# Quine を書こう
# ～自己を出力する不思議なプログラム～

nsfisis(いまむら)

## はじめに～Quine とは～

### Quine の定義

Quine(クワイン)とは、自分自身のソースコードと一致する文字列を出力するようなプログラムのことです。PHP なら、

```
$ php a.php > output.txt
```

と実行したとき、`output.txt` と `a.php` が完全に一致するような `a.php` を「Quine」と呼びます。一見すると不可能にすら思えますが、ほとんどの言語で簡単に書くことができます。

例として、PHP で書かれた Quine を一つ見てみましょう。

**simple.php**

```
<?eval($s='printf("<?eval(\$s=%c%s%c);
\n",39,$s,39);');
```

※紙面の都合上改行が挟まっていますが、一行で入力してください。背景色が同じなら同じ行です。

これを実行すると、上記の文字列がそのまま出力されます。

```
$ php simple.php
<?eval($s='printf("<?eval(\$s=%c%s%c);\n",39,
$s,39);');
```

これが Quine の定義です。

なお、この記事に記載しているコードはすべて GitHub のリポジトリへアップロードしていますので、実行の際はそちらを用いることをお勧めします(記事末尾に URL と QR コードがあります)。

### 自明な Quine

さて、上記の定義を見た勘の良い方なら、こんなことに気付くかもしれません。「空のファイルでも条件を満たすのでは？」`main` 関数を書かなければならない言語ならばいざ知らず、PHP のような言語なら空のソースコードは適格なプログラムです。これを「実行」すれば確かに空の出力が得られ、それは自己のソースコードと一致します。

更に PHP の場合、PHP タグの外に書かれたテキストはそのまま出力されるため、次のような PHP コードも考えられます。

**illegal-quine.php**

```
Hello, PHPerKaigi!
```

これも `php` コマンドで実行すればソースコードと同じ出力が得られます。また、自身をファイルとして読み込む次のようなコードも考えられるかもしれません。

**illegal-quine2.php**

```
<?echo file_get_contents(__FILE__);
```

ただ、一般的に「Quine」と言ったときにこのようなコードは認めないことが多いようです(認めたとしても考察するほどの面白みはありませんが)。

## Quine を書いてみよう！

### Let's Try!

さあ、まずは何も考えずに Quine の作成に挑戦してみましょう。文字列を出力するプログラムですから、次のようになるはずです。

**try-quine.php**

```
<?php echo '...';
```

この `...` に `try-quine.php` の内容そのものを書くことができれば完成ですね！では書いてみましょう。

**try-quine2.php**

```
<?php echo '<?php echo \'...\';';
```

この `...` に `try-quine2.php` の内容そのものを書くことができれば完成ですね！では書いてみましょう。

**try-quine3.php**

```
<?php echo '<?php echo \'<?php echo \'<?php
echo \\\'...\\\';\';\';';
```

この `...` に `try-quine3.php` の内容そのものを書くことができれば……いえ、そろそろ紙面がもったいないので止めておきましょう。

このアプローチでは、この調子で無限に置き換え作業を続けることになってしまいます。根本的な問題は、文字列の中でそれ自身を参照する必要があることです。一見すると無限の入れ子構造が必要に思えます。

### 解決のアイデア(その一)

これを解決する最初のアイデアは、変数を使った置き換えです。次の例を見てください。

**try-quine4.php**

```
<?php $s = '<?php $s = \'...\'; echo
strtr($s, [str_repeat(\'.\', 3) => $s]);';
echo strtr($s, [str_repeat('.', 3) => $s]);
```

`echo` で出力していた文字列を一度変数 `$s` へと代入します。あとは、先ほどから繰り返していた「`...` に `try-quine4.php` の内容そのものを書く」を文字列置換によって実現します。`strtr()` を使って `$s` 内の `...` を `$s` へと置き換えています。

ここで、`strtr($s, ['...' => $s])` とは書けないことに注意してください。もしこう書いてしまうと、その `...` もまた `$s` へと置き換えられてしまうからです。

これで上手くいくでしょうか。動かしてみましょう。

```
$ php try-quine4.php
<?php $s = '<?php $s = '...'; echo strtr($s,
[str_repeat('.', 3) => $s]);'; echo strtr($s,
[str_repeat('.', 3) => $s]);
```

元のソースコードにとても似ていますが少しだけ違いがありますね。`$s` の中でエスケープしていた `\'` が単独の `'` に戻ってしまいました。元々のソースでエスケープされていた文字は、出力時にもエスケープしてやる必要があります。エスケープ処理を自分で実装してもいいのですが、PHP にはおあつらえ向きの関数 `addslashes()` があります。これは、クォートやバックスラッシュ、NUL バイトをエスケープしてくれる関数です。もちろん Quine を実装するために用意したに違いありません。今回はこれを使いましょう。

**try-quine5.php**

```
<?php $s = '<?php $s = \'...\'; echo
strtr($s, [str_repeat(\'.\', 3) =>
addslashes($s)]);'; echo strtr($s,
[str_repeat('.', 3) => addslashes($s)]);
```

これを実行してみます。

```
$ php try-quine5.php
<?php $s = '<?php $s = \'...\'; echo
strtr($s, [str_repeat(\'.\', 3) =>
addslashes($s)]);'; echo strtr($s,
[str_repeat('.', 3) => addslashes($s)]);
```

元のソースコードと完全に一致する出力が得られました！これにて Quine 完成です。なお、`diff` コマンドを使うともう少しスマートに確かめられます。

```
$ php try-quine5.php > result
$ diff -s try-quine5.php result
Files try-quine5.php and result are identical
```

シェルによってはプロセス置換を使ってもいいですね。

```
$ diff -s try-quine5.php <(php try-
quine5.php)
```

Quine の基本構造は次のようになります。

1. ソースコード全体を表す変数を、自分自身を何らかのプレースホルダで置き換えた上で用意する
2. その変数のプレースホルダを、その変数の値で置換する
3. 結果を出力する

## 解決のアイデア（その二）

Quine 自体は前述の方法で作れるようになりましたが、少々重複が多いのが気になるところです。もっと簡潔に書けないでしょうか？

実は `eval()`（動的な文字列をプログラムとしてその場で解釈し、実行する処理）を持つ言語では、より短く Quine を書けることがあります。PHP でも `eval()` を活用することでより簡潔な Quine を書くことができます。

先ほどの `try-quine5.php` において、`$s` 中の `...` を `$s` 自身で置き換えた文字列とは、`try-quine5.php` のソースコードそのものです。では、これをそのまま実行すれば `try-quine5.php` を実行したことになりますね？文字列を PHP コードとして実行する処理、つまり `eval()` の出番です。

**try-quine6.php**

```
<?php $s = 'echo strtr("<?php \$s = \'...\';
eval(\$s);", [str_repeat(\'.\', 3) =>
addcslashes($s, chr(39))]);'; eval($s);
```

このコードで `eval()` される文字列は以下のとおりです。

```
echo strtr("<?php \$s = '...'; eval(\$s);",
[str_repeat('.', 3) => addcslashes($s,
chr(39))]);
```

これを見ると、

1. ソースコード全体を表す変数を、自分自身を何らかのプレースホルダで置き換えた上で用意する
2. その変数のプレースホルダを、その変数の値で置換する
3. 結果を出力する

という Quine の基本処理が実行されていることがわかります。`try-quine5.php` と比べ、`strtr()` 関連の処理が一回しか書かれていないことがわかるでしょうか。

なお、`eval()` を使う場合、`addslashes()` によってダブルクォート等がエスケープされてしまうと有効な PHP コードではなくなってしまうので、シングルクォートだけをエスケープするよう `addcslashes()` を用いています。こちらはエスケープ対象の文字を指定することができ、`chr(39)` とは `'` のことです。

冒頭に掲載した `simple.php` は、この考え方で更に短くしたものです。置換処理自体は `printf()` の `%s` 指定子でおこなっており、エスケープ対象であるシングルクォートの出力も `printf()` の `%c` 指定子を使っています。

## 応用 Quine

基本的な Quine の仕組みを理解すれば、より複雑で面白い挙動をする Quine を作ることができます。ここでは代表的な応用例を紹介します。

### Quine アスキーアート

Quine は本質的にはただの文字列の出力プログラムです。ソースコードを特定の形に整形することで、面白い見た目の Quine を作れます。

こちらをご覧ください。

**quine-japan.php**

```
<?eval(preg_replace("/\s/","",$s='
p                                r
i              ntf("             <
?             eval(pr            e
g            _replace(           \
"             /\s/\",            \
"              \",\$             s
=                                %
c%s%c));\n",047,$s,047,39,39);'));
```

日の丸の形をした Quine です(比率その他には目をつぶってください)。これは `$s` の中にある空白や改行を、`eval()` へ渡す前に `preg_replace()` で取り除くことによって整形を実現しています。

この類の Quine は比較的簡単なテクニックで作ることができますが、その割に見た目に華があるので、Quine 初心者が作ってみるのにお勧めです。

整形済みのソースコードを文字列として持っておいてそのままそれを出力する方式(このコードで採用しているもの)と、整形されていないソースコードを文字列として持っておいて出力の際に動的に整形処理をおこなう方式の二種類があります。

### 状態を持つ Quine

Quine の定義上、通常は実行のたびに同じ出力を返しますが、実行ごとに出力が変化する Quine も作ることができます(厳密な意味での Quine ではありませんが、「変則 Quine」といった呼び名で Quine の一種として扱われることが多いようです)。

これを実現するためには、ソースコードの一部に状態を持たせておき、置換処理をおこなう中で状態の更新をおこないます。単純な例を見てみましょう。

**countup-quine.php**

```
<?$n=0;eval($s='printf("<?\$n=%d;eval(\
$s=%c%s%c);\n",$n+1,39,$s,39);');
```

この Quine は、実行のたびに `$n` の値が 1 ずつ増えていきます。

これだけでは面白くないので、もう少し複雑な例を見てみましょう。こちらです。

**quine-puzzle.php**

```
<?php $z='bcdefghia';eval($s=strtr('$M="array_map";$S="st
r_split";$C="chr";$zp=strpos($z,"a");[$dx,$dy]=match($arg
v[1              ]??              nul              l){
"h"              =>[              1,0              ],"
j"=         >[0   ,-1   ],"k"=>[0   ,1]   ,"l"=>[-1   ,0]
,de         fau   lt=   >[0,0],};   $zx   =$zp%3;$z   y=i
ntd         iv(   $zp         ,3)   ;$s         x=$   zx+
$dx         ;$s   y=$         zy+   $dy         ;if   ($s
x<0         ||2   <$s   x)$sx=$zx   ;if   ($sy<0||2   <$s
y)$         sy=   $zy   ;$sp=$sy*   3+$   sx;[$z[$s   p],
$z[         $zp   ]]=   [$z         [$z         p],   $z[
$sp         ]];   ech   o("         <?p         hp"   .$C
(32         ).$   C(3   6)."z=".$   C(3   9).$z.$C(   39)
.";         eva   l("   .$C(36)."   s=s   trtr(".$C   (39
));              $n=              $M(              $S,
$S(              "00              000              111
1141424414143341142414424344111143434",5));$m=$M($S,$S("0
00001100101111",3));$i=(new("ArrayObject")($S($s."//".$s,
)))              ->g              etI              ter
ato              r()              ;$B              =fn
($_   =1)   =>$   M(f   n()=>prin   t([   $i->curre   nt(
),$   i->   nex   t()   ][0]),ran   ge(   1,$_*3));   $W=
fn(   $_=   1)=   >pr   int         (st   r_r         epe
at(   $C(   32)   ,$_   *3)         );$   N=f         n()
=>p   rint($C(1   0))   ;$B(7);$N   ();   for($y=0;   $y<
3;$   y++){$B(1   9);   $N();$B()   ;$W   (5);$B();   $W(
5);         $B(   );$         W(5   );$   B()   ;$N   ();
$B(         );$   W(5         );$   B()   ;$W   (5)   ;$B
();         $W(   5);   $B();$N()   ;fo   r($l=0;$l   <10
;$l         ++)   {$B   ();for($x   =0;   $x<3;$x++   ){$
W(1              );$              M(f              n($
_)=              >$_              ?$B              ():
$W(),$m[$n[$M("ord",$S($z))[$y*3+$x]-97][intdiv($l,2)]]);
$W(1);$B();}$N();}$B();$W(5);$B();$W(5);$B();$W(5);$B();$
N()              ;$B              ();              $W(
5);              $B(              );$              W(5
);$   B();$W(5)   ;$B   ();$N();$   B(1              9);
$N(   );}$B(9);   ech   o($C(39).   ",[              $C(
32)         =>"   .$C   (34   ).$   C(3              4).
",$         C(1   0)=   >".   $C(   34)              .$C
(34         )."   ]))   ;");//$M=   "ar              ray
_ma         p";   $S=   "str_spli   t";              $C=
"ch         r";   $zp   =st   rpo   s($              z,"
a")         ;[$   dx,   $dy   ]=m   atc              h($
arg         v[1   ]??   null){"h"   =>[              1,0
],"         j"=   >[0   ,-1],"k"=   >[0              ,1]
,"l              "=>              [-1              ,0]
,de              fau              lt=              >[0
,0],};$zx=$zp%3;$zy=intdiv($zp,3);$sx=$zx+$dx;$sy=$zy+//$
M="array_map";$S="str_split',[chr(32)=>"",chr(10)=>""]));
```

この Quine はスライドパズルになっており、引数で渡す値によって出力が変化します。「8」を右に動かしたいなら `php quine-puzzle.php l` と、「6」を下に動かしたいなら `php quine-puzzle.php j` と実行すれば、次の盤面が新たなソースコードとして出力されます。そのソースコードももちろんパズルになっており、もう一度実行することで二手動かした状態の盤面が得られます。操作は Vim の移動キーと対応しており、それぞれ `h` が左、`j` が下、`k` が上、`l` が右移動です。

## おわりに

今回紹介したのは Quine のほんの一端にすぎません。より深く Quine を知るには、末尾の参考文献にも挙げている『あなたの知らない超絶技巧プログラミングの世界』という書籍がお勧めです。本記事、特に「Quine を書いてみよう！」の章は、同書の第三章の説明を大いに参考にさせていただきました。

この記事で Quine に興味を持っていただけたなら、ぜひご自身だけの Quine を書いてみてはいかがでしょうか。

最後に、拙作の Quine をいくつか紹介します。

- https://github.com/nsfisis/9-puzzle-quine.php
  - PHP
  - 記事中に記載したスライドパズル Quine
- https://github.com/nsfisis/cohackpp
  - PHP
  - 友人の結婚祝いに贈った Quine。実行すると「ご結婚おめでとうございます」と一文字ずつ表示される
- https://github.com/nsfisis/phperbiglt-2025
  - Python + PHP
  - 「巳」の形の Python コードと「午」形の PHP コードが交互に切り替わる Quine
- https://github.com/nsfisis/twitter2x-quine
  - Ruby
  - Twitter のロゴと X のロゴが交互に切り替わる Quine
- https://github.com/nsfisis/pong-wars-quine.rb
  - Ruby
  - 一時期 SNS で話題になった「Pong Wars」を実装した Quine
- https://github.com/nsfisis/trick-2025
  - Ruby
  - Ruby プログラムにルビを振って出力する Quine
  - TRICK 2025 というコンテストで入賞した作品

また、本記事中に出てくるソースコードは下記の GitHub リポジトリに同じファイル名でアップロードしています。`src/` ディレクトリ以下を参照してください。また、`test.sh` を走らせることで Quine になっているかどうかのテストが可能です。

https://github.com/nsfisis/phperkaigi-2026-brochure-article

## 参考文献


- 遠藤侑介『あなたの知らない超絶技巧プログラミングの世界』技術評論社、2015 年
  - https://gihyo.jp/book/2015/978-4-7741-7643-7


余白を埋める Quine、ヨハクワイン

**blanquine.php**

```
                    <?eval(
                  preg_replace
                 ("/\s/" ,"",$s='
                printf(      "<?eval
               (\n%spr          eg_repl
             ace\n%s              (\"/\s/
           \"%s,\"                  \",\$s=
         %c%s%c)                    );\n",(
       $r="str                        _repeat
     ")($w=c                            hr(32),
   28),$r(                                $w,26),
 $w,39,$  s,39);/*printf("<?eval(\n%spreg_re  place\n
 %s(\"/\    s/\"%s,\"\",\$s=%c%s%c));\n",($r="    str_rep
eat")($      w=chr(32),28),$r($w,26),$w,39,$s,3      9);prin
                    tf("<?e
                    val(\n%
                    spreg_r
   eplace\n%s(\"/\s/\"%s,\"\",\$s=%c%s%c));\n",($r="
   str_repeat")($w=chr(32),28),$r($w,26),$w,39,$s,39
   );printf("<?eval(\n%spreg_replace\n%s(\"/\s/\"%s,
                    \"\",\$
        s=%c%s%       c));\n"         ,($r="s
      tr_repe         at")($w         =chr(32
     ),28),$          r($w,26          ),$w,39
    ,$s,39)    ;printf("<?eva           l(\n%sp
  reg_rep        lace\n%s(\"             /\s/\"%
 s,\"\",            \$s=%c%s             %c));\n
",($r="               str_r               epeat")
```

```
                    ($w=chr
                   (32),28
                  ),$r($w
                 ,26),$w
                ,39,$s,
               39);pri
              ntf("<?
        eval(\n%spreg_replace\n%s(\"/\s/\"%s,\"
        \",\$s=%c%s%c));\n",($r="str_repeat")($
        w=chr(32),28),$r($w,26),$w,39,$s,39);pr
        intf("<?eval(\n%spreg_replace\n%s(\"/\s
        /\"%s,\                         "\",\$s
        =%c%s%c                         ));\n",
        ($r="st                         r_repea
        t")($w=                         chr(32)
        ,28),$r($w,26),$w,39,$s,39);printf("<?e
        val(\n%spreg_replace\n%s(\"/\s/\"%s,\"\
        ",\$s=%c%s%c));\n",($r="str_repeat")($w
        =chr(32),28),$r($w,26),$w,39,$s,39);pri
        ntf("<?                         eval(\n
        %spreg_                         replace
        \n%s(\"                         /\s/\"%
        s,\"\",                         \$s=%c%
        s%c));\n",($r="str_repeat")($w=chr(32),
        28),$r($w,26),$w,39,$s,39);printf("<?ev
        al(\n%spreg_replace\n%s(\"/\s/\"%s,\"\"
        ,\$s=%c%s%c));\n",($r="str_repeat*/'));
```