Задача А. Разбор утверждения

Имя входного файла: стандартный ввод Имя выходного файла: стандартный вывод

Ограничение по времени: 2 секунды Ограничение по памяти: 256 мегабайт

На вход вашей программе дается утверждение в следующей грамматике:

```
      ⟨Файл⟩
      ::=
      ⟨Выражение⟩

      ⟨Выражение⟩
      ::=
      ⟨Дизъюнкция⟩ | ⟨Дизъюнкция⟩ '->' ⟨Выражение⟩

      ⟨Дизъюнкция⟩
      ::=
      ⟨Конъюнкция⟩ | ⟨Дизъюнкция⟩ '&' ⟨Отрицание⟩

      ⟨Конъюнкция⟩
      ::=
      ⟨Отрицание⟩ | ⟨Конъюнкция⟩ '&' ⟨Отрицание⟩

      ⟨Отрицание⟩
      ::=
      '!' ⟨Отрицание⟩ | ⟨Переменная⟩ | '(' ⟨Выражение⟩ ')'

      ⟨Переменная⟩
      ::=
      ('A'...'Z') {'A'...'Z' | '0'...'9' | '''}*
```

Имена переменных не содержат пробелов. Между символами оператора '->' нет пробелов. В остальных местах пробелы могут присутствовать. Символы табуляции и возврата каретки должны трактоваться как пробелы. Символ апострофа ('') имеет код 39.

Вам требуется написать программу, разбирающую утверждение и строящую его дерево разбора, и выводящую полученное дерево в единственной строке без пробелов в следующей грамматике:

Формат входных данных

В единственной строке входного файла дано утверждение в грамматике из условия. Размер входного файла не превышает 100 КБ.

Формат выходных данных

В единственной строке выходного файла выведите дерево разбора утверждения без пробелов.

стандартный ввод				
!A&!B->!(A B)				
стандартный вывод				
(->,(&,(!A),(!B)),(!(,A,B)))				
стандартный ввод				
P1'->!QQ->!R10&S !T&U&V				
стандартный вывод				
(->,P1',(->,(!QQ),(,(&,(!R10),S),(&,(&,(!T),U),V))))				

Задача В. Формализация метаязыка

Имя входного файла: стандартный ввод Имя выходного файла: стандартный вывод

Ограничение по времени: 15 секунд Ограничение по памяти: 256 мегабайт

Для записи доказательств классического исчисления высказываний часто используется метаязык, позволяющий сильно сократить текст доказательства в ущерб его формальности. Давайте попробуем формализовать этот метаязык, хотя бы частично.

Доказательство на формализуемом метаязыке состоит из строк вида $\Gamma \vdash \alpha$, при этом каждая строчка либо (а) аксиома (если часть строки за турникетом – аксиома), либо (б) гипотеза (формула за турникетом взята из контекста), либо (в) результат применения Modus Ponens к предыдущим строкам $\Gamma \vdash \alpha$ и $\Gamma \vdash \alpha \to \beta$ (гипотезы во всех трёх строках должны совпадать с точностью до перестановки), либо (г) результат применения теоремы о дедукции.

Поясним пункт (г) подробнее: будем говорить, что формула является результатом применения теоремы о дедукции, если она может быть получена из некоторой строки, указанной выше в доказательстве, путём применения теоремы о дедукции, возможно, многократного. Не переносимые через турникет теоремой о дедукции гипотезы должны совпадать с точностью до перестановки. Например, вторая строка есть результат применения теоремы о дедукции к первой строке:

$$\alpha, \beta, \gamma, \zeta \vdash \delta \to \zeta$$
$$\zeta, \delta, \alpha \vdash \gamma \to \beta \to \zeta$$

Другие мета-операции, помимо перечисленных, например, размножение или удаление гипотез, в данной формализации для простоты не допускаются.

Формат входных данных

На вход подаётся доказательство в следующем формате:

```
      ⟨Файл⟩
      ::=
      { ⟨Строка⟩ '\n' } +

      ⟨Строка⟩
      ::=
      ⟨Контекст⟩ '| -' ⟨Выражение⟩

      ⟨Контекст⟩
      ::=
      ⟨Выражение⟩ [', ' ⟨Выражение⟩] *

      |
      ''
      ⟨Выражение⟩ '| ' ⟨Выражение⟩

      |
      'Выражение⟩ '->' ⟨Выражение⟩

      |
      '! ' ⟨Выражение⟩ ')'

      |
      '(' ⟨Выражение⟩ ')'

      |
      'Переменная⟩

      ⟨Переменная⟩
      ::=
      ('A'...'Z') {'A'...'Z' | '0'...'9' | '''}*
```

Операторы '&' и '|' левоассоциативны. Оператор '->' правоассоциативен. Операторы в порядке уменьшения приоритета: '!', '&', '|', '->'.

Имена переменных не содержат пробелов. Между символами одного оператора нет пробелов ('->' и '|-'). В остальных местах пробелы могут присутствовать. Символы табуляции $(9, '\t')$ и возврата каретки $(13, '\t')$ должны трактоваться как пробелы.

Гарантируется, что входной файл синтаксически корректен и содержит как минимум одну строку с формулой, однако, в нём могут найтись недоказанные строки, то есть строки, не получаемые ни одним из указанных выше способов. Объём входного файла – не более 3 Мб.

Формат выходных данных

Выведите проаннотированное входное доказательство: к каждой строке входного файла дополнительно в квадратных скобках укажите её порядковый номер и правило, использованное для вывода. Посимвольного равенства формул в выходном файле соответствующим формулам во входном

Матлог 2023 (у2021) СПб, ИТМО, Весна 2023 года

файле не требуется, достаточно любого эквивалентного написания. Порядок гипотез в строке также должен совпадать с порядком гипотез в соответствующей строке исходного файла. Если строка не является доказанной, укажите это. Ошибки не распространяются транзитивно и некорректная строка может быть использована для обоснования других строк. Если корректных обоснований может быть несколько, укажите любое.

Схема аксиом	Номер
$\alpha \to \beta \to \alpha$	1
$(\alpha \to \beta) \to (\alpha \to \beta \to \gamma) \to (\alpha \to \gamma)$	2
$\alpha \to \beta \to \alpha \& \beta$	3
$\alpha \& \beta \to \alpha$	4
$\alpha \& \beta \to \beta$	5
$\alpha \to \alpha \vee \beta$	6
$\beta \to \alpha \vee \beta$	7
$(\alpha \to \gamma) \to (\beta \to \gamma) \to (\alpha \lor \beta \to \gamma)$	8
$(\alpha \to \beta) \to (\alpha \to \neg \beta) \to \neg \alpha$	9
$\neg\neg\alpha\to\alpha$	10

стандартный ввод	стандартный вывод	
-A&B->A	[1] -(A&B)->A [Ax. sch. 4]	
-A&B->B	[2] -(A&B)->B [Ax. sch. 5]	
A&B - A	[3] A&B -A [Ded. 1]	
A&B -B	[4] A&B -B [Ded. 2]	
A&B -B->A->B&A	[5] A&B -B->A->(B&A) [Ax. sch. 3]	
A&B -A->B&A	[6] A&B -A->(B&A) [M.P. 4, 5]	
A&B -B&A	[7] A&B -B&A [M.P. 3, 6]	
-A&B->B&A	[8] -(A&B)->(B&A) [Ded. 7]	
A->B,C,D -E->D	[1] A->B,C,D -E->D [Incorrect]	
A->B,C,D,D -D	[2] A->B,C,D,D -D [Hyp. 4]	
A->B,C,D,D -D->E	[3] A->B,C,D,D -D->E [Incorrect]	
A->B,C,D -E	[4] A->B,C,D -E [Incorrect]	
D,A->B,C,D -E	[5] D,A->B,C,D -E [M.P. 2, 3]	
E,D,C -(A->B)->D	[6] E,D,C -(A->B)->D [Ded. 1]	

Задача С. Нормализация метаязыка

Имя входного файла: стандартный ввод Имя выходного файла: стандартный вывод

Ограничение по времени: 15 секунд Ограничение по памяти: 256 мегабайт

Дано доказательство на метаязыке из предыдущей задачи:

$$\Gamma_1 \vdash \alpha_1$$

$$\Gamma_2 \vdash \alpha_2$$

$$\dots$$

$$\Gamma_n \vdash \alpha_n$$

Перестройте доказательство в доказательство формулы α_n из контекста Γ_n на предметном языке классического исчисления высказываний.

Формат входных данных

Соответствует входному формату из предыдущей задачи. Гарантируется, что доказательство корректно. Объём входного файла – до 3 Мб, при этом можно так выбрать подстроки из входного доказательства (доказательство может содержать формулы, не обязательные для вывода Γ_n), что выходной файл будет не больше 10 Мб.

Формат выходных данных

Доказательство с использованием следующей грамматики:

```
\langle \Phiайл\rangle ::= { \langle Kонтекст\rangle 'I-' \langle Bыражение\rangle '\n' } { \langle Bыражение\rangle '\n'}* \langle Kонтекст\rangle ::= \langle Bыражение\rangle [',' \langle Bыражение\rangle]*
```

Первая строка выходного файла должна соответствовать $\Gamma_n \vdash \alpha_n$ (тот же порядок гипотез, то же дерево разбора для всех формул), текст далее должен являться выводом α_n из Γ_n в классическом исчислении высказываний.

стандартный ввод	стандартный вывод	
A -A	-A->A	
-A->A	$A \rightarrow (A \rightarrow A) \rightarrow A$	
	A->A->A	
	(A->A->A)->(A->(A->A)->A)->A	
	$(A \rightarrow (A \rightarrow A) \rightarrow A) \rightarrow A \rightarrow A$	
	A->A	
A,B,C -A	A -C->B->A	
A -C->B->A	A->B->A	
	(A->B->A)->C->A->B->A	
	C->A->B->A	
	A	
	A->C->A	
	C->A	
	(C->A)->(C->A->B->A)->C->B->A	
	(C->A->B->A)->C->B->A	
	C->B->A	
-A&B->A	A&B - A B	
A&B -A	A&B	
A&B -A->A B	(A&B)->A	
A&B -A B	A	
	A->(A B)	
	A B	

Задача D. Полнота исчисления высказываний

Имя входного файла: стандартный ввод Имя выходного файла: стандартный вывод

Ограничение по времени: 1 секунда Ограничение по памяти: 256 мегабайт

В данной задаче требуется построить доказательство выражения классического исчисления высказываний (в варианте натурального вывода), если оно существует.

Формат входных данных

На вход дается единственная строка — высказывание, записанное в соответствии со следующей грамматикой:

```
⟨Выражение⟩ ::= ⟨Выражение⟩ '&' ⟨Выражение⟩ | ⟨Выражение⟩ '| '⟨Выражение⟩ | ⟨Выражение⟩ '| '⟨Выражение⟩ | '(' ⟨Выражение⟩ ')' | '_|_' | ⟨Переменная⟩ | ⟨Переменная⟩ | ('A'...'Z') ('A'...'Z' | '0'...'9' | ''')*
```

Операторы '&' и '|' левоассоциативны. Оператор '->' правоассоциативен. Операторы в порядке уменьшения приоритета: '&', '|', '->'.

<u>Будьте внимательны!</u> В случае копирования грамматики из PDF-файла помните, что ASCII код апострофа — 39. Код символов в PDF-файле часто отличается от желаемой кодировки.

Имена переменных не содержат пробелов. Между символами одной связки/константы нет пробелов ('->' и '_|_'). В остальных местах пробелы могут присутствовать. Символы табуляции должны трактоваться как пробелы.

Общая длина выражения не превышает 255 символов. Всего в выражении не будет использовано больше семи различных переменных.

Формат выходных данных

Если высказывание опровержимо (не является общезначимым), укажите этот факт в единственной строке выдачи, приведя опровергающую оценку:

```
\langle \text{Опровержима} \rangle ::= \text{`Formula is refutable ['} \langle \text{Оценка} \rangle \text{']'} \langle \text{Оценка} \rangle ::= \langle \text{Переменная} \rangle \text{':='} \langle \text{Значение} \rangle \text{ [','} \langle \text{Оценка} \rangle \text{]} \langle \text{Значение} \rangle ::= \text{`T'} \mid \text{`F'}
```

Иначе выведите доказательство. Каждая строка доказательства — узел дерева, пустых строк быть не должно (кроме последней строки). Дочерние узлы указываются перед родительским узлом. В начале строки — уровень узла в квадратных скобках, потом через пробел — формула, в конце строки — обозначение правила, также через пробел и в квадратных скобках. Для обозначения лжи используйте комбинацию «_|_»: подчёркивание (ASCII 95), вертикальная черта (ASCII 124), подчёркивание (ASCII 95). В остальном следуйте формату из примеров.

Доказанное во входном файле высказывание должно быть заключением самого верхнего правила. В данном высказывании отрицание термов ($\neg \varphi$) передавайте как ($\varphi \to \bot$). В доказательстве вы можете пользоваться следующими правилами. Посылки правил должны идти в указанном порядке, переставлять их нельзя — однако, гипотезы в контексте могут быть произвольно переставлены.

Матлог 2023 (у2021) СПб, ИТМО, Весна 2023 года

Обозначение	Посылки	Заключение
Ax		$\Gamma, \varphi \vdash \varphi$
E->	$\Gamma \vdash \varphi \to \psi, \Gamma \vdash \varphi$	$\Gamma \vdash \psi$
I->	$\Gamma, \varphi \vdash \psi$	$\Gamma \vdash \varphi \to \psi$
I&	$\Gamma \vdash \varphi, \ \Gamma \vdash \psi$	$\Gamma \vdash \varphi \& \psi$
El&	$\Gamma \vdash \varphi \& \psi$	$\Gamma \vdash \varphi$
Er&	$\Gamma \vdash \varphi \ \& \ \psi$	$\Gamma \vdash \psi$
Ill	$\Gamma \vdash \varphi$	$\Gamma \vdash \varphi \lor \psi$
Irl	$\Gamma \vdash \psi$	$\Gamma \vdash \varphi \lor \psi$
Εl	$\Gamma, \varphi \vdash \rho, \ \Gamma, \psi \vdash \rho, \ \Gamma \vdash \varphi \lor \psi$	$\Gamma \vdash \rho$
E!!	$\Gamma, \varphi \to \bot \vdash \bot$	$\Gamma \vdash \varphi$

стандартный ввод	стандартный вывод
A->(A&(A->A))	[2] A -A [Ax]
	[3] A,A -A [Ax]
	[2] A -A->A [I->]
	[1] A -A&(A->A) [I&]
	[O] -A->(A&(A->A)) [I->]
_ _ A	[2] _ _,A->_ _ _ [Ax] [1] _ _ -A [E!!]
	[0] >A [I->]
A->(A->_ _)	Formula is refutable [A:=T]

Задача Е. Поверхностные кванторы

Имя входного файла: стандартный ввод Имя выходного файла: стандартный вывод

Ограничение по времени: 1 секунда Ограничение по памяти: 256 мегабайт

По формуле φ , не использующей отрицаний, конъюнкций и дизъюнкций, постройте эквивалентную формулу с поверхностными кванторами ψ и доказательства $\varphi \vdash \psi$ и $\psi \vdash \varphi$.

Формула задаётся в синтаксисе формальной арифметики для упрощения языка (в этом случае не требуется поддерживать произвольные предикатные и функциональные символы). Доказательство не должно использовать никаких нелогических аксиом.

Формат входных данных

```
(выражение)
                      ⟨унарное⟩ | ⟨унарное⟩'->'⟨выражение⟩
    (унарное)
                      ⟨предикат⟩ | '('⟨выражение⟩')'
                      (переменная)
                 ::=
                      'a'...'z'{'a'...'z' | '0'...'9'}*
                      \langle \text{терм} \rangle = \langle \text{терм} \rangle
   (предикат)
                 ::=
        (терм)
                      ⟨слагаемое⟩ | ⟨терм⟩'+'⟨слагаемое⟩
                 ::=
  (слагаемое)
                 ::=
                      ⟨умножаемое⟩ | ⟨слагаемое⟩'*'⟨умножаемое⟩
                      \langleпеременная\rangle | '('\langleтерм\rangle')'
(умножаемое)
                      '0' | (умножаемое)'',
```

Формат выходных данных

В первой строке — заголовок доказательства ($\varphi \vdash \psi$). В последующих строках — доказательство утверждения в формальной арифметике, не использующее нелогических аксиом, по формуле на строку. То есть, последовательность формул формальной арифметики, каждая из которых — либо аксиома исчисления предикатов, либо получается из предыдущих по правилу Modus Ponens или по правилам с кванторами. Далее в файле должен идти заголовок обратного доказательства ($\psi \vdash \varphi$) и само доказательство.

Кроме отсутствия кванторов в глубине формулы, никаких других требований на формулу ψ не накладывается. В частности, вы можете исключить какие-то кванторы, переименовать связанные переменные: для формулы $\varphi := x = 0 \to \forall x.x = 0$ рассмотреть $\psi := \forall y.x = 0 \to y = 0$, и т.п.

При необходимости в доказательстве вы можете использовать остальные связки (!, & и |) и все соответствующие аксиомы из исчисления высказываний.

```
стандартный ввод
0=0->@x.x=0
                                стандартный вывод
(0=0->0x.x=0) | -(0x.0=0->x=0)
(0=0->0x.x=0)->(0=0->((0x.x=0)->x=0))->(0=0->x=0)
0=0->0x.x=0
(0=0->((0x.x=0)->x=0))->(0=0->x=0)
(0x.x=0)-x=0
((0x.x=0)->x=0)->0=0->((0x.x=0)->x=0)
0=0->((0x.x=0)->x=0)
0=x<-0=0
(0=0->x=0)->(0=0->0=0->0=0)->(0=0->x=0)
(0=0->0=0->0=0)->(0=0->x=0)
(0=0->0=0->0=0)->(0x.0=0->x=0)
(0=0->0=0->0=0)
(0x.0=0->x=0)
(0x.0=0->x=0) | -(0=0->0x.x=0)
0x.0=0->x=0
(0x.0=0->x=0)->0=0->x=0
0=0->x=0
0=0->0x.x=0
```