The Penrose package

Andrew Stacey
loopspace@mathforge.org

v1.4 from 2021/01/22

Contents

1 Introduction

This is a TikZ library for drawing Penrose tiles (kite/dart, rhombus, and pentagon versions). It provides two methods of drawing: one in which an automatic pattern is built, and one where the tiles can be placed "by hand". The tiles can be shaped and (hopefully!) still fit together. For full user documentation, see the `penrose.pdf` file.

2 Implementation

1 ⟨*package⟩

2 ⟨@@=penrose⟩

2.1 Initialisation

We use the `spath3` library for manipulating the paths that will make up the tiles.

```
3 \ProvidesExplFile {tikzlibrarypenrose.code.tex} {2021/01/22} {1.4} {TikZ pics for Penrose an
4 \RequirePackage{spath3}
5 \usetikzlibrary{spath3}
```

Now we move in to the realm of LaTeX3.

```
6 \ExplSyntaxOn
```

Start with some basic paths (lines) for the sides of the tiles so that we know that we have well-defined tiles at the outset. These are globally defined as we will frequently want to define them in one tikzpicture and use them in another.

```
\tl_new:N \g__penrose_side_a_tl
\tl_new:N \g__penrose_side_b_tl
\tl_new:N \g__penrose_side_c_tl
\tl_new:N \g__penrose_side_d_tl
\tl_new:N \g__penrose_side_e_tl
\tl_new:N \g__penrose_side_A_tl
\tl_new:N \g__penrose_side_B_tl
\tl_new:N \g__penrose_side_C_tl
\tl_new:N \g__penrose_side_D_tl
\tl_new:N \g__penrose_side_E_tl
\tl_gset:Nn \g__penrose_side_a_tl
{
  \pgfsyssoftpath@movetotoken{0pt}{0pt}
  \pgfsyssoftpath@linetotoken{1pt}{0pt}
}
\tl_gset_eq:NN \g__penrose_side_b_tl \g__penrose_side_a_tl
\tl_gset_eq:NN \g__penrose_side_c_tl \g__penrose_side_a_tl
\tl_gset_eq:NN \g__penrose_side_d_tl \g__penrose_side_a_tl
\tl_gset_eq:NN \g__penrose_side_e_tl \g__penrose_side_a_tl
\tl_gset_eq:NN \g__penrose_side_A_tl \g__penrose_side_a_tl
\tl_gset_eq:NN \g__penrose_side_B_tl \g__penrose_side_a_tl
\tl_gset_eq:NN \g__penrose_side_C_tl \g__penrose_side_a_tl
\tl_gset_eq:NN \g__penrose_side_D_tl \g__penrose_side_a_tl
\tl_gset_eq:NN \g__penrose_side_E_tl \g__penrose_side_a_tl
```

\l__penrose_tmpa_fp
\l__penrose_tmpb_fp
\l__penrose_tmpc_fp
\l__penrose_tmpa_tl
\l__penrose_tmpb_tl
\l__penrose_tmpc_tl
\l__penrose_tmpd_tl
\l__penrose_tmpa_int
\l__penrose_tmpb_int
\l__penrose_xa_dim
\l__penrose_ya_dim
\l__penrose_xb_dim
\l__penrose_yb_dim
\l__penrose_tmpa_prop

We need a few temporary variables to hold intermediate calculations.

```
\fp_new:N \l__penrose_tmpa_fp
\fp_new:N \l__penrose_tmpb_fp
\fp_new:N \l__penrose_tmpc_fp
\str_new:N \l__penrose_tmpa_str
\str_new:N \l__penrose_tmpb_str
\tl_new:N \l__penrose_tmpa_tl
\tl_new:N \l__penrose_tmpb_tl
\tl_new:N \l__penrose_tmpc_tl
\tl_new:N \l__penrose_tmpd_tl
\tl_new:N \l__penrose_tmp_tile_path_tl
\tl_new:N \g__penrose_smuggle_tl
\int_new:N \l__penrose_tmpa_int
\int_new:N \l__penrose_tmpb_int
\dim_new:N \l__penrose_xa_dim
\dim_new:N \l__penrose_ya_dim
\dim_new:N \l__penrose_xb_dim
\dim_new:N \l__penrose_yb_dim
\prop_new:N \l__penrose_tmpa_prop
```

(*End definition for* `\l__penrose_tmpa_fp` *and others.*)

2.2 Creating the Tiles

__penrose_normalise_path:Nn

When defining the path for a side, we normalise so that it starts at the origin and ends at `(1pt,0pt)`.

```
\cs_new_nopar:Npn \__penrose_normalise_path:Nn #1#2
{
```

Get the initial point of the path and convert to floating point.

```
  \group_begin:
  \spath_initialpoint:Nn \l__penrose_tmpa_tl {#2}
  \fp_set:Nn \l__penrose_tmpa_fp {\tl_head:N \l__penrose_tmpa_tl}
  \tl_set:Nx \l__penrose_tmpa_tl {\tl_tail:N \l__penrose_tmpa_tl}
  \fp_set:Nn \l__penrose_tmpb_fp {\tl_head:N \l__penrose_tmpa_tl}
```

Get the final point of the path, and compute the difference of the final and initial points.

The resulting numbers, say a and b, will be put into a matrix to rotate and scale the path. The formula for the matrix is:

$$\frac{1}{a^2+b^2}\begin{bmatrix} a & b \\ -b & a \end{bmatrix}$$

```
  \spath_finalpoint:Nn \l__penrose_tmpa_tl {#2}
  \fp_set:Nn \l__penrose_tmpa_fp
  {\tl_head:N \l__penrose_tmpa_tl - \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpa_tl {\tl_tail:N \l__penrose_tmpa_tl}
  \fp_set:Nn \l__penrose_tmpb_fp
  {\tl_head:N \l__penrose_tmpa_tl - \l__penrose_tmpb_fp}
```

Now compute the square of the length of the path for scaling.

```
  \fp_set:Nn \l__penrose_tmpc_fp
  {\l__penrose_tmpa_fp^2 + \l__penrose_tmpb_fp^2}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_tmpa_fp/\l__penrose_tmpc_fp}
  \fp_set:Nn \l__penrose_tmpb_fp {\l__penrose_tmpb_fp/\l__penrose_tmpc_fp}
  \fp_set:Nn \l__penrose_tmpc_fp {-\l__penrose_tmpb_fp}
```

Now construct the matrix.

```
  \tl_set:Nx \l__penrose_tmpb_tl
  {
    {\fp_use:N \l__penrose_tmpa_fp}
    {\fp_use:N \l__penrose_tmpc_fp} % swapped
    {\fp_use:N \l__penrose_tmpb_fp} % swapped
    {\fp_use:N \l__penrose_tmpa_fp}
  }
```

Get the initial point back again for the translation part.

```
  \spath_initialpoint:Nn \l__penrose_tmpa_tl {#2}
```

But we need to premultiply by the matrix because of how the transformations are applied.

```
  \fp_set:Nn \l__penrose_tmpa_fp
  {
    (-1) * \l__penrose_tmpa_fp * \tl_head:N \l__penrose_tmpa_tl
    + (-1) * \l__penrose_tmpb_fp * \tl_tail:N \l__penrose_tmpa_tl
  }
  \fp_set:Nn \l__penrose_tmpb_fp
  {
    (-1) * \l__penrose_tmpa_fp * \tl_tail:N \l__penrose_tmpa_tl
    +  \l__penrose_tmpb_fp * \tl_head:N \l__penrose_tmpa_tl
  }
```

Finally, we apply the transformation to the path.

```
  \tl_put_right:Nx \l__penrose_tmpb_tl {
    {\fp_to_dim:N \l__penrose_tmpa_fp}
    {\fp_to_dim:N \l__penrose_tmpb_fp}
  }
  \spath_transform:NnV \l__penrose_tmpa_tl {#2} \l__penrose_tmpb_tl
  \tl_gset_eq:NN \g__penrose_smuggle_tl \l__penrose_tmpa_tl
  \group_end:
  \tl_set_eq:NN #1 \g__penrose_smuggle_tl
  \tl_gclear:N \g__penrose_smuggle_tl
}
\cs_generate_variant:Nn \__penrose_normalise_path:Nn {NV, cn, cV}
\cs_new_protected_nopar:Npn \__penrose_normalise_path:N #1
{
  \__penrose_normalise_path:NV #1#1
}
\cs_generate_variant:Nn \__penrose_normalise_path:N {c}
```

(*End definition for* `\__penrose_normalise_path:Nn`.)

`\SetPenrosePath` This sets the path corresponding to a particular side to the current path, and normalises it.

```
\NewDocumentCommand \SetPenrosePath { m }
{
  \pgfsyssoftpath@getcurrentpath\l__penrose_tmpa_tl
  \__penrose_normalise_path:N \l__penrose_tmpa_tl
  \tl_gset_eq:cN {g__penrose_side_#1_tl} \l__penrose_tmpa_tl
}
```

`\tikz_scan_point:n` This is a wrapper around `\tikz@scan@one@point` to make it easier to use with LaTeX3 variables.

```
\cs_new_nopar:Npn \tikz_scan_point:n #1
{
  \tikz@scan@one@point\pgfutil@firstofone#1\relax
}
\cs_generate_variant:Nn \tikz_scan_point:n {V}
```

(*End definition for* `\tikz_scan_point:n`.)

`\__penrose_make_tile:nnn` This builds the tile path from its pieces. The arguments are the name of the tile, the descriptions of the sides, and a token list of the coordinates.

```
\cs_new_nopar:Npn \__penrose_make_tile:nnn #1#2#3
{
```

Get the first coordinate and initialise the path with a move to this point.

```
  \group_begin:
  \tl_set:Nn \l__penrose_tmpa_tl {#3}
  \tl_set:Nx \l__penrose_tmpb_tl {\tl_head:N \l__penrose_tmpa_tl}
  \tl_set:Nn \l__penrose_tmpa_tl {\pgfsyssoftpath@movetotoken}
  \tikz_scan_point:V \l__penrose_tmpb_tl
  \tl_put_right:Nx \l__penrose_tmpa_tl
  {
```

```
    {\dim_use:N \pgf@x}{\dim_use:N \pgf@y}
  }
  \tl_set_eq:NN \l__penrose_tmp_tile_path_tl \l__penrose_tmpa_tl
```

Now we have our path initialised, we can start appending the side paths according to the specification in the second argument.

We append the initial coordinate to the end of the list to make a closed cycle.

```
  \tl_set:Nn \l__penrose_tmpa_tl {#3}
  \tl_put_right:Nx \l__penrose_tmpa_tl {{\tl_head:N \l__penrose_tmpa_tl}}
```

Now we walk through the description of the sides, adding the specified paths to our tile path.

```
  \tl_map_inline:nn {#2} {
```

Clone the path for this side.

```
    \tl_set:Nx \l__penrose_tmpc_tl {\tl_head:n {##1}}
    \tl_set_eq:Nc \l__penrose_tmpd_tl {g__penrose_side_ \tl_use:N \l__penrose_tmpc_tl _tl}
```

Strip off the next coordinate, and convert it to a point.

```
    \tl_set:Nx \l__penrose_tmpb_tl {\tl_head:N \l__penrose_tmpa_tl}
    \tl_set:Nx \l__penrose_tmpa_tl {\tl_tail:N \l__penrose_tmpa_tl}
    \tikz_scan_point:V \l__penrose_tmpb_tl
```

Store the resulting coordinate.

```
    \fp_set:Nn \l__penrose_tmpa_fp { \pgf@x }
    \fp_set:Nn \l__penrose_tmpb_fp { \pgf@y }
```

Now get the next coordinate.

```
    \tl_set:Nx \l__penrose_tmpb_tl {\tl_head:N \l__penrose_tmpa_tl}
    \tikz_scan_point:V \l__penrose_tmpb_tl
```

We want the difference between the two coordinates.

```
    \fp_set:Nn \l__penrose_tmpa_fp {\pgf@x - \l__penrose_tmpa_fp}
    \fp_set:Nn \l__penrose_tmpb_fp {\pgf@y - \l__penrose_tmpb_fp}
```

This is converted into a transformation matrix.

```
    \fp_set:Nn \l__penrose_tmpc_fp {-\l__penrose_tmpb_fp}
    \tl_set:Nx \l__penrose_tmpb_tl
    {
      {\fp_use:N \l__penrose_tmpa_fp}
      {\fp_use:N \l__penrose_tmpb_fp} % not swapped
      {\fp_use:N \l__penrose_tmpc_fp} % not swapped
      {\fp_use:N \l__penrose_tmpa_fp}
      {0}
      {0}
    }
```

The transformation is applied to the cloned path.

```
    \spath_transform:NV \l__penrose_tmpd_tl \l__penrose_tmpb_tl
```

And this is welded to the tile path.

```
    \spath_weld:NV \l__penrose_tmp_tile_path_tl \l__penrose_tmpd_tl
  }
```

At the end we close the path.

```
  \spath_close:N \l__penrose_tmp_tile_path_tl
  \tl_gset_eq:NN \g__penrose_smuggle_tl \l__penrose_tmp_tile_path_tl
  \group_end:
  \tl_gclear_new:c {g__penrose_tile_#1_tl}
  \tl_gset_eq:cN {g__penrose_tile_#1_tl} \g__penrose_smuggle_tl
  \tl_gclear:N \g__penrose_smuggle_tl
}
```

(*End definition for* `\__penrose_make_tile:nnn`.)

`\__penrose_make_tile:nn` A wrapper around the above which allows us to specify the second two arguments as two items in a token list.

```
\cs_new_nopar:Npn \__penrose_make_tile:nn #1#2
{
  \__penrose_make_tile:nnn {#1} #2
}
\cs_generate_variant:Nn \__penrose_make_tile:nn {nV}
```

(*End definition for* `\__penrose_make_tile:nn`.)

2.3 Specifying the Tiles

The tile specifications are contained in a `prop`.

```
\prop_new:N \g__penrose_tiles_prop
```

`\__penrose_add_coordinate:Nnn` `\__penrose_add_coordinate:w` Process a coordinate through `fp` and adds it to a token list.

```
\cs_new_nopar:Npn \__penrose_add_coordinate:Nnn #1#2#3 {
  \fp_set:Nn \l__penrose_tmpa_fp{#2}
  \fp_set:Nn \l__penrose_tmpb_fp{#3}
  \tl_put_right:Nx #1
  {
    {(\fp_use:N \l__penrose_tmpa_fp, \fp_use:N \l__penrose_tmpb_fp)}
  }
}
```

Wrapper around the add coordinate command to split at a comma.

```
\cs_new_nopar:Npn \__penrose_add_coordinate:w #1#2,#3 \q_stop
{
  \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {#2}{#3}
}
```

(*End definition for* `\__penrose_add_coordinate:Nnn` *and* `\__penrose_add_coordinate:w`.)

Now we specify the tiles. The specification is a clockwise list of the vertices together with the labels of the corresponding sides. There are three basic paths, `a`, `b`, `c`, and their complements (which are capitalised).

- Thin Rhombus.

```
    \tl_clear:N \l__penrose_tmpa_tl
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {0}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(18)}{sind(18)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {2*cosd(18)}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(18)}{-sind(18)}
```

```
    \prop_gput:Nnx \g__penrose_tiles_prop {thin~ rhombus}
    {{a A B b} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Thick Rhombus.

```
    \tl_clear:N \l__penrose_tmpa_tl
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {0}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(36)}{sind(36)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {2*cosd(36)}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(36)}{-sind(36)}

    \prop_gput:Nnx \g__penrose_tiles_prop {thick~ rhombus}
    {{B a A b} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Dart.

```
    \tl_clear:N \l__penrose_tmpa_tl
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {0}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl
    {2*sind(18)*cosd(108)}{2*sind(18)*sind(108)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {2*sind(18)}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl
    {2*sind(18)*cosd(108)}{-2*sind(18)*sind(108)}

    \prop_gput:Nnx \g__penrose_tiles_prop {dart}
    {{c a A C} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Kite.

```
    \tl_clear:N \l__penrose_tmpa_tl
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {0}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(36)}{sind(36)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {1}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(36)}{-sind(36)}

    \prop_gput:Nnx \g__penrose_tiles_prop {kite}
    {{a c C A} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Golden Triangle.

```
    \tl_clear:N \l__penrose_tmpa_tl
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {0}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(18)}{sind(18)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(18)}{-sind(18)}

    \prop_gput:Nnx \g__penrose_tiles_prop {golden~ triangle}
    {{a c b} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Reverse Golden Triangle.

```
    \tl_clear:N \l__penrose_tmpa_tl
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {0}{0}
```

```
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(18)}{sind(18)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(18)}{-sind(18)}

    \prop_gput:Nnx \g__penrose_tiles_prop {reverse~ golden~ triangle}
    {{B C A} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Golden Gnomon

```
    \tl_clear:N \l__penrose_tmpa_tl
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {0}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(36)}{sind(36)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {2*cosd(36)}{0}

    \prop_gput:Nnx \g__penrose_tiles_prop {golden~ gnomon}
    {{C b A} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Reverse Golden Gnomon

```
    \tl_clear:N \l__penrose_tmpa_tl
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {0}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {2*cosd(36)}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(36)}{-sind(36)}
    \prop_gput:Nnx \g__penrose_tiles_prop {reverse~ golden~ gnomon}
    {{a B c} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Primary Pentagon (pentagon 5)

```
    \tl_clear:N \l__penrose_tmpa_tl
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {0}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(108)}{sind(108)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl
    {1+cosd(72)+cosd(144)}{sind(72)+sind(144)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {1+cosd(72)}{sind(72)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {1}{0}
    \prop_gput:Nnx \g__penrose_tiles_prop {pentagon~ 5}
    {{a a a a a} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Secondary Pentagon (pentagon 3)

```
    \prop_gput:Nnx \g__penrose_tiles_prop {pentagon~ 3}
    {{A b a a b} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Tertiary Pentagon (pentagon 2)

```
    \prop_gput:Nnx \g__penrose_tiles_prop {pentagon~ 2}
    {{d A e c A} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Pentagram

```
    \tl_clear:N \l__penrose_tmpa_tl
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {1}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {1-cosd(36)}{-sind(36)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl
```

```
    {1-cosd(36)-cosd(108)}{-sind(36)-sind(108)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(108)}{-sind(108)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl
    {-1+3*cosd(108)+cosd(36)}{-sind(36)-sind(108)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl
    {-1+2*cosd(108)+cosd(36)}{-sind(36)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {-1+2*cosd(108)}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {2*cosd(108)}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(108)}{sind(108)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {0}{0}
    \prop_gput:Nnx \g__penrose_tiles_prop {pentagram}
    {{C E C E C E C E C E} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Boat

```
    \tl_clear:N \l__penrose_tmpa_tl
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {-1+2*cosd(108)}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {2*cosd(108)}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(108)}{sind(108)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {0}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {1}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {1-cosd(36)}{-sind(36)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl
    {-1+2*cosd(108)+cosd(36)}{-sind(36)}
    \prop_gput:Nnx \g__penrose_tiles_prop {boat}
    {{C E C E B D B} {\tl_use:N \l__penrose_tmpa_tl}}
```

- Diamond.

```
    \tl_clear:N \l__penrose_tmpa_tl
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {0}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(18)}{sind(18)}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {2*cosd(18)}{0}
    \__penrose_add_coordinate:Nnn \l__penrose_tmpa_tl {cosd(18)}{-sind(18)}
    \prop_gput:Nnx \g__penrose_tiles_prop {diamond}
    {{D B B D} {\tl_use:N \l__penrose_tmpa_tl}}
```

`\BakePenroseTile` This is the user wrapper around the tile creation macros.

```
\NewDocumentCommand \BakePenroseTile {m}
{
  \prop_get:NnN \g__penrose_tiles_prop {#1} \l__penrose_tmpa_tl
  \__penrose_make_tile:nV {#1} \l__penrose_tmpa_tl
}
```

For backwards compatibility.

```
\cs_set_eq:NN \MakePenroseTile \BakePenroseTile
```

`\UsePenroseTile` This is the command that actually places a tile on the page. The first argument is optional and is for styling.

```
\NewDocumentCommand \UsePenroseTile {O{} m}
{
```

We need to transform the tile to correspond to the current transformation matrix. To ensure that we only transform the current tile, we clone it first.

```
  \tl_set_eq:Nc \l__penrose_tmp_tile_path_tl {g__penrose_tile_#2_tl}
```

The transformation matrix returned by PGF appears to be transposed from what it should be. (This needs a little more investigation, it might be that I've implemented the multiplication incorrectly here.)

```
  \pgfgettransform \l__penrose_tmpa_tl
%  \tl_clear:N \l__penrose_tmpb_tl
%  \tl_set:Nx \l__penrose_tmpb_tl {{\tl_head:N \l__penrose_tmpa_tl}}
%  \tl_set:Nx \l__penrose_tmpa_tl {\tl_tail:N \l__penrose_tmpa_tl}
%  \tl_put_right:Nx \l__penrose_tmpb_tl
%  {
%    {\tl_item:Nn \l__penrose_tmpa_tl {1}}
%  }
%  \tl_put_right:Nx \l__penrose_tmpb_tl
%  {
%    {\tl_item:Nn \l__penrose_tmpa_tl {2}}
%  }
%  \tl_set:Nx \l__penrose_tmpa_tl {\tl_tail:N \l__penrose_tmpa_tl}
%  \tl_set:Nx \l__penrose_tmpa_tl {\tl_tail:N \l__penrose_tmpa_tl}
%  \tl_put_right:NV \l__penrose_tmpb_tl \l__penrose_tmpa_tl
```

Apply the transformation, protocol the path, and render it.

```
  \spath_transform:NV \l__penrose_tmp_tile_path_tl \l__penrose_tmpa_tl
  \spath_tikz_path:nV {#1} \l__penrose_tmp_tile_path_tl
}
```

`save␣Penrose␣path` This is a style for a user to take a path and make it into the path for one of the sides. It needs to store both that side and the reverse.

```
\tikzset{
  save~ Penrose~ path/.code={
    \tikz@addmode{
```

Get the current path.

```
      \pgfsyssoftpath@getcurrentpath\l__penrose_tmpa_tl
```

Normalise the path and save.

```
      \__penrose_normalise_path:N \l__penrose_tmpa_tl
      \tl_gclear_new:c {g__penrose_side_#1_tl}
      \tl_gset_eq:cN {g__penrose_side_#1_tl} \l__penrose_tmpa_tl
```

Now create the reverse path. The name is the upper case version.

```
      \tl_set:Nx \l__penrose_tmpb_tl {\str_uppercase:n {#1}}
```

Reverse the path, and relocate to the interval $[0, 1]$.

```
      \spath_reverse:N \l__penrose_tmpa_tl
      \spath_transform:Nnnnnnn \l__penrose_tmpa_tl {-1} {0} {0} {-1} {1} {0}
      \tl_gclear_new:c {g__penrose_side_ \tl_use:N \l__penrose_tmpb_tl _tl}
```

```
      \tl_gset_eq:cN {g__penrose_side_ \tl_use:N \l__penrose_tmpb_tl _tl} \l__penrose_tmpa_t
    }
  },
  clone~ Penrose~ side~ path/.style~ 2~ args={
    spath/set~ name=Penrose~side,
    spath/clone~ global={#1}{#2}
  },
  spath/prefix/Penrose~side/.style={
    spath/set~ prefix=g__penrose_side_,
  },
  spath/suffix/Penrose~side/.style={
    spath/set~ suffix=_tl,
  },
  clone~ Penrose~ tile~ path/.style~ 2~ args={
    spath/set~ name=Penrose~tile,
    spath/clone~ global={#1}{#2}
  },
  spath/prefix/Penrose~tile/.style={
    spath/set~ prefix=g__penrose_tile_,
  },
  spath/suffix/Penrose~tile/.style={
    spath/set~ suffix=_tl,
  },
  expand~ key/.code={
    \exp_args:NV \pgfkeysalso #1
  }
}
```

(*End definition for* `save Penrose path`*. This function is documented on page* **??**.)

Create the basic tile shapes.

```
\BakePenroseTile {thin~ rhombus}
\BakePenroseTile {thick~ rhombus}
\BakePenroseTile {dart}
\BakePenroseTile {kite}
\BakePenroseTile {golden~ triangle}
\BakePenroseTile {reverse~ golden~ triangle}
\BakePenroseTile {golden~ gnomon}
\BakePenroseTile {reverse~ golden~ gnomon}
\BakePenroseTile {pentagon~ 5}
\BakePenroseTile {pentagon~ 3}
\BakePenroseTile {pentagon~ 2}
\BakePenroseTile {pentagram}
\BakePenroseTile {boat}
\BakePenroseTile {diamond}
```

2.4 User-Definable Tiles

The commands in this section are for enabling a user to define their own tile.

`\__penrose_transform_side:nn` Apply a transformation to make a given side lie on the x-axis. First argument is the tile, second is the side.

```
\cs_new_nopar:Npn \__penrose_transform_side:nn #1#2
{
```

Get our tile data.

```
\prop_get:NnN \g__penrose_tiles_prop {#1} \l__penrose_tmpa_tl
```

Start with the edge list.

Initialise the counter.

```
\int_zero:N \l__penrose_tmpb_int
\int_incr:N \l__penrose_tmpb_int
```

Get the path type list.

```
\tl_set:Nx \l__penrose_tmpc_tl {\tl_head:N \l__penrose_tmpa_tl}
```

Iterate through the path type list, looking for the requested path.

```
\tl_map_inline:Nn \l__penrose_tmpc_tl {
  \str_if_eq:nnT {##1} {#2} {
    \tl_map_break:
  }
  \int_incr:N \l__penrose_tmpb_int
}
```

Get the coordinate list.

```
\tl_set:Nx \l__penrose_tmpc_tl {\tl_tail:N \l__penrose_tmpa_tl}
```

Strip off the outer braces.

```
\tl_set:Nx \l__penrose_tmpc_tl {\tl_item:Nn \l__penrose_tmpc_tl {1}}
```

Add the first coordinate at the end.

```
\tl_put_right:Nx \l__penrose_tmpc_tl
{{\tl_item:Nn \l__penrose_tmpc_tl {1}}}
```

Get the coordinates for this edge.

```
\tl_set:Nx \l__penrose_tmpa_tl
{\tl_item:Nn \l__penrose_tmpc_tl {\int_use:N \l__penrose_tmpb_int}}
\tl_set:Nx \l__penrose_tmpb_tl
{\tl_item:Nn \l__penrose_tmpc_tl {\int_use:N \l__penrose_tmpb_int + 1}}
```

Translate second to origin.

```
\tikz_scan_point:V \l__penrose_tmpb_tl
\dim_set_eq:Nc \l__penrose_xa_dim {pgf@x}
\dim_set_eq:Nc \l__penrose_ya_dim {pgf@y}
```

Rotate and scale first to unit x-vector.

```
\tikz_scan_point:V \l__penrose_tmpa_tl
\dim_set_eq:Nc \l__penrose_xb_dim {pgf@x}
\dim_set_eq:Nc \l__penrose_yb_dim {pgf@y}

\dim_sub:Nn \l__penrose_xb_dim {\l__penrose_xa_dim }
\dim_sub:Nn \l__penrose_yb_dim {\l__penrose_ya_dim }
```

And normalise the vector along it.

```
\pgfpointnormalised{\pgfpoint{\l__penrose_xb_dim}{\l__penrose_yb_dim}}
\dim_set_eq:Nc \l__penrose_xb_dim {pgf@x}
\dim_set_eq:Nc \l__penrose_yb_dim {pgf@y}
```

Now rotate so that the x–axis lies along the edge.

```
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\l__penrose_xb_dim}{-\l__penrose_yb_dim}}
    {\pgfpoint{\l__penrose_yb_dim}{\l__penrose_xb_dim}}

  \pgftransformshift{\pgfpoint{-\l__penrose_xa_dim}{-\l__penrose_ya_dim}}

}
```

(*End definition for* `\__penrose_transform_side:nn`.)

```
\cs_generate_variant:Nn \tl_if_eq:nnT {nVT}
\cs_generate_variant:Nn \__penrose_transform_side:nn {nx,nV}
```

`\TransformAlongSide` Make this available outside the LaTeX3 environment.

```
\DeclareDocumentCommand \TransformAlongSide {m m}
{
  \__penrose_transform_side:nx {#1}{#2}
}
```

(*End definition for* `\TransformAlongSide`.)

`\__penrose_coordinates_at_vertices:n` This places TikZ coordinates at the vertices of the tile.

```
\cs_new_nopar:Npn \__penrose_coordinates_at_vertices:n #1
{
```

Get our tile data

```
  \prop_get:NnN \g__penrose_tiles_prop {#1} \l__penrose_tmpa_tl
```

Start with the edge list

```
  \tl_set:Nx \l__penrose_tmpb_tl {\tl_head:N \l__penrose_tmpa_tl}
```

Get the coordinate list

```
  \tl_set:Nx \l__penrose_tmpc_tl {\tl_tail:N \l__penrose_tmpa_tl}
```

Strip off the outer braces

```
  \tl_set:Nx \l__penrose_tmpc_tl
  {\tl_item:Nn \l__penrose_tmpc_tl {1}}
```

Add the first coordinate at the end

```
  \tl_put_right:Nx \l__penrose_tmpc_tl
  {{\tl_item:Nn \l__penrose_tmpc_tl {1}}}
```

Get the first coordinate

```
  \tl_set:Nx \l__penrose_tmpa_tl {\tl_head:N \l__penrose_tmpc_tl}
  \tl_set:Nx \l__penrose_tmpc_tl {\tl_tail:N \l__penrose_tmpc_tl}
```

Iterate through the path type list, looking for the requested path

```
  \tl_map_inline:Nn \l__penrose_tmpb_tl {
    \tl_set:Nx \l__penrose_tmpd_tl {
      \exp_not:N \coordinate
      (-edge~ ##1~ start)~ at \tl_use:N \l__penrose_tmpa_tl;
    }
    \tl_use:N \l__penrose_tmpd_tl
    \tl_set:Nx \l__penrose_tmpa_tl {\tl_head:N \l__penrose_tmpc_tl}
    \tl_set:Nx \l__penrose_tmpc_tl {\tl_tail:N \l__penrose_tmpc_tl}
    \tl_set:Nx \l__penrose_tmpd_tl {
```

```
      \exp_not:N \coordinate
      (-edge~ ##1~ end)~ at \tl_use:N \l__penrose_tmpa_tl;
    }
    \tl_use:N \l__penrose_tmpd_tl
  }
}
```

(*End definition for* `\__penrose_coordinates_at_vertices:n`.)

`\CoordinatesAtVertices` User-accessible wrapper around the above.

```
\DeclareDocumentCommand \CoordinatesAtVertices {m}
{
  \__penrose_coordinates_at_vertices:n {#1}
}
```

(*End definition for* `\CoordinatesAtVertices`.)

\DefineTile

This is the user function for defining a tile.

```
\DeclareDocumentCommand \DefineTile { m m m }
{
```

Clear the temporary variable.

```
  \tl_clear:N \l__penrose_tmpa_tl
```

The 3rd parameter is a list of coordinates at vertices, iterate through them and add them to the list.

```
  \tl_map_inline:nn {#3} {
    \__penrose_add_coordinate:w \l__penrose_tmpa_tl ##1 \q_stop
  }
```

Now we make a list of the edge types (from the 2nd parameter), using a prop to keep track of whether an edge is repeated.

```
  \prop_clear:N \l__penrose_tmpa_prop
  \tl_map_inline:nn {#2} {
    \prop_if_in:NnTF \l__penrose_tmpa_prop {##1}
    {
      \prop_put:Nnn \l__penrose_tmpa_prop {##1} {1}
    }
    {
      \prop_put:Nnn \l__penrose_tmpa_prop {##1} {0}
    }
  }
```

Having established their multiplicity, we now create the edges with their names, appending numbers to their names if used more than once.

```
  \tl_clear:N \l__penrose_tmpb_tl
  \tl_map_inline:nn {#2}
  {
    \tl_clear:N \l__penrose_tmpc_tl
    \tl_put_right:Nn \l__penrose_tmpc_tl {##1}
    \int_compare:nF {\prop_item:Nn \l__penrose_tmpa_prop {##1} == 0} {
      \tl_put_right:Nx \l__penrose_tmpc_tl
      {\prop_item:Nn \l__penrose_tmpa_prop {##1}}
      \prop_put:Nnx \l__penrose_tmpa_prop {##1}
      {\int_eval:n {\prop_item:Nn \l__penrose_tmpa_prop {##1} + 1}}
    }
    \tl_put_right:Nx \l__penrose_tmpb_tl {{ \l__penrose_tmpc_tl }}
  }
```

Finally, we can create our tile and add it to the global tile prop.

```
  \prop_gput:Nnx \g__penrose_tiles_prop {#1}
  {{\tl_use:N \l__penrose_tmpb_tl} {\tl_use:N \l__penrose_tmpa_tl}}
```

Having created the tile, we make a TikZ pic to place it on the page.

```
  \tikzset{
    #1/.pic={
      \begin{scope}
```

Were we given a tile to align ourselves against?

```
      \pgfkeysgetvalue{/tikz/Penrose/alignment~ location}{\prloc}
      \ifx\prloc\pgfutil@empty
      \else
```

Yes, we were. So we adjust our position accordingly. The first job is to transform so that we're along the edge of the receiving tile.

```
      \begingroup
```

We get the locations of the start and end of the receiving tile. As `pic` sets the node prefix, we have to temporarily suspend that (hence working in a group).

```
        \tikzset{name~ prefix~ ..}%
        \tikz@scan@one@point\pgfutil@firstofone%
      (\prloc-edge~ \pgfkeysvalueof{/tikz/Penrose/alignment~ edge}~ start)%
      \global\pgf@xa=\pgf@x
      \global\pgf@ya=\pgf@y
```

2.5 Lindenmayer System

This is an implementation of the Lindenmayer System description of Penrose tilings as a way of generating tilings from a specific starting seed.

The implementation uses `prop`s to store *rules* and *actions*. The rules are used to expand the starting seed to a certain level, after which the actions are carried out. The syntax is based on the PGF library, but as we're already using LaTeX3 it is reimplemented in that.

These are the rules for generating rhombus tilings.

```
\prop_new:N \g__penrose_rhombus_lms_rule_prop
\prop_gput:Nnn \g__penrose_rhombus_lms_rule_prop {T} {[f*sT][f>g]}
\prop_gput:Nnn \g__penrose_rhombus_lms_rule_prop {t} {[f_st][f>G]}
\prop_gput:Nnn \g__penrose_rhombus_lms_rule_prop {G} {[f+sG][sf>g][sf*sT]}
\prop_gput:Nnn \g__penrose_rhombus_lms_rule_prop {g} {[f-sg][sf>G][sf_st]}
```

These are the rules for generating kite and dart tilings.

```
\prop_new:N \g__penrose_kite_lms_rule_prop
\prop_gput:Nnn \g__penrose_kite_lms_rule_prop {T} {[f*sT][f>st][+sg]}
\prop_gput:Nnn \g__penrose_kite_lms_rule_prop {t} {[f_st][f>sT][-sG]}
\prop_gput:Nnn \g__penrose_kite_lms_rule_prop {G} {[f*+sG][sT]}
\prop_gput:Nnn \g__penrose_kite_lms_rule_prop {g} {[f-_sg][st]}
```

These are the rules for generating pentagon tilings.

```
\prop_new:N \g__penrose_pentagon_lms_rule_prop
\prop_gput:Nnn \g__penrose_pentagon_lms_rule_prop {P}
{[s>P][1sF+Q][1+sF+Q][1*sF+Q][1-sF+Q][1_sF+Q]} % pentagon 5
\prop_gput:Nnn \g__penrose_pentagon_lms_rule_prop {Q}
{[s>P][1+sFR][1*sF*R][1-sF+Q][1_sF+Q][1sF+Q][->fsD]} % pentagon 3
\prop_gput:Nnn \g__penrose_pentagon_lms_rule_prop {R}
{[s>P][1-sF+Q][1+sF*R][1*sFR][1_sF*R][1sFR][_>fsD][>fsD]} % pentagon 2
\prop_gput:Nnn \g__penrose_pentagon_lms_rule_prop {G}
{
  [s>G]
  [se[>d+R][e1B]]
  [+se[>d+R][e1B]]
  [-se[>d+R][e1B]]
  [*se[>d+R][e1B]]
  [_se[>d+R][e1B]]
} % pentagram
\prop_gput:Nnn \g__penrose_pentagon_lms_rule_prop {B}
{
  [s>G]
  [se[>d+R][e1B]]
  [+se[>d+R][e1B]]
  [-se[>d+R][e1B]]
} % boat
\prop_gput:Nnn \g__penrose_pentagon_lms_rule_prop {D}
{[s>d+R][s>eG][se1B]} % diamond
```

Each of the standard tilings can also be drawn using triangles using the same rules.

```
\prop_gset_eq:NN \g__penrose_rtriangle_lms_rule_prop
\g__penrose_rhombus_lms_rule_prop
\prop_gset_eq:NN \g__penrose_ktriangle_lms_rule_prop
\g__penrose_kite_lms_rule_prop
```

These hold the various actions.

```
\prop_new:N \g__penrose_default_lms_action_prop
\prop_new:N \g__penrose_rhombus_lms_action_prop
\prop_new:N \g__penrose_kite_lms_action_prop
\prop_new:N \g__penrose_rtriangle_lms_action_prop
\prop_new:N \g__penrose_ktriangle_lms_action_prop
\prop_new:N \g__penrose_pentagon_lms_action_prop
```

We need some parameters.

```
\dim_new:N \l__penrose_step_dim
\dim_set:Nn \l__penrose_step_dim {1cm}
```

These are the defaults, which will be used in all the rule sets.

```
\prop_gput:Nnn \g__penrose_default_lms_action_prop {[} {\group_begin:}
\prop_gput:Nnn \g__penrose_default_lms_action_prop {]} {\group_end:}
\prop_gput:Nnn \g__penrose_default_lms_action_prop {f}
{\pgftransformxshift{\l__penrose_step_dim}}
\prop_gput:Nnn \g__penrose_default_lms_action_prop {s} {
  \fp_set:Nn \l__penrose_tmpa_fp { 2 * sind(18) * \l__penrose_step_dim }
  \dim_set:Nn \l__penrose_step_dim {\fp_to_dim:N \l__penrose_tmpa_fp}
}
```

The rhombus rules need a variety of turns.

```
\prop_gput:Nnn \g__penrose_rhombus_lms_action_prop {+}
{\pgftransformrotate{144}}
\prop_gput:Nnn \g__penrose_rhombus_lms_action_prop {*}
{\pgftransformrotate{108}}
\prop_gput:Nnn \g__penrose_rhombus_lms_action_prop {-}
{\pgftransformrotate{216}}
\prop_gput:Nnn \g__penrose_rhombus_lms_action_prop {_}
{\pgftransformrotate{252}}
\prop_gput:Nnn \g__penrose_rhombus_lms_action_prop {>}
{\pgftransformrotate{180}}
```

Up to now, the actions for the rhombus and its triangle replacement are the same.

```
\prop_gset_eq:NN \g__penrose_rtriangle_lms_action_prop
\g__penrose_rhombus_lms_action_prop
```

Now we do the actions that actually draw something.

```
\prop_gput:Nnn \g__penrose_rhombus_lms_action_prop {T} {
  \group_begin:
```

As we go through, we keep track of how many tiles we've drawn.

```
  \int_gincr:N \g__penrose_tile_int
```

Set up the position, size, and angle correctly.

```
  \pgftransformrotate{198}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim*2*cosd(18)}
  \pgftransformxshift{-\fp_to_dim:N \l__penrose_tmpa_fp}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
```

Now we draw the thin rhombus, applying every style we can possibly imagine. The `Penrose tile` style gets the current tile and total tile numbers passed to it.

```
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
```

```
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ thin~ rhombus/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{thin~rhombus}
  \group_end:
}
```

Same for the thick rhombus.

```
\prop_gput:Nnn \g__penrose_rhombus_lms_action_prop {G} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)/(2*cosd(36))}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ thick~ rhombus/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{thick~rhombus}
  \group_end:
}
```

Now we do the same for the kite and dart tiling.

```
\prop_gput:Nnn \g__penrose_kite_lms_action_prop {+} {\pgftransformrotate{36}}
\prop_gput:Nnn \g__penrose_kite_lms_action_prop {*} {\pgftransformrotate{108}}
\prop_gput:Nnn \g__penrose_kite_lms_action_prop {-} {\pgftransformrotate{-36}}
\prop_gput:Nnn \g__penrose_kite_lms_action_prop {_}
{\pgftransformrotate{-108}}
\prop_gput:Nnn \g__penrose_kite_lms_action_prop {>} {\pgftransformrotate{180}}

\prop_gset_eq:NN \g__penrose_ktriangle_lms_action_prop
\g__penrose_kite_lms_action_prop

\prop_gput:Nnn \g__penrose_kite_lms_action_prop {T} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \pgftransformrotate{36}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ kite/.try,
```

```
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{kite}
  \group_end:
}
\prop_gput:Nnn \g__penrose_kite_lms_action_prop {g} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \pgftransformrotate{144}
  \pgftransformxshift{-\l__penrose_step_dim * 2 * sin(18)}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ dart/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{dart}
  \group_end:
}
```

Now we set up the actions for the triangle variations.

```
\prop_gput:Nnn \g__penrose_rtriangle_lms_action_prop {T} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \pgftransformrotate{18}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ reverse~ golden~ triangle/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{reverse~ golden~ triangle}
  \group_end:
}
\prop_gput:Nnn \g__penrose_rtriangle_lms_action_prop {t} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \pgftransformrotate{-18}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
```

```
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ golden~ triangle/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{golden~ triangle}
  \group_end:
}
\prop_gput:Nnn \g__penrose_rtriangle_lms_action_prop {G} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \pgftransformrotate{180}
  \pgftransformxshift{-\l__penrose_step_dim}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)/(2*cosd(36))}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ reverse~ golden~ gnomon/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{reverse~ golden~ gnomon}
  \group_end:
}
\prop_gput:Nnn \g__penrose_rtriangle_lms_action_prop {g} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \pgftransformrotate{180}
  \pgftransformxshift{-\l__penrose_step_dim}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)/(2*cosd(36))}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ golden~ gnomon/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
```

```
  ]{golden~ gnomon}
  \group_end:
}
\prop_gput:Nnn \g__penrose_ktriangle_lms_action_prop {T} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \pgftransformrotate{18}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ reverse~ golden~ triangle/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{reverse~ golden~ triangle}
  \group_end:
}
\prop_gput:Nnn \g__penrose_ktriangle_lms_action_prop {t} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \pgftransformrotate{-18}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ golden~ triangle/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{golden~ triangle}
  \group_end:
}
\prop_gput:Nnn \g__penrose_ktriangle_lms_action_prop {G} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \pgftransformrotate{180}
  \pgftransformxshift{-\l__penrose_step_dim}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)/(2*cosd(36))}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
```

```
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ reverse~ golden~ gnomon/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{reverse~ golden~ gnomon}
  \group_end:
}
\prop_gput:Nnn \g__penrose_ktriangle_lms_action_prop {g} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \pgftransformrotate{180}
  \pgftransformxshift{-\l__penrose_step_dim}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)/(2*cosd(36))}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ golden~ gnomon/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{golden~ gnomon}
  \group_end:
}
```

Now we do the same for the pentagonal tilings.
The rules need a variety of turns.

```
\int_new:N \l__penrose_pentagon_parity_int
\seq_new:N \l__penrose_pentagon_parity_seq
\seq_set_from_clist:Nn \l__penrose_pentagon_parity_seq {odd,even}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {1} {
  \int_set:Nn \l__penrose_pentagon_parity_int
  {3 -  \l__penrose_pentagon_parity_int}
  }
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {+}
{\pgftransformrotate{72}}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {*}
{\pgftransformrotate{144}}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {-}
{\pgftransformrotate{288}}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {_}
{\pgftransformrotate{216}}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {>}
{\pgftransformrotate{180}}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {|}
{\pgftransformxscale{-1}}
```

The scale factor is different.

```
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {s} {
  \fp_set:Nn \l__penrose_tmpa_fp
```

```
  {
    1/(2 + 2 * cosd(72) ) * \l__penrose_step_dim
  }
  \dim_set:Nn \l__penrose_step_dim {\fp_to_dim:N \l__penrose_tmpa_fp}
}
```

And we tend to work better vertically.

```
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {f} {
  \fp_set:Nn \l__penrose_tmpa_fp {  tand(54)/2 * \l__penrose_step_dim }
  \pgftransformmyshift{\fp_to_dim:N \l__penrose_tmpa_fp}
}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {F} {
  \fp_set:Nn \l__penrose_tmpa_fp {  tand(54) * \l__penrose_step_dim }
  \pgftransformmyshift{\fp_to_dim:N \l__penrose_tmpa_fp}
}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {d} {
  \fp_set:Nn \l__penrose_tmpa_fp
  {
    (tand(54)/2 - tand(72)/2 + sind(36) ) * \l__penrose_step_dim
  }
  \pgftransformmyshift{\fp_to_dim:N \l__penrose_tmpa_fp}
}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {e} {
  \fp_set:Nn \l__penrose_tmpa_fp
  {
    tand(54) * cosd(36) * \l__penrose_step_dim
  }
  \pgftransformmyshift{\fp_to_dim:N \l__penrose_tmpa_fp}
}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {P} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/2}
  \pgftransformxshift{-\fp_to_dim:N \l__penrose_tmpa_fp}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim*tand(54)/2}
  \pgftransformmyshift{-\fp_to_dim:N \l__penrose_tmpa_fp}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ pentagon/.try,
    every~
    \seq_item:Nn \l__penrose_pentagon_parity_seq
    {\l__penrose_pentagon_parity_int}
    \space pentagon/.try,
    every~ pentagon~ 5/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{pentagon~5}
```

```
  \group_end:
}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {Q} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/2}
  \pgftransformxshift{-\fp_to_dim:N \l__penrose_tmpa_fp}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim*tand(54)/2}
  \pgftransformyshift{-\fp_to_dim:N \l__penrose_tmpa_fp}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ pentagon/.try,
    every~
    \seq_item:Nn \l__penrose_pentagon_parity_seq
    {\l__penrose_pentagon_parity_int}
    \space pentagon/.try,
    every~ pentagon~ 3/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{pentagon~3}
  \group_end:
}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {R} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/2}
  \pgftransformxshift{-\fp_to_dim:N \l__penrose_tmpa_fp}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim*tand(54)/2}
  \pgftransformyshift{-\fp_to_dim:N \l__penrose_tmpa_fp}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ pentagon/.try,
    every~
    \seq_item:Nn \l__penrose_pentagon_parity_seq
    {\l__penrose_pentagon_parity_int}
    \space pentagon/.try,
    every~ pentagon~ 2/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
```

```
  ]{pentagon~2}
  \group_end:
}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {G} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
%  \pgftransformrotate{198}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim*cosd(72)}
  \pgftransformxshift{\fp_to_dim:N \l__penrose_tmpa_fp}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim*tand(54)*cosd(72)}
  \pgftransformyshift{\fp_to_dim:N \l__penrose_tmpa_fp}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ pentagram/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{pentagram}
  \group_end:
}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {B} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
%  \pgftransformrotate{198}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim*cosd(72)}
  \pgftransformxshift{\fp_to_dim:N \l__penrose_tmpa_fp}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim*tand(54)*cosd(72)}
  \pgftransformyshift{\fp_to_dim:N \l__penrose_tmpa_fp}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ boat/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{boat}
  \group_end:
}
\prop_gput:Nnn \g__penrose_pentagon_lms_action_prop {D} {
  \group_begin:
  \int_gincr:N \g__penrose_tile_int
  \pgftransformrotate{90}
```

```
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim*cosd(18)}
  \pgftransformxshift{-\fp_to_dim:N \l__penrose_tmpa_fp}
  \fp_set:Nn \l__penrose_tmpa_fp {\l__penrose_step_dim/(1cm)}
  \pgftransformscale{\fp_use:N \l__penrose_tmpa_fp}
  \tl_set:Nx \l__penrose_tmpc_tl
  {
    {\int_use:N  \g__penrose_tile_int}
    {\int_use:N \g__penrose_tiles_int}
  }
  \UsePenroseTile[
    every~ Penrose~ tile/.try,
    every~ diamond/.try,
    Penrose~ tile~ \int_use:N \g__penrose_tile_int/.try,
    Penrose~ tile/.try/.expand~ once=\l__penrose_tmpc_tl
  ]{diamond}
  \group_end:
}
```

`\__penrose_make_lms:Nnnn` This creates the token list of actions, starting with the seed. The arguments are: a token list to store the result in, the name of the system, the number of iterations, and the initial state.

```
\cs_new_nopar:Npn \__penrose_make_lms:Nnnn #1#2#3#4
{
  \group_begin:
```

On the first time round, we start with the given seed.

```
  \tl_set:Nn \l__penrose_tmpb_tl {#4}
```

We repeat the specified number of times.

```
  \prg_replicate:nn {#3} {
```

Duplicate the current state.

```
    \tl_set_eq:NN \l__penrose_tmpa_tl \l__penrose_tmpb_tl
```

Clear the receiving token list.

```
    \tl_clear:N \l__penrose_tmpb_tl
```

Walk through the current list, appending to the receiving list according to the rules.

```
    \tl_map_inline:Nn \l__penrose_tmpa_tl
    {
```

If a rule exists, copy that.

```
      \prop_if_in:cnTF {g__penrose_#2_lms_rule_prop} {##1}
      {
        \tl_put_right:Nx \l__penrose_tmpb_tl
        {\prop_item:cn {g__penrose_#2_lms_rule_prop} {##1} }
      }
      {
```

Otherwise, just copy the token.

```
        \tl_put_right:Nn \l__penrose_tmpb_tl {##1}
      }
    }
  }
```

We've done all this inside a group, now pass the result outside.

```
  \tl_set:Nn \l__penrose_tmpa_tl {
    \group_end:
    \tl_set:Nn #1
  }
  \tl_put_right:Nx \l__penrose_tmpa_tl {{\tl_use:N \l__penrose_tmpb_tl}}
  \tl_use:N \l__penrose_tmpa_tl
}
\cs_generate_variant:Nn \__penrose_make_lms:Nnnn {Nnnx}
```

(*End definition for* `\__penrose_make_lms:Nnnn`.)

`\__penrose_invoke_lms:Nn` This carries out the actions specified by the resulting rules.

```
\cs_new_nopar:Npn \__penrose_invoke_lms:Nn #1#2
{
  \group_begin:
```

Walk through the given list, carrying out the corresponding action if it exists. If not, look at the default. Otherwise, just do nothing.

```
  \tl_map_inline:Nn #1 {
    \prop_if_in:cnTF {g__penrose_#2_lms_action_prop} {##1}
    {
      \prop_item:cn {g__penrose_#2_lms_action_prop} {##1}
    }
    {
      \prop_if_in:cnT {g__penrose_default_lms_action_prop} {##1}
      {
        \prop_item:cn {g__penrose_default_lms_action_prop} {##1}
      }
    }
  }
  \group_end:
}
```

(*End definition for* `\__penrose_invoke_lms:Nn`.)

We keep track of the number of tiles.

```
\int_new:N \g__penrose_tile_int
\int_new:N \g__penrose_tiles_int
```

`\PenroseDecomposition` This is the user macro to invoke the decomposition. The arguments are: optional styles, the name, number of iterations, and starting seed.

```
\NewDocumentCommand \PenroseDecomposition { O{} m m m }
{
  \group_begin:
  \tikzset{#1}
  \__penrose_make_lms:Nnnx \l__penrose_tmpa_tl {#2} {#3} {#4}
  \__penrose_count_lms:N \l__penrose_tmpa_tl
  \int_gzero:N \g__penrose_tile_int
  \int_set:Nn \l__penrose_pentagon_parity_int {2}
  \__penrose_invoke_lms:Nn \l__penrose_tmpa_tl {#2}
  \group_end:
}
```

__penrose_count_lms:N This counts the number of tiles in the string.

```
\cs_new_nopar:Npn \__penrose_count_lms:N #1
{
  \int_gzero:N \g__penrose_tiles_int
  \tl_map_inline:Nn #1
  {
    \tl_if_eq:nnT {##1} {T}
    {
      \int_gincr:N \g__penrose_tiles_int
    }
    \tl_if_eq:nnT {##1} {t}
    {
      \int_gincr:N \g__penrose_tiles_int
    }
    \tl_if_eq:nnT {##1} {G}
    {
      \int_gincr:N \g__penrose_tiles_int
    }
    \tl_if_eq:nnT {##1} {g}
    {
      \int_gincr:N \g__penrose_tiles_int
    }
  }
}
```

(*End definition for* `\__penrose_count_lms:N`.)

This is a `\tikzset` mechanism for setting the dimensions of the tiling.

```
\tikzset{
  Penrose~ step/.code={
    \dim_set:Nn \l__penrose_step_dim {#1}
  }
}
```

We're done with LaTeX3, so turn off the syntax.

```
\ExplSyntaxOff
```

2.6 TikZ Pictures

New in TikZ3.0 is the ability to make pictures that can be reused. Penrose tiles seems an obvious use for this. These pictures can be placed alongside other tiles, matching by edge type.

There are a variety of constants that are frequently used and reused, so we define them all here. These are the PGF versions.

```
\pgfmathsetmacro\pr@chphi{cos(18)}
\pgfmathsetmacro\pr@shphi{sin(18)}
\pgfmathsetmacro\pr@cphi{cos(36)}
\pgfmathsetmacro\pr@sphi{sin(36)}
\pgfmathsetmacro\pr@invphi{2/(sqrt(5)+1)}
\pgfmathsetmacro\pr@phi{(sqrt(5)+1)/2}
\pgfmathsetmacro\pr@invphisq{\pr@invphi*\pr@invphi}
\pgfmathsetmacro\pr@ominvphisq{\pr@invphi - \pr@invphisq}
\pgfmathsetmacro\pr@ominvphi{1 - \pr@invphi}
```

```
\newif\if@edgealign
\def\pr@getfirst#1#2\pr@stop{#1}%
\def\test@edgealign#1{%
  \pgfkeysgetvalue{/tikz/Penrose/alignment edge}{\@penrose@tmpa}%
  \edef\@penrose@tmpb{#1}%
  \edef\@penrose@tmpa{\expandafter\pr@getfirst\@penrose@tmpa.\pr@stop}%
  \ifx\@penrose@tmpa\@penrose@tmpb
  \@edgealigntrue
  \else
  \@edgealignfalse
  \fi
}%
\newif\if@newedge
\def\test@newedge#1{%
  \pgfkeysgetvalue{/tikz/Penrose/alignment new edge}{\@penrose@tmpa}%
  \edef\@penrose@tmpb{#1}%
  \edef\@penrose@tmpa{\@penrose@tmpa}%
  \ifx\@penrose@tmpa\@penrose@tmpb
  \@newedgetrue
  \else
  \@newedgefalse
  \fi
}%
```

The implementation is essentially the same for each, so only the first will be commented.

```
\tikzset{
```

The key `align with=<tile> along <edge>` is used to set the parameters for placing a tile next to an existing one. For most tiles, that's enough to specify how the new tile should be placed. Some, though, need more information. For those, use `align with=<tile> along <edge> using <number>`.

```
  align with/.code args={#1 along #2}{%
    \pgfutil@in@{ using }{#2}
    \ifpgfutil@in@%
    \tikzset{
      Penrose/alignment location=#1,
      Penrose/alignment set edges=#2,
    }%
    \else
    \tikzset{
      Penrose/alignment location=#1,
      Penrose/alignment edge=#2,
    }%
    \fi
  },
  Penrose/alignment set edges/.code args={#1 using #2}{%
    \tikzset{
      Penrose/alignment edge=#1,
      Penrose/alignment new edge=#2
    },
  },
  Penrose/alignment location/.initial={},
  Penrose/alignment edge/.initial=a,
```

```
    Penrose/alignment new edge/.initial={},
```

Default clipping style.

```
    every Penrose tile clip/.style={clip},
```

This is the code for setting up a `pic`.

```
    thin rhombus/.pic={
      \begin{scope}
```

Were we given a tile to align ourselves against?

```
      \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
      \ifx\prloc\pgfutil@empty
      \else
```

Yes, we were. So we adjust our position accordingly. The first job is to transform so that we're along the edge of the receiving tile.

```
        \begingroup
```

We get the locations of the start and end of the receiving tile. As `pic` sets the node prefix, we have to temporarily suspend that (hence working in a group).

```
        \tikzset{name prefix ..}%
        \tikz@scan@one@point\pgfutil@firstofone%
        (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
        \global\pgf@xa=\pgf@x
        \global\pgf@ya=\pgf@y
        \tikz@scan@one@point\pgfutil@firstofone%
        (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
        \global\pgf@xb=\pgf@x
        \global\pgf@yb=\pgf@y
        \endgroup
```

We store the initial points in `\pgf@xa` and `\pgf@ya` but we want `\pgf@xb` and `\pgf@yb` to be a vector along the edge.

```
        \advance\pgf@xb by -\pgf@xa
        \advance\pgf@yb by -\pgf@ya
```

We shift to the start of the edge.

```
        \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
```

And normalise the vector along it.

```
        \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
        \pgf@xb=\pgf@x
        \pgf@yb=\pgf@y
```

Now rotate so that the x–axis lies along the edge.

```
        \pgftransformtriangle%
        {\pgfpoint{0pt}{0pt}}%
        {\pgfpoint{\pgf@xb}{\pgf@yb}}%
        {\pgfpoint{-\pgf@yb}{\pgf@xb}}
```

The next job is to shift and rotate the current tile so that the correct edge ends up against the receiving tile.

```
        \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}b\relax
        \pgftransformrotate{-18}%
        \pgftransformshift{\pgfpoint{-\pr@chphi cm}{\pr@shphi cm}}%
        \else
        \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}B\relax
        \pgftransformrotate{18}%
```

```
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}a\relax
    \pgftransformrotate{198}%
    \pgftransformshift{\pgfpoint{-2*\pr@chphi cm}{0 cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}A\relax
    \pgftransformrotate{162}%
    \pgftransformshift{\pgfpoint{-\pr@chphi cm}{-\pr@shphi cm}}%
    \fi\fi\fi\fi
    \fi
```

Now that the transformation is finalised, we can render the tile. We clip against the tile path so that the tiles don't "bleed". If we didn't do this, drawing the tile would result in overlaps which can look a bit ugly. On the other hand, tight clipping can lead to "gaps" between the tiles so we make this optional by enclosing it in a style.

```
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every thin rhombus clip/.try
    ]{thin rhombus}
    \UsePenroseTile[
      every Penrose tile/.try,
      every thin rhombus/.try,
      pic actions
    ]{thin rhombus}
```

These draw the arcs that designate the joining rules. We draw full circles so that it doesn't matter what shape the tiles are.

```
    \UsePenroseTile[
      every Penrose arc clip/.try,
      every thin rhombus arc clip/.try
    ]{thin rhombus}
    \path[every circle arc/.try] (18:1) circle[radius=1/4];
    \path[every long arc/.try] (-18:1) circle[radius=1/4];
```

Lastly, we put coordinates at each vertex, labelled by which edge they are.

```
\coordinate (-edge a start) at (0,0);
\coordinate (-edge a end) at (18:1);
\coordinate (-edge A start) at (18:1);
\coordinate (-edge A end) at (2*\pr@chphi,0);
\coordinate (-edge B start) at (2*\pr@chphi,0);
\coordinate (-edge B end) at (-18:1);
\coordinate (-edge b start) at (-18:1);
\coordinate (-edge b end) at (0,0);
    \end{scope}
  },
```

This is a shortcut for installing the `pic` type.

```
  thin rhombus/.style={
    every Penrose pic/.try,
    pic type=thin rhombus,
  },
```

Same again, but for the thick rhombus.

```
  thick rhombus/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
```

```
    \ifx\prloc\pgfutil@empty
    \else
    \begingroup
    \tikzset{name prefix ..}%
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
    \global\pgf@xa=\pgf@x
    \global\pgf@ya=\pgf@y
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
    \global\pgf@xb=\pgf@x
    \global\pgf@yb=\pgf@y
    \endgroup
    \advance\pgf@xb by -\pgf@xa
    \advance\pgf@yb by -\pgf@ya
    \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
    \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
    \pgf@xb=\pgf@x
    \pgf@yb=\pgf@y
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\pgf@xb}{\pgf@yb}}%
    {\pgfpoint{-\pgf@yb}{\pgf@xb}}
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}b\relax
    \pgftransformrotate{144}%
    \pgftransformshift{\pgfpoint{-\pr@cphi cm}{-\pr@sphi cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}B\relax
    \pgftransformrotate{36}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}a\relax
    \pgftransformrotate{-36}%
    \pgftransformshift{\pgfpoint{-\pr@cphi cm}{\pr@sphi cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}A\relax
    \pgftransformrotate{216}%
    \pgftransformshift{\pgfpoint{-2*\pr@cphi cm}{0 cm}}%
    \fi\fi\fi\fi
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every thick rhombus clip/.try
    ]{thick rhombus}
    \UsePenroseTile[
      every Penrose tile/.try,
      every thick rhombus/.try,
      pic actions
    ]{thick rhombus}
    \UsePenroseTile[
      every Penrose arc clip/.try,
      every thick rhombus arc clip/.try
    ]{thick rhombus}
\path[every circle arc/.try] (2*\pr@cphi,0) circle[radius=1/4];
\path[every long arc/.try] (0,0) circle[radius=3/4];
```

```
\coordinate (-edge B start) at (0,0);
\coordinate (-edge B end) at (36:1);
\coordinate (-edge a start) at (36:1);
\coordinate (-edge a end) at (2*\pr@cphi,0);
\coordinate (-edge A start) at (2*\pr@cphi,0);
\coordinate (-edge A end) at (-36:1);
\coordinate (-edge b start) at (-36:1);
\coordinate (-edge b end) at (0,0);
    \end{scope}
  },
  thick rhombus/.style={
    every Penrose pic/.try,
    pic type=thick rhombus,
  },
```

Now the kite.

```
  kite/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
    \ifx\prloc\pgfutil@empty
    \else
    \begingroup
    \tikzset{name prefix ..}%
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
    \global\pgf@xa=\pgf@x
    \global\pgf@ya=\pgf@y
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
    \global\pgf@xb=\pgf@x
    \global\pgf@yb=\pgf@y
    \endgroup
    \advance\pgf@xb by -\pgf@xa
    \advance\pgf@yb by -\pgf@ya
    \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
    \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
    \pgf@xb=\pgf@x
    \pgf@yb=\pgf@y
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\pgf@xb}{\pgf@yb}}%
    {\pgfpoint{-\pgf@yb}{\pgf@xb}}%
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}c\relax
    \pgftransformrotate{-72}%
    \pgftransformshift{\pgfpoint{-\pr@cphi cm}{\pr@sphi cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}C\relax
    \pgftransformrotate{-108}%
    \pgftransformshift{\pgfpoint{-1 cm}{0 cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}a\relax
    \pgftransformrotate{36}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}A\relax
    \pgftransformrotate{144}%
```

```
    \pgftransformshift{\pgfpoint{-\pr@cphi cm}{-\pr@sphi cm}}%
    \fi\fi\fi\fi
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every kite clip/.try
    ]{kite}
    \UsePenroseTile[
      every Penrose tile/.try,
      every kite/.try,
      pic actions
    ]{kite}
    \UsePenroseTile[
      every Penrose arc clip/.try,
      every kite arc clip/.try
    ]{kite}
\path[every circle arc/.try] (0,0) circle[radius=\pr@invphi];
\path[every long arc/.try] (1,0) circle[radius=\pr@invphisq];
\coordinate (-edge a start) at (0,0);
\coordinate (-edge a end) at (36:1);
\coordinate (-edge c start) at (36:1);
\coordinate (-edge c end) at (1,0);
\coordinate (-edge C start) at (1,0);
\coordinate (-edge C end) at (-36:1);
\coordinate (-edge A start) at (-36:1);
\coordinate (-edge A end) at (0,0);
    \end{scope}
  },
```

The dart is next.

```
  dart/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
    \ifx\prloc\pgfutil@empty
    \else
    \begingroup
    \tikzset{name prefix ..}%
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
    \global\pgf@xa=\pgf@x
    \global\pgf@ya=\pgf@y
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
    \global\pgf@xb=\pgf@x
    \global\pgf@yb=\pgf@y
    \endgroup
    \advance\pgf@xb by -\pgf@xa
    \advance\pgf@yb by -\pgf@ya
    \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
    \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
    \pgf@xb=\pgf@x
    \pgf@yb=\pgf@y
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\pgf@xb}{\pgf@yb}}%
```

```
    {\pgfpoint{-\pgf@yb}{\pgf@xb}}%
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}c\relax
    \pgftransformrotate{108}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}C\relax
    \pgftransformrotate{72}%
    \pgftransformshift%
    {\pgfpoint{\pr@invphi*\pr@shphi cm}{-\pr@invphi*\pr@chphi cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}a\relax
    \pgftransformrotate{-36}%
    \pgftransformshift%
    {\pgfpoint{\pr@invphi*\pr@shphi cm}{\pr@invphi*\pr@chphi cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}A\relax
    \pgftransformrotate{216}%
    \pgftransformshift{\pgfpoint{-\pr@invphi cm}{0 cm}}%
    \fi\fi\fi\fi
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every dart clip/.try
    ]{dart}
    \UsePenroseTile[
      every Penrose tile/.try,
      every dart/.try,
      pic actions
    ]{dart}
    \UsePenroseTile[
      every Penrose arc clip/.try,
      every dart arc clip/.try
    ]{dart}
\path[every circle arc/.try] (\pr@invphi,0) circle[radius=\pr@ominvphi];
\path[every long arc/.try] (0,0) circle[radius=\pr@ominvphisq];
\coordinate (-edge c start) at (0,0);
\coordinate (-edge c end) at (108:\pr@invphi);
\coordinate (-edge a start) at (108:\pr@invphi);
\coordinate (-edge a end) at (\pr@invphi,0);
\coordinate (-edge A start) at (\pr@invphi,0);
\coordinate (-edge A end) at (-108:\pr@invphi);
\coordinate (-edge C start) at (-108:\pr@invphi);
\coordinate (-edge C end) at (0,0);
    \end{scope}
  },
  kite/.style={
    every Penrose pic/.try,
    pic type=kite,
  },
  dart/.style={
    every Penrose pic/.try,
    pic type=dart,
  },
```

The golden triangle.

```
  golden triangle/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
    \ifx\prloc\pgfutil@empty
    \else
    \begingroup
    \tikzset{name prefix ..}%
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
    \global\pgf@xa=\pgf@x
    \global\pgf@ya=\pgf@y
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
    \global\pgf@xb=\pgf@x
    \global\pgf@yb=\pgf@y
    \endgroup
    \advance\pgf@xb by -\pgf@xa
    \advance\pgf@yb by -\pgf@ya
    \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
    \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
    \pgf@xb=\pgf@x
    \pgf@yb=\pgf@y
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\pgf@xb}{\pgf@yb}}%
    {\pgfpoint{-\pgf@yb}{\pgf@xb}}%
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}B\relax
    \pgftransformrotate{18}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}C\relax
    \pgftransformrotate{-90}%
    \pgftransformshift{\pgfpoint{-\pr@chphi cm}{\pr@shphi cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}A\relax
    \pgftransformrotate{162}%
    \pgftransformshift{\pgfpoint{-\pr@chphi cm}{-\pr@shphi cm}}%
    \fi\fi\fi
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every golden triangle clip/.try
    ]{golden triangle}
    \UsePenroseTile[
      every Penrose tile/.try,
      every golden triangle/.try,
      pic actions
    ]{golden triangle}
\coordinate (-edge a start) at (0,0);
\coordinate (-edge a end) at (18:1);
\coordinate (-edge c start) at (18:1);
\coordinate (-edge c end) at (-18:1);
\coordinate (-edge b start) at (-18:1);
\coordinate (-edge b end) at (0,0);
    \end{scope}
```

```
  },
  golden triangle/.style={
    every Penrose pic/.try,
    pic type=golden triangle,
  },
```

The reverse golden triangle (is there a better name?).

```
  reverse golden triangle/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
    \ifx\prloc\pgfutil@empty
    \else
    \begingroup
    \tikzset{name prefix ..}%
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
    \global\pgf@xa=\pgf@x
    \global\pgf@ya=\pgf@y
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
    \global\pgf@xb=\pgf@x
    \global\pgf@yb=\pgf@y
    \endgroup
    \advance\pgf@xb by -\pgf@xa
    \advance\pgf@yb by -\pgf@ya
    \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
    \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
    \pgf@xb=\pgf@x
    \pgf@yb=\pgf@y
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\pgf@xb}{\pgf@yb}}%
    {\pgfpoint{-\pgf@yb}{\pgf@xb}}%
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}b\relax
    \pgftransformrotate{162}%
    \pgftransformshift{\pgfpoint{-\pr@chphi cm}{-\pr@shphi cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}c\relax
    \pgftransformrotate{-90}%
    \pgftransformshift{\pgfpoint{-\pr@chphi cm}{\pr@shphi cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}a\relax
    \pgftransformrotate{18}%
    \fi\fi\fi
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every reverse golden triangle clip/.try
    ]{reverse golden triangle}
    \UsePenroseTile[
      every Penrose tile/.try,
      every reverse golden triangle/.try,
      pic actions
    ]{reverse golden triangle}
\coordinate (-edge B start) at (0,0);
```

```
\coordinate (-edge B end) at (18:1);
\coordinate (-edge C start) at (18:1);
\coordinate (-edge C end) at (-18:1);
\coordinate (-edge A start) at (-18:1);
\coordinate (-edge A end) at (0,0);
    \end{scope}
  },
  reverse golden triangle/.style={
    every Penrose pic/.try,
    pic type=reverse golden triangle,
  },
```

The golden gnomon.

```
  golden gnomon/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
    \ifx\prloc\pgfutil@empty
    \else
    \begingroup
    \tikzset{name prefix ..}%
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
    \global\pgf@xa=\pgf@x
    \global\pgf@ya=\pgf@y
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
    \global\pgf@xb=\pgf@x
    \global\pgf@yb=\pgf@y
    \endgroup
    \advance\pgf@xb by -\pgf@xa
    \advance\pgf@yb by -\pgf@ya
    \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
    \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
    \pgf@xb=\pgf@x
    \pgf@yb=\pgf@y
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\pgf@xb}{\pgf@yb}}%
    {\pgfpoint{-\pgf@yb}{\pgf@xb}}%
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}c\relax
    \pgftransformrotate{144}%
    \pgftransformshift{\pgfpoint{-\pr@cphi cm}{-\pr@sphi cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}B\relax
    \pgftransformrotate{-144}%
    \pgftransformshift{\pgfpoint{-2*\pr@cphi cm}{0 cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}a\relax
    \fi\fi\fi
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every golden gnomon clip/.try
    ]{golden gnomon}
    \UsePenroseTile[
```

```
      every Penrose tile/.try,
      every golden gnomon/.try,
      pic actions
    ]{golden gnomon}
\coordinate (-edge C start) at (0,0);
\coordinate (-edge C end) at (36:1);
\coordinate (-edge b start) at (36:1);
\coordinate (-edge b end) at (2*\pr@cphi,0);
\coordinate (-edge A start) at (2*\pr@cphi,0);
\coordinate (-edge A end) at (0,0);
    \end{scope}
  },
  golden gnomon/.style={
    every Penrose pic/.try,
    pic type=golden gnomon,
  },
```

The reverse golden gnomon.

```
  reverse golden gnomon/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
    \ifx\prloc\pgfutil@empty
    \else
    \begingroup
    \tikzset{name prefix ..}%
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
    \global\pgf@xa=\pgf@x
    \global\pgf@ya=\pgf@y
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
    \global\pgf@xb=\pgf@x
    \global\pgf@yb=\pgf@y
    \endgroup
    \advance\pgf@xb by -\pgf@xa
    \advance\pgf@yb by -\pgf@ya
    \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
    \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
    \pgf@xb=\pgf@x
    \pgf@yb=\pgf@y
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\pgf@xb}{\pgf@yb}}%
    {\pgfpoint{-\pgf@yb}{\pgf@xb}}%
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}C\relax
    \pgftransformrotate{36}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}b\relax
    \pgftransformrotate{-36}%
    \pgftransformshift{\pgfpoint{-\pr@cphi cm}{\pr@sphi cm}}%
    \else
    \if\pgfkeysvalueof{/tikz/Penrose/alignment edge}A\relax
    \pgftransformrotate{180}%
    \pgftransformshift{\pgfpoint{-2*\pr@cphi cm}{0 cm}}%
    \fi\fi\fi
```

```
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every reverse golden gnomon clip/.try
    ]{reverse golden gnomon}
    \UsePenroseTile[
      every Penrose tile/.try,
      every reverse golden gnomon/.try,
      pic actions
    ]{reverse golden gnomon}
\coordinate (-edge a start) at (0,0);
\coordinate (-edge a end) at (2*\pr@cphi,0);
\coordinate (-edge B start) at (2*\pr@cphi,0);
\coordinate (-edge B end) at (-36:1);
\coordinate (-edge c start) at (-36:1);
\coordinate (-edge c end) at (0,0);
    \end{scope}
  },
  reverse golden gnomon/.style={
    every Penrose pic/.try,
    pic type=reverse golden gnomon,
  },
```

The primary pentagon.

```
  pentagon 5/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
    \ifx\prloc\pgfutil@empty
    \else
    \begingroup
    \tikzset{name prefix ..}%
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
    \global\pgf@xa=\pgf@x
    \global\pgf@ya=\pgf@y
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
    \global\pgf@xb=\pgf@x
    \global\pgf@yb=\pgf@y
    \endgroup
    \advance\pgf@xb by -\pgf@xa
    \advance\pgf@yb by -\pgf@ya
    \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
    \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
    \pgf@xb=\pgf@x
    \pgf@yb=\pgf@y
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\pgf@xb}{\pgf@yb}}%
    {\pgfpoint{-\pgf@yb}{\pgf@xb}}%
    \test@newedge{1}%
    \if@newedge
    \pgftransformrotate{180}%
    \pgftransformshift{\pgfpoint{-1 cm}{0 cm}}%
    \else
```

```
    \test@newedge{2}%
    \if@newedge
    \pgftransformshift{\pgfpoint{1 cm}{0 cm}}%
    \pgftransformrotate{108}%
    \pgftransformshift{\pgfpoint{-1 cm}{0 cm}}%
    \else
    \test@newedge{3}%
    \if@newedge
    \pgftransformshift%
    {\pgfpoint{\pr@phi * \pr@shphi cm}{- \pr@phi * \pr@chphi cm}}%
    \pgftransformrotate{36}%
    \else
    \test@newedge{4}%
    \if@newedge
    \pgftransformshift{\pgfpoint{- \pr@shphi cm}{-\pr@chphi cm}}%
    \pgftransformrotate{-36}%
    \else
    \test@newedge{5}%
    \if@newedge
    \pgftransformrotate{-108}%
    \fi\fi\fi\fi\fi
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every pentagon clip/.try,
      every pentagon 5 clip/.try
    ]{pentagon 5}
    \UsePenroseTile[
      every Penrose tile/.try,
      every pentagon/.try,
      every pentagon 5/.try,
      pic actions
    ]{pentagon 5}
\coordinate (-edge a1 start) at (0,0);
\coordinate (-edge a1 end) at (1,0);
\coordinate (-edge a2 start) at (1,0);
\coordinate (-edge a2 end) at (1+\pr@shphi,\pr@chphi);
\coordinate (-edge a3 start) at (1+\pr@shphi,\pr@chphi);
\coordinate (-edge a3 end) at (\pr@cphi-\pr@shphi,\pr@sphi+\pr@chphi);
\coordinate (-edge a4 start) at (\pr@cphi-\pr@shphi,\pr@sphi+\pr@chphi);
\coordinate (-edge a4 end) at (-\pr@shphi,\pr@chphi);
\coordinate (-edge a5 start) at (-\pr@shphi,\pr@chphi);
\coordinate (-edge a5 end) at (0,0);
    \end{scope}
  },
  pentagon 5/.style={
    every Penrose pic/.try,
    pic type=pentagon 5,
  },
```

The secondary pentagon.

```
  pentagon 3/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
    \ifx\prloc\pgfutil@empty
```

```
    \else
    \begingroup
    \tikzset{name prefix ..}%
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
    \global\pgf@xa=\pgf@x
    \global\pgf@ya=\pgf@y
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
    \global\pgf@xb=\pgf@x
    \global\pgf@yb=\pgf@y
    \endgroup
    \advance\pgf@xb by -\pgf@xa
    \advance\pgf@yb by -\pgf@ya
    \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
    \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
    \pgf@xb=\pgf@x
    \pgf@yb=\pgf@y
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\pgf@xb}{\pgf@yb}}%
    {\pgfpoint{-\pgf@yb}{\pgf@xb}}%
    \test@edgealign{a}%
    \if@edgealign
    \pgftransformrotate{180}%
    \pgftransformshift{\pgfpoint{-1 cm}{0 cm}}%
    \else
    \test@edgealign{B}%
    \if@edgealign%
    \test@newedge{1}%
    \if@newedge
    \pgftransformshift{\pgfpoint{1 cm}{0 cm}}%
    \pgftransformrotate{108}%
    \pgftransformshift{\pgfpoint{-1 cm}{0 cm}}%
    \else
    \pgftransformrotate{-108}%
    \fi
    \else
    \test@newedge{1}%
    \if@newedge
    \pgftransformshift%
    {\pgfpoint{\pr@phi * \pr@shphi cm}{- \pr@phi * \pr@chphi cm}}%
    \pgftransformrotate{36}%
    \else
    \pgftransformshift{\pgfpoint{- \pr@shphi cm}{-\pr@chphi cm}}%
    \pgftransformrotate{-36}%
    \fi\fi\fi
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every pentagon clip/.try,
      every pentagon 3 clip/.try
    ]{pentagon 3}
    \UsePenroseTile[
```

```
      every Penrose tile/.try,
      every pentagon/.try,
      every pentagon 3/.try,
      pic actions
    ]{pentagon 3}
\coordinate (-edge A start) at (0,0);
\coordinate (-edge A end) at (1,0);
\coordinate (-edge b1 start) at (1,0);
\coordinate (-edge b1 end) at (1+\pr@shphi,\pr@chphi);
\coordinate (-edge a1 start) at (1+\pr@shphi,\pr@chphi);
\coordinate (-edge a1 end) at (\pr@cphi-\pr@shphi,\pr@sphi+\pr@chphi);
\coordinate (-edge a2 start) at (\pr@cphi-\pr@shphi,\pr@sphi+\pr@chphi);
\coordinate (-edge a2 end) at (-\pr@shphi,\pr@chphi);
\coordinate (-edge b2 start) at (-\pr@shphi,\pr@chphi);
\coordinate (-edge b2 end) at (0,0);
    \end{scope}
  },
  pentagon 3/.style={
    every Penrose pic/.try,
    pic type=pentagon 3,
  },
```

The tertiary pentagon.

```
  pentagon 2/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
    \ifx\prloc\pgfutil@empty
    \else
    \begingroup
    \tikzset{name prefix ..}%
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
    \global\pgf@xa=\pgf@x
    \global\pgf@ya=\pgf@y
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
    \global\pgf@xb=\pgf@x
    \global\pgf@yb=\pgf@y
    \endgroup
    \advance\pgf@xb by -\pgf@xa
    \advance\pgf@yb by -\pgf@ya
    \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
    \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
    \pgf@xb=\pgf@x
    \pgf@yb=\pgf@y
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\pgf@xb}{\pgf@yb}}%
    {\pgfpoint{-\pgf@yb}{\pgf@xb}}%
    \test@edgealign{D}%
    \if@edgealign
    \pgftransformrotate{180}%
    \pgftransformshift{\pgfpoint{-1 cm}{0 cm}}%
    \else
    \test@edgealign{a}%
```

```
    \if@edgealign%
    \test@newedge{1}%
    \if@newedge
    \pgftransformshift{\pgfpoint{1 cm}{0 cm}}%
    \pgftransformrotate{108}%
    \pgftransformshift{\pgfpoint{-1 cm}{0 cm}}%
    \else
    \pgftransformrotate{-108}%
    \fi
    \else
    \test@newedge{1}%
    \if@newedge
    \pgftransformshift%
    {\pgfpoint{\pr@phi * \pr@shphi cm}{- \pr@phi * \pr@chphi cm}}%
    \pgftransformrotate{36}%
    \else
    \pgftransformshift{\pgfpoint{- \pr@shphi cm}{-\pr@chphi cm}}%
    \pgftransformrotate{-36}%
    \fi\fi\fi
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every pentagon clip/.try,
      every pentagon 2 clip/.try
    ]{pentagon 2}
    \UsePenroseTile[
      every Penrose tile/.try,
      every pentagon/.try,
      every pentagon 2/.try,
      pic actions
    ]{pentagon 2}
\coordinate (-edge d start) at (0,0);
\coordinate (-edge d end) at (1,0);
\coordinate (-edge A1 start) at (1,0);
\coordinate (-edge A1 end) at (1+\pr@shphi,\pr@chphi);
\coordinate (-edge c1 start) at (1+\pr@shphi,\pr@chphi);
\coordinate (-edge c1 end) at (\pr@cphi-\pr@shphi,\pr@sphi+\pr@chphi);
\coordinate (-edge c2 start) at (\pr@cphi-\pr@shphi,\pr@sphi+\pr@chphi);
\coordinate (-edge c2 end) at (-\pr@shphi,\pr@chphi);
\coordinate (-edge A2 start) at (-\pr@shphi,\pr@chphi);
\coordinate (-edge A2 end) at (0,0);
    \end{scope}
  },
  pentagon 2/.style={
    every Penrose pic/.try,
    pic type=pentagon 2,
  },
```

The pentagram.

```
  pentagram/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
    \ifx\prloc\pgfutil@empty
    \else
    \begingroup
```

```
\tikzset{name prefix ..}%
\tikz@scan@one@point\pgfutil@firstofone%
(\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
\global\pgf@xa=\pgf@x
\global\pgf@ya=\pgf@y
\tikz@scan@one@point\pgfutil@firstofone%
(\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
\global\pgf@xb=\pgf@x
\global\pgf@yb=\pgf@y
\endgroup
\advance\pgf@xb by -\pgf@xa
\advance\pgf@yb by -\pgf@ya
\pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
\pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
\pgf@xb=\pgf@x
\pgf@yb=\pgf@y
\pgftransformtriangle%
{\pgfpoint{0pt}{0pt}}%
{\pgfpoint{\pgf@xb}{\pgf@yb}}%
{\pgfpoint{-\pgf@yb}{\pgf@xb}}%
\test@newedge{2}%
\if@newedge
\pgftransformshift{\pgfpoint{1 cm}{0 cm}}%
\pgftransformrotate{72}%
\else
\test@newedge{3}%
\if@newedge
\pgftransformrotate{-72}%
\pgftransformshift{\pgfpoint{2 * \pr@shphi cm}{0 cm}}%
\else
\test@newedge{4}%
\if@newedge
\pgftransformshift{\pgfpoint{1 cm + 2 * \pr@shphi cm}{0 cm}}%
\else
\test@newedge{5}%
\if@newedge
\pgftransformrotate{216}%
\pgftransformshift{\pgfpoint{\pr@cphi cm}{\pr@sphi cm}}%
\else
\test@newedge{6}%
\if@newedge
\pgftransformshift{\pgfpoint{1cm}{0cm}}%
\pgftransformrotate{-72}%
\pgftransformshift{\pgfpoint{\pr@cphi cm}{\pr@sphi cm}}%
\else
\test@newedge{7}%
\if@newedge
\pgftransformrotate{144}%
\pgftransformshift{\pgfpoint{\pr@shphi cm}{\pr@chphi cm}}%
\else
\test@newedge{8}%
\if@newedge
\pgftransformshift{\pgfpoint{1 cm + 2*\pr@shphi cm}{0cm}}%
\pgftransformrotate{216}%
```

```
    \pgftransformshift{\pgfpoint{\pr@cphi cm}{\pr@sphi cm}}%
    \else
    \test@newedge{9}%
    \if@newedge
    \pgftransformshift{\pgfpoint{-2*\pr@shphi cm}{0cm}}%
    \pgftransformrotate{72}%
    \else
    \test@newedge{10}%
    \if@newedge
    \pgftransformrotate{144}%
    \pgftransformshift{\pgfpoint{-1 cm}{0 cm}}%
    \else
    \fi\fi\fi\fi\fi\fi\fi\fi\fi
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every pentagram clip/.try
    ]{pentagram}
    \UsePenroseTile[
      every Penrose tile/.try,
      every pentagram/.try,
      pic actions
    ]{pentagram}
\coordinate (-edge C1 start) at (1,0);
\coordinate (-edge C1 end) at (0,0);
\coordinate (-edge C2 start) at (0,0);
\coordinate (-edge C2 end) at (-\pr@shphi,\pr@chphi);
\coordinate (-edge C3 start) at (-\pr@shphi,\pr@chphi);
\coordinate (-edge C3 end) at (-2*\pr@shphi,0);
\coordinate (-edge C4 start) at (-2*\pr@shphi,0);
\coordinate (-edge C4 end) at (-1-2*\pr@shphi,0);
\coordinate (-edge C5 start) at (-1-2*\pr@shphi,0);
\coordinate (-edge C5 end) at (-\pr@cphi,-\pr@sphi);
\coordinate (-edge C6 start) at (-\pr@cphi,-\pr@sphi);
\coordinate (-edge C6 end) at (-\pr@cphi-\pr@shphi,-\pr@sphi-\pr@chphi);
\coordinate (-edge C7 start) at (-\pr@cphi-\pr@shphi,-\pr@sphi-\pr@chphi);
\coordinate (-edge C7 end) at (-\pr@shphi,-\pr@chphi);
\coordinate (-edge C8 start) at (-\pr@shphi,-\pr@chphi);
\coordinate (-edge C8 end) at (1-\pr@cphi+\pr@shphi,-\pr@sphi-\pr@chphi);
\coordinate (-edge C9 start) at (1-\pr@cphi+\pr@shphi,-\pr@sphi-\pr@chphi);
\coordinate (-edge C9 end) at (1-\pr@cphi,-\pr@sphi);
\coordinate (-edge C10 start) at (1-\pr@cphi,-\pr@sphi);
\coordinate (-edge C10 end) at (1,0);
    \end{scope}
  },
  pentagram/.style={
    every Penrose pic/.try,
    pic type=pentagram,
  },
```

The boat.

```
  boat/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
    \ifx\prloc\pgfutil@empty
```

```
    \else
    \begingroup
    \tikzset{name prefix ..}%
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
    \global\pgf@xa=\pgf@x
    \global\pgf@ya=\pgf@y
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
    \global\pgf@xb=\pgf@x
    \global\pgf@yb=\pgf@y
    \endgroup
    \advance\pgf@xb by -\pgf@xa
    \advance\pgf@yb by -\pgf@ya
    \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
    \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
    \pgf@xb=\pgf@x
    \pgf@yb=\pgf@y
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\pgf@xb}{\pgf@yb}}%
    {\pgfpoint{-\pgf@yb}{\pgf@xb}}%
    \test@edgealign{d}%
    \if@edgealign
    \pgftransformrotate{180}%
    \pgftransformshift{\pgfpoint{\pr@cphi cm - 1cm}{\pr@sphi cm}}%
    \else
    \test@edgealign{b}%
    \if@edgealign%
    \test@newedge{2}%
    \if@newedge
    \pgftransformrotate{144}%
    \pgftransformshift{\pgfpoint{-1 cm}{0 cm}}%
    \else
    \pgftransformrotate{216}%
    \pgftransformshift{\pgfpoint{\pr@cphi cm}{\pr@sphi cm}}%
    \fi
    \else
    \test@edgealign{c}%
    \if@edgealign%
    \test@newedge{2}%
    \if@newedge
    \pgftransformshift{\pgfpoint{1 cm}{0 cm}}%
    \pgftransformrotate{72}%
    \else
    \test@newedge{3}%
    \if@newedge
    \pgftransformrotate{-72}%
    \pgftransformshift{\pgfpoint{2 * \pr@shphi cm}{0 cm}}%
    \else
    \test@newedge{4}%
    \if@newedge
    \pgftransformshift{\pgfpoint{1 cm + 2 * \pr@shphi cm}{0 cm}}%
    \else
```

```
    \fi\fi\fi\fi\fi\fi
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every boat clip/.try
    ]{boat}
    \UsePenroseTile[
      every Penrose tile/.try,
      every boat/.try,
      pic actions
    ]{boat}
\coordinate (-edge C1 start) at (1,0);
\coordinate (-edge C1 end) at (0,0);
\coordinate (-edge C2 start) at (0,0);
\coordinate (-edge C2 end) at (-\pr@shphi,\pr@chphi);
\coordinate (-edge C3 start) at (-\pr@shphi,\pr@chphi);
\coordinate (-edge C3 end) at (-2*\pr@shphi,0);
\coordinate (-edge C4 start) at (-2*\pr@shphi,0);
\coordinate (-edge C4 end) at (-1-2*\pr@shphi,0);
\coordinate (-edge B1 start) at (-1-2*\pr@shphi,0);
\coordinate (-edge B1 end) at (-\pr@cphi,-\pr@sphi);
\coordinate (-edge D start) at (-\pr@cphi,-\pr@sphi);
\coordinate (-edge D end) at (1-\pr@cphi,-\pr@sphi);
\coordinate (-edge B2 start) at (1-\pr@cphi,-\pr@sphi);
\coordinate (-edge B2 end) at (1,0);
    \end{scope}
  },
  boat/.style={
    every Penrose pic/.try,
    pic type=boat,
  },
```

The diamond.

```
  diamond/.pic={
    \begin{scope}
    \pgfkeysgetvalue{/tikz/Penrose/alignment location}{\prloc}
    \ifx\prloc\pgfutil@empty
    \else
    \begingroup
    \tikzset{name prefix ..}%
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} start)%
    \global\pgf@xa=\pgf@x
    \global\pgf@ya=\pgf@y
    \tikz@scan@one@point\pgfutil@firstofone%
    (\prloc-edge \pgfkeysvalueof{/tikz/Penrose/alignment edge} end)%
    \global\pgf@xb=\pgf@x
    \global\pgf@yb=\pgf@y
    \endgroup
    \advance\pgf@xb by -\pgf@xa
    \advance\pgf@yb by -\pgf@ya
    \pgftransformshift{\pgfpoint{\pgf@xa}{\pgf@ya}}%
    \pgfpointnormalised{\pgfpoint{\pgf@xb}{\pgf@yb}}
    \pgf@xb=\pgf@x
    \pgf@yb=\pgf@y
```

```
    \pgftransformtriangle%
    {\pgfpoint{0pt}{0pt}}%
    {\pgfpoint{\pgf@xb}{\pgf@yb}}%
    {\pgfpoint{-\pgf@yb}{\pgf@xb}}%
    \test@edgealign{d}%
    \if@edgealign%
    \test@newedge{2}%
    \if@newedge
    \pgftransformshift{\pgfpoint{1cm}{0cm}}%
    \pgftransformrotate{-162}%
    \else
    \pgftransformrotate{-18}%
    \fi
    \else
    \test@newedge{2}%
    \if@newedge
    \pgftransformrotate{162}%
    \pgftransformshift{\pgfpoint{- 2 * \pr@chphi cm}{0cm}}%
    \else
    \pgftransformshift{\pgfpoint{- \pr@cphi cm}{-\pr@sphi cm}}%
    \pgftransformrotate{18}%
    \fi\fi
    \fi
    \UsePenroseTile[
      every Penrose tile clip/.try,
      every diamond clip/.try
    ]{diamond}
    \UsePenroseTile[
      every Penrose tile/.try,
      every diamond/.try,
      pic actions
    ]{diamond}
\coordinate (-edge D1 start) at (0,0);
\coordinate (-edge D1 end) at (\pr@chphi,\pr@shphi);
\coordinate (-edge B1 start) at (\pr@chphi,\pr@shphi);
\coordinate (-edge B1 end) at (2*\pr@chphi,0);
\coordinate (-edge B2 start) at (2*\pr@chphi,0);
\coordinate (-edge B2 end) at (\pr@chphi,-\pr@shphi);
\coordinate (-edge D2 start) at (\pr@chphi,-\pr@shphi);
\coordinate (-edge D2 end) at (0,0);
    \end{scope}
  },
  diamond/.style={
    every Penrose pic/.try,
    pic type=diamond,
  },
}
```

⟨/package⟩