Нижегородский государственный университет
им. Н.И.Лобачевского

Институт Информационных Технологий, Математики и Механики

Регулярные выражения в C++ (Regex)

Содержание

- Регулярные выражения
- Синтаксис
- Примеры
- Задача

Регулярные выражения

- **Регуля́рные выраже́ния*** (англ. *regular expressions*) — формальный язык поиска и осуществления манипуляций с подстроками в тексте, основанный на использовании метасимволов (символов-джокеров, англ. *wildcard characters*). Для поиска используется строка-образец (англ. *pattern*, по-русски её часто называют «шаблоном», «маской»), состоящая из символов и метасимволов и задающая правило поиска. Для манипуляций с текстом дополнительно задаётся строка замены, которая также может содержать в себе специальные символы.

*определение из Википедии: https://ru.wikipedia.org/wiki.Регулярные_выражения

Регулярные выражения

- Функции регулярных выражений:
  - проверка наличия искомого образца в заданном тексте;
  - определение подстроки текста, которая сопоставляется образцу;
  - определение групп символов, соответствующих отдельным частям образца.
- Пример использования регулярных выражений:
  - найти все последовательности символов *«кот»* в любом контексте, как то: *«кот»*, *«котлета»*, *«терракотовый»*;
  - найти отдельно стоящее слово *«самолет»* и заменить его на *«аэроплан»*;
  - найти слово *«посуда»*, которому предшествует слово *«стеклянная»* или *«пластиковая»*;
  - убрать из текста все предложения, в которых упоминается слово *«комментарий»* или *«тест»*.

Регулярные выражения в C++

- Стандартная библиотека в C++11. В Visual Studio 2008 и 2010 пространство имен std::tr1, в более поздних std.
- Поддерживаемые синтаксисы:
  - basic
  - extended
  - ECMAScript (по умолчанию в VS)
  - awk
  - grep
  - egrep
  - и т.д.

Регулярные выражения в C++

- Основные методы класса regex:
  - ***regex (const CharT* s, flag_type f = std::regex_constants::ECMAScript);*** - конструктор, CharT* s – шаблон, flag_type f – тип синтаксиса
- Функции с регулярными выражениями:
  - ***bool regex_match(string str, regex rx);*** – функция сравнивает строку str с шаблоном rx. Если строка соответствует шаблону, то возвращает true, иначе false
  - ***bool regex_search(string str, smatch match, regex rx);*** – функция ищет в строке str соответствующие шаблону rx вхождения и записывает его в match. Возвращает true при нахождение соответствия
  - ***string regex_replace(string str, regex rx, string replace, …);*** – функция заменяет все вхождения шаблона rx в строку str на строку replace и возвращает полученную строку.
  - Возможны иные прототипы у каждой из функций

Синтаксис ECMAScript

- В грамматике ECMAScript специальное значение имеют следующие символы:
  - `^ $ \ . * + ? ( ) [ ] { } |`
- Остальные символы считаются обычными.

Синтаксис ECMAScript

- «\» - экранирующий символ, после него символ интерпретируется по другому. Специальный становиться обычным, а обычный интерпретируется как специальный.
- «^» - соответствие началу строки. Например: «^A» соответствует «A» в «An E», но не в «an A».
- «$» - соответствует концу строки. Пример шаблона: «A$».
- Повторение предыдущего символа:
  - «*» - 0 и более раз;
  - «+» - 1 и более раз;
  - «?» - 0 или 1 раз;
  - «{n}» - ровно n вхождений;
  - «{n,m}» - вхождений от n до m раз;
  - Пример «ba+» соответствует «ba», «baa», «baaa» и тд, но не «b».
- «.» - один любой символ, кроме переноса строки.
- «[…]» - соответствие из набора символов …. Пример: [0-9] или [0123456789]– соответствие цифре. [^0-9] – не цифра.
- «|» - или.
- «(…)» - захватывающие скобки

Синтаксис ECMAScript

- \b – соответствует границе слова.
- \B – соответствует не словообразующей границе. Не словообразующая граница соответствует позиции, в которой предыдущий и следующий символы являются символами одного типа.
- \d – соответствует цифровому символу.
- \D – соответствует нецифровому символу.
- \n, \r – соответствуют переводу строки и возврату каретки.
- \s – одиночный символ пустого пространства. Пробел, tab и т.п.
- \S – одиночный символ непустого пространства.
- \t – горизонтальный tab.
- \v – вертикальный tab.
- \w – любой цифробуквенный символ, включая нижние подчеркивание. Эквивалент [A-Za-z0-9_].
- \W – любой не цифробуквенный символ.
- И т.д.

Примеры

- (2[0-3]|[0-1]\d):[0-5]\d – время в формате «часы:минуты»
- ^((8|\+7)[\-]?)?(\(?\d{3}\)?[\-]?)?[\d\-]{7,10}$ - номер телефона в любом доступном формате:
  - (8|\+7) – номер начинается с 8 или +7
  - [\-]? – разделение в виде «-» или « », которое может отсутствовать.
  - ((8|\+7)[\-]?)? – начальный префикс может отсутствовать. (495)123456
  - \(?\d{3}\)? – наличие трех цифр, в скобках и без.
  - [\d\-]{7,10} – наличие от 7 до 10 цифр и разделителей

Есть ли ошибка в данном шаблоне номера телефона?

Примеры

- Необходима замена [\d\-]{7,10} на ([\d][\-]?){7} для исключения номеров вида:

  8(123)4567890123

- Улучшенный шаблон по поиску номера телефона: ^((8|\+7)[\-]?)?(\(?\d{3}\)?[\-]?)?([\d][\-]?){7}$

Задача

- В файле WarAndPeace.txt, содержится текст 4-х томов «Войны и Мир» на английском языке. Задача:
  - Необходимо найти количество вхождения слова «War» в текст произведения без учетом регистров.
  - Необходимо заменить все такие вхождения на слово «Peace».
  - В получившемся тексте необходимо найти количество вхождений буквосочетания «peace», без учета регистров.