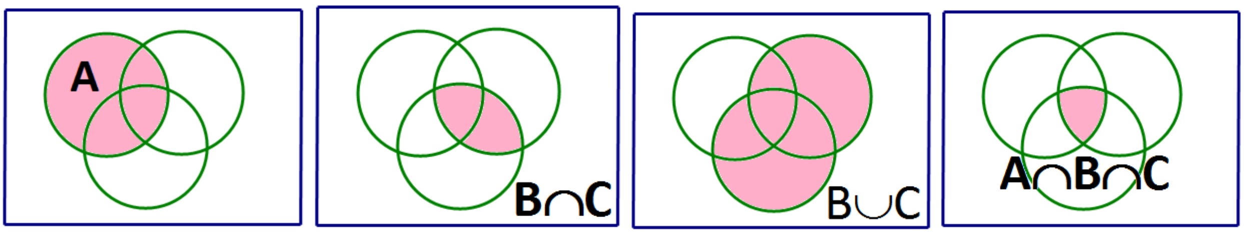
ЛОГИЧЕСКИЕ СХЕМЫ

1 уровень (6 баллов)
Задача «Отрицание 1-2»
Построить логическую схему из одного элемента NOT и любого числа элементов XOR, AND и OR, которая имеет два входа x и y и два выхода, на которые должны поступать сигналы NOT x и NOT y. (ответ можно представить как в форме логической схемы, так и в форме логического выражения)
Решение:
z = x XOR y
u = NOT z = 1 XOR x XOR y = 1+x+y
v = y XOR u = y+1+x+y = 1+x = NOT x
w = x XOR u = x+1+x+y = 1+y = NOT y
v и w – выходы

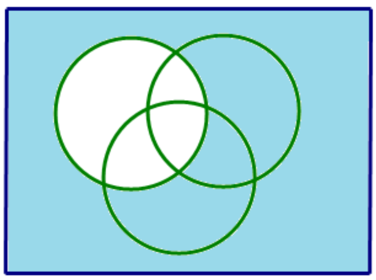
2 **уровень (11 баллов)**
Задача «Отрицание 2-3»
Построить логическую схему из двух элементов NOT и любого числа элементов AND и OR, которая имеет три входа x, y и z и три выхода, на которые должны поступать сигналы NOT x, NOT y, NOT z.
(ответ можно представить как в форме логической схемы, так и в форме логического выражения)

Решение:
Для наглядного решения этой задачи полезны диаграммы Эйлера, которые позволяют изобразить основные логические операции графически.
Сопоставим переменной x множество точек A (изобразим их в виде круга), переменной y – множество B, переменной z– множество C. Поместим эти множества внутрь пространства точек E (внутренность прямоугольника), тогда дополнениям множеств A, B и C до E будут соответствовать отрицания переменных: NOT x, NOT y и NOT z (эти множества естественным образом интерпретируются, когда мы рассматриваем предикаты — высказывания с переменными, тогда, например, множество A состоит из точек, где логическое утверждение x истинно и т. д.).



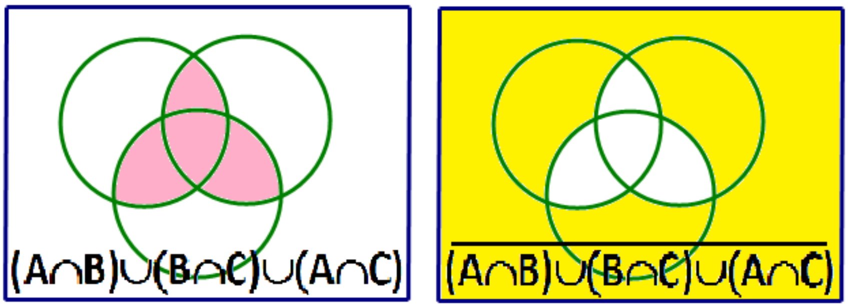
Тогда утверждению y AND z будет соответствовать пересечение множеств A и B, а утверждению y OR z – объединение и т. д.

Нужно построить три множества, первое из которых (дополнение A) построить выглядит так (отмечено голубым цветом):



Надо суметь его построить, используя вышеперечисленные операции над множествами и операцию дополнения множества, которая соответствует логической операции отрицания.

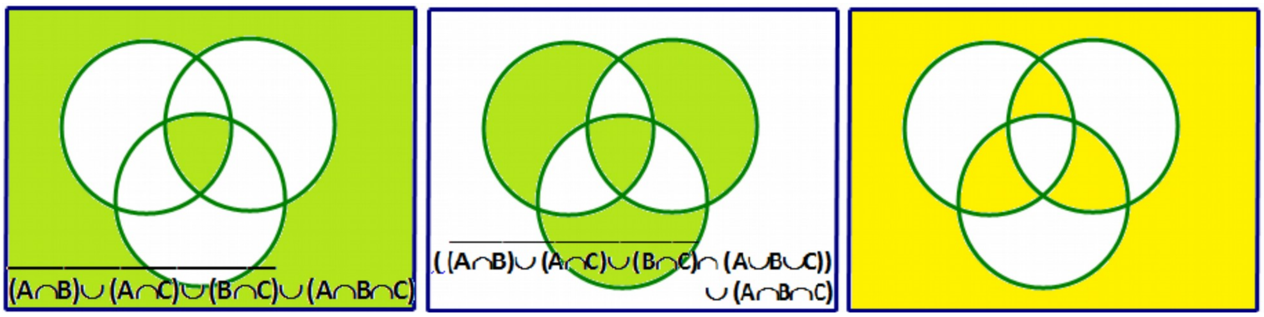
Ниже показан пример с операцией дополнения:



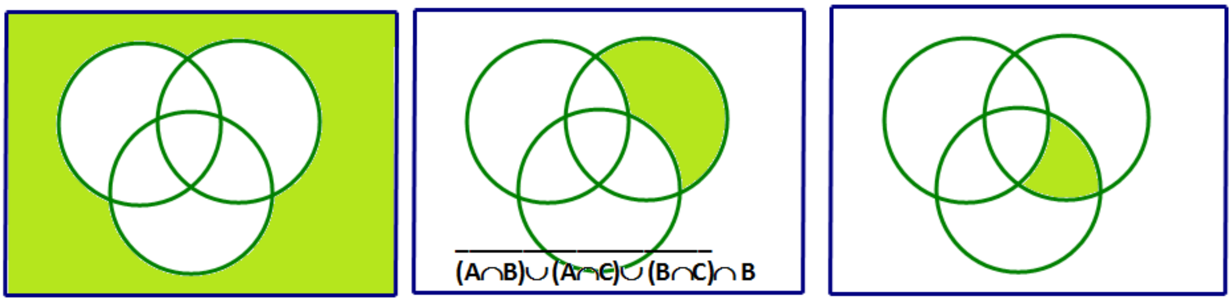
Перейдём к построению решения.

Обозначим множество на верхнем рисунке справа D1— это дополнение множества, изображенного на рисунке слева.

Используя операции пересечения и объединения, а затем ещё одно отрицание, нетрудно построить следующие множества:



Справа показано множество D2 - дополнение множества на среднем рисунке



На рисунке слева показано пересечение множеств D1 и D2 (обозначим его T), на среднем рисунке показано пересечение множеств D1 и B (обозначим его Tb), на рисунке справа показано пересечение множеств T2 с B и C (обозначим его Tbc).

Теперь нетрудно собрать из трех типов кусочков, показанных на последнем рисунке дополнение каждого из множеств A, B и C. Например, дополнением A будет объединение множеств T, Tb, Tc и Tbc.

Теперь перейдем от диаграмм Эйлера к записи логических выражений им соответствующих:

D1 будет соответствовать выражение d1=NOT ((x AND y) OR (x AND z) OR (y AND z))
D2 будет соответствовать выражение d2=NOT (d1 AND (x OR y OR z) OR (x AND y AND z))
T будет соответствовать выражение t = d1 AND d2
Tb будет соответствовать выражение t_b= d1 AND y
Tc будет соответствовать выражение t_c = d1 AND z
Tbc будет соответствовать выражение t_{bc}= d2 AND y AND z

Итак на первый выход подается сигнал описываемый функцией $u = t$ OR t_b OR t_c OR t_{bc}

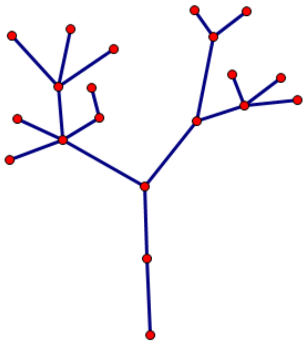
Аналогично, на второй и третий выходы, подаются сигналы:
$v = t$ OR t_a OR t_c OR t_{ac}

$w = t$ OR t_a OR t_b OR t_{ab}
где t_a= d1 AND x, t_{ac}= d2 AND x AND z, t_{ab}= d2 AND x AND y
u, v и w – выходы

ГРАФЫ

Задача «Экономный сбор информации» - вершинное покрытие графа

Уровень 1 (3 балла)

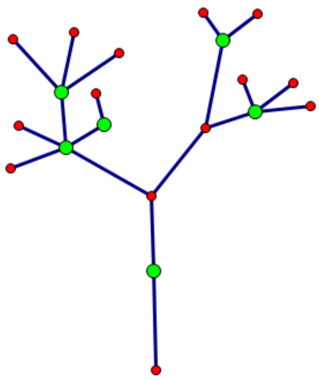


На рисунке показана дерево иерархической подчиненности нескольких организаций, изображенных точками. Нижняя вершина — корень дерева — является головной организацией. Статистическому бюро нужно собрать сведения о всех организациях. Каждая организация может предоставить сведения о себе, а также тех организациях, которые находятся в её непосредственном подчинении и той организации, в непосредственном подчинении которой она сама находится (то есть на графе — это сама вершина и соседние с ней). *Какие из организаций достаточно опросить, чтобы получить информацию о всех организациях, сделав наименьшее число запросов?* Такой набор вершин называется наименьшим вершинным покрытием графа. Тогда вопрос задачи будет звучать так: *найдите все наименьшие вершинные покрытия заданного в условии дерева.*

Примечание. Вершинным покрытием называется подмножество вершин графа, содержащее по крайней мере одного соседа каждой вершины, не входящей в это множество. Наименьшим покрытием будем называть покрытие с наименьшим числом вершин.

Уровень 2 (6 баллов)
Описать алгоритм, который строит какое-либо наименьшее вершинное покрытие любого дерева и обосновать его корректность.
Решение
1) наименьшее вершинное покрытие состоит из 6; одно из решений показано на рисунке.
2) Алгоритм:

Начать с листьев дерева (лист - это вершина, являющаяся концом только одного ребра — иными словами, это организация, у которых нет подчиненных) и занести в вершинное покрытие те вершины, которые связаны с листьями ребром - их родителей (то есть, запросить отчёты от всех организаций, которым они непосредственно подчиняются), затем удалить из дерева все его листья, их родителей, родителей родителей и рёбра, являющиеся их концами.



Затем удалить из дерева листья, их непосредственных предков и все рёбра, концами которых являются эти вершины (рёбра, инцидентные этим вершинам).
Получим несколько деревьев (лес) с меньшим числом рёбер, к которым можно применить этот же алгоритм.

Предложенный алгоритм строит наименьшее вершинное покрытие. Для доказательства отметим, что если в наименьшее покрытие входит лист дерева, то можно построить другое наименьшее покрытие, в которое этот лист не входит. Рассмотрим родителя этого листа, если он входит в вершинное покрытие, то лист можно убрать из покрытия и число вершин в нём уменьшится, что противоречит тому, что оно является наименьшим. Если не входит, то мы можем включить её в покрытие, исключив лист и число вершин в покрытии не изменится. Таким образом в построенное наименьшее покрытие всегда войдут все родители листьев. Тогда они покроют не только листья, но и родителей родителей, которые поэтому можно исключить из рассмотрения. Именно это и делает предложенный алгоритм. Применяя метод математической индукции по числу вершин в дереве, получаем искомый результат.

Работу алгоритма полезно проверить на примерах из задачи по комбинаторике, в которой подсчитывается число вершинных покрытий одинарного дерева.

КОМБИНАТОРИКА

Уровень 1 (3 балла)

В задаче на графы строились наименьшие вершинные покрытия деревьев. В этой задаче требуется *найти количество M(n) вершин в наименьших вершинных покрытиях цепи с n вершинами* (на рисунке показана цепь с 4 вершинами, в ней можно выбрать две средние вершины в качестве вершинного покрытия, но одной любой вершины уже не хватит, поэтому M(4)=2).

Ответ

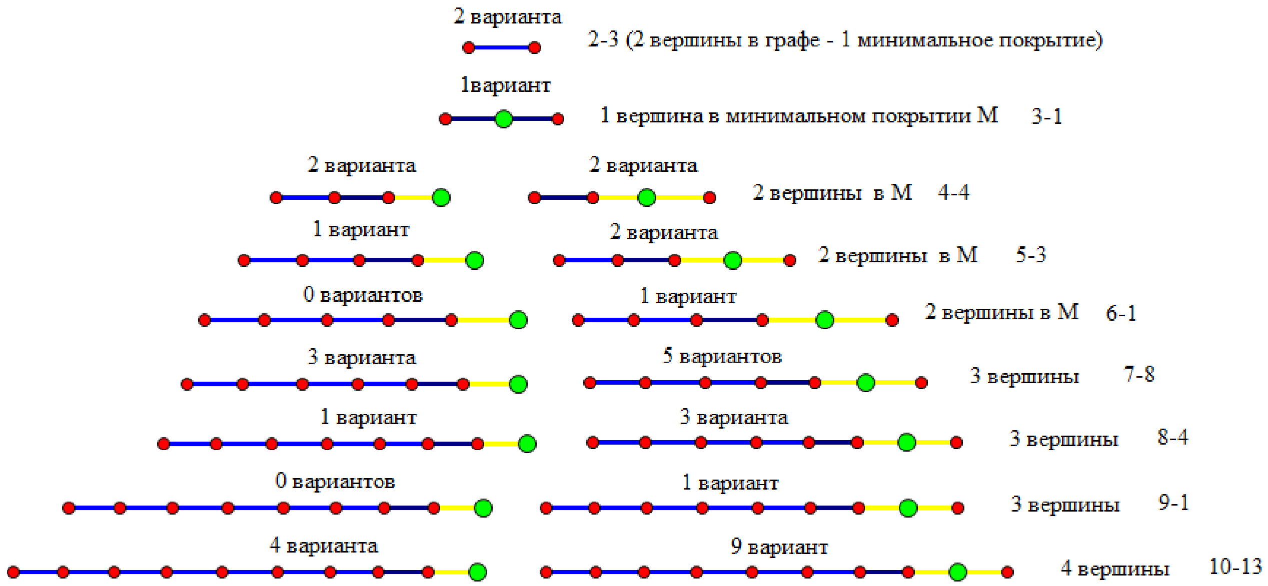
Количество вершин в минимальном вершинном покрытии M(3n)=n, M(3n-1)=n, M(3n-2)=n или с использованием операции взятия целой части M(n)=[(n+2)/3].

Уровень 2 (9 баллов)

Сколько различных наименьших покрытий цепь с n вершинами? Найдите число покрытий N(n) для n=1, 2, 3, 4, 5, 6, 7, 8, 9, 10.

Решение

Введём обозначение N(n;k) – число наименьших покрытий k вершинами цепи с n вершинами. Тогда, например, N(4;2)=4, а N(4;1)=0.



Уровень 3 (12 баллов)

Выведите рекуррентную формулу для N(n).

Решение

Разобьём все наименьшие покрытия цепи на следующие случаи:

1) в покрытие входит крайняя справа вершина и не входит соседняя с ней,
2) в покрытие входит следующая за крайней справа вершина, но входят соседние с ней,
3) в покрытие входят две крайние справа вершины, но не входит соседняя с ними вершина слева,
4) в покрытие не входит крайняя вершина, но входят две следующие за ней.

Это разбивает варианты, на группы с меньшими числами вершин:
N(n;k)=N(n-2;k-1)+N(n-3;k-1)+N(n-3;k-2)+N1
где последний случай требует аналогичного разбиения на группы, в зависимости от того, входит ли в покрытие, соседняя (с уже имеющимися двумя) слева вершина.

Однако нетрудно доказать, что три соседние вершины не могут входить в минимальное покрытие, так как среднюю из них можно тогда отбросить, уменьшив число вершин в покрытии. Поэтому четвёртый вариант может быть переписан в такой форме:
4') в покрытие не входит крайняя вершина, но входят две следующие за ней, причём соседние с этой парой вершин в покрытие не входят, тогда
N1=N(n-4;k-2)
и окончательная рекуррентная формула
N(n;k)=N(n-2;k-1)+N(n-3;k-1)+N(n-3;k-2)+N(n-4;k-2)

Нетрудно видеть, что N(3n;n)=1, а N(3n;n-1)=0. Для этого вершины одинарного дерева надо разбить на тройки соседних, и из каждой тройки выбрать среднюю. Будем отталкиваться от этого случая — числа вершин, кратных трём. При увеличении числа вершин этого одинарного дерева на 1, минимальное покрытие увеличится на 1 вершину. Посчитаем N(3n+1;n+1).
По выведенной рекуррентной формуле:
N(3n+1;n+1)=N(3n-1;n)+N(3n-2;n)+N(3n-2;n-1)+N(3n-3;n-1)
последнее слагаемое равно 1, так как N(3n-3;n-1)=N(3(n-1);n-1)=1
к остальным слагаемым применим снова эту рекуррентную формулу
N(3n-1;n)=N(3n-3;n-1)+N(3n-4;n-1)+N(3n-4;n-2)+N(3n-5;n-2)
N(3n-3; n-1)=1 M(3n-4)=M(3(n-1)-1)=n-1, поэтому N(3n-4; n-2)=0,
аналогично M(3n-5)=M(3(n-1)-2)=n-1 и N(3n-5; n-2)=0
Итак N(3n-1;n)=1+N(3n-4; n-1)
N(3n-2;n)=N(3n-4;n-1)+N(3n-5;n-1)+N(3n-5;n-2)+N(3n-6;n-2)=N(3n-4;n-1)+N(3n-5;n-1)+0+1
N(3n-2;n-1)=N(3n-4;n-2)+N(3n-5;n-2)+N(3n-5;n-3)+N(3n-6;n-3)=0

Итак:
N(3n+1;n+1)=1+N(3n-4; n-1)+N(3n-4;n-1)+N(3n-5;n-1)+1+1=3+2*N(3n-4; n-1)+N(3n-5;n-1)
Аналогично:
N(3n+2;n+1)=N(3n;n)+N(3n-1;n)+N(3n-1;n-1)+N(3n-2;n-1)
N(3n;n)=1
N(3n-1;n)=N(3n-3;n-1)+N(3n-4;n-1)+N(3n-4;n-2)+N(3n-5;n-2)=1+N(3n-4;n-1), так как
N(3n-3;n-1)=1, N(3n-4;n-2)=0, N(3n-5;n-2)=0
N(3n-1;n-1)=N(3n-3;n-2)+N(3n-4;n-2)+N(3n-4;n-3)+N(3n-5;n-3)=0
N(3n-2;n-1)=N(3n-4;n-2)+N(3n-5;n-2)+N(3n-5;n-3)+N(3n-6;n-3)=0

что в результате даёт:
N(3n+2;n+1)=2+N(3n-4;n-1)

Ответ
N(3n;n)=1
N(3n+1;n+1)=3+2*N(3n-4; n-1)+N(3n-5;n-1)
N(3n+2;n+1)=2+N(3n-4;n-1)

В сокращённой форме (без упоминания о количестве вершин в минимальном покрытии):
N'(3n)=1
N'(3n+1)=3+2*N'(3n-4)+N'(3n-5)
N'(3n+2)=2+N'(3n-4)

РЕГУЛЯРНЫЕ ВЫРАЖЕНИЯ

Задача «Бусы»
Уровень 1 (3 балла)
Бусы составляются из красных (к) и синих (с) бусинок по следующим правилам:
1) бусы могут состоять из одной красной бусины: <БУСЫ> => к
2) слева к уже готовым бусам можно добавить сначала красную, а потом синюю бусины: <БУСЫ> => кс<БУСЫ>
3) справа к уже готовым бусам можно добавить или красную или синюю бусину:
<БУСЫ> => <БУСЫ>к
<БУСЫ> => <БУСЫ>с
Опишите с помощью регулярных выражений все возможные виды бус, построенные по этим правилам. Чем меньше операций итерации будет использовано в ответе, тем лучше.

Решение и ответ: (кс)*к(к+с)*
Нетрудно видеть, что возможны все виды бус, кроме тех, которые начинаются с синего звена, поэтому короткий ответ: к(к+с)*
(Если не учитывать ориентацию бус, то есть разрешить их «переворачивать», то невозможными окажутся только бусы из одной синей бусины).

Уровень 2 (9 баллов)
Бусы составляются из красных (з) и синих (с) бусин по следующим правилам:
1) бусы могут состоять из двух бусин — красной и следующей за ней синей: <БУСЫ> => кс
2) слева к уже готовым бусам можно добавить красную бусину, одновременно добавляя справа синюю:
<БУСЫ> => к<БУСЫ>с
Докажите, что множество этих бус нельзя описать с помощью регулярного выражения.

Решение:
Поскольку при составлении бус при каждом добавлении синей бусины справа всегда добавляется красная бусина слева, число красных бусин в каждых так построенных бусах будет равно числу синих. Причём сначала идут подряд красные бусины, а затем такое же количество синих. Предположим, что мы сумели описать эти бусы регулярным выражением. Тогда в нем обязательно должна быть операция итерации, так как число всех возможных бус бесконечно: ….(w_1+w_2+...+w_k+...+w_n)*…
Если найдётся итерация такого общего вида, то в число всех бус войдут и бусы более простого вида (w_k)*, причём, можно считать что внутри выражения w_k уже не будет знаков сложения и итерации, иначе мы могли бы повторить предыдущее рассуждение, пока не дойдём до более простых частей бус с такими свойствами.

Тогда w_k=x_1...x_j...x_m, где каждое x_j либо красная (к) либо синяя (с) бусина.

Если в w_k только красные бусины, то (так как в описывающее структуру бус выражение входит итерация w_k) в множество всех бус входят бусы с любым числом w_k, и равенство числа красных и синих бусин будет нарушаться. Если же в w_k входят и красные и синие бусины (поровну), то повторение w_k порождает бусы, в которых правее синих появляются красные звенья (иными словами, красные и синие будут чередоваться), а такие бусы в описанное множество не входят.

ЛОГИКА (МИРЫ ТАРСКОГО)

Задача. Шашки

Уровень 1 (6 балов)

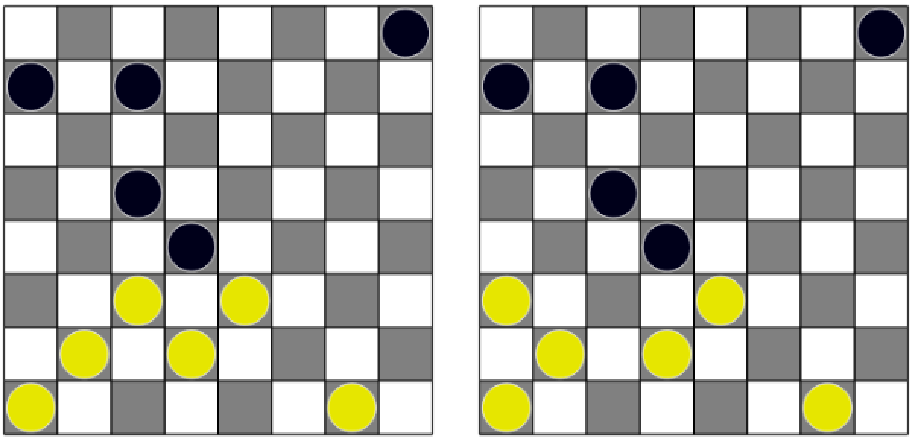
Двое игроков расставляют фигуры для игры в шашки (то есть фигуры ставятся только на черные поля доски). Один игрок расставляет как угодно любое число белых шашек в нижней части доски, второй игрок — любое число чёрных в верхней, но так, чтобы чёрные стояли на горизонталях, расположенных выше горизонталей, на которых стоят белые.

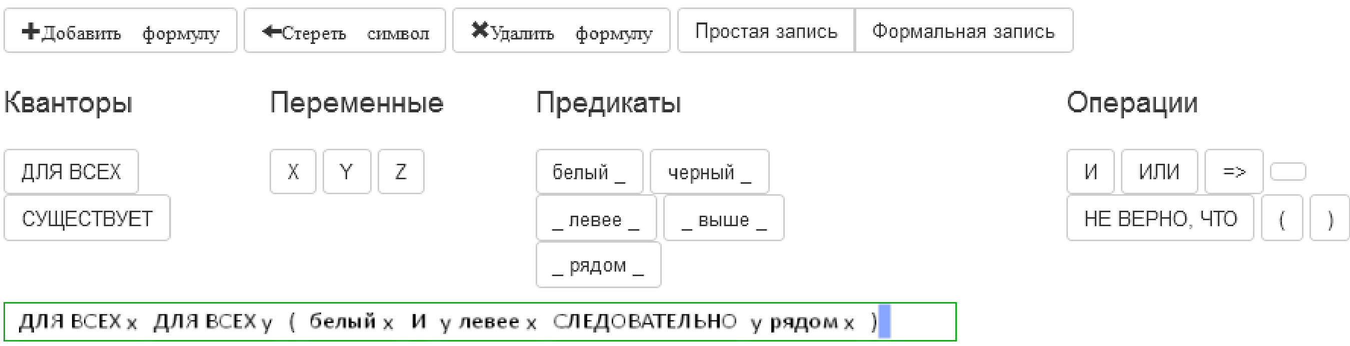
Первый ход делают белые. Ситуация перед началом игры считается благоприятной для белых, если в соседней клетке по диагонали рядом с хотя бы одной белой фигурой есть чёрная, которую она может «съесть», перескочив через неё по диагонали на соседнюю свободную клетку (по шашечным правилам). *Постройте логическую формулу благоприятной для белых ситуации перед началом игры.*

Формула должна быть представлена в префиксной форме, что означает, в начлае формулы друг за другом должны идти кванторы (например, «Для всех x Существует y, такой что Для всех z …» далее следует выражение без кванторов).

В качестве примера логической формулы высказывания приведём формальную запись описания такого расположения шашек: «Если шашечная фигура стоит левее белой (то есть, находится на вертикали, которая левее вертикали, на которой стоит исходная белая), то она стоит рядом с ней».

На рисунке показано, какими переменными, свойствами (предикатами), кванторами («для всех», «существует») и логическими операциями можно пользоваться при записи ответа.





Ответ.

1) ДЛЯ ВСЕХ x ДЛЯ ВСЕХ (белая(x) И чёрная(y) => y выше x)

2) СУЩЕСТВУЕТ x, ТАКОЙ ЧТО СУЩЕСТВУЕТ y, ТАКОЙ ЧТО ДЛЯ ВСЕХ z (белая(x) И чёрная(y) И y рядом с x И НЕ ВЕРНО, ЧТО (z рядом с y И z выше y И (y левее x И z левее y ИЛИ x левее y И y левее z)))

Уровень 2 (9 баллов)

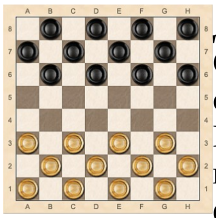
Предположим что в условиях предыдущей задачи имеется возможность использовать любое число переменных и любое число кванторов («Для любых», «Существует такой, что»).

Можно ли тогда для обычной шахматной доски 8х8 записать утверждение о том, что все фигуры установлены на начальную для игры в шашки позицию: белые занимают сплошь три нижних ряда, а чёрные верхние три (имеются в виду только чёрные клетки рядов), используя только заданные свойства, отношения, логические операции и любое число кванторов и переменных?

Решение.

Возможно. Проблема в том, как учесть границы доски и/или размер доски?

Для этого можно использовать следующее соображение: если фигура является, например, граничной справа, то слева от неё будет находиться цепочка из 7 (для обычной доски) фигур, каждая из которых стоит левее и рядом с предыдущей.



Для цепочки из трёх фигур (доски 3х3) это можно записать как:

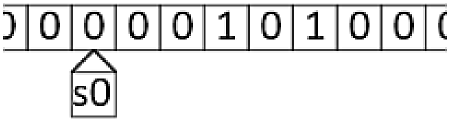
Существует x, такой что Существует y, такой что Существует z, такой что (y рядом с x И z рядом с y И y левее x И z левее y).

Если записать аналогичное утверждения с 8 переменными и кванторами для каждой границы и учитывая цвет фигур, то получим описание существования фигур, ближайших к углам доски. Далее это утверждение можно соединять с утверждениями о том, что рядом с такими фигурами стоит по две фигуры такого же цвета. Нужно будет разделить утверждения для разных краёв доски и фигур разного цвета, поскольку для дальнейшего описания придётся использовать условия на «соседей соседей», стоящих по одну сторону.

Разумеется утверждение будет очень длинным. Его можно было бы укоротить если вводить обозначения для уже построенных условия и отношений. Например, Угловая(x) – для обозначения фигуры, стоящей в углу доски.

АЛГОРИТМ. МАШИНА ТЬЮРИНГА

Уровень 1 (3 балла)
На ленте стоят две единицы, между которыми стоит ноль. Остальная часть ленты заполнена нулями. Головка машины указывает на ноль, стоящий на две позиции левее левой единицы (см. рисунок). Нужно сконструировать машину Тьюринга, которая, кроме начального (s) и конечного (f) состояний, имеет ещё одно (q) и не использует иных символов алфавита, кроме {1;0}.



Эта машина должна поставить на ленту как можно больше символов 1 (не обязательно подряд) и остановиться. Попробуйте добиться того, чтобы *на ленте после остановки машины было не меньше 5 единиц*.

Решение.

Бросающееся в глаза решение — заменить единицами нули, стоящие правее головки до крайней справа единицы: s0->s1R; s1->q1R; q0->q1R; q1->f1R.

Уровень 2 (9 баллов)

В условиях первой задачи цикла попробуйте добиться того, чтобы *на ленте после остановки машины было не меньше 8 единиц*. Можно ли построить машину с такими свойствами?

Решение

!!! в условии была допущена ошибка в рисунке — каретка должна стоять на одну позицию правее, тогда решение приведено ниже. Для данной картинки 8 единиц поставить не удастся, но доказательство этого утверждения требует большого перебора вариантов и не предполагалось авторами задачи.

Такое решение есть, для его построения нужно учесть возможность использования вновь записанной на ленту информации:
s0->q1L; q0->s1R; s1->q1R; q1->f1N.

Уровень 3 (11 баллов)

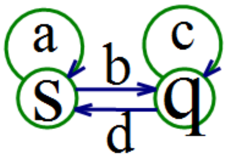
Рассмотрим все машины Тьюринга с двумя состояниями - начальным (s) и «промежуточным» (q) (конечного состояния в машине нет), головка которых может двигаться только вправо (R). Алфавит машин состоит только из двух символов {1;0}. Изначально на ленте записано некоторое число символов 1 (не обязательно идущих подряд), остальная часть ленты заполнена символами 0, а считывающая головка указывает на крайнюю левую единицу. Машина останавливается, если встречается комбинация состояния и символа на ленте, для которой действие машин не определено.

Найдите такую конфигурацию расстановки единиц, для которой ни одна из машин с указанными в задаче ограничениями не может удалить все единицы заданной конфигурации, после чего остановиться. Постарайтесь найти минимальную конфигурацию с таким свойством. Ответ обоснуйте.

*Дополнительный вопрос (не обязательный). Попробуйте с помощью регулярных выражений описать все возможные конфигурации, для которых среди указанных машин найдётся такая, которая удалит все единицы и остановится.

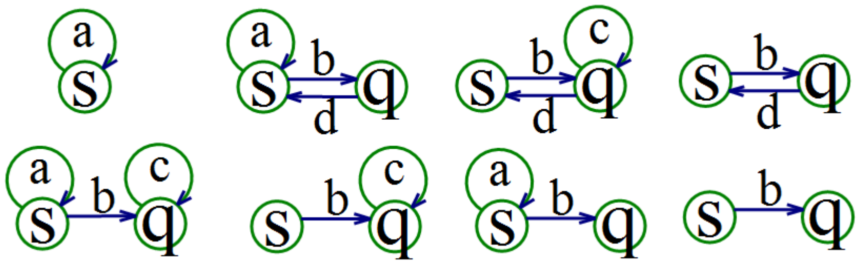
Решение

Поскольку головка машины движется только вправо, она никогда не вернётся к повторной обработке уже обработанного символа. Поэтому работу такой машины можно представить так называемым конечным автоматом, на вход которого подаётся поток входных символов (можно представить, что они записаны на ленте, а лента вводится в автомат, как вводилась с перфоленты информация в первые ЭВМ). Такой автомат можно изобразить графом, у которого вершины состояния, а рёбра — переходы по указанным на них входным символам. Поскольку наша машина должна стереть все единицы, не умаляя общности можно считать, что при вводе любого символа в любом состоянии машины на ленту будет записываться ноль. На рисунке показан граф автомата с двумя состояниями p и q и всеми вариантами переходами между ними по символам, обозначенным как a, b, c и d (a и b здесь различны, также и c и d различны, в остальном они могут быть любые).



Так как машина не имеет конечного состояния она останавливается когда встречается комбинация состояния и входного символа, на которую не предусмотрено реакции. То есть в указанном выше автомате нужно удалить один или более переходов, уситывая что работа начинается в состоянии s, поэтому, если из него нет переходов в q, то переходы из q выполняться уже не будут.

Таким образом можно перечислить все возможные автоматы:



Для каждого из них можно описать регулярные множества, на которых они определены (и, следовательно, можно сделать автомат превращающий все такие конфигурации в нулевые):
Первый (слева в верхнем ряду) определен на наборах из символов aaa...ab. Так как головка стоит на первой единице, то это наборы вида 111...10. Это можно записать как a*b или 1*0.
Второй (в верхнем ряду слева) определен на наборах (111...10d)*111...10c.
Третий — на наборах (bc*d)*a - это краткая запись множества набором, задаваемых таким регулярным выражением (00*1)*1+(01*0)*1+(10*1)*0+(11*0)*0.
Четвёртый — на наборах (bd)*a и (bd)*bc = (00)*1+(01)*1+(10)*0+(11)*0
Пятый (первый в нижнем ряду слева) — a*bc*d = 0*10*1+0*11*0+1*00*1+1*01*0
Шестой — bc*d или a, иначе 00*1+01*0+10*1+11*0+0+1
Седьмой — a*bc (то же что a*bd) = 0*10+0*11+1*00+1*11
Восьмой — bc (bd) или a, то есть 00+01+10+11+0+1

Теперь нетрудно найти наборы, которые не описываются ни одним из этих выражений, например, 11001. Можно было не выписывать всех регулярных выражений, а проверить действия всех восьми типов автоматов на этом наборе. Например, для второго автомата рассмотрим два случая a=1 и a=0. В первом случае должно быть b=0, d=0, тогда работа автомата на наборе 110010...0 будет характеризоваться такой последовательностью переходов: sssqssqsqsq… то есть машина не остановится. Аналогично рассматриваются и другие варианты.

Ответ. 11001