EXTENDED EXAMPLES

Numbered Groups

Matcher Objects

The `regex_replace` Function

Delroy A. Brinkerhoff

GROUPS

ARITHMETIC EXPRESSIONS

- May be a constant or a variable
- Recursively formed by combining with operators
- Grouped with parentheses and treated as a whole

REGULAR EXPRESSIONS

- `(r)`
  - Called a capturing or numbered group
  - Captures or saves matching text
- `(?:r)`
  - Non-capturing, non-numbered group
  - Consumes but does not save matching text
- Grouped expressions are treated as a whole

ADDING NUMBERED GROUPS

```
string re = "([Dd]eposit|[1-9][0-9]*):[^:]+:[^:]+:\\d*\\.\\d{2}";
if (! regex_match(entry, regex(re)))
    continue;
```

```
smatch m;
string re = "([Dd]eposit|[1-9][0-9]*):([^:]+):([^:]*):(\\d*\\.\\d{2})";

if (! regex_match(entry, m, regex(re) ))
    continue;
```

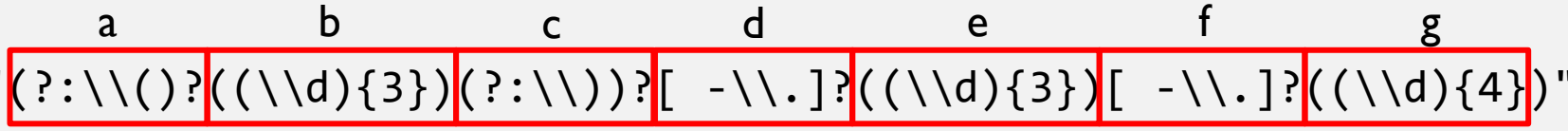
DATA EXTRACTION WITH smatch AND NUMBERED GROUPS

```
smatch m;
string re = "([Dd]eposit|[1-9][0-9]*):([^:]+):([^:]*):(\\d*\\.\\d{2})";

if (! regex_match(entry, m, regex(re) ))
    continue;

string type = m[1];
string date = m[2];
string to = m[3];
double amount = stod(m[4]);
```

NESTED & NON-CAPTURING GROUPS

- $((r_1)\{m,n\})(?:(r_2)?)((?:r_3)\{n\})(((r_4)))$
  - RE number capturing groups left to right in the order of the opening parenthesis
  - RE do not number non-capturing groups
  - There isn't a syntactic limit to how deeply RE can nest groups
  - The r's are regular expressions

REGULAR EXPRESSION GROUPS EXAMPLE



- (123) 456-7890
- 123-456-7890
- 123 456 7890
- 123.456.7890
- 1234567890

a. 0 or 1 opening parenthesis; no groups
b. 3 digits; groups 1 and 2
c. 0 or 1 closing parenthesis; no groups
d. 0 or 1 space, dash, or period; no groups
e. 3 digits; groups 3 and 4
f. 0 or 1 space, dash, or period; no groups
g. 4 digits; groups 5 and 6

CONSISTENT DATA FORMATS

```
if (regex_match(phone, regex("^$|^#.*")))
    continue;

string re = "(?:\\()?((\\d){3})(?:\\))?[ -\\.]?((\\d){3})[ -\\.]?((\\d){4})";
string format = "($1) $3-$5";

if (regex_match(phone, regex(re)))
    cout << regex_replace(phone, regex(re), format) << endl;
else
    cerr << "Unsupported format: " << phone << endl;
```