**CMSC330 Ruby RegExp Cheat Sheet**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| String search operations | |  | Character classes | | | |
| s.index(target,pos) | Find target, start at pos |  | . (period) | | Any character | |
| s.sub(old,new) | Substitute new for 1st old |  | \d | | Digit [0-9] | |
| s.gsub(old,new) | Substitute new for all old |  | \s | | Whitespace [\t\r\n\f\s] | |
| s.split(target) | Split string on target |  | \w | | Alphanumeric [A-Za-z0-9] | |
| s =~ /pattern/ | Contains: First position *or* nil |  | \D | | Non-digit [^0-9] | |
| s !~ /pattern/ | Not contains: true if *not* found |  | \S | | Non-whitespace [^\t\r\n\f\s] | |
|  |  |  | \W | | Non-word [^A-Za-z0-9] | |
| Regular expressions | |  | /[abcd]/ | | Character class | |
| /Ruby/ | Exact match |  | /[a-z]/ | | Character range | |
| /RubyOcaml/ | Concatenation |  | /[^0-9]/ , /[^abc]/ | | Match not in class | |
| /(Ruby)(Ocaml)/ | Concatenation with grouping |  |  | |  | |
| /(Ruby|Ocaml)/ | Match either |  | Anchors/Boundaries | | | |
| /R(uby|Regular)/ | Match |  | ^ | | Beginning of line (after \n) | |
| /(Ruby)\*/ | Match 0 or more (in order) |  | $ | | End of line | |
| /(Ruby)+/ | Match 1 or more |  | \A | | Beginning of string (ignores \n) | |
| /(Ruby)?/ | Match 0 or 1 |  | \z | | End of string | |
| /(Ruby){3}/ | Match exactly 3 |  | \b | | Word boundary (change) | |
| /(Ruby){3,}/ | Match 3 or more |  | *(anchors and boundaries do not consume characters)* | | | |
| /(Ruby){3,5}/ | Match 3 to 5 |  |  | |  | |
|  |  |  | Metacharacters | | | |
| Extracting substrings | |  | ^ [ ] . $ { } \* ( ) \ + | ? < > | | Must be escaped | |
| $n (eg, $1, $2, …) | Back reference |  |  | |  | |
|  | *Matches groups in parens* |  | Special characters | | | |
|  | *Use (?:abc) to ignore group* |  | \\ | Backslash | [\b] | Backspace |
| s.scan(/pattern/) | Return array of matches |  | \n | New line | \r | Carriage return |
| s.scan(/(p1)(p2)/ | Return array of arrays |  | \t | Tab | \v | Vertical tab |
| str.scan(regexp) { |match| block } *short for* | |  | \f | Form feed | \e | Esc character |
| str.scan(regexp).each { |match| block } | |  | \xhh | | Hex character hh | |
|  | Applies code block to matches |  | \Oxxx | | Octal character xxx | |
|  | in order |  |  | |  | |
|  |  |  |  | |  | |
| Regexp class and objects | |  |  | |  | |
| Regexp.new(str) | Create Regexp object with str |  |  | |  | |
| Examples: | Regexp.new('\w+') |  |  | |  | |
|  | Regexp.new("abc" + "[0-9]{2}") |  |  | |  | |
| Regexp.new(regexp) | Create with literal regexp |  |  | |  | |
|  | Regexp.new(\Ruby\) |  |  | |  | |

1. Order of precedence: \*, {n}, + bind most tightly, then concatenate, then |

2. RegExp references

Official reference: <http://ruby-doc.org/core-2.4.0/Regexp.html>

Rubular online editor <https://rubular.com>