



В задачах сегодняшнего дня (кроме `regrsm2`) вы должны написать программу, которая будет выводить на экран одну строку (в задаче `regnum` — две строки) — искомое регулярное выражение. Не забывайте символы `$` и `^` в тех выражениях, которые должны ловить целую строку.

Например, для задания регулярного выражения, распознающего строки, начинающиеся с символа обратный слэш “\” можно сдать следующую программу:

```
print("^\\\\")
```

### Regstring. Провераем строку

Имя входного файла:

Имя выходного файла:

Выведите регулярное выражение, проверяющее, что данное строка является корректной заключенной в кавычки строковой константой языка C. Считайте, что в строке не встречается символов перевода строки.

Например, строки `"test test"` и `"test \" test"` являются корректными строковыми константами, а `"test\"test"` — нет.

### Regif. Присванивание в if'ax

Имя входного файла:

Имя выходного файла:

Программисты, переходящие с паскаля на C-подобные языки, часто делают ошибки, записывая в операторе `if` одиночное равенство, а не двойное: `'if (a=0)'` вместо `'if (a==0)'`. Напишите регулярное выражение, позволяющее отлавливать такие ошибки. А именно, оно должно срабатывать (т.е. находить совпадение) на строках кода, содержащих присваивание внутри условия `if'a`, и не срабатывать на коде, такой ошибки не содержащей.

Поскольку правильные скобочные последовательности не являются регулярными языком и довольно сложно задаются даже `perl-compatible` регулярными выражениями, от вас не требуется корректно обрабатывать записи вида

```
if (a==0) b=c;
```

(т.е. когда одиночное равенство не находится в `if'e` потому, что все скобки сбалансировались и условие `if'a` кончилось). Вместо этого считайте, что после условия `if'a` всегда идет либо открывающая фигурная скобка, либо конец строки.

Считайте, что ваше выражение должно будет обрабатывать одиночные строки кода, т.е. его использование будет проходить следующим образом: разбили код на строки и в каждой строке отдельно поискали совпадения с этим регулярным выражением. Естественно, считайте, что условие `if'a` всегда находится в одной строке.

Для упрощения задачи ваше регулярное выражение может считать, что в строках не встречается строковых констант и комментариев.

### Regprime. Провераем на простоту

Имя входного файла:

Имя выходного файла:

В этой задаче будем натуральное число `N` записывать `N` единицами подряд, например, число 4 будем записывать как `'1111'`

Выведите регулярное выражение, проверяющее, что данное число, записанное указанным образом, является составным. Выражение должно работать для произвольных натуральных чисел, больших единицы.

Например, регулярное выражение `'^(11){2,}$'` правильно работает для записей чисел от 2 до 8, но дает неправильный результат для числа 9.

### Regnum. Конвертация чисел

Имя входного файла:

Имя выходного файла:

Пусть у вас есть текст, в котором встречаются вещественные числа, записанные с десятичной точкой и, возможно, с экспоненциальной частью (например, `'1.25'`, `'1.25e10'`, `'1.25e+10'` или `'1.25E-10'`). Ваша задача — с помощью регулярных выражений нормализовать их, а именно:

- заменить в записи всех вещественных чисел десятичную точку на десятичную запятую,
- удалить все ведущие нули как у мантииссы, так и у экспоненциальной части (обратите внимание, что перед десятичной запятой все равно должен остаться как минимум один ноль).

А именно, вам надо написать регулярное выражение и строку замены, которые будут решать эту задачу. (В строке замены можно использовать ссылки на скобки в начальной строке в виде `'\2'`.)

Вы должны сдать программу, которая выводит две строки: регулярное выражение и строку замены.

Задача строго не определена: вы должны корректно обрабатывать разумные записи, но неразумные записи (например, `'1.1e1e1.1e1'`) можете обрабатывать как угодно, их не будет в тестах. Также считайте, что целые числа, а также числа без десятичной точки нормализовать не надо.

### Пример

<code>0.1+012.3e-09</code>	<code>0,1+12,3e-9</code>



## RegPCMS2. Разбор таблицы результатов

Имя входного файла: regpcms2.in

Имя выходного файла: regpcms2.out

Во входном файле вам дана html-страничка — результаты некоторой олимпиады по программированию, сформированные системой PCMS2 (см. пример на страничке параллели.) Выведите в выходной файл информацию по всем *успешным* посылкам в этом контексте. Каждую посылку выводите на отдельной строке в следующем формате:

<название команды> <идентификатор задачи> <время в минутах> <количество штрафных посылок по этой задаче>

Задачи нумеруются заглавными латинскими буквами, начиная с А. Посылки сортируйте по времени сдачи, при равных временах — по названию команды. Названия команд выводите в той же кодировке, в которой они заданы во входном файле (т.е. просто выводите те же байты).

P.S. Напишите программы с использованием регулярных выражений для поиска нужных фрагментов текста.

### Пример

Выставлен на страничке параллели.