Random. Kinda.

Polyglot Programming DC 2015

Brock Wilcox
awwaiid@thelackthereof.org

```
$"='314747"7"84561"12"277057"10"53708"12"2466"25"'
;sub _{$;.=chr(rand(24)+64)}while($"){srand$";$"=~
s/.*?"//;rand(24)for(1.."$\"");$"=~s/.*?"//;_()for
(1..5)}$;=~y/BV/ ./;$;=~s/\w+/\u\L$&/g;print$;.$/
```

DEMO

```
awwaiid@mirabel:~/projects/perl/japh$ perl
$"='314747"7"84561"12"277057"10"53708"12"2466"25"'
;sub _{$;.=chr(rand(24)+64)}while($"){srand$";$"=~
s/.*?"//;rand(24)for(1.."$\"");$"=~s/.*?"//;_()for
(1..5)}$;=~y/BV/ ./;$;=~s/\w+/\u\L$&/g;print$;.$/

Just Another Perl Hacker.
```

JAPH is a fun obfuscation contest!

The game: Make a block of inscrutable code that is equivalent to:

```
print "Just Another Perl Hacker.\n"
```

Let's see how this one works.

Hint: I tend to prefer "abstract-concept" JAPHs more than just syntax trickery. But the trickery is kinda fun anyway.

```
# Built in thing that re-formats code
perl -MO=Deparse
```

```
# perl -MO=Deparse returns this

$" = '314747"7"84561"12"277057"10"53708"12"2466"25"';
sub _ {
    $; .= chr rand(24) + 64;
}
while ($") {
    srand $";
    $" =~ s/.*?"//;
    rand 24 foreach (1 .. qq[$"]);
    $" =~ s/.*?"//;
    _  foreach (1 .. 5);
}
$; =~ tr/BV/ ./;
$; =~ s/\w+/\u\L$&\E/g;
print $; . $/;
```

```
# Now we do some naming
# $" -> $magic
# $; -> $result
# $/ -> "\n" # Built-in obfu!

$magic = '314747"7"84561"12"277057"10"53708"12"2466"25"';
sub build {
    $result .= chr rand(24) + 64;
}
while ($magic) {
    srand $magic;
    $magic =~ s/.*?"//;
    rand 24 foreach (1 .. qq[$magic]);
    $magic =~ s/.*?"//;
    build() foreach (1 .. 5);
}
$result =~ tr/BV/ ./;
$result =~ s/\w+/\u\L$&\E/g;
print $result . "\n";
```

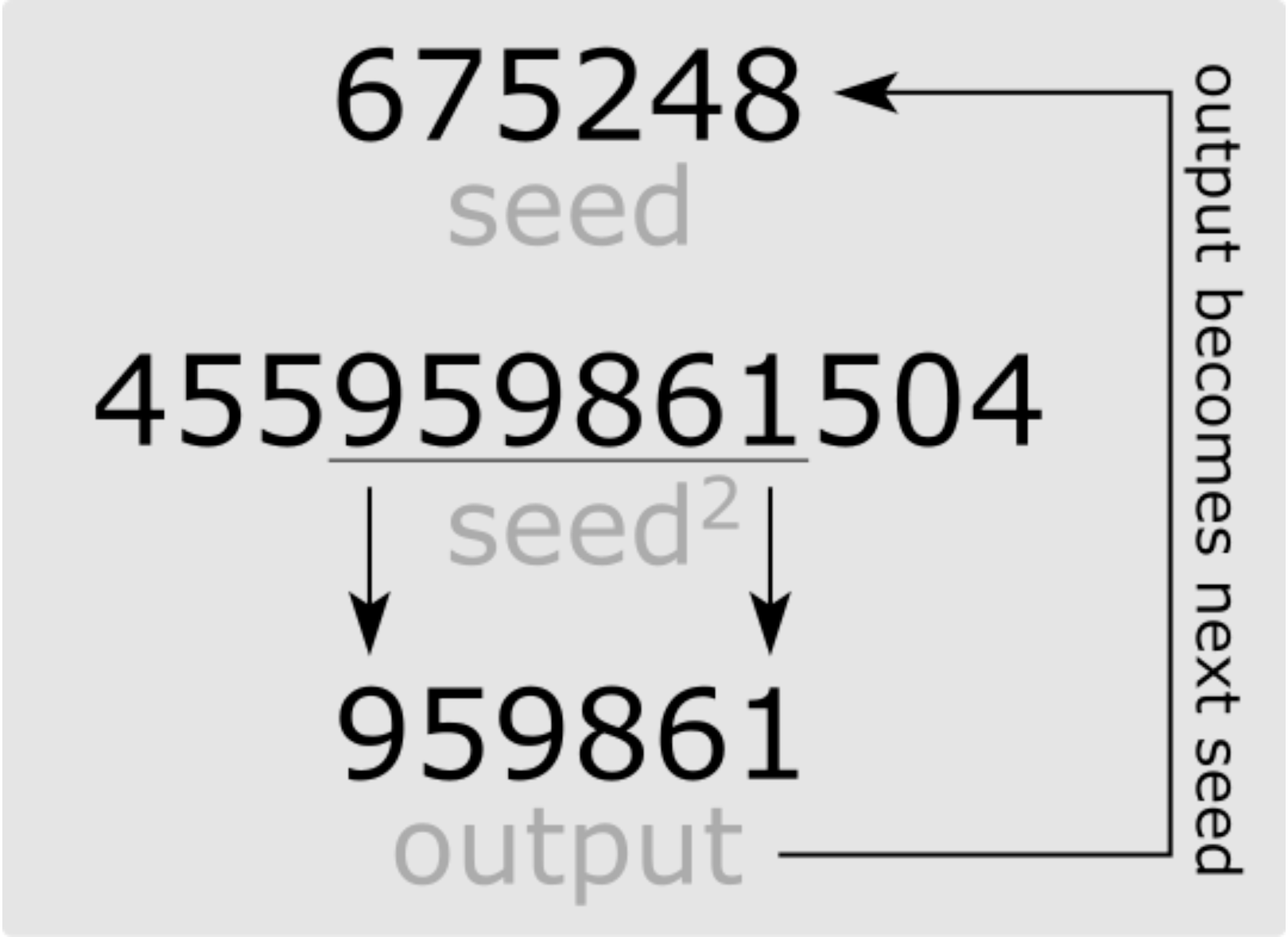
```
# Seed, offset, seed, offset, ...
$magic = '314747"7"84561"12"277057"10"53708"12"2466"25"';

# Take a random char and add it to the $result
sub build {
    $result .= chr rand(24) + 64;
}

while ($magic) {

    # Seed with the first number
    srand $magic;

    # Strip out the first number
    $magic =~ s/.*?"//;

    # jump forward $magic rand numbers (offset)
    rand 24 foreach (1 .. qq[$magic]);

    # Strip out the offset
    $magic =~ s/.*?"//;

    # Grab 5 random chars
    build() foreach (1 .. 5);
}

# Clean up some stuff
$result =~ tr/BV/ ./;

# Fix capitalization
$result =~ s/\w+/\u\L$&\E/g;

print $result . "\n";
```

In other words, go through some seeds, skip offset number of random numbers, grab the next 5 random numbers, turn them into characters.

Do some slight tweaking. Print the results.

Pseudo Random Number Generators (prng)

Middle Square Method


675248
seed
455959861504
seed²
959861
output
output becomes next seed

```
#!/usr/bin/env perl6

sub middle-square($size, $seed) {
  my $square = $seed ** 2;
  my $square_padded = sprintf("%0{ $size*2 }d", $square);
  $square_padded.split('')[$size/2+1..$size/2+$size].join;
}

sub MAIN($size, $seed) {
  my $new-val = middle-square($size, $seed);
  say $new-val;
  #  sleep 0.5;
  MAIN($size, $new-val);
}
```

DEMO

```
# Who put the Pseu in Pseudo?

./rand.p6 6 675248    # gets stuck in 625000 loop
./rand.p6 10 28373744 # Ends up with 0000000000!
```

References!

Middle Square Method
https://en.wikipedia.org/wiki/Middle-square_method

Very common one
https://en.wikipedia.org/wiki/Mersenne_Twister

Interesting Finite-Automata method
https://en.wikipedia.org/wiki/Rule_30

Moar!
https://en.wikipedia.org/wiki/List_of_random_number_generators

THE END