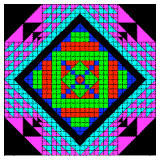
More PictureWorld

Divide, conquer, and glue

Friday, Sep 22, 2006

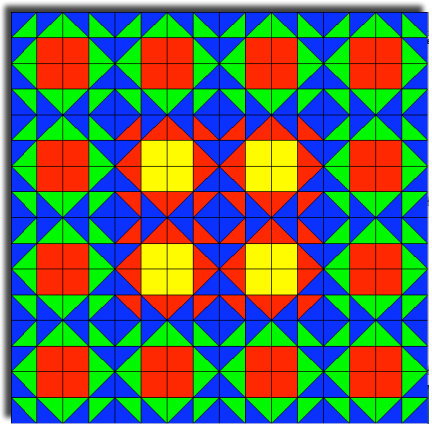


CS111 Computer Programming

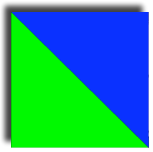
Department of Computer Science
Wellesley College

A Quilt Problem

How do we build this complex quilt …



… from simple primitive parts?

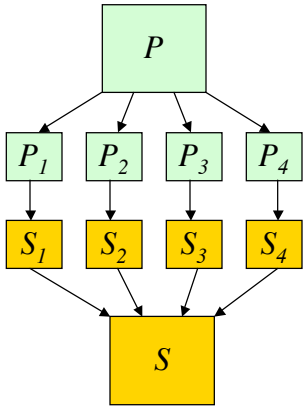


triangles(Color.green, Color.blue)

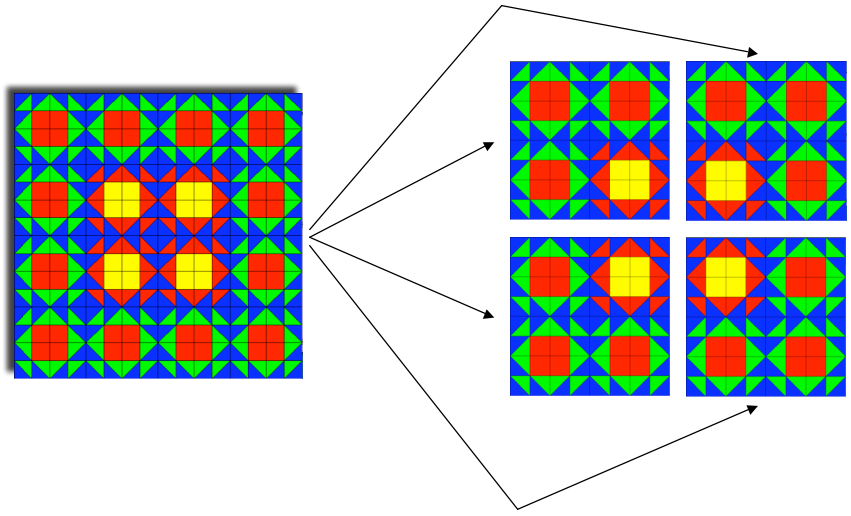
patch(Color.red)

Recall big idea #3: Divide, conquer & glue

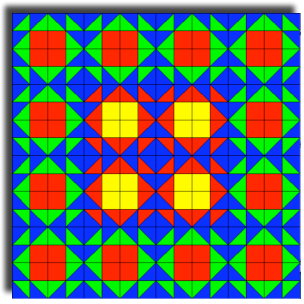
Divide
problem P into subproblems.
Conquer
each of the subproblems, &
Glue (combine)
the solutions to the subproblems into a solution S for P.

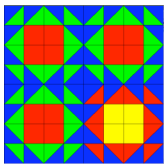


Divide the Quilt in Subproblems

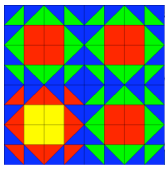


Conquer the Subproblems via "Wishful Thinking"

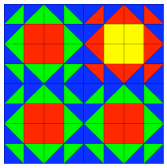




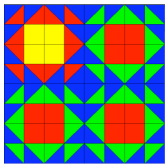
clockwise270(quadrant())



quadrant()

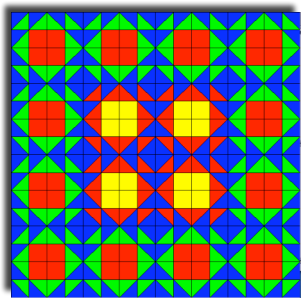


clockwise180(quadrant())



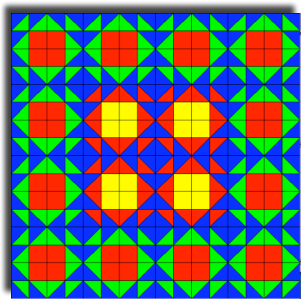
clockwise90(quadrant())

Glue the Solutions to Solve the Problem



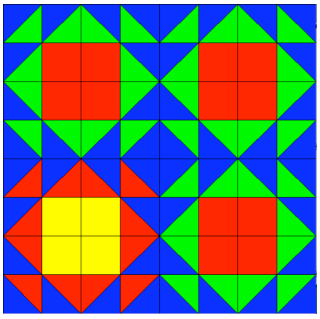
```
public Picture quilt () {
  return fourPics(clockwise270(quadrant()),
                  quadrant,
                  clockwise180(quadrant()),
                  clockwise90(quadrant())
                  );
}
```

Abstracting Over the Glue



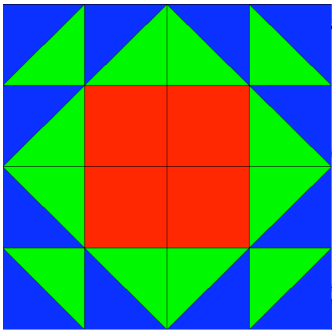
```
public Picture quilt() {
  return rotations (quadrant());
}

public Picture rotations (Picture p) {
  return fourPics(clockwise270(p), p,
                  clockwise180(p), clockwise90(p));
}
```


Now Flesh out quadrant()



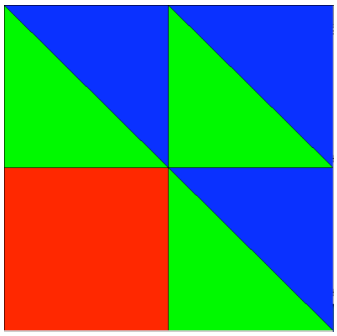
quadrant()

Continue the Descent …



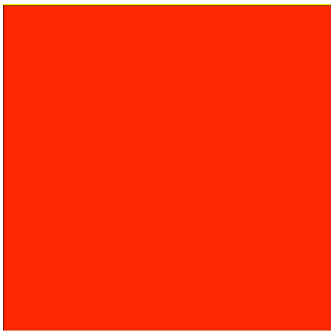
star(Color.red, Color.green, Color.blue)

And Descend Some More …

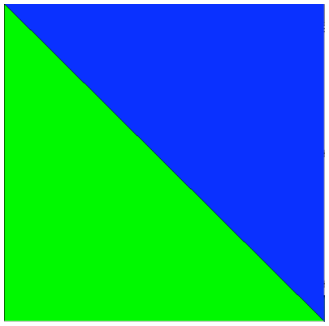


starQuadrant(Color.red, Color.green, Color.blue)

Until we Reach Primitives



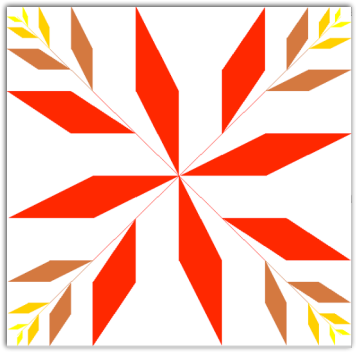
patch(Color.red)



triangles(Color.green, Color.blue)

A New Problem

How do we build this picture …



autumnLeaves()

… from this primitive?

twoLeaves(Color.orange)

Branching Out

branch4()

branch3()

branch2()

So a panda walks into a bar . . .

The "Real" PictureWorld Primitives

```
// Constructor method for Line sublcass of Picture.
// Creates a Picture that is a line in the unit square
// from point (x1, y1) to (x2, y2) with color c.
public Line (double x1, double y1, double x2, double y2, Color c);

// Constructor method for Rect subclass of Picture.
// Creates a Picture that is a rectangle in the unit square
// spanning the corners (x1, y1) and (x2, y2).  If isFilled is true,
// the rectangle is filled with color c; otherwise only the border of
// the rectangle is color c.
public Rect (double llx, double lly, double urx, double ury,
             Color c, boolean isFilled);

// Constructor method for Poly subclass of Picture.
// Creates an "empty" polygon Picture in the unit square, where
// c controls the color and isFilled controls filledness.
// Constructor method for Poly subclass of Picture
public Rect (Color c, boolean isFilled);

// Adds a new point to the given polygon.
// Instance method for Poly subclass of Picture
public void addPoint (double x, double y);
```


Patches and Triangles

```
public Picture patch (Color c) {
  return overlay (new Rect(0.0, 0.0, 1.0, 1.0, Color.black, false),
                  new Rect(0.0, 0.0, 1.0, 1.0, c, true));
}

public Picture triangles (Color c1, Color c2) {
  return overlay (new Rect(0.0, 0.0, 1.0, 1.0, Color.black, false),
                  overlay (new Line(0.0, 1.0, 1.0, 0.0, Color.black),
                           overlay(tri(c1, true),
                                   clockwise180(tri(c2, true)))));
 }

 public Picture tri (Color c, boolean isFilled) {
   Poly triPoly = new Poly (c, isFilled);
   triPoly.addPoint(0.0, 0.0);
   triPoly.addPoint(0.0, 1.0);
   triPoly.addPoint(1.0, 0.0);
   return triPoly;
 }
```