оканчивается на 0 и содержит 4 цифры. Чему равно основание этой системы счисления N?*

**Решение (1 способ, подбор):**

1. запись числа 30 в системе с основанием N длиннее, чем в десятичной (4 цифры против двух), поэтому основание N меньше 10
2. это дает шанс решить задачу методом подбора, переводя в разные системы, начиная с N = 2 до N = 9
3. переводим:

30 = 111102 = 10103 = …

1. дальше можно не переводить, поскольку запись 10103 удовлетворяет условию: заканчивается на 0 и содержит 4 цифры
2. можно проверить, что при N ≥ 4 запись числа 30 содержит меньше 4 цифр, то есть не удовлетворяет условию
3. Ответ: 3.

**Решение (2 способ, неравенства):**

1. запись числа 30 в системе с основанием N содержит ровно 4 цифры тогда и только тогда, когда старший разряд – третий, то есть



1. первая часть двойного неравенства  дает (в целых числах) 
2. вторая часть неравенства  дает (в целых числах) 
3. объединяя результаты пп. 2 и 3 получаем, что N = 3
4. заметим, что условие «оканчивается на 0» – лишнее, ответ однозначно определяется по количеству цифр
5. Ответ: 3.

# 17 (повышенный уровень, время – 2 мин)

**Тема**: Составление запросов для поисковых систем с использованием логических выражений.

**Что нужно знать**:

* таблицы истинности логических операций «И», «ИЛИ», «НЕ» (см. презентацию «Логика»)
* если в выражении нет скобок, сначала выполняются все операции «НЕ», затем – «И», затем – «ИЛИ»
* логическое произведение A∙B∙C∙… равно 1 (выражение истинно) только тогда, когда все сомножители равны 1 (а в остальных случаях равно 0)
* логическая сумма A+B+C+… равна 0 (выражение ложно) только тогда, когда все слагаемые равны 0 (а в остальных случаях равна 1)
* правила преобразования логических выражений (законы алгебры логики):

|  |  |  |
| --- | --- | --- |
| Закон | Для **И** | Для **ИЛИ** |
| двойного отрицания |  | |
| исключения третьего |  |  |
| исключения констант | A · 1 = A; A · 0 = 0 | A + 0 = A; A + 1 = 1 |
| повторения | A · A = A | A + A = A |
| поглощения | A · (A + B) = A | A + A · B = A |
| переместительный | A · B = B · A | A + B = B + A |
| сочетательный | A · (B · C) = (A · B) · C | A + (B + C) = (A + B) + C |
| распределительный | A + B · C = (A + B) · (A + C) | A · (B + C) = A · B + A · C |
| де Моргана |  |  |

* ввод какого-то слова (скажем, **кергуду**) в запросе поисковой системы означает, что пользователь ищет Web-страницы, на которых встречается это слово
* операция «И» всегда **ограничивает** поиск, то есть, в ответ на запрос **кергуду И бамбарбия** поисковый сервер выдаст **меньше** страниц, чем на запрос **кергуду**, потому что будет искать страницы, на которых есть оба этих слова одновременно
* операция «ИЛИ» всегда **расширяет** поиск, то есть, в ответ на запрос   
  **кергуду ИЛИ бамбарбия** поисковый сервер выдаст **больше** страниц, чем на запрос **кергуду**, потому что будет искать страницы, на которых есть хотя бы одно из этих слов (или оба одновременно)
* если в запросе вводится фраза в кавычках, поисковый сервер ищет страницы, на которых есть в точности эта фраза, а не просто отдельные слова; взятие словосочетания в кавычки **ограничивает** поиск, то есть, в ответ на запрос **"кергуду бамбарбия"** поисковый сервер выдаст **меньше** страниц, чем на запрос **кергуду бамбарбия**, потому что будет искать только те страницы, на которых эти слова стоят одно за другим

### Пример задания:

*В таблице приведены запросы и количество найденных по ним страниц некоторого сегмента сети Интернет:*

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| США |Япония | Китай | 450 |
| Япония | Китай | 260 |
| (США & Япония) | (США & Китай) | 50 |

*Сколько страниц*  *(в тысячах)* *будет найдено по запросу*

**США**

*Считается, что все запросы выполнялись практически одновременно, так что набор страниц, содержащих все искомые слова, не изменялся за время выполнения запросов.*

**Решение:**

1. заметим, что в силу тождества  последний запрос в таблице равносилен такому:

(США & Япония) | (США & Китай) ⇔ США & (Япония | Китай)

1. тогда вводя обозначение для областей

A = США, B = Япония | Китай,

получаем стандартную задачу с двумя переменными:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| А | B | 450 |
| B | 260 |
| А & B | 50 |
| А | ? |

1. имеем по формуле (см. решения ниже)

**NA** **=** **NA|B** **-** **NB** + **NA&B= 450 – 260 + 50 = 240**

1. Ответ: 240

### Ещё пример задания:

*В таблице приведены запросы и количество найденных по ним страниц некоторого сегмента сети Интернет:*

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| Ростов & (Орёл & Курск | Белгород) | 370 |
| Ростов & Белгород | 204 |
| Ростов & Орёл & Курск & Белгород | 68 |

*Сколько страниц*  *(в тысячах)* *будет найдено по запросу*

**Ростов & Орёл & Курск**

*Считается, что все запросы выполнялись практически одновременно, так что набор страниц, содержащих все искомые слова, не изменялся за время выполнения запросов.*

**Решение:**

1. заметим, что во всех четырёх запросах есть «сомножитель» «Ростов &», поэтому эта задача равносильна такой:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| Орёл & Курск | Белгород | 370 |
| Белгород | 204 |
| Орёл & Курск & Белгород | 68 |
| Орёл & Курск | ? |

1. теперь обозначим A = Орёл & Курск и получим задачу с двумя областями:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| A | Белгород | 370 |
| Белгород | 204 |
| A & Белгород | 68 |
| A | ? |

1. по формуле для задачи с двумя областями (см. задачи, разобранные ниже)

**NA|B** **=** **NA** **+** **NB** **-** **NA&B**

получаем

**NA** **=** **NA|B** **-** **NB** + **NA&B**

1. вычисляем: 370 – 204 + 68 = 234.
2. Ответ: 234.

### Ещё пример задания:

*В таблице приведены запросы и количество найденных по ним страниц некоторого сегмента сети Интернет:*

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| Ухо | 35 |
| Подкова | 25 |
| Наковальня | 40 |
| Ухо | Подкова | Наковальня | 70 |
| Ухо & Наковальня | 10 |
| Ухо & Подкова | 0 |

*Сколько страниц*  *(в тысячах)* *будет найдено по запросу*

**Подкова & Наковальня**

*Считается, что все запросы выполнялись практически одновременно, так что набор страниц, содержащих все искомые слова, не изменялся за время выполнения запросов.*

**Решение (вариант 1, рассуждения по диаграмме):**

1. построим диаграмму Эйлера-Венна

*Ухо*

*Подкова*

*Наковальня*

1

2

3

4

5

1. количество сайтов, удовлетворяющих запросу в области **i**, будем обозначать через **Ni**
2. здесь 5 областей, причём известны следующие данные:



1. нас интересует область 4. Находим ответ прямой подстановкой:



1. таким образом, ответ – 20.

**Решение (вариант 2, рассуждения по диаграмме):**

1. пп. 1-2 такие же, как в варианте 1
2. заметим, что в прямую сумму величин областей Ухо, Подкова и Наковальня дважды входят области 2 и 4, поэтому для вычисления  достаточно вычесть из суммы Ухо+Подкова+Наковальня размер их объединения (Ухо | Подкова | Наковальня) и величину области 2 (Ухо & Наковальня).
3. тогда сразу получаем

.

1. ответ – 20.

### Еще пример задания:

*В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:*

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| пирожное & выпечка | 3200 |
| пирожное | 8700 |
| выпечка | 7500 |

*Сколько страниц*  *(в тысячах)* *будет найдено по запросу*

**пирожное | выпечка**

**Решение (вариант 1, рассуждения по диаграмме):**

П

В

1

2

3

1. построим диаграмму Эйлера-Венна, обозначив области «пирожное» (через П) и «выпечка» (В) :
2. количество сайтов, удовлетворяющих запросу в области **i**, будем обозначать через **Ni**
3. несложно сообразить, что число сайтов в интересующей нас области равно

**N1 + N2 + N3 = (N1 + N2)+ (N3 + N2) – N2**

1. поскольку нам известно, что по условию

**N1 + N2 = 8700**

**N3 + N2 = 7500**

**N2 = 3200**

сразу получаем

**N1 + N2 + N3 = 8700 + 7500 - 3200 = 13000**

1. таким образом, ответ – 13000.

**Решение (вариант 2, общая формула):**

A

В

1. сначала выведем формулу, о которой идет речь; построим диаграмму Эйлера-Венна для двух переменных **A** и **B**:
2. обозначим через **NA**, **NB**, **NA&B** и **NA|B** число страниц, которые выдает поисковый сервер соответственно по запросам **A**, **B**, **A & B** и   
   **A | B**
3. понятно, что если области **A** и **B** не пересекаются, справедлива формула **NA|B=NA+NB**
4. если области пересекаются, в сумму **NA+NB** область пересечения **NA&B** входит дважды, поэтому в общем случае

**NA|B** **=** **NA** **+** **NB** **-** **NA&B**

1. в данной задаче

**NП = 8700, NВ = 7500, NП&В = 3200**

1. тогда находим число сайтов в интересующей нас области по формуле

**NП|B** **=** **NП** **+** **NB** **–** **NП&B = 8700 + 7500 – 3200 = 13000**

1. таким образом, ответ – 13000.

**Решение (вариант 3, решение системы уравнений):**

П

В

1

2

3

1. нарисуем области «пирожное» (обозначим ее через П) и «выпечка» (В) в виде диаграммы (кругов Эйлера); при их пересечении образовались три подобласти, обозначенные числами 1, 2 и 3;
2. составляем уравнения, которые определяют запросы, заданные в условии:

**пирожное & выпечка N2 = 3200**

**пирожное N1 + N2 = 8700**

**выпечка N2 + N3 = 7500**

1. подставляя значение **N2** из первого уравнения в остальные, получаем

**N1 = 8700 - N2 = 8700 – 3200 = 5500**

**N3 = 7500 - N2 = 7500 – 3200 = 4300**

1. количество сайтов по запросу **пирожное | выпечка** равно

**N1 + N2 + N3 = 5500 + 3200 + 4300 = 13000**

1. таким образом, ответ – 13000.

### Еще пример задания:

*В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:*

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| **Динамо & Рубин** | 320 |
| **Спартак & Рубин** | 280 |
| **(Динамо | Спартак) & Рубин** | 430 |

*Сколько страниц*  *(в тысячах)* *будет найдено по запросу*

**Рубин & Динамо & Спартак**

**Решение (вариант 1, круги Эйлера, полная диаграмма):**

Динамо

Рубин

3

1

Спартак

2

1. в этой задаче неполные данные, так как они не позволяют определить размеры всех областей; однако их хватает для того, чтобы ответить на поставленный вопрос
2. обозначим области, которые соответствуют каждому запросу

|  |  |  |
| --- | --- | --- |
| ***Запрос*** | ***Области*** | ***Количество страниц (тыс.)*** |
| **Динамо & Рубин** | **1+2** | 320 |
| **Спартак & Рубин** | **2+3** | 280 |
| **(Динамо | Спартак) & Рубин** | **1+2+3** | 430 |
| **Рубин & Динамо & Спартак** | **2** | ? |

1. из таблицы следует, что в суммарный результат первых двух запросов область 2 входит дважды (1 + 2 + 2 + 3), поэтому, сравнивая этот результат с третьим запросом (1 + 2 + 3), сразу находим результат четвертого:

N2 = (320 + 280) – 430 = 170

1. таким образом, ответ – 170.

**Решение (вариант 2, круги Эйлера, неполная диаграмма):**

Динамо

Рубин

3

1

Спартак

2

1. заметим, что в этой задаче все запросы (в том числе и тот, результат которого нужно найти, имеют вид

**X & Рубин**

1. поэтому часть «**& Рубин**» в каждом из запросов можно просто отбросить, тогда останется только две области:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| **Динамо-1** | 320 |
| **Спартак-1** | 280 |
| **Динамо-1 | Спартак-1** | 430 |

здесь добавление «-1» в имени области обозначает «пересечение с областью **Рубин**»

1. требуется найти размер области «**Динамо-1 & Спартак-1**»
2. для диаграммы с двумя областями можно использовать общую формулу

**NA|B** **=** **NA** **+** **NB** **-** **NA&B**

1. из которой следует

**NA&B** **=** **NA** **+** **NB** **-** **NA|B**

1. в данном случае получаем

**NA&B** = (320 + 280) – 430 = 170

1. таким образом, ответ – 170.

### Ещё пример задания:

*В таблице приведены запросы к поисковому серверу. Расположите номера запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу. Для обозначения логической операции «ИЛИ» в запросе используется символ* |*, а для логической операции «И» – &.*

1) **принтеры & сканеры & продажа**

2) **принтеры & сканеры**

3) **принтеры | сканеры**

4) **принтеры | сканеры | продажа**

**Решение (вариант 1, рассуждение с использованием свойств операций «И» и «ИЛИ»):**

1. меньше всего результатов выдаст запрос с наибольшими ограничениями – первый (нужны одновременно принтеры, сканеры и продажа)
2. на втором месте – второй запрос (одновременно принтеры и сканеры)
3. далее – третий запрос (принтеры или сканеры)
4. четвертый запрос дает наибольшее количество результатов (принтеры или сканеры или продажа)
5. таким образом, верный ответ – 1234 .

|  |
| --- |
| **Возможные проблемы**:   * + нужно внимательно читать условие, так как в некоторых задачах требуется перечислить запросы в порядке убывания количества результатов, а в некоторых – в порядке возрастания   + можно ошибиться в непривычных значках: «И» = &, «ИЛИ» = | (эти обозначения привычны для тех, кто программирует на языке Си)   + можно перепутать значение операций «И» и «ИЛИ», а также порядок выполнения цепочки операций (сначала – «И», потом – «ИЛИ»)   + для сложных запросов не всегда удастся так просто расположить запросы по возрастанию (или убыванию) ограничений |

**Решение (вариант 2, через таблицы истинности):**

1. каждое из условий можно рассматривать как сложное высказывание
2. обозначим отдельные простые высказывания буквами:

**A:** **принтеры** (на странице есть слово «принтеры»)

**B: сканеры**

**C: продажа**

1. запишем все выражения-запросы через логические операции

, , , 

1. здесь присутствуют три переменные, А, B и C (хотя второе и третье выражения от С не зависят!), поэтому для составления таблицы истинности нужно рассмотреть 8 = 232333 всевозможных комбинаций этих логических значений
2. выражение  равно 1 (истинно) только при , в остальных случаях – равно 0 (ложно)
3. выражение  равно 1 только при , в остальных случаях – равно 0
4. выражение  равно 0 только при , в остальных случаях – равно 1
5. выражение  равно 0 только при , в остальных случаях –  1
6. запишем результаты пп. 5-8 в виде таблицы истинности

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A | B | C |  |  |  |  |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |

1. по таблице видим, что наименьшая «область действия» у первого выражения, поисковый сервер выдаст наименьшее число запросов
2. область, где , включает в себя[[13]](#footnote-13) всю область, где  и еще один вариант, поэтому «поисковик» выдаст больше запросов, чем для первого случая
3. аналогично делаем вывод, что область  включает всю область  и расширяет ее, а область  – это расширение области 
4. таким образом, верный ответ – 1234 .

|  |
| --- |
| **Возможные проблемы**:   * + решение достаточно громоздко, хотя позволяет с помощью простых операций решить задачу, не рискуя ошибиться при вычислениях «в уме» в сложных случаях   + если переменных более трех, таблица получается большая, хотя заполняется несложно |

**Решение (вариант 3, через диаграммы):**

1. запишем все ответы через логические операции

, , , 

1. покажем области, определяемые этими выражениями, на диаграмме с тремя областями

A

B

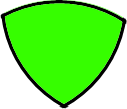
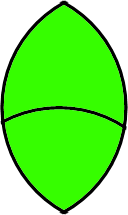
С



A

B

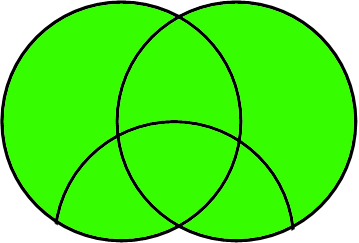
С



A

B

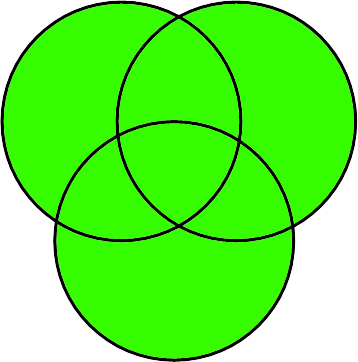
С



A

B

С



1. сравнивая диаграммы, находим последовательность областей в порядке увеличения: (1,2,3,4), причем каждая следующая область в этом ряду охватывает целиком предыдущую (как и предполагается в задании, это важно!)
2. таким образом, верный ответ – 1234 .

|  |
| --- |
| **Возможные проблемы**:   * + получается громоздкий рисунок, если используется более трех переменных (более трех кругов) |

### Еще 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, для которых данное слово является ключевым*** |
| *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 (вариант 1, использованием свойств операций «И» и «ИЛИ»):**

1. обратим внимание на такой факт[[14]](#footnote-14) (справа указано количество сайтов по каждому запросу)

**сканер 200**

**принтер 250**

**принтер | сканер 450**

поскольку последнее число равно сумме двух предыдущих, можно сразу же придти к выводу, что в этом сегменте сети нет сайтов, на которых ключевыми словами являются одновременно принтер и сканер:

С

П

М

**(П | С) & M**

**принтер & сканер 0**

диаграмма Эйлера для этого случая показана на рисунке справа:

1. с этого момента все просто: для того, чтобы определить, сколько сайтов удовлетворяют заданному условию

достаточно просто сложить числа, соответствующие запросам **принтер & монитор** и   
**сканер & монитор**

1. таким образом, правильный ответ: 40 + 50 = 90.

|  |
| --- |
| **Возможные проблемы**:   * + обратите внимание, что в условии была лишняя информация: мы нигде не использовали количество сайтов в данном сегменте Интернета (1000) и количество сайтов с ключевым словом *монитор* (450)   + не всегда удается «раскрутить» задачу в уме, здесь это несложно благодаря «удачному» условию |

**Решение (вариант 3, таблицы истинности):**

1. для сокращения записи обозначим через C, П, М высказывания «ключевое слово на сайте – *сканер*» (соответственно *принтер*, *монитор*)
2. если рассматривать задачу с точки зрения математической логики, здесь есть три переменных, с помощью которых можно составить всего 8 запросов, выдающих различные результаты

|  |  |  |  |
| --- | --- | --- | --- |
|  | **С** | **П** | **М** |
| ? | 0 | 0 | 0 |
| ? | 0 | 0 | 1 |
| ? | 0 | 1 | 0 |
| ? | 0 | 1 | 1 |
| ? | 1 | 0 | 0 |
| ? | 1 | 0 | 1 |
| ? | 1 | 1 | 0 |
| ? | 1 | 1 | 1 |
| всего | 200 | 250 | 450 |

1. составим таблицу истинности, в которую добавим левый столбец и последнюю строку, где будем записывать количество сайтов, удовлетворяющих условиям строки и столбца (см. рисунок справа); например, первая строка соответствует сайтам, на которых нет ни одного из заданных ключевых слов; такая схема непривычна, но она существенно упрощает дело
2. сумма в последней строчке получается в результате сложения всех чисел из тех строк первого столбца, где в данном столбце стоят единицы. Например, сумма в столбце **С** – складывается из четырех чисел в последних четырех строчках первого столбца. Мы пока не знаем, сколько результатов возвращает каждый из восьми запросов отдельно, поэтому в первом столбце стоят знаки вопроса
3. добавим в таблицу истинности остальные запросы, которые есть в условии, в том числе и тот, который нас интересует:

**П | С = принтер | сканер 450**

**П & М = принтер & монитор 40**

**C & М = сканер & монитор 50**

**(П | C) & М = (принтер | сканер) & монитор ?**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **С** | **П** | **М** | **П | С** | **П & М** | **C & М** | **(П | C) & М** |
| ? | 0 | 0 | 0 | 0 |  |  |  |
| ? | 0 | 0 | 1 | 0 |  |  |  |
| ? | 0 | 1 | 0 | 1 |  |  |  |
| ? | 0 | 1 | 1 | 1 |  |  |  |
| ? | 1 | 0 | 0 | 1 |  |  |  |
| ? | 1 | 0 | 1 | 1 |  |  |  |
| **0** | 1 | 1 | 0 | 1 |  |  |  |
| **0** | 1 | 1 | 1 | 1 |  |  |  |
| всего | 200 | 250 | 450 | 450 |  |  |  |

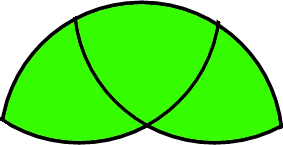
1. проанализируем столбец **П | С** в этой таблице: его сумма (450) складывается из суммы столбцов С (200) и П (250) – выделены ярким зеленым цветом – плюс последние две строчки (голубой фон), то есть, 450 = 200 + 250 + X, откуда сразу получаем, что X = 0, то есть, последним двум строчкам (запросам) не удовлетворяет ни одного сайта
2. теперь составим таблицы истинности для остальных запросов, отбросив заведомо «нулевые» варианты:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **С** | **П** | **М** | **П | С** | **П & М** | **C & М** | **(П | C) & М** |
| ? | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ? | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| ? | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| **40** | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| ? | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| **50** | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| всего | 200 | 250 | 450 | 450 | 40 | 50 | **90** |

из оставшихся шести строк таблицы запросы П & М и С & М затрагивают только по одной строчке, поэтому сразу можем вписать соответствующие числа в первый столбец; в последнем запросе, который нас интересует, присутствуют именно эти две строки, то есть, для получения нужно сложить 40 и 50

1. таким образом, правильный ответ: 40 + 50 = 90.

**Решение (вариант 3, через диаграммы и систему уравнений):**



С

П

М

**(П | С) & M**

1

2

3

4

5

6

7

1. для сокращения записи обозначим через C, П, М высказывания «ключевое слово на сайте – *сканер*» (соответственно *принтер*, *монитор*) и нарисуем эти области виде диаграммы (кругов Эйлера); интересующему нас запросу **(П | C) & M** соответствует объединение областей 4, 5 и 6 («зеленая зона» на рисунке)
2. количество сайтов, удовлетворяющих запросу в области **i**, будем обозначать через **Ni**
3. составляем уравнения, которые определяют запросы, заданные в условии:

**сканер N1 + N2 + N4 + N5 = 200**

**принтер N2 + N3 + N5 + N6 = 250**

**принтер | сканер N1 + N2 + N4 + N5 + N3 + N6 = 450**

из первого и третьего уравнений сразу следует

**200 + N3 + N6 = 450 ⇒ N3 + N6 = 250**

далее из второго уравнения

**N2 + N5 + 250 = 250 ⇒ N2 + N5 = 0**

поскольку количество сайтов не может быть отрицательной величиной, **N2 = N5 = 0**

1. посмотрим, что еще мы знаем (учитываем, что **N5 = 0**):

**принтер & монитор N5 + N6 = 40 ⇒ N6 = 40**

**сканер & монитор N4 + N5 = 50 ⇒ N4 = 50**

1. окончательный результат:

**(принтер | сканер) & монитор N4 + N5 + N6 = N4 + N6 = 40 + 50 = 90**

1. таким образом, правильный ответ 90.

|  |
| --- |
| **Возможные проблемы**:   * + внимательнее с индексами переменных, очень легко по невнимательности написать **N5** вместо **N6** и получить совершенно другой результат   + этот метод ярко демонстрирует, что в общем случае мы получаем систему уравнения с семью неизвестными (или даже с восемью, если задействована еще и область вне всех кругов); решать такую систему вручную достаточно сложно, поэтому на экзамене всегда будет какое-то условие, сильно упрощающее дело, ищите его |

### Еще пример задания:

*В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:*

|  |  |  |  |
| --- | --- | --- | --- |
|  | ***Запрос*** | ***Количество страниц (тыс.)*** | |
| 1 | мезозой | 50 |
| 2 | кроманьонец | 60 |
| 3 | неандерталец | 70 |
| 4 | мезозой | кроманьонец | 80 |
| 5 | мезозой | неандерталец | 100 |
| 6 | неандерталец & (мезозой | кроманьонец) | 20 |

*Сколько страниц*  *(в тысячах)* *будет найдено по запросу*

**кроманьонец & (мезозой | неандерталец)**

**Решение (способ 1, круги Эйлера):**

М

К

Н

**K & (M | Н)**

1

2

3

4

5

6

7

1. обозначим области «мезозой», «кроманьонец» и «неандерталец» буквами М, К и Н; пронумеруем подобласти, получившиеся в результате пересечений кругов (см. рисунок справа)
2. через **N­i** обозначим количество сайтов в области с номером **i**
3. нас интересует результат запроса

**кроманьонец & (мезозой | неандерталец)**

то есть N­2 + N5 + N6(зеленая область на рисунке)

1. из первых двух запросов следует, что

N1 + N2 + N4 + N5 = 50 (мезозой)

N2 + N3 + N5 + N6 = 60 (кроманьонец)

1. складывая левые и правые части уравнений, получаем

(1) N1 + 2·N2 + N3 + N4 + 2·N5 + N6 = 110

1. в то же время из запроса 4 получаем

(2) N1 + N2 + N3 + N4 + N5 + N6 = 80 (мезозой | кроманьонец)

1. вычитая из уравнения (1) уравнение (2), отдельно левые и правые части, получаем

N2 + N5 = 30 (мезозой & кроманьонец)

вспомним, что наша цель – определить N­2 + N5 + N6, поэтому остается найти N6

1. из запросов 1 и 3 следует, что

N1 + N2 + N4 + N5 = 50 (мезозой)

N4 + N5 + N6 + N7 = 70 (неандерталец)

1. складывая левые и правые части уравнений, получаем

(3) N1 + N2 + 2·N4 + 2·N5 + N6 + N7 = 120

1. в то же время из запроса 5 получаем

(4) N1 + N2 + N4 + N5 + N6 + N7 = 100 (мезозой | неандерталец)

1. вычитая из уравнения (3) уравнение (4), отдельно левые и правые части, получаем

(5) N4 + N5 = 20 (мезозой & неандерталец)

1. теперь проанализируем запрос 6:

неандерталец & (мезозой | кроманьонец)

(6) N4 + N5 + N­6 = 20

1. вычитая из уравнения (6) уравнение (5) получаем N6 = 0, поэтому

N2 + N5 + N6 = N2 + N5 = 30

1. таким образом, ответ – 30.

**Решение (способ 2, М.С. Коротков, г. Челябинск, Лицей № 102):**

1. пп. 1-3 такие же, как в первом способе;
2. из запросов 1 и 6 следует, что
3. N4 + N5 + N6 + N7 = 70 (неандерталец)
4. N4 + N5 + N­6 = 20 неандерталец & (мезозой | кроманьонец)
5. вычитая (2) из (1), сразу получаем, что N7 = 50
6. из запросов 5 и 4 следует, что
7. N1 + N2 + N4 + N5 + N6 + N7 = 100 (мезозой | неандерталец)
8. N1 + N2 + N3 + N4 + N5 + N6 = 80 (мезозой | кроманьонец)
9. вычитая (4) из (3), сразу получаем, что N7 - N3 = 20
10. в п. 3 мы уже определили, что N7 = 50, поэтому 50 - N3 = 20, откуда N3 = 30
11. из запроса 2 получаем

N2 + N3 + N5 + N6 = 60 (кроманьонец)

поэтому размер интересующей нас области равен

N2 + N5 + N6 = 60 – N3 = 60 – 30 = 30

1. таким образом, ответ – 30.

М

К

Н

**K & (M | Н)**

**Решение (способ 3, круги Эйлера, И.Б. Курбанова, г. Санкт-Петербург,** **ГОУ СОШ № 594):**

1. обозначим: М – мезозой, К – кроманьонец, Н – неандерталец.
2. нас интересует результат запроса (см. диаграмму Эйлера)

**K & (M | Н)**

1. т.к. по условию М = 50, К = 60, а объединение этих множеств М | К = 80, можно сделать вывод, что область пересечения

М

К

Н

**K & (M | Н)**

**M & K = 50 + 60 – 80 = 30;**

1. т.к. по условию М = 50, Н = 70, а объединение этих множеств М | Н = 100, можно сделать вывод, что область пересечения

**M & Н = 50 + 70 – 100 = 20;**

1. заметим, что M & Н = 20 и Н & (М | К) = 20, следовательно **множества Н и К не пересекаются** (К & Н = 0);
2. перерисуем диаграмму Эйлера так, чтобы множества К и Н не пересекались (см. рисунок справа); из новой схемы видно, что

**К & (М | Н) = (К & М) | (К & Н) = К & М = 30**

1. ответ: 30

Задачи для тренировки[[15]](#footnote-15):

Во всех задачах для обозначения логической операции «ИЛИ» в запросе используется символ |, а для логической операции «И» – символ &.

1. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу.

А) **физкультура**

Б) **физкультура & подтягивания & отжимания**

В) **физкультура & подтягивания**

Г) **физкультура | фитнесс**

1. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу.

А ) **волейбол | баскетбол | подача**

Б) **волейбол | баскетбол | подача | блок**

В) **волейбол | баскетбол**

Г) **волейбол & баскетбол & подача**

1. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу.

A ) **чемпионы | (бег & плавание)**

Б ) **чемпионы & плавание**

В ) **чемпионы | бег | плавание**

Г) **чемпионы & Европа & бег & плавание**

1. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу.

А ) **музыка | классика | Моцарт | серенада**

Б) **музыка | классика**

В) **музыка | классика | Моцарт**

Г) **музыка & классика & Моцарт**

1. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу.

А) **реферат | математика | Гаусс**

Б) **реферат | математика | Гаусс | метод**

В) **реферат | математика**

Г) **реферат & математика & Гаусс**

1. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу. Ответ запишите в виде последовательности соответствующих букв.

a) **Америка | путешественники | Колумб**

b) **Америка | путешественники | Колумб | открытие**

c) **Америка | Колумб**

d) **Америка & путешественники & Колумб**

1. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке убывания количества страниц, которые найдет поисковый сервер по каждому запросу. Ответ запишите в виде последовательности соответствующих букв.

а ) **Информатика & уроки & Excel**

b ) **Информатика | уроки | Excel | диаграмма**

с) **Информатика | уроки | Excel**

d) **Информатика | Excel**

1. В таблице приведены запросы к поисковому серверу, условно обозначенные буквами от А до Г. Расположите запросы в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу. Ответ запишите в виде последовательности соответствующих букв.

А ) **Гренландия & Климат & Флора & Фауна**

Б ) **Гренландия & Флора**

В ) **(Гренландия & Флора) | Фауна**

Г) **Гренландия & Флора & Фауна**

1. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке убывания количества страниц, которые найдет поисковый сервер по каждому запросу.

а) **спорт | футбол**

b) **спорт | футбол | Петербург | Зенит**

с) **спорт | футбол | Петербург**

d) **спорт & футбол & Петербург & Зенит**

1. Каким условием нужно воспользоваться для поиска в сети Интернет информации о цветах, растущих на острове Тайвань или Хонсю

1) **цветы & (Тайвань | Хонсю)**

2) **цветы & Тайвань & Хонсю**

3) **цветы | Тайвань | Хонсю**

4) **цветы & (остров | Тайвань | Хонсю)**

1. 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, для которых данное слово является ключевым*** |
| *сомики* | *250* |
| *меченосцы* | *200* |
| *гуппи* | *500* |

Сколько сайтов будет найдено по запросу

**сомики | меченосцы | гуппи**

если по запросу**сомики & гуппи**было найдено 0 сайтов*,* по запросу**сомики & меченосцы** – 20, а по запросу**меченосцы & гуппи**– 10.

1. 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, для которых данное слово является ключевым*** |
| *сомики* | *250* |
| *меченосцы* | *200* |
| *гуппи* | *500* |

Сколько сайтов будет найдено по запросу

**(сомики & меченосцы) | гуппи**

если по запросу**сомики | гуппи**было найдено 750 сайтов*,* по запросу**сомики & меченосцы** – 100, а по запросу**меченосцы & гуппи**– 0.

1. 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, для которых данное слово является ключевым*** |
| *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.

1. В таблице приведены запросы к поисковому серверу, условно обозначенные буквами от А до Г. Расположите запросы в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу. Ответ запишите в виде последовательности соответствующих букв.

А ) **(огурцы & помидоры) & (прополка | поливка)**

Б ) **огурцы | помидоры**

В ) **огурцы**

Г) **огурцы & помидоры**

1. В таблице приведены запросы к поисковому серверу, условно обозначенные буквами от А до Г. Расположите запросы в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу. Ответ запишите в виде последовательности соответствующих букв.

А ) **экзамен | тестирование**

Б ) **(физика | химия) & (экзамен | тестирование)**

В ) **физика & химия & экзамен & тестирование**

Г) **физика | химия | экзамен | тестирование**

1. В таблице приведены запросы к поисковому серверу, условно обозначенные буквами от А до Г. Расположите запросы в порядке возрастания количества страниц, которые найдет поисковый сервер по каждому запросу. Ответ запишите в виде последовательности соответствующих букв.

А ) **сомики | меченосцы | содержание**

Б ) **сомики & содержание**

В ) **сомики & меченосцы & разведение & содержание**

Г) **(сомики | меченосцы) & содержание**

1. В таблице приведены запросы к поисковому серверу. Расположите номера запросов в порядке **возрастания** количества страниц, которые найдет поисковый сервер по каждому запросу. Для обозначения логической операции «ИЛИ» в запросе используется символ |, а для логической операции «И» – &.

1 ) **канарейки | щеглы | содержание**

2 ) **канарейки & содержание**

3 ) **канарейки & щеглы & содержание**

4) **разведение & содержание & канарейки & щеглы**

1. В таблице приведены запросы к поисковому серверу. Расположите номера запросов в порядке **убывания** количества страниц, которые найдет поисковый сервер по каждому запросу. Для обозначения логической операции «ИЛИ» в запросе используется символ |, а для логической операции «И» – &.

1 ) **барокко | (классицизм & ампир)**

2 ) **барокко | классицизм**

3 ) **барокко | ампир | классицизм**

4) **классицизм & ампир**

1. В таблице приведены запросы к поисковому серверу. Расположите номера запросов в порядке **убывания** количества страниц, которые найдет поисковый сервер по каждому запросу. Для обозначения логической операции «ИЛИ» в запросе используется символ |, а для логической операции «И» – &.

1 ) **барокко | (классицизм & ампир)**

2 ) **барокко | классицизм**

3 ) **(классицизм & ампир) | (барокко & модерн)**

4) **барокко | ампир | классицизм**

1. В таблице приведены запросы к поисковому серверу. Расположите номера запросов в порядке **убывания** количества страниц, которые найдет поисковый сервер по каждому запросу. Для обозначения логической операции «ИЛИ» в запросе используется символ |, а для логической операции «И» – &.

1 ) **зайцы & кролики**

2 ) **зайцы & (кролики | лисицы)**

3 ) **зайцы & кролики & лисицы**

4) **кролики | лисицы**

1. В таблице приведены запросы к поисковому серверу. Расположите номера запросов в порядке **возрастания** количества страниц, которые найдет поисковый сервер по каждому запросу. Для обозначения логической операции «ИЛИ» в запросе используется символ |, а для логической операции «И» – &.

1 ) **кролики | лисицы**

2 ) (**зайцы & кролики**) | **(лисицы & волки)**

3 ) **зайцы & кролики** & **лисицы & волки**

4) **зайцы & кролики**

1. В таблице приведены запросы к поисковому серверу. Расположите номера запросов в порядке **возрастания** количества страниц, которые найдет поисковый сервер по каждому запросу. Для обозначения логической операции «ИЛИ» в запросе используется символ |, а для логической операции «И» – &.

1 ) **шкафы | столы | стулья**

2 ) **шкафы | (стулья & шкафы)**

3 ) **шкафы & столы**

4) **шкафы | стулья**

1. В таблице приведены запросы к поисковому серверу. Расположите номера запросов в порядке **убывания** количества страниц, которые найдет поисковый сервер по каждому запросу. Для обозначения логической операции «ИЛИ» в запросе используется символ |, а для логической операции «И» – &.

1 ) **яблоки | сливы**

2 ) **сливы | (сливы & груши)**

3 ) **яблоки | груши | сливы**

4) **(яблоки | груши) & сливы**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *фрегат* |  *эсминец* | *3000* |
| *фрегат* | *2000* |
| *эсминец* | *2500* |

Сколько страниц (в тысячах) будет найдено по запросу

**фрегат & эсминец**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *фрегат &*  *эсминец* | *500* |
| *фрегат* | *2000* |
| *эсминец* | *2500* |

Сколько страниц (в тысячах) будет найдено по запросу

**фрегат | эсминец**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *фрегат &*  *эсминец* | *500* |
| *фрегат* | *эсминец* | *4500* |
| *эсминец* | *2500* |

Сколько страниц (в тысячах) будет найдено по запросу

**фрегат**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *крейсер* |  *линкор* | *7000* |
| *крейсер* | *4800* |
| *линкор* | *4500* |

Сколько страниц (в тысячах) будет найдено по запросу

**крейсер & линкор**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *торты* |  *пироги* | *12000* |
| *торты* &  *пироги* | *6500* |
| *пироги* | *7700* |

Сколько страниц (в тысячах) будет найдено по запросу

**торты**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *пирожное* |  *выпечка* | *14200* |
| *пирожное* | *9700* |
| *пирожное* &  *выпечка* | *5100* |

Сколько страниц (в тысячах) будет найдено по запросу

**выпечка**

1. В таблице приведены запросы к поисковому серверу. Расположите номера запросов в порядке **убывания** количества страниц, которые найдет поисковый сервер по каждому запросу. Для обозначения логической операции «ИЛИ» в запросе используется символ |, а для логической операции «И» – &.

1 ) **хвост & лапы & (усы | документы)**

2 ) **усы & хвост & лапы & документы**

3 ) **лапы & хвост**

4) **лапы | хвост**

1. В таблице приведены запросы к поисковому серверу. Расположите номера запросов в порядке **убывания** количества страниц, которые найдет поисковый сервер по каждому запросу. Для обозначения логической операции «ИЛИ» в запросе используется символ |, а для логической операции «И» – &.

1 ) **барокко | классицизм**

2 ) **барокко | (классицизм & модерн)**

3 ) **(барокко & ампир) | (классицизм & модерн)**

4) **барокко | ампир | классицизм | модерн**

1. Некоторый сегмент сети Интернет состоит из 5000 сайтов. Поисковый сервер в автоматическом режиме составил таблицу ключевых слов для сайтов этого сегмента. Вот ее фрагмент:

|  |  |
| --- | --- |
| ***Ключевое слово*** | ***Количество сайтов, для которых данное слово является ключевым*** |
| *принтеры* | *400* |
| *сканеры* | *300* |
| *мониторы* | *500* |

Сколько сайтов будет найдено по запросу

**(принтеры | мониторы) & сканеры**

если по запросу**принтеры | сканеры**было найдено 600 сайтов*,* по запросу**принтеры | мониторы** – 900, а по запросу**сканеры | мониторы**– 750.

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *шахматы* |  *теннис* | *7770* |
| *теннис* | *5500* |
| *шахматы* &  *теннис* | *1000* |

Сколько страниц (в тысячах) будет найдено по запросу

**шахматы**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Атос* &  *Портос* | *335* |
| *Атос & Арамис* | *235* |
| Атос &  *Портос & Арамис* | *120* |

Сколько страниц (в тысячах) будет найдено по запросу

**Атос & (Портос | Арамис)**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *васильки* &  *ландыши* | *650* |
| *ландыши & лютики* | *230* |
| ландыши &  *(васильки | лютики)* | *740* |

Сколько страниц (в тысячах) будет найдено по запросу

**ландыши & васильки & лютики**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *декабрь* &  *январь & февраль* | *113* |
| *декабрь & январь* | *225* |
| декабрь &  *(январь | февраль)* | *645* |

Сколько страниц (в тысячах) будет найдено по запросу

**декабрь & февраль**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *март* &  *май & июнь* | *150* |
| *март & май* | *420* |
| март &  *(май | июнь)* | *520* |

Сколько страниц (в тысячах) будет найдено по запросу

**март & июнь**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *март* & *май* | *472* |
| *май & апрель* | *425* |
| май &  *(март | апрель)* | *620* |

Сколько страниц (в тысячах) будет найдено по запросу

**март & апрель & май**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Фрегат | Эсминец* | *3400* |
| *Фрегат & Эсминец* | *900* |
| *Фрегат* | *2100* |

Сколько страниц (в тысячах) будет найдено по запросу

**Эсминец**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Пушкин | Лермонтов* | *5200* |
| *Лермонтов* | *3000* |
| *Пушкин & Лермонтов* | *1200* |

Сколько страниц (в тысячах) будет найдено по запросу

**Пушкин**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Лебедь & (Рак | Щука)* | *320* |
| *Лебедь & Рак* | *200* |
| *Лебедь & Рак & Щука* | *50* |

Сколько страниц (в тысячах) будет найдено по запросу

**Лебедь & Щука**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Пекин & (Москва | Токио)* | *338* |
| *Пекин & Москва* | *204* |
| *Пекин & Токио* | *184* |

Сколько страниц (в тысячах) будет найдено по запросу

**Пекин & Москва & Токио**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Москва & Лондон* | *255* |
| *Москва & Париж* | *222* |
| *Москва & Париж & Лондон* | *50* |

Сколько страниц (в тысячах) будет найдено по запросу

**Москва & (Париж | Лондон)**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Попугай & (Антилопа | Тапир)* | *340* |
| *Попугай & Антилопа* | *220* |
| *Попугай & Тапир* | *190* |

Сколько страниц (в тысячах) будет найдено по запросу

**Попугай & Антилопа & Тапир**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Лошадь & (Пони | Мустанг)* | *350* |
| *Лошадь & Пони* | *235* |
| *Лошадь & Пони & Мустанг* | *65* |

Сколько страниц (в тысячах) будет найдено по запросу

**Лошадь & Мустанг**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Маркиз & Виконт* | *320* |
| *Маркиз & Граф* | *575* |
| *Маркиз & Граф & Виконт* | *55* |

Сколько страниц (в тысячах) будет найдено по запросу

**Маркиз & (Граф | Виконт)**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Суфле* | *450* |
| *Корзина* | *200* |
| *Эклер* | *490* |
| *Суфле & Корзина* | *70* |
| *Суфле & Эклер* | *160* |
| *Корзина & Эклер* | *0* |

Сколько страниц (в тысячах) будет найдено по запросу

**Суфле | Корзина | Эклер**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Стольник* | *375* |
| *Рында* | *315* |
| *Парус* | *290* |
| *Стольник & Рында* | *85* |
| *Стольник & Парус* | *0* |
| *Стольник | Рында | Парус* | *840* |

Сколько страниц (в тысячах) будет найдено по запросу

**Парус & Рында**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Слобода* | *515* |
| *Пилигрим* | *175* |
| *Пилигрим & Равелин* | *105* |
| *Слобода & Равелин* | *70* |
| *Слобода & Пилигрим* | *0* |
| *Слобода | Равелин | Пилигрим* | *765* |

Сколько страниц (в тысячах) будет найдено по запросу

**Равелин**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *(Суворов & Альпы) | (Суворов & Варшава)* | *1100* |
| *Суворов & Варшава* | *600* |
| *Суворов & Варшава & Альпы* | *50* |

Сколько страниц (в тысячах) будет найдено по запросу

**Суворов & Альпы**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *(Испания & Америка) | (Испания & Индия)* | *2800* |
| *Испания & Америка* | *1600* |
| *Испания & Индия & Америка* | *150* |

Сколько страниц (в тысячах) будет найдено по запросу

**Испания & Индия**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Китай & (Америка | Испания & Индия)* | *590* |
| *Китай & Испания & Индия* | *180* |
| *Китай & Америка* | *560* |

Сколько страниц (в тысячах) будет найдено по запросу

**Китай & Америка & Испания & Индия**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *(макаки | павианы & гиббоны) & шимпанзе* | *154* |
| *шимпанзе & павианы & гиббоны* | *120* |
| *шимпанзе & макаки & павианы & гиббоны* | *32* |

Сколько страниц (в тысячах) будет найдено по запросу

**макаки & шимпанзе**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Пушкин & Лермонтов* | *540* |
| *Лермонтов & Толстой & Достоевский* | *280* |
| *Толстой & Лермонтов & Пушкин & Достоевский* | *150* |

Сколько страниц (в тысячах) будет найдено по запросу

**(Пушкин | Толстой & Достоевский) & Лермонтов**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *(галера | бриг & фрегат) & корвет* | *620* |
| *галера & корвет* | *560* |
| *фрегат & галера & корвет & бриг* | *70* |

Сколько страниц (в тысячах) будет найдено по запросу

**корвет & бриг & фрегат**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *(стол | стул & диван) & кровать* | *890* |
| *стол & кровать* | *780* |
| *кровать & стул & диван* | *320* |

Сколько страниц (в тысячах) будет найдено по запросу

**диван & стол & кровать & стул**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *дуб & берёза* | *156* |
| *берёза & роза & ножницы* | *252* |
| *роза & берёза & дуб & ножницы* | *65* |

Сколько страниц (в тысячах) будет найдено по запросу

**(дуб | роза & ножницы) & берёза**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *США |Япония | Китай* | *450* |
| *Япония | Китай* | *260* |
| *США* | *290* |

Сколько страниц (в тысячах) будет найдено по запросу

**(США & Япония) | (США & Китай)**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Ростов & (Орёл & Курск | Белгород)* | *370* |
| *Ростов & Белгород* | *204* |
| *Ростов & Орёл & Курск & Белгород* | *68* |

Сколько страниц (в тысячах) будет найдено по запросу

**Ростов & Орёл & Курск?**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Пилот* | *700* |
| *Пилот | Вертолёт | Акула* | *1200* |
| *Пилот & Вертолёт & Акула* | *0* |
| *Пилот & Акула* | *110* |
| *Пилот & Вертолёт* | *220* |
| *Вертолёт & Акула* | *330* |

Сколько страниц (в тысячах) будет найдено по запросу

**Вертолёт | Акула?**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Пчела & Улей & Город* | *0* |
| *Пчела | Улей | Город* | *1100* |
| *Пчела & Город* | *120* |
| *Пчела & Улей* | *210* |
| *Улей & Город* | *290* |
| *Пчела* | *700* |

Сколько страниц (в тысячах) будет найдено по запросу

**Улей | Город?**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Англия & (Уэльс & Шотландия | Ирландия)* | *450* |
| *Англия & Ирландия* | *304* |
| *Англия & Уэльс & Шотландия & Ирландия* | *87* |

Сколько страниц (в тысячах) будет найдено по запросу

**Англия & Уэльс & Шотландия?**

1. В таблице приведены запросы и количество страниц, которые нашел поисковый сервер по этим запросам в некотором сегменте Интернета:

|  |  |
| --- | --- |
| ***Запрос*** | ***Количество страниц (тыс.)*** |
| *Новосибирск & (Красноярск & Хабаровск | Норильск)* | *570* |
| *Новосибирск & Норильск* | *214* |
| *Новосибирск & Красноярск & Хабаровск?* | *424* |

Сколько страниц (в тысячах) будет найдено по запросу

**Новосибирск & Красноярск & Хабаровск & Норильск?**

# 18 (повышенный уровень, время – 3 мин)

**Тема**: Основные понятия математической логики.

**Про обозначения**

К сожалению, обозначения логических операций И, ИЛИ и НЕ, принятые в «серьезной» математической логике (**∧**,**∨**,**¬**), неудобны, интуитивно непонятны и никак не проявляют аналогии с обычной алгеброй. Автор, к своему стыду, до сих пор иногда путает **∧** и **∨**. Поэтому на его уроках операция «НЕ» обозначается чертой сверху, «И» – знаком умножения (поскольку это все же логическое умножение), а «ИЛИ» – знаком «+» (логическое сложение).   
В разных учебниках используют разные обозначения. К счастью, в начале задания ЕГЭ приводится расшифровка закорючек (**∧**, **∨**,**¬**), что еще раз подчеркивает проблему. Далее во всех решениях приводятся два варианта записи.

**Что нужно знать**:

* условные обозначения логических операций

**¬ A, **  не A (отрицание, инверсия)

**A ∧ B, ** A и B (логическое умножение, конъюнкция)

**A ∨ B, **  A или B (логическое сложение, дизъюнкция)

**A** → **B**  импликация (следование)

* таблицы истинности логических операций «И», «ИЛИ», «НЕ», «импликация» (см. презентацию «Логика»)
* операцию «импликация» можно выразить через «ИЛИ» и «НЕ»:

**A** → **B = ¬ A ∨ B** или в других обозначениях **A** → **B = **

* если в выражении нет скобок, сначала выполняются все операции «НЕ», затем – «И», затем – «ИЛИ», и самая последняя – «импликация»
* иногда полезны формулы де Моргана[[16]](#footnote-16):

**¬ (A ∧ B) = ¬ A ∨ ¬ B **

**¬ (A ∨ B) = ¬ A ∧ ¬ B **

### Пример задания:

**Р-14.** Элементами множества А являются натуральные числа. Известно, что выражение

(*x* {2, 4, 6, 8, 10, 12}) → (((*x* {4, 8, 12, 116}) ∧ ¬(*x* A)) → ¬(*x* {2, 4, 6, 8, 10, 12}))

истинно (т. е. принимает значение 1) при любом значении переменной х.

Определите наименьшее возможное значение суммы элементов множества A.

**Решение:**

1. Заметим, что в задаче, кроме множества *A*, используются еще два множества:

P = {2, 4, 6, 8, 10, 12} Q = {4, 8, 12, 116}

1. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q*

1. перейдем к более простым обозначениям



1. раскрываем обе импликации по формуле :



1. теперь используем закон де Моргана :



1. поскольку это выражение должно быть равно 1, то *A* должно быть истинным везде, где ложно 
2. тогда минимальное допустимое множество *A* – это  (по закону де Моргана)
3. переходим ко множествам

= {4, 8, 12, 116}

= {2, 4, 6, 8, 10, 12}

1. тогда – это все натуральные числа, которые входят одновременно в  и ; они выделены жёлтым цветом: {4, 8, 12}
2. именно эти числа и должны быть «перекрыть» множеством *А*min, поэтому минимальный состав множества A – это *А*min = {4, 8, 12}, сумма этих чисел равна 24
3. Ответ: 24.

### Пример задания:

**Р-13.** *На числовой прямой даны два отрезка: P = [37; 60] и Q = [40; 77]. Укажите наименьшую возможную длину такого отрезка A, что формула*



*тождественно истинна, то есть принимает значение 1 при любом значении переменной х.*

**Решение:**

1. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q*

1. перейдем к более простым обозначениям



1. раскрываем обе импликации по формуле :



1. теперь используем закон де Моргана :



1. в таком виде выражение уже смотрится совсем не страшно; Сразу видно, что отрезок  должен перекрыть область на числовой оси, которая не входит в область :

40

*x*



77



60

37



1. по рисунку видно, что не перекрыт только отрезок [40;60] (он выделен жёлтым цветом), его длина – 20, это и есть правильный ответ.
2. Ответ: 20.

### Ещё пример задания:

**Р-12.** *На числовой прямой даны два отрезка: P = [10,39] и Q = [23, 58]. Выберите из предложенных вариантов такой отрезок A, что логическое выражение*

((*x* ∈ *P*) **∧** (*x* ∈ *A*) ) → ((*x* ∈ *Q*) **∧** (*x* ∈ *A*) )

*тождественно истинна, то есть принимает значение 1 при любом значении переменной х.*

1) [5, 20] 2) [15, 35] 3) [25, 45] 4) [5, 65]

**Решение:**

1. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q*

1. перейдем к более простым обозначениям

**P ⋅A** → **Q** **⋅A**

1. раскроем импликацию через операции НЕ и ИЛИ ():



1. раскроем инверсию первого слагаемого по закону де Моргана ():



1. теперь применим закон поглощения



к последним двум слагаемым:



1. для того, чтобы выражение было истинно при всех *x*, нужно, чтобы  было истинно там, где ложно , то есть там, где истинно  (жёлтая область на рисунке)

10

23

*x*



58



39

1. таким образом, *A* должно быть ложно на отрезке [10,23], такое отрезок в предложенном наборе один – это отрезок [25, 45]
2. Ответ: 3.

### Ещё пример задания:

**Р-11.** *На числовой прямой даны два отрезка: P = [10,30] и Q = [25, 55]. Определите наибольшую возможную длину отрезка A, при котором формула*

( *x* ∈ *A*) → ((*x* ∈ *P*) **∨** (*x* ∈ *Q*) )

*тождественно истинна, то есть принимает значение 1 при любом значении переменной х.*

1) 10 2) 20 3) 30 4) 45

**Решение:**

1. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q*

1. перейдем к более простым обозначениям

**A** → **(P** + **Q)**

1. раскроем импликацию через операции НЕ и ИЛИ ():



1. для того, чтобы выражение было истинно при всех *x*, нужно, чтобы  было истинно там, где ложно  (жёлтая область на рисунке)

10

25

*x*



55



30

1. поэтому максимальный отрезок, где *A* может быть истинно (и, соответственно,  ложно) – это отрезок [10,55], имеющий длину 45
2. Ответ: 4.

### Ещё пример задания:

**Р-10.** *На числовой прямой даны два отрезка: P = [10,20] и Q = [25, 55]. Определите наибольшую возможную длину отрезка A, при котором формула*

( *x* ∈ *A*) → ((*x* ∈ *P*) **∨** (*x* ∈ *Q*) )

*тождественно истинна, то есть принимает значение 1 при любом значении переменной х.*

1) 10 2) 20 3) 30 4) 45

**Решение:**

1. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q*

1. перейдем к более простым обозначениям

**A** → **(P** + **Q)**

1. раскроем импликацию через операции НЕ и ИЛИ ():



1. для того, чтобы выражение было истинно при всех *x*, нужно, чтобы  было истинно там, где ложно  (жёлтая область на рисунке)

10

20

*x*



55



25

1. поскольку области истинности  и  разделены, максимальный отрезок, где *A* может быть истинно (и, соответственно,  ложно) – это наибольший из отрезков  и , то есть отрезок [25,55], имеющий длину 30
2. Ответ: 3.

### Ещё пример задания:

**Р-09.** *На числовой прямой даны два отрезка: P = [14,34] и Q = [24, 44]. Выберите такой отрезок A, что формула*

( *x* ∈ *A*) → ((*x* ∈ *P*) ≡ (*x* ∈ *Q*) )

*тождественно истинна, то есть принимает значение 1 при любом значении переменной х. Если таких отрезков несколько, укажите тот, который имеет большую длину.*

1) [15, 29] 2) [25, 29] 3) [35,39] 4) [49,55]

**Решение:**

1. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q*

1. перейдем к более простым обозначениям

**A** → **(P** ≡ **Q)**

1. выражение **R** = (**P** ≡ **Q)** истинно для всех значений *x*, при которых **P** и **Q** равны (либо оба ложны, либо оба истинны)
2. нарисуем область истинности выражения **R** = (**P** ≡ **Q)** на числовой оси (жёлтые области)

14

24

*x*



44



34

1. импликация **A** → **R** истинна за исключением случая, когда **A=1** и **R=0**, поэтому на полуотрезках [14,24[ и ]34,44], где **R=0**, выражение **A** должно быть обязательно ложно; никаких других ограничений не накладывается
2. из предложенных ответов этому условия соответствуют отрезки [25,29] и [49,55]; по условию из них нужно выбрать самый длинный
3. отрезок [25,29] имеет длину 4, а отрезок [49,55] – длину 6, поэтому выбираем отрезок [49, 55]
4. Ответ: 4.

### Ещё пример задания:

**Р-08.** *На числовой прямой даны два отрезка: P = [20, 50] и Q = [10, 60]. Выберите такой отрезок A, что формула*

( (*x* ∈ *P*) → (*x* ∈ *А*) ) /\ ( (*x* ∈ *A*) → (*x* ∈ *Q*) )

*тождественно истинна, то есть принимает значение 1 при любом значении переменной х. Если таких отрезков несколько, укажите тот, который имеет большую длину.*

1) [5, 40] 2) [15, 54] 3) [30,58] 4) [5, 70]

**Решение:**

1. в этом выражении две импликации связаны с помощью операции И (конъюнкции), поэтому для истинности всего выражения обе импликации должны быть истинными
2. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q*

1. перейдем к более простым обозначениям в обоих условиях

**(P** → **A)**  /\ **(A** → **Q)**

и выразим импликацию через операции ИЛИ и НЕ:

,



1. выражение  должно быть истинно на всей числовой оси; обозначим область, которую перекрывает выражение – это две полуоси

20

50

*x*



1. отсюда следует, что отрезок A должен полностью перекрывать отрезок P; этому условию удовлетворяют варианты ответов 2 и 4
2. выражение  тоже должно быть истинно на всей числовой оси; выражение  должно перекрывать все, кроме отрезка, который перекрывает выражение:

10

60

*x*



1. поэтому начало отрезка должно быть внутри отрезка [10,20], а его конец – внутри отрезка [50,60]
2. этим условиям удовлетворяет только вариант 2.
3. Ответ: 2.

### Ещё пример задания:

**Р-07.** *На числовой прямой даны два отрезка: Р = [35, 55] и Q = [45, 65]. Выберите такой отрезок А, что обе приведённые ниже формулы истинны при любом значении переменной х:*

(*x* ∈ *P*) → (*x* ∈ *A*)

(¬ (*x* ∈ *А*)) → (¬(*x* ∈ *Q*))

*Если таких отрезков несколько, укажите тот, который имеет большую длину.*

1) [40,50] 2) [30,60] 3) [30,70] 4) [40, 100]

**Решение:**

1. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q*

1. перейдем к более простым обозначениям в первом условии **P** → **A** и выразим импликацию через операции ИЛИ и НЕ: 
2. выражение  должно быть истинно на всей числовой оси; обозначим область, которую перекрывает выражение - это две полуоси

35

55

*x*



1. отсюда следует, что отрезок A должен полностью перекрывать отрезок P; этому условию удовлетворяют варианты ответов 2 и 3
2. аналогично разбираем и преобразуем второе выражение



1. и находим, что для того, чтобы обеспечить истинность второго выражения на всей оси отрезок A должен полностью перекрыть отрезок Q; этому условию удовлетворяют варианты ответов 3 и 4
2. объединяя результаты п. 5 и 7, получаем, что условию задачи соответствует только отрезок 3.
3. Ответ: 3.

### Ещё пример задания:

**Р-06.** *На числовой прямой даны два отрезка: P =* [2, 10] *и Q =* [6, 14]*. Выберите такой отрезок A, что формула*

( (*x* ∈ *А*) → (*x* ∈ *P*) ) \/ (*x* ∈ *Q*)

*тождественно истинна, то есть принимает значение 1 при любом значении переменной х.*

1) [0, 3] 2) [3, 11] 3) [11, 15] 4)[15, 17]

**Решение:**

1. два условия связаны с помощью операции \/ («ИЛИ»), поэтому должно выполняться хотя бы одно из них
2. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q*

1. тогда получаем, переходя к более простым обозначениям:

*Z =* (**A***→***P***) +* **Q**

1. представим импликацию **A** *→* **P** через операции «ИЛИ» и «НЕ»: , так что получаем
2. это значит, что для тождественной истинности выражения *Z* нужно, чтобы для любого *x* было выполнено одно из условий: , **P***,* **Q**; из всех этих выражений нам **неизвестно только** 
3. посмотрим, какие интервалы перекрываются условиями**P** и**Q**:

2

6

10

14

*x*



1. видим, что отрезок [2,14] перекрыт, поэтому выражение  должно перекрывать оставшуюся часть; таким образом, должно быть истинно на интервалах (– ∞,2) и (14,∞) и, соответственно, выражение **A** (без инверсии) может быть истинно только внутри отрезка [2,14]
2. из всех отрезков, приведенных в условии, только отрезов [3,11] (вариант 2) находится целиком внутри отрезка [2,14], это и есть правильный ответ
3. Ответ: 2.

**Решение (вариант 2, А.Н. Евтеев):**

1. пп. 1-4 такие же, как и в предыдущем способе решения
2. полученное после преобразований выражение  должно быть истинно при любом *x*
3. логическая сумма истинна во всех случаях кроме одного: если все слагаемые ложны, следовательно выражение  ложно только когда **A** = 1, **P** = 0 и **Q** = 0
4. поэтому если область истинности **A** выйдет за пределы отрезка [2,14], где одновременно ложны **P** и **Q**, то  будет ложно
5. это значит, что **A** может быть истинно только внутри отрезка [2,14]
6. из всех отрезков, приведенных в условии, только отрезов [3,11] (вариант 2) находится целиком внутри отрезка [2,14], это и есть правильный ответ
7. Ответ: 2.

**Решение (таблицы истинности, Е.А. Смирнов):**

1. пп. 1-4 такие же, как и в предыдущем способе решения
2. если рассматривать все значения ***x*** на числовой прямой, то логические значения формул могут измениться только при переходе через граничные точки заданных промежутков
3. эти точки (2,6,10 и 14) разбивают числовую прямую на несколько интервалов, для каждого из которых можно определить логическое значение выражения 

|  |  |  |  |
| --- | --- | --- | --- |
| x | *P* | *Q* |  |
| x < 2 | 0 | 0 | 0 |
| 2 < x < 6 | 1 | 0 | 1 |
| 6 < x < 10 | 1 | 1 | 1 |
| 10 < x < 14 | 0 | 1 | 1 |
| x > 14 | 0 | 0 | 0 |

для упрощения записи не будем рассматривать значения формул на концах отрезков, так как это не влияет на решение

1. по условию выражение  должно быть равно 1 при любых значениях ***x***, то есть, в соответствующем столбце таблицы должны быть все единицы; отсюда можно найти, каким должно быть значение  (и соответствующее значение ) для каждого интервала:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| x | *P* | *Q* |  |  |  |  |
| x < 2 | 0 | 0 | 0 | 1 | 0 | 1 |
| 2 < x < 6 | 1 | 0 | 1 | любое | любое | 1 |
| 6 < x < 10 | 1 | 1 | 1 | любое | любое | 1 |
| 10 < x < 14 | 0 | 1 | 1 | любое | любое | 1 |
| x > 14 | 0 | 0 | 0 | 1 | 0 | 1 |

1. таким образом, значениедолжно быть равно 0 вне отрезка [2,14]; из всех отрезков, приведенных в условии, только отрезов [3,11] (вариант 2)
2. Ответ: 2.

### Ещё пример задания:

**Р-05.** *На числовой прямой даны два отрезка: P =* [2, 20] *и Q =* [15, 25]*. Выберите такой отрезок A, что формула*

( (*x* ∉ *А*) → (*x* ∉ *P*) ) \/ (*x* ∈ *Q*)

*тождественно истинна, то есть принимает значение 1 при любом значении переменной х.*

1) [0, 15] 2) [10, 25] 3) [2, 10] 4)[15, 20]

**Решение (отрезки на оси):**

1. два условия связаны с помощью операции \/ («ИЛИ»), поэтому должно выполняться хотя бы одно из них
2. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q*

1. учтем, что в формуле используется знак ∉ («не принадлежит»), поэтому при переходе к более простым обозначениям получаем:



1. представим импликацию  через операции «ИЛИ» и «НЕ»: , так что получаем
2. это значит, что для тождественной истинности выражения *Z* нужно, чтобы для любого *x* было выполнено одно из условий: ,*,* **Q**; из всех этих выражений нам **неизвестно только** 
3. посмотрим, какие интервалы перекрываются условиямии**Q**; область  состоит из двух участков числовой оси, которые не входят в отрезок [2,20], а область **Q** – это отрезок [15,25]:

2

15

20

25

*x*



1. таким образом, область истинности выражения  должна перекрывать оставшуюся часть – отрезок [2,15]
2. из всех отрезков, приведенных в условии, только отрезок [0,15] (вариант 1) полностью перекрывает отрезок [2,15], это и есть правильный ответ
3. Ответ: 1.

**Решение (таблицы истинности, Е.А. Смирнов):**

1. пп. 1-4 такие же, как и в предыдущем способе решения
2. если рассматривать все значения ***x*** на числовой прямой, то логические значения формул могут измениться только при переходе через граничные точки заданных промежутков
3. эти точки (2,15,20 и 25) разбивают числовую прямую на несколько интервалов, для каждого из которых можно определить логическое значение выражения 

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | *P* |  | *Q* |  |
| x < 2 | 0 | 1 | 0 | 1 |
| 2 < x < 15 | 1 | 0 | 0 | 0 |
| 15 < x < 20 | 1 | 0 | 1 | 1 |
| 20 < x < 25 | 0 | 1 | 1 | 1 |
| x > 25 | 0 | 1 | 0 | 1 |

для упрощения записи не будем рассматривать значения формул на концах отрезков, так как это не влияет на решение

1. по условию выражение  должно быть равно 1 при любых значениях ***x***, то есть, в соответствующем столбце таблицы должны быть все единицы; отсюда можно найти, каким должно быть значение для каждого интервала:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| x | *P* |  | *Q* |  |  |  |
| x < 2 | 0 | 1 | 0 | 1 | любое | 1 |
| 2 < x < 15 | 1 | 0 | 0 | 0 | 1 | 1 |
| 15 < x < 20 | 1 | 0 | 1 | 1 | любое | 1 |
| 20 < x < 25 | 0 | 1 | 1 | 1 | любое | 1 |
| x > 25 | 0 | 1 | 0 | 1 | любое | 1 |

1. таким образом, область истинности выражения  должна перекрывать отрезок [2,15]
2. из всех отрезков, приведенных в условии, только отрезок [0,15] (вариант 1) полностью перекрывает отрезок [2,15], это и есть правильный ответ
3. Ответ: 1.

### Ещё пример задания:

**Р-04.** *На числовой прямой даны три отрезка: P =* [10, 25]*, Q =* [15, 30] *и R=*[25,40]*. Выберите такой отрезок A, что формула*

( (*x* ∈ *Q*) → (*x* ∉ *R*) ) /\ (*x* ∈ *A*) /\ (*x* ∉ *P*)

*тождественно* ***ложна****, то есть принимает значение 0 при любом значении переменной х.*

1) [0, 15] 2) [10, 40] 3) [25, 35] 4)[15, 25]

**Решение (способ 1):**

1. три условия связаны с помощью операции /\ (логическое «И»), поэтому для того, чтобы выражение было тождественно равно нулю, для каждого значения ***x*** по крайней мере одно из них должно был ложно
2. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q,* **R***: x ∈ R*

1. учтем, что в формуле дважды используется знак ∉ («не принадлежит»), поэтому при переходе к более простым обозначениям получаем:



1. представим импликацию  через операции «ИЛИ» и «НЕ»: , так что получаем
2. роль сомножителя **A** состоит в том, чтобы обнулить выражение везде, где произведение  равно 1; поэтому для этих значений ***x*** выражение **A** должно быть равно нулю, а для остальных ***x*** его значение не играет роли
3. область истинности выражения  по закону де Моргана совпадает с областью истинности выражения , то есть это область вне общей части отрезков Q и R (она показана жёлтым цветом на рисунке):

10

15

30

40

*x*

25



1. теперь умножим это выражение на  (ему соответствует область вне отрезка [10,25]), построив область ; эта область, где одновременно истинны  и , выделена фиолетовым цветом:

10

15

30

40

*x*

25



1. как следует из п. 4, в фиолетовой области на предыдущем рисунке выражение **A** должно быть обязательно равно 0, и только внутри отрезка [10,30] может быть истинно
2. таким образом, среди ответов нужно найти отрезок, который целиком помещается внутри отрезка [10,30]
3. этому условию удовлетворяет только отрезок [15,25] (ответ 4)
4. Ответ: 4.

**Решение (способ 2, инверсия и преобразование):**

1. пп. 1-4 такие же, как и в первом способе
2. выражение  тождественно ложно тогда и только тогда, когда обратное ему, , тождественно истинно; таким образом, если выполнить инверсию для , мы сведём задачу к задаче из демо-варианта ЕГЭ-2013, разобранной выше
3. имеем, используя законы де Моргана:



1. выражение  истинно на общей части (пересечении) отрезков Q и R, то есть, на отрезке [25,30]
2. добавляя к этому диапазону отрезок P, получим отрезок [10,30], где истинно выражение 

10

15

30

40

*x*

25



1. остальную часть числовой оси (при ***x*** меньше 10 и ***x*** больше 30) должно перекрыть выражение , то есть  должно быть ложно вне отрезка [10,30]
2. таким образом, среди ответов нужно найти отрезок, который целиком помещается внутри отрезка [10,30]
3. этому условию удовлетворяет только отрезок [15,25] (ответ 4)
4. Ответ: 4.

**Решение (таблицы истинности, Е.А. Смирнов):**

1. пп. 1-5 такие же, как и в первом способе решения
2. если рассматривать все значения ***x*** на числовой прямой, то логические значения формул могут измениться только при переходе через граничные точки заданных промежутков
3. эти точки (10,15,25, 30 и 40) разбивают числовую прямую на несколько интервалов, для каждого из которых можно определить логическое значение выражения 

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| x | *P* |  | *Q* |  | *R* |  |  |  |
| x < 10 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| 10 < x < 15 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 15 < x < 25 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 25 < x < 30 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| 30 < x < 40 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| x > 40 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |

для упрощения записи не будем рассматривать значения формул на концах отрезков, так как это не влияет на решение

1. по условию выражение  должно быть равно 0 при любых значениях ***x***, то есть, в соответствующем столбце таблицы должны быть все единицы; отсюда можно найти, каким должно быть значение для каждого интервала:

|  |  |  |  |
| --- | --- | --- | --- |
| x |  |  |  |
| x < 10 | 1 | 0 | 0 |
| 10 < x < 15 | 0 | любое | 0 |
| 15 < x < 25 | 0 | любое | 0 |
| 25 < x < 30 | 0 | любое | 0 |
| 30 < x < 40 | 1 | 0 | 0 |
| x > 40 | 1 | 0 | 0 |

1. таким образом, среди ответов нужно найти отрезок, который целиком помещается внутри отрезка [10,30]
2. этому условию удовлетворяет только отрезок [15,25] (ответ 4)
3. Ответ: 4.

### Ещё пример задания:

**Р-03.** *На числовой прямой даны три интервала: P =* (5, 10)*, Q =* [10, 20] *и R =* [25,40]*. Выберите такой отрезок A, что выражения*

(*x* ∈ *A*) → (*x* ∈ *P*) *и* (*x* ∈ *Q*) → (*x* ∈ *R*)

*тождественно* ***равны****, то есть принимают одинаковые значения при любом значении переменной* ***х****.*

1) [7, 20] 2) [2, 12] 3) [10,25] 4)[20, 30]

**Решение (способ 1, отрезки на числовой прямой):**

1. обратите внимание, что интервал P – это открытый интервал; это необходимо для того, чтобы можно было выполнить заданное условие в точках стыковки отрезков
2. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q,* **R***: x ∈ R*

1. перейдём к более простым обозначениям:

, 

1. выразим импликации через операции «ИЛИ» и «НЕ»:

, 

1. заметим, что неизвестная величина **A** входит только в выражение 
2. общая идея состоит в том, чтобы построить на числовой оси область истинности для полностью известного выражения , а затем дополнить отрезок P до этой области; это «дополнение» будет соответствовать области 
3. построим область  – объединение отрезка R и области вне отрезка Q:

10

15

40

*x*

25



5

20



обратим внимание, что область  (выделена жёлтым цветом) в данном случае совпадает с 

1. теперь рассмотрим область  (выделена голубым цветом)

10

15

40

*x*

25



5

20



1. чтобы область истинности выражения  совпала с жёлтой областью, выражение  должно «перекрыть» всю фиолетовую область (возможно, заходя в область )
2. поэтому выражение обязательно должно быть истинно на отрезке [10,20]; обязательно должно быть ложно на полуосях  и , а на отрезке [5,10] его значение может быть любым (там выполнение требований обеспечивает область )
3. из предложенных вариантов ответов этим требованиям удовлетворяет только отрезок [7,20] (ответ 1)
4. Ответ: 1.

**Решение (способ 2, таблицы истинности, Е.А. Смирнов):**

1. пп. 1-6 такие же, как и в первом способе решения
2. если рассматривать все значения ***x*** на числовой прямой, то логические значения формул могут измениться только при переходе через граничные точки заданных промежутков
3. эти точки (5, 10, 20, 25 и 40) разбивают числовую прямую на несколько интервалов, для каждого из которых можно определить логическое значение выражения 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x | *P* | *Q* |  | *R* |  |
| x < 5 | 0 | 0 | 1 | 0 | 1 |
| 5 < x < 10 | 1 | 0 | 1 | 0 | 1 |
| 10 < x < 20 | 0 | 1 | 0 | 0 | 0 |
| 20 < x < 25 | 0 | 0 | 1 | 0 | 1 |
| 25 < x < 40 | 0 | 0 | 1 | 1 | 1 |
| x > 40 | 0 | 0 | 1 | 0 | 1 |

для упрощения записи не будем рассматривать значения формул на концах отрезков, так как это не влияет на решение

1. по условию выражение  должно быть равно выражению  при любых значениях ***x***, отсюда можно найти, каким должно быть значение  (и соответствующее значение ) для каждого интервала:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x |  |  | *P* |  |  |
| x < 5 | 1 | 1 | 0 | 1 | 0 |
| 5 < x < 10 | 1 | 1 | 1 | любое | любое |
| 10 < x < 20 | 0 | 0 | 0 | 0 | 1 |
| 20 < x < 25 | 1 | 1 | 0 | 1 | 0 |
| 25 < x < 40 | 1 | 1 | 0 | 1 | 0 |
| x > 40 | 1 | 1 | 0 | 1 | 0 |

1. таким образом, среди ответов нужно найти отрезок, который перекрывает отрезок [10,20] и, возможно, заходит внутрь отрезка [5,10]
2. из предложенных вариантов ответов этим требованиям удовлетворяет только отрезок [7,20] (ответ 1)
3. Ответ: 1.

### Ещё пример задания:

**Р-02.** *На числовой прямой даны три интервала: P =* (10, 15)*, Q =* [5, 20] *и R =* [15,25]*. Выберите такой отрезок A, что выражения*

(*x* ∉ *A*) → (*x* ∈ *P*) *и* (*x* ∈ *Q*) → (*x* ∈ *R*)

*принимают* ***различные*** *значения при любых* ***x****.*

1) [7, 20] 2) [2, 15] 3) [5,12] 4)[20, 25]

**Решение (способ 1, отрезки на числовой прямой):**

1. обратите внимание, что интервал P – это открытый интервал; это необходимо для того, чтобы можно было выполнить заданное условие в точках стыковки отрезков
2. для того, чтобы упростить понимание выражения, обозначим отдельные высказывания буквами

**A***: x ∈ А,* **P***: x ∈ P,* **Q***: x ∈ Q,* **R***: x ∈ R*

1. перейдём к более простым обозначениям:

, 

1. выразим импликации через операции «ИЛИ» и «НЕ»:

, 

1. заметим, что неизвестная величина **A** входит только в выражение 
2. общая идея состоит в том, чтобы построить на числовой оси область истинности для полностью известного выражения , а затем дополнить отрезок P до «обратной» области, в которой выражение  ложно; это «дополнение» будет соответствовать области 
3. построим область  – объединение отрезка R и области вне отрезка Q:

10

15

*x*

25



5

20



1. теперь рассмотрим область  (выделена голубым цветом)

10

15

*x*

25



5

20



1. чтобы выполнить заданное условие (противоположность значений  и  при любых ***x***), область истинности выражения  должна совпадать с областью, где выражение  ложно; для этого выражение  должно «перекрыть» всю фиолетовую область (возможно, заходя в область ), но не должно заходить в «жёлтую» область:

10

15

*x*

25



5

20



1. из предложенных вариантов ответов этим требованиям удовлетворяет только отрезок [5,12] (ответ 3)
2. Ответ: 3.

**Решение (способ 2, таблицы истинности, Е.А. Смирнов):**

1. пп. 1-6 такие же, как и в первом способе решения
2. если рассматривать все значения ***x*** на числовой прямой, то логические значения формул могут измениться только при переходе через граничные точки заданных промежутков
3. эти точки (5, 10, 15, 20 и 25) разбивают числовую прямую на несколько интервалов, для каждого из которых можно определить логическое значение выражения 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x | *P* | *Q* |  | *R* |  |
| x < 5 | 0 | 0 | 1 | 0 |  |
| 5 < x < 10 | 0 | 1 | 0 | 0 |  |
| 10 < x < 15 | 1 | 1 | 0 | 0 |  |
| 15 < x < 20 | 0 | 1 | 0 | 1 |  |
| 20 < x < 25 | 0 | 0 | 1 | 1 |  |
| x > 25 | 0 | 0 | 1 | 0 |  |

для упрощения записи не будем рассматривать значения формул на концах отрезков, так как это не влияет на решение

1. по условию выражение  должно быть НЕ равно выражению  при любых значениях ***x***, отсюда можно найти, каким должно быть значение  для каждого интервала:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x |  |  | *P* |  |
| x < 5 | 1 | 0 | 0 | 0 |
| 5 < x < 10 | 0 | 1 | 0 | 1 |
| 10 < x < 15 | 0 | 1 | 1 | любое |
| 15 < x < 20 | 1 | 0 | 0 | 0 |
| 20 < x < 25 | 1 | 0 | 0 | 0 |
| x > 25 | 1 | 0 | 0 | 0 |

1. таким образом, среди ответов нужно найти отрезок, который перекрывает отрезок [5,10] и, возможно, заходит внутрь отрезка [10,15]
2. из предложенных вариантов ответов этим требованиям удовлетворяет только отрезок [5,12] (ответ 3)
3. Ответ: 3.

### Ещё пример задания:

**Р-01.** *Какое из приведённых имен удовлетворяет логическому условию:*(первая буква согласная → вторая буква согласная) /\ (предпоследняя буква гласная → последняя буква гласная)*?*

1) КРИСТИНА 2) МАКСИМ 3) СТЕПАН 4) МАРИЯ

**Решение:**

1. два условия связаны с помощью операции /\ («И»), поэтому должны выполняться одновременно
2. импликация ложна, если ее первая часть («посылка») истинна, а вторая («следствие») – ложна
3. первое условие «*первая буква согласная → вторая буква согласная*» ложно тогда, когда первая буква согласная, а вторая – гласная, то есть для ответов 2 и 4
4. второе условие «*предпоследняя буква гласная → последняя буква гласная*» ложно тогда, когда предпоследняя буква гласная, а последняя – согласная, то есть, для ответа 3
5. таким образом, для варианта 1 (КРИСТИНА) оба промежуточных условия и исходное условие в целом истинны
6. ответ: 1.

### Ещё пример задания:

**Р-00.** *Для какого из указанных значений X истинно высказывание* **¬((X > 2)→(X > 3))**?

1) 1 2) 2 3) 3 4) 4

**Решение (вариант 1, прямая подстановка):**

1. определим порядок действий: сначала вычисляются результаты отношений в скобках, затем выполняется импликация (поскольку есть «большие» скобки), затем – отрицание (операция «НЕ») для выражения в больших скобках
2. выполняем операции для всех приведенных возможных ответов (1 обозначает истинное условие, 0 – ложное); сначала определяем результаты сравнения в двух внутренних скобках:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **X** | **X > 2** | **X > 3** | **(X > 2)→(X > 3)** | **¬((X > 2)→(X > 3))** |
| 1 | 0 | 0 |  |  |
| 2 | 0 | 0 |  |  |
| 3 | 1 | 0 |  |  |
| 4 | 1 | 1 |  |  |

1. по таблице истинности операции «импликация» находим третий столбец (значение выражения в больших скобках), применив операцию «импликация» к значениям второго и третьего столбцов (в каждой строке):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **X** | **X > 2** | **X > 3** | **(X > 2)→(X > 3)** | **¬((X > 2)→(X > 3))** |
| 1 | 0 | 0 | 1 |  |
| 2 | 0 | 0 | 1 |  |
| 3 | 1 | 0 | 0 |  |
| 4 | 1 | 1 | 1 |  |

1. значение выражения равно инверсии третьего столбца (меняем 1 на 0 и наоборот):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **X** | **X > 2** | **X > 3** | **(X > 2)→(X > 3)** | **¬((X > 2)→(X > 3))** |
| 1 | 0 | 0 | 1 | 0 |
| 2 | 0 | 0 | 1 | 0 |
| 3 | 1 | 0 | 0 | 1 |
| 4 | 1 | 1 | 1 | 0 |

1. таким образом, ответ – 3.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + можно «забыть» отрицание (помните, что правильный ответ – всего один!)   + можно перепутать порядок операций (скобки, «НЕ», «И», «ИЛИ», «импликация»)   + нужно помнить таблицу истинности операции «импликация», которую очень любят составители тестов[[17]](#footnote-17)   + этот метод проверяет только заданные числа и не дает общего решения, то есть не определяет все множество значений X, при которых выражение истинно |

**Решение (вариант 2, упрощение выражения):**

1. обозначим простые высказывания буквами:

**A = X > 2, B = X > 3**

1. тогда можно записать все выражение в виде

**¬(A → B)** или 

1. выразим импликацию через «ИЛИ» и «НЕ» (см. выше):

**¬(A → B)= ¬(¬A ∨ B)** или 

1. раскрывая по формуле де Моргана операцию «НЕ» для всего выражения, получаем

**¬(¬A ∨ B)= A ∧ ¬B** или 

1. таким образом, данное выражение истинно только тогда, когда A истинно (**X > 2**), а B – ложно (**X ≤ 3**), то есть для всех X, таких что **2 < X ≤ 3**
2. из приведенных чисел только 3 удовлетворяет этому условию,
3. таким образом, ответ – 3.

|  |
| --- |
| **Возможные проблемы**:   * + нужно помнить законы логики (например, формулы де Моргана)   + при использовании формул де Моргана нужно не забыть заменить «И» на «ИЛИ» и наоборот   + нужно не забыть, что инверсией (отрицанием) для выражения **X > 3** является **X ≤ 3**, а не **X < 3** |

**Решение (вариант 3, использование свойств импликации):**

1. обозначим простые высказывания буквами:

**A = X > 2, B = X > 3**

1. тогда исходное выражение можно переписать в виде **¬(A→B)=1** или **A→B=0**
2. импликация **A→B** ложна в одном единственном случае, когда **A = 1** и **B = 0**; поэтому заданное выражение истинно для всех X, таких что **X > 2** и **X ≤ 3**
3. из приведенных чисел только 3 удовлетворяет этому условию,
4. таким образом, ответ – 3.

|  |
| --- |
| **Выводы:**   1. в данном случае, наверное, проще третий вариант решения, однако он основан на том, что импликация ложна только для одной комбинации исходных данных; не всегда этот прием применим 2. второй и третий варианты позволяют не только проверить заданные значения, но и получить *общее* решение – все множество X, для которых выражение истинно; это более красиво для человека, обладающего математическим складом ума. |

# 19 (повышенный уровень, время – 5 мин)

**Тема**: Работа с массивами и матрицами в языке программирования[[18]](#footnote-18).

**Что нужно знать**:

* работу цикла **for** (цикла с переменной)
* массив – это набор однотипных элементов, имеющих общее имя и расположенных в памяти рядом
* для обращения к элементу массива используют квадратные скобки, запись **A[i]** обозначает элемент массива **A** с номером (индексом) **i**
* матрица (двухмерный массив) – это прямоугольная таблица однотипных элементов
* если матрица имеет имя A, то обращение **A[i,k]** обозначает элемент, расположенный на пересечении строки **i** и столбца **k**
* элементы, у которых номера строки и столбца совпадают, расположены на главной диагонали[[19]](#footnote-19)

|  |  |  |  |
| --- | --- | --- | --- |
| A[1,1] |  |  |  |
|  | A[2,2] |  |  |
|  |  | A[3,3] |  |
|  |  |  | A[4,4] |

* выше главной диагонали расположены элементы, у которых номер строки **меньше** номера столбца:

|  |  |  |  |
| --- | --- | --- | --- |
|  | A[1,2] | A[1,3] | A[1,4] |
|  |  | A[2,3] | A[2,4] |
|  |  |  | A[3,4] |
|  |  |  |  |

* ниже главной диагонали расположены элементы, у которых номер строки **больше** номера столбца:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| A[2,1] |  |  |  |
| A[3,1] | A[3,2] |  |  |
| A[4,1] | A[4,2] | A[4,3] |  |

### Пример задания:

*В программе описан одномерный целочисленный массив с индексами от 0 до 10. Ниже представлен фрагмент программы, обрабатывающей данный массив:*

**s:=0;**

**n:=10;**

**for i:=0 to n-3 do begin**

**s:=s+A[i]-A[i+2]**

**end;**

*В начале выполнения этого фрагмента в массиве находились трёхзначные натуральные числа. Какое наибольшее значение может иметь переменная s после выполнения данной программы?*

**Решение**:

1. сначала попытаемся понять, что же делает эта программа; возьмем массив из пяти элементов (*n* = 5):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** |
| ***A*[0]** | ***A*[1]** | ***A*[2]** | ***A*[3]** | ***A*[4]** |

1. переменная s будет изменяться следующим образом:

**s := 0**

**s := s + A[0] – A[2]**

**s := s + A[1] – A[3]**

**s := s + A[2] – A[4]**

1. в итоге после всех действий

**s:= A[0]–~~A[2]~~+A[1]–A[3]+~~A[2]~~–A[4] = A[0] + A[1] – A[3] – A[4]**

1. это значит, что значение **s** всегда будет равно сумме двух первых элементов массива минус сумма двух последних элементов
2. все числа – трёхзначные, то есть принадлежат отрезку [100;999]
3. максимальное значение **s** равно 999 + 999 – 100 – 100 = 1798
4. обратите внимание, что это число не зависит от размера массива
5. ответ: 1798.

### Ещё пример задания:

*В программе используется одномерный целочисленный массив A с индексами от 0 до 9. Значения элементов равны 6; 9; 7; 2; 1; 5; 0; 3; 4; 8 соответственно, т.е. A[0] = 6; A[1] = 9 и т.д.*

*Определите значение переменной* **c** *после выполнения следующего фрагмента программы, записанного ниже на разных языках программирования.*

**c := 0;**

**for i := 1 to 9 do**

**if A[i-1] < A[i] then begin**

**c := c + 1;**

**t := A[i];**

**A[i] := A[i-1];**

**A[i-1] := t**

**end;**

**Решение**:

1. сначала попытаемся понять, что же делает эта программа:

* в цикле рассматриваются пары соседних элементов, начиная с пары (**A[0]**,**A[1]**) и заканчивая парой (**A[8]**,**A[9]**);
* если предыдущий элемент **A[i-1]** меньше следующего (**A[i]**), они меняются местами через вспомогательную переменную **t**; таким образом, цикл выполняет один этап (один проход по массиву) метода сортировки массива ***по убыванию***, который называется «методом пузырька»
* начальное значение переменной **c**, которая нас интересует, равно нулю; при каждой перестановке оно увеличивается на 1 (начиная с нуля), то есть, **c** – счётчик перестановок

1. для первой пары выполняется условие **A[0]< A[1]**, поэтому выполняется перестановка:

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

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

1. следующая перестановка будет для пары (**A[1]**, **A[2]**):

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

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

1. следующая – для пары (**A[4]**, **A[5]**):

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

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

1. и далее еще 3 перестановки, в результате которых значение 0 перемещается до конца массива:

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

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

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

1. всего было сделано 6 перестановок, при каждой счётчик увеличивался на 1, поэтому после выполнения этого фрагмента значение переменной **c** будет равно 6
2. ответ: 6.

### Пример задания:

*В программе используется одномерный целочисленный массив A с индексами от 1 до 25. Ниже представлен фрагмент программы, в котором задаются значения элементов:*

**n:= 25;**

**A[1]:= 2;**

**for i:= 2 to n do begin**

**A[i]:= 2\*A[i–1] mod 10;**

**end;**

*Чему будет равно значение A[25] после выполнения фрагмента программы?*

**Решение**:

1. заметим особенность: внутри цикла берется остаток от деления **2\*A[i–1]**на 10, то есть последняя цифра десятичной записи; поэтому все элементы массива – однозначные числа
2. если бы не было этого взятия остатка, каждое последующее число в 2 раза больше предыдущего, цепочка начинается с 2, поэтому в массиве были бы записаны степени числа 2: 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 и т.д.
3. выделим последние цифры в этой цепочке:

2, 4, 8, 6, 2, 4, 8, 6, 2, 4, …

они повторяются через 4 элемента

1. таких полных групп в массиве с 25 элементами будет 25 div 4 = 6; эти 6 групп займут первые 24 элемента, а 25-м будет первый элемент в четвёрке, то есть 2
2. Ответ: 2.

### Ещё пример задания:

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

**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**:

1. 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** и т.д. до **A[9]=0**

1. рассмотрим второй цикл, в котором операторы

**k:=A[i];**

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

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

меняют местами элементы **A[i]** и **A[9-i]**

1. 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]↔A]9], A[1]↔A]8], A[2]↔A]7], A[3]↔A]6] и A[4]↔A]5]

1. в результате массив оказывается «развернут» наоборот, элемент A[0] (он был равен 9) стал последним, следующий (A[1]=8) – предпоследним и т.д., то есть получили

0 1 2 3 4 5 6 7 8 9

1. Ответ: 2.

### Ещё пример задания:

*Дан фрагмент программы, обрабатывающей двухмерный массив A размера n×n.*

**k := 1;**

**for i:=1 to n do begin**

**c := A[i,i];**

**A[i,i] := A[k,i];**

**A[k,i] := c;**

**end**

*Представим массив в виде квадратной таблицы, в которой для элемента массива A[i,j] величина i является номером строки, а величина j – номером столбца, в котором расположен элемент. Тогда данный алгоритм меняет местами*

1) два столбца в таблице

2) две строки в таблице

3) элементы диагонали и k-ой строки таблицы

4) элементы диагонали и k-го столбца таблицы

**Решение**:

1. сначала разберемся, что происходит внутри цикла; легко проверить (хотя бы ручной прокруткой, если вы сразу не узнали стандартный алгоритм), что операторы

**c := A[i,i];**

**A[i,i] := A[k,i];**

**A[k,i] := c;**

меняют местами значения **A[i,i]** и **A[k,i]**, используя переменную **c** в качестве вспомогательной ячейки;

1. элемент матрицы **A[i,i]**, у которого номера строки и столбца одинаковые, стоит на главной диагонали; элемент **A[k,i]** стоит в том же столбце **i**, но в строке с номером **k**; это значит, что в столбце **i** меняются местами элемент на главной диагонали и элемент в строке **k**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | **i** |  |
| **k** |  |  | **A[k,i]** |  |
|  |  |  |  |  |
| **i** |  |  | **A[i,i]** |  |
|  |  |  |  |  |

1. так как эти операторы находятся в цикле, где переменная **i** принимает последовательно все значения от 1 до **n**, обмен выполняется для всех столбцов матрицы; то есть, все элементы главной диагонали меняются с соответствующими элементами строки **k**
2. перед циклом стоит оператор присваивания **k := 1;**, а после него переменная **k** не меняется; поэтому в программе элементы главной диагонали обмениваются с первой строкой
3. таким образом, правильный ответ – 3.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * сложность этой задачи в том, что все действия нужно «прокручивать в уме» (или на бумаге), не используя компьютер для отладки * главная проблема – не перепутать столбцы и строки; номер строки – это (по соглашению) первый индекс элемента матрицы, а номер столбца – второй |

|  |
| --- |
| **Совет:**   * чтобы понять, что делает программа, часто бывает полезно сделать ручную прокрутку на матрице небольшого размера, например, 3 на 3 или 4 на 4. * если матрица небольшая (скажем, 5 на 5) можно (а иногда и нужно) вообще сделать все вычисления вручную и посмотреть, что получится |

### Еще пример задания:

*Значения двух массивов A[1..100] и B[1..100] задаются с помощью следующего фрагмента программы:*

**for n:=1 to 100 do**

**A[n] := (n-80)\*(n-80);**

**for n:=1 to 100 do**

**B[101-n] := A[n];**

*Какой элемент массива B будет наибольшим?*

1) B[1] 2) B[21] 3) B[80] 4) B[100]

**Решение**:

1. здесь два цикла, в первом из них заполняется массив **А**, во втором – массив **В**
2. в элемент массива **A[n]** записывается квадрат числа **n-80**; все элементы массива **А** неотрицательны (как квадраты чисел)
3. посмотрим чему равны некоторые элементы массива **А**:

A[1] = (1–80)2  = (–79)2 = 792 A[2] = (2–80)2  = (–78)2 = 782

...

A[80] = (80–80)2  = (0)2 = 0A[81] = (81–80)2  = (1)2 = 1

...

A[99] = (99–80)2  = 192 A[100] = (100–80)2  = 202

1. таким образом, при увеличении n от 1 до 80 значение A[n] уменьшается от 792  до нуля, а потом (для n > 80) возрастает до 202
2. отсюда следует, что максимальное значение в массиве A – это A[1] = 792
3. во втором цикле для всех номеров n от 1 до 100 выполняется оператор

**B[101-n] := A[n];**

который просто переписывает элементы массива A в массив В, «развертывая» массив в обратном порядке (элемент A[1] будет записан в B[100], а A[100] – в B[1])

1. A[1], наибольший элемент массива А, будет записан в B[100], поэтому B[100] – наибольший элемент в массиве В
2. таким образом, правильный ответ – 4.

### Еще пример задания:

*Значения элементов двухмерного массива A[1..10,1..10] задаются с помощью следующего фрагмента программы:*

**for i:=1 to 10 do**

**for k:=1 to 10 do**

**if i > k then**

**A[i,k] := 1**

**else A[i,k] := 0;**

*Чему равна сумма элементов массива после выполнения этого фрагмента программы?*

**Решение**:

1. в программе есть вложенный цикл, в котором переменная **i** обозначает строку, а **k** – столбец матрицы
2. элементы, для которых **i=k** – это главная диагональ матрицы, поэтому элементы, для которых **i > k** (только они будут равны 1), находятся под главной диагональю
3. в первой строке единичных элементов нет, во второй есть один такой элемент, в третьей – 2, в последней (10-ой) их 9, поэтому сумма элементов массива равна

1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 = 45

1. таким образом, правильный ответ – 45.
2. при большом размере массива (например, 100 на 100) суммирование может оказаться трудоемким, поэтому лучше вспомнить формулу для вычисления суммы элементов арифметической прогрессии (именно такая прогрессия у нас, с шагом 1):

,

где  - количество элементов, а и  – соответственно первый и последний элементы последовательности; в данном случае имеем

.

1. если приведенная выше формула прочно забыта, можно попытаться сгруппировать слагаемые в пары с равной суммой (как сделал, будучи школьником, великий математик К.Ф. Гаусс), например:



### Еще пример задания:

*Значения элементов двухмерного массива A[1..10,1..10] сначала равны 5. Затем выполняется следующий фрагмент программы:*

**for i:=1 to 5 do**

**for j:=1 to 4 do begin**

**A[i,j]:=A[i,j]+5; { 1 }**

**A[j,i]:=A[j,i]+5; { 2 }**

**end;**

*Сколько элементов массива будут равны 10?*

**Решение (вариант 1, анализ алгоритма)**:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |

1. обратим внимание, что в двойном цикле переменная **i** изменяется от 1 до 5, а **j** – от 1 до 4 (на 1 шаг меньше)
2. внутри цикла в операторе, отмеченном цифрой 1 в комментарии, в записи **A[i,j]** переменная **i** – это строка, а **j** – столбец, поэтому по одному разу обрабатываются все элементы массива, выделенные зеленым цветом:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |

1. это значит, что если оставить только один первый оператор внутри цикла, все выделенные элементы увеличиваются на 5 и станут равны 10
2. теперь рассмотрим второй оператор внутри цикла: в записи **A[j,i]** переменная **i** – это столбец, а **j** – строка, поэтому по одному разу обрабатываются (увеличиваются на 5 ) все элементы массива, выделенные рамкой красного цвета на рисунке справа
3. теперь хорошо видно, что левый верхний угол массива (квадрат 4 на 4, синяя область) попадает в обе области, то есть, эти 16 элементов будут дважды увеличены на 5: они станут равны 15 после выполнения программы
4. элементы, попавшие в зеленый и красный «хвостики» обрабатываются (увеличиваются на 5) по одному разу, поэтому они-то и будут равны 10
5. всего таких элементов – 8 штук
6. таким образом, правильный ответ – 8.

**Решение (вариант 2, прокрутка небольшого массива)**:

1. понятно, что в программе захватывается только левый верхний угол массива, остальные элементы не меняются
2. сократим размер циклов так, чтобы можно было легко выполнить программу вручную; при этом нужно сохранить важное свойство: внутренний цикл должен содержать на 1 шаг меньше, чем внешний:

**for i:=1 to 3 do**

**for j:=1 to 2 do begin**

**A[i,j]:=A[i,j]+5; { 1 }**

**A[j,i]:=A[j,i]+5; { 2 }**

**end;**

1. выполняя вручную этот вложенный цикл, получаем

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| 1 | 15 | 15 | 10 | 5 | 5 |
| 2 | 15 | 15 | 10 | 5 | 5 |
| 3 | 10 | 10 | 5 | 5 | 5 |
| 4 | 5 | 5 | 5 | 5 | 5 |
| 5 | 5 | 5 | 5 | 5 | 5 |

1. видим, что в самом углу получился квадрат 2 на 2 (по количеству шагов внутреннего цикла), в котором все элементы равны 15; по сторонам этого квадрата стоят 4 элемента, равные 10, их количество равно удвоенной стороне квадрата
2. в исходном варианте внутренний цикл выполняется 4 раза, поэтому угловой квадрат будет иметь размер 4 на 4; тогда 8 элементов, граничащих с его сторонами, будут равны 10
3. таким образом, правильный ответ – 8.

|  |
| --- |
| **Возможные проблемы**:   * упрощая задачу, нельзя потерять ее существенные свойства: например, здесь было важно, что внутренний цикл содержит на 1 шаг меньше, чем внешний |

# 20 (повышенный уровень, время – 5 мин)

**Тема**: Анализ программы, содержащей подпрограммы, циклы и ветвления.

**Что нужно знать**:

* операции целочисленного деления (**div**) и взятия остатка (**mod**)
* как работают операторы присваивания, циклы и условные операторы в языке программирования

### Пример задания:

**P-06**. *Ниже записан алгоритм. Укажите наименьшее пятизначное число , при вводе которого алгоритм печатает сначала 4, а потом 2.*

**var x, y, a, b: longint;**

**begin**

**a := 0;**

**b := 0;**

**readln(x);**

**while x > 0 do begin**

**y := x mod 10;**

**if y > 3 then a := a + 1;**

**if y < 8 then b := b + 1;**

**x := x div 10**

**end;**

**writeln(a);**

**writeln(b)**

**end.**

**Решение:**

1. видим, что в последней строке выводятся на экран переменные a и b, поэтому сначала нужно определить, что они обозначают в программе
2. перед началом цикла переменные a и b обнуляются
3. на каждом шаге цикла при выполнении некоторых условий переменные a и b увеличиваются на 1, то есть представляют собой счётчики
4. увеличение переменных зависит от значения y = x mod 10, то есть от последней цифры числа
5. если последняя цифра числа больше 3, увеличивается счётчик a, если меньше 8 – счётчик b;
6. в конце каждого шага цикла операция x:=x div 10 отсекает последнюю цифру в десятичной записи числа
7. цикл заканчивается, когда перестаёт выполняться условие x > 0, то есть, когда все цифры исходного числа отброшены
8. таким образом, делаем вывод: после завершения цикла в переменной a находится количество цифр, больших 3, в десятичной записи числа, а в переменной b – количество цифр, меньших 8
9. если было выведено 4 и 2, то в числе 4 цифры больше 3 и 2 цифры меньше 8
10. так как число пятизначное, есть 4 + 2 – 5 = одна цифра, которая больше 3 и меньше 8 одновременно; она должна быть минимальной, поэтому эта цифра 4
11. для того чтобы число было минимальным, ещё одна цифра должна быть минимальной и меньшей 3 – это старшая 1, и три цифры минимальные из цифр, больших или равных 8, то есть три цифры 8
12. ответ: 14888.

### Ещё пример задания:

**P-05**. *Ниже записан алгоритм. Сколько существует таких чисел, при вводе которых алгоритм печатает сначала 2, а потом 12?*

**var x, a, b: integer;**

**begin**

**readln(x);**

**a:=0; b:=0;**

**while x>0 do begin**

**a:=a + 1;**

**b:=b + (x mod 10);**

**x:=x div 10;**

**end;**

**writeln(a); write(b);**

**end.**

**Решение:**

1. видим, что в последней строке выводятся на экран переменные a и b, поэтому сначала нужно определить, что они обозначают в программе
2. перед началом цикла переменные a и b обнуляются
3. на каждом шаге цикла при выполнении некоторого условия переменная a увеличивается на 1, а b увеличивается на x mod 10, то есть, на остаток от деления x на 10 – это последняя цифра десятичной записи числа x
4. в конце каждого шага цикла операция x:=x div 10 отсекает последнюю цифру в десятичной записи числа
5. цикл заканчивается, когда перестаёт выполняться условие x > 0, то есть, когда все цифры исходного числа отброшены
6. таким образом, делаем вывод: после завершения цикла в переменной a находится количество цифр в десятичной записи числа, а в переменной b – их сумма
7. если было выведено 2 и 12, то в числе 2 цифры, и их сумма равна 12; таким образом, нам нужно найти все двузначные числа, в котором сумма значений цифр равна 12
8. число 12 может быть разложено на два слагаемых, меньших 10, как

12 = 3 + 9 = 4 + 8 = 5 + 7 = 6 + 6 = 7 + 5 = 8 + 4 = 9 + 3,

нам подходят числа 39, 48, 57, 66, 75, 84 и 93

1. всего таких чисел - 7
2. ответ: 7.

### Ещё пример задания:

**P-04**. *Ниже записан алгоритм. Укажите наименьшее из таких чисел , при вводе которых алгоритм печатает сначала 2, а потом 15.*

**var x, a, b: integer;**

**begin**

**readln(x);**

**a:=0; b:=1;**

**while x>0 do begin**

**a:=a+1;**

**b:=b\*(x mod 10);**

**x:= x div 10**

**end;**

**writeln(a); write(b)**

**end.**

**Решение:**

1. видим, что в последней строке выводятся на экран переменные a и b, поэтому сначала нужно определить, что они обозначают в программе
2. перед началом цикла переменная a обнуляется, а переменная b равна 1
3. на каждом шаге цикла при выполнении некоторого условия переменная a увеличивается на 1, а b умножается на x mod 10, то есть, на остаток от деления x на 10 – это последняя цифра десятичной записи числа x
4. в конце каждого шага цикла операция x:=x div 10 отсекает последнюю цифру в десятичной записи числа
5. цикл заканчивается, когда перестаёт выполняться условие x > 0, то есть, когда все цифры исходного числа отброшены
6. таким образом, делаем вывод: после завершения цикла в переменной a находится количество цифр в десятичной записи числа, а в переменной b – их произведение
7. если было выведено 2 и 15, то в числа 2 цифры, и их произведение равно 15; таким образом, нам нужно найти минимальное двузначное число, в котором произведение значений цифр равно 15
8. поскольку число 15 может быть разложено на два сомножителя, меньших 10, только как 3⋅5, минимальное подходящее число – 35.
9. ответ: 35.

### Ещё пример задания:

**P-03**. *Ниже записан алгоритм. Укажите наименьшее из таких чисел , при вводе которых алгоритм печатает сначала 3, а потом 2.*

**var x, a, b, c: integer;**

**begin**

**readln(x);**

**a:= 0; b:= 0;**

**while x > 0 do begin**

**c:= x mod 2;**

**if c = 0 then a:= a + 1**

**else b:= b + 1;**

**x:= x div 10;**

**end;**

**writeln(a);**

**writeln(b);**

**end.**

**Решение:**

1. видим, что в последний строках выводятся на экран переменные a и b, поэтому сначала нужно определить, что они обозначают в программе
2. перед началом цикла обе переменные обнуляются
3. на каждом шаге цикла при выполнении некоторого условия переменная a увеличивается на 1, а если это условие не выполняется, то на 1 увеличивается b; таким образом, обе переменных – счётчики
4. теперь посмотрим на условие c = 0: в предыдущей строке в переменную c записывается остаток от деления числа x на 2, то есть, переменная c определяет четность числа или, что равносильно, **чётность его последней цифры**
5. если последняя цифра чётная, то увеличивается счётчик a, а если нечётная – увеличивается счётчик b
6. в конце каждого шага цикла операция x:=x div 10 отсекает последнюю цифру в десятичной записи числа
7. таким образом, делаем вывод: после завершения цикла в переменной a находится количество чётных цифр в десятичной записи числа, а в переменно b – количество нечётных цифр
8. если было выведено 3 и 2, то в числа 5 цифр, из них 3 чётных и 2 нечётных; таким образом, нам нужно найти минимальное пятизначное число, в котором 3 чётные и 2 нечётные цифры
9. минимальная чётная цифра – это 0, минимальная начётная – 1; 0 не может стоять на первом месте, поэтому число начинается с 1
10. для получения минимального числа после 1 должны идти нули и последняя цифра – снова 1
11. ответ: 10001

### Ещё пример задания:

**P-02**. *Ниже записан алгоритм. После выполнения алгоритма было напечатано 3 числа. Первые два напечатанных числа – это числа 9 и 81. Какое наибольшее число может быть напечатано третьим?*

**var x, y, z: integer;**

**r, a, b: integer;**

**begin**

**readln(x, у);**

**if у > x then begin**

**z:= x; x:= у; у:= z;**

**end;**

**a:= x; b:= y;**

**while b > 0 do begin**

**r:= a mod b;**

**a:= b;**

**b:= r;**

**end;**

**writeln(a);**

**writeln(x);**

**write(у);**

**end.**

**Решение:**

1. сложность этой задачи состоит в том, чтобы разобраться в алгоритме
2. сначала вводятся два числа и переставляются так, чтобы в переменной **x** было наибольшее число, а в переменной **y** – наименьшее из двух:

**if у > x then begin**

**z:= x; x:= у; у:= z;**

**end;**

1. затем исходные значения копируются в переменные **a** и **b** и с ними выполняется следующий алгоритм

**while b > 0 do begin**

**r:= a mod b;**

**a:= b;**

**b:= r;**

**end;**

его суть сводится к тому, что меньшее из двух чисел, **a** и **b**, каждый раз заменяется на остаток от деления большего на меньшее до тех пор, пока этот остаток не станет равен нулю;

1. делаем вывод, что это классический [Алгоритм Евклида](http://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%95%D0%B2%D0%BA%D0%BB%D0%B8%D0%B4%D0%B0), который служит для вычисления наибольшего общего делителя (НОД) двух чисел; это делитель в результате оказывается в переменной **a**
2. смотрим, что выводится на экран: сначала значение переменной **a** (наибольший общий делитель исходных чисел, НОД(x,y)), затем значение **x** (большее из исходных чисел) и значение **y** (меньшее из исходных чисел)
3. по условию первое число – 9, второе – 81, поэтому третье число должно быть меньше, чем 81, и НОД(81,y) = 9
4. наибольшее число, которое меньше 81 и делится на 9, равно 72 (обратите внимание, что исходные числа не могут быть равны, потому что в этом случае их НОД был бы равен 81)
5. ответ: 72

### Ещё пример задания:

**P-01**. *Ниже записана программа. Получив на вход число , эта программа печатает два числа,  и . Укажите наибольшее из таких чисел , при вводе которых алгоритм печатает сначала 3, а потом 7.*

**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.**

**Решение:**

1. для решения задачи необходимо понять, что делает эта программа
2. если это не видно сразу, можно выполнить ручную прокрутку для какого-то простого числа, например, для числа 251:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **оператор** | **условие** | **x** | **L** | **M** |
| **readln(x);** |  | **251** | **?** | **?** |
| **L:=0; M:=0;** |  |  | **0** | **0** |
| **while x > 0 do…** | **251 > 0? да** |  |  |  |
| **L:=L+1;** |  |  | **1** |  |
| **if M<(x mod 10) then…** | **M <(251 mod 10)? да** |  |  |  |
| **M:=x mod 10;** |  |  |  | **1** |
| **x:=x div 10;** |  | **25** |  |  |
| **while x > 0 do…** | **25 > 0? да** |  |  |  |
| **L:=L+1;** |  |  | **2** |  |
| **if M<(x mod 10) then…** | **M <(25 mod 10)? да** |  |  |  |
| **M:=x mod 10;** |  |  |  | **5** |
| **x:=x div 10;** |  | **2** |  |  |
| **while x > 0 do…** | **2 > 0? да** |  |  |  |
| **L:=L+1;** |  |  | **3** |  |
| **if M<(x mod 10) then…** | **M <(2 mod 10)? нет** |  |  |  |
| **x:=x div 10;** |  | **0** |  |  |
| **while x > 0 do…** | **0 > 0? нет** |  |  |  |
| **writeln(L); write(M);** |  |  | **3** | **5** |

1. можно догадаться, что в результате работы программы в переменной L окажется число цифр числа, а в переменной M – наибольшая цифра, но это предположение нужно постараться доказать
2. нужно вспомнить (и запомнить), что для целого числа** остаток от деления на 10 (**x mod 10**) – это последняя цифра в десятичной записи числа, а целочисленное деление (**x div 10**) отсекает последнюю цифру, то есть из 123 получается 12
3. рассмотрим цикл, число шагов которого зависит от изменения переменной **x**:

**while x > 0 do begin**

**...**

**x:= x div 10; { отсечение последней цифры }**

**end;**

здесь оставлены только те операторы, которые влияют на значение **x**

1. из приведенного цикла видно, что на каждом шаге от десятичной записи **x** отсекается последняя цифра до тех пор, пока все цифры не будут отсечены, то есть **x** не станет равно 0; поэтому **цикл выполняется столько раз, сколько цифр в десятичной записи введенного числа**
2. на каждом шаге цикла переменная **L** увеличивается на 1:

**L:=L+1;**

других операторов, меняющих значение **L**, в программе нет; поэтому после завершения цикла **в переменной L действительно находится количество цифр**

1. теперь разберемся с переменной **M**, которая сначала равна 0; оператор, в котором она меняется, выглядит так:

**if M < (x mod 10) then begin**

**M:=x mod 10;**

**end;**

учитывая, что **x mod 10** – это последняя цифра десятичной записи числа, получается что если эта цифра больше, чем значение M, она записывается в переменную M;

1. этот оператор выполняется в цикле, причем выражение **x mod 10** по очереди принимает значения всех цифр исходного числа; поэтому после завершения циклам **в переменной M окажется наибольшая из всех цифр**, то есть наша догадка подтверждается
2. итак, по условию задачи фактически требуется найти наибольшее трехзначное число, в котором наибольшая цифра – 7; очевидно, что это 777.
3. ответ: 777.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + это очень неплохая задача на понимание, тут достаточно сложно «вызубрить» метод решения, можно только освоить последовательность (системность) анализа   + ручной прокрутки в такой задаче недостаточно, по её результатам можно угадать алгоритм, но можно и не угадать; в критическом случае можно сделать ручную прокрутку для нескольких чисел им попытаться понять закономерность |

### Ещё пример задания:

**P-00**. *Ниже записана программа. Получив на вход число , эта программа печатает два числа,  и . Укажите наибольшее из таких чисел , при вводе которых алгоритм печатает сначала 3, а потом 120.*

**var x, L, M: integer;**

**begin**

**readln(x);**

**L:=0; M:=1;**

**while x > 0 do begin**

**L:=L+1;**

**M:= M\*(x mod 8);**

**x:= x div 8;**

**end;**

**writeln(L); write(M);**

**end.**

**Решение:**

1. для решения задачи необходимо понять, что делает эта программа; повторяя рассуждения из предыдущего примера, выясняем, что
   1. переменная **L** с каждым шагом цикла увеличивается на 1
   2. переменная **x** на каждом шаге цикла делится на 8 и остаток отбрасывается

поэтому можно сделать вывод, что в конце цикла переменная **L** будет равна **количеству цифр** введенного числа, записанного в *восьмеричной* системе счисления; таким образом, восьмеричная запись числа содержит ровно 3 цифры

1. выражение **x mod 8** – это последняя цифра восьмеричной записи числа; на каждом шаге цикла переменная **M** умножается на эту величину, поэтому в результате в **M** будет записано **произведение всех цифр** восьмеричной записи введенного числа
2. по условию это произведение равно 120, то есть , где *a*, *b* и *с* – числа от 0 до 7 (которые в восьмеричной системе счисления записываются одной цифрой)
3. поскольку нам нужно наибольшее число, перебираем делители числа 120, начиная со старшей цифры – 7; видим, что 120 на 7 не делится, поэтому такой цифры в восьмеричной записи числа нет
4. но 120 делится на 6, поэтому старшей цифрой может быть 6 – только в том случае, когда второй сомножитель можно представить в виде произведения двух чисел в интервале 1..6
5. делим 120 на 6, получаем 20; это число представляется как произведение 5 и 4, каждое из этих чисел записывается в виде одной восьмеричной цифры, то есть, они нам подходят
6. вспомним, что нас интересует максимальное число, поэтому цифры нужно выстроить в порядке убывания: 6548
7. заметим, что мы получили число в восьмеричной системе, а ответ нужно дать в десятичной; переводим: 6548 = 6·82 + 5·81 + 4·80 = 428.
8. ответ: 428.

|  |
| --- |
| **Возможные ловушки и проблемы**:   * + поскольку в цикле идет деление на 8, мы получаем цифры числа в восьмеричной системе; каждая из них должна быть в интервале 0..7 (не может быть 8 и 9)   + на последнем шаге нужно не забыть перевести число из восьмеричной системы в десятичную |

# 21 (повышенный уровень, время – 6 мин)

**Тема**: Анализ программы с подпрограммами.

**Что нужно знать**:

* функция – это вспомогательный алгоритм, который возвращает некоторое значение–результат
* в Паскале функция располагается выше основной программы и оформляется следующим образом (вместо многоточия могут быть любые операторы):

**function F(x: integer):integer;**

**begin**

**...**

**F:= <*результат функции*>**

**end;**

* в заголовке функции записывают имя функции, в скобках – список параметров, далее через двоеточие – тип возвращаемого значения; в приведенном примере функция F принимает один целый параметр, к которому внутри функции нужно обращаться по имени x, и возвращает целое число
* результат функции записывается в специальную переменную, имя которой совпадает с именем функции; объявлять эту переменную не нужно
* если параметров несколько, для каждого из них указывают тип:

**function F(x: integer; y: integer):integer;**

* если несколько соседних параметров имеют одинаковый тип, можно их объединить в список:

**function F(x, y: integer):integer;**

* следующая программа ищет наименьшее значение функции F(x) на интервале [a,b], просматривая значения от a до b с шагом 1:

**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 , то абсцисса, соответствующая точке минимума, вычисляется по формуле



этот результат можно получить (вывести, если забыли), например, так:

* в критической точке (точке минимума, точке максимума или точке перегиба) производная функции обращается в 0;
* находим производную 
* приравниваем ее к нулю: .
* если квадратный трехчлен задан в виде , то абсцисса, соответствующая точке минимума, вычисляется по формуле



### Пример задания:

*Определите, какое значение H нужно ввести, чтобы число, напечатанное в результате выполнения следующего алгоритма, было наименьшим.*

**var a,b,t,M,R,H :integer;**

**Function F(H, x: integer):integer;**

**begin**

**F := 11\*(x-H)\*(x-H)+13;**

**end;**

**BEGIN**

**readln(H);**

**a := -10; b := 30;**

**M := a; R := F(H, a);**

**for t := a to b do begin**

**if (F(H, t) > R) then begin**

**M := t;**

**R := F(H, t)**

**end**

**end;**

**write(R)**

**END.**

**Решение:**

1. заметим, величина H в программе не изменяется, то есть фактически выполняет роль константы; она передаётся в функцию и влияет на значение функции
2. что в программе есть цикл, в котором переменная t принимает последовательно все целые значения в интервале от a до b:

**for t:=a to b do begin**

**...**

**end;**

1. до начала цикла в переменную M записывается значение a, а в переменную R – значение функции в точке a:

**M:=a; R:=F(H,a);**

1. внутри цикла есть условный оператор, в котором вычисляется значение функции F(t) и сравнивается со значением переменной R:

**if (F(H,t) > R)then begin**

**M:=t;**

**R:=F(H,t)**

**end;**

если новое значение функции больше, чем значение R, в R записывается значение функции в точке t, а в переменной M запоминается само значение t (аргумент функции, соответствующий значению в R)

1. в результате анализа пп. 1-3 можно сделать вывод, что цикл ищет максимум функции F(H,t) на интервале от a до b
2. заметим, что выводится значение R, а величина M не выводится и не влияет на вычисление R, поэтому можно не обращать на неё внимания
3. функция F вычисляет значение

**F:=11\*(x-H)\*(x-H) + 13;**

1. график этой эта функции – парабола с ветвями, направленными вверх (коэффициент при x2 = 11 > 0)
2. вершина параболы находится в точке *x = H*, ветви идут симметрично влево и вправо вверх
3. при изменении *H* парабола двигается влево или вправо (но не вверх-вниз!)
4. итак, мы ищем максимальное значение квадратичной функции, и хотим, чтобы это значение было наименьшим
5. давайте подвигаем параболу в пределах отрезка [*a; b*]:

*a*

*b*

*R*

*a*

*b*

*R*

*a*

*b*

*R*

1. видно, что минимальное значение максимума будет тогда, когда вершина параболы будет расположена точно в середине отрезка [*a; b*]
2. отсюда требуемое значение *H* равно среднему арифметическому между *a = –*10 *и b =* 30:

*H =* (–10 + 30) / 2 = 10

1. Ответ: 10.

### Ещё пример задания (И. Тощенко):

*Напишите в ответе число различных значений входной переменной k, при которых программа выдаёт тот же ответ, что и при входном значении k = 35. Значение k = 35 также включается в подсчёт различных значений k.*

**var k, i : longint;**

**function F(x: longint) : longint;**

**begin**

**F:=2\*x\*x+3\*x+2**

**end;**

**begin**

**i := 15;**

**readln(K);**

**while (i> 0) and (F(i) > K) do**

**i:=i-1;**

**writeln(i)**

**end.**

**Решение** (И. Тощенко)**:**

1. Вычислим значения функции F при i=1,2,3…

*i=*0: *f*(0)*=2*

*i=*1: *f*(1)*=7*

*i=*2: *f*(2)*=16*

*i=*3: *f*(3)*=29*

*i=*4: *f*(4)*=46*

1. Заданное значение К попадает в отрезок [29;45].
2. Следовательно, всего 45-29+1=17 чисел.
3. ответ: 17.

### Ещё пример задания:

*Напишите в ответе число различных значений входной переменной k, при которых программа выдаёт тот же ответ, что и при входном значении k = 64. Значение k = 64 также включается в подсчёт различных значений k.*

**var k, i : longint;**

**function f(n: longint) : longint;**

**begin**

**f := n \* n**

**end;**

**begin**

**readln(k);**

**i := 12;**

**while (i>0) and (f(i)>=k) do**

**i := i-1;**

**writeln(i)**

**end.**

**Решение:**

1. заметим, что функция F(x) вычисляет квадрат переданного ей числа
2. в теле основной программы выполняется цикл с условием, который заканчивается, когда значение функции станет меньше k
3. на каждом шаге цикла уменьшается значение переменной i, начиная с 12; цикл также заканчивается, когда значение переменной i станет равно 0
4. после окончания цикла программа выводит значение переменной i.
5. итак, функция выводит первое натуральное значение i, квадрат которого меньше, чем введённое с клавиатуры значение переменной k
6. при k = 64 программа выведет значение 7, поскольку это наибольшее натуральное число, квадрат которого меньше, чем 64
7. фактически нужно ответить на вопрос: сколько есть таких чисел k, которые меньше или равны 82 = 64 и больше, чем 72 = 49 (легко проверить, что при k=65 программа выведет значение 8, в при k=49 – значение 6)
8. в диапазоне [50;64] всего 64-50+1=15 чисел, это и есть правильный ответ.
9. ответ: 15.

### Ещё пример задания:

*Определите, количество чисел K, для которых следующая программа выведет такой же результат, что и для K = 24:*

**var i, k: integer;**

**function F(x:integer):integer;**

**begin**

**F:=x\*x\*x;**

**end;**

**begin**

**i := 12;**

**readln(K);**

**while (i>0) and (F(i) > K) do**

**i:=i-1;**

**writeln(i);**

**end.**

**Решение:**

1. заметим, что функция F(x) вычисляет куб переданного ей числа
2. перед началом цикла значение переменной i равно 12, в цикле оно уменьшается
3. цикл while останавливается, когда переменная i становится равна нулю или значение функции F(i) становится меньше или равно K
4. таким образом, в данной фактически требуется найти количество натуральных чисел в диапазоне [1..12], куб которых больше, чем K = 24
5. определим, у скольких чисел куб меньше, чем 24; это все числа, меньшие, чем , то есть, только числа 1 и 2; поэтому программа выведет 2 – первое число, куб которого меньше или равен 24
6. остаётся определить, когда программа выведет именно 2; это случится при всех K, при которых , то есть при ; в этот диапазон входит 27-8 = 19 чисел
7. Ответ: 19.

### Ещё пример задания:

*Определите, количество чисел K, для которых следующая программа выведет такой же результат, что и для K = 24:*

**var i, k: integer;**

**function F(x:integer):integer;**

**begin**

**if x = 1 then**

**F:=1**

**else F:=x\*F(x-1);**

**end;**

**begin**

**i := 15;**

**readln(K);**

**while (i>0) and (F(i) > K) do**

**i:=i-1;**

**writeln(i);**

**end.**

**Решение:**

1. заметим, что рекурсивная функция F(x) вычисляет факториал переданного ей числаx, то есть произведение x!=1⋅2⋅3⋅...⋅(x-1) ⋅x
2. повторяя рассуждения предыдущей задачи, определяем, что функция выведет количество натуральных чисел, факториалы которых меньше или равны K
3. выпишем факториалы первых натуральных чисел:

1! = 1, 2! = 2, 3! = 6, 4! = 24, 5! = 120, 6! = 720, …

1. из этого ряда факториалы первых четырех чисел меньше или равны 24, поэтому при K=24 функция выведет число 4
2. программа выведет именно 4 при всех K, при которых , то есть при ; в этот диапазон входит 120-24 = 96 чисел
3. Ответ: 96.

### Ещё пример задания:

*Определите, какое число будет напечатано в результате выполнения следующего алгоритма:*

**var a, b, t, N, P :integer;**

**Function F(x: integer):integer;**

**begin**

**F := 16\*(9-x)\*(9-x)+127;**

**end;**

**BEGIN**

**a := -25; b := 25;**

**P := 130;**

**N := 0;**

**for t := a to b do begin**

**if (F(t) > P) then begin**

**N := N+1;**

**end;**

**end;**

**write(N);**

**END.**

**Решение:**

1. заметим, что в конце работы программы на экран выводится значение переменной N
2. в программе значение N сначала обнуляется, а затем на каждом шаге цикла увеличивается на 1 при условии, что значение функции F(t) больше значения P = 130; таким образом, N – это счётчик точек с целочисленными значениями на отрезке [-25;25], в которых значение функции больше, чем 130
3. график функция 16\*(9-x)\*(9-x)+127 – парабола с ветвями вверх, минимальное значение в точке x = 9 равно 127
4. значение функции при x = 8 и x = 10 (рядом с точкой минимума) равны 16+127 = 143, поэтому только в одной точке x = 9 не выполняется условие F(t) > P
5. всего на интервале [-25;25] есть 51 точка с целочисленными координатами; во всех, за исключением одной условие F(t) > P выполняется, то есть счётчик увеличивается на 1
6. Ответ: 50.

### Ещё пример задания:

*Определите, какое число будет напечатано в результате выполнения следующего алгоритма:*

**Var a,b,t,M,R:integer;**

**Function F(x:integer):integer;**

**begin**

**F:=(x\*x-4)\*(x\*x-4)+6;**

**end;**

**BEGIN**

**a:=-10; b:=10;**

**M:=a; R:=F(a);**

**for t:=a to b do begin**

**if (F(t)<R)then begin**

**M:=t;**

**R:=F(t);**

**end;**

**end;**

**write(M+6);**

**END.**

**Решение:**

1. заметим, что в программе есть цикл, в котором переменная t принимает последовательно все целые значения в интервале от a до b:

**for t:=a to b do begin**

**...**

**end;**

1. до начала цикла в переменную M записывается значение a, а в переменную R – значение функции в точке a:

**M:=a; R:=F(a);**

1. внутри цикла есть условный оператор, в котором вычисляется значение функции F(t) и сравнивается со значением переменной R:

**if (F(t)<R)then begin**

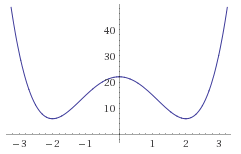
**M:=t;**

**R:=F(t);**

**end;**

если новое значение функции меньше, чем значение R, в R записывается значение функции в точке t, а в переменной M запоминается само значение t (аргумент функции, соответствующий значению в R)

1. в результате анализа пп. 1-3 можно сделать вывод, что цикл ищет минимум функции F(t) на интервале от a до b, и после выполнения цикла в переменной M оказывается значение аргумента t, при котором функция достигает минимума на заданном интервале (здесь это интервал [-10, 10])
2. функция, которая используется в программе, – это квадратичная парабола: , её ветви направлены вверх (коэффициент при  положительный, равен 1); она имеет два минимума в точках  и 



1. обе точки минимума находятся на отрезке [-10;10], поэтому программа найдёт одну из этих точек; вопрос: какую именно?
2. для квадратичной параболы обе точки минимума имеют одинаковую -координату, а запоминание новой точки минимума происходит только тогда, когда только что вычисленное значение F(t) станет **строго меньше**, чем хранящееся в переменной R:

**if (F(t) < R)then begin**

**M:=t;**

**R:=F(t);**

**end;**

1. поэтому в точке второго минимума  никаких изменений не произойдет, и в переменной M останется значение «–2»; таким образом, будет найдет первый минимум
2. обратим внимание, что на экран выводится не M, а M+6, поэтому результат будет равен

(-2)+6=4

1. Ответ: 4.

### Ещё пример задания:

*Определите, какое число будет напечатано в результате выполнения следующего алгоритма:*

**Var a,b,t,M,R:integer;**

**Function F(x:integer):integer;**

**begin**

**F:=4\*(x-1)\*(x-3);**

**end;**

**BEGIN**

**a:=-20; b:=20;**

**M:=a; R:=F(a);**

**for t:=a to b do begin**

**if (F(t)<R)then begin**

**M:=t;**

**R:=F(t);**

**end;**

**end;**

**write(M);**

**END.**

**Решение (способ 1, ручная прокрутка, перебор):**

1. заметим, что в программе есть цикл, в котором переменная t принимает последовательно все целые значения в интервале от a до b:

**for t:=a to b do begin**

**...**

**end;**

1. до начала цикла в переменную M записывается значение a, а в переменную R – значение функции в точке a:

**M:=a; R:=F(a);**

1. внутри цикла есть условный оператор, в котором вычисляется значение функции F(t) и сравнивается со значением переменной R:

**if (F(t)<R)then begin**

**M:=t;**

**R:=F(t);**

**end;**

если новое значение функции меньше, чем значение R, в R записывается значение функции в точке t, а в переменной M запоминается само значение t (аргумент функции, соответствующий значению в R)

1. в результате анализа пп. 1-3 можно сделать вывод, что цикл ищет минимум функции F(t) на интервале от a до b, и после выполнения цикла в переменной M оказывается значение аргумента t, при котором функция достигает минимума на заданном интервале (здесь это интервал [-20, 20])
2. функция F вычисляет значение

**F:=4\*(x-1)\*(x-3);**

1. перебираем все значения t от a до b, и для каждого вычисляем соответствующее значение функции:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **t** | -20 | -19 | -18 | -17 | -16 | -15 | -14 | -13 | -12 | -11 | -10 | -9 | -8 | -7 | -6 | -5 | -4 | -3 | -2 | -1 | 0 |
| **F** | 1932 | 1760 | 1596 | 1440 | 1292 | 1152 | 1020 | 896 | 780 | 672 | 572 | 480 | 396 | 320 | 252 | 192 | 140 | 96 | 60 | 32 | 12 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **t** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |  |
| **F** | 0 | -4 | 0 | 12 | 32 | 60 | 96 | 140 | 192 | 252 | 320 | 396 | 480 | 572 | 672 | 780 | 896 | 1020 | 1152 | 1292 |  |

1. по таблице находим, что минимальное значение –4 достигается при t=2
2. таким образом, ответ: 2.

|  |
| --- |
| **Возможные проблемы**:   * + заполнение таблицы, особенно при большом интервале, очень трудоемко, велика возможность ошибки |

**Решение (способ 2, математический анализ):**

1. повторяя рассуждения пп. 1-5 из предыдущего способа решения, находим, что программа ищет значение t, при котором функция F(t) принимает минимальное значение на интервале от a до b.
2. запишем функцию в виде квадратного трёхчлена:



1. график этой функции – парабола, оси которой направлены вверх, поэтому функция имеет минимум
2. найдем абсциссу точки минимума, которая совпадает с абсциссой точки минимума функции



1. таким образом, ответ: 2.

**Решение (способ 3, математический анализ, свойства параболы):**

1. повторяя рассуждения пп. 1-5 из первого способа решения, находим, что программа ищет значение t, при котором функция F(t) принимает минимальное значение на интервале от a до b.
2. заданная функция имеет корни в точках 
3. график этой функции – парабола, оси которой направлены вверх (коэффициент при  равен 4 > 0), поэтому функция имеет минимум
4. парабола симметрична относительно вертикальной прямой, проходящей через вершину, поэтому абсцисса вершины – это среднее арифметическое корней:



1. таким образом, ответ: 2.

### Ещё пример задания:

*Определите, какое число будет напечатано в результате выполнения следующего алгоритма:*

**Var a,b,t,M,R:integer;**

**Function F(x:integer):integer;**

**begin**

**F:=x\*x + 4\*x + 8;**

**end;**

**BEGIN**

**a:=-10; b:=10;**

**M:=a; R:=F(a);**

**for t:=a to b do begin**

**if (F(t)> R)then begin**

**M:=t;**

**R:=F(t);**

**end;**

**end;**

**write(R);**

**END.**

**Решение:**

1. рассуждая так же, как и в предыдущем примере, можно показать, что программа ищет *наибольшее значение* функции F(t) на интервале от a до b
2. заметим, что выводится не абсцисса, а именно это найденное наибольшее з*начение функции:*

**write(R);**

1. график заданной функции  – это парабола, ветви которой направлены вверх, то есть она имеет точку минимума, но не точку максимума
2. поэтому нужно проверить значения функции на концах отрезка и выбрать из них наибольшее
3. при t=-10 получаем F(t)=68
4. при t=10 получаем F(t)=148
5. таким образом, ответ: 148.

### Еще пример задания (Л.А. Тумарина, г. Электросталь):

*Определите, какое число будет напечатано в результате выполнения следующего алгоритма:*

**Var a,b,t,M,R:integer;**

**Function F(x:integer):integer;**

**begin**

**F:=4\*(x-1)\*(x-3);**

**end;**

**BEGIN**

**a:=-20; b:=0;**

**M:=a; R:=F(a);**

**for t:=a to b do begin**

**if (F(t)<R)then begin**

**M:=t;**

**R:=F(t);**

**end;**

**end;**

**write(M);**

**END.**

**Решение:**

1. рассуждая так же, как и в примере 1, определяем, что программа ищет значение t, при котором функция F(t) принимает *минимальное* значение на интервале от a до b.
2. запишем функцию в виде квадратного трёхчлена:



1. график этой функции – парабола, оси которой направлены вверх, поэтому функция имеет минимум
2. найдем абсциссу точки минимума, которая совпадает с абсциссой точки минимума функции



1. однако это значение не входит в интервал [-20; 0], поэтому нужно проверить значения функции на концах отрезка и выбрать из них наименьшее; ответом будет соответствующее значение t.
2. при t=-20 получаем F(-20)=4\*(-21)\*(-23)=1932
3. при t=0 получаем F(0)= **4\*(-1)\*(-3)=12**, это значение меньше, чем **F(-20)**, поэтому минимум на заданном интервале достигается при t=0
4. таким образом, ответ: 0

# 22 (повышенный уровень, время –7 мин)

**Тема**: динамическое программирование.

**Что нужно знать**:

* динамическое программирование – это способ решения сложных задач путем сведения их к более простым задачам того же типа
* с помощью динамического программирования решаются задачи, которые требуют полного перебор вариантов:
  + «подсчитайте количество вариантов…»
  + «как оптимально распределить…»
  + «найдите оптимальный маршрут…»
* динамическое программирование позволяет ускорить выполнение программы за счет использования дополнительной памяти; полный перебор не требуется, поскольку запоминаются решения всех задач с меньшими значениями параметров

### Пример задания:

*Исполнитель Калькулятор преобразует число, записанное на экране. У исполнителя*

*три команды, которым присвоены номера:*

**1. прибавь 1**

**2. прибавь 2**

**3. прибавь следующее**

*Первая из них увеличивает число на экране на 1, вторая увеличивает это число на 2, а третья прибавляет к числу на экране число, большее на 1 (к числу 3 прибавляется 4, к числу 9 прибавляется 10 и т. д.). Программа для исполнителя Калькулятор– это последовательность команд. Сколько есть программ, которые число 2 преобразуют в число 10?*

**Решение (1 способ, составление таблицы):**

1. заметим, что при выполнении любой из команд число увеличивается (не может уменьшаться)
2. для начального числа 2 количество программ равно 1: существует только одна пустая программа, не содержащая ни одной команды; если через ** обозначить количество разных программ для получения числа *N* из начального числа 2, то **.
3. теперь рассмотрим общий случай, чтобы построить рекуррентную формулу, связывающую ** с предыдущими элементами последовательности **, то есть с решениями таких же задач для меньших N
4. число N могло быть получено одной из трёх операций сложения:

* увеличением на 1 числа N-1;
* увеличением на 2 числа N-2;
* из некоторого числа X увеличением на X+1 (следующее число), так что N = X + X + 1, откуда X = (N – 1) / 2; так могут быть получены только нечетные числа;

поэтому

** для чётных чисел

** для нечётных чисел

1. остается по этой формуле заполнить таблицу для всех значений от 2 до 10:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **N** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|  | **1** | **1** | **2** | **4** | **6** | **11** | **17** | **30** | **47** |

1. ответ – 47.

### Ещё пример задания:

*Исполнитель Май4 преобразует число, записанное на экране. У исполнителя*

*три команды, которым присвоены номера:*

**1. прибавь 1**

**2. прибавь 2**

**3. прибавь 4**

*Первая из них увеличивает число на экране на 1, вторая увеличивает это число на 2, а третья – на 4. Программа для исполнителя Май4 – это последовательность команд. Сколько есть программ, которые число 21 преобразуют в число 30?*

**Решение (1 способ, составление таблицы):**

1. заметим, что при выполнении любой из команд число увеличивается (не может уменьшаться)
2. все числа, меньшие начального числа 21, с помощью этого исполнителя получить нельзя, для них количество программ будет равно 0
3. для начального числа 21 количество программ равно 1: существует только одна пустая программа, не содержащая ни одной команды; если через ** обозначить количество разных программ для получения числа *N* из начального числа 21, то **.
4. теперь рассмотрим общий случай, чтобы построить рекуррентную формулу, связывающую ** с предыдущими элементами последовательности **, то есть с решениями таких же задач для меньших N
5. любое число N > 21 могло быть получено одной из трёх операций сложения соответственно из чисел N-1, N-2 и N-4, поэтому

**

1. остается по этой формуле заполнить таблицу для всех значений от 21 до 30:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **N** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** |
|  | **1** | **1** | **2** | **3** | **6** | **10** | **18** | **31** | **55** | **96** |

1. ответ – 96.

### Ещё пример задания:

*У исполнителя Утроитель две команды, которым присвоены номера:*

**1. прибавь 1**

**2. умножь на 3**

*Первая из них увеличивает число на экране на 1, вторая – утраивает его.*

*Программа для Утроителя – это последовательность команд.*

*Сколько есть программ, которые число 1 преобразуют в число 20?*

**Решение (1 способ, составление таблицы):**

1. заметим, что при выполнении любой из команд число увеличивается (не может уменьшаться)
2. начнем с простых случаев, с которых будем начинать вычисления: для чисел 1 и 2, меньших, чем 3, существует только одна программа, состоящая только из команд сложения; если через ** обозначить количество разных программ для получения числа *N* из 1, то **.
3. теперь рассмотрим общий случай, чтобы построить рекуррентную формулу, связывающую ** с предыдущими элементами последовательности **, то есть с решениями таких же задач для меньших N
4. если число N не делится на 3, то оно могло быть получено только последней операцией сложения, поэтому **
5. если N делится на 3, то последней командой может быть как сложение, так и умножение
6. поэтому для получения ** нужно сложить ** (количество программ с последней командой сложения) и **(количество программ с последней командой умножения). В итоге получаем:

если N не делится на 3: **

если N делится на 3: **

1. остается заполнить таблицу для всех значений от 1 до N:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **N** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** |
|  | **1** | **1** | **2** | **2** | **2** | **3** | **3** | **3** | **5** | **5** | **5** | **7** | **7** | **7** | **9** | **9** | **9** | **12** | **12** | **12** |

1. Заметим, что количество вариантов меняется только в тех столбцах, где N делится на 3, поэтому из всей таблицы можно оставить только эти столбцы:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **N** | **1** | **3** | **6** | **9** | **12** | **15** | **18** | **21** |
|  | **1** | **2** | **3** | **5** | **7** | **9** | **12** | **15** |

1. заданное число 20 попадает в последний интервал (от 18 до 21), поэтому …
2. ответ – 12.

**Решение (2 способ, подстановка – вычисления по формулам «с конца»):**

1. п. 1-6 выполняются так же, как и при первом способе; главная задача – получить рекуррентную формулу:

если N не делится на 3: **

если N делится на 3: **

с начальными условиями **

1. начинаем с заданного конечного числа 20; применяем первую формулу (**), пока не дойдем до числа, делящегося на 3 (это 18):

**

1. далее применяем вторую формулу (**):

**

1. применяем первую формулу для 17:

**

1. применяем вторую формулу для обоих слагаемых:

**

где учтено, что **

1. с помощью первой формулы переходим в правой части к числам, делящимся на 3:

**

а затем применяем вторую формулу для каждого слагаемого

**

1. снова используем первую формулу

**

а затем – вторую:

**

1. и еще раз

**

1. ответ – 12.

**Решение (3 способ, О.В. Шецова, лицей № 6, г. Дубна):**

1. будем составлять таблицу из трех столбцов: в первом записывается получаемое число от 1 до 20, во втором – какой последней командой может быть получено это число, а в третьем вычисляем количество различных программ для получения этого числа из 1
2. очевидно, что число 1 может быть получено с помощью одной единственной (пустой) программы:

|  |  |  |
| --- | --- | --- |
| Число | Как можно получить? | Количество программ |
| **1** |  | **1** |

1. число 2 не делится на 3, поэтому его можно получить только командой сложения (+1), значит, количество программ для 2 совпадает с количеством программ для 1:

|  |  |  |
| --- | --- | --- |
| Число | Как можно получить? | Количество программ |
| **1** |  | **1** |
| **2** | **+1** | **= 1** |

1. число 3 делится на 3, поэтому его можно получить с помощью двух команд: +1 (из 2) и \*3 (из 1):

|  |  |  |
| --- | --- | --- |
| Число | Как можно получить? | Количество программ |
| **1** |  | **1** |
| **2** | **+1** | **1** |
| **3** | **+1 \*3** | **1 + 1 = 2** |

1. числа 4 и 5 не делятся на 3, поэтому их можно получить только с помощью команды +1, а число 6 может быть получено двумя командами:

|  |  |  |
| --- | --- | --- |
| Число | Как можно получить? | Количество программ |
| **1** |  | **1** |
| **2** | **+1** | **1** |
| **3** | **+1 \*3** | **1 + 1 = 2** |
| **4** | **+1** | **2** |
| **5** | **+1** | **2** |
| **6** | **+1 \*3** | **2 + 1 = 3** |

1. следующая группа – 7, 8 (не делятся на 3) и 9 (делится на 3):

|  |  |  |
| --- | --- | --- |
| Число | Как можно получить? | Количество программ |
| **1** |  | **1** |
| **2** | **+1** | **1** |
| **3** | **+1 \*3** | **1 + 1 = 2** |
| **4** | **+1** | **2** |
| **5** | **+1** | **2** |
| **6** | **+1 \*3** | **2 + 1 = 3** |
| **7** | **+1** | **3** |
| **8** | **+1** | **3** |
| **9** | **+1 \*3** | **3 + 2 = 5** |

1. далее – 10, 11 и 12:

|  |  |  |
| --- | --- | --- |
| Число | Как можно получить? | Количество программ |
| **1** |  | **1** |
| **2** | **+1** | **1** |
| **3** | **+1 \*3** | **1 + 1 = 2** |
| **4** | **+1** | **2** |
| **5** | **+1** | **2** |
| **6** | **+1 \*3** | **2 + 1 = 3** |
| **7** | **+1** | **3** |
| **8** | **+1** | **3** |
| **9** | **+1 \*3** | **3 + 2 = 5** |
| **10** | **+1** | **5** |
| **11** | **+1** | **5** |
| **12** | **+1 \*3** | **5 + 2 = 7** |

1. и так далее, вот полностью заполненная таблица (до конечного числа 20):

|  |  |  |
| --- | --- | --- |
| Число | Как можно получить? | Количество программ |
| **1** |  | **1** |
| **2** | **+1** | **1** |
| **3** | **+1 \*3** | **1 + 1 = 2** |
| **4** | **+1** | **2** |
| **5** | **+1** | **2** |
| **6** | **+1 \*3** | **2 + 1 = 3** |
| **7** | **+1** | **3** |
| **8** | **+1** | **3** |
| **9** | **+1 \*3** | **3 + 2 = 5** |
| **10** | **+1** | **5** |
| **11** | **+1** | **5** |
| **12** | **+1 \*3** | **5 + 2 = 7** |
| **13** | **+1** | **7** |
| **14** | **+1** | **7** |
| **15** | **+1 \*3** | **7 + 2 = 9** |
| **16** | **+1** | **9** |
| **17** | **+1** | **9** |
| **18** | **+1 \*3** | **9 + 3 = 12** |
| **19** | **+1** | **12** |
| **20** | **+1** | **12** |

1. ответ – количество программ, с помощью которых можно получить число 20 из 1, – считываем из последней ячейки третьего столбца
2. ответ – 12.

**Решение (4 способ, М.В. Кузнецова и её ученики, г. Новокузнецк):**

1. пусть ** – искомое конечное число, ** количества программ получения числа **
2. тогда для построения рекуррентной формулы определения **, нужно знать 2 факта:
3. какой может быть последняя команда и сколько есть видов этого последнего действия?
4. для каждого «последнего» действия нужно знать число программ получения предыдущего числа, сумма этих количеств и есть искомое значение *–* число программ получения числа **.

Например, общее количество программ получения числа 6 с помощью Утроителя равно **, т.к. есть ДВА способа завершения программ получения этого значения: 6=5+1 и 6=2∙3 .

1. число программ получения числа ** зависит от числа программ получения предыдущего значения, и что программы получения чисел, кратных 3-м могут завершаться 2-мя способами: ** или **, а все остальные числа получают только первым способом: **.
2. составим рекуррентную формулу для определения числа программ получения числа **:

при ** имеем **

если ** не кратно 3: **

если ** делится на 3: **

1. с помощью это формулы заполняем таблицу следующим образом:

– в первом столбце записываем все натуральные числа от 1 до заданного **;

– во втором столбце – числа, на единицу меньшие (из которых может быть получено ** последней операцией сложения с 1);

– в третьем столбце для чисел, кратных 3-м, записываем частное от деления числа, записанного в первом столбце, на 3 (из этого числа может быть получено ** последней операцией умножения на 3);

– в последнем столбце вычисляем *,* складывая соответствующие значения для тех строк, номера которых записаны во втором и третьем столбцах:

|  |  |  |  |
| --- | --- | --- | --- |
| **N** | **N-1** | **N/3** | **K(N)** |
| 1 | – |  | **1** |
| 2 | 1 |  | **1** |
| 3 | 2 | 1 | **1**+**1**= **2** |
| 4 | 3 |  | **2** |
| 5 | 4 |  | **2** |
| 6 | 5 | 2 | **2**+1=**3** |
| 7 | 6 |  | **3** |
| 8 | 7 |  | **3** |
| 9 | 8 | 3 | 3 + 2=**5** |
| 10 | 9 |  | **5** |
| 11 | 10 |  | **5** |
| 12 | 11 | 4 | 5 + 2 = **7** |
| 13 | 12 |  | **7** |
| 14 | 13 |  | **7** |
| 15 | 14 | 5 | 7 + 2 = **9** |
| 16 | 15 |  | **9** |
| 17 | 16 |  | **9** |
| 18 | 17 | 6 | 9+3 = **12** |
| 19 | 18 |  | **12** |
| 20 | 19 |  | **12** |

1. ответ – 12.

**Решение (5 способ, А. Сидоров):**

1. основная идея – число программ, преобразующих начальное число 1 в конечное 20 с помощью заданных в условии команд, равно числу программ, преобразующих конечное число 20 в начальное 1 с помощью обратных команд: «**вычти** **1**» и «**раздели** **на** **3**»
2. будем строить «обратное дерево» – дерево всех способов преобразования **конечного числа в начальное**; это лучше (в сравнении с построением «прямого» дерева, от начального числа к конечному), потому что операция умножения необратима – каждое число можно умножить на 3, но не каждое можно разделить на 3; из-за этого сразу отбрасываются тупиковые ветви, не дающие новых решений
3. рисуем сокращенное дерево, в котором черные стрелки показывают действие первой команды («прибавь 1»), а красные – действие второй команды («умножь на 3»); красные стрелки подходят только к тем числам, которые делятся на 3:

**20←19←18←17←16←15←14←13←12←11←10←9←8←7←6←5←4←3←2←1**

**↑ ↑ ↑ ↑ ↑ ↑**

**6 5 4 3 2 1**

1. чтобы получить количество программ для каждого числа из верхней строки, нужно сложить соответствующие количества программ для всех чисел из нижнего ряда, которые не больше данного (программы с умножением), и добавить 1 (программа, состоящая из одних сложений)
2. очевидно, что для получения 1 существует одна (пустая) программа; тогда для числа 2 тоже получается одна программа, а для числа 3 – две программы:

**(2)(1)**

**3←2←1**

**↑**

**1**

**(1)**

1. далее, для чисел 4 и 5 получаем 2 программы (после числа 3 нет «разветвлений» – подходящих красных стрелок) , а для числа 6 – 3 программы, так как «подошло» еще одно разветвление (6 можно получить умножением 2 на 3), а для числа 2 мы уже подсчитали количество программ – оно равно 1:

**(3) (2)(1)**

**6←5←4←3←2←1**

**↑ ↑**

**2 1**

**(1) (1)**

**1 + 1 + 1 = 3**

1. находить число программ для следующих чисел нам уже не понадобится, потому что при умножении на 3 они дают числа, большие, чем заданное конечное число 20
2. запишем полученные результаты в самой нижней строке для всех множителей от 1 до 6:

**(3) (2)(1)**

**20←19←18←17←16←15←14←13←12←11←10←9←8←7←6←5←4←3←2←1**

**↑ ↑ ↑ ↑ ↑ ↑**

**6 5 4 3 2 1**

**(3) (2) (2) (2) (1) (1)**

1. теперь остается сложить все числа в скобках в нижнем ряду (количество программ с командами умножения) и добавить 1 (одна программа, состоящая только из команд сложения):

3 + 2 + 2 + 2 + 1 + 1 + 1 = 12

1. ответ – 12.

|  |
| --- |
| **Возможные проблемы**:   * + неверно определенные начальные условия   + неверно выведенная рекуррентная формула   + ошибки при заполнении таблицы (невнимательность)   + второй способ (подстановка), как правило, приводит к бОльшему количеству вычислений; конечно, можно отдельно выписывать все полученные ранее значения , но тогда мы фактически придем к табличному методу |

### Еще пример задания:

*У исполнителя Калькулятор две команды, которым присвоены номера:*

**1. прибавь 1**

**2. увеличь вторую с конца цифру на 1**

*Первая из них увеличивает число на экране на 1, вторая – увеличивает на 1 число десятков. Если перед выполнением команды 2 вторая с конца цифра равна 9, она не изменяется. Программа для Калькулятора – это последовательность команд.*

*Сколько есть программ, которые число 15 преобразуют в число 28?*

**Решение (1 способ, составление таблицы):**

1. заметим, что при выполнении любой из команд число увеличивается (не может уменьшаться)
2. при заданных командах очередное число N может быть получено двумя способами:
3. увеличением на 1 (для всех чисел, больших начального числа)
4. увеличением числа десятков на 1 (то есть, фактически командой «+10») – для всех чисел, больших или равных 25; например, число 24 не может быть получено этой командой (14 + 10 = 24), потому что число 14 меньше, чем начальное значение 15
5. таким образом, рекуррентные формулы принимают вид

 для всех чисел, меньших, чем 25

 для чисел, больших или равных 25

1. других способов получения числа с помощью исполнителя с заданными командами нет, то есть мы таким образом рассматриваем все возможные программы
2. начальное значение:  (число 15 можно получить единственной пустой программой)
3. далее заполняем таблицу:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** |
|  | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **2** | **3** | **4** | **5** |

1. Ответ: 5

### Еще пример задания:

*У исполнителя Калькулятор две команды, которым присвоены номера:*

**1. прибавь 1**

**2. увеличь две младшие цифры на 1**

*Первая из них увеличивает число на экране на 1, вторая – увеличивает на 1 число десятков и число единиц. Если перед выполнением команды 2 какая-либо из двух младших цифр равна 9, она не изменяется. Программа для Калькулятора – это последовательность команд.*

*Сколько есть программ, которые число 23 преобразуют в число 48?*

**Решение (1 способ, составление таблицы):**

1. заметим, что при выполнении любой из команд число увеличивается (не может уменьшаться)
2. при заданных командах очередное число N может быть получено двумя способами:
3. увеличением на 1 (для всех чисел, больших начального числа)
4. увеличением обеих цифр на 1 в результате выполнения команды 2 (то есть, фактически командой «+11») – для всех чисел, больших или равных 23 + 11 = 34, которые **НЕ** оканчиваются на 0;
5. увеличением *только младшей* цифры на 1 в результате выполнения команды 2 (то есть, фактически командой «+1») – для всех чисел от 91 до 99, но в нашем диапазоне (23..48) таких нет
6. увеличением *только старшей* цифры на 1 в результате выполнения команды 2 (то есть, фактически командой «+10») – для всех чисел, больших 34 и имеющих 9 на конце; в нашем случае под этот вариант подходит только число 39
7. таким образом, рекуррентные формулы принимают вид

 для всех чисел, меньших, чем 34, а также для всех чисел, оканчивающихся на 0

 для чисел, больших или равных 34, кроме 39

 для числа 39

1. других способов получения числа с помощью исполнителя с заданными командами нет, то есть мы таким образом рассматриваем все возможные программы
2. начальное значение:  (число 23 можно получить единственной пустой программой)
3. далее заполняем таблицу:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **23** | **…** | **33** | **34** | **35** | **36** | **37** | **38** | **39** | **40** | **41** | **42** | **43** | **44** | **45** | **46** | **47** | **48** |
|  | **1** | **…** | **1** | **2** | **3** | **4** | **5** | **6** | **8** | **8** | **9** | **10** | **11** | **12** | **14** | **17** | **21** | **26** |

здесь многоточия означают, что для всех чисел от 23 до 33 включительно количество программ равно 1;

1. например, для числа 47 количество программ вычисляется как

 = 17 + 4 = 21

а для числа 39 –как

 = 6 + 1 + 1 = 8

1. Ответ: 26

# 23 (высокий уровень, время – 10 мин)

**Тема**: Преобразование логических выражений.

**Про обозначения**

К сожалению, обозначения логических операций И, ИЛИ и НЕ, принятые в «серьезной» математической логике (**∧**, **∨**, **¬**), неудобны, интуитивно непонятны и никак не проявляют аналогии с обычной алгеброй. Автор, к своему стыду, до сих пор иногда путает **∧** и **∨**. Поэтому на его уроках операция «НЕ» обозначается чертой сверху, «И» – знаком умножения (поскольку это все же логическое умножение), а «ИЛИ» – знаком «+» (логическое сложение).   
В разных учебниках используют разные обозначения. К счастью, в начале задания ЕГЭ приводится расшифровка закорючек (**∧**, **∨**, **¬**), что еще раз подчеркивает проблему.

**Что нужно знать**:

* условные обозначения логических операций

**¬ A, **  не A (отрицание, инверсия)

**A ∧ B, ** A и B (логическое умножение, конъюнкция)

**A ∨ B, **  A или B (логическое сложение, дизъюнкция)

**A** → **B**  импликация (следование)

**A** ↔ **B, ** эквиваленция (эквивалентность, равносильность)

* таблицы истинности логических операций «И», «ИЛИ», «НЕ», «импликация», «эквиваленция» (см. презентацию «Логика»)
* операцию «импликация» можно выразить через «ИЛИ» и «НЕ»:

**A** → **B = ¬ A ∨ B** или в других обозначениях **A** → **B = **

* операцию «эквиваленция» также можно выразить через «ИЛИ» и «НЕ»:

**A** ↔ **B = ¬ A ∧ ¬ B ∨ A ∧ B** или в других обозначениях **A** ↔ **B = **

* если в выражении нет скобок, сначала выполняются все операции «НЕ», затем – «И», затем – «ИЛИ», потом – «импликация», и самая последняя – «эквиваленция»
* логическое произведение A∙B∙C∙… равно 1 (выражение истинно) только тогда, когда все сомножители равны 1 (а в остальных случаях равно 0)
* логическая сумма A+B+C+… равна 0 (выражение ложно) только тогда, когда все слагаемые равны 0 (а в остальных случаях равна 1)
* правила преобразования логических выражений (законы алгебры логики):

|  |  |  |
| --- | --- | --- |
| Закон | Для **И** | Для **ИЛИ** |
| двойного отрицания |  | |
| исключения третьего |  |  |
| исключения констант | A · 1 = A; A · 0 = 0 | A + 0 = A; A + 1 = 1 |
| повторения | A · A = A | A + A = A |
| поглощения | A · (A + B) = A | A + A · B = A |
| переместительный | A · B = B · A | A + B = B + A |
| сочетательный | A · (B · C) = (A · B) · C | A + (B + C) = (A + B) + C |
| распределительный | A + B · C = (A + B) · (A + C) | A · (B + C) = A · B + A · C |
| де Моргана |  |  |

### Пример задания:

**P-31**.Сколько различных решений имеет система логических уравнений

**(x1 → x2)** ∧ **(x3 → x4 → x5 → x6) = 0**

**(y1 → y2 → y3)** ∧ **(y4 → y5 → y6) = 1**

**(x2 → y2)** ∧ **(x3 → y3) = 1**

где x1, …, x6, y1, …, y6, – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.

**Решение:**

1. заметим, что первые два уравнения независимы друг от друга, их связывает третье уравнение
2. в этих уравнениях есть характерные элементарные уравнения вида

**x1 → x2 → ... → xN = 0 (\*)**

**x1 → x2 → ... → xN = 1 (\*\*)**

1. количество решений уравнения (\*), с нулём в правой части, обозначим через *Z­N*, где *N* – количество переменных; количество решений уравнения (\*\*), с единицей в правой части, обозначим через *K­N*
2. цепочка импликаций выполняется слева направо:

**(x1 → x2 → ... → xN-1) → xN = 0 (\*)**

1. каждая логическая переменная может принимать только 2 значения, 0 и 1, поэтому для *N* переменных получаем 2*N* комбинаций
2. в правой части уравнения может быть только 0 или 1, поэтому если сложить количество решений уравнений (\*) и (\*\*), должно получиться 2*N*, то есть *ZN + KN =* 2*N*
3. импликация равна нулю только в случае **1 → 0**, то есть равенство (\*) выполняется тогда и только тогда, когда *xN =* 0 и справедливо равенство (\*\*) с *N-*1 переменными:

**x1 → x2 → ... → xN-1 = 1**

1. поэтому число решений уравнения (\*) равно *Z­N = KN-*1
2. c другой стороны, *KN-1 =* 2*N-1*– *ZN-1* , так что получаем рекуррентную формулу

*Z­N =* 2*N-1*– *ZN-1*

1. начальное значение для вычислений определим и уравнения **x1 → x2 = 0**, которое имеет одно решение, то есть *Z­*2*=*1
2. далее получаем

*Z­*3*=* 22 – *Z*2 = 4 – 1 = 3

*Z­*4*=* 23 – *Z*3 = 8 – 3 = 5

1. теперь аналогично найдём формулу для вычисления *KN*; мы уже знаем, что *KN =* 2*N*– *ZN* , кроме того, *Z­N = KN-*1, поэтому получаем

*KN =* 2*N*– *KN-1*

1. начальное значение для вычислений определим и уравнения **x1 → x2 = 1**, которое имеет три решения, то есть *K­*2*=*3
2. далее получаем

*K­*3*=* 23 – *K*2 = 8 – 3 = 5

*K*4*=* 24 – *K*3 = 16 – 5 = 11

1. рассмотрим первое уравнение в исходной системе:

**(x1 → x2)** ∧ **(x3 → x4 → x5 → x6) = 0**

здесь 6 переменных, возможно 26 = 64 варианта их комбинаций

из этих 64 вариантов нужно вычесть число решений аналогичного уравнения с единицей в правой части; для этого обе скобки должны быть равны 1

1. уравнения

**x1 → x2 = 1**

**x3 → x4 → x5 → x6 = 1**

имеют соответственно *K*2 *=* 3и *K*4 *=* 11 решений

1. обе скобки включают различные переменные, поэтому общее число решений первого уравнения равно 64 *– K*2 *∙* *K*4 *=* 64 *–* 3 *∙* 11 = 64 *–* 33 = 31
2. аналогично разбираем второе уравнение

**(y1 → y2 → y3)** ∧ **(y4 → y5 → y6) = 1**

здесь в правой части единица, поэтому число решений равно *K*3 *∙* *K*3 *=* 5 *∙* 5 = 25

1. теперь рассмотрим третье уравнение, связывающее первые два: ∙

это уравнение запрещает комбинации (*x*2 = 1, *y*2 = 0) и (*x*3 = 1, *y*3 = 0)

1. подставляем в первые два уравнения (*x*2 = 1, *y*2 = 0):

**(x1 → 1)** ∧ **(x3 → x4 → x5 → x6) = 0** (a)

**(y1 → 0 → y3)** ∧ **(y4 → y5 → y6) = 1** (b)

первая скобка в уравнении (a) всегда равна 1 при любом **x1**, поэтому число решений уравнения (a) равно 2 ∙ *Z*4 = 2 ∙ 5 = 10

1. скобка (**y1 → 0 → y3**) равна 1 кроме случаев, когда **y1 → 0 = 1** и **y3= 0** (только один вариант, **y1= 0** и **y3= 0**), поэтому уравнение **y1 → 0 → y3 = 1** имеет 4 – 1 = 3 решения, а уравнение (b) имеет всего 3 ∙ *K*3 = 3 ∙ 5 = 15
2. таким образом, система уравнений (a-b) имеет всего 10 ∙ 15 = 150 решений
3. теперь подставляем в первые два уравнения (*x*3 = 1, *y*3 = 0):

**(x1 → x2)** ∧ **(1 → x4 → x5 → x6) = 0** (c)

**(y1 → y2 → 0)** ∧ **(y4 → y5 → y6) = 1** (d)

первая скобка в уравнении (a) всегда равна 1 при любом **x1**, поэтому число решений уравнения (a) равно 2 ∙ *Z*4 = 2 ∙ 5 = 10

1. уравнение (c) содержит 5 переменных, то есть может иметь до 25 = 32 решений, из них нужно вычесть решения такого же уравнения с единицей в правой части:

**(x1 → x2)** ∧ **(1 → x4 → x5 → x6) = 1** (e)

1. первая скобка в уравнении (e) равна 1 в трёх случаях, для второй выполним отдельные вычисления; рассмотрим уравнение

**1 → x1 → ... → xN = 1**

количество его решений обозначим через *LN*; аналогично уравнению (\*\*) легко получить рекуррентное выражение *LN* = 2*N*– *LN*-1 с начальным условием *L*1 = 1, поэтому

*L*2*=* 22 – *L*1 = 4 – 1 = 3

*L*3*=* 23 – *L*2 = 8 – 3 = 5

1. таким образом, уравнение (e) имеет всего *K*2 *∙ L*3 *=* 3 ∙ 5 = 15 решений, следовательно, уравнение (с) имеет 32 – 15 = 17 решений
2. рассмотрим уравнение (d):

**(y1 → y2 → 0)** ∧ **(y4 → y5 → y6) = 1** (d)

1. первая скобка равна 1 в одном случае, когда (*y*1 = 1, *y*2 = 0); вторая скобка равна 1 в *K*3 = 5 случаях, поэтому уравнение (d) имеет 5 решений
2. таким образом, система уравнений (c-d) имеет 17 ∙ 5 = 85 решений
3. среди 150 решений системы (a-b) и 85 решений системы (c-d) есть одинаковые, они определяются системой

**(x1 → 1)** ∧ **(1 → x4 → x5 → x6) = 0** (f)

**(y1 → 0 → 0)** ∧ **(y4 → y5 → y6) = 1** (g)

уравнение (f) включает 4 переменных, оно имеет 24 – 2 ∙ *L*3 = 16 – 2 ∙ 5 = 6

в уравнении (g) первая скобка равна 1 в одном случае (*y*1 = 1), число решений уравнения (g) равно *K*3 = 5

поэтому число решений системы (f-g) равно 5∙6 = 30

1. в итоге из 31 ∙ 25 решений первых двух уравнений исходной системы нужно вычесть (150 + 85 – 30) решений, которые отсекаются третьим уравнением, получаем

31 ∙ 25 – (150 + 85 – 30) = 570

1. Ответ: 570.

**Решение (метод отображения для уравнений, И.Н. Филиппов, г. Зеленодольск):**

1. заметим, что первые два уравнения независимы друг от друга, их связывает третье уравнение, поэтому можно поступить так:

* найти решения первого уравнения
* найти решения второго уравнения
* найти множество решений первых двух уравнений
* из множества решений первых двух уравнений выкинуть те, которые не удовлетворяют последнему уравнению

1. найдем решения первого уравнения методом отображения:

**(x1 → x2) ∧ (x3 → x4 → x5 → x6) = 0**

1. расставим порядок действий в левой части уравнения:

**I V II III IV**

**(x1 → x2)** ∧ **(x3 → x4 → x5 → x6) = 0**

1. вспомним таблицу истинности для импликации и составим схему отображения для этого действия:

**x1 x1→x2**

0 0

1 1

2

|  |  |  |
| --- | --- | --- |
| **x1** | **x2** | **x1→x2** |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

1. построим таблицу для определения количества 0 и 1 для I – IV действия:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **x1** | **I** |  |  | **x3** | **II** | **III** | **IV** |
| **0** | 1 | 1 | **0** | 1 | 1 | 3 | 5 |
| **1** | 1 | 3 | **1** | 1 | 3 | 5 | 11 |

1. в V действии перемножаются результаты I и IV действия. Вспомним таблицу истинности для конъюнкции:

|  |  |  |
| --- | --- | --- |
| **I** | **IV** | **V** |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

1. с помощью данной таблицы истинности найдем результат V действия:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **I** | **IV** | **V** |
| **0** | 1 | 5 | 1·5+1·11+3·5=**31** |
| **1** | 3 | 11 | 3·11=33 |

таким образом, первое уравнение имеет 31 решение

1. аналогично решаем второе уравнение системы:

**I II V III IV**

**(y1 → y2 → y3)** ∧ **(y4 → y5 → y6) = 1**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **y1** | **I** | **II** |  |  | **y4** | **III** | **IV** |
| **0** | 1 | 1 | 3 | **0** | 1 | 1 | 3 |
| **1** | 1 | 3 | 5 | **1** | 1 | 3 | 5 |

1. с помощью таблицы истинности конъюнкции найдем результат V действия:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **II** | **IV** | **V** |
| **0** | 3 | 3 | 3·3+3·5+5·3=39 |
| **1** | 5 | 5 | 5·5=**25** |

второе уравнение имеет 25 решений

1. поскольку первые два уравнения независимы друг от друга, система из первых двух уравнений имеет 31·25=**775** решений
2. теперь проверим, какие ограничения накладывает третье уравнение; рассмотрим случай, когда третье уравнение не имеет решения, т.е. (x2 → y2) ∧ (x3 → y3) = 0. Конъюнкция ложна, когда хотя бы один из множителей равен нулю. Рассмотрим отдельно каждый случай
3. первый случай (x2 → y2)=0, т.е. x2=1,y2 =0. При этом x3,y3 – любые. Найдем решение следующей системы уравнений:

**(x1 → x2)** ∧ **(x3 → x4 → x5 → x6) = 0**

**(y1 → y2 → y3)** ∧ **(y4 → y5 → y6) = 1**

**x2 = 1**

**y2 = 0**

1. таблица для первого уравнения: x2=1 накладывает ограничения на I действие (первый множитель всегда будет равен 1), по II – IV действиям ограничений нет (количество 0 и 1 см. выше)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **I** |  |  | **I** | **IV** | **V** |
| **0** | 0 | **0** | 0 | 5 | 0·5+0·11+2·5=**10** |
| **1** | 2 | **1** | 2 | 11 | 2·11=22 |

первое уравнение имеет 10 решений

1. решим второе уравнение при y2 = 0. Чтобы учесть y2 = 0 необходимо для I действия изменить схему отображения (выделенные строки):

|  |  |  |
| --- | --- | --- |
| **y1** | **y2** | **y1→y2**  **y1 y1→y2**  0 0  1 1 |
| 0 | **0** | 1 |
| 0 | 1 | 1 |
| 1 | **0** | 0 |
| 1 | 1 | 1 |

1. для II действия схема прежняя. III и IV действия не зависят от y2, поэтому количество 0 и 1 для них не изменится

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **y1** | **I** | **II** |  |  | **II** | **IV** | **V** |
| **0** | 1 | 1 | 1 | **0** | 1 | 3 | 1·3+1·5+3·3=17 |
| **1** | 1 | 1 | 3 | **1** | 3 | 5 | 3·5=**15** |

второе уравнение имеет 15 решений

1. таким образом, для первого случая система имеет 10·15=**150** решений
2. второй случай (x3 → y3)=0, т.е. x3=1,y3 =0. При этом x2,y2 – любые. Найдем решение следующей системы уравнений:

**(x1 → x2)** ∧ **(x3 → x4 → x5 → x6) = 0**

**(y1 → y2 → y3)** ∧ **(y4 → y5 → y6) = 1**

**x3 = 1**

**y3 = 0**

1. решим первое уравнение (для второго множителя x3=1 учтем сразу в таблице):

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **x3** | **II** | **III** | **IV** |  |  | **I** | **IV** | **V** |
| **0** | 0 | 1 | 1 | 3 | **0** | 1 | 3 | 1·3+1·5+3·3=**17** |
| **1** | 1 | 1 | 3 | 5 | **1** | 3 | 5 | 3·5=15 |

всего 17 решений

1. решим второе уравнение при y3 = 0 (для II действия учитывается как в предыдущем случае):

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **y1** | **I** | **II** |  |  | **II** | **IV** | **V** |
| **0** | 1 | 1 | 3 | **0** | 3 | 3 | 3·3+3·5+1·3=27 |
| **1** | 1 | 3 | 1 | **1** | 1 | 5 | 1·5=**5** |

всего 5 решений

1. таким образом, для второго случая система имеет 17·5=**85** решений
2. теперь необходимо учесть, что первый и второй случай имеют некоторое общее множество решений при (x2 → y2)=0 и (x3 → y3)=0. Найдем решение следующей системы уравнений:

**(x1 → x2)** ∧ **(x3 → x4 → x5 → x6) = 0**

**(y1 → y2 → y3)** ∧ **(y4 → y5 → y6) = 1**

**x2 = 1**

**y2 = 0**

**x3 = 1**

**y3 = 0**

1. для решения первого уравнения возьмем соответствующие найденные значения из первого и второго случая:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **I** | **IV** | **V** |
| **0** | 0 | 3 | 0·3+0·5+2·3=**6** |
| **1** | 2 | 5 | 3·5=15 |

всего 6 решений

1. также решим второе уравнение:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **II** | **IV** | **V** |
| **0** | 1 | 3 | 1·3+1·5+1·3=11 |
| **1** | 1 | 5 | 1·5=**5** |

всего 5 решений

1. первый и второй случай имеют 6·5=**30** общих решений
2. поэтому, исходная система уравнений не имеет решения в 150+85-30=**205** случаях
3. следовательно, система уравнений имеет 775-205=**570** решений
4. Ответ: 570.

### Ещё пример задания:

**P-30**.(Муфаззалов Д.Ф., Уфа, УГАТУ) Сколько различных решений имеет система логических уравнений

**(x1** ∧ **x2)** ∨ **(x1** ∨ **x3) ∧ (x1** ∨ **y1)=0**

**(x2** ∧ **x3)** ∨ **(x2** ∨ **x4) ∧ (x2** ∨ **y2)=1**

**(x3** ∧ **x4)** ∨ **(x3** ∨ **x5) ∧ (x3** ∨ **y3)=0**

**(x4** ∧ **x5)** ∨ **(x4** ∨ **x6) ∧ (x4** ∨ **y4)=1**

**(x5** ∧ **x6)** ∨ **(x5** ∨ **x7) ∧ (x5** ∨ **y5)=0**

**(x6** ∧ **x7)** ∨ **(x6** ∨ **x8) ∧ (x6** ∨ **y6)=1**

где x1, …, x8, y1, …, y6, – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.

**Решение (метод отображения):**

1. уравнения в нечётных строках однотипны, отличаются только сдвигом номеров переменных; уравнения на чётных строках однотипны, отличаются только сдвигом номеров переменных;
2. первое и второе уравнения связаны только через пару , второе и третье – только через пару  и т.д.;
3. таким образом, для того, чтобы определить количество решений первого уравнения, нужно построить *отображение* всех пар  на пары ; это значит, для каждого варианта пары  определить количество различных решений, соответствующих всем возможным парам ;
4. а для того, чтобы определить количество решений системы из двух первых уравнений нужно построить *отображение* всех пар  на пары ; это значит, для каждого варианта пары  определить количество различных решений, соответствующих всем возможным парам ;
5. следует иметь в виду, что отображение, упомянутое в пункте 3, и отображение, упомянутое в пункте 4, будут различными; первое отображение будет применяться при нахождении количества решений из нечётного числа первых уравнений системы, а второе – чётного.
6. методами, описанными ранее, построим отображение переменных в уравнения на нечётных строках:

|  |  |
| --- | --- |
|  |  |
| (0,0) | (0,0) |
| (0,1) | (0,1) |
| (1,0) | (1,0) |
| (1,1) | (1,1) |

и отображение переменных в уравнения на чётных строках:

|  |  |
| --- | --- |
|  |  |
| (0,0) | (0,0) |
| (0,1) | (0,1) |
| (1,0) | (1,0) |
| (1,1) | (1,1) |

Черные стрелки не увеличивают количество решений; красным цветом обозначены стрелки, которые увеличивают количество решений в 2 раза. Увеличение связано с тем, что в каждом уравнении присутствует одна переменная, которая не участвует в отображении, но может дать дополнительное количество решений.

1. Используя эти схемы, заполняем таблицу, начиная со всех значений, равных 1:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Число уравнений | **0** | **1** | **2** | **3** | **4** | **5** | **6** |
| (0,0) | 1 | 2 | 4 | 8 | 24 | 48 | 128 |
| (0,1) | 1 | 1 | 6 | 4 | 32 | 24 | 176 |
| (1,0) | 1 | 2 | 2 | 12 | 12 | 64 | 64 |
| (1,1) | 1 | 1 | 3 | 6 | 16 | 32 | 88 |

складывая значения в последнем столбце, находим, что система из 6 уравнений имеет 128+176+64+88=456 решений.

1. Ответ: 456.

### Ещё пример задания:

**P-29**.Сколько различных решений имеет система логических уравнений

**(x1 → x2) ∧ (x2 → x3) ∧ (x3 → x4) = 1**

**(y1 → y2) ∧ (y2 → y3) ∧ (y3 → y4) = 1**

**(z1 → z2) ∧ (z2 → z3) ∧ (z3 → z4) = 1**

**x1 ∧ y2 ∧ z3 = 0**

где x1, …, x4, y1, …, y4, z1, …, z4, – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.

**Решение (использование свойств битовых цепочек):**

1. перепишем систему с более понятными обозначениями:



1. первые 3 уравнения однотипны; рассмотрим первое из них:



1. рассмотрим решение этого уравнения как битовую цепочку 
2. все импликации должны быть равны 1, в цепочке *X* запрещена комбинация 10, поэтому после первой единицы далее следуют только единицы; вот все 5 решений *X*:

*X* = 0000 0001 0011 0111 1111

1. второе и третье уравнения не зависят от первого и имеют такую же структуру; вот все их решения  и :

*Y* = 0000 0001 0011 0111 1111

*Z* = 0000 0001 0011 0111 1111

1. если бы система состояла бы только из первых трёх уравнений, общее количество решений было бы равно 5 ∙ 5 ∙ 5 = 125
2. теперь рассмотрим последнее уравнение, связывающее *X*, *Y* и *Z*:



1. таким образом, нужно исключить все решения, где 
2. у нас есть одно решение *X* с , два решения *Y* с  и три решения *Z* с ; поэтому из 125 нужно отбросить 1 ∙ 2 ∙ 3 = 6 решений; остаётся 125 – 6 = 119 решений
3. ответ: 119.

### Ещё пример задания:

**P-28**.Сколько различных решений имеет система логических уравнений

**(x1 ∨ y1) ∧ ((x2 ∧ y2) → (x1 ∧ y1)) = 1**

**(x2 ∨ y2) ∧ ((x3 ∧ y3) → (x2 ∧ y2)) = 1**

**...**

**(x5 ∨ y5) ∧ ((x6 ∧ y6) → (x5 ∧ y5)) = 1**

**x6 ∨ y6 = 1**

где x1, …, x7, y1, …, y7, – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.

**Решение (использование свойств битовых цепочек):**

1. перепишем систему с более понятными обозначениями:



1. первые 5 уравнений однотипны, отличаются только сдвигом номеров переменных
2. будем рассматривать каждое решение как пару битовых цепочек (цепочек нулей и единиц)

 и 

1. выполним замену переменных



1. пока «забудем» про первые сомножители  в уравнениях, тогда остается система



которую можно свернуть в одно уравнение:



1. в каждой скобке запрещена комбинация , это значит, что в цепочке  запрещена комбинация 01
2. в свою очередь, это значит, что цепочка имеет структуру «сначала все единицы, потом все нули», вот все возможные цепочки длины 6:

**111111 111000 000000**

**111110 110000**

**111100 100000**

1. теперь нужно перейти к исходным переменным, то есть, к цепочкам  и 
2. пусть ; в исходном уравнении есть ещё ограничение , это та скобка, которую мы «временно забыли», получаем систему



первое уравнение означает, что , второе говорит о том, что, по крайней мере, одно из этих значение равно 1; таких пар две: (0;1) и (1;0)

1. это означает, что каждый ноль в цепочке даёт два решения в исходных переменных
2. теперь исследуем вариант ; добавляя ограничение , получаем систему



первое уравнение означает, что , второе говорит о том, что, по крайней мере, одно из этих значение равно 1; существует только одна такая пара: (1;1)

1. таким образом, каждый ноль в цепочке увеличивает в два раза число решений в исходных переменных, а единицы не меняют количества
2. например, цепочка *Z =* **111111** не содержит нулей и даёт только одно решение
3. цепочка *Z =* **111110** содержит один ноль и даёт 21 = 2 решения
4. цепочка *Z =* **111100** содержит два ноля и даёт 22 = 4 решения
5. общее количество решений системы 20 + 21 + 22 +23 + 24 + 25 +26 = 127.
6. ответ: 127.

### Ещё пример задания:

**P-27**. *Сколько различных решений имеет система логических уравнений*

**(x1 ∨ x2) ∧ (x1 ∧ x2 → x3) ∧ (¬x1 ∨ y1) = 1**

**(x2 ∨ x3) ∧ (x2 ∧ x3 → x4) ∧ (¬x2 ∨ y2) = 1**

**…**

**(x6 ∨ x7) ∧ (x6 ∧ x7 → x8) ∧ (¬x6 ∨ y6) = 1**

**(x7 ∨ x8) ∧ (¬x7 ∨ y7) = 1**

**¬x8 ∨ y8 = 1**

*где x1, …, x8, y1, …, y8, – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение (вариант 1, использование свойств битовых цепочек, М.А. Ройтберг):**

1. перепишем систему с более понятными обозначениями:



1. первые 6 уравнений однотипны, отличаются только сдвигом номеров переменных
2. будем рассматривать каждое решение как пару битовых цепочек (цепочек нулей и единиц)

 и 

1. первые сомножители в первых уравнениях, , означают, что в цепочке  не может быть двух нулей подряд (иначе эта скобка в первых 6 уравнениях и всё произведение равны нулю)
2. вторые скобки, , означают, что если в цепочке  встретились две единицы подряд, то потом будут только единицы, поскольку в противном случае  и все произведение равно 0
3. пока «забудем» про третьи сомножители (); тогда цепочка  в любом решении выглядит так: сначала чередуются нули и единицы, а с некоторого места идут только единицы
4. такая цепочка полностью определяется позицией последнего нуля, т.е. есть всего 9 таких цепочек (позиция последнего нуля от 0 до 8, 0 значит, что нулей нет)

X0 = 11111111

X1 = 01111111

X2 = 10111111

X3 = 01011111

X4 = 10101111

X5 = 01010111

X6 = 10101011

X7 = 01010101

X8 = 10101010

1. третий сомножитель в каждом выражении – это импликация 
2. это означает, что если (,) – решение и , то ; если же , то для есть два возможных значения – 0 и 1
3. поэтому для каждого из указанных выше девяти векторов с количество возможных цепочек  равно , где  –количество нулей в соответствующем векторе 
4. поэтому для  (нет нулей) получаем 20 = 1 решение, для  и  (один нуль) – 21 = 2 решения; для  и  (два нуля) – 22 = 4 решения; для  и  (три нуля) – 23 = 8 решений; для  и  (четыре нуля) – 24 = 16 решений
5. общее количество решений системы 1 + 2\*(21 + 22 +23 + 24) = 1 +2\*30 = 61.
6. ответ: 61.

**Решение (вариант 2, последовательное подключение уравнений, метод отображений):**

1. перепишем систему с более понятными обозначениями:



1. первые 6 уравнений однотипны, отличаются только сдвигом номеров переменных
2. первое и второе уравнения связаны только через пару , второе и третье – только через пару  и т.д.;
3. таким образом, для того, чтобы определить количество решений системы из первых двух уравнений, нам нужно построить *отображение* всех пар  на пары ; это значит, для каждого варианта пары  определить количество различных решений, соответствующих всем возможным парам 
4. покажем это на примере; рассмотрим второе уравнение:



1. пусть , тогда первая скобка равна 0 и решений нет, никаких стрелок в таблице отображения пока нет:

|  |  |
| --- | --- |
|  |  |
| (0,0) | (0,0) |
| (0,1) | (0,1) |
| (1,0) | (1,0) |
| (1,1) | (1,1) |

1. если , получаем уравнение



Которое имеет четыре решения:  и могут быть любыми; при этом пара  отображается на две пары и на две пары :

|  |  |
| --- | --- |
|  |  |
| (0,0) | (0,0) |
| (0,1) | (0,1) |
| (1,0) | (1,0) |
| (1,1) | (1,1) |

1. если , получаем уравнение



Здесь и  может принимать любые значения (0 или 1), а , всего получается 2 решения. Поскольку , одно решение соответствуют паре , а другое – паре :

|  |  |
| --- | --- |
|  |  |
| (0,0) | (0,0) |
| (0,1) | (0,1) |
| (1,0) | (1,0) |
| (1,1) | (1,1) |

2

2

1. наконец, если , получаем уравнение



Здесь обязательно  и , всего одно решение, и оно соответствует паре :

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| *a* | (0,0) | (0,0) |  |
| *b* | (0,1) | (0,1) |  |
| *c* | (1,0) | (1,0) |  |
| *d* | (1,1) | (1,1) |  |

2

2

1. теперь обозначим через  и  число решений, соответствующих связующим парам (0,0), (0,1), (1,0) и (1,1), соответственно
2. как видно из схемы, при подключении нового уравнения эти значения преобразуются по рекуррентным формулам:



1. используя эти формулы, заполняем таблицу, начиная со всех значений, равных 1:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Число уравнений | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| *a* | 1 | 1 | 2 | 2 | 4 | 4 | 8 |
| *b* | 1 | 1 | 2 | 2 | 4 | 4 | 8 |
| *c* | 1 | 2 | 2 | 4 | 4 | 8 | 8 |
| *d* | 1 | 3 | 5 | 9 | 13 | 21 | 29 |

1. итак, мы получили, что система из первых шести уравнений имеет всего 8+8+8+29=53 решения; теперь подключаем пятое равнение, которое связано с первыми четырьмя через пару :



1. заметим, что последнее, 8-е уравнение, связано с предыдущими только через , поэтому, подключая 5-е уравнение, строим отображение пары  на :

при  решений нет

при  два решения (), оба при 

при  одно решение (), отображается на

при  одно решение (), отображается на

1. из таблицы (см. выше) следует, что система из 4-х уравнений имеет

8 решений при , которые не дают решений системы из 7 уравнений;

8 решений при , которые дают по 2 решения (всего 16 при );

8 решений при , которые дают по одному решению (всего 8 при );

29 решений при , которые дают по одному решению (всего 29 при);

1. всего получаем 8 решений при  и 45 решений при ;
2. теперь подключаем 8-е уравнение



1. при  последнее уравнение имеет два решение (), поэтому вся система уравнений имеет 2·8 = 16 решений с 
2. при  это уравнение имеет одно решение (), поэтому вся система уравнений имеет 45 решений с 
3. Ответ: 16 + 45 = 61.

### Ещё пример задания:

**P-26**. *Сколько различных решений имеет система логических уравнений*

**(x1 ∨ x2) ∧ (x1 ∧ x2 → x3) ∧ (x1 ∨ y1) = 1**

**(x2 ∨ x3) ∧ (x2 ∧ x3 → x4) ∧ (x2 ∨ y2) = 1**

**(x3 ∨ x4) ∧ (x3 ∧ x4 → x5) ∧ (x3 ∨ y3) = 1**

**(x4 ∨ x5) ∧ (x4 ∧ x5 → x6) ∧ (x4 ∨ y4) = 1**

**(x5 ∨ x6) ∧ (x5 ∨ y5) = 1**

**x6 ∨ y6 = 1**

*где x1, …, x6, y1, …, y6, – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение (вариант 1, битовые цепочки, М.А. Ройтберг):**

1. перепишем систему с более понятными обозначениями:



1. решением уравнения будут два набора логических переменных,  и , которые можно представить в виде битовых цепочек:

, 

1. первые 4 уравнения однотипны, отличаются только сдвигом номеров переменных; рассмотрим сначала их
2. сомножитель  должен быть равен 1, поэтому в цепочке  не должно быть двух идущих подряд нулей (иначе  и все произведение равно 0)
3. сомножитель  должен быть равен 1, поэтому если в цепочке  появились две единицы подряд, то дальше идут только единицы (иначе  и все произведение равно 0)
4. таким образом, если не учитывать (пока) сомножитель  в каждом из уравнений, существует всего 7 допустимых цепочек , каждая из которых определяется положением последнего нуля (0 – вообще нет нуля):

1. теперь рассмотрим сомножители , которые тоже должны быть равны 1; если , то сразу получаем ; если же , то есть два варианта
2. таким образом, для каждой цепочки  количество соответствующих цепочек  равно , где через  обозначено число единиц в цепочке 
3. в цепочке  есть 6 единиц, в цепочкахи  – по пять, в цепочкахи  – по четыре, в и  – по три
4. поэтому общее число решений вычисляется как 26 +2∙(255555 + 24 + 23) = 64 + 2∙(325555 + 16 + 8) = 176
5. Ответ: 176.

**Решение (вариант 2, последовательное подключение уравнений, метод отображений):**

1. перепишем систему с более понятными обозначениями:



1. первые 4 уравнения однотипны, отличаются только сдвигом номеров переменных
2. первое и второе уравнения связаны только через пару , второе и третье – только через пару  и т.д.;
3. таким образом, для того, чтобы определить количество решений системы из первых двух уравнений, нам нужно построить *отображение* всех пар  на пары ; это значит, для каждого варианта пары  определить количество различных решений, соответствующих всем возможным парам 
4. покажем это на примере; рассмотрим второе уравнение:



1. пусть , тогда первая скобка равна 0 и решений нет, никаких стрелок в таблице отображения пока нет:

|  |  |
| --- | --- |
|  |  |
| (0,0) | (0,0) |
| (0,1) | (0,1) |
| (1,0) | (1,0) |
| (1,1) | (1,1) |

1. если , получаем уравнение



Которое имеет два решения:  или 1, а; при этом пара  отображается на одну пару и на одну пару :

|  |  |
| --- | --- |
|  |  |
| (0,0) | (0,0) |
| (0,1) | (0,1) |
| (1,0) | (1,0) |
| (1,1) | (1,1) |

1. если , получаем уравнение



Здесь и , и  могут принимать любые значения (0 или 1), всего получается 4 решения. Поскольку , два решения соответствуют паре , а другие два – паре :

|  |  |
| --- | --- |
|  |  |
| (0,0) | (0,0) |
| (0,1) | (0,1) |
| (1,0) | (1,0) |
| (1,1) | (1,1) |

2

2

1. наконец, если , получаем уравнение



Здесь обязательно , а  может быть любым (0 или 1), всего 2 решения, и оба они соответствуют паре :

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| *a* | (0,0) | (0,0) |  |
| *b* | (0,1) | (0,1) |  |
| *c* | (1,0) | (1,0) |  |
| *d* | (1,1) | (1,1) |  |

2

2

2

1. теперь обозначим через  и  число решений, соответствующих связующим парам (0,0), (0,1), (1,0) и (1,1), соответственно
2. как видно из схемы, при подключении нового уравнения эти значения преобразуются по рекуррентным формулам:



1. используя эти формулы, заполняем таблицу, начиная со всех значений, равных 1:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Число уравнений | 0 | 1 | 2 | 3 | 4 |
| *a* | 1 | 2 | 2 | 4 | 4 |
| *b* | 1 | 2 | 2 | 4 | 4 |
| *c* | 1 | 1 | 2 | 2 | 4 |
| *d* | 1 | 3 | 8 | 18 | 40 |

1. итак, мы получили, что система из первых четырёх уравнений имеет всего 4+4+4+40=52 решения; теперь подключаем пятое равнение, которое связано с первыми четырьмя через пару :



1. заметим, что последнее, 6-е уравнение, связано с предыдущими только через , поэтому, подключая 5-е уравнение, строим отображение пары  на :

при  решений нет

при  одно решение (), отображается на

при  два решения (), оба при 

при  два решения (), оба при 

1. из таблицы (см. выше) следует, что система из 4-х уравнений имеет 4 решения при , которые не дают решений системы из 5 уравнений;

4 решения при , которые дают по 1 решению (всего 4 при );

4 решения при , которые дают по 2 решения (всего 8 при );

40 решений при , которые дают по 2 решения (всего 80 при);

1. всего получаем 8 решений при  и 84 решения при ;
2. теперь подключаем 6-е уравнение



1. при  это уравнение имеет одно решение (), поэтому вся система уравнений имеет 8 решений с 
2. при  последнее уравнение имеет два решение (), поэтому вся система уравнений имеет 2·84 = 168 решений с 
3. Ответ: 8 + 168 = 176.

### Ещё пример задания:

**P-25**. *Сколько различных решений имеет система логических уравнений*

**(x1 → x2) ∧ (x2 → x3) = 1**

**¬x1 ∧ y1 ∧ z1 ∨ x1 ∧ ¬y1 ∧ z1 ∨ x1 ∧ y1 ∧ ¬z1 = 1**

**¬x2 ∧ y2 ∧ z2 ∨ x2 ∧ ¬y2 ∧ z2 ∨ x2 ∧ y2 ∧ ¬z2 = 1**

**¬x3 ∧ y3 ∧ z3 ∨ x3 ∧ ¬y3 ∧ z3 ∨ x3 ∧ y3 ∧ ¬z3 = 1**

*где x1, …, x3, y1, …, y3, z1, …, z3 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение (последовательное подключение уравнений):**

1. перепишем уравнения с помощью более простых обозначений:



1. заметим, что последние 3 уравнения независимы друга от друга, и вся система связана только через первое уравнение
2. рассмотрим второе уравнение



оно имеет три решения, каждое из которых соответствует единичному значению одного из слагаемых:







1. аналогичные уравнения 3-4 тоже имеют по три решения
2. теперь рассмотрим множество решений системы уравнений 2-3



при ограничении, которое накладывается первым уравнением:



1. поскольку импликация дает ложное значение (0) только для случая 1→0, первое уравнение в исходной системе запрещает комбинацию .
2. рассмотрим решение уравнений 2 и 3:

|  |  |
| --- | --- |
|  |  |
| (0,1,1)  (1,0,1)  (1,1,0) | (0,1,1)  (1,0,1)  (1,1,0) |

Эти уравнения независимы, поэтому система уравнений 2-3 (без дополнительных ограничений) имеет 3⋅3=9 решений

При ограничении :

* в случае  имеем только одно решение системы, когда  в уравнении  2, то есть 
* для двух решений уравнения 3, когда , подходят все 3 отдельных решения уравнения 2

поэтому количество решений системы уравнений 2-3 при ограничении  вычисляется как 1 + 3 + 3 = 7 решений

1. рассуждая аналогично, подключаем уравнение 4 и ограничение , получаем, что количество решений системы уравнений 2-4 при ограничении  вычисляется как 1 + 7 + 7 = 15 решений
2. Ответ: 15.

**Решение (метод отображений, решение А.Н. Носкина):**

1. п. 1-4 совпадают с предыдущим вариантом решения
2. построим правило отображения троек переменных.

**1 уравнение**

**3 уравнение**

**2 уравнение**

**x2y2z2**

**x1y1z1**

**x3y3z3**

110

101

011

011

011

101

101

110

110

Если бы не было никаких ограничений, то данная система имела бы 9 решений.

1. так как система имеет ограничения в виде первого уравнения,

**(x1→x2) ∧ (x2→x3) = 1**

то убираем все связи где выше указанные импликации ложны, а именно:

для **(x1→x2): x1 = 1, x2 = 0,**

**(x2→x3): x2 = 1, x3 = 0.**

**1 уравнение**

**3 уравнение**

**2 уравнение**

**x2y2z2**

**x1y1z1**

**x3y3z3**

110

101

011

011

011

101

101

110

110

1. Заполняем таблицу, вычисляя количество решений при подключении каждого последующего уравнения.

|  |  |  |  |
| --- | --- | --- | --- |
| **xyz** | **1 уравнение** | **2 уравнение** | **3 уравнение** |
| 011 | 1 | 1 | 1 |
| 101 | 1 | 3 | 7 |
| 110 | 1 | 3 | 7 |

1. Складываем все результаты: 1 + 7 + 7 = 15.
2. Ответ: 15.

### Ещё пример задания:

**P-24**. *Сколько различных решений имеет система логических уравнений*

**¬(x1 ≡ x2) ∧ ¬(x1 ≡ x3) ∧ (x2 ≡ x3) = 0**

**¬(x3 ≡ x4) ∧ ¬(x3 ≡ x5) ∧ (x4 ≡ x5) = 0**

**¬(x5 ≡ x6) ∧ ¬(x5 ≡ x7) ∧ (x6 ≡ x7) = 0**

**¬(x7 ≡ x8) ∧ ¬(x7 ≡ x9) ∧ (x8 ≡ x9) = 0**

*где x1, x2, …, x9 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение (последовательное включение уравнений):**

1. заметим два важных момента:
   1. все 4 уравнения – однотипные
   2. первое связано со вторым только через переменную **x3**, второе с третьим – только через **x5**, третье с четвертым – только через **x7**
2. разберем подробно одно первое уравнение; поскольку в нем используется операция И (конъюнкция) и правая часть равна нулю (ложное значение), имеет смысл проверить ситуации, когда первое уравнение истинно: это будет тогда, когда **x2 ≡ x3**, а **x1** не равно этому значению, то есть в двух случаях: (**x1,x2,x3**)=(1,0,0) и (**x1,x2,x3**)=(0,1,1)
3. поскольку логическое уравнение с тремя переменными может иметь не более 8 = 23 решений, вычитаем два решения из этого количества и находим, что первое уравнение имеет 8 – 2 = 6 решений, причем в трёх из них **x3** = 0, а в трёх других **x3** = 1.
4. подключаем второе уравнение: для каждого из трёх решений первого при **x3** = 0 получаем три решения второго, и для каждого из трёх решений первого при **x3** = 1 получаем ещё три решения второго, всего система из двух уравнений имеет 3\*3 + 3\*3 = 18 решений
5. далее продолжаем таблицу:

|  |  |
| --- | --- |
| число уравнений | решений |
| 1 | 3(при **x3**= 0) + 3(при **x3**= 1) = 6 |
| 2 | 3\*3 + 3\*3 = 9(при **x5**= 0) + 9(при **x5**= 1) = 18 |
| 3 | 9\*3 + 9\*3 = 27(при **x7**= 0) + 27(при **x7**=1) = 54 |
| 4 | 27\*3 + 27\*3 = 81 + 81 = 162 |

1. Ответ: 162

**Решение (метод отображений[[20]](#footnote-20), решение А.Н. Носкина):**

1. сначала построим таблицу, в которой переберем все варианты **x1,** **x2, x3**, поскольку в первом логическом уравнении три переменных, то таблица будет иметь 8 строк (8 = 23);

уберем из таблицы (желтая заливка) такие значения *x3*, при которых первое уравнение не имеет решения.

|  |  |  |
| --- | --- | --- |
| **x1** | **x2** | **x3** |
| 0 | 0 | 0 |
| 1 |
| 1 | 0 |
| **1** |
| 1 | 0 | **0** |
| 1 |
| 1 | 0 |
| 1 |

1. анализируя таблицу, строим правило отображения пар переменных

(например паре **x1x2=00 соответствуют** пары **x2x3= 00 и 01)**.

**x1x2 x2x3**

00

00

01

01

10

10

11

11

1. теперь построим таблицу, в которой переберем все варианты **x3,** **x4, x5**, поскольку во втором логическом уравнении три переменных, то таблица будет иметь 8 строк (8 = 23);

уберем из таблицы (желтая заливка) такие значения *x5*, при которых второе уравнение не имеет решения.

|  |  |  |
| --- | --- | --- |
| **X3** | **X4** | **X5** |
| 0 | 0 | 0 |
| 1 |
| 1 | 0 |
| **1** |
| 1 | 0 | **0** |
| 1 |
| 1 | 0 |
| 1 |

1. анализируя таблицу, строим правило отображения пар переменных, связывая с переменными первого логического уравнения

**x1x2 x2x3 x4x5**

00

00

01

01

10

10

11

11

00

01

10

11

1. Заполняем таблицу, вычисляя количество пар переменных, при котором система имеет решение.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **x1x2** | **x2x3** | **x4x5** | **x6x7** | **x8x9** |
| 00 | 1 | 1 | 3 | 9 | 27 |
| 01 | 1 | 2 | 6 | 18 | 54 |
| 10 | 1 | 2 | 6 | 18 | 54 |
| 11 | 1 | 1 | 3 | 9 | 27 |

1. Складываем все результаты: 27 + 54 + 54 + 27 = 162
2. Ответ: 162.

**Решение (метод отображений, решение Ел.А. Мирончик):**

1. для построения отображения построим таблицу решения первого уравнения:

|  |  |  |
| --- | --- | --- |
| **x1** | **x2** | **x3** |
| 0 | 0 | 0 |
| 1 |
| 1 | 0 |
| 1 | 0 | 1 |
| 1 | 0 |
| 1 |

1. уравнения связаны только через одну переменную, значит множество значений переменной **x1** приведет к множеству значений переменной **x3** это отображение повторится на переходе от **x3** к **x5**,далее к **x7** и**x9**
2. построим отображение:

0

1

0

1

1. в таблицу необходимо включить переменные **x1,x3**,**x5**, **x7** и**x9;н**а старте имеем одно значение **x1**=0 и одно значение **x1**=1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **X1** | **X3** | **X5** | **X7** | **X9** |
| 0 | 1 | 3 | 9 | 27 | 81 |
| 1 | 1 | 3 | 9 | 27 | 81 |

1. складываем все результаты: 81 + 81 = 162
2. ответ: 162.

### Ещё пример задания:

**P-23**. *Сколько различных решений имеет система логических уравнений*

**(x1 → x2) → (x3 → x4) = 1**

**(x3 → x4) → (x5 → x6) = 1**

*где x1, x2, …, x6 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение (метод замены переменных):**

1. используем замену переменных (заметим, что каждая из новых переменных независима от других, это важно!):

**Y1 = x1 → x2, Y2 = x3 → x4, Y3 = x5 → x6**

тогда система запишется в виде

**Y1 → Y­2 = 1**

**Y2 → Y­3 = 1**

1. можно объединить эти уравнения в одно

**(Y1 → Y­2) ∧ (Y2 → Y­3) = 1**

для того, чтобы это равенство было выполнено, ни одно из импликаций не должна быть ложной, то есть в битовой цепочке, составленной из значений переменных **Y1**, **Y­2**, **Y­3**, не должно быть последовательности «10»; вот все возможные варианты:

|  |  |  |
| --- | --- | --- |
| **Y1** | **Y2** | **Y3** |
| 0 | 0 | 0 |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

1. теперь вернемся к исходным переменным; импликация **x1 → x2** дает 0 при одном наборе исходных переменных (**x1,x2**) = (1,0) и 1 при трёх наборах (**x1,x2**) = {(0,0), (0,1), (1,1)}
2. учитывая, что каждая из новых переменных **Y1**, **Y­2**, **Y­3**, независима от других; для каждой строки полученной таблицы просто перемножаем количество вариантов комбинация исходных переменных:

|  |  |  |  |
| --- | --- | --- | --- |
| **Y1** | **Y2** | **Y3** | **вариантов** |
| 0 | 0 | 0 | 1\*1\*1=1 |
| 0 | 0 | 1 | 1\*1\*3=3 |
| 0 | 1 | 1 | 1\*3\*3=9 |
| 1 | 1 | 1 | 3\*3\*3=27 |

1. складываем все результаты: 1 + 3 + 9 + 27 = 40
2. Ответ: 40.

**Решение (метод отображений, решение А.Н. Носкина):**

1. сначала построим таблицу, в которой переберем все варианты **x1,** **x2, x3,x4**, поскольку в первом логическом уравнении четыре переменных, то таблица будет иметь 16 строк (16=24); уберем из таблицы (желтая заливка) такие значения **x4**, при которых первое уравнение не имеет решения.

|  |  |  |  |
| --- | --- | --- | --- |
| **x1** | **X2** | **X3** | **X4** |
| 0 | 0 | 0 | 0 |
| 1 |
| 1 | **0** |
| 1 |
| 1 | 0 | 0 |
| 1 |
| 1 | **0** |
| 1 |
| 1 | 0 | 0 | 0 |
| 1 |
| 1 | 0 |
| 1 |
| 1 | 0 | 0 |
| 1 |
| 1 | **0** |
| 1 |

1. анализируя таблицу, строим правило отображения пар переменных

(например паре **x1x2=00 соответствуют** пары **x3x4 = 00,01 и 11)**.

**x1x2 x3x4**

00

00

01

01

10

10

11

11

1. заполняем таблицу, вычисляя количество пар переменных, при котором система имеет решение.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **x1x2** | **x3x4** | **x5x6** |
| 00 | 1 | 4 | 13 |
| 01 | 1 | 4 | 13 |
| 10 | 1 | 1 | 1 |
| 11 | 1 | 4 | 13 |

1. складываем все результаты: 13 + 13 + 1 + 13 = 40
2. Ответ: 40.

### Ещё пример задания:

**P-22**. *Сколько различных решений имеет система логических уравнений*

**(x1 → x2) ∧ (x2 → x3) ∧ (x3 → x4)= 1**

**(у1 → у2) ∧ (у2 → у3) ∧ (у3 → у4) = 1**

**(¬y1 ∨ x1) ∧ (¬y2 ∨ x2) ∧ (¬y3 ∨ x3) ∧ (¬y4 ∨ x4) = 1**

*где x1, x2, …, x4* и *y1, y2, …, y4 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение:**

1. видим, что первые два уравнения независимы друг от друга (в первое входят только *x1, x2, …, x4*, а во второе – только *y1, y2, …, y4*)
2. третье уравнение связывает первые два, поэтому можно поступить так:

* найти решения первого уравнения
* найти решения второго уравнения
* найти множество решений первых двух уравнений
* из множества решений первых двух уравнений выкинуть те, которые не удовлетворяют последнему уравнению

1. найдем решения первого уравнения; каждая из логических переменных *x1, x2, …, x4* может принимать только два значения: «ложь» (0) и «истина» (1), поэтому решение первого уравнения можно записать как битовую цепочку длиной 4 бита: например, 0011 означает, что   
   **x1**=**x2***=*0 и **x3**=**x4***=*1
2. вспомним, что импликация **x1→x2** ложна только для ***x1*** **=** **1***и* ***x2*** **=** **0**, поэтому битовая цепочка, представляющая собой решение первого уравнения, не должна содержать сочетания «10»; это дает такие решения (других нет!):

**(x1***,* **x2***,* **x3***,* **x4) =**0000 0001 0011 0111 1111

1. видим, что второе уравнение полностью совпадает по форме с первым, поэтому все его решения:

**(y1***,* **y2***,* **y3***,* **y4) =**0000 0001 0011 0111 1111

1. поскольку первые два уравнения независимы друг от друга, система из первых двух уравнений имеет 5·5=25 решений: каждому решению первого соответствует 5 разных комбинаций переменных *y1, y2, …, y4*, которые решают второе, и наоборот, каждому решению второго соответствует 5 разных комбинаций переменных *x1, x2, …, x4*, которые решают первое:

**(y1***,* **y2***,* **y3***,* **y4) =**0000 0001 0011 0111 1111

**(x1***,* **x2***,* **x3***,* **x4) =**0000 0000 0000 0000 0000

0001 0001 0001 0001 0001

0011 0011 0011 0011 0011

0111 0111 0111 0111 0111

1111 1111 1111 1111 1111

1. теперь проверим, какие ограничения накладывает третье уравнение; вспомнив формулу, которая представляет импликацию через операции «НЕ» и «ИЛИ» (), можно переписать третье уравнение в виде

**(y1 → x1) ∧ (y2 → x2) ∧ (y3 → x3) ∧ (y4 → x4) = 1**

1. импликация **y1→x1** ложна только для ***y1*** **=** **1***и* ***x1*** **=** **0**, следовательно, такая комбинация запрещена, потому что нарушает третье уравнение; таким образом, набору с ***y1*** **=** **1**:

**(y1***,* **y2***,* **y3***,* **y4) =**1111

соответствует, с учетом третьего уравнения, только одно решение первого, в котором ***x1*** **=** **1**

**(x1***,* **x2***,* **x3***,* **x4) =**1111

поэтому множество решений «редеет»:

**(y1***,* **y2***,* **y3***,* **y4) =**0000 0001 0011 0111 1111

**(x1***,* **x2***,* **x3***,* **x4) =**0000 0000 0000 0000

0001 0001 0001 0001

0011 0011 0011 0011

0111 0111 0111 0111

1111 1111 1111 1111 1111

1. аналогично двигаемся дальше по третьему уравнению; второй сомножитель равен 0, если импликация **y2→x2** ложна, то есть только для ***y2*** **=** **1***и* ***x2*** **=** **0**, это «прореживает» предпоследний столбец:

**(y1***,* **y2***,* **y3***,* **y4) =**0000 0001 0011 0111 1111

**(x1***,* **x2***,* **x3***,* **x4) =**0000 0000 0000

0001 0001 0001

0011 0011 0011

0111 0111 0111 0111

1111 1111 1111 1111 1111

1. аналогично проверяем еще два ограничения, отбрасывая все решения, для которых ***y3*** **=** **1***и* ***x3*** **=** **0**, а также все решения, для которых ***y4*** **=** **1***и* ***x4*** **=** **0**:

**(y1***,* **y2***,* **y3***,* **y4) =**0000 0001 0011 0111 1111

**(x1***,* **x2***,* **x3***,* **x4) =**0000

0001 0001

0011 0011 0011

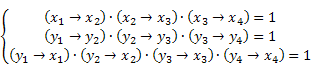
0111 0111 0111 0111

1111 1111 1111 1111 1111

1. итак, остается одно решение при **(y1***,* **y2***,* **y3***,* **y4)=1111**, два решения при **(y1***,* **y2***,* **y3***,* **y4)=0111**, три решения при**(y1***,* **y2***,* **y3***,* **y4)=0011**, четыре решения при**(y1***,* **y2***,* **y3***,* **y4)=0001** и 5 решений при **(y1***,* **y2***,* **y3***,* **y4)=0000**
2. всего решений 1+2+3+4+5=15.

**Решение (метод отображений, решение Е.А. Мирончик):**

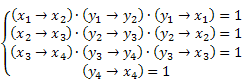
1. перепишем систему в виде



1. выражения в левой части первого и второго уравнения равны 1, следовательно, и произведение левых частей равно 1:



1. запишем систему, равносильную исходной системе:



1. Первые три уравнения системы зависит от четырех переменных. Индексы соседних уравнений отличаются на 1. Общими переменными для соседних уравнений является пара переменных. Зная количество пар по первому уравнению определим количество пар и т.д.



1. построим дерево решений первого уравнения:

|  |  |  |  |
| --- | --- | --- | --- |
| *x1* | *y1* | *x2* | *y2* |
| 0 | 0 | 0 | 0 |
| 1 |
| 1 | 0 |
| 1 |
| 1 | Х |  |
| 1 | 0 | 1 | 0 |
| 1 |
| 1 | 1 | 1 |

1. по дереву решений строим отображение, записывая его в любом виде

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| а) с помощью двудольного графа   |  |  | | --- | --- | | ***00*** | ***00*** | | ***01*** | ***01*** | | ***10*** | ***10*** | | ***11*** | ***11*** | | б) с помощью матрицы смежности:   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | | ***Приемник*** | | | | | ***00*** | ***01*** | ***10*** | ***11*** | | ***Источник*** | ***00*** | *+* | *+* | *+* | *+* | | ***01*** |  |  |  |  | | ***10*** |  |  | *+* | *+* | | ***11*** |  |  |  | *+* | |

1. Выполняем вычисления по первым трём уравнениям. Последнему уравнению, не удовлетворяет пара (0,1). Значит, в последнем столбце следует убрать число, соответствующее паре (01).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *Количество решений* | | | |
| *x1, y1* | *x2, y2* | *x3, y3* | *x4, y4* |
| 00 | 1 | 1 | 1 | 1 |
| 01 | 1 | 1 | 1 | ~~1~~ |
| 10 | 1 | 2 | 3 | 4 |
| 11 | 1 | 3 | 6 | 10 |

1. всего решений 1+4+10=15.

### Ещё пример задания:

**P-21**. *Сколько различных решений имеет система логических уравнений*

**X1**→**X2** **∨****X3** **∧** **¬X4** **= 1**

**X3**→**X4** **∨****X5** **∧** **¬X6** **= 1**

**X5**→**X6** **∨****X1** **∧** **¬X2** **= 1**

*где x1, x2, …, x6 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение:**

1. перепишем уравнения в более простом виде, заменим знаки **∨** и **∧** соответственно на (логические) сложение и умножение:



1. вспомним, что сначала выполняется логическое умножение, потом логические сложение и только потом – импликация, поэтому уравнения можно переписать в виде



1. раскрывая импликацию по формуле , получаем



1. далее замечаем, что , и , поэтому можно ввести новые переменные ,  и , и переписать уравнения в виде



1. пусть , тогда из первого уравнения сразу имеем  и далее из второго ; при этом третье автоматически выполняется; получили одно решение
2. теперь пуст , тогда из последнего уравнения имеем , а из второго – , при этом первое уравнение справедливо
3. таким образом, система уравнений относительно переменных имеет два решения: (0,0,0) и (1,1,1)
4. теперь вернемся обратно к исходным переменным; значению  соответствует единственный вариант ; значению  соответствуют остальные 3 пары возможных значений 
5. то же самое можно сказать про  и : нулевое значение дает один набор соответствующих исходных переменных, а единичное – три
6. переменные ,  и  независимы друг от друга, так как каждая из них составлена из разных X-переменных, поэтому Y-решение (0,0,0) (см. п. 7) дает только одно X-решение, а Y-решение (1,1,1) – 3·3·3=27 решений
7. всего решений 1 + 27 = 28.

**Решение (метод отображений[[21]](#footnote-21), решение А.Н. Носкина):**

1. сначала построим таблицу, в которой переберем все варианты **x1,** **x2, x3,x4**, поскольку в первом логическом уравнении четыре переменных, то таблица будет иметь 16 строк (16=24); уберем из таблицы (желтая заливка) такие значения **x3**, **x4**, при которых первое уравнение не имеет решения.

|  |  |  |  |
| --- | --- | --- | --- |
| **x1** | **X2** | **X3** | **X4** |
| 0 | 0 | 0 | 0 |
| 1 |
| 1 | 0 |
| 1 |
| 1 | 0 | 0 |
| 1 |
| 1 | 0 |
| 1 |
| 1 | 0 | **0** | **0** |
| **1** |
| 1 | 0 |
| **1** |
| 1 | 0 | 0 |
| 1 |
| 1 | 0 |
| 1 |

1. Анализируя таблицу, строим правило отображения пар переменных

(например, паре **x1x2=10 соответствует только** пара **x3x4 = 10)**.

**Внимание!** Для пары **x1x2 = 10** нет связей **x3x4 = 00,10 и 11**

**x1x2 x3x4**

00

00

01

01

10

10

11

11

1. теперь рассмотрим, как влияет на правило отображения третье уравнение. Для этого построим таблицу, в которой переберем все варианты **x1,** **x2, x5,x6**.

Уберем из таблицы (желтая заливка) такие значения **x6**, при которых третье уравнение не имеет решения.

|  |  |  |  |
| --- | --- | --- | --- |
| **x1** | **X2** | **X5** | **X6** |
| 0 | 0 | 0 | 0 |
| 1 |
| 1 | **0** |
| 1 |
| 1 | 0 | 0 |
| 1 |
| 1 | **0** |
| 1 |
| 1 | 0 | 0 | 0 |
| 1 |
| 1 | 0 |
| 1 |
| 1 | 0 | 0 |
| 1 |
| 1 | **0** |
| 1 |

Анализ таблицы показывает, что еще исключаются 3 связи, а именно для пары

**x1x2** **=** **00 нет связей с x5x6** **=** **10**

**x1x2** **=** **01 нет связей с x5x6** **=** **10**

**x1x2** **=** **10 нет связей с x5x6** **=** **10**

1. На основе выше сказанного уточним ранее приведенное правило отображения пар переменных, исключив три лишних связи.

**x1x2 x3x4**

11

11

10

01

10

01

00

00

1. заполняем таблицу, вычисляя количество пар переменных, при котором система имеет решение:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **x1x2** | **x3x4** | **x5x6** |
| 00 | 1 | 3 | 9 |
| 01 | 1 | 3 | 9 |
| 10 | 1 | 1 | 1 |
| 11 | 1 | 3 | 9 |

1. складываем все результаты: 9 + 9 + 1 + 9 = 28.
2. Ответ: 28.

### Ещё пример задания:

**P-20**. *Сколько различных решений имеет логическое уравнение*

**X1**→**X2** → **X3** → **X4** → **X5** → **X6 = 1**

*где x1, x2, …, x6 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение (вариант 1, табличный метод, динамическое программирование):**

1. в левой части заданного уравнения стоят последовательно несколько операций импликации, скобок нет, поэтому порядок выполнения операций определяется приоритетом этих операций; в данном случае все операции имеют одинаковый приоритет
2. операции, имеющие одинаковый приоритет, выполняются слева направо, то есть первой выполняется импликация **X1**→**X2**, а последней – последняя импликация

**((((X1**→**X2)** → **X3)** → **X4)** → **X5)** → **X6**

1. каждая логическая переменная может принимать значение «истина» (1) или «ложь» (0)
2. для набора из 6 независимых логических переменных существует 26 =64 разных комбинаций значений этих переменных
3. рассмотрим первую импликацию, **X1**→**X2**; она дает в трёх случаях 1, и в одном – 0:

|  |  |  |
| --- | --- | --- |
| **X1** | **X2** | **X1**→**X2** |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

1. посмотрим, как меняется количество решений, если «подключить» следующую переменную;
   * если **X1=0**, то **X1**→**X2** **=1** (из нулей получаются  единиц)
   * если **X1=1**, то **X1**→**X2** **=0** при **X2** **=0** и **X1**→**X2** **=1** при **X2** **=1** (из  единиц получаются  нулей и  единиц)
2. исходя из этого, можно составить формулы для вычисления количества нулей  и количества единиц  для уравнения с  переменными:

, 

1. для одной переменной имеем 1 ноль и 1 единицу, поэтому начальные условия для расчёта:



1. составим таблицу, которую будем заполнять слева направо, вычисляя число нулей и единиц по приведенным выше формулам; в таблице показано, как строится следующий столбец таблицы для :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **число переменных** | **1** | **2** | **3** | **4** | **5** | **6** |
| нулей | **1** | **1** | **3** | **5** | **11** | **21** |
| единиц | **1** | **3** | **5** | **2\*3+5=11** | **21** | **43** |

1. таким образом, ответ: 43 решения.

**Решение (вариант 2, «с хвоста»):**

1. те же рассуждения, что и в п. 1-4 решения по варианту 1
2. если **X6** **=1**, то левая часть уравнения равна 1, то есть равенство выполняется; комбинаций с **X6** **=1** ровно половина от общего количества, то есть 32
3. теперь проверяем варианты с **X6** **=0**; сразу получаем, что для выполнения заданного уравнения нужно, чтобы **(X1**→**X2** → **X3** → **X4** → **X5)=0**; иначе получим 1 → **X6** = 1 →0 = 0
4. проверим отдельно случаи **X5****=0** и **X5****=1**
5. пусть **X6** **=** **0** и **X5** **=1**; в этом случае никогда не будет выполнено условие   
   **(X1**→**X2** → **X3** → **X4** → **X5)=0**, решений нет
6. пусть **X6** **=** **X5** **=0**; в этом случае условие **(X1**→**X2** → **X3** → **X4** → **X5)=0** выполняется только при **(X1**→**X2** → **X3** → **X4)=1**; если **X4****=1**, это условие всегда верно, поэтому получаем еще 8 решений – 8 комбинаций, где **X6** **=** **X5** **=0** и **X4****=1** (1/8 всех комбинаций)
7. теперь рассмотрим случаи, когда **X6** **=** **X5** **=** **X4****=0**; рассуждая аналогично, находим, что условие **(X1**→**X2** → **X3** → **0)=1** верно при **(X1**→**X2** → **X3)=0**, это сразу дает **X3** **=0** и   
   **(X1**→**X2)=1**
8. при всех известных значениях остальных переменных (**X6** **=** **X5** **=** **X4****=****X3****=0**) условие   
   **(X1**→**X2)=1** истинно в трёх случаях: (**X1,X­2**) =(0,0) , (0,1) и (1,1), это дает еще 3 решения
9. таким образом, ответ: 32 + 8 + 3 = 43 решения.

**Решение (вариант 3, ещё раз с хвоста):**

1. те же рассуждения, что и в п. 1-4 решения по варианту 1
2. число решений такого уравнения с n неизвестными обозначим через **Kn**. Число решений аналогичного уравнения с нулем в правой части обозначим через **Zn**. Тогда **K­6** **=** **26** **–** **Z6**.
3. теперь найдем **Z6**: необходимо, чтобы **X6** **=** **0** и **X1**→**X2** → **X3** → **X4** → **X5** **=1**, то есть   
   **Z6** **=** **K5** **=** **25** **–** **Z5**.
4. аналогично для любого **n** **>** **1** получаем формулу: **Zn** **=** **Kn****-****1** **=** **2n-1** **-** **Zn-1**
5. конечная точка рекурсии, очевидно, **Z1** **=** **1**.
6. откатываясь назад, находим, что

**Z2** **=** **K1** **=** **21** **–** **Z1** **=** **2** **–** **1** **=** **1**

**Z3** **=** **K2** **=** **22** **–** **Z2** **=** **4** **–** **1** **=** **3**

**Z4** **=** **K3** **=** **23** **–** **Z3** **=** **8** **–** 3 **=** 5

**Z5** **=** **K4** **=** **24** **–** **Z4** **=** 16 **–** 5 **=** 11

**Z6** **=** **K5** **=** **25** **–** **Z5** **=** 32 **–** **11** **=** 21

**Z7** **=** **K6** **=** **26** **–** **Z6** **=** 64 **–** 2**1** **=** 4**3**

1. таким образом, ответ: 43 решения.

**Решение (вариант 4, приведение к базису «И-ИЛИ-НЕ», Е.Н. Смирнова):**

1. те же рассуждения, что и в п. 1-4 решения по варианту 1
2. заменяем импликацию по формуле ; на первом шаге получаем



1. далее по той же формуле



инверсию в первом слагаемом раскроем по закону де Моргана ():



1. сделав те же операции с оставшейся скобкой, получаем



1. и, применяя ту же формулу еще раз, получим уравнение



1. при  остальные 5 переменных можно выбирать любым способом, это дает 25= 32 решения4444444
2. при  и  решений нет
3. при  получаем 23 = 8 решений при  (можно выбирать ,  и  произвольно)
4. при  сразу находим, что , это дает еще 3 решения, при которых истинно выражение 
5. таким образом, ответ: 32 + 8 + 3 = 43 решения.

**Решение (вариант 5, в общем виде, Е.В. Хламов):**

1. очевидно, что при X6=1выражение истинно всегда (при любых значениях остальных переменных); это составляет ровно половину от полного множества решений, а именно 2n-1 решений.
2. при X6=0 решениями будут только те строки, которые удовлетворяют уравнению X1→X2 → X3 → X4 → X5=0, а этому уравнению удовлетворяют в свою очередь все решения

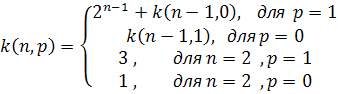
X1→X2 → X3 → X4=1 и X5=0

1. очевидно также что уравнениеX1→X2=1 имеет три решения, а уравнение X1→X2=0 – одно решение
2. рассмотрим аналогичное уравнение с *n* переменными:

**X1**→**X2** → **X3** → **X4** → **…** → **Xn = p (\*)**

обозначим буквой *n* количество переменных и буквой *p*-правую часть уравнения \*.

1. тогда можно выписать рекуррентную формулу для количества решений:



1. Выполним вычисления для *n =* 6 *и p =* 1

*k*(6,1) = 32 + *k*(5,0) = 32 + *k*(4,1) = 32 + 8 + *k*(3,0) = 32 + 8 + *k*(2,1) = 32 + 8 + 3 = 43

1. Ответ: 43

### Ещё пример задания:

**P-19**. *Сколько различных решений имеет система уравнений*

**X1** **∨ X2** **∧** **X3 = 1**

**X2** **∨ X3** **∧** **X4 = 1**

**...**

**X8** **∨ X9** **∧** **X10 = 1**

*где x1, x2, …, x10 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение (последовательное подключение уравнений):**

1. рассмотрим сначала все решения первого уравнения; его левая части истинна, когда **X1=1** (при этом **X2** и **X3** могут быть любыми), а также когда **X1=0** и **X2=X3**=1:

|  |  |  |
| --- | --- | --- |
| **X1** | **X2** | **X3** |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |
| 1 | 1 | 1 |

1. заметим, что первое и второе уравнения связаны через последние две переменных, в данном случае это **X2** и **X3**
2. пусть **i** – число переменных в уравнениях; введем обозначения:

**Ki**  – количество решений, в которых последние две переменные принимают   
значения (0,0)

**Li**  – количество решений, в которых последние две переменные принимают   
значения (0,1)

**Mi**  – количество решений, в которых последние две переменные принимают   
значения (1,0)

**Ni**  – количество решений, в которых последние две переменные принимают   
значения (1,1)

1. из таблицы видим, что **K3=1**, **L3=1**, **M3=1** и **N3=2**
2. теперь подключаем второе уравнение; посмотрим, к чему приводят разные комбинации последних двух переменных:

|  |  |  |  |
| --- | --- | --- | --- |
| **X1** | **X2** | **X3** | **X4** |
| 0 | 1 | 1 | 0 |
| 1 |
| 1 | 0 | 0 | **×** |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 |
| 1 | 1 | 1 | 0 |
| 1 |

1. находим, что   
   комбинация (0,0) не дает ни одного решения,   
   комбинация (0,1) дает одно решение, и при этом (**X3,X4**)=(1,1)

комбинация (1,0) дает два решения, причем (**X3,X4**)=(0,0) или (0,1)

комбинация (1,1) дает два решения, причем (**X3,X4**)=(1,0) или (1,1)

1. из предыдущего пункта делаем вывод, что

**Ki+1 = Mi** (комбинация (0,0) появилась из (1,0) на предыдущем шаге)

**Li+1 = Mi** (комбинация (0,1) появилась из (1,0) на предыдущем шаге)

**Mi+1 = Ni** (комбинация (1,0) появилась из (1,1) на предыдущем шаге)

**Ni+1 = Li+Ni** (комбинация (1,1) появляется из (0,1) и (1,1))

1. используя эти рекуррентные формулы, заполняем таблицу для **i=4,…,10**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **i** | **Ki** | **Li** | **Mi** | **Ni** | **Всего** |
| 3 | 1 | 1 | 1 | 2 | 5 |
| 4 | 1 | 1 | 2 | 3 | 7 |
| 5 | 2 | 2 | 3 | 4 | 11 |
| 6 | 3 | 3 | 4 | 6 | 16 |
| 7 | 4 | 4 | 6 | 9 | 23 |
| 8 | 6 | 6 | 9 | 13 | 34 |
| 9 | 9 | 9 | 13 | 19 | 50 |
| 10 | 13 | 13 | 19 | 28 | 73 |

1. таким образом, ответ: 13 + 13 + 19 + 28 = 73 решения.

**Решение (метод отображений**[[22]](#footnote-22)**, решение А.Н. Носкина):**

1. построим таблицу, в которой переберем все варианты **x1,** **x2, x3**, поскольку в первом логическом уравнении три переменных, то таблица будет иметь 8 строк (8 = 23); уберем из таблицы (желтая заливка) такие значения **x2 и x3**, при которых первое уравнение не имеет решения.

|  |  |  |
| --- | --- | --- |
| **x1** | **x2** | **x3** |
| 0 | **0** | **0** |
| **1** |
| 1 | **0** |
| 1 |
| 1 | 0 | 0 |
| 1 |
| 1 | 0 |
| 1 |

1. анализируя таблицу, строим правило отображения пар переменных

(например, паре **x1x2** **=** **00 не соответствуют** ни одной пары **x2x3** , и наоборот паре **x1x2= 01 соответствует только** пара **x3x4** **=** **11)**.

**x1x2 x2x3**

00

00

01

01

10

10

11

11

1. Заполняем таблицу, вычисляя количество пар переменных, при котором система имеет решение.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **x1x2** | **x2x3** | **x3x4** | **x4x5** | **x5x6** | **x6x7** | **x7x8** | **x8x9** | **x9x10** |
| 00 | 0 | 1 | 1 | 2 | 3 | 4 | 6 | 9 | 13 |
| 01 | 1 | 1 | 1 | 2 | 3 | 4 | 6 | 9 | 13 |
| 10 | 1 | 1 | 2 | 3 | 4 | 6 | 9 | 13 | 19 |
| 11 | 1 | 2 | 3 | 4 | 6 | 9 | 13 | 19 | 28 |

1. таким образом, ответ: 13 + 13 + 19 + 28 = 73 решения.

### Ещё пример задания:

**P-18**. *Сколько различных решений имеет логическое уравнение*

**(¬X1****∨****X2)** **∧** **(¬X2****∨****X3)** **∧** **(¬X3****∨****X4)** **∧** **(¬X4****∨****X5)** **∧** **(¬X5****∨****X6) = 1**

*где x1, x2, …, x6 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение:**

1. перепишем уравнение, заменив знаки логических операций:



1. учитывая, что , заменяем все выражения в скобках на импликацию:



1. решение уравнения можно записать в виде шести двоичных знаков, которые обозначают соответственно, переменные 
2. далее вспомним, что импликация дает ложное значение, если её первая часть (посылка) истинна, а вторая (следствие) ложно, поэтому изсразу следует, что 
3. это значит, что в исходном выражении появится нуль, если в цепочке битов, соответствующей значениям переменных, появится комбинация 10, то есть предыдущее значение истинно, а следующее за ним – ложно
4. поэтому решениями этого уравнения будут все комбинации значений переменных, для которых в соответствующей битовой цепочке нет последовательности 10;
5. таких цепочек всего 7:

000000, 000001, 000011, 000111, 001111, 011111, 111111

1. таким образом, ответ: 7 решений.

### Ещё пример задания:

**P-17**. *Сколько различных решений имеет система уравнений*

**¬X1** **∨ X2 = 1**

**¬X2** **∨ X3 = 1**

**...**

**¬X9** **∨ X10 = 1**

*где x1, x2, …, x10 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение (последовательное решение, через единицы):**

1. количество комбинаций 10 логических переменных равно 210 = 1024, поэтому вариант с построением полной таблицы истинности отпадает сразу
2. сначала рассмотрим первое уравнение ; согласно таблице истинности операции «ИЛИ» оно имеет 3 решения (точнее, с учетом других переменных, 3 группы решений): (0,0,\*), (0,1,\*) и (1,1,\*); здесь звездочка означает, что остальные 8 переменных могут быть любыми
3. выпишем все решения в столбик, чтобы была видна закономерность:

(0,0,\*)

(0,1,\*)

(1,1,\*)

1. заметим, что при X2 = 0 значение X1 должно быть равно 0, а при X2 = 1 значение X1 может быть любым
2. второе уравнение, рассматриваемое отдельно, тоже имеет 3 группы решений: (x1,0,0,\*), (x1,0,1,\*) и (x1,1,1,\*), где x1, – некоторое логическое значение переменной X1
3. решения системы первых двух уравнений – это те комбинации значений переменных, которые удовлетворяют одновременно и первому, и второму
4. из п. 4 следует, что при X2 = 0 значение X1 должно быть равно 0, а при X2 = 1 значение X1 может быть любым, поэтому решение системы двух первых уравнений включает 4 группы: из (x1,0,0,\*) и (x1,0,1,\*) при X1 = 0получаем две группы

(0,0,0,\*) и (0,0,1,\*)

и из (x1,1,1,\*) получается еще две:

(0,1,1,\*) и (1,1,1,\*).

1. таким образом, система из двух уравнений имеет 4 решения
2. выпишем все решения в столбик, чтобы была видна закономерность:

(0,0,0,\*)

(0,0,1,\*)

(0,1,1,\*)

(1,1,1,\*)

1. таким образом, если X­3 = 0, все предыдущие переменные определяются однозначно – они должны быть равны нулю (идем по системе «снизу вверх»); если же X­3 = 1, то предыдущие переменные могут быть любыми, второе уравнение их не ограничивает
2. поэтому при увеличении числа переменных на единицу количество решений также увеличивается на единицу
3. аналогично доказывается, что система из 3 уравнений имеет 5 решений, и т.д., то есть, система из 9 уравнений с 10 переменными имеет 11 решений
4. таким образом, ответ: 11 решений.

**Решение (последовательное решение, через нули):**

1. сначала рассмотрим первое уравнение ; согласно таблице истинности операции «ИЛИ» оно НЕ выполняется только в одном случае (точнее, с учетом других переменных, для одной группы комбинаций): (1,0,\*) здесь звездочка означает, что остальные 8 переменных могут быть любыми
2. общее количество комбинаций X1 и X2 ­­равно 22 = 4, поэтому число решений первого уравнения равно 4 – 1 = 3
3. второе уравнение, рассматриваемое отдельно, тоже ложно только для одной комбинации имеет 3 группы решений: (x1,1,0,\*), где x1, – некоторое логическое значение переменной X1
4. теперь рассмотрим вместе первое и второе уравнения и определим, в скольких случаях хотя бы одно из них неверно
5. множества (1,0,x­­3,\*) и (x1,1,0,\*) не пересекаются, потому что в первом X2 = 0, а во втором X2= 1, поэтому система из двух уравнений не выполнена для 4-х комбинаций:

(1,0,0,\*), (1,0,1,\*), (0,1,0,\*) и (1,1,0,\*)

1. общее количество комбинаций трех логический переменных равно 23 = 8, поэтому количество решений системы из двух уравнений равно 8 – 4 = 4
2. аналогично доказывается, что система из 3 уравнений имеет 5 решений, и т.д., то есть, система из 9 уравнений с 10 переменными имеет 11 решений
3. таким образом, ответ: 11 решений.

**Решение (табличный метод):**

1. рассмотрим все решения первого уравнения  по таблице истинности:

|  |  |  |
| --- | --- | --- |
|  | **X2** | **X1** |
| 1 | 0 | 0 |
| 0 | 0 | 1 |
| 1 | 1 | 0 |
| 1 | 1 | 1 |

1. строчка, выделенная красным фоном, не удовлетворяет условию, поэтому дальше ее рассматривать не будем
2. теперь подключаем третью переменную и второе уравнение:

|  |  |  |
| --- | --- | --- |
| **X3** | **X2** | **X1** |
| ? | 0 | 0 |
| ? | 1 | 0 |
| ? | 1 | 1 |

1. при каких значениях переменной X3 будет верно условие****? Очевидно, что на это уже не влияет X­1 (этот столбец выделен зеленым цветом). Если X2= 1, то сразу получаем, что X3 = 1 (иначе **):**

|  |  |  |
| --- | --- | --- |
| **X3** | **X2** | **X1** |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |
| 1 | 1 | 1 |

1. как видно из таблицы, верхняя строчка предыдущей таблицы (где были все нули) дает два решения при подключении очередного уравнения, а все остальные – по одному
2. понятно, что такая же ситуация будет продолжаться и дальше, то есть, при добавлении каждой новой переменной число решений увеличивается на 1
3. рассуждая таким образом и дальше, получаем, что для 3-х уравнений с 4-мя переменными будет 5 решений, для 4 уравнений – 6 решений, …, а для 9 уравнений – 11 решений
4. обратите внимание на форму таблицы – единицы и нули образуют два треугольника
5. таким образом, ответ: 11 решений.

|  |
| --- |
| **Рекомендации**:   * + по-видимому, лучший способ решения задач этого типа основан на двух идеях:     1. замена переменных (если она возможна), позволяющая сократить количество неизвестных и таким образом упростить решение     2. последовательное решение уравнений, начиная с первого, затем система из первых двух, первых трех и т.д.   + для записи хода решения и минимизации путаницы лучше использовать табличный метод, при котором все переменные, от которых зависит очередное уравнение, размещены в крайних левых столбцах таблицы |

### Еще пример задания:

**P-16**. *Сколько различных решений имеет система уравнений*

**¬(X1**≡**X2)** **∨ (X3**≡**X4) = 1**

**¬(X3**≡**X4)** **∨ (X5**≡**X6) = 1**

**¬(X5**≡**X6)** **∨ (X7**≡**X8) = 1**

**¬(X7**≡**X8)** **∨ (X9**≡**X10) = 1**

*где x1, x2, …, x10 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение:**

1. количество комбинаций 10 логических переменных равно 210 = 1024, поэтому вариант с построением полной таблицы истинности отпадает сразу
2. заметим, что при обозначениях , , ,  и  мы получаем систему из 4 уравнений и 5 независимыми переменными; эта система уравнений относится к типу, который рассмотрен в предыдущей разобранной задаче:

**¬Y1** **∨ Y2 = 1**

**¬Y2** **∨ Y3 = 1**

**¬Y3** **∨ Y4 = 1**

**¬Y4** **∨ Y5 = 1**

1. как следует из разбора предыдущей задачи, такая система имеет 5+1 = 6 решений для переменных Y1 … Y5
2. теперь нужно получить количество решений в исходных переменных, X1 … X10; для этого заметим, что переменные Y1 … Y5 независимы;
3. предположим, что значение Y1 известно (0 или 1); поскольку , по таблице истинности операции «эквивалентность» (истина, когда два значения одинаковы), есть **две** соответствующих пары (X1;X2) (как для случая Y1 = 0, так и дляслучая Y1 = 1)
4. у нас есть 5 переменных Y1 … Y5, каждая их комбинация дает 2 пары (X1;X2), 2 пары (X3;X4), 2 пары (X5;X6), 2 пары (X7;X8) и 2 пары (X9;X10), то есть всего 25 = 32 комбинации исходных переменных
5. таким образом, общее количество решений равно 6 ·32 = 192
6. ответ: 192 решения

**Решение (метод отображений[[23]](#footnote-23), решение А.Н. Носкина):**

1. сначала построим таблицу, в которой переберем все варианты **x1,** **x2, x3,x4**, поскольку в первом логическом уравнении четыре переменных, то таблица будет иметь 16 строк (16=24); уберем из таблицы (желтая заливка) такие значения **x4**, при которых первое уравнение не имеет решения.

|  |  |  |  |
| --- | --- | --- | --- |
| **x1** | **X2** | **X3** | **X4** |
| 0 | 0 | 0 | 0 |
| **1** |
| 1 | **0** |
| 1 |
| 1 | 0 | 0 |
| 1 |
| 1 | 0 |
| 1 |
| 1 | 0 | 0 | 0 |
| 1 |
| 1 | 0 |
| 1 |
| 1 | 0 | 0 |
| **1** |
| 1 | **0** |
| 1 |

1. анализируя таблицу, строим правило отображения пар переменных (например, паре **x1x2=00 соответствуют** пара **x3x4= 00 и 11,** и наоборот, для пары **x1x2=00** нет связей **x3x4= 01 и 10)**.

**x1x2 x3x4**

00

00

01

01

10

10

11

11

1. Заполняем таблицу, вычисляя количество пар переменных, при котором система имеет решение.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **x1x2** | **x3x4** | **x5x6** | **x7x8** | **x9x10** |
| 00 | 1 | 4 | 12 | 32 | 80 |
| 01 | 1 | 2 | 4 | 8 | 16 |
| 10 | 1 | 2 | 4 | 8 | 16 |
| 11 | 1 | 4 | 12 | 32 | 80 |

1. таким образом, ответ: 80+ 16 + 16 + 80 = 192 решения.

### Еще пример задания:

**P-15**. *Сколько различных решений имеет система уравнений*

**(X1****∧****X2)** **∨ (¬X1****∧****¬X2)** **∨ (¬X3****∧****X4)** **∨ (X3****∧****¬X4) = 1**

**(X3****∧****X4)** **∨ (¬X3****∧****¬X4)** **∨ (¬X5****∧****X6)** **∨ (X5****∧****¬X6) = 1**

**(X5****∧****X6)** **∨ (¬X5****∧****¬X6)** **∨ (¬X7****∧****X8)** **∨ (X7****∧****¬X8) = 1**

**(X7****∧****X8)** **∨ (¬X7****∧****¬X8)** **∨ (¬X9****∧****X10)** **∨ (X9****∧****¬X10) = 1**

*где x1, x2, …, x10 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение:**

1. количество комбинаций 10 логических переменных равно 210 = 1024, поэтому вариант с построением полной таблицы истинности отпадает сразу
2. решать такую систему «в лоб» достаточно сложно, нужно попробовать ее упростить
3. заметим, что

**(X1****∧****X2)** **∨ (¬X1****∧****¬X2) = (X1 ≡ X2),**

где символ **≡** означает операцию «эквивалентность» (значения равны);

1. кроме того,

**(¬X3****∧****X4)** **∨ (X3****∧****¬X4) = (X3 ⊕ X4) = ¬(X3 ≡ X4),**

где символ **⊕** означает операцию «исключающее ИЛИ» (значения НЕ равны); это операция, обратная эквивалентности

1. используем замену переменных, выделив члены, объединяющие пары исходных переменных (X1 и X2, X3 и X4, X5 и X6, X7 и X8, X9 и X10)

**Y1 = ¬(X1 ≡ X2) Y2 = ¬(X3 ≡ X4)**

**Y3 = ¬(X5 ≡ X6) Y4 = ¬(X7 ≡ X8)**

**Y5 = ¬(X9 ≡ X10)**

1. при этих обозначения система уравнений преобразуется к виду

**¬Y1** **∨ Y2 = 1**

**¬Y2** **∨ Y3 = 1**

**¬Y3** **∨ Y4 = 1**

**¬Y4** **∨ Y5 = 1**

1. как показано выше (при разборе пред-предыдущей задачи), такая система имеет 5+1 = 6 решений для независимых переменных Y1 … Y5
2. предположим, что значение Y1 известно (0 или 1); поскольку , по таблице истинности операции «эквивалентность» есть **две** соответствующих пары (X1;X2) (как для случая Y1 = 0, так и дляслучая Y1 = 1)
3. у нас есть 5 переменных Y1 … Y5, каждая их комбинация дает 2 пары (X1;X2), 2 пары (X3;X4), 2 пары (X5;X6), 2 пары (X7;X8) и 2 пары (X9;X10), то есть всего 25 = 32 комбинации исходных переменных
4. таким образом, общее количество решений равно 6 ·32 = 192
5. ответ: 192 решения

**Решение (метод отображений[[24]](#footnote-24), решение А.Н. Носкина):**

1. упростим систему уравнений, заметим, что

**(X1****∧****X2)** **∨ (¬X1****∧****¬X2) = (X1 ≡ X2),**

где символ **≡** означает операцию «эквивалентность» (значения равны);

1. кроме того,

**(¬X3****∧****X4)** **∨ (X3****∧****¬X4) = (X3 ⊕ X4) = ¬(X3 ≡ X4),**

где символ **⊕** означает операцию «исключающее ИЛИ» (значения НЕ равны); это операция, обратная эквивалентности;

1. при этих обозначения система уравнений преобразуется к виду

**(X1 ≡ X2)∨ ¬(X3 ≡ X4) = 1**

**(X3 ≡ X4)∨ ¬(X5 ≡ X6) = 1**

**(X5 ≡ X6)∨ ¬(X7 ≡ X8) = 1**

**(X7 ≡ X8)∨ ¬(X9 ≡ X10) = 1**

1. построим таблицу, в которой переберем все варианты **x1,** **x2, x3,x4**, поскольку в первом логическом уравнении четыре переменных, то таблица будет иметь 16 строк (16=24);

уберем из таблицы (желтая заливка) такие значения **x4**, при которых первое уравнение не имеет решения.

|  |  |  |  |
| --- | --- | --- | --- |
| **x1** | **X2** | **X3** | **X4** |
| 0 | 0 | 0 | 0 |
| 1 |
| 1 | 0 |
| 1 |
| 1 | 0 | **0** |
| 1 |
| 1 | 0 |
| **1** |
| 1 | 0 | 0 | **0** |
| 1 |
| 1 | 0 |
| **1** |
| 1 | 0 | 0 |
| 1 |
| 1 | 0 |
| 1 |

1. анализируя таблицу, строим правило отображения пар переменных

(например, паре **x1x2=01 соответствуют** пара **x3x4** **=** **01 и 10,** и, наоборот, для пары **x1x2** **=** **01** нет связей **x3x4** **=** **00 и 11).**

**x1x2 x3x4**

00

00

01

01

10

10

11

11

1. заполняем таблицу, вычисляя количество пар переменных, при котором система имеет решение:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **x1x2** | **x3x4** | **x5x6** | **x7x8** | **x9x10** |
| 00 | 1 | 2 | 4 | 8 | 16 |
| 01 | 1 | 4 | 12 | 32 | 80 |
| 10 | 1 | 4 | 12 | 32 | 80 |
| 11 | 1 | 2 | 4 | 8 | 16 |

1. таким образом, ответ: 16 +80 + 80 +16 = 192 решения.

### Еще пример задания:

**P-14**. *Сколько различных решений имеет система уравнений*

**((X1**≡**X2)** **∧ (X3**≡**X4))** **∨ (¬(X1**≡**X2)** **∧ ¬(X3**≡**X4)) = 1**

**((X3**≡**X4)** **∧ (X5**≡**X6))** **∨ (¬(X3**≡**X4)** **∧ ¬(X5**≡**X6)) = 1**

**((X5**≡**X6)** **∧ (X7**≡**X8))** **∨ (¬(X5**≡**X6)** **∧ ¬(X7**≡**X8)) = 1**

**((X7**≡**X8)** **∧ (X9**≡**X10))** **∨ (¬(X7**≡**X8)** **∧ ¬(X9**≡**X10)) = 1**

*где x1, x2, …, x10 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение:**

1. количество комбинаций 10 логических переменных равно 210 = 1024, поэтому вариант с построением полной таблицы истинности отпадает сразу
2. решать такую систему «в лоб» достаточно сложно, нужно попробовать ее упростить
3. рассмотрим первое уравнение, заменив обозначения логических операций на более простые:

,

где  и . Выражение в левой части последнего равенства – это операция эквивалентности между Y­1 и Y2, то есть первое уравнение запишется в виде



1. аналогично, вводя обозначения ,  и , запишем исходную систему в виде

**(Y1** ≡**Y2) = 1**

**(Y2** ≡**Y3) = 1**

**(Y3** ≡**Y4) = 1**

**(Y4** ≡**Y5) = 1**

заметим, что все переменные здесь независимы друг от друга

1. найдем решение этой системы относительно **независимых** переменных Y1 … Y5
2. первое уравнение имеет два решения (с учетом остальных переменных – две группы решений): (0,0,\*) и (1,1,\*), где \* обозначает остальные переменные, которые могут быть любыми
3. второе уравнение тоже имеет две группы решений: (Y1,0,0,\*) и (Y 1,1,1,\*), где Y 1 обозначает некоторое значение переменной Y 1
4. теперь ищем решения, которые удовлетворяют и первому, и второму уравнению; очевидно, что их всего 2: (0,0,0,\*) и (1,1,1,\*)
5. рассуждая дальше аналогичным образом, приходим к выводу, что система имеет всего два решения относительно переменных Y1 … Y5: все нули и все единицы
6. теперь нужно получить количество решений в исходных переменных, X1 … X10; для этого вспомним, что переменные Y1 … Y5 независимы;
7. предположим, что значение Y1 известно (0 или 1); поскольку , по таблице истинности операции «эквивалентность» (истина, когда два значения одинаковы), есть **две** соответствующих пары (X1;X2) (как для случая Y1 = 0, так и дляслучая Y1 = 1)
8. у нас есть 5 переменных Y1 … Y5, каждая их комбинация дает 2 допустимых пары (X1;X2), 2 пары (X3;X4), 2 пары (X5;X6), 2 пары (X7;X8) и 2 пары (X9;X10), то есть всего 25 = 32 комбинации исходных переменных
9. таким образом, общее количество решений равно 2 ·32 = 64
10. ответ: 64 решения

**Решение (табличный метод):**

1. так же, как и в предыдущем варианте, с помощью замену переменных сведем систему к виду:

**(Y1** ≡**Y2) = 1**

**(Y2** ≡**Y3) = 1**

**(Y3** ≡**Y4) = 1**

**(Y4** ≡**Y5) = 1**

1. рассмотрим все решения первого уравнения  по таблице истинности:

|  |  |  |
| --- | --- | --- |
|  | **Y2** | **Y1** |
| 1 | 0 | 0 |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

1. строчки, выделенные красным фоном, не удовлетворяют условию, поэтому дальше их рассматривать не будем
2. теперь подключаем третью переменную и второе уравнение:

|  |  |  |
| --- | --- | --- |
| **Y3** | **Y2** | **Y1** |
| ? | 0 | 0 |
| ? | 1 | 1 |

1. при каких значениях переменной X3 будет верно условие****? Очевидно, что на это уже не влияет Y­1 (этот столбец выделен зеленым цветом). Cразу получаем два решения**:**

|  |  |  |
| --- | --- | --- |
| **Y3** | **Y2** | **Y1** |
| 0 | 0 | 0 |
| 1 | 1 | 1 |

1. как видно из таблицы, каждая строчка предыдущей таблицы дает одно решение при подключении очередного уравнения, поэтому для любого количества переменных система имеет 2 решения – все нули и все единицы
2. так же, как и в предыдущем способе, переходим к исходным переменным и находим общее количество решений: 2 ·32 = 64
3. ответ: 64 решения

**Решение (метод отображений[[25]](#footnote-25), решение А.Н. Носкина):**

1. построим таблицу, в которой переберем все варианты **x1,** **x2, x3,x4**, поскольку в первом логическом уравнении четыре переменных, то таблица будет иметь 16 строк (16=24);

уберем из таблицы (желтая заливка) такие значения **x4**, при которых первое уравнение не имеет решения.

|  |  |  |  |
| --- | --- | --- | --- |
| **x1** | **X2** | **X3** | **X4** |
| 0 | 0 | 0 | 0 |
| **1** |
| 1 | **0** |
| 1 |
| 1 | 0 | **0** |
| 1 |
| 1 | 0 |
| **1** |
| 1 | 0 | 0 | **0** |
| 1 |
| 1 | 0 |
| **1** |
| 1 | 0 | 0 |
| **1** |
| 1 | **0** |
| 1 |

1. анализируя таблицу, строим правило отображения пар переменных

(например, паре **x1x2** **=** **00 соответствуют** пара **x3x4** **=** **00 и 11,** и, наоборот, для пары **x1x2** **=** **00** нет связей **x3x4** **=** **01 и 10).**

**x1x2 x3x4**

00

00

01

01

10

10

11

11

1. заполняем таблицу, вычисляя количество пар переменных, при котором система имеет решение.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **x1x2** | **x3x4** | **x5x6** | **x7x8** | **x9x10** |
| 00 | 1 | 2 | 4 | 8 | 16 |
| 01 | 1 | 2 | 4 | 8 | 16 |
| 10 | 1 | 2 | 4 | 8 | 16 |
| 11 | 1 | 2 | 4 | 8 | 16 |

1. таким образом, ответ: 16 +16 + 16 +16 = 64 решения.

### Еще пример задания:

**P-13**. *Сколько различных решений имеет система уравнений*

**(X2** ≡ **X1)** **∨ (X2** **∧** **X3)** **∨ (¬X2** **∧¬** **X3)= 1**

**(X3** ≡ **X1)** **∨ (X3** **∧** **X4)** **∨ (¬X3** **∧¬** **X4)= 1**

**...**

**(X9** ≡ **X1)** **∨ (X9** **∧** **X10)** **∨ (¬X9** **∧¬** **X10)= 1**

**(X10** ≡ **X1) = 0**

*где x1, x2, …, x10 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение (табличный метод):**

1. количество комбинаций 10 логических переменных равно 210 = 1024, поэтому вариант с построением полной таблицы истинности отпадает сразу
2. перепишем уравнения, используя более простые обозначения операций





...





1. заметим, что по свойству операции эквивалентности, поэтому уравнения можно переписать в виде





...





1. первое уравнение выполняется, когда есть X2 равно X1 или X3
2. по таблице истинности находим 6 вариантов (для удобства мы будем записывать сначала столбец для X1, а потом для всех остальных в обратном порядке):

|  |  |  |
| --- | --- | --- |
| **X1** | **X3** | **X2** |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

обратите внимание, что в каждой строчке в первых двух столбцах должно быть по крайней мере одно значение, равное значению в третьем столбце (который выделен желтым)

1. добавим еще одно уравнение и еще одну переменную X­4:

|  |  |  |  |
| --- | --- | --- | --- |
| **X1** | **X4** | **X3** | **X2** |
| 0 | ? | 0 | 0 |
| 0 | ? | 1 | 0 |
| 0 | ? | 1 | 1 |
| 1 | ? | 0 | 0 |
| 1 | ? | 0 | 1 |
| 1 | ? | 1 | 1 |

1. чтобы «подключить» второе уравнение, нужно использовать то же самое правило: каждой строчке в первых двух столбцах должно быть, по крайней мере, одно значение, равное значению в третьем столбце (который выделен желтым); это значит, что в первой и последней строчках (где X­1 = X3) значение X4 может быть любое (0 или 1), а в остальных строчках – только строго определенное:

|  |  |  |  |
| --- | --- | --- | --- |
| **X1** | **X4** | **X3** | **X2** |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 |

1. таким образом, количество решений при подключении очередного уравнения к системе возрастает на 2!
2. подключили X5 – получили 10 решений, X6 – получили 12 решений, X7 – получили 14 решений, X8 – получили 16 решений, X9 – получили 18 решений, X10 – получили 20 решений.
3. остается одно последнее уравнение **(X10** ≡ **X1)** **=** **0,** из которого следует, что X10 не равен X­1
4. из таблицы следует, что только в первой и последней строчках значения первой и последней переменных совпадают, то есть из полученных 20 решений нужно отбросить 2
5. таким образом, получается 20 – 2 = 18 решений
6. ответ: 18 решений

### Еще пример задания:

**P-12**. *Сколько различных решений имеет система уравнений*

**(X1** **∧** **X2)** **∨ (¬X1** **∧** **¬X2)** **∨ (X1** ≡ **X3) = 1**

**(X2** **∧** **X3)** **∨ (¬X2** **∧** **¬X3)** **∨ (X2** ≡ **X4) = 1**

**...**

**(X8** **∧** **X9)** **∨ (¬X8** **∧** **¬X9)** **∨ (X8** ≡ **X10) = 1**

*где x1, x2, …, x10 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение (табличный метод):**

1. количество комбинаций 10 логических переменных равно 210 = 1024, поэтому вариант с построением полной таблицы истинности отпадает сразу
2. перепишем уравнения, используя более простые обозначения операций





...



1. заметим, что по свойству операции эквивалентности, поэтому уравнения можно переписать в виде





...



1. сделать замену переменных так, чтобы новые переменные был независимы друг от друга, здесь довольно затруднительно, поэтому будем решать уравнения последовательно табличным методом
2. рассмотрим все возможные комбинации первых двух переменных ­X1 и X2, и сразу попытаемся для каждой из них подобрать значения третьей так, чтобы выполнялось первое уравнение :

|  |  |  |
| --- | --- | --- |
| **X3** | **X2** | **X1** |
| ? | 0 | 0 |
| ? | 0 | 1 |
| ? | 1 | 0 |
| ? | 1 | 1 |

1. очевидно, что в первой и последней строчках таблицы, где , значения X3 могут быть любыми, то есть каждая из этих строчек дает два решения; в то же время во второй и третьей строках, где , мы сразу получаем, что для выполнения первого равнения необходимо , то есть, эти две строчки дают по одному решению:

|  |  |  |
| --- | --- | --- |
| **X3** | **X2** | **X1** |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

1. заметим, что количество решений для каждой строчки исходной таблицы (с двумя переменными) определялось лишь тем, равны значения в двух последних столбцах (X2 и X1) или не равны;
2. также заметим, что в новой таблице в самой верхней и самой нижней строках значения X3 и X2 равны, а в остальных не равны (их 4 штуки); поэтому на следующем шаге (при подключении четвертой переменной и третьего уравнения) верхняя и нижняя строки дадут 2 варианта с равными X­4 и X3, и 2 + 4 = 6 вариантов, где X­4 и X3 не равны
3. в общем виде: если на шаге **i** в таблице решений есть

**ni** строк, где значения в двух самых левых столбцах таблицы равны, и …

**mi** строк, где значения в двух самых левых столбцах таблицы не равны,

то на следующем шаге будет столько же (**ni**) строк с равными значения в двух самых последних столбцах и **ni+mi**строк с неравными значениями

1. эту последовательность можно записать в виде таблицы (**i** – число задействованных переменных):

|  |  |  |  |
| --- | --- | --- | --- |
| **i** |  |  | всего решений |
| 3 | 2 | 4 | 6 |
| 4 | 2 | 2+4=6 | 8 |
| 5 | 2 | 2+6=8 | 10 |
| 6 | 2 | 2+8=10 | 12 |
| 7 | 2 | 2+10=12 | 14 |
| 8 | 2 | 2+12=14 | 16 |
| 9 | 2 | 2+14=16 | 18 |
| 10 | 2 | 2+16=18 | 20 |

1. таким образом, для системы с 10 переменными общее количество решений равно 2 + 18 = 20
2. ответ: 20 решений

### Еще пример задания:

**P-11**. *Сколько различных решений имеет система уравнений*

**(X1** **∧** **X2)** **∨ (¬X1** **∧** **¬X2)** **∨ (X2** **∧** **X3)** **∨ (¬X2** **∧** **¬X3) = 1**

**(X2** **∧** **X3)** **∨ (¬X2** **∧** **¬X3)** **∨ (X3** **∧** **X4)** **∨ (¬X3** **∧** **¬X4) = 1**

**...**

**(X8** **∧** **X9)** **∨ (¬X8** **∧** **¬X9)** **∨ (X9** **∧** **X10)** **∨ (¬X9** **∧** **¬X10) = 1**

*где x1, x2, …, x10 – логические переменные? В ответе не нужно перечислять все различные наборы значений переменных, при которых выполнено данное равенство. В качестве ответа нужно указать количество таких наборов.*

**Решение (табличный метод):**

1. количество комбинаций 10 логических переменных равно 210 = 1024, поэтому вариант с построением полной таблицы истинности отпадает сразу
2. перепишем уравнения, используя более простые обозначения операций





...



1. заметим, что по свойству операции эквивалентности, поэтому уравнения можно переписать в виде





...



1. сделать замену переменных так, чтобы новые переменные был независимы друг от друга, здесь довольно затруднительно, поэтому будем решать уравнения последовательно табличным методом
2. рассмотрим все возможные комбинации первых двух переменных ­X1 и X2, и сразу попытаемся для каждой из них подобрать значения третьей так, чтобы выполнялось первое уравнение :

|  |  |  |
| --- | --- | --- |
| **X3** | **X2** | **X1** |
| ? | 0 | 0 |
| ? | 0 | 1 |
| ? | 1 | 0 |
| ? | 1 | 1 |

1. очевидно, что в первой и последней строчках таблицы, где , значения X3 могут быть любыми, то есть каждая из этих строчек дает два решения; в то же время во второй и третьей строках, где , мы сразу получаем, что для выполнения первого равнения необходимо , то есть, эти две строчки дают по одному решению:

|  |  |  |
| --- | --- | --- |
| **X3** | **X2** | **X1** |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 0 | 1 |
| 1 | 1 | 0 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

1. заметим, что количество решений для каждой строчки исходной таблицы (с двумя переменными) определялось лишь тем, равны значения в двух последних столбцах (X2 и X1) или не равны;
2. переставим строки так, чтобы сверху стояли те строки, в которых X2 = X3:

|  |  |  |
| --- | --- | --- |
| **X3** | **X2** | **X1** |
| 0 | 0 | 0 |
| 0 | 0 | 1 |
| 1 | 1 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 1 |

1. также заметим, что в новой таблице в четырех строках значения X2 = X3, а в остальных 2-х эти переменные не равны;
2. поэтому на следующем шаге (при подключении четвертой переменной и третьего уравнения) 4 первые строки дадут по 2 варианта (всего 4·2=8) решений, из них 4 штуки с равными X­4 и X3, и 4 варианта, где X­4 и X3 не равны
3. две нижние строки, где X2 ≠ X3, дадут 2 варианта, где X­4 и X3 равны
4. в общем виде: если на шаге **i** в таблице решений есть

**ni** строк, где значения в двух самых левых столбцах таблицы равны, и …

**mi** строк, где значения в двух самых левых столбцах таблицы не равны,

то на следующем шаге будет (**ni+mi**) строк с равными значения в двух самых последних столбцах и **ni**строк с неравными значениями

1. эту последовательность можно записать в виде таблицы (**i** – число задействованных переменных):

|  |  |  |  |
| --- | --- | --- | --- |
| **i** |  |  | всего решений |
| 3 | 4 | 2 | 6 |
| 4 | 4+2=6 | 4 | 10 |
| 5 | 6+4=10 | 6 | 16 |
| 6 | 10+6=16 | 10 | 26 |
| 7 | 16+10=26 | 16 | 42 |
| 8 | 26+16=42 | 26 | 68 |
| 9 | 42+26=68 | 42 | 110 |
| 10 | 68+42=110 | 68 | 178 |

1. таким образом, для системы с 10 переменными общее количество решений равно   
   110 + 68 = 178
2. ответ: 178 решений

**Решение (использование дерева для представления решения):**

1. идея представления множества решений в виде дерева использовалась, например, в решениях **О.А. Тузовой** (Санкт-Петербург, школа № 550) и **М.В. Демидовой** (г. Пермь, гимназия №17); как верно отметила О.А. Тузова, предложенный выше табличный метод по сути представляет собой компактную запись дерева
2. так же, как и в предыдущем варианте решения, перейдем к равносильной системе уравнений





...



1. все переменные логические, в принятых обозначениях каждая из них может быть равна 1 или 0; для X1 получаем два варианта, которые можно представить в виде

X1

1. при этом X­2 может быть любым, то есть, имеем всего 4 варианта

X1

X2

1. теперь рассматриваем переменную X3; если X1 = X2, то уравнение  выполняется при любом X3; если X1 ≠ X2, то это уравнение сразу дает X3 = X2; дерево получается уже неполным, число решений первого уравнения – 6:

X1

X2

X3

1. рассуждая аналогично, находим, что на следующем шаге (подключение переменной X4 и второго уравнения) получается 10 решений, затем – 16 и т.д.; в результате получается удвоенная последовательность Фибоначчи (2, 4, 6, 10, 16, 26, …), в которой каждый следующий элемент равен сумме двух предыдущих:

|  |  |
| --- | --- |
| **i** | число решений |
| 3 | 6 |
| 4 | 10 |
| 5 | 16 |
| 6 | 26 |
| 7 | 42 |
| 8 | 68 |
| 9 | 110 |
| 10 | 178 |

1. в некоторых вариантах такой подход рассматривался совместно с *методом декомпозиции*: сначала предполагаем, что X1 = 0 и находим все решения для этого варианта; затем находим все решения при X1 = 1; после этого общее количество решений вычисляется как сумма полученных двух чисел
2. ответ: 178 решений

### Ещё пример задания:

**P-10**. *Каково наибольшее целое число X, при котором истинно высказывание*

**(50 < X·X)** → **(50 > (X+1)·(X+1))**

**Решение (вариант 1):**

1. это операция импликации между двумя отношениями  и 
2. попробуем сначала решить неравенства

, 

1. обозначим эти области на оси X:















на рисунке фиолетовые зоны обозначают область, где истинно выражение , голубая зона – это область, где истинно 

1. вспомним таблицу истинности операции «импликация»:

|  |  |  |
| --- | --- | --- |
| A | B | A→B |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

1. согласно таблице, заданное выражение истинно везде, кроме областей, где  и ; область истинности выделена зеленым цветом
2. поэтому наибольшее целое число, удовлетворяющее условию – это первое целое число, меньшее , то есть, 7
3. таким образом, верный ответ – 7 .

|  |
| --- |
| **Возможные проблемы**:   * + в этом примере потребовалось применить знания не только (и не столько) из курса информатики, но и умение решать неравенства   + нужно не забыть правила извлечения квадратного корня из обеих частей неравенства (операции с модулями) |

**Решение (вариант 2, преобразование выражения):**

1. сначала можно преобразовать импликацию, выразив ее через «ИЛИ» и «НЕ»:



1. это значит, что выражение истинно там, где  или 
2. дальнейшие действия точно такие же, как и в варианте 1.

|  |
| --- |
| **Возможные проблемы**:   * + нужно помнить формулу для преобразования импликации |

**Решение (вариант 3, математический):**

1. это операция импликации между двумя отношениями  и 
2. пусть  – истинно, тогда, с учетом того, что , находим, что  – ложно, таким образом, импликация  ложна
3. следовательно, импликация может быть истинной только при ; поскольку в этом случае высказывание ложно, то  при любом 
4. максимальное целое значение X, при котором , равно 7
5. таким образом, верный ответ – 7 .

### Еще пример задания:

**P-09**. *Каково наибольшее целое число X, при котором истинно высказывание*

**(10 < X·(X+1))** → **(10 > (X+1)·(X+2))**

**Решение (в целых числах):**

1. это операция импликации между двумя отношениями:

 и 

1. конечно, здесь можно применить тот же способ, что и в предыдущем примере, однако при этом понадобится решать квадратные уравнения (*не хочется…*)
2. заметим, что по условию нас интересуют **только целые числа,** поэтому можно попытаться как-то преобразовать исходное выражение, получив равносильное высказывание (как понятно из предыдущего примера, точные значения корней нас совершенно не интересуют!)
3. рассмотрим неравенство : очевидно, что может быть как положительным, так и отрицательным числом;
4. легко проверить, что в области  высказывание  истинно при всех целых , а в области  – при всех целых  (чтобы не запутаться, удобнее использовать **нестрогие неравенства**,  и , вместо  и )
5. поэтому для целых  можно заменить  на равносильное выражение



1. область истинности выражения  – объединение двух бесконечных интервалов:











1. теперь рассмотрим второе неравенство : очевидно, что так же может быть как положительным, так и отрицательным числом;
2. в области  высказывание  истинно при всех целых , а в области  – при всех целых , поэтому для целых  можно заменить  на равносильное выражение



1. область истинности выражения  – закрытый интервал, обозначенный голубой полоской















1. вспомним таблицу истинности операции «импликация»:

|  |  |  |
| --- | --- | --- |
| A | B | A→B |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

1. согласно таблице, заданное выражение истинно везде, кроме областей, где  и ; область истинности выделена на рисунке зеленым цветом;
2. обратите внимание, что значение  уже **не** входит в зеленую зону, потому что там  и , то есть импликация дает 0
3. по схеме видно, что максимальное целое число в зеленой области – 2
4. таким образом, верный ответ – 2.

|  |
| --- |
| **Возможные проблемы**:   * + нужно помнить, что мы рассматриваем значения выражения только для целых , при этом появляются свои особенности: может появиться желание продлить зеленую область до точки , что приведет к неверному ответу, потому что там уже  и |

### Еще пример задания:

**P-08**. *Сколько различных решений имеет уравнение*

**((K ∨ L)** → **(L ∧ M ∧ N)) = 0**

*где K, L, M, N – логические переменные? В ответе не нужно перечислять все различные наборы значений K, L, M и N, при которых выполнено данное равенство. В качестве ответа Вам нужно указать количество таких наборов.*

**Решение (вариант 1, разделение на части):**

1. перепишем уравнение, используя более простые обозначения операций:

**((K + L)** → **(L · M · N)) = 0**

1. из таблицы истинности операции «импликация» (см. первую задачу) следует, что это равенство верно тогда и только тогда, когда одновременно

**K + L = 1** и **L · M · N = 0**

1. из первого уравнения следует, что хотя бы одна из переменных, K или L, равна 1 (или обе вместе); поэтому рассмотрим три случая
2. если K = 1 и L = 0, то второе равенство выполняется при любых М и N; поскольку существует 4 комбинации двух логических переменных (00, 01, 10 и 11), имеем **4** разных решения
3. если K = 1 и L = 1, то второе равенство выполняется при М **·** N = 0; существует 3 таких комбинации (00, 01 и 10), имеем еще **3** решения
4. если K = 0, то обязательно L = 1 (из первого уравнения); при этом второе равенство выполняется при М **·** N = 0; существует 3 таких комбинации (00, 01 и 10), имеем еще **3** решения
5. таким образом, всего получаем 4 + 3 + 3 = 10 решений.

|  |
| --- |
| **Совет**:   * + лучше начинать с того уравнения, где меньше переменных |

|  |
| --- |
| **Возможные проблемы**:   * + есть риск потерять какие-то решения при переборе вариантов |

**Решение (вариант 2, через таблицы истинности):**

1. перепишем уравнение, используя более простые обозначения операций:

**((K + L)** → **(L · M · N)) = 0**

1. построим таблицу для логического выражения

**X = ((K + L)** → **(L · M · N))**

и подсчитаем, сколько в ней нулей, это и будет ответ

1. наше выражение зависит от четырех переменных, поэтому в таблице будет **24** **=** **16** строчек (16 возможных комбинация четырех логических значений)
2. подставляем различные комбинации в формулу для **X;** несмотря на большое количество вариантов, таблица строится легко: достаточно вспомнить, что выражение **K** **+** **L** ложно только при **K** **=** **L** **=** **0**, а выражение **L·M·N** истинно только при **L** **=** **M** **=** **N** **=** **1**.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *K* | *L* | *M* | *N* | *K+L* | *L·M·N* | *X* |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |

1. в последнем столбце 10 нулей; это значит, что есть 10 разных комбинаций, при которых выражение **X** равно нулю, то есть исходное уравнение имеет 10 решений
2. таким образом, всего 10 решений.

|  |
| --- |
| **Возможные проблемы**:   * + нужно строить таблицу истинности функции от 4 переменных, это трудоемко, легко ошибиться |

### Еще пример задания:

**P-07**. *Укажите значения переменных К, L, M, N, при которых логическое выражение*

**(¬(М ∨ L) ∧ К)** → **(¬К ∧ ¬М ∨ N)**

*ложно. Ответ запишите в виде строки из 4 символов: значений переменных К, L, М и N (в указанном порядке). Так, например, строка 1101 соответствует тому, что К=1, L=1, M=0, N=1.*

**Решение (вариант 1, анализ исходного выражения):**

1. запишем уравнение, используя более простые обозначения операций (условие «выражение ложно» означает, что оно равно логическому нулю):

****

1. из формулировки условия следует, что выражение должно быть ложно только для одного набора переменных
2. из таблицы истинности операции «импликация» (см. первую задачу) следует, что это выражение ложно тогда и только тогда, когда одновременно

**** и ****

1. первое равенство (логическое произведение равно 1) выполняется тогда и только тогда, когда ****и **;** отсюда следует ****(логическая сумма равна нулю), что может быть только при ****; таким образом, три переменных мы уже определили
2. из второго условия, , при  и  получаем ****
3. таким образом, правильный ответ – 1000.

|  |
| --- |
| **Возможные проблемы**:   * + переменные однозначно определяются только для ситуаций «сумма = 0» (все равны 0) и «произведение = 1» (все равны 1), в остальных случаях нужно рассматривать разные варианты   + не всегда выражение сразу распадается на 2 (или более) отдельных уравнения, каждое из которых однозначно определяет некоторые переменные |

**Решение (вариант 2, упрощение выражения):**

1. запишем уравнение, используя более простые обозначения операций:

****

1. заменим импликацию по формуле ****:

****

1. раскроем инверсию сложного выражения по формуле де Моргана ****:

****

1. упростим выражение ****:

****

1. мы получили уравнение вида «сумма = 0», в нем все слагаемые должны быть равны нулю
2. поэтому сразу находим ****
3. таким образом, правильный ответ – 1000.

|  |
| --- |
| **Замечание**:   * этот способ работает всегда и дает более общее решение; в частности, можно легко обнаружить, что уравнение имеет несколько решений (тогда оно не сведется к форме «сумма = 0» или «произведение = 1») |

|  |
| --- |
| **Возможные проблемы**:   * + нужно помнить правила преобразования логических выражений и хорошо владеть этой техникой |

### Еще пример задания:

**P-06**. *Составьте таблицу истинности для логической функции*

**X = (А ↔ B) ∨ ¬(A** → **(B ∨ C))**

*в которой столбец значений аргумента А представляет собой двоичную запись числа 27, столбец значений аргумента В – числа 77, столбец значений аргумента С – числа 120. Число в столбце записывается сверху вниз от старшего разряда к младшему. Переведите полученную двоичную запись значений функции X в десятичную систему счисления.*

**Решение (вариант 1):**

1. запишем уравнение, используя более простые обозначения операций:



1. это выражение с тремя переменными, поэтому в таблице истинности будет 23=8 строчек; следовательно, двоичная запись чисел, по которым строятся столбцы таблицы А, В и С, должна состоять из 8 цифр

|  |  |  |  |
| --- | --- | --- | --- |
| *А* | *В* | *С* | *X* |
| 0 | 0 | 0 |  |
| 0 | 1 | 1 |  |
| 0 | 0 | 1 |  |
| 1 | 0 | 1 |  |
| 1 | 1 | 1 |  |
| 0 | 1 | 0 |  |
| 1 | 0 | 0 |  |
| 1 | 1 | 0 |  |

1. переведем числа 27, 77 и 120 в двоичную систему, сразу дополняя запись до 8 знаков нулями в начале чисел

27 = 000110112 77 = 010011012 120 = 011110002

1. теперь можно составить таблицу истинности (см. рисунок справа), в которой строки переставлены в сравнении с традиционным порядком[[26]](#footnote-26); зеленым фоном выделена двоичная записи числа 27 (биты записываются сверху вниз), синим – запись числа 77 и розовым – запись числа 120:
2. вряд ли вы сможете сразу написать значения функции Х для каждой комбинации, поэтому удобно добавить в таблицу дополнительные столбцы для расчета промежуточных результатов (см. таблицу ниже)
3. заполняем столбцы таблицы:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *А* | *В* | *С* |  |  |  |  | *X* |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |

значение ** равно 1 только в тех строчках, где А = В

значение **равно 1 только в тех строчках, где В = 1 или С = 1

значение **равно 0 только в тех строчках, где А = 1 и В + С = 0

значение * –* этоинверсия предыдущего столбца (0 заменяется на 1, а 1 – на 0)

результат Х (последний столбец) – это логическая сумма двух столбцов, выделенных фиолетовым фоном

1. чтобы получить ответ, выписываем биты из столбца Х сверху вниз: Х = 101010112
2. переводим это число в десятичную систему: 101010112 = 27 + 25 + 23 + 21 + 20 = 171
3. таким образом, правильный ответ – 171.

|  |
| --- |
| **Возможные проблемы**:   * + нужно помнить таблицы истинности логических операций   + легко запутаться в многочисленных столбцах с однородными данными (нулями и единицами) |

**Решение (вариант 2, преобразование логической функции):**

1. выполним пп. 1-5 так же, как и в предыдущем способе
2. запишем уравнение, используя более простые обозначения операций:



1. раскроем импликацию через операции И, ИЛИ и НЕ ():

**

1. раскроем инверсию для выражения ** по формуле де Моргана:

**

1. таким образом, выражение приобретает вид 
2. отсюда сразу видно, что Х = 1 только тогда, когда А = В или (А = 1 и В = С = 0):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *А* | *В* | *С* | *X* | *Примечание* |
| 0 | 0 | 0 | 1 | А = В |
| 0 | 1 | 1 | 0 |  |
| 0 | 0 | 1 | 1 | А = В |
| 1 | 0 | 1 | 0 |  |
| 1 | 1 | 1 | 1 | А = В |
| 0 | 1 | 0 | 0 |  |
| 1 | 0 | 0 | 1 | А = 1, В = С = 0 |
| 1 | 1 | 0 | 1 | А = В |

1. чтобы получить ответ, выписываем биты из столбца Х сверху вниз: Х = 101010112
2. переводим это число в десятичную систему: 101010112 = 27 + 25 + 23 + 21 + 20 = 171
3. таким образом, правильный ответ – 171.

|  |
| --- |
| **Возможные проблемы**:   * + нужно помнить правила преобразования логических выражений и хорошо владеть этой техникой |

### Еще пример задания:

**P-05**. *A, B и С – целые числа, для которых истинно высказывание*

**¬(А = B) ∧ ((A > B)**→**(B > C)) ∧ ((B > A)**→**(С > B))**

*Чему равно В, если A = 45 и C = 43?.*

**Решение (вариант 1):**

1. обратим внимание, что это сложное высказывание состоит из трех простых

**¬(А = B)**

**(A > B)**→**(B > C)**

**(B > A)**→**(С > B)**

1. эти простые высказывания связаны операцией **∧** (И, конъюнкция), то есть, они должны выполняться одновременно
2. из **¬(А = B)=1** сразу следует, что **А ≠ B**
3. предположим, что **A > B**, тогда из второго условия получаем **1**→**(B > C)=1**; это выражение может быть истинно тогда и только тогда, когда **B > C = 1**
4. поэтому имеем **A > B > C**, этому условию соответствует только число 44
5. на всякий случай проверим и вариант **A < B**, тогда из второго условия получаем   
   0 →**(B > C)=1**; это выражение истинно при любом **B**;   
   теперь смотрим третье условие: получаем **1**→**(С > B)=1**; это выражение может быть истинно тогда и только тогда, когда **C > B**, и тут мы получили противоречие, потому что нет такого числа B, для которого **C > B > A**
6. таким образом, правильный ответ – 44.

**Решение (вариант 2, интуитивный):**

1. заметим, что между A и C расположено единственное число 44, поэтому можно предполагать, что именно это и есть ответ
2. проверим догадку, подставив в заданное выражение *A = 45, B = 44 и C = 43*

**¬(45 = 44) ∧ ((45 > 44)**→**(44 > 43)) ∧ ((44 > 45)**→**(43 > 44))**

1. заменим истинные условия на 1, а ложные – на 0:

**¬(0) ∧ (1**→**1) ∧ (0**→**0)**

1. вычисляем по таблице результаты операций **¬** (НЕ, отрицание) и → (импликация):

**1 ∧ 1 ∧ 1**

1. остается применить операцию **∧** (И, конъюнкция) – получаем 1, то есть, выражение истинно, что нам и нужно
2. таким образом, правильный ответ – 44.

|  |
| --- |
| **Возможные проблемы**:   * + не всегда удается сразу догадаться |

### Еще пример задания:

**P-04**. *Сколько различных решений имеет уравнение*

**(K ∧ L ∧ M) ∨ (¬L ∧ ¬M ∧ N) = 0**

*где K, L, M, N – логические переменные? В ответе не нужно перечислять все различные наборы значений K, L, M и N, при которых выполнено данное равенство. В качестве ответа Вам нужно указать количество таких наборов.*

**Решение (поиск неподходящих комбинаций):**

1. перепишем уравнение, используя более простые обозначения операций:

****

1. здесь используется сложение двух логических произведений, которое равно 1 если одно из двух слагаемых истинно
2. поскольку произведения включают много переменных, можно предположить, что они равны 1 в небольшом числе случаев, поэтому мы попытаемся найти количество решений «обратного» уравнения

**** (\*)

а потом вычесть это число из общего количества комбинаций значений переменных K, L, M, N (для четырех логических переменных, принимающих два значения (0 или 1), существует 24=16 различных комбинаций)

1. уравнение **** имеет два решения: требуется, чтобы **,** а  может принимать любые (логические) значения, то есть, 0 или 1; эти два решения – 1110 и 1111
2. уравнение **** такжеимеет два решения: требуется, чтобы **,** , а  может быть равно 0 или 1; эти два решения – 0001 и 1001
3. среди полученных четырех решений нет одинаковых, поэтому уравнение (\*) имеет 4 решения
4. это значит, что исходное уравнение истинно для всех остальных 16-4=12 комбинаций переменных K, L, M, N
5. таким образом, правильный ответ – 12.

|  |
| --- |
| **Возможные проблемы**:   * + не всегда удается догадаться, что неверных комбинаций меньше   + нужно проверять, что среди найденных решений нет одинаковых |

### Еще пример задания:

**P-03**. *Каково наибольшее целое положительное число* **X***, при котором истинно высказывание:*

**(X·(X + 3) > X·X + 7)** → **(X·(X + 2) ≤ X·X + 11)**

**Решение (преобразование выражений):**

1. несмотря на страшный вид, эта задача решается очень просто; сначала раскроем скобки в обеих частях импликации:

**(X·X + 3·X > X·X + 7)** → **(X·X + 2·X ≤ X·X + 11)**

1. теперь в каждой части вычтем **X·X** из обеих частей неравенства:

**(3·X > 7)** → **(2·X ≤ 11)**

1. в целых числах это равносильно:

**(X ≥ 3)** → **(X ≤ 5)**

1. вспомним, как раскрывается импликация через операции ИЛИ и НЕ: 
2. учитывая, что , имеем , следовательно

**(X < 3) или (X ≤ 5)**

1. это равносильно высказыванию **(X ≤ 5)**
2. таким образом, ответ – 5.

### Еще пример задания:

**P-02**. *Сколько различных решений имеет уравнение*

**¬((J**→**K)** →**(M****∧****N)) ∨ ¬((M****∧****N)** → **(¬J****∨****K)) ∨ (M****∧****N****∧****K****∧****L) = 0**

*где J, K, L, M, N – логические переменные? В ответе не нужно перечислять все различные наборы значений J, K, L, M и N, при которых выполнено данное равенство. В качестве ответа Вам нужно указать количество таких наборов.*

**Решение (вариант 1, упрощение выражения):**

1. перепишем уравнение, используя более простые обозначения операций:

****

1. логическая сумма трех слагаемых равна нулю, поэтому каждое из них должно быть равно нулю
2. обозначим сумму двух первых слагаемых через **** и попытаемся «свернуть» ее; для этого представим импликацию в виде ****, тогда

****

1. выполним замены и , тогда

****

1. раскроем импликацию через «ИЛИ» и «НЕ» (****):

****

1. теперь применим формулу де Моргана **:**

****

1. заметим, что в третьем слагаемом  тоже есть сомножитель , поэтому уравнение можно переписать в виде



или



1. это равенство выполняется, тогда и только тогда, когда оба слагаемых равны нулю;
2. учитывая, что в первом слагаемом есть сомножитель , а во втором –, это может быть в двух случаях:
3. – любое (0 или 1)
4. 
5. рассмотрим случай «а»: условию  удовлетворяют 3 пары (M,N): (0,0), (0,1) и (1,0); из условия  сразу получаем, что  и ; учитывая, что – любое (0 или 1), в случае «а» получаем 6 разных решений;
6. в случае «б» условие  сразу дает ; преобразуем второе условие с помощью формулы де Моргана:



это значит, что при  получаем  и – любое (2 решения), а при  имеем  и – любое (еще 2 решения)

1. проверяем, что все решения разные, поэтому всего найдено 6 + 2 + 2 = 10 решений
2. ответ – 10.

**Решение (вариант 2, использование свойств импликации):**

1. выполнив шаги 1-4 из первого варианта решения, получим

****

при заменах и 

1. поскольку нужно, чтобы ****, оба слагаемых равны нулю, то есть, обе импликации истинны:  и 
2. отсюда по таблице истинности операции «импликация» находим, что это может быть в двух случаях:
3. – любое (0 или 1)
4. 
5. рассмотрим случай «а»: условию  удовлетворяют 3 пары (M,N): (0,0), (0,1) и (1,0); из условия  сразу получаем, что  и ; учитывая, что – любое (0 или 1), в случае «а» получаем 6 разных решений;
6. в случае «б» условие  сразу дает ; преобразуем второе условие с помощью формулы де Моргана и перепишем третье:

, 

это значит, что при  получаем  и – любое (2 решения), а при  имеем  и – любое (еще 2 решения)

1. проверяем, что все решения разные, поэтому всего найдено 6 + 2 + 2 = 10 решений
2. ответ – 10.

|  |
| --- |
| **Возможные проблемы**:   * + это уравнение требует достаточно сложных преобразований; если вы не уверены в своих теоретических знаниях, лучше составить таблицу истинности (для 5 переменных в ней будет 32 строки) и аккуратно подставить все возможные комбинации переменных   + не всегда удается найти («увидеть») закономерности, позволяющие упростить выражение   + нужно проверять, чтобы среди найденных решений не было одинаковых |

### Еще пример задания:

**P-01**. *Сколько различных решений имеет уравнение*

**((J**→**K)** →**(M****∧****N****∧****L)) ∧ ((M****∧****N****∧****L)** → **(¬J****∨****K)) ∧ (M**→**J) = 1**

*где J, K, L, M, N – логические переменные? В ответе не нужно перечислять все различные наборы значений J, K, L, M и N, при которых выполнено данное равенство. В качестве ответа Вам нужно указать количество таких наборов.*

**Решение (вариант 1, использование свойств импликации):**

1. перепишем уравнение, используя более простые обозначения операций:

****

1. логическое произведение трех сомножителей равно единице, поэтому каждый из них должен быть тоже равен единице
2. учитывая, что **,** и выполняя замены и , получаем

**.**

1. рассмотрим последнюю импликацию, которая должна быть равна 1: ; по таблице истинности импликации сразу находим, что возможны три варианта:
2. 
3. 
4. 
5. поскольку все (в том числе и первые две) импликации должны быть равны 1, по таблице истинности импликации сразу определяем, что , то есть



1. в случае «а» последнее уравнение превращается в  и не имеет решений
2. в случае «б» имеем , тогда как  и  – произвольные; поэтому есть 4 решения, соответствующие четырем комбинациям  и 
3. в случае «в» получаем , то есть для  есть единственное решение (), а для  – три решения (при ; и ;  и )
4. проверяем, что среди решений, полученных в п. 7 и 8 нет одинаковых
5. таким образом, всего есть 4 + 1 + 3 = 8 решений
6. ответ – 8

**Решение (вариант 2, использование свойств импликации, А.М. Фридлянд, УГАТУ):**

1. перепишем уравнение, используя более простые обозначения операций:

****

1. логическое произведение трех сомножителей равно единице, поэтому каждый из них должен быть тоже равен единице
2. учитывая, что **,** и выполняя замены и , получаем

**.**

1. преобразуем первые две скобки: , где знак  означает операцию «эквивалентность». Так как это выражение должно быть истинным, значения  и  совпадают. Поэтому исходное уравнение распадается на 2 случая:
2. 
3. 
4. В случае *а*) из первого уравнения сразу получаем, что . Тогда третье уравнение справедливо при любом , а второе имеет 7 решений (любое, кроме ).
5. в случае *б*) из второго уравнения получаем: , но тогда из третьего уравнения следует, что  (иначе ), а тогда и  (иначе ).
6. таким образом, всего есть 7 + 1 = 8 решений
7. ответ – 8

**Решение (вариант 3, декомпозиция, автор идеи – А. Сидоров, ЭПИ МИСИС):**

1. перепишем уравнение, используя более простые обозначения операций:

****

1. идея заключается в том, что мы выбираем одну какую-нибудь переменную и отдельно рассматриваем случаи, когда она равна 0 и 1; такой подход, когда большая задача разбивается на несколько более простых, называют *декомпозицией*
2. логическое произведение трех сомножителей равно единице, поэтому каждый из них должен быть тоже равен единице
3. например, пусть ; тогда требуется, чтобы , по таблице истинности импликации получается, что при этом  может быть любое («из лжи следует что угодно»);
4. выполним второй шаг декомпозиции: рассмотрим отдельно варианты  и 
5. при  и  получаем

****

это равенство истинно, если , а такого не может быть, то есть в этом случае решений нет

1. при  и  получаем

****

это равенство истинно только при  (иначе первая скобка равна нулю), но у нас никак не ограничены значения  и  поэтому получается, что при  и  есть 4 решения (при  и всех 4-х различных комбинациях  и )

1. теперь проверяем вариант, когда ; при этом

****

так как должно быть , по таблице истинности операции импликация сразу получаем  и уравнение преобразуется к виду

****

1. выполним второй шаг декомпозиции: рассмотрим отдельно варианты  и 
2. при  получаем , откуда сразу следует, что  (3 решения: ;  и )
3. при  получаем , откуда сразу следует, что  (1 решение: )
4. таким образом, уравнение всего имеет 4+3+1 = 8 решений
5. ответ – 8

**Решение (вариант 4, декомпозиция, автор идеи – А. Сидоров, ЭПИ МИСИС):**

1. та же декомпозиция, но в другом порядке
2. сделаем сначала декомпозицию по 
3. рассмотрим вариант, когда ; подставляя это значение в уравнение

****

получаем

****

1. учитывая, что  при любом  («из лжи следует все, что угодно»), находим

****

1. отсюда сразу следует, что  и по таблице истинности операции импликация определяем, что ; учитывая это, получаем



этого не может быть, потому что первая скобка равна нулю; поэтому при  решений нет

1. теперь пусть , тогда  и , поэтому остается уравнение



1. выполним декомпозицию по переменной 
2. при  получаем , что верно при условии ; из всех 8-ми комбинаций значений переменных ,  и  только одна этому условию не удовлетворяет (), поэтому имеем 7 решений
3. при  получаем , что верно при условии ; из 8-ми комбинаций значений переменных ,  и  только одна () удовлетворяет этому условию, поэтому имеем 1 решение
4. таким образом, уравнение всего имеет 7+1 = 8 решений
5. ответ – 8

|  |
| --- |
| **Возможные проблемы**:   * + при использовании метода декомпозиции важен порядок выбора переменных для разбиения; можно рекомендовать в первую очередь делать декомпозицию по той переменной, которая чаще всего встречается в уравнении   + нужно помнить, что импликация равна нулю только в случае , часто именно это свойство позволяет упростить решение |

**Решение (вариант 5, комбинированный, Т.Н. Наумова, ХМАО, Пыть-Ях, МОУ СОШ №5):**

1. перепишем уравнение, используя более простые обозначения операций:

****

1. имеем логическое произведение трех выражений, которое истинно тогда и только тогда, когда каждое выражение истинно; таким образом, нужно решить систему логических уравнений

****

1. идея состоит в том, чтобы найти все решения одного из уравнений и проверить истинность остальных двух для всех полученных на предыдущем шаге комбинаций значений переменных
2. рассмотрим первое уравнение: ****; оно справедливо в двух случаях:
   1. , – любое, или , где звездочка означает, что переменная может принимать значения 0 или 1; всего получается **8 вариантов**
   2. , , что дает ещё **три варианта**:   
       – два варианта

 – один вариант

1. остается проверить истинность второго (**)**и третьего (****) равенств для этих 11 вариантов; сразу видим, что импликация  ложна только тогда, когда , то есть для комбинации (10111), а импликация  ложна для  при любых значениях остальных переменных:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **J** | **K** | **L** | **M** | **N** |  |  |  |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | ✓ |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | ✓ |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 | ✓ |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | ✓ |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | ✓ |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | ✓ |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 | ✓ |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 |  |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 |  |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |  |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | ✓ |

1. таким образом, остается 8 вариантов, отмеченных галочками справа от таблицы
2. ответ – 8

### Еще пример задания:

**P-00**. *Сколько различных решений имеет уравнение*

**((J** → **K)** → **(M ∧ N ∧ L)) ∧ ((J ∧ ¬K)**→ **¬(M ∧ N ∧ L)) ∧ (M** → **J)=** **1**

*где J, K, L, M, N – логические переменные? В ответе не нужно перечислять все различные наборы значений J, K, L, M и N, при которых выполнено данное равенство. В качестве ответа Вам нужно указать количество таких наборов.*

**Решение (вариант 1, упрощение выражения):**

1. перепишем уравнение, используя более простые обозначения операций:

****

1. попытаемся использовать замену переменных

****

1. тогда****
2. с учетом этих обозначений преобразуем исходное уравнение к виду:

****

1. раскрываем импликации по правилу ****:

****

1. перемножаем первые две скобки, учитывая, что ****:

****

1. снова раскрываем скобки

****

1. возвращаемся к исходным переменным, вспоминая, что ****

****

1. далее используем равенства  и , два слагаемых обращаются в нуль:

****

1. выносим общий множитель из первых двух слагаемых, в скобках остается выражение ****

****

****

1. такие образом, уравнение разбивается на два:

 (\*)

**** (\*\*)

1. из уравнения  следует, что  и хотя бы одна из переменных  не равна 1; поэтому уравнение (\*) имеет 7 решений (за исключением случая )
2. уравнение (\*\*) имеет единственное решение 
3. среди решений уравнений (\*) и (\*\*) нет одинаковых (в первом случае , а во втором - ), поэтому исходное уравнение имеет 7 + 1 = 8 решений.
4. ответ – 8.

# 24 (C1) (повышенный уровень, время – 30 мин)

**Тема**: Исправление ошибок в простой программе с условными операторами.

**Что нужно знать**:

* правила построения программы на Паскале[[27]](#footnote-27), Бэйсике или Си
* правила работы с переменными (объявление, ввод, вывод, оператор присваивания)
* *ветвление* – это выбор одного из двух возможных вариантов действий в зависимости от того, выполняется ли некоторое условие;
* на блок-схеме алгоритма ветвление изображается в виде блока-ромба с одним входом и двумя выходами:

**a = b?**

**нет**

**да**

**полная форма ветвления**

**блок-1**

**блок-2**

Рисунок 1

**a = b?**

**нет**

**да**

**неполная форма ветвления**

**блок-1**

Рисунок 2

* если условие, записанное внутри ромба, истинно (ответ «да» на вопрос «**a=b?**»), выполняются все команды, входящие в **блок-1** (ветка «да»), иначе (если условие ложно) выполняются все команды в **блоке-2** (ветка «нет»)
* в неполной форме условного оператора **блок-2** пустой (отсутствует); теоретически можно сделать наоборот – так, чтобы **блок-1** оказался пустой, но это очень неграмотное решение, поскольку оно осложняет понимание алгоритма, запутывает его
* одна команда ветвления может находиться внутри другой, например, так:

**a = b?**

**нет**

**да**

**блок-10**

**блок-2**

**a = c?**

**да**

**блок-11**

**блок-12**

**нет**

Рисунок 3

* на этой схеме **блок-10** выполняется, когда **a=b**; **блок-11** – когда **a=b=c**, **блок-12** – когда **a=b**, но **a≠c** и, наконец, **блок-2** – когда **a≠b**
* на этой схеме (Рисунок 3) одна команда ветвления (с условием «**a=c**») вложена в другую (с условием «**a=b**»), каждая из них – это ветвление в полной форме; если блок-12 будет пустой (отсутствует), внутреннее ветвление имеет неполную форму; аналогично, если блок-2 пустой, то внешнее ветвление имеет неполную форму
* *условный оператор* **if–else** служит для организации ветвления в программе на языке Паскаль
* условный оператор может иметь полную или неполную форму; вот фрагменты программы, реализующие ветвления, показанные на рисунках 1 и 2:

|  |  |
| --- | --- |
| *полная форма:* | *неполная форма:* |
| **if a = b then begin**  **{ блок-1 }**  **end**  **else begin**  **{ блок-2 }**  **end;** | **if a = b then begin**  **{ блок-1 }**  **end;** |

здесь вместо комментариев в фигурных скобках (они выделены синим цветом) могут стоять любые операторы языка программирования (в том числе операторы присваивания, другие условные операторы, циклы, вызовы процедур и т.п.)

* обычно при записи программы операторы, находящиеся внутри обоих блоков, сдвигают вправо на 2-3 символа (запись «лесенкой»), это позволяет сразу видеть начало и конец блока (конечно, если «лесенка» сделана правильно)
* после **else** не надо (нельзя!) ставить какое-то условие, эта часть выполняется тогда, когда условие после **if** неверно (частая ошибка – после **else** пытаются написать условие, обратное тому, которое стоит после соответствующего ему **if**)
* в Паскале перед **else** не ставится точка с запятой, поскольку это ключевое слово обозначает не начало нового оператора, а вторую часть условного оператора **if–else**
* слова **begin** и **end** (их называют также «операторные скобки») ограничивают **блок-1** и **блок-2**; если внутри блока всего один оператор, эти «скобки» можно не писать, например, допустимы такие операторы

|  |  |  |
| --- | --- | --- |
| **if a = b then**  **c:=1**  **else c:=0;** | **if a = b then begin**  **c:=1;**  **end**  **else c:=0;** | **if a = b then c:=1;** |

* а вот такие операторы недопустимы

|  |  |  |
| --- | --- | --- |
| **if a = b then begin**  **c:=1**  **else c:=0;** | **if a = b then**  **c:=1;**  **end**  **else c:=0;** | **if a = b then**  **c:=1;**  **d:=1;**  **else x:=1;** |

* в первом случае есть **begin**, но забыли про соответствующий ему **end**;
* во втором фрагменте наоборот, есть **end**, а **begin** отсутствует;

|  |
| --- |
| **if a = b then begin**  **c:=1;**  **d:=1;**  **end**  **else x:=1;** |

* третий случай более сложный: судя по записи «лесенкой», здесь внутри блока-1 находятся 2 оператора, а операторных скобок **begin-end** нет; в результате получилось, что оператор **c:=1** находится внутри блока-1, он выполняется только при условии **a=b**;   
  оператор **d:=1** выполняется всегда, после того, как условный оператор закончил работу; а **else** вообще «висит» непонятно как, тут транслятор выдаст ошибку; исправить эту программу можно так, как показано справа (добавив пару **begin-end**):
* условный оператор может находиться внутри другого условного оператора, как в блоке-1, так и в блоке-2; например, схема на Рисунке 3 может быть записана на Паскале так:

|  |
| --- |
| **if a = b then begin**  **{ блок-10 }**  **if a = c then begin**  **{ блок-11 }**  **end**  **else begin**  **{ блок-12 }**  **end;**  **if a = c then begin**  **{ блок-11 }**  **end**  **else begin**  **{ блок-12 }**  **end;**  **end**  **else begin**  **{ блок-2 }**  **end;** |

* ключевая тема этого задания ЕГЭ – использование вложенных условных операторов, причем в тексте задания фрагмент программы обычно записан без отступов «лесенкой» или с неправильными отступами, например, так:

|  |  |
| --- | --- |
| **if a = b then begin**  **if a = c then**  **c:=1;**  **end**  **else c:=0;** | **if a = b then**  **if a = c then**  **c:=1**  **else c:=0;** |

Чтобы разобраться с работой этих программ, нужно определить, к какому из условных операторов **if** относится часть **else**; для этого используют такое правило: «любой **else** относится к ближайшему **if**».

Рассмотрим фрагмент слева, в нем перед **else** стоит **end**, поэтому для него нужно найти соответствующий ему **begin**; таким образом определяем, что **else** относится к первому (внешнему) условному оператору.

В правом фрагменте перед **else** нет **end**, поэтому он относится к ближайшему по тексту внутреннему условному оператору.

Блок-схемы для двух фрагментов показаны ниже, желтым цветом выделен «переехавший» блок:

**a = b?**

**нет**

**да**

**c:=0;**

**a = c?**

**да**

**c:=1;**

**нет**

Рисунок 4

**a = b?**

**нет**

**да**

**a = c?**

**да**

**c:=1;**

**нет**

Рисунок 5

**c:=0;**

* в условных операторах можно использовать сложные условия, которые строятся из простых отношений (**<, <=, >, >=, =, <>**) с помощью логических операций **not** («НЕ», отрицание), **and** («И», одновременное выполнение двух условий) и **or** («ИЛИ», выполнение хотя бы одного из двух условий)
* в сложном условии сначала выполняются действия в скобках, потом – **not**, затем – **and**, затем – **or** и, наконец, отношения;   
  операции равного уровня (приоритета) выполняются последовательно слева направо
* поскольку отношения в Паскале имеют низший приоритет, в сложном условии их приходится брать в скобки:

|  |
| --- |
| **if (a = b) or (b < c) and (c <> d) then begin**  **...**  **end;** |

* в приведенном выше примере сначала определяются результаты сравнения (выражения в скобках), затем выполняется операция **and** («И»), а затем – **or** («ИЛИ»)

### Пример задания:

*На обработку поступает последовательность из четырёх неотрицательных целых чисел (некоторые числа могут быть одинаковыми). Нужно написать программу, которая выводит на экран количество нечётных чисел в исходной последовательности и максимальное нечётное число. Если нечётных чисел нет, требуется на экран вывести «NO». Известно, что вводимые числа не превышают 1000. Программист написал программу неправильно. Вот она:*

**const n = 4;**

**var i, x: integer;**

**var maximum, count: integer;**

**begin**

**count := 0;**

**maximum := 999;**

**for i := 1 to n do begin**

**read(x);**

**if x mod 2 <> 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.**

*Последовательно выполните следующее.*

*1. Напишите, что выведет эта программа при вводе последовательности: 2 9 4 3*

*2. Приведите пример такой последовательности, содержащей хотя бы одно нечётное число, что, несмотря на ошибки, программа печатает 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. Обратите внимание, что требуется найти ошибки в имеющейся программе, а не написать свою, возможно, использующую другой алгоритм решения. Исправление ошибки должно затрагивать только строку, в которой находится ошибка.*

**Решение:**

1. обратим внимание на две строки в начале программы, которые начинаются с ключевого слова var: **это не ошибка**, такое повторение, действительно, допустимо в языке Паскаль; возможно, это была одна из ловушек разработчиков КИМ, которую они применили на реальном ЕГЭ-2014
2. теперь выполним программу для заданной последовательности 2 9 4 3, записывая все изменения переменных в таблицу:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | условие | i | x | maximum | count | вывод |
| count:=0; |  |  |  |  | 0 |  |
| maximum:=999; |  |  |  | 999 |  |  |
| **for i := 1 to n do** |  | 1 |  |  |  |  |
| **read(x);** |  |  | 2 |  |  |  |
| **if x mod 2 <> 0 then** | нет |  |  |  |  |  |
| **end** |  | 2 |  |  |  |  |
| **read(x);** |  |  | 9 |  |  |  |
| **if x mod 2 <> 0 then** | да |  |  |  |  |  |
| **count:=count+1;** |  |  |  |  | 1 |  |
| **if x > maximum then** | нет |  |  |  |  |  |
| **end** |  | 3 |  |  |  |  |
| **read(x);** |  |  | 4 |  |  |  |
| **if x mod 2 <> 0 then** | нет |  |  |  |  |  |
| **end** |  | 4 |  |  |  |  |
| **read(x);** |  |  | 3 |  |  |  |
| **if x mod 2 <> 0 then** | да |  |  |  |  |  |
| **count:=count+1;** |  |  |  |  | 2 |  |
| **if x > maximum then** | нет |  |  |  |  |  |
| end |  | 5 |  |  |  |  |
| **if count > 0 then** | да |  |  |  |  |  |
| **writeln(count);** |  |  |  |  |  | 2 |
| **writeln(maximum)** |  |  |  |  |  | 999 |

1. при ручной прокрутке программы мы увидели, что она правильно подсчитала количество нечётных чисел во входной последовательности, но неверно определила максимум: значение переменной maximum, которое было выведено на экран, осталось равным начальному значению 999, так как все остальные нечётные числа были меньше этого начального значения; поэтому ответ на п. 1 задания должен быть таким:
2. Программа выведет числа 2 и 999.
3. поскольку все числа по условию неотрицательны и не превышают 1000, программа всегда будет выдавать 999 вместо максимального нечётного числа; в то же время мы выяснили, что количество нечётных чисел в последовательности считается правильно; поэтому любая последовательность, содержащая 999, будет обрабатываться правильно
4. таким образом, правильный ответ на п. 2 должен быть таким:
5. Программа работает правильно для последовательности: 2 9 3 999.
6. теперь будем искать ошибки; как уже отмечалось, повторное использование ключевого слова var допустимо и указывать это в качестве ошибки нельзя!
7. как следует из результатов ручной прокрутки программы, во многих случаях она выдаёт неверный результат из-за того, что неверно задано начальное значение переменной maximum: оно должно быть **меньше**, чем любой возможный результат;
8. наименьшее нечётное неотрицательное число – это 1, поэтому можно принять в качестве начального значения maximum любое число, меньшее единицы (на самом деле, программа будет правильно работать и для 1), например:
9. **Ошибка 1**. maximum:=999;

Исправление: maximum:=0;

1. если теперь (с исправленной первой ошибкой) сделать ручную прокрутку программы, то мы увидим, что на последовательности 2 9 4 3 она выдает сначала 2, а потом – 4, то есть, значение максимума вычисляется опять наверно;
2. откуда появится число 4 в переменной maximum? оно будет записана в результате выполнения оператора **if x > maximum then maximum:=i**, который записывает в переменную maximum не значение полученного числа (x), а его номер (i); таким образом, мы нашли вторую ошибку:

**Ошибка 2**. **if x > maximum then maximum:=i**

Исправление: **if x > maximum then maximum:=x**

### Ещё пример задания:

*Требовалось написать программу, которая вводит с клавиатуры натуральное число N, не превышающее 109, и выводит сумму чётных цифр в десятичной записи этого числа или 0, если чётных цифр нет. Программист торопился и написал программу неправильно. Вот она:*

**var N: longint;**

**s: integer;**

**begin**

**readln(N); (1)**

**s := 0; (2)**

**while N > 1 do begin (3)**

**if N mod 2 = 0 then begin (4)**

**s := N mod 10; (5)**

**end; (6)**

**N := N div 10; (7)**

**end; (8)**

**write(s); (9)**

**end.**

*Последовательно выполните следующее.*

*1. Напишите, что выведет эта программа при вводе числа 1984.*

*2. Приведите пример числа, при вводе которого программа выдаст верный ответ.*

*3. Найдите в программе все ошибки (их может быть одна или несколько).*

*Для каждой ошибки выпишите строку, в которой она допущена, и приведите эту же строку в исправленном виде.*

*Обратите внимание: вам нужно исправить приведённую программу, а не написать свою. Вы можете только заменять ошибочные строки, но не можете удалять строки или добавлять новые. Заменять следует только ошибочные строки: за исправления, внесённые в строки, не содержащие ошибок, баллы будут снижаться.*

**Решение:**

1. начнем с того, что разберёмся в условии задачи: ограничение N < 109 введено для того, чтобы число поместилось в 4-байтовую переменную целого типа, для совместимости со всеми трансляторами Паскаля соответствующая переменная объявлена как longint
2. теперь разберём программу; строка (1) – это ввод исходного числа; очевидно, что переменная s – это результат, поскольку именно она выводится в строке (9)
3. в строке (2) переменная s обнуляется, это естественно при накоплении суммы
4. строки (3)-(8) – это цикл, который выполняется пока N > 1; на каждом шаге цикла N делится на 10 нацело, то есть, из десятичной записи числа отбрасывается последняя цифра (строка (7)); такой цикл используется для того, чтобы перебрать все цифры числа, но обычно ставят условие «N > 0», поскольку в приведенном варианте цикл остановится при N = 1; однако в данном случае это **не влияет на результат**, поскольку по условию нас интересуют только чётные цифры, а 1 – нечётная
5. в строке (4) проверяется чётность числа (и одновременно чётность его последней цифры!), если число чётное, в строке (5) в переменную s записывается остаток от деления числа на 10, то есть последняя цифра десятичной записи этого числа
6. таким образом, после выполнения цикла будет выведена последняя рассмотренная цифра, для которой сработает условие в строке (4)
7. поскольку цифры перебираются с конца, выводится первая чётная цифра в записи числа
8. начнём выполнять задание:

1. При вводе числа 1984 будет выведено число 8 – значение первой чётной цифры числа.

1. когда программа выдаст верный ответ? очевидно, что тогда, когда сумма чётных цифр и значение первой чётной цифры совпадают; это возможно, если в числе

* нет чётных цифр (сумма останется равной 0)
* все чётные цифры – нули
* одна чётная цифра
* одна ненулевая чётная цифра, а нули стоят после неё (нули не меняют суммы!)

2. Программа выдаст правильный ответ для N = 1981.

1. как исправить программы? очевидно, нужно, чтобы она считала сумму чётных цифр, то есть, получив очередную чётную цифру, нужно добавить её к «старому» значению переменно s, изменив строку (5) так:

**s := s + N mod 10; (5)**

3. Ошибка допущена в строке

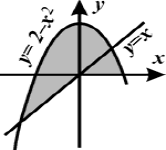
**s := N mod 10;**

Эта строка должна в правильной программе выглядеть так

**s := s + N mod 10;**

### Ещё пример задания:

*Требовалось написать программу, которая вводит с клавиатуры координаты точки на плоскости (***x***,* **y** *– действительные числа) и определяет принадлежность точки заштрихованной области, включая ее границы. Программист торопился и написал программу неправильно. Вот она:*



**var x,y: real;**

**begin**

**readln(x,y);**

**if y>=x then**

**if y>=0 then**

**if y<=2-x\*x then**

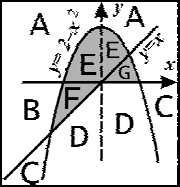
**write('принадлежит')**

**else**

**write('не принадлежит')**

**end.**

*Последовательно выполните следующее:*



*1. Перерисуйте и заполните таблицу, которая показывает, как работает программа при аргументах, принадлежащих различным областям (A, B, C, D, E, F и G). Точки, лежащие на границах областей, отдельно не рассматривать.*

*В столбцах условий укажите "да", если условие выполнится, "нет" если условие не выполнится, "—" (прочерк), если условие не будет проверяться, «не изв.», если программа ведет себя по-разному для разных значений, принадлежащих данной области. В столбце "Программа выведет" укажите, что программа выведет на экран. Если программа ничего не выводит, напишите "—" (прочерк). Если для разных значений, принадлежащих области, будут выведены разные тексты, напишите «не изв». В последнем столбце укажите "да" или "нет".*

*2. Укажите, как нужно доработать программу, чтобы не было случаев ее неправильной работы. (Это можно сделать несколькими способами, поэтому можно указать любой способ доработки исходной программы) .*

**Решение:**

1. начнем заполнять таблицу, выписывая истинность каждого из трёх условий
2. условие y>=x истинно выше прямой y=x, то есть в областях A, B, E, F

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Область** | **y>=x?** | **y>=0?** | **y<=2-x\*x?** | **вывод** | **верно?** |
| **A** | **да** |  |  |  |  |
| **B** | **да** |  |  |  |  |
| **C** | **нет** |  |  |  |  |
| **D** | **нет** |  |  |  |  |
| **E** | **да** |  |  |  |  |
| **F** | **да** |  |  |  |  |
| **G** | **нет** |  |  |  |  |

1. условие y>=0 истинно выше прямой y=0, то есть в областях A, E, G, однако **это условие проверяется только тогда, когда первое условие, y>=x, истинно**; поэтому для всех областей, где первое условие неверно (это области C, D, G), сразу в столбце второго условия ставим прочерк (условие не будет проверяться)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Область** | **y>=x?** | **y>=0?** | **y<=2-x\*x?** | **вывод** | **верно?** |
| **A** | **да** | **да** |  |  |  |
| **B** | **да** | **нет** |  |  |  |
| **C** | **нет** | **–** |  |  |  |
| **D** | **нет** | **–** |  |  |  |
| **E** | **да** | **да** |  |  |  |
| **F** | **да** | **нет** |  |  |  |
| **G** | **нет** | **–** |  |  |  |

1. третье условие выполняется для областей «внутри» параболы, то есть для E, F, G, D; однако оно проверяется только тогда, когда первые два истинны (для А и Е), в остальных строках ставим прочерк:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Область** | **y>=x?** | **y>=0?** | **y<=2-x\*x?** | **вывод** | **верно?** |
| **A** | **да** | **да** | **нет** |  |  |
| **B** | **да** | **нет** | **–** |  |  |
| **C** | **нет** | **–** | **–** |  |  |
| **D** | **нет** | **–** | **–** |  |  |
| **E** | **да** | **да** | **да** |  |  |
| **F** | **да** | **нет** | **–** |  |  |
| **G** | **нет** | **–** | **–** |  |  |

1. как следует из текста программы, она выведет что-то на экран только в том случае, когда выполняются первые два условия и программа выходит на третье: для области А будет выведено «не принадлежит», для области Е – «принадлежит», именно в этих двух случаях программа работает правильно, в остальных – нет:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Область** | **y>=x?** | **y>=0?** | **y<=2-x\*x?** | **вывод** | **верно?** |
| **A** | **да** | **да** | **нет** | **не принадлежит** | **да** |
| **B** | **да** | **нет** | **–** | **–** | **нет** |
| **C** | **нет** | **–** | **–** | **–** | **нет** |
| **D** | **нет** | **–** | **–** | **–** | **нет** |
| **E** | **да** | **да** | **да** | **принадлежит** | **да** |
| **F** | **да** | **нет** | **–** | **–** | **нет** |
| **G** | **нет** | **–** | **–** | **–** | **нет** |

1. для того, чтобы доработать программу, проще всего составить одно сложное условие, описывающее всю заштрихованную область
2. в данном случае удобно представить данную область в виде объединения областей, первая из которых включает области E+G, а вторая – области E+F
3. область E+G соответствует условию (y>=0) and (y <=2-x\*x)
4. область E+F соответствует условию (y>=x) and (y <=2-x\*x)
5. объединение областей выполняется с помощью операции ИЛИ (or), так что полное условие принимает вид

(y>=0) and (y <=2-x\*x) or (y>=x) and (y <=2-x\*x)

поскольку операция И (and) имеет более высокий приоритет, чем ИЛИ (or), порядок выполнения операций тут правильный; в случае сомнений можно поставить дополнительные скобки:

((y>=0) and (y <=2-x\*x)) or ((y>=x) and (y <=2-x\*x))

1. поскольку в обоих условиях есть условие y <=2-x\*x, запись можно немного сократить:

(y <=2-x\*x) and ((y>=x) or (y>=0))

1. доработанная программа выглядит так:

**var x,y: real;**

**begin**

**readln(x,y);**

**if** (y <=2-x\*x) and ((y>=x) or (y>=0)) **then**

**write('принадлежит')**

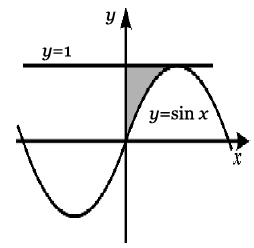
**else**

**write('не принадлежит')**

**end.**

### Ещё пример задания:

*Требовалось написать программу, которая вводит с клавиатуры координаты точки на плоскости (***x***,* **y** *– действительные числа) и определяет принадлежность точки заштрихованной области, включая ее границы. Программист торопился и написал программу неправильно. Вот она:*



**var x,y: real;**

**begin**

**readln(x,y);**

**if y <= 1 then**

**if x >= 0 then**

**if y >= sin(x) then**

**write('принадлежит')**

**else write('не принадлежит')**

**end.**

*Последовательно выполните следующее: 1) Приведите пример таких чисел* **x***,* **y***, при которых программа неверно решает поставленную задачу. 2) Укажите, как нужно доработать программу, чтобы не было случаев ее неправильной работы. (Это можно сделать несколькими способами, поэтому можно указать любой способ доработки исходной программы) .*

**Решение:**

1. сначала лучше отложить в сторону программу и попытаться написать условие, которым должны отвечать точки, попавшие в выделенную область
2. заштрихованная область ограничена по координате , она находится

* справа от оси , что равносильно **условию**  (с учетом границы здесь и далее получаем нестрогие неравенства)
* слева от первого максимума функции ; из математики мы знаем, что эта функция достигает максимума при , поэтому получаем **второе условие** 

1. заштрихованная область ограничена с двух сторон по координате : она находится

* ниже линии , откуда следует **третье условие** 
* выше линии , что дает **четвертое условие** 

1. итак, точка находится в заданной области, если все эти четыре условия выполняются одновременно; можно предположить, что в программе нужно использовать четыре вложенных условных оператора или один условный оператор, в котором четыре простых условия (отношения , ,  и ) связаны с помощью логической операции **and** («И», одновременное выполнение всех условий)
2. теперь смотрим на программу: здесь три (а не четыре!) вложенных условных оператора с простыми отношениями, поэтому явно какое-то условие не учтено; легко найти, что «забыли» условие 
3. оператор **write('принадлежит')** помещен внутрь всех трех условных операторов, то есть, он выполнится тогда, когда три (а не четыре!) условия истинны;











1. отметим на рисунке область, где выполняются все нужные условия, кроме  (красная зона);
2. для всех точек, которые находятся в «красной» зоне программа выдаст сообщение «принадлежит», хотя в самом деле эти точки не принадлежит заданной области; одна из таких точек имеет координаты 
3. теперь выясним, когда программа выдает сообщение «не принадлежит»

**if y <= 1 then**

**if x >= 0 then**

**if y >= sin(x) then**

**write('принадлежит')**

**else write('не принадлежит')**

1. судя по записи «лесенкой», **else** относится к самому первому оператору **if**, однако в самом деле это не так; перед словом **else** нет **end**, поэтому ищем ближайший **if**: это самый внутренний оператор, правильная запись «лесенкой» выглядит так:

**if y <= 1 then**

**if x >= 0 then**

**if y >= sin(x) then**

**write('принадлежит')**

**else write('не принадлежит')**

1. этот фрагмент программы соответствует блок-схеме, которая  
     
   показана на рисунке справа:

**да**

**нет**

**принадлежит**

**y >= sin(x)**

**да**

**x >= 0**

**да**

**y <= 1**

**не принадлежит**

**нет**

**нет**

1. по схеме видим, что при  (первое условие ложно),  
   а также при  (второе условие ложно) программа  
   вообще не выдает никакого сообщения, то есть,  
   работает неправильно; таким образом, координаты любой точки, для которой  или , могут быть указаны в ответе как пример  
   набора входных данных, при которых программа работает  
   неправильно
2. итак, первая часть ответа такова

примеры входных данных, на которых программа работает неверно:  
**(x=3.14, y=0.5)** (неправильно определяет принадлежность точки области)   
**(x=0, y=2)** или **(x=-1, y=0)** (не выдает вообще никакого сообщения)

1. остается исправить эту программу;   
   начнем с самого «лобового способа»: добавим в программу четвертый (вложенный) условный оператор, проверяющий условие , и еще три блока **else**, чтобы выводить строку «не принадлежит» в том случае, когда хотя бы один из них не сработал:

**if x <= pi/2 then**

**if y <= 1 then**

**if x >= 0 then**

**if y >= sin(x) then**

**write('принадлежит')**

**else write('не принадлежит')**

**else write('не принадлежит')**

**else write('не принадлежит')**

**else write('не принадлежит');**

обратите внимание, что точка с запятой есть только после самого последнего оператора **write**, так как остальные стоят перед ключевым словом **else**, перед которым точка с запятой не ставится

1. хотя приведенный выше метод дает работоспособную программу, она получается слишком длинная и некрасивая для такой простой задачи; достаточно сказать, что оператор   
   **write('не принадлежит')** повторяется в тексте 4 раза
2. более элегантное решение формулируется на словах так: «точка принадлежит области, если выполняются одновременно 4 приведенных выше условия, а иначе – не принадлежит»; а вот реализация на Паскале (приведем программу-ответ целиком):

**var x,y: real;**

**begin**

**readln(x,y);**

**if (x >= 0) and (x <= pi/2) and**

**(y <= 1) and (y >= sin(x)) then**

**write('принадлежит')**

**else write('не принадлежит');**

**end.**

здесь использовано сложное условие, в котором 4 отношения связаны операциями **and** («И», требуется одновременное выполнение всех условий)

|  |
| --- |
| **Возможные проблемы**:   * + как правило, в этой задаче требуется использовать знания из курса математики (решение уравнений, графики функций, область допустимых значений, составление уравнений прямой по приведенному графику)   + как показывает анализ опубликованных задач этого типа, нужно уметь, прежде всего, разбираться в серии вложенных условных операторов в полной и неполной форме   + неправильная «лесенка» в записи сбивает с толку и подталкивает к неверному решению; чтобы разобраться в программе, лучше на черновике построить блок-схему алгоритма и правильную «лесенку»   + чтобы не запутаться, к какому оператору относится **else**, используйте следующее правило: * если перед **else** нет слова **end**, нужно искать ближайший сверху условный оператор **if** * если перед **else** стоит **end** (конец блока), нужно искать парный ему **begin** (начало блока) и соответствующий условный оператор **if ... then begin**   + проверяйте, все ли необходимые условия учтены в программе, это особенно актуально для *немонотонных* функций типа синуса или косинуса (немонотонные функции на некоторых участках возрастают при увеличении аргумента, а на некоторых – убывают); например, в этой задаче можно пропустить необходимость выполнения условия   + не перепутайте, где нужно использовать операцию **and** («И», одновременное выполнение условий), а где – **or** («ИЛИ», хотя бы одно условие)   + нужно внимательно проверять, всегда ли программа выдает сообщение, если заданное условие не выполняется   + часто бывает полезно нарисовать блок-схему алгоритма, которая позволяет увидеть ход выполнения программы при всех возможных вариантах   + проверяйте, включает ли заданная область свои границы; если включает – в отношениях будут нестрогие неравенства (**<=**, **>=**), если не включает – строгие (**<**, **>**)   + при оценке работы можно (при абсолютно правильном решении) потерять баллы из-за синтаксических ошибок в программе (скобки, точки с запятой, неправильное написание оператора и т.п.); не забывайте, что * в сложном условии все простые условия (отношения) нужно брать в скобки, так как в Паскале отношения при вычислении логического выражения имеют самый низкий приоритет * перед **else** точка с запятой никогда не ставится * в конце программы после последнего **end** ставится точка |

|  |
| --- |
| **За что снимают баллы:**   * + неправильно определены входные данные, при которых исходная программа работает неверно   + исправлены не все ошибки в программе, например, легко «просмотреть», что необходимо еще условие   + программа работает правильно в большем количестве случаев, чем исходная, но не для всех возможных исходных данных   + перепутаны знаки < и >, логические операции **or** и **and**   + неверно расставлены операторные скобки **begin-end**   + синтаксические ошибки (знаки пунктуации – запятые, точки, точки с запятой; неверное написание ключевых слов); чтобы получить 3 балла, нужно при абсолютно правильном решении сделать не более одной синтаксической ошибки; на 2 балла – до двух ошибок, на 1 балл – до трех ошибок |

# 25 (C2) (высокий уровень, время – 30 мин)

**Тема**: Обработка массива (написать программу из 10-15 строк на языке программирования или алгоритм на естественном языке).

**Что нужно знать**:

* *массив* – это набор однотипных элементов, имеющих общее имя и расположенных в памяти рядом
* для обращения к элементу массива используют квадратные скобки, запись **A[i]** обозначает элемент массива **A** с номером (индексом) **i**
* для обработки всех элементов массива используется цикл вида[[28]](#footnote-28)

**for i:=1 to N do begin**

**{ что-то делаем с элементом A[i] }**

**end;**

переменная **i** обозначает номер текущего элемента массива, она меняется от 1 до N с шагом 1, то есть мы «проходим» последовательно все элементы

* *матрица* (двухмерный массив) – это прямоугольная таблица однотипных элементов
* если матрица имеет имя **A**, то обращение **A[i,k]** обозначает элемент, расположенный на пересечении строки **i** и столбца **k**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **k** |  |
|  |  |  |  |
|  |  |  |  |
| **i** |  | **A[i,k]** |  |
|  |  |  |  |

* каждая строка матрицы – это обычный (одномерный, линейный) массив; для того, чтобы обработать строку **i** в матрице из **M** столбцов, нужно использовать цикл, в котором меняется номер столбца **k**:

**for k:=1 to M do begin**

**{ что-то делаем с элементом A[i,k] }**

**end;**

* каждый столбец матрицы – это обычный (одномерный, линейный) массив; для того, чтобы обработать столбец **k** в матрице из **N** строк, нужно использовать цикл, в котором изменяется номер строки **i**:

**for i:=1 to N do begin**

**{ что-то делаем с элементом A[i,k] }**

**end;**

* условие задачи записано на нескольких языках (алгоритмический язык, Паскаль, Бейсик и Си); в принципе, решение можно писать и на любом другом языке, в том числе на естественном языке или в виде блок-схемы; но нужно помнить, что

Если вы пишете решение на языке, в котором есть встроенные функции для обработки массивов (списков), например, на Python, использовать эти функции НЕЛЬЗЯ; в первую очередь, это касается функций (методов) **min**, **max**, **sort**.

### Пример задания:

**Р-04**. *Дан целочисленный массив из 20 элементов. Элементы массива могут принимать целые значения от –10 000 до 10 000 включительно. Опишите на естественном языке или на одном из языков программирования алгоритм, позволяющий найти и вывести количество пар элементов массива, сумма которых нечётна и положительна. Под парой подразумевается два подряд идущих элемента массива.*

*Исходные данные объявлены так, как показано ниже. Запрещается использовать переменные, не описанные ниже, но использовать все описанные переменные не обязательно.*

|  |  |
| --- | --- |
| Паскаль | Алгоритмический язык |
| **const N = 20;**  **var a: array [1..N] of integer;**  **i, j, k: integer;**  **begin**  **for i := 1 to N do**  **readln(a[i]);**  **...**  **end.** | алг  нач  цел N = 20  цел таб a[1:N]  цел i, j, k  нц для i от 1 до N  ввод a[i]  кц  ...  кон |

**Решение:**

1. даже если вы хорошо владеете программированием, сначала лучше (прежде всего, для себя) написать алгоритм на русском языке
2. в задании нужно перебрать все пары соседних элементов, начиная от пары (a[1], a[2]) до пары (a[N-1],a[N]) и подсчитать, для скольких пар сумма элементов нечётна и положительна
3. поскольку нам нужно что-то считать, придётся использовать переменную счётчик; сначала её значение равно 0, затем, при каждой найденной подходящей паре, значение счётчика увеличивается на 1; в качестве счётчика можно использовать любую целую переменную из тех, что были объявлены в условии, например, k:

**k := 0;**

**перебрать все пары**

**если пара подходящая, то**

**k := k + 1**

1. как же перебрать все пары? примем, что переменная i будет хранить номер первого элемента в паре, то есть, будем рассматривать пары (a[i],a[i+1])
2. очевидно, что нужно организовать цикл, который перебирает все значения i в интервале от 1 (для первой пары, (a[1], a[2])) до N-1 (для последней пары, (a[N-1],a[N]))

**for i:=1 to N-1 do ...**

1. как определить, что пара (a[i],a[i+1]) подходящая? нужно вычислить её сумму a[i]+a[i+1] и проверить, верно ли, что эта сумма нечётна и положительна
2. нечётность проверим, вычислив остаток от деления на 2; если этот остаток не равен 0, число нечётное
3. получается такой цикл, после которого нужно не забыть вывести результат – значение счётчика k:

**k := 0;**

**for i:=1 to N-1 do**

**if (a[i]+a[i+1] > 0) and ((a[i]+a[i+1]) mod 2 <> 0) then**

**k := k + 1;**

**writeln(k);**

1. для того, чтобы не вычислять дважды сумму в условном операторе, можно было записать её в свободную переменную j:

**k := 0;**

**for i:=1 to N-1 do begin**

**j := a[i]+a[i+1];**

**if (j > 0) and (j mod 2 <> 0) then**

**k := k + 1**

**end;**

**writeln(k);**

1. возможен и ответ на естественном языке (русском):

**Записать значение 0 в переменную** k**. Затем в цикле перебрать все пары соседних элементов, начиная с пары (a[1],a[2]) до пары (a[N-1],a[N]). Для каждой пары вычислить сумму, если эта сумма положительна и чётна (остаток от её деления на 2 не равен нулю), увеличить значение k на единицу. После завершения цикла вывести значение k.**

|  |
| --- |
| **Возможные проблемы**:   * + не забудьте сказать, что нужно вывести после окончания работы программы   + не забудьте задать правильное начальное значение для счётчика   + не забудьте, что в Паскале каждое из простых условий (отношений) нужно брать в скобки   + не забудьте, что если в теле цикла нужно выполнить несколько операторов, их нужно заключать в «операторные скобки» begin-end   + цикл по i должен заканчиваться не на N, а на N-1, чтобы не было выхода за границы массива (рассматриваются элементы a[i] и a[i+1])   + если вы достаточно хорошо владеете русским языком для того, чтобы понятно излагать свои мысли, с точки зрения тактики рекомендуется писать алгоритм на русском языке – по крайней мере, тут не снизят за пропущенную точку с запятой   + просмотрите внимательно диапазон, в котором находятся исходные числа; дело в том, что во многих языках, например, в Паскале и в Си, остаток от деления отрицательного числа на положительное – число отрицательное, например (-7) mod 2 = -1, поэтому определять, например, «неделимость» элемента массива на 3 с помощью условия **a[i]** **mod** **2** **=** **1** нельзя (не будет работать для отрицательных чисел), нужно использовать условие   **a[i]** **mod** **2** <> **0.** |

### Ещё пример задания:

**Р-03**. *Дан целочисленный массив из 20 элементов. Элементы массива могут принимать целые значения от 0 до 1000. Опишите на русском языке или на одном из языков программирования алгоритм, позволяющий найти и вывести минимальное значение среди элементов массива, которые имеют чётное значение и не делятся на три. Гарантируется, что в исходном массиве есть хотя бы один элемент, значение которого чётно и не кратно трем.*

*Исходные данные объявлены так, как показано ниже. Запрещается использовать переменные, не описанные ниже, но использовать все описанные переменные не обязательно.*

***Естественный язык:***

**Объявляем массив A из 20 элементов.**

**Объявляем целочисленные переменные I, J, MIN.**

**В цикле от 1 до 20 вводим элементы массива A с 1-го по 20-й.**

***Паскаль:***

**const N=20;**

**var a: array [1..N] of integer;**

**i, j, min: integer;**

**begin**

**for i:=1 to N do**

**readln(a[i]);**

**…**

**end.**

**Решение:**

1. даже если вы хорошо владеете программированием, сначала лучше (прежде всего, для себя) написать алгоритм на русском языке
2. здесь требуется найти минимальный элемент из всех, которые имеют чётное значение и не делятся на 3
3. делимость одного целого числа на другое проверяется с помощью операции взятия остатка (в Паскале – операция mod): первое число делится на второе, если остаток от деления равен 0
4. тогда условие, определяющее отбор нужных элементов, запишется в виде

(a[i] mod 2 = 0) and (a[i] mod 3 <> 0)

1. стандартный цикл поиска минимального элемента, удовлетворяющего условию, выглядит так:

**for i:=1 to N do**

**if <условие верно> and (a[i] < min) then**

**min := a[i];**

1. остается один вопрос: каким должно быть начальное значение переменной min? его нужно выбрать таким, чтобы для первого же «подходящего» элемента выполнилось условие a[i] < min, и это «временное» начальное значение было бы заменено на реальное
2. к счастью, диапазон входных чисел ограничен (от 0 до 1000), поэтому можно выбрать любое значение, больше 1000, например, 1001 или 9999[[29]](#footnote-29)
3. таким образом, решение задачи на естественном языке выглядит так:

*Записываем в переменную* min *значение 1001.  
Затем в цикле просматриваем все элементы массива, с первого до последнего. Если остаток от деления очередного элемента на 2 равен 0 и остаток от его деления на 3 не равен нулю и значение элемента меньше, чем значение переменной* min*, записать в переменную* min *значение рассматриваемого элемента массива. Затем переходим к следующему элементу.  
После окончания работы цикла выводим значение переменной* min*.*

1. аналогичное решение на Паскале:

**min:=1001;**

**for i:=1 to N do**

**if (a[i] mod 2=0) and (a[i] mod 3 <> 0) and (a[i]<min) then**

**min:=a[i];**

**writeln(min);**

|  |
| --- |
| **Возможные проблемы**:   * + не забудьте сказать, что нужно вывести после окончания работы программы   + если вы достаточно хорошо владеете русским языком для того, чтобы понятно излагать свои мысли, с точки зрения тактики рекомендуется писать алгоритм на русском языке – по крайней мере, тут не снизят за пропущенную точку с запятой   + просмотрите внимательно диапазон, в котором находятся исходные числа; дело в том, что во многих языках, например, в Паскале и в Си, остаток от деления отрицательного числа на положительное – число отрицательное, например (-7) mod 3 = -1, поэтому определять, например, «неделимость» элемента массива на 3 с помощью условия **a[i]** **mod** **3** **=** **1** нельзя (не будет работать для отрицательных чисел), нужно использовать условие   **a[i]** **mod** **3** <> **0.** |

### Ещё пример задания:

**Р-02**. *Опишите на русском языке или одном из языков программирования алгоритм получения из заданного целочисленного массива размером 30 элементов другого массива, который будет содержать модули значений элементов первого массива (не используя специальной функции, вычисляющей модуль числа).*

**Решение:**

1. даже если вы хорошо владеете программированием, сначала лучше (прежде всего, для себя) написать алгоритм на русском языке (или на псевдокоде – это нечто среднее между словесным алгоритмом и готовой программой)
2. по условию нужно выделить в памяти два массива одинакового размера, назовем их **A** и **B**; обозначим размер массивов через **N**, индексы элементов изменяются от 1 до **N**;
3. в цикле в каждый элемент **B[i]** массива **B** нужно записать модуль соответствующего элемента **A[i]** массива **A**, это нужно сделать для всех **i** от 1 до **N**
4. есть небольшая сложность: запрещено использовать стандартную функцию вычисления модуля; согласно определению модуля решение может быть такое: если элемент **A[i]** больше или равен нулю, записываем в **B[i]** его значение без изменений, а если меньше нуля – меняем знак, то есть, в **B[i]** записываем (**-A[i]**)
5. решение в виде алгоритма на русском языке может выглядеть так:  
   «Выделяем в памяти второй массив того же размера. В цикле рассматриваем все элементы первого массива с первого до последнего. Если текущий элемент больше нуля или равен нулю, в соответствующий элемент второго массива записываем его значение без изменений; если текущий элемент меньше нуля, во второй массив записываем значение элемента с обратным знаком. Выводим второй массив на экран».
6. осталось написать программу, практически дословно реализующую это решение:

**const N = 30;**

**var a, b:array[1..N] of integer;**

**i: integer;**

**begin**

**for i:=1 to N do { ввод всех элементов массива с клавиатуры }**

**read(a[i]);**

**for i:=1 to N do { формирование массива B }**

**if a[i] < 0 then**

**b[i]:= -a[i]**

**else b[i]:= a[i];**

**writeln('Результат:');**

**for i:=1 to N do { вывод всех элементов массива B }**

**write(b[i], ' ');**

**end.**

1. размер массива грамотно задавать через константу (**const N = 30;**), а не вписывать число в каждый цикл; тогда, если нужно будет переделать программу для массива другого размера, достаточно будет изменить всего одно число в начале программы

|  |
| --- |
| **Возможные проблемы**:   * + проверяйте правильность минимального и максимального значения переменной цикла в заголовке цикла **for**   + не забывайте вывести результат в конце работы программы |

### Еще пример задания:

**Р-01**. *Опишите на русском языке или одном из языков программирования алгоритм подсчета максимального количества подряд идущих совпадающих элементов в целочисленном массиве длины 30.*

**Решение:**

1. сначала нужно понять задачу; предположим, что в массиве есть одинаковые элементы, стоящие рядом:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **1** | **2** | **2** | **1** | **1** | **1** | **1** | **3** | **3** | **2** | **2** |

1. самая длинная цепочка стоящих рядом элементов в данном случае состоит из 4-х единиц (она выделена желтым фоном)
2. нам нужно по крайней мере две переменных: для хранения номера текущего элемента (при обработке массива в цикле) и для хранения максимального количества идущих подряд элементов (обозначим ее **kMax**)
3. в целом (пока неточный) алгоритм может выглядеть так: «пройти весь массив, подсчитывая для каждого элемента длину цепочки подряд идущих одинаковых чисел, если эта длина больше **kMax**, то записать ее в **kMax**»
4. отсюда сразу следует, что необходима еще одна переменная (обозначим ее через **k**), показывающая для каждого элемента массива длину цепочки одинаковых чисел, которая заканчивается на этом элементе:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **1** | **1** | **2** | **2** | **1** | **1** | **1** | **1** | **3** | **3** | **2** | **2** |
| **k** | 1 | 2 | 1 | 2 | 1 | 2 | 3 | 4 | 1 | 2 | 1 | 2 |
| **kMax** | 1 | 2 | 2 | 2 | 2 | 2 | 3 | 4 | 4 | 4 | 4 | 4 |

1. следующий шаг к решению: нужно понять, как изменять переменную **k** при проходе по массиву; можно сделать так: если очередной элемент равен предыдущему, счетчик **k** увеличиваем на единицу, а если не равен – записываем в него 1 (цепочка одинаковых чисел кончилась, началась новая, в ней пока один элемент)
2. при таком подходе проблема может возникнуть при просмотре первого элемента, потому что для него нет предыдущего; поэтому описанную выше процедуру будем в цикле применять ко всем элементам массива, начиная **со второго** (а не с первого); в самом начале программы запишем в **k** и **kMax** по единице – таким образом, мы «вручную» (без цикла) рассмотрели первый элемент массива
3. теперь можно написать алгоритм на русском языке:  
   «Выделим две вспомогательные переменные, **k** и **kMax**, и запишем в каждую из них по единице. В цикле рассматриваем все элементы массива со **второго** до последнего, если очередной элемент равен предыдущему, увеличиваем **k**; если **k > kMax**, записываем в **kMax** значение **k**. В конце цикла в **kMax** окажется требуемое значение».
4. этот алгоритм реализуется в такой программе:

**const N =30;**

**var a: array[1..N] of integer;**

**i, k, kMax: integer;**

**begin**

**for i:=1 to N do readln(A[i]); { ввод массива }**

**k := 1; { обрабатываем A[1] }**

**kMax := 1;**

**for i:=2 to N do begin { а теперь в цикле A[2]...A[N} }**

**if A[i] = A[i-1] then { цепочка продолжается }**

**k := k + 1**

**else k := 1; { цепочка закончилась }**

**if k > kMax then kMax := k;**

**end;**

**writeln(kMax);**

**end.**

|  |
| --- |
| **Возможные проблемы**:   * + как видим, основная сложность в этой задаче – не написать программу, а придумать хороший (часто еще нужно – быстрый) алгоритм   + проверьте, что будет записано в переменные до начала цикла (определены ли их начальные значения)   + проверяйте, не выйдет ли индекс за границу массива в начале или в конце цикла   + будьте внимательны с «крайними» случаями, например, нужно обязательно убедиться, что программа работает, когда интересующая нас цепочка стоит в самом начале или в самом конце массива |

### Еще пример задания:

**Р-00**. *Дан целочисленный квадратный массив 10 х 10. Опишите на русском языке или на одном из языков программирования алгоритм вычисления суммы максимальных элементов из каждой строки. Напечатать значение этой суммы. Предполагается, что в каждой строке такой элемент единственный.*

**Решение:**

1. суть задачи: среди элементов каждой строки нужно выбрать максимальный, и все эти выбранные значения сложить
2. несложно сразу написать алгоритм на русском языке:   
   «Чтобы накапливать сумму, нужно ввести целую переменную **Sum**, в которую в самом начале записываем 0 ; далее в цикле просматриваем все строки, для каждой строки находим максимальный элемент и прибавляем его значение к **Sum**. Для определения максимального элемента в строке вводим переменную **max** и сначала записываем в нее значение первого элемента этой строки. Затем в цикле просматриваем все остальные элементы, начиная со **второго** до конца массива. Если очередной элемент больше значения **max**, записываем в **max** значение этого элемента».
3. сначала напишем программу на псевдокоде:

**const N=10;**

**{ ввод матрицы N на N }**

**Sum := 0;**

**for i:=1 to N do begin**

**{ max := максимальный элемент в i-ой строке }**

**Sum := Sum + max;**

**end;**

1. остается записать на Паскале те части, которые взяты в фигурные скобки, и до конца оформить программу; по правилам ЕГЭ можно не писать в программе команды для ввода массива, поэтому мы оставим на этом месте комментарий:

**const N=10;**

**var A: array[1..N,1..N] of integer;**

**i, k, max, Sum: integer;**

**begin**

**{ ввод матрицы N на N }**

**Sum := 0;**

**for i:=1 to N do begin**

**max := A[i,1];**

**for k:=2 to N do**

**if A[i,k] > max then max := A[i,k];**

**Sum := Sum + max;**

**end;**

**writeln(Sum);**

**end.**

|  |
| --- |
| **Возможные проблемы**:   * + проверьте, правильно ли заданы (и заданы ли вообще) начальные значения для всех переменных   + проверьте, правильно ли расставлены операторные скобки **begin-end**, ограничивающие тело цикла; их обязательно нужно ставить, если в теле цикла несколько операторов   + проверяйте, не выйдет ли индекс за границу массива в начале или в конце цикла   + не перепутайте номер строки (это первый индекс) и номер столбца (второй индекс)   + для надежности не рекомендуется использовать в одной программе переменные **i** и **j**, потому что они слишком похоже выглядят, вот пример ошибочного решения в этой задаче:   **for i:=1 to N do begin**  **max := A[i,1];**  **for i:=2 to N do**  **if A[j,i] > max then max := A[i,j];**  **Sum := Sum + max;**  **end;**   * + если вы все же используете переменные **i** и **j**, нужно писать их очень четко, чтобы они отличались друг от друга |

|  |
| --- |
| **Немного тактики:**   * + в этом задании можно написать алгоритм на русском языке, а можно (вместо этого) написать компьютерную программу на одном из языков программирования   + если вы хорошо умеете выражать свои мысли по-русски, ~~собаководы~~ эксперты рекомендуют писать **только** алгоритм на русском языке; дело в том, что если вы сделаете много ошибок в программе, оценка будет снижена даже при абсолютно правильном алгоритме   + если вам сложно изъясняться на родном языке, а легче записать свои мысли на Паскале или Си – пишите программу, но тщательно проверяйте ее на предмет возможных случайных ошибок-опечаток, которые можно сделать просто по невнимательности:   + задавайте все начальные значения для переменных   + проверяйте правильность написания ключевых слов   + если в теле цикла несколько операторов, заключайте их в блок **begin-end** (операторные скобки)   + проверяйте начальное и конечное значение переменной цикла   + если вы используете циклы while или repeat, проверьте, что переменная цикла изменяется в теле цикла (иначе в программе будет зацикливание, а на ЕГЭ – потерянные баллы)   + выводите на экран именно то, что требуется по условию   + ставьте точку с запятой в конце операторов   + НЕ ставьте точку с запятой перед **else** (Паскаль)   + ставьте точку в конце последнего оператора **end** (Паскаль) |

|  |
| --- |
| **За что снимают баллы:**   * + задано неверное начальное значение переменных (или вообще не задано)   + неверно указано условие завершения цикла   + «забыли» изменять переменную цикла в цикле **while** (**repeat**)   + перепутаны знаки < и >, логические операции **or** и **and**   + неверно расставлены операторные скобки **begin-end**   + программа не выводит результат или выводит не то, что спрашивают   + синтаксические ошибки (знаки пунктуации – запятые, точки, точки с запятой; неверное написание ключевых слов) допускаются в разумных пределах (если они не искажают замысел автора) |

# 

# 26 (C3) (высокий уровень, время – 30 мин)

**Тема**: Дерево игры. Поиск выигрышной стратегии.

**Что нужно знать**:

* в простых играх можно найти выигрышную стратегию, просто перебрав все возможные варианты ходов соперников
* для примера рассмотрим такую игру: сначала в кучке лежит 5 спичек; два игрока убирают спички по очереди, причем за 1 ход можно убрать 1 или 2 спички; выигрывает тот, кто оставит в кучке 1 спичку
* первый игрок может убрать одну спичку (в этом случае их останется 4), или сразу 2 (останется 3), эти два варианта можно показать на схеме:

1 игрок

* если первый игрок оставил 4 спички, второй может своим ходом оставить 3 или 2; а если после первого хода осталось 3 спички, второй игрок может выиграть, взяв две спички и оставив одну:

2 игрок

1 игрок

* если осталось 3 или 2 спички, то 1-ый игрок (в обеих ситуациях) выиграет своим ходом:

1 игрок

2 игрок

1 игрок

* простроенная схема называется «деревом игры», она показывает все возможные варианты, начиная с некоторого начального положения (для того, чтобы не загромождать схему, мы не рисовали другие варианты, если из какого-то положения есть выигрышный ход)
* в любой ситуации у игрока есть два возможных хода, поэтому от каждого узла этого дерева отходят две «ветки», такое дерево называется *двоичным* (если из каждого положения есть три варианта продолжения, дерево будет *троичным*)
* проанализируем эту схему; если первый игрок своим первым ходом взял две спички, то второй сразу выигрывает; если же он взял одну спичку, то своим вторым ходом он может выиграть, независимо от хода второго игрока
* кто же выиграет при правильной игре? для этого нужно ответить на вопросы: 1) «Может ли первый игрок выиграть, независимо от действий второго?», и 2) «Может ли второй игрок выиграть, независимо от действий первого?»
* ответ на первый вопрос – «да»; действительно, убрав всего одну спичку первым ходом, 1-ый игрок всегда может выиграть на следующем ходу
* ответ на второй вопрос – «нет», потому что если первый игрок сначала убрал одну спичку, второй всегда проиграет, если первый не ошибется
* таким образом, при правильной игре выиграет первый игрок; для этого ему достаточно первым ходом убрать всего одну спичку
* в некоторых играх, например, в рэндзю (крестики-нолики на бесконечном поле) нет выигрышной стратегии, то есть, при абсолютно правильной игре обоих противников игра бесконечна (или заканчивается ничьей); кто-то может выиграть только тогда, когда его соперник по невнимательности сделает ошибку
* полный перебор вариантов реально выполнить только для очень простых игр; например, в шахматах сделать это за приемлемое время не удается (дерево игры очень сильно разветвляется, порождая огромное количество вариантов)
* все позиции в простых играх делятся на выигрышные и проигрышные
* **выигрышная позиция** – это такая позиция, в которой игрок, делающий первый ход, может гарантированно выиграть при любой игре соперника, если не сделает ошибку; при этом говорят, что у него есть выигрышная стратегия – алгоритм выбора очередного хода, позволяющий ему выиграть
* если игрок начинает играть в **проигрышной** позиции, он обязательно проиграет, если ошибку не сделает его соперник; в этом случае говорят, что у него нет выигрышной стратегии; таким образом, общая стратегия игры состоит в том, чтобы своим ходом создать проигрышную позицию для соперника
* выигрышные и проигрышные позиции можно охарактеризовать так:
  + позиция, из которой все возможные ходы ведут в выигрышные позиции – **проигрышная**;
  + позиция, из которой хотя бы один из возможных ходов ведет в проигрышную позицию - **выигрышная**, при этом стратегия игрока состоит в том, чтобы перевести игру в эту проигрышную (для соперника) позицию.

### Ещё пример задания:

*Два игрока, Петя и Ваня, играют в следующую игру. Перед игроками лежит куча камней. Игроки ходят по очереди, первый ход делает Петя. За один ход игрок может добавить в кучу один или три камня или увеличить количество камней в куче в два раза. Например, имея кучу из 15 камней, за один ход можно получить кучу из 16, 18 или 30 камней. У каждого игрока, чтобы делать ходы, есть неограниченное количество камней. Игра завершается в тот момент, когда количество камней в куче становится не менее 35. Победителем считается игрок, сделавший последний ход, т.е. первым получивший кучу, в которой будет 35 или больше камней. В начальный момент в куче было S камней; 1 ≤ S ≤ 34.*

*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. Выполните следующие задания. Во всех случаях обосновывайте свой ответ.*

***Задание 1***

*а) Укажите все такие значения числа S, при которых Петя может выиграть в один ход. Обоснуйте, что найдены все нужные значения S, и укажите выигрывающие ходы.*

*б) Укажите такое значение S, при котором Петя не может выиграть за один ход, но при любом ходе Пети Ваня может выиграть своим первым ходом. Опишите выигрышную стратегию Вани.*

***Задание 2***

*Укажите два таких значения S, при которых у Пети есть выигрышная стратегия, причём одновременно выполняются два условия:*

*− Петя не может выиграть за один ход;*

*− Петя может выиграть своим вторым ходом независимо от того, как будет ходить Ваня.*

*Для каждого указанного значения S опишите выигрышную стратегию Пети.*

***Задание 3***

*Укажите значение S, при котором одновременно выполняются два условия:*

*− у Вани есть выигрышная стратегия, позволяющая ему выиграть первым или вторым ходом при любой игре Пети;*

*− у Вани нет стратегии, которая позволит ему гарантированно выиграть первым ходом.*

*Для указанного значения S опишите выигрышную стратегию Вани.*

*Постройте дерево всех партий, возможных при этой выигрышной стратегии Вани (в виде рисунка или таблицы). На рисунке на рёбрах дерева указывайте, кто делает ход; в узлах – количество камней в позиции.*

**Решение (способ 1, таблица):**

1. **Задание 1а.** Последним ходом может быть «+1», «+3» или «\*2». Выиграть последним ходом «+1» можно, если S = 34. Ходом «+2» можно выиграть при S=32, S=33 и S=34. Ходом «\*2» можно выиграть из любой позиции при S > 17. Можно составить таблицу, в которой «В1» обозначает выигрыш за один ход:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | … | 34 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | В1 | В1 | … | В1 |

Поэтому ответ должен быть такой:

**Задание 1а.** *Петя может выиграть за один ход при любом S > 17. Он должен увеличить вдвое число камней, при этом в куче всегда получится не менее 36 камней.*

1. **Задание 1б**. Ваня может выиграть в один ход тогда, когда все ходы Пети из текущей позиции ведут в выигрышные позиции. Это будет при S = 17:

**Задание 1б.** Ваня может гарантированно выиграть своим первым ходом при S = 17. В этом случае Петя своим первым ходом может получить в куче 18, 19 или 34 камня, то есть, выиграть за один ход не может. В любой из этих позиций Ваня выигрывает своим первым ходом, удваивая количество камней.

Позицию S = 17 отмечаем в таблице как проигрышную (за 1 ход):

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | … | 34 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | П1 | В1 | В1 | … | В1 |

1. Для того, чтобы Петя смог выиграть своим вторым ходом, ему нужно своим первым ходом перевести игру в проигрышную (для Вани) позицию, то есть, получить 17 камней. Он может сделать это при S = 14 (ходом «+3») или при S = 16 (ходом «+1»).

**Задание 2.** При S = 14 или S = 16 Петя своим первым ходом может получить 17 камней, переведя игру в проигрышную (для Вани) позицию. Поэтому своим вторым ходом Петя всегда выиграет.

В таблице обозначим эти позиции как выигрышные (за 2 хода):

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | … | 34 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  | В2 |  | В2 | П1 | В1 | В1 | … | В1 |

1. для выполнения задания № 3 нужно найти такие позиции, из которых **все возможные ходы** ведут в выигрышные позиции, помеченные как В1 или В2; это позиции S = 13 и S = 15: при S = 13 можно получить 14, 16 или 26 камней, все эти позиции выигрышные; при S = 15 можно получить 16, 18 или 30 камней, это так же выигрышные позиции
2. В задании требуется найти только одну подходящую позицию, выбираем S = 13.

**Задание 3.** При S = 13 после первого хода Пети в куче будет 14, 16, или 26 камней. Если в куче получилось 14 или 16 камней, Ваня выиграет своим вторым ходом (см. задание 2). Если получилось 26 камней, Ваня выигрывает первым ходом, удвоив количество камней.

Строим дерево игры, рассматривая на каждом шаге **все возможные ходы Пети** и **только выигрышный ход Вани**:

52

36

40

68

+1

+3

\*2

+3

+1

\*2

\*2

\*2

\*2

+1

\*2

+3

**Петя**

**Ваня**

**Петя**

**Ваня**

У нас получилось не совсем дерево, потому что на первом ходу Ваня из двух позиций (S=14 и S=16) приводит игру к проигрышной для Пети позиции S=17. Для сокращения записи можно привести стрелки в один узел. Зелёные прямоугольники обозначают выигрыш Вани.

### Ещё пример задания:

*Два игрока, Петя и Ваня, играют в следующую игру. Перед игроками лежит куча камней. Игроки ходят по очереди, первый ход делает Петя. За один ход игрок может добавить в кучу один камень или увеличить количество камней в куче в два раза. Например, имея кучу из 15 камней, за один ход можно получить кучу из 16 или 30 камней. У каждого игрока, чтобы делать ходы, есть неограниченное количество камней. Игра завершается в тот момент, когда количество камней в куче становится не менее 22. Победителем считается игрок, сделавший последний ход, то есть первым получивший кучу, в которой будет 22 или больше камней.*

*В начальный момент в куче было S камней, 1 ≤ S ≤ 21. 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.*

*Выполните следующие задания. Во всех случаях обосновывайте свой ответ.*

*1. а) Укажите все такие значения числа S, при которых Петя может выиграть в один ход. Обоснуйте, что найдены все нужные значения S, и укажите выигрывающий ход для каждого указанного значения S.*

*б) Укажите такое значение S, при котором Петя не может выиграть за один ход, но при любом ходе Пети Ваня может выиграть своим первым ходом. Опишите выигрышную стратегию Вани.*

*2. Укажите два таких значения S, при которых у Пети есть выигрышная стратегия, причём  
– Петя не может выиграть за один ход, и  
– Петя может выиграть своим вторым ходом, независимо от того, как будет ходить Ваня.*

*Для каждого указанного значения S опишите выигрышную стратегию Пети.*

*3. Укажите значение S, при котором:  
– у Вани есть выигрышная стратегия, позволяющая ему выиграть первым или вторым ходом при любой игре Пети, и  
– у Вани нет стратегии, которая позволит ему гарантированно выиграть первым ходом.*

*Для указанного значения S опишите выигрышную стратегию Вани.*

*Постройте дерево всех партий, возможных при этой выигрышной стратегии Вани (в виде рисунка или таблицы). На рёбрах дерева указывайте, кто делает ход, в узлах – количество камней в куче.*

**Решение (способ 1, таблица):**

1. **Вопрос 1а.** Последним ходом может быть «+1» или «\*2». Выиграть последним ходом «+1» можно, если S = 21. Ходом «\*2» можно выиграть из любой позиции при S > 10 (сюда входит и 21!). Можно составить таблицу, в которой «В1» обозначает выигрыш за один ход:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
|  |  |  |  |  |  |  |  |  |  |  | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 |

Поэтому ответ должен быть такой:

«1а. *Петя может выиграть за один ход при любом S > 10. Он должен увеличить вдвое число камней, при этом в куче всегда получится не менее 22 камней.*»

1. **Вопрос 1б.** Для ответа на этот вопрос нужно найти позицию, из которой все возможные ходы ведут к выигрышу за 1 ход, то есть к позиции, отмеченной в таблице как «В1». Например, это позиция при S = 10: ход «+1» ведёт в выигрышную позицию S = 11, а ход «\*2» ведёт в выигрышную позицию S = 20. Поэтому позицию S = 10 отметим в таблице как «×­1» (проигрыш за 1 ход):

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
|  |  |  |  |  |  |  |  |  |  | **×­1** | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 |

Ответ на вопрос 1б должен быть такой:

«1б. *При S = 10 Петя не может выиграть в один ход, потому что при его ходе «+1» число камней в куче становится равно 11 (меньше 22), а при ходе «\*2» число камней в куче становится равно 20 (также меньше 22). Других возможных ходов у Пети нет. Из любой позиции после одного хода Пети (это может быть 11 или 20), Ваня может выиграть своим первых ходом, удвоив количество камней в куче.*»

1. **Вопрос 2**. Пете, для того, чтобы гарантированно выиграть на втором ходу, нужно из начальной позиции перевести игру в проигрышную позицию, отмеченную знаком «×1». Пока мы нашли одну такую позицию: S = 10. Петя может перевести игру в эту позицию из позиций

S = 9 (ходом «+1») и S = 5 (ходом «\*2»)

В таблице отмечаем эти положения как «В­­2» – гарантированный выигрыш за 2 хода:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
|  |  |  |  |  | В2 |  |  |  | В2 | **×1** | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 |

Поэтому ответ должен быть такой:

«2. *Из позиций S = 9 и S = 5 Петя не может выиграть в один ход, но Петя может выиграть своим вторым ходом, независимо от того, как будет ходить Ваня. При S = 9 ходом «+1» Пете нужно перевести игру в позицию S = 10, которая является проигрышной (см. ответ на вопрос 1б). При S = 5 Петя переводит игру в ту же позицию ходом «\*2».*»

1. **Вопрос 3**. Нужно найти такую позицию, из которой оба возможных хода Пети ведут в позиции, отмеченные в таблице как «В­1» (выигрыш в 1 ход) и «В­2» (выигрыш в 2 хода). Например, это позиция S = 8, из которой можно «попасть» только в S = 9 («В2») и S = 16 («В1»). Отмечаем эту позицию как «×2» – проигрыш в два хода:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
|  |  |  |  |  | В2 |  |  | **×2** | В2 | **×­1** | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 | В1 |

Поэтому ответ должен быть такой:

«3. *В позиции S = 8 у Вани есть выигрышная стратегия, которая позволяет ему выиграть первым или вторым ходом. Если Петя выбирает ход «+1», в куче становится 9 камней и Ваня выигрывает на 2-м ходу (см. ответ на вопрос 2). Если Петя выбирает ход «\*2», Ваня выигрывает первым ходом, удвоив число камней в куче.*»

1. Остается нарисовать дерево возможных вариантов игры из позиции S = 8. Для этого используем построенную таблицу:

Здесь красным цветом выделены позиции, в которых игра заканчивается.

Обратите внимание, что на каждом шаге мы рассматриваем все возможные ходы Пети и только один лучший ход Вани. Например, в позиции S = 11 Ваня может сделать ход «+1» и получить 12 камней в куче, но тогда он проиграет (Петя следующим ходом удвоит число камней и получит 24 камня). Этот ход мы не рассматриваем, потому что мы хотим доказать, что **у Вани** есть выигрышная стратегия – ему достаточно хода «\*2», после которого он выиграет. В то же время нужно рассмотреть **все возможные ответы Пети**, чтобы доказать, что у него нет шансов на выигрыш при правильной игре Вани. В этом суть теории игр – добиться лучшего результата в худшем случае, то есть при безошибочной игре соперника.

Построенное дерево можно записать и в другой форме, например, «положив его на бок»:

Ещё один вариант – представить дерево в виде таблицы:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Начальная позиция | 1-й ход Пети  (все варианты) | 1-й ход Вани  (ход по стратегии) | 2-й ход Пети  (все варианты) | 2-й ход Вани  (ход по стратегии) |
| 8 | 9 | 10 | 11 | **22 (выигрыш)** |
| 20 | **40 (выигрыш)** |
| 16 | **32 (выигрыш)** |  |  |

**Решение (способ 2, математический, О.В. Лучникова, г. Новокузнецк):**

1. **Вопрос 1а.** Петя выигрывает первым ходом:

П1

Петя должен правильно выбрать одно из двух возможных действий (+1 **ИЛИ** \*2), которое переведет кучу камней к состоянию ≥22. Таким образом, получаем совокупность неравенств:



1. **Вопрос 1б**. Ваня выигрывает первым ходом:

В1

П1

Любое действие Пети (И +1 **И** \*2) должно привести кучу камней к состоянию **.** Только это может обеспечить выигрыш Вани на следующем ходу. Таким образом, получаем систему:



1. **Вопрос 2.** Назовите два значения S, при которых Петя может выиграть своим вторым ходом?

П2

В1

П1

1. Петя должен выиграть, а это значит, он должен правильно выбрать один из двух возможных вариантов действий (+1 **ИЛИ** \*2), которое переведет кучу камней к состоянию . Только это может обеспечить ему выигрыш при любом действии его противника Вани. Таким образом, получаем совокупность:



1. **Вопрос 3**. При каком *S* Ваня выигрывает своим первым или вторым ходом?
2. Сначала найдем, при каком *S* Ваня гарантированно выигрывает именно вторым ходом.

В2

П2

В1

П1



Таким образом, получаем, что нет такого количества камней *S*, которые гарантировали бы выигрыш Вани именно после его второго хода при любых действиях Пети.

1. Найдем, при каких значениях *S* Петя не сможет победить ни после первого, ни после второго хода. Т.е. любое действие Пети приведет кучу камней к такому состоянию, при котором Ваня сможет выиграть после 1 или после второго хода:

В1

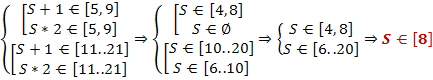
П1

В2

П2

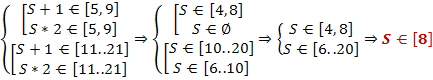
В1

П1



1. Совокупность решений первой и второй частей – и есть все множество решений третьего вопроса. Т.е. S = 8.
2. Построим дерево игры для S = 8

**Важное замечание** по поводу решения этой задачи методом О.В. Лучниковой (**Г.  Сергеев**, ГБОУ Гимназия 1551 г. Москвы): в случае, когда возможных ходов не два, а больше, при ответе на вопрос 3 прямое применение этого метода может привести к неверному результату. Действительно, система



означает, что один из возможных ходов ведёт в позицию типа В2 (выигрыш в два хода), а другой – в позицию типа В­1 (выигрыш в один хода). Если есть еще и другие возможные ходы, они могут вести в проигрышные позиции, тогда, выбрав один из этих ходов, Петя может выиграть. Таким образом, к этой системе нужно добавить условие «все возможные ходы ведут в позиции типа В1 или В­2». Еще раз отметим, что в задачах с двумя возможными ходами оно выполнится автоматически. Кроме того, нужно учесть, что из ответа на этот вопрос нужно исключить ответ на вопрос 1б, то есть позиции, из которых есть гарантированный выигрыш в 1 ход.

Детали решения в случае трёх возможных ходов см. в следующей разобранной задаче (**решение Г. Сергеева**).

**Решение (способ 3, «холмы и ямы», А. Козлов, г. Северобайкальск):**

1. будем обозначать на рисунке выигрышные позиции «холмом» (возвышенностью), а проигрышные – «ямой» (впадиной); таким образом, задача игрока – «посадить соперника в яму», то есть создать для него проигрышную позицию
2. **Вопрос 1а.** Последним ходом может быть «+1» или «\*2». Выиграть последним ходом «+1» можно, если S = 21. Ходом «\*2» можно выиграть из любой позиции при S > 10 (сюда входит и 21!). Таким образом, можно выделить первый «холм», стартовав с которого игрок выигрывает в один ход (число 1 над «холмом»):

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |
| S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |

Поэтому ответ должен быть такой:

«1а. *Петя может выиграть за один ход при любом S > 10. Он должен увеличить вдвое число камней, при этом в куче всегда получится не менее 22 камней.*»

1. **Вопрос 1б.** Для ответа на этот вопрос нужно найти позицию, из которой все возможные ходы ведут к выигрышу за 1 ход, то есть к позиции, отмеченной в таблице как «В1». Например, это позиция при S = 10: ход «+1» ведёт в выигрышную позицию S = 11, а ход «\*2» ведёт в выигрышную позицию S = 20. Поэтому позицию S = 10 отметим в таблице как «яму» и укажем внизу 1 (проигрыш за 1 ход):

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |
| S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
|  |  |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  |  |  |  |

Ответ на вопрос 1б должен быть такой:

«1б. *При S = 10 Петя не может выиграть в один ход, потому что при его ходе «+1» число камней в куче становится равно 11 (меньше 22), а при ходе «\*2» число камней в куче становится равно 20 (также меньше 22). Других возможных ходов у Пети нет. Из любой позиции после одного хода Пети (это может быть 11 или 20), Ваня может выиграть своим первых ходом, удвоив количество камней в куче.*»

1. **Вопрос 2**. Пете, для того, чтобы гарантированно выиграть на втором ходу, нужно из начальной позиции перевести игру в проигрышную позицию, отмеченную знаком «×1». Пока мы нашли одну такую позицию: S = 10. Петя может перевести игру в эту позицию из позиций

S = 9 (ходом «+1») и S = 5 (ходом «\*2»)

В таблице отмечаем эти положения как «холмы» с индексом 2 – гарантированный выигрыш за 2 хода:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | 2 |  |  |  | 2 |  |  |  |  |  |  | 1 |  |  |  |  |  |
| S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
|  |  |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  |  |  |  |

Поэтому ответ должен быть такой:

«2. *Из позиций S = 9 и S = 5 Петя не может выиграть в один ход, но Петя может выиграть своим вторым ходом, независимо от того, как будет ходить Ваня. При S = 9 ходом «+1» Пете нужно перевести игру в позицию S = 10, которая является проигрышной (см. ответ на вопрос 1б). При S = 5 Петя переводит игру в ту же позицию ходом «\*2».*»

1. **Вопрос 3**. Нужно найти такую позицию, из которой оба возможных хода Пети ведут в позиции, отмеченные в таблице как «холмы» с метками 1 (выигрыш в 1 ход) или 2 (выигрыш в 2 хода). Например, это позиция S = 8, из которой можно «попасть» только в S = 9 («холм-2») и S = 16 («холм-1»). Отмечаем эту позицию как «яму» с меткой 2 – проигрыш в два хода:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | 2 |  |  |  | 2 |  |  |  |  |  |  | 1 |  |  |  |  |  |
| S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
|  |  |  |  |  |  |  |  | 2 |  | 1 |  |  |  |  |  |  |  |  |  |  |  |

Поэтому ответ должен быть такой:

«3. *В позиции S = 8 у Вани есть выигрышная стратегия, которая позволяет ему выиграть первым или вторым ходом. Если Петя выбирает ход «+1», в куче становится 9 камней и Ваня выигрывает на 2-м ходу (см. ответ на вопрос 2). Если Петя выбирает ход «\*2», Ваня выигрывает первым ходом, удвоив число камней в куче.*»

### Ещё пример задания:

*Здесь и в задачах для тренировки условие записано в сокращенном виде для экономии места. Полную форму записи условия см. в первой разобранной задаче.*

*Два игрока, Петя и Ваня, играют в следующую игру. Перед игроками лежит куча камней. Игроки ходят по очереди, первый ход делает Петя. За один ход игрок может добавить в кучу один камень, добавить в кучу три камня или увеличить количество камней в куче в два раза. Например, имея кучу из 15 камней, за один ход можно получить кучу из 16, 18 или 30 камней. У каждого игрока, чтобы делать ходы, есть неограниченное количество камней. Игра завершается в тот момент, когда количество камней в куче становится не менее 30.   
В начальный момент в куче было S камней, 1 ≤ S ≤ 29.*

*1. При каких S: 1а) Петя выигрывает первым ходом; 1б) Ваня выигрывает первым ходом?*

*2. Назовите три значения S, при которых Петя может выиграть своим вторым ходом?*

*3. При каких S Ваня выигрывает своим первым или вторым ходом?*

**Решение (способ 2, математический, Г.  Сергеев**, г. Москва**):**

1. **Вопрос 1а.** Петя выигрывает первым ходом:

П1

Петя должен правильно выбрать один из трёх возможных вариантов действий   
(+1 **ИЛИ** +3 **ИЛИ** \*2), которое переведет кучу камней к состоянию ≥30. Таким образом, получаем совокупность неравенств:

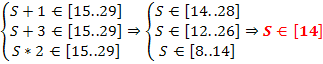


1. **Вопрос 1б.** Ваня выигрывает первым ходом

В1

П1

Любое действие Пети (**И** +1 **И** +3 **И** \*2) должно привести кучу камней к состоянию   
**.** Только это может обеспечить выигрыш Вани на следующем ходу. Таким образом, получаем **систему**:



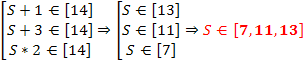
1. Назовите три значения S, при которых Петя может выиграть своим вторым ходом?

П2

В1

П1

Петя должен выиграть, а это значит, он должен правильно выбрать один из трёх возможных вариантов действий (+1 **ИЛИ** +3 **ИЛИ** \*2), которое переведет кучу камней к состоянию . Только это может обеспечить ему выигрыш при любом действии его противника Вани. Таким образом, получаем совокупность:



1. **Вопрос 3.** При каком S Ваня выигрывает своим первым или вторым ходом?

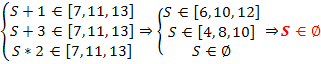
Сначала найдем, при каком S Ваня выигрывает своим вторым ходом.

В2

П2

В1

П1



Таким образом, получаем, что нет такого количества камней S, которые гарантировали бы выигрыш Вани именно после его второго хода при любых действиях Пети.

1. Найдем, при каких значениях S любое действие Пети приведет кучу камней к такому состоянию, при котором Ваня сможет выиграть после 1 или после второго хода:

В1

П1

В2

П2

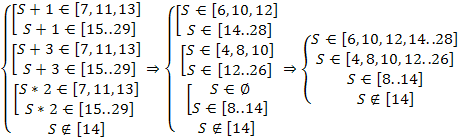
В1

П1

1. Составим систему на основе следующих условий:
   1. любой ход Пети ведет в позицию выигрыша в два хода () или в один ход ()



* 1. текущая позиция не совпадает с проигрышной позицией в один ход (), иначе, кроме нужных значений S, мы здесь получим ещё ответ на вопрос 1б



1. итак, **ответ на вопрос 3**: S = 10 или 12.
2. Построим дерево игры для S = 10. Обратите внимание, что после ходов Пети +1 и +3 Ваня своим следующим ходом сводит игру к одной и той же проигрышной (для Пети) позиции   
   S = 14.

# 27(C4) (высокий уровень, время – 55 мин)

**Тема**: Обработка данных, вводимых в виде символьных строк (написать программу средней сложности из 30-50 строк) или последовательности чисел.

**Что нужно знать**:

* символьная строка – это цепочка символов, которая может обрабатываться как единое целое
* для обращения к символу с номером **i** строки **s** используется запись **s[i],** это говорит о том, что строка – особый вариант массива, в котором хранятся символы
* знак сложения при работе с символьными строками означает сцепку, объединение двух строк в одну (добавление второй строки в конец первой), например:

**s := '123' + '456'; { получили '123456' }**

* с помощью функции **Ord** можно получить код символа; цифры имеют коды от 48 (цифра 0) до 57 (цифра 9), например

**k := Ord('1'); { получили 49 }**

то же самое можно сделать с помощью преобразования типа (привести **char** к **integer**)

**k := integer('1'); { получили 49 }**

* с помощью функции **Chr** можно сделать обратный переход: получить символ по его коду, например

**c := Chr(49); { получили символ '1' }**

то же самое можно сделать с помощью преобразования типа (привести **integer** к **char**)

**c := char(49); { получили символ '1' }**

* для работы со строками в наиболее распространенных Паскаль-средах (*Turbo Pascal, Borland Pascal, PascalABC,* среда *АЛГО*) используют стандартные функции (здесь **s** – это переменная типа **string**, символьная строка; **n** и **r** – целые переменные)

|  |  |
| --- | --- |
| **n := Length(s);** | записать длину строки **s** в целую переменную **n** |
| **s1 := Copy(s, 2, 5);** | записать в символьную строку **s1** подстроку строки **s**, которая начинается с символа с номером 2 и состоит из 5 символов (важно – не со 2-го по 5-ый символ!) |
| **n := Pos('Вася', s);** | записать в целую переменную **n** номер символа, с которого в строке **s** начинается подстрока 'Вася' (если ее нет, в переменную **n** записывается 0); так же можно искать отдельные символы  (важно: сначала указываем, **что** ищем, а потом – **где**) |
| **n := StrToInt(s);** | преобразовать строку **s** в целое число и записать результат в переменную **n** (*PascalABC, Delphi*) |

и процедуры

|  |  |
| --- | --- |
| **Delete(s, 2, 5);** | удалить из строки **s** 5 символов, начиная со второго |
| **Insert('Вася', s, 3);** | вставить в строку **s** фрагмент 'Вася', начиная с третьего символа (между 2-м и 3-м) |
| **Val(s, n, r);** | преобразовать строку **s** в целое число и записать результат в переменную **n**; если при этом произошла ошибка, в переменной **r** будет номер ошибочного символа, если все нормально – ноль |

* структура (в Паскале она называется «запись», *record*) – это сложный тип данных, который может включать в себя несколько элементов – полей; поля могут иметь различный тип
* записи в Паскале объявляются с помощью ключевого слова **record**; в простейшем случае можно выделить память под одну запись так:

**var x: record**

**name: string;**

**code: integer;**

**end;**

эта запись состоит из двух полей: символьной строки **name** и целого числа **code**

* записи очень удобны для работы, когда все данные в целом представляют собой единый блок информации, например, данные об ученике; если не использовать записи, было бы нужно выделять в памяти отдельно символьную строку и отдельно целую переменную, причем эти данные внешне были бы никак не связаны, поэтому программа с записями часто получается логичнее и понятнее как для автора, так и для того, кто будет в ней разбираться
* для обращения к полям записи используют точку, например **x.name** означает «поле **name** записи **x**»
* можно сразу объявить массив записей:

**var Info: array[1..100] of record**

**name: string;**

**code: integer;**

**end;**

это 100 одинаковых записей, имеющих общее имя **Info** и расположенных в памяти рядом; в каждой структуре есть поля **nаme** и **code**; чтобы работать с полями записи с номером **k** используют обращения вида **Info[k].name** и **Info[k].code**

**Сложность алгоритмов:**

* обозначение  говорит о том, что при увеличении в 2 раза размера массива данных количество операций тоже увеличивается примерно в 2 раза (для больших N)
* сложность  имеет алгоритм с одним или несколькими простыми (не вложенными!) циклами в каждом из которых выполняется N шагов (как при поиске минимального элемента)
* количество операций для алгоритма, имеющего сложность , вычисляется по формуле , где a и b – некоторые постоянные
* если в одном алгоритме решения задачи используется несколько циклов от 1 до N, а во втором – только один цикл, то алгоритм с одним циклом, как правило, эффективнее (хотя оба алгоритма имеют сложность , постоянная  в каждом случае своя, для алгоритма с несколькими циклами она будет больше)
* для алгоритма, имеющего сложность , количество операций пропорционально квадрату размера массива, то есть, если N увеличить в 2 раза, то количество операций увеличивается примерно в 4 раза (например, в программе используется два вложенных цикла, в каждом из которых N шагов); сложность  имеют простые способы сортировки массивов: метод «пузырька», метод выбора
* при больших N функция  растет значительно быстрее, чем , поэтому алгоритм, имеющий сложность  всегда менее эффективен, чем алгоритм сложности 
* иногда встречаются алгоритмы сложности  (три вложенных цикла от 1 до N), при больших N они работают медленнее, чем любой алгоритм сложности , то есть, менее эффективны
* для многих задач известны только алгоритмы экспоненциальной сложности, когда размер массива входит в показатель степени, например , для больших N такие задачи не решаются за приемлемое время (например, «взламывание» шифров)

### Пример задания:

*На вход программе подаются сведения о номерах школ учащихся, участвовавших в олимпиаде. В первой строке сообщается количество учащихся N, каждая из следующих N строк имеет формат:*

**<Фамилия> <Инициалы> <номер школы>**

*где <Фамилия> – строка, состоящая не более чем из 20 символов, <Инициалы> – строка, состоящая из 4-х символов (буква, точка, буква, точка), <номер школы> – не более чем двузначный номер. <Фамилия> и <Инициалы>, а также <Инициалы> и <номер школы> разделены одним пробелом. Пример входной строки:*

**Иванов П.С. 57**

*Требуется написать как можно более эффективную программу (укажите используемую версию языка программирования, например, Borland Pascal 7.0), которая будет выводить на экран информацию, из какой школы было меньше всего участников (таких школ может быть несколько). При этом необходимо вывести информацию только по школам, пославшим хотя бы одного участника. Следует учитывать, что N>=1000.*

**Как правильно понимать условие?**

1. на первый вопрос – как именно вводятся данные – находим ответ в самом начале условия: вроде бы «дежурная» фраза «на вход программе подаются…» означает, что данные нужно читать не из файла, а со стандартного входного потока; это, в свою очередь, значит, что можно использовать привычные операторы **read** (**readln**), предполагая, что кто-то вводит эти данные с клавиатуры вручную[[30]](#footnote-30)
2. итак, сначала вводится количество записей в файле **N**, а затем **N** строк с информацией; заметим, что из всей этой информации нас интересует (в каждой строке) только номер школы, остальное можно просто отбрасывать
3. номер школы стоит после второго пробела в строке
4. «*<номер школы> – не более чем двузначный номер*» – крайне важная информация; собственно, только она и позволяет найти хорошее решение задачи; это значит, что школ не более 99!
5. что означает выражение «как можно более эффективная программа»?

* прежде всего, данные читаются только один раз, за один проход, нельзя «вернуться» и прочитать что-то вновь
* в программе не выполняются никакие лишние действия
* используемые алгоритмы имеют минимальную сложность (см. выше)
* расходуется минимальный возможный объем памяти; например, чтобы найти количество отрицательных элементов массива, не нужно вводить второй массив; если нам достаточно держать в памяти одну введенную строку, не нужно одновременно хранить все прочитанные строки

1. зачем нужно уточнение «*N>=1000*»? этим авторы задачи намекают на то, что не нужно считывать все данные в оперативную память, а потом уже их обрабатывать; основная обработка должна быть сделана сразу, в том же цикле, где читаются входные данные
2. мы будем считать, что в исходных данных нет ошибок (так принято на олимпиадах и экзаменах), иначе обработка разнообразных ошибок будет составлять основную часть программы

**Решение:**

1. по условию, единственная информация, которая нам нужна в итоге для вывода результата – это количество участников по каждой школе
2. так как номер школы состоит (по условию!) не более, чем из двух цифр, всего может быть не более 99 школ (с номерами от 1 до 99)
3. поэтому можно ввести массив **C** из 99 элементов; для всех **k** от 1 до 99 элемент **C[k]** будет ячейкой-счетчиком, в которой накапливается число участников от школы с номером **k**; сначала во все элементы этого массива записываются нуль (обнуление счетчиков):

**for k:=1 to 99 do C[k]:=0;**

во многих системах программирования на Паскале все глобальные переменные автоматически обнуляются, и таким образом, этот цикл ничего не дает; однако на всякий случай нужно продемонстрировать эксперту, который будет проверять часть **С** вашей работы, что вы понимаете суть дела («счетчик необходимо сначала обнулить»)

1. основной цикл обработки вводимых строк можно записать на псевдокоде так:

**for i:=1 to N do begin**

**{ читаем очередную строку }**

**{ определяем номер школы k }**

**C[k] := C[k] + 1; { увеличиваем счетчик k-ой школы }**

**end;**

1. поскольку данные вводятся в виде символьной строки, нужно выделить в памяти переменную **s** типа **string**
2. для чтения очередной строки будем использовать оператор **readln**
3. остается понять, как выделить из строки номер школы; по условию он закодирован в последней части строки, после второго пробела; значит, нужно найти этот второй пробел, вырезать из строки весь «хвост» после этого пробела, и преобразовать его из символьного формата в числовой
4. чтобы найти первый пробел и «отрезать» первую часть строки с этим пробелом, можно использовать команды

**p := Pos(' ', s);**

**s := Copy(s, p+1, Length(s)-p);**

первая команда определяет номер первого пробела и записывает его в целую переменную **p**, в вторая – записывает в строку **s** весь «хвост», стоящий за этим пробелом, начиная с символа с номером **p+1**; длина хвоста равна **Length(s)-p**, где **Length(s)** – длина строки;

1. поскольку нас интересует часть после **второго** пробела, эти две строчки нужно повторить два раза, в результате в переменной **s** окажется символьная запись номера школы;
2. заметим, что можно избежать дублирования двух строк, «свернув» их во внутренний цикл, но это вряд ли сильно упростит запись:

**for k:=1 to 2 do begin**

**p := Pos(' ', s);**

**s := Copy(s, p+1, Length(s)-p);**

**end;**

1. в пп. 8-10 описан достаточно общий метод, при котором инициалы могут быть любой длины, (но без пробела); в данном случае в условии четко сказано, что инициалы представляют собой именно 4 символа (буква, точка, буква, точка), поэтому можно найти первый пробел, а затем взять «хвост», который идет через 6 символов от него:

|  |  |  |
| --- | --- | --- |
| **p := Pos(' ', s);**  **s := Copy(s,p+6,Length(s));** | или так | **p := Pos(' ', s);**  **Delete(s, 1, p+5);** |

1. для преобразования номера школы из символьного вида в числовой можно использовать функцию **Val**:

**Val(s, k, r);**

эта процедура (*Turbo Pascal, Borland Pascal, PascalABC,* среда *АЛГО*) преобразует символьную строку **s** в числовое значение **k**; с помощью переменной **r** обнаруживается ошибка: если раскодировать число не удалось (в строке не число), в **r** будет записан нуль (здесь мы не будем обрабатывать эту ошибку, полагая, что все данные правильные);

если вы работаете на *ПаскалеABC* (никто не может вам запретить написать, что этот так), вместо **Val** можно использовать более удобную и понятную функцию **StrToInt**:

**k := StrToInt(s);**

1. таким образом, основной цикл выглядит так:

**for i:=1 to N do begin**

**readln(s); { читаем очередную строку }**

**{ выделяем часть после второго пробела }**

**p := Pos(' ', s);**

**Delete(s, 1, p+5);**

**{ определяем номер школы k }**

**Val(s, k, r);**

**C[k] := C[k] + 1; { увеличиваем счетчик k-ой школы }**

**end;**

1. дальше стандартным алгоритмом определяем в массиве **C** минимальный элемент **Min**, не учитывая нули (школы, из которых не было участников):

**Min := N;**

**for k:=1 to 99 do**

**if (C[k] <> 0) and (C[k]<Min) then Min := C[k];**

здесь интересна первая строчка, **Min:=N**: по условию всего было **N** участников, поэтому минимальное значение не может быть больше **N**; обратите внимание, что привычный вариант (который начинается с **Min:=C[1]**) работает неверно, если из первой школы не было ни одного участника

1. и выводим на экран номера всех школ (обратите внимание – **номера!**), для которых **C[k]=Min**:

**for k:=1 to 99 do**

**if C[k] = Min then writeln(k);**

1. остается «собрать» программу, чтобы получилось полное решение; максимальное количество школ мы задали в виде константы **LIM**:

**const LIM = 99;**

**var C:array[1..LIM] of integer;**

**i, p, N, k, r, Min: integer;**

**s:string;**

**begin**

**for k:=1 to 99 do C[k]:=0;**

**readln(N);**

**for i:=1 to N do begin**

**readln(s); { читаем очередную строку }**

**{ выделяем часть после второго пробела }**

**p := Pos(' ', s);**

**Delete(s, 1, p+5);**

**{ определяем номер школы k }**

**Val(s, k, r);**

**C[k] := C[k] + 1; { увеличиваем счетчик k-ой школы }**

**end;**

**Min := N;**

**for k:=1 to LIM do**

**if (C[k] <> 0) and (C[k]<Min) then Min := C[k];**

**for k:=1 to LIM do**

**if C[k] = Min then writeln(k);**

**end.**

|  |
| --- |
| **На что обратить внимание:**   * + внимательно читайте условие, убедитесь, что вы понимаете смысл каждой строчки; для каждой мелочи постарайтесь определить, зачем она добавлена в условие, что она дает для решения задачи, что ограничивает, что не разрешает делать   + определите, какая именно информация из условия нужна для решения задачи, а какая – не нужна   + определите, что именно требуется вывести на экран в результате работы программы   + начинайте составлять программу с больших блоков, записывая ее сначала на псевдокоде, а потом уточняя детали   + проверяйте «крайние» варианты (например, возможность выхода за границы массива)   + проверьте, правильно ли заданы (и заданы ли вообще) начальные значения для всех переменных   + будьте внимательны, когда в массиве есть «мертвые» элементы, которые не нужно учитывать; проверяйте, что в этом случае ваши алгоритмы (например, поиск минимального элемента) работают правильно   + проверьте, правильно ли расставлены операторные скобки **begin-end**, ограничивающие тело цикла; их обязательно нужно ставить, если в теле цикла несколько операторов   + при использовании функции **Pos** не забывайте, что первый параметр – **что** ищем (образец), а второй – **где** ищем   + чтобы эксперту было легче понять вашу программу (особенно, если она получилась «нестандартной»), пишите комментарии; объясняйте, что хранится в основных переменных   + если это возможно, желательно работать только с целыми числами; этим вы избежите проблем, связанных с округлением и неточностью хранения дробных вещественных чисел в памяти компьютера |

### Еще пример задания:

*На вход программе подаются сведения о сдаче экзаменов учениками 9-х классов некоторой средней школы. В первой строке сообщается количество учеников N, которое не меньше 10, но не превосходит 100, каждая из следующих N строк имеет следующий формат:*

**<Фамилия> <Имя> <оценки>,**

*где <Фамилия> – строка, состоящая не более чем из 20 символов, <Имя> – строка, состоящая не более чем из 15 символов, <оценки> – через пробел три целых числа, соответствующие оценкам по пятибалльной системе. <Фамилия> и <Имя>, а также <Имя> и <оценки> разделены одним пробелом. Пример входной строки:*

**Иванов Петр 4 5 3**

*Требуется написать как можно более эффективную программу (укажите используемую версию языка программирования, например, Borland Pascal 7.0), которая будет выводить на экран фамилии и имена трех худших по среднему баллу учеников. Если среди остальных есть ученики, набравшие тот же средний балл, что и один из трех худших, то следует вывести и их фамилии и имена.*

**Как правильно понимать условие?**

1. как и в предыдущей задаче, данные «подаются на вход программе», то есть, их можно читать с помощью операторов **read** (**readln**), предполагая, что кто-то вводит эти данные с клавиатуры вручную
2. «количество учеников не меньше 10, но не превосходит 100», здесь только вторая часть – полезная информация, она намекает на то, что придется все введенные данные одновременно держать в памяти, выделив массив (или массивы) размером 100 элементов
3. сказано, что фамилия имеет длину не более 20 символов, а имя – не более 15; здесь, по сути, важно лишь то, что фамилия и имя (вместе) занимают меньше 255 символов, то есть, «влезут» в стандартное ограничение (255 символов) для типа **string** в классических версиях Паскаля
4. после фамилии и имени записаны три оценки (а не одно число, как в прошлой задаче), причем по условию нас НЕ интересуют эти числа, а интересует только средний балл каждого ученика;
5. важно! средний балл – это вещественное число (может иметь дробную часть), тут уже стоит задуматься: все задачи обычно составляются так, чтобы они решались «хорошо», в то же время операции с дробными числами (почти) всегда выполняются с ошибками, поскольку большинство вещественных чисел нельзя точно (стандартными методами) представить в памяти реального компьютера
6. следующий шаг к правильному решению: поскольку число оценок у всех учеников одинаковое, средний балл для каждого это сумма его оценок, деленная на 3; поэтому вместо среднего балла мы можем сравнивать суммы баллов – целые числа!
7. требуется вывести фамилии и имена (баллы не нужны!) трех худших учеников, причем их может быть и больше, если несколько «худших» набрали одинаковую сумму баллов
8. если бы требовался один худший – все решается поиском по массиву; первая идея – найти самого худшего (1 проход), затем – 2-ого с конца (еще 1 проход), и, наконец, 3-его (всего три прохода по массиву)
9. это не лучший вариант (на экзамене будут сняты баллы) по двум причинам:

* в таком методе решения три прохода по массиву, а в самом деле достаточно одного (см. далее), значит, программа неэффективна
* непонятно, что делать в том случае, если худших – больше трех (в предельном случае – вообще все!) – за это также снимут баллы (программа работает не для всех вариантой входных данных)

1. возникает следующий вариант – отсортировать массив про возрастанию суммы (и, следовательно, среднего балла), одновременно переставляя имена и фамилии, а затем вывести самых худших, которые после сортировки окажутся в начале массива
2. этот вариант тоже плох, потому что программа неэффективна; «школьные» алгоритмы сортировки (метод «пузырька», метод выбора) имеют сложность , а надо попытаться найти метод со сложностью 

**Решение (общий подход):**

1. сначала составим программу в самом общем виде на псевдокоде, чтобы определить ее основные блоки, а потом будем их постепенно «расшифровывать» через операторы языка программирования:

**{ читаем все данные и запоминаем их }**

**{ находим три худших результата }**

**{ выводим фамилии и имена тех, чей результат меньше или**

**равен «третьему худшему» }**

1. до того, как начать писать «нормальный» код, нужно определить, как хранить данные; в данном случае нужно запомнить несколько данных по каждому ученику, их удобнее объединить в запись с двумя полями (фамилия-имя и сумма баллов); таких записей нужно выделить в памяти не менее 100 (по условию), то есть, массив из 100 элементов:

**const LIM=100;**

**var Info: array[1..LIM] of record**

**name: string;**

**sum: integer;**

**end;**

**Чтение данных:**

1. после того, как мы прочитали фактическое число учеников N, в цикле считываем и расшифровываем информацию о них, сохраняя все данные в структурах

**for i:=1 to N do begin**

**{ считываем строку данных }**

**Info[i].name := { фамилия и имя };**

**Info[i].sum := { сумма баллов };**

**end;**

1. здесь, в принципе, можно использовать тот же подход, что и в первой задаче – читаем строку целиком, затем «разбираем» ее на части с помощью стандартных функций – однако, для разнообразия, мы используем другой подход – будем читать информацию **посимвольно**, то есть, считывая по одному символу в переменную **c** типа **char**;
2. сначала в поле **name** очередной структуры записываем пустую строку **''**(в которой нет ни одного символа, длина равна нулю)

**Info[i].name := ''; { пустая строка }**

1. затем считываем символы фамилии и сразу приписываем их в конец поля **name**:

**repeat**

**read ( c );**

**Info[i].name := Info[i].name + c;**

**until c = ' '; { пока не прочитали пробел }**

1. затем также читаем из входного потока имя, до пробела, и записываем его в конец того же поля **name**:

**repeat**

**read ( c );**

**Info[i].name := Info[i].name + c;**

**until c = ' '; { пока не прочитали пробел }**

заметьте, что эти два цикла одинаковы, поэтому ввод имени и фамилии можно записать в виде вложенного цикла так:

**Info[i].name := ''; { пустая строка }**

**for k:=1 to 2 do**

**repeat**

**read ( c );**

**Info[i].name := Info[i].name + c;**

**until c = ' '; { пока не прочитали пробел }**

1. важно! обратите внимание, что для организации внутреннего цикла используется другая переменная, **k** (а не **i**, потому что **i** – переменная главного цикла, она обозначает номер текущего ученика)
2. теперь во входном потоке остались три числа, которые мы можем последовательно считывать в целую переменную **mark**, а затем – добавлять к полю **Info[i].sum**:

**Info[i].sum := 0;**

**for k:=1 to 3 do begin**

**read(mark);**

**Info[i].sum := Info[i].sum + mark;**

**end;**

**readln;**

1. последняя команда **readln** пропускает все оставшиеся символы до новой строки (из этой мы прочитали все, что нужно)
2. вот полный цикл ввода данных, после его окончания все исходные данные будут записаны в первые **N** записей массива **Info**:

**for i:=1 to N do begin**

**{ ввод имени и фамилии }**

**Info[i].name := '';**

**for k:=1 to 2 do**

**repeat**

**read(c);**

**Info[i].name := Info[i].name + c;**

**until c = ' ';**

**{ ввод и суммирование оценок }**

**Info[i].sum := 0;**

**for k:=1 to 3 do begin**

**read(mark);**

**Info[i].sum := Info[i].sum + mark;**

**end;**

**readln;**

**end;**

**Поиск трех худших данных:**

1. теперь нужно придумать, как за один проход по массиву найти три худших результата;
2. как бы мы решили эту задачу, если бы нам нужно было просмотреть столбик чисел и найти три минимальных? можно сделать, например, так:

* на бумажке вести записи в три столбика, в первом записывать минимальное число, в втором – следующее по величине, в третьем – «третье минимальное»
* сначала пишем первое число в первый столбик, оно – минимальное, потому что других мы не еще видели; пусть это число 14:

|  |  |  |
| --- | --- | --- |
| минимум | второе | третье |
| 14 |  |  |

* пусть следующее число – 12; оно меньше минимального, поэтому его нужно записывать в первый столбец, а «старое» минимальное число «переедет» во второй столбец

|  |  |  |
| --- | --- | --- |
| минимум | второе | третье |
| ~~14~~ |  |  |
| 12 | 14 |  |

* пусть дальше идет число 10 – теперь оно станет минимальным, его нужно записывать в первый столбец; при этом 12 «переедет» из первого столбца во второй, а 14 – из второго в третий

|  |  |  |
| --- | --- | --- |
| минимум | второе | третье |
| ~~14~~ |  |  |
| ~~12~~ | ~~14~~ |  |
| 10 | 12 | 14 |

* пусть следующее число – 11; оно больше минимального, но меньше «второго», поэтому его нужно поставить во второй столбец; число 12 из второго столбца перемещается в третий, а число 14 из третьего столбца удаляется из кандидатов в «три минимальных»

|  |  |  |
| --- | --- | --- |
| минимум | второе | третье |
| ~~14~~ |  |  |
| ~~12~~ | ~~14~~ |  |
| 10 | ~~12~~ | ~~14~~ |
|  | 11 | 12 |

* просмотрев таким образом весь столбик чисел, за один проход (!) можно найти три минимальных элемента
* остается только переложить этот алгоритм на язык программирования

1. выделим в памяти три целых переменных: **min1** (минимальный), **min2** («второй минимальный»), **min3** («третий минимальный»), в виде начальных значений запишем в каждую из них число, заведомо превышающее максимальную возможную сумму трех оценок, например, 20 (**>5+5+5**)
2. полный цикл поиска выглядит так:

**min1 := 20; min2 := 20; min3 := 20;**

**for i:=1 to N do begin**

**if Info[i].sum < min1 then begin { новый min1 }**

**min3 := min2; min2 := min1;**

**min1 := Info[i].sum;**

**end**

**else if Info[i].sum < min2 then begin { новый min2 }**

**min3 := min2;**

**min2 := Info[i].sum;**

**end**

**else if Info[i].sum < min3 then { новый min3 }**

**min3 := Info[i].sum;**

**end;**

1. обратим внимание на два момента: во-первых, когда переезжают два элемента, сначала нужно перемещать второй на место третьего, а потом – первый на место второго:

**min3 := min2;**

**min2 := min1;**

эти операторы нельзя менять местами, иначе «старое» значение **min2** будет потеряно;

во-вторых, если проверять условие **Info[i].sum < min2** нужно только тогда, когда очередная сумма не меньше, чем **min1**, поэтому каждый следующий условный оператор стоит в **else**-блоке предыдущего, то есть, выполняется только тогда, когда предыдущий не сработал

1. итак, мы нашли три минимальных результата, и остается вывести на экран фамилии и имена тех, у кого сумма баллов меньше или равна **min3**:

**for i:=1 to N do**

**if Info[i].sum <= min3 then**

**writeln(Info[i].name);**

1. на всякий случай приведем полную программу, она получилась довольно длинная

**const LIM = 100;**

**var Info: array[1..LIM] of record**

**name: string;**

**sum: integer;**

**end;**

**i, k, N, mark, min1, min2, min3: integer;**

**c: char;**

**begin**

**readln(N);**

**{ ввод исходных данных }**

**for i:=1 to N do begin**

**Info[i].name := '';**

**for k:=1 to 2 do**

**repeat**

**read(c);**

**Info[i].name := Info[i].name + c;**

**until c = ' ';**

**Info[i].sum := 0;**

**for k:=1 to 3 do begin**

**read(mark);**

**Info[i].sum := Info[i].sum + mark;**

**end;**

**readln;**

**end;**

**{ поиск трех минимальных }**

**min1 := 20; min2 := 20; min3 := 20;**

**for i:=1 to N do begin**

**if Info[i].sum <min1 then begin**

**min3 := min2; min2 := min1;**

**min1 := Info[i].sum;**

**end**

**else if Info[i].sum <min2 then begin**

**min3 := min2;**

**min2 := Info[i].sum;**

**end**

**else if Info[i].sum <min3 then**

**min3 := Info[i].sum;**

**end;**

**{ вывод результата }**

**for i:=1 to N do**

**if Info[i].sum <= min3 then**

**writeln(Info[i].name);**

**end.**

1. эту задачу можно решить и без записей, используя два массива: массив символьных строк **name** и массив целых чисел **sum**, они объявляются так:

**var name: array[1..MAX] of string;**

**sum: array[1..MAX] of integer;**

после этого в приведенной программе нужно заменить везде **Info[i].name** на **name** и **Info[i].sum** на **sum.**

|  |
| --- |
| **На что обратить внимание:**   * + в исходных данных выделите то, что не нужно для решения задачи; при чтении эти части можно просто пропускать;   + если нам не нужны фамилия и имя отдельно, можно хранить их вместе, в виде одной строки   + если нас интересует только сумма оценок, не нужно хранить их в памяти по отдельности   + если можно при решении задачи обойтись без вещественных чисел, сделав все вычисления только с целыми числами – нужно поступить именно так (иначе снимут баллы), поскольку операции с вещественными числами во многих случаях случаев выполняются неточно   + алгоритм сложности  (например, сортировку) нужно использовать только тогда, когда нет алгоритма сложности ; как правило, в задачах ЕГЭ такой алгоритм всегда можно (попытаться) найти; за неэффективный алгоритм при оценке решения будут сняты баллы |

|  |
| --- |
| **За что снимают баллы:**   * + программа работает не для всех исходных данных, не обрабатывает некоторые частные случаи   + неверно реализован алгоритм поиска минимального элемента, сортировки и т.п.   + неэффективность алгоритма:   + используется алгоритм, имеющий сложность , когда есть алгоритм сложности   + используется несколько проходов по массиву, когда достаточно одного   + лишний расход памяти ( используются дополнительные массивы или размер массива определен неверно)   + используются операции с вещественными числами, когда можно все решить в целых числах   + переменная не описана или описана неверно   + переменным не присвоены нужные начальные значения (например, не обнуляются счетчики) или присвоены неверные значения   + нет вывода результата в конце программы   + перепутаны знаки < и >, логические операции **or** и **and**   + применяется недопустимая операция, например, **div** или **mod** для вещественных чисел   + неверно расставлены операторные скобки **begin-end**   + в цикле **for** используется вещественная переменная (Паскаль)   + в цикле **while** или **repeat** не изменяется переменная цикла, из-за чего происходит зацикливание   + синтаксические ошибки (знаки пунктуации – запятые, точки, точки с запятой; неверное написание ключевых слов); чтобы получить 4 балла, при абсолютно верном решении нужно сделать не более одной синтаксической ошибки; на 3 балла – до трех ошибок, на 2 балла – до пяти и на 1 балл – до семи ошибок |

1. Самылкина Н.Н., Островская Е.М. Информатика: тренировочные задания. – М.: Эксмо, 2009. [↑](#footnote-ref-1)
2. Для разделения имен каталогов в адресе в разных операционных системах применяют прямой слэш «/» или обратный слэш «\». В системе *Windows*, которая наиболее распространена в России, стандартным разделителем считается «\», именно такой знак чаще всего используется в задачах ЕГЭ. [↑](#footnote-ref-2)
3. Интересующиеся могут посмотреть на коды русских букв в кодировке КОИ-8R

   и ужаснуться, осознав, что было бы при использовании букв В и Г. [↑](#footnote-ref-3)
4. Крылов С.С., Ушаков Д.М. ЕГЭ 2010. Информатика. Тематическая рабочая тетрадь. — М.: Экзамен, 2010. [↑](#footnote-ref-4)
5. Используя ассемблер (язык машинных кодов с символьными командами), можно добраться до бита переноса и использовать его. [↑](#footnote-ref-5)
6. Кроме *логического* сдвига вправо, о котором идет речь, есть еще *арифметический*, при котором старший бит не меняется. [↑](#footnote-ref-6)
7. Здесь считается, что 1 Кбит = 1024 бит = 210 бит. [↑](#footnote-ref-7)
8. О.Б. Богомолова, Д.Ю. Усенков. Задача о передаче: решение задачи ЕГЭ с помощью диаграмм Ганта // Информатика, № 7, 2011. [↑](#footnote-ref-8)
9. Часто килобайт обозначают «Кб», а мегабайт – «Мб», но в демо-тестах разработчики ЕГЭ привели именно такие обозначения. [↑](#footnote-ref-9)
10. Фактически это не другой способ решения, а более строгое обоснование предыдущего алгоритма. [↑](#footnote-ref-10)
11. Т.Е. Чуркина. ЕГЭ. Информатика. Практикум по выполнению типовых тестовых заданий ЕГЭ.М.: Экзамен, 2010. [↑](#footnote-ref-11)
12. Такая процедура называется *топологической сортировкой графа*. [↑](#footnote-ref-12)
13. Каждая следующая область в полученном решении должна полностью включать предыдущую. Если это не так, тогда или вы ошиблись при построении таблицы истинности, или (не дай Бог!) в условии есть ошибка. [↑](#footnote-ref-13)
14. Как мы увидим далее, при использовании других методов решения, это условие принципиально облегчает решение данной задачи. Во всех известных автору вариантах подобных задач такое упрощающее условие было. [↑](#footnote-ref-14)
15. Источники заданий:

    Демонстрационные варианты ЕГЭ 2004-2013 гг.

    Тренировочные работы МИОО.

    Гусева И.Ю. ЕГЭ. Информатика: раздаточный материал тренировочных тестов. — СПб: Тригон, 2009.

    Якушкин П.А., Ушаков Д.М. Самое полное издание типовых вариантов реальных заданий ЕГЭ 2010. Информатика. — М.: Астрель, 2009.

    Чуркина Т.Е. ЕГЭ 2011. Информатика. Тематические тренировочные задания. — М.: Эксмо, 2010.

    Якушкин П.А., Лещинер В.Р., Кириенко Д.П. ЕГЭ 2011. Информатика. Типовые тестовые задания. — М.: Экзамен, 2011. [↑](#footnote-ref-15)
16. **Огастес (Август) де Морган –** шотландский математик и логик. [↑](#footnote-ref-16)
17. … но которая, к сожалению, почти не нужна на практике. ☺ [↑](#footnote-ref-17)
18. Здесь рассматривается только язык Паскаль, который является наиболее распространенным в школах России. [↑](#footnote-ref-18)
19. В этом примере используется стандартная нумерация для Паскаля: индексы начинаются с единицы. [↑](#footnote-ref-19)
20. Метод отображений предложен Ел.А. Мирончик и Ек.А. Мирончик ([http://kpolyakov.spb.ru/download/b15mirn.zip](http://kpolyakov.narod.ru/download/b15mirn.zip)). [↑](#footnote-ref-20)
21. Метод отображений предложен Ел.А. Мирончик и Ек.А. Мирончик ([http://kpolyakov.spb.ru/download/b15mirn.zip](http://kpolyakov.narod.ru/download/b15mirn.zip)). [↑](#footnote-ref-21)
22. Метод отображений предложен Ел.А. Мирончик и Ек.А. Мирончик ([http://kpolyakov.spb.ru/download/b15mirn.zip](http://kpolyakov.narod.ru/download/b15mirn.zip)). [↑](#footnote-ref-22)
23. Метод отображений предложен Ел.А. Мирончик и Ек.А. Мирончик ([http://kpolyakov.spb.ru/download/b15mirn.zip](http://kpolyakov.narod.ru/download/b15mirn.zip)). [↑](#footnote-ref-23)
24. Метод отображений предложен Ел.А. Мирончик и Ек.А. Мирончик ([http://kpolyakov.spb.ru/download/b15mirn.zip](http://kpolyakov.narod.ru/download/b15mirn.zip)). [↑](#footnote-ref-24)
25. Метод отображений предложен Ел.А. Мирончик и Ек.А. Мирончик ([http://kpolyakov.spb.ru/download/b15mirn.zip](http://kpolyakov.narod.ru/download/b15mirn.zip)). [↑](#footnote-ref-25)
26. Проверьте, что обычно (когда комбинации располагаются по возрастанию соответствующих двоичных чисел), столбец значений аргумента А представляет собой двоичную запись числа 15 = 11112, столбец значений аргумента В – числа 51 = 1100112, столбец значений аргумента С – числа 85 = 101010102. [↑](#footnote-ref-26)
27. Далее при разборе задачи используется язык Паскаль, который наиболее распространен в школах России. [↑](#footnote-ref-27)
28. По традиции нумерация элементов массива в Паскале обычно начинается с единицы, далее N обозначает размер массива (количество элементов). [↑](#footnote-ref-28)
29. Вообще говоря, в данной задаче не требуется находить номер минимального элемента, поэтому сначала можно записать в переменную min число 1000 – проверьте, что программа все равно выдаст верное значение. [↑](#footnote-ref-29)
30. Или используется перенаправление входного потока из командной строки, но это уже абсолютно неважно… [↑](#footnote-ref-30)