Regular Expressions

A regular expression describes a language using three operations.
A regular expression (RE) describes a language.

It uses the three regular operations. These are called union/or, concatenation and star.

Brackets ( and ) are used for grouping, just as in normal math.
The symbol + means union or or.

Example:

\[ 0 + 1 \]

means either a zero or a one.
The *concatenation* of two REs is obtained by writing the one after the other.

Example:

\[(0 + 1) \, 0\]

corresponds to \(\{00, 10\}\).

\[(0 + 1) \,(0 + \varepsilon)\]

corresponds to \(\{00, 0, 10, 1\}\).
The symbol $*$ is pronounced star and means zero or more copies.

Example:

$$a^*$$

corresponds to any string of $a$’s: \{ε, a, aa, aaa, …\}.

$$(0 + 1)^*$$

corresponds to all binary strings.
An RE for the language of all binary strings of length at least 2 that begin and end in the same symbol.
Example

An RE for the language of all binary strings of length at least 2 that begin and end in the same symbol.

\[0(0+1)^*0 + 1(0+1)^*1\]

Note *precedence* of regular operators: *star* always refers to smallest piece it can, *or* to largest piece it can.
Example

Consider the regular expression

\[((0+1)^*1+\varepsilon)(00)^*\ 00\]
Consider the regular expression

\[((0+1)^* 1 + \varepsilon) (00)^* 00\]

This RE is for the set of all binary strings that end with an even nonzero number of 0’s.

Note that different language to:

\[((0+1)^* (00)^* 00\]
If one forms RE by the or of REs $R$ and $S$, then result is union of $R$ and $S$.

If one forms RE by the concatenation of REs $R$ and $S$, then the result is all strings that can be formed by taking one string from $R$ and one string from $S$ and concatenating.

If one forms RE by taking the star of RE $R$, then the result is all strings that can be formed by taking any number of strings from the language of $R$ (possibly the same, possibly different), and concatenating.
If language $L$ is $\{ma, pa\}$ and language $M$ is $\{be, bop\}$, then

$L + M$ is $\{ma, pa, be, bop\}$;

$LM$ is $\{mabe, mabop, pabe, pabop\}$; and

$L^*$ is $\{\varepsilon, ma, pa, mama, \ldots, pamamapa, \ldots\}$.

Notation: If $\Sigma$ is some alphabet, then $\Sigma^*$ is the set of all strings using that alphabet.
An RE for Decimal Numbers

English: “Some digits followed maybe by a point and some more digits.”

RE:

\((\varepsilon + . \ D^*)\)

where \(D\) stands for a digit.
Kleene’s Theorem.  There is an FA for a language if and only there is an RE for the language.

Proof (to come) is algorithmic.

Regular language is one accepted by some FA or described by an RE.
Applications of REs

• Specify piece of programming language, e.g. real number. This allows automated production of tokenizer for identifying the pieces.

• Complex search and replace.

• Many UNIX commands take regular expressions.
Practice

Give an RE for each of the following three languages:

1. All binary strings with at least one 0
2. All binary strings with at most one 0
3. All binary strings starting and ending with 0
Solutions to Practice

1. \((0 + 1) \times 0(0 + 1)\) *
2. \(1 \times 1 \times 01\) *
3. \(0(0 + 1) \times 0 + 0\)

In each case several answers are possible.
A regular expression (RE) is built up from individual symbols using the three Kleene operators: union (+), concatenation, and star (*). The star of a language is obtained by all possible ways of concatenating strings of the language, repeats allowed; the empty string is always in the star of a language.