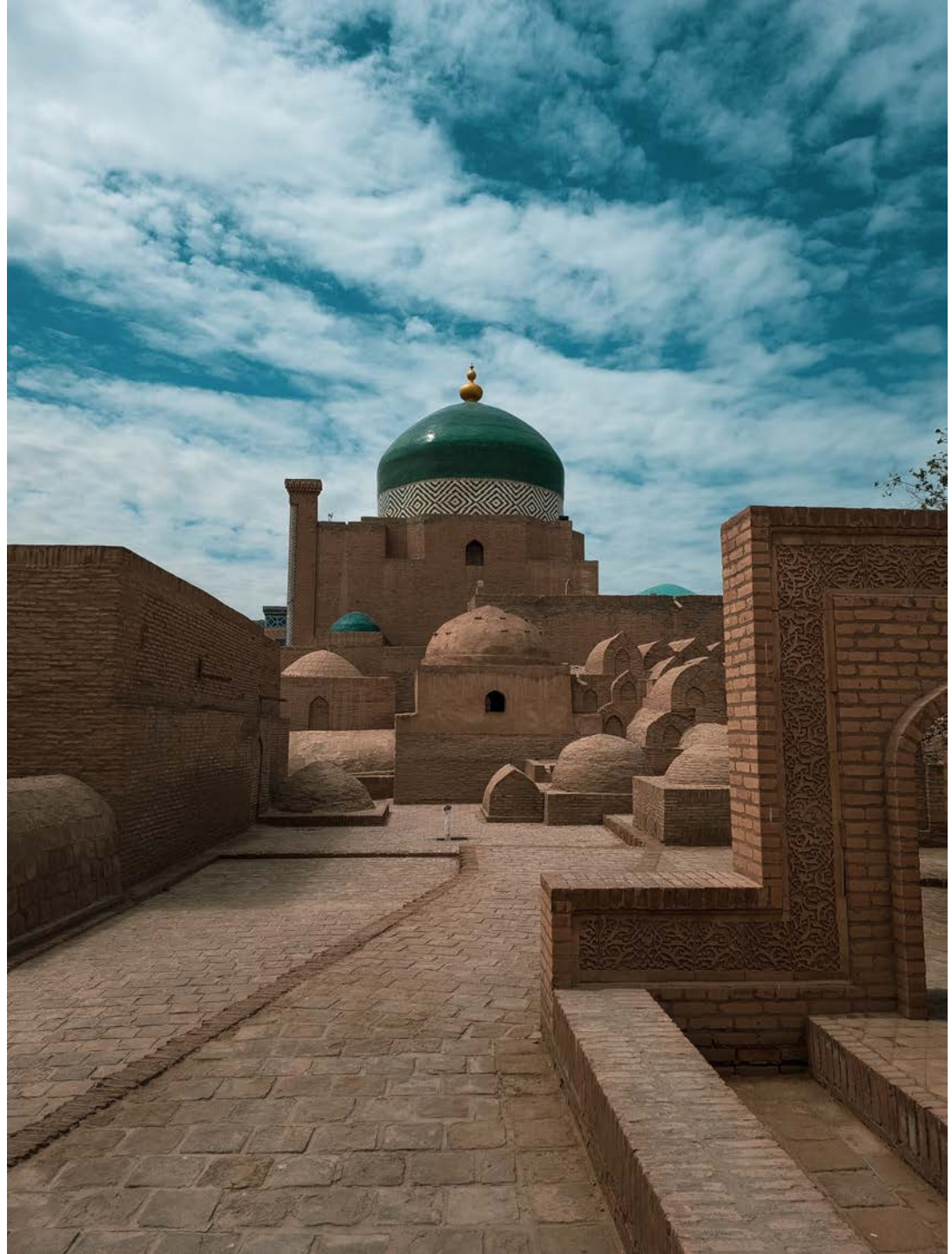
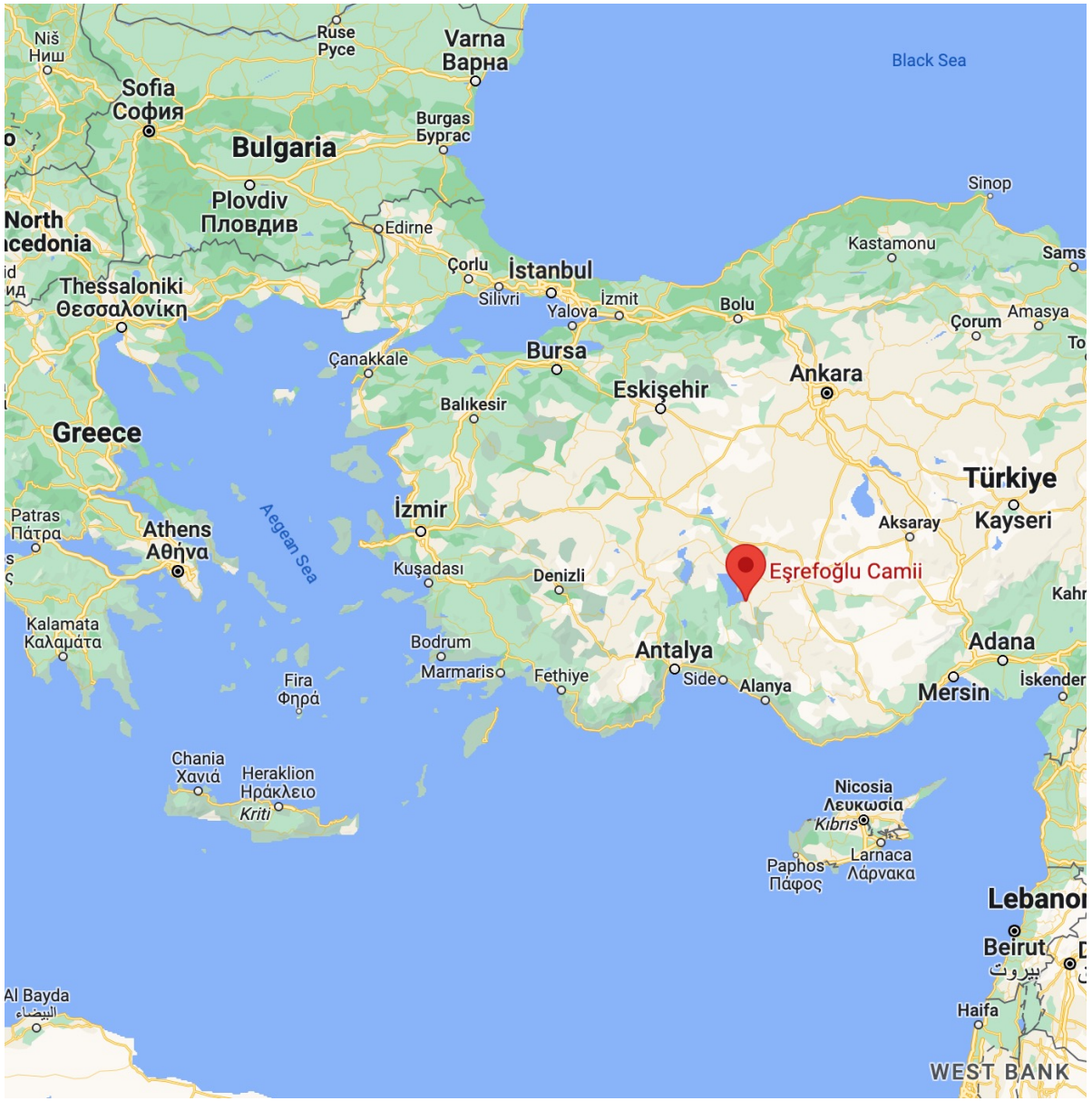
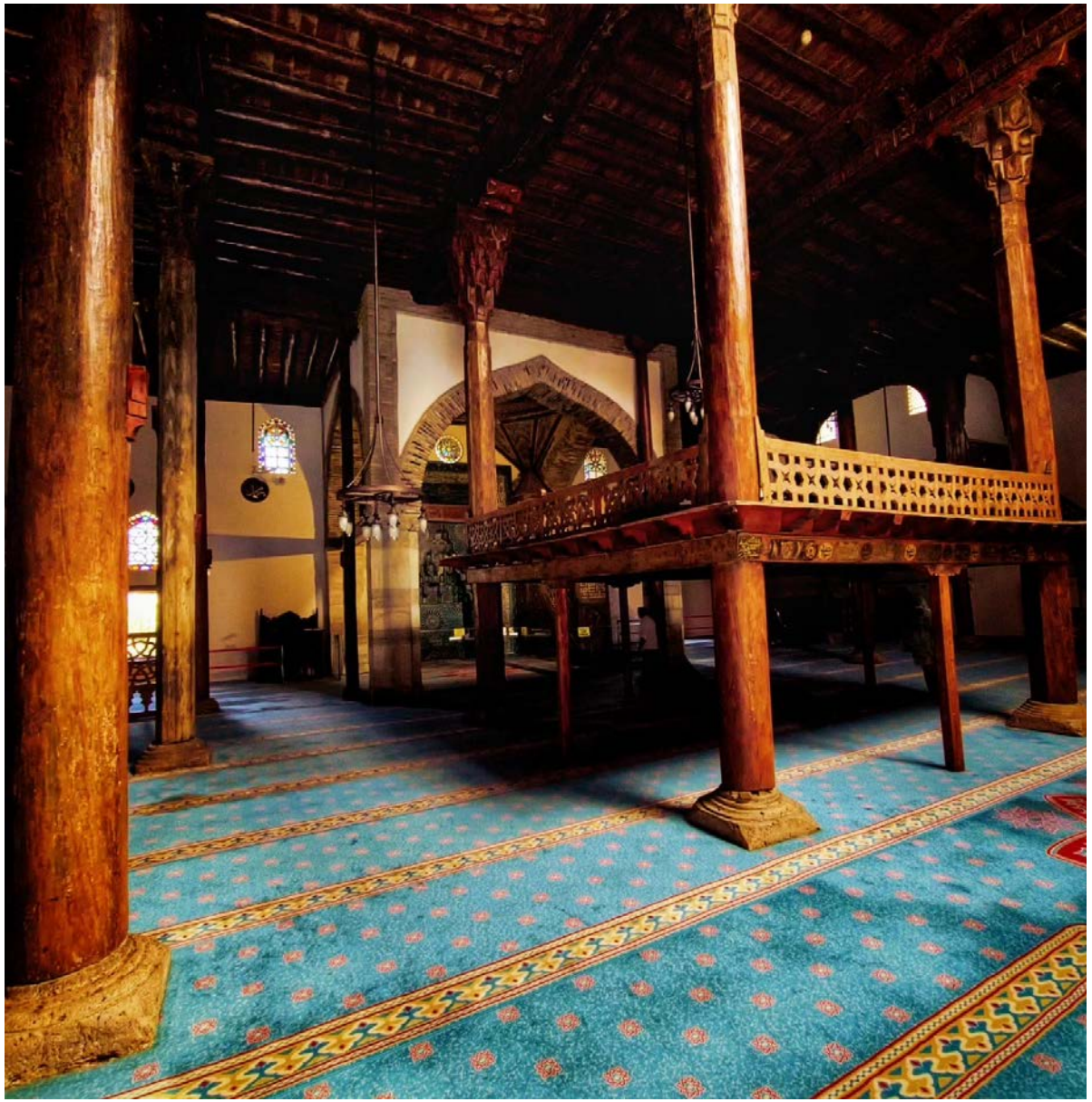
Bugün: 05.Eylül.2025

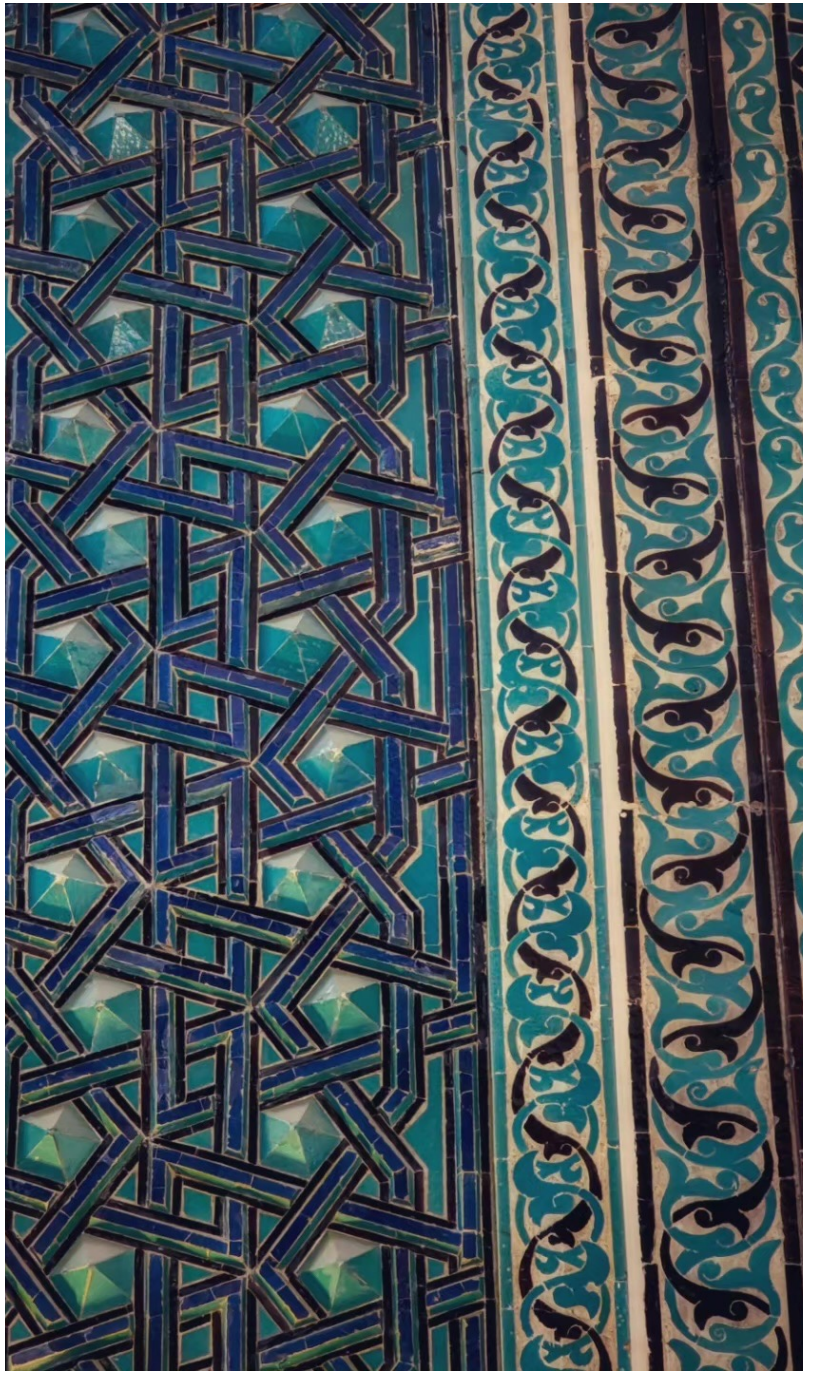
Geometri Desenleri

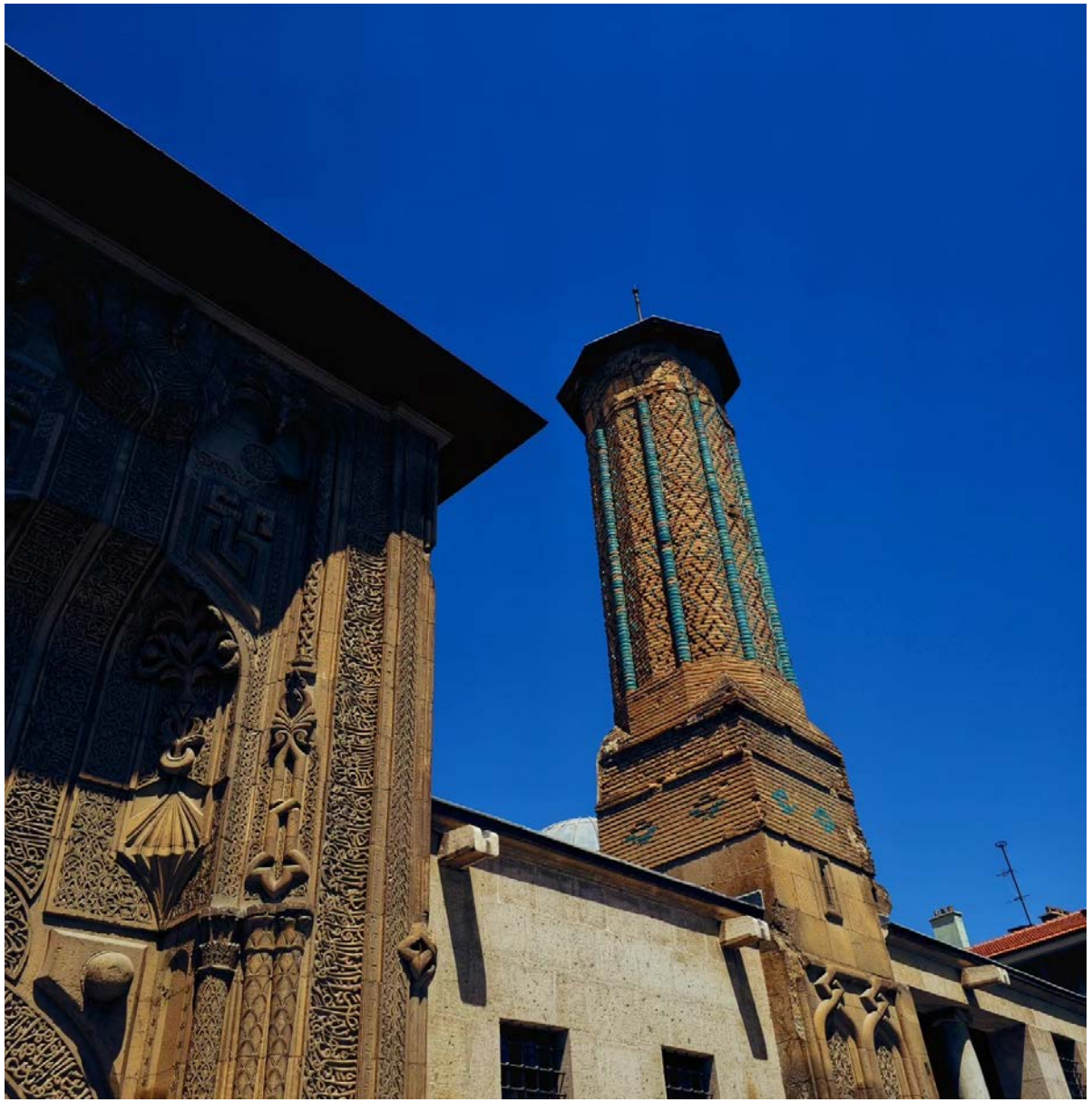
Yansıma transformasyon fonksiyonu

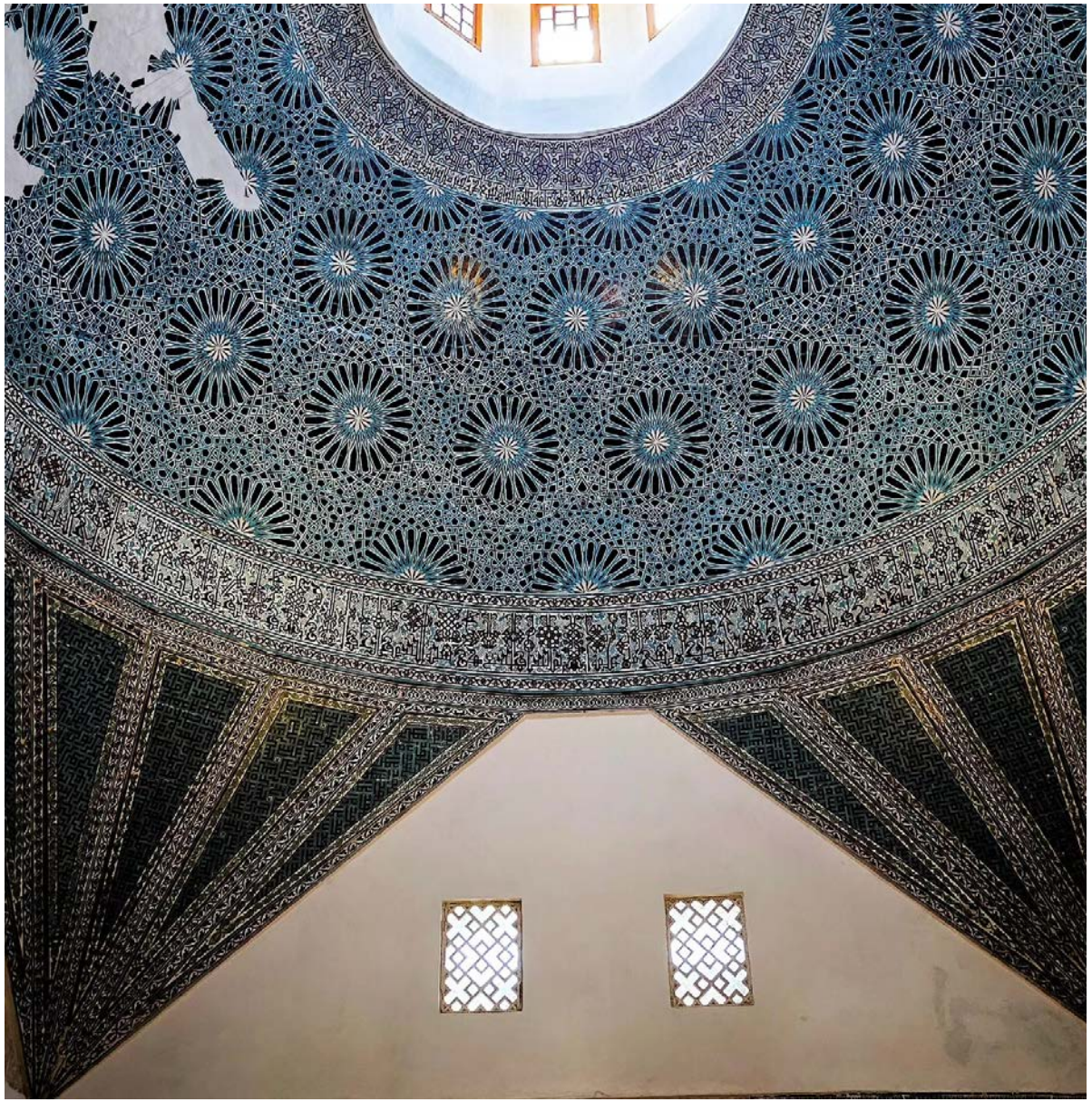



Black Sea
Varna
Варна
Ruse
Русе
Niš
Ниш
Sofia
София
Bulgaria
Burgas
Бургас
Plovdiv
Пловдив
Edirne
North
Macedonia
Thessaloniki
Θεσσαλονίκη
Sinop
Kastamonu
Çorlu
İstanbul
Silivri
İzmit
Yalova
Bolu
Çorum
Amasya
Çanakkale
Bursa
Ankara
Eskişehir
Balıkesir
Greece
Türkiye
İzmir
Aksaray
Kayseri
Patras
Πάτρα
Athens
Αθήνα
Aegean Sea
Eşrefoğlu Camii
Kuşadası
Denizli
Kalamata
Καλαμάτα
Bodrum
Marmaris
Fethiye
Antalya
Side
Alanya
Adana
Mersin
İskender
Fira
Φηρά
Chania
Χανιά
Heraklion
Ηράκλειο
Kriti
Nicosia
Λευκωσία
Kıbrıs
Larnaca
Λάρνακα
Paphos
Πάφος
Lebanon
Beirut
بيروت
Haifa
Al Bayda
البيضاء
WEST BANK



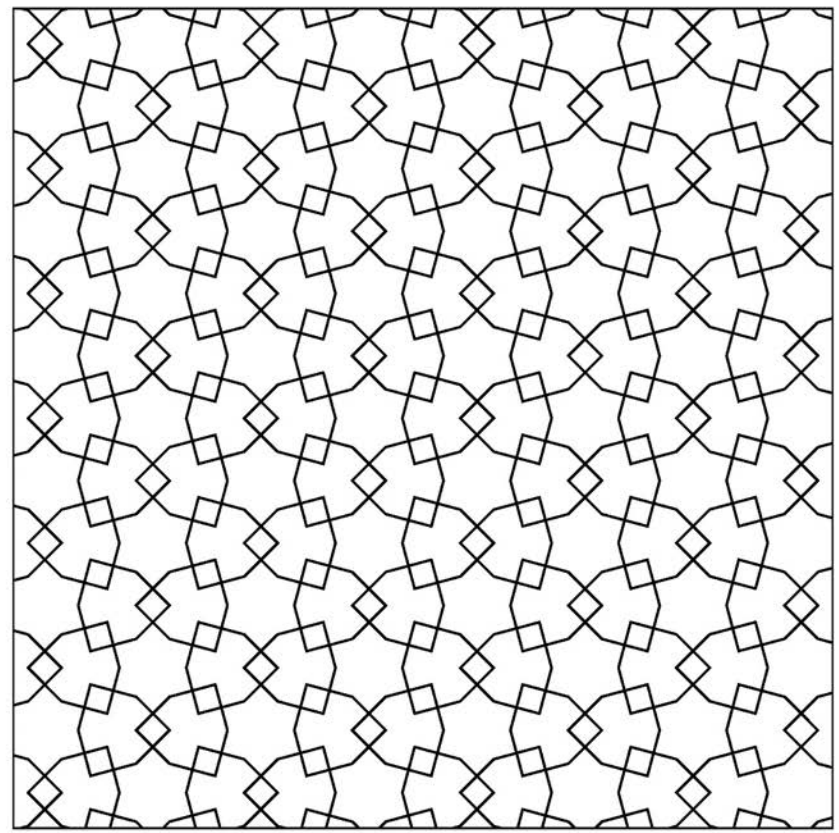




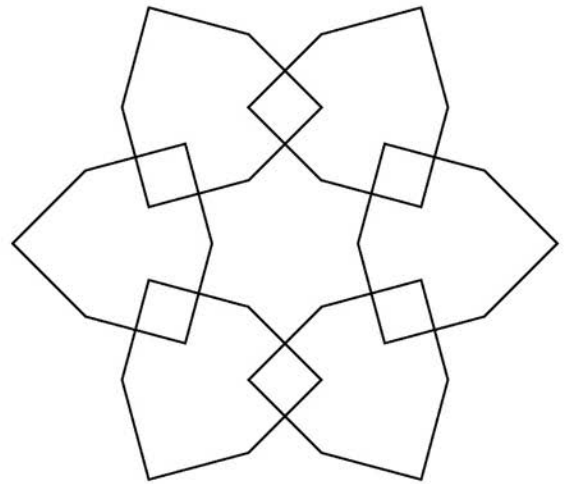


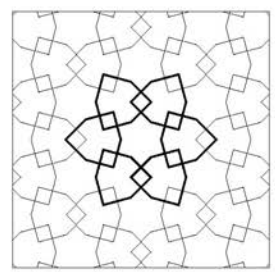
Geometrik Deseni Kodlamak
Örnek 6

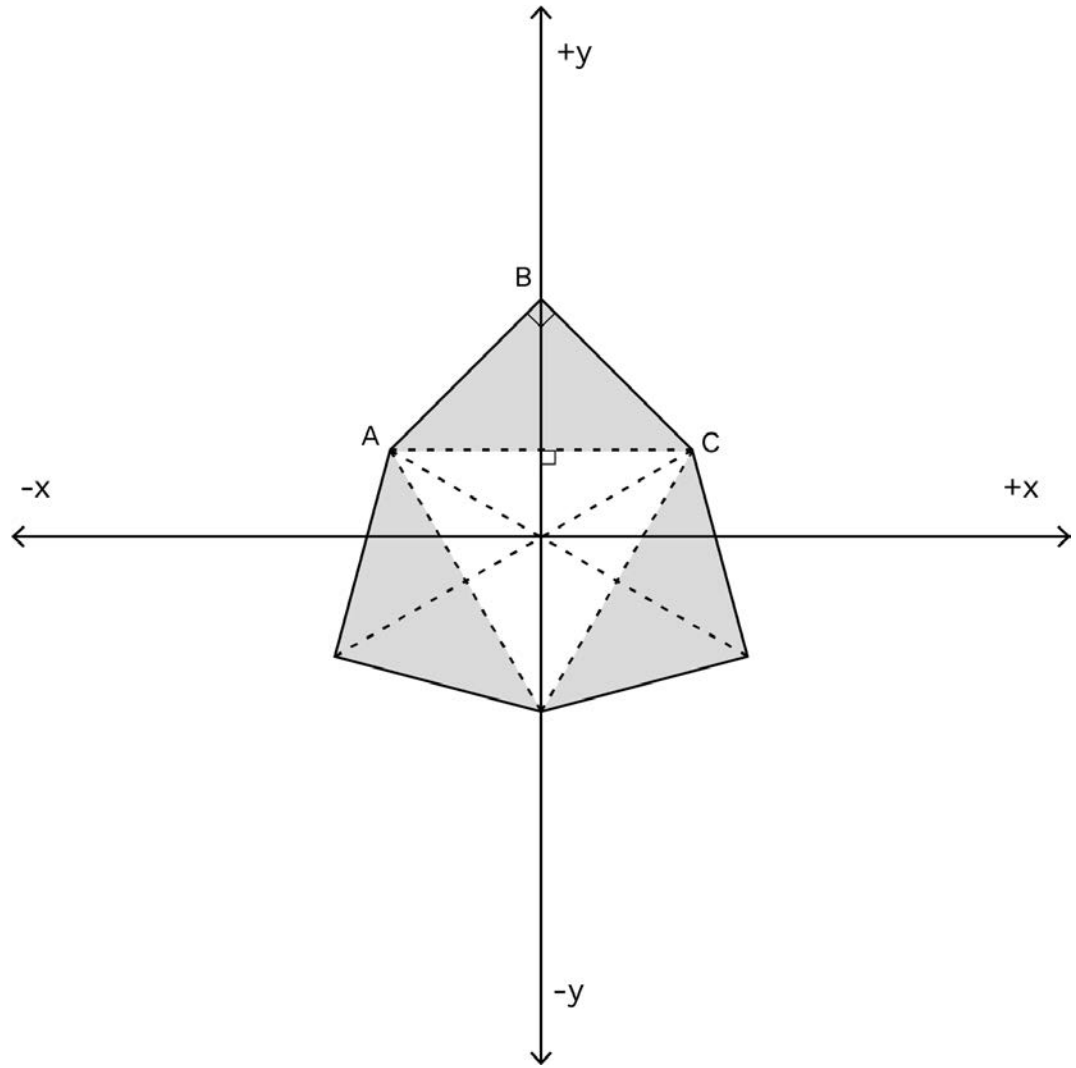
Aşağıdaki deseni inceleyin ve bu deseni oluşturan temel görsel bileşeni bulmaya çalışın



Motif

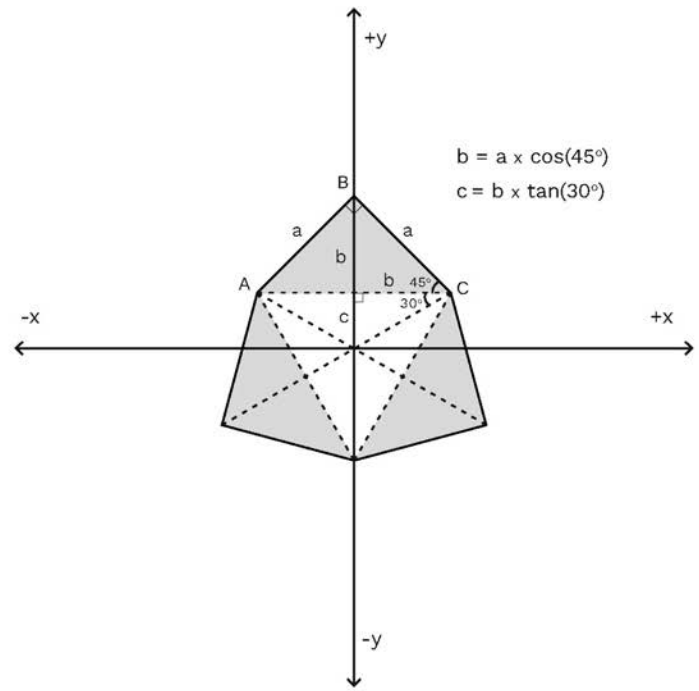





+y
B
A
C
-x
+x
-y

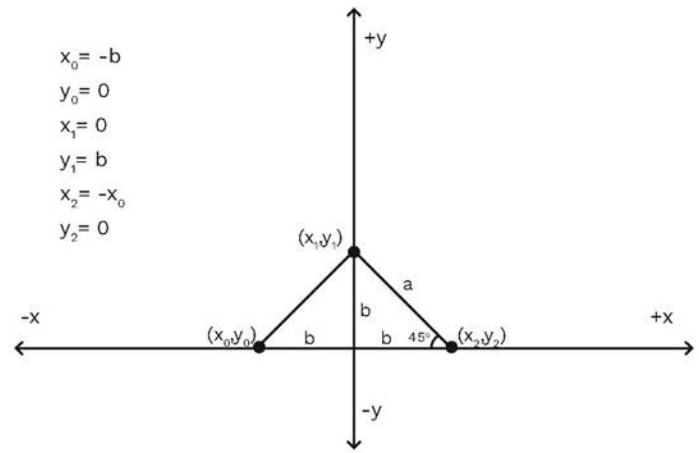
Temel Görsel Bileşini İnceleyelim

Aşama 1 : Vertex noktalarını bulalım



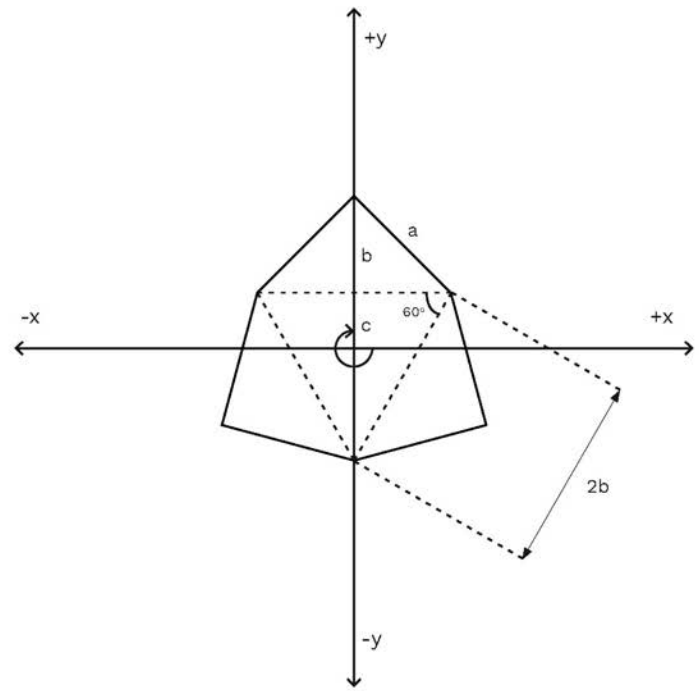
Temel Görsel Bileşini İnceleyelim

Aşama 2 : Temel yapıyı oluşturan üçgenin vertex noktalarını bulalım



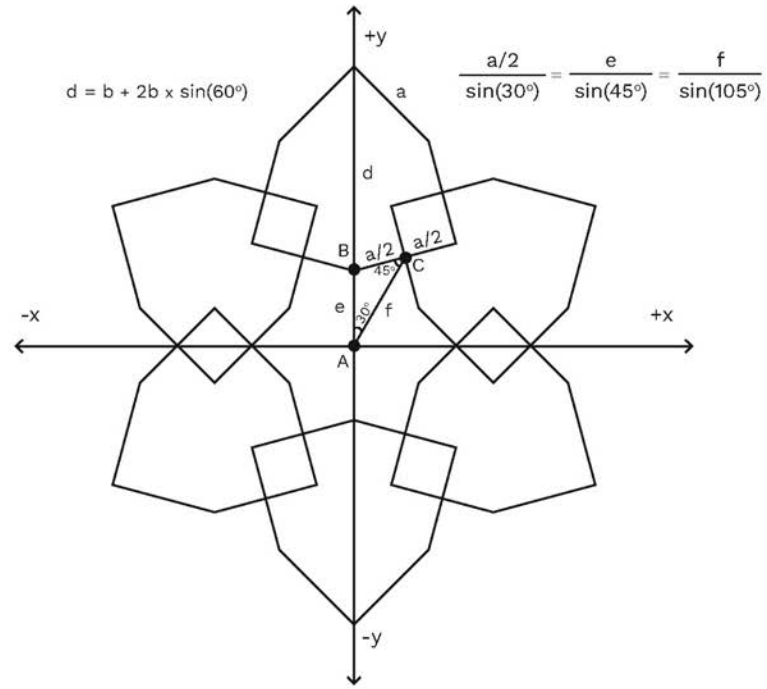
Temel Görsel Bileşini İnceleyelim

Adım 3: Üçgen şeklinin kopyalanması, çevrilmesi ve orijin etrafında üç kez döndürülmesi gerekiyor



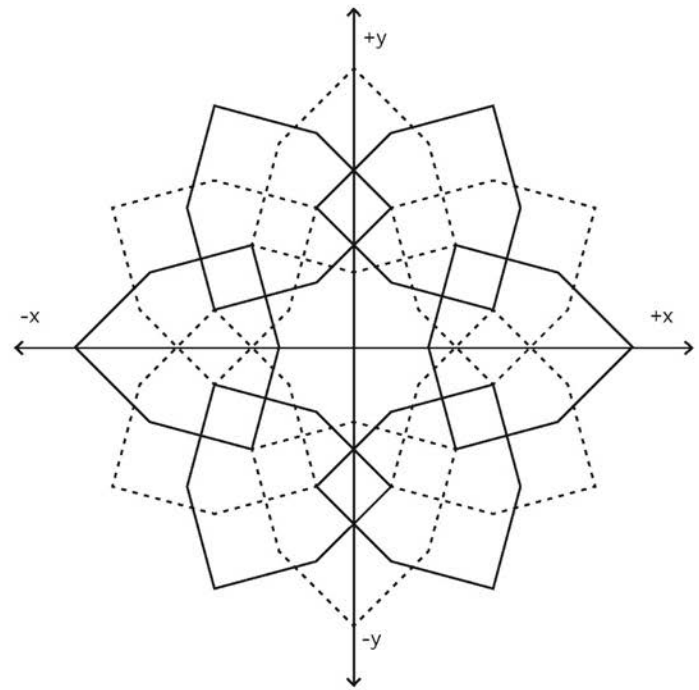
Temel Görsel Bileşini İnceleyelim

Adım 4 : Temel şeklin kesişimleri arasında mini kareler oluşturulur. Karenin boyutu şeklin öteleme boyutuna bağlıdır. Burada kare boyutunun “a” harfinin yarısına eşit olduğunu varsayıyoruz.



Temel Görsel Bileşini İnceleyelim

Adım 5: Motifin yerleşimini tamamlamak için 90 derecelik bir dönüşe ihtiyacı vardır

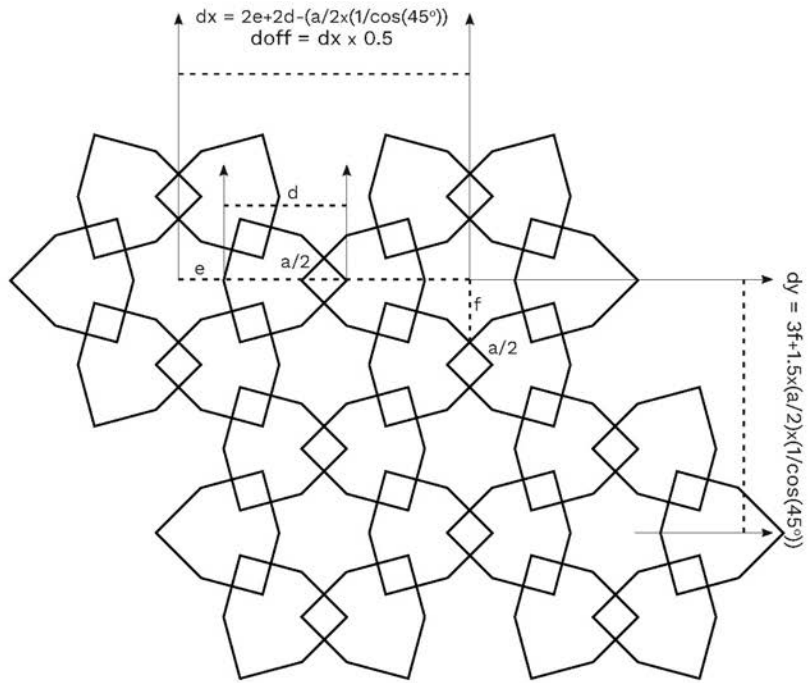


Motifi Oluşturmak

```
//scale factor
let a = 60;

function setup() {
   createCanvas(400, 400);
   angleMode(DEGREES);
   noLoop();
   noFill();
}

function draw() {

   b = a * cos(45);
   c = b * tan(30);

   let x0, y0, x1, y1, x2, y2;
   x0 = -b;
   y0 = 0;
   x1 = 0;
   y1 = b;
   x2 = -x0;
   y2 = 0;

   push();
      translate(width * 0.5, height * 0.5);
      rotate(90);
      for (let k = 0; k < 6; k++) {
         push();
            rotate(k * 60);
            translate(0, 2 * b * sin(60) - c + 0.5 * a * (sin(45) / sin(30)));
            rotate(60);
            for (let i = 0; i < 3; i++) {
               push();
                  rotate(120 * i);
                  translate(0, -c);
                  beginShape();
                  vertex(x0, -y0);
                  vertex(x1, -y1);
                  vertex(x2, -y2);
                  endShape();
               pop();
            }
         pop();
      }
   pop();
}
```

Bezeme Yapısını İnceleyelim

Adım 6 : Yerleşimdeki dx, dy ve doff değerlerini hesaplamamız gerekiyor.



Bezeme Kodu

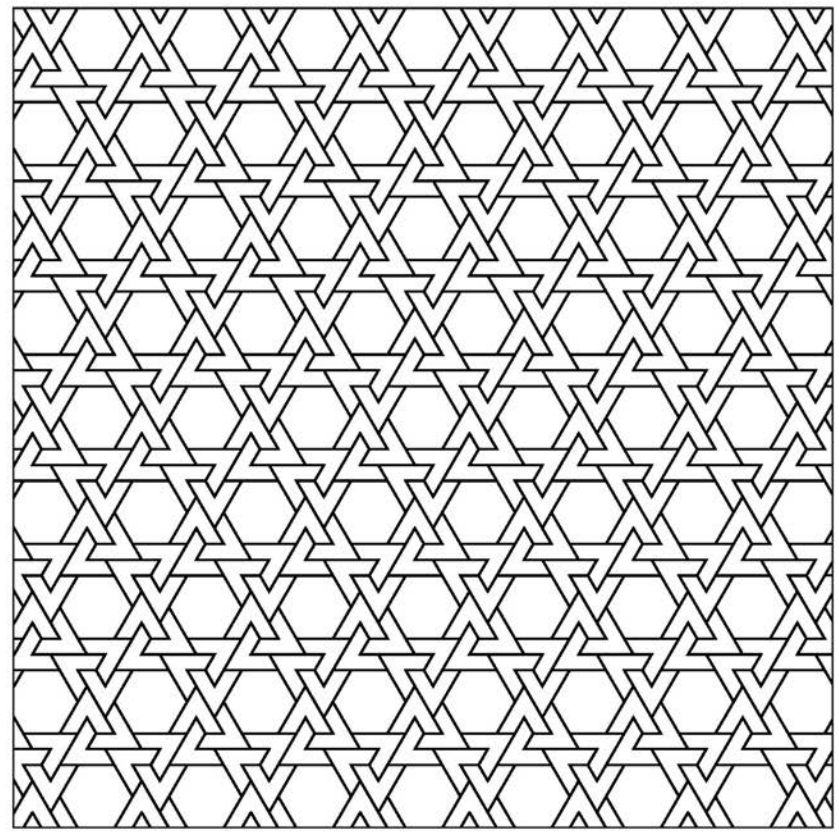
```
//Motif class
class Motif {
    constructor(a) {
        this.a = a;
    }

    display() {
        let a = this.a;
        let b = a * cos(45);
        let c = b * tan(30);
        let d = 2 * b * sin(60) - c + 0.5 * a * (sin(45) / sin(30));

        let x0, y0, x1, y1, x2, y2;
        x0 = -b;
        y0 = 0;
        x1 = 0;
        y1 = b;
        x2 = -x0;
        y2 = 0;

        rotate(90);
        for (let k = 0; k < 6; k++) {
            push();
                rotate(k * 60);
                translate(0, d);
                rotate(60);
                for (let i = 0; i < 3; i++) {
                    push();
                        rotate(120 * i);
                        translate(0, -c);
                        beginShape();
                        vertex(x0, -y0);
                        vertex(x1, -y1);
                        vertex(x2, -y2);
                        endShape();
                    pop();
                }
            pop();
        }
    }
}

//scale factor
let a = 40;
let motif = new Motif(a);
let nRow;
let nCol;
let dx, dy;
let doff;
```


Bezeme Kodu

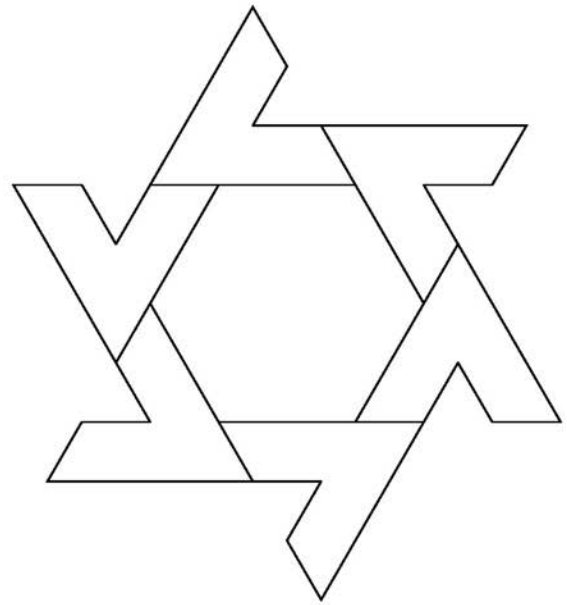
```
function setup() {
    createCanvas(800, 800);
    angleMode(DEGREES);
    noLoop();
    noFill();
    let b = a * cos(45);
    let c = b * tan(30);
    let d = b + 2 * b * sin(60);
    let e = 0.5 * a * (sin(45) / sin(30));
    let f = e * (sin(105) / sin(45));

    dx = 2 * e + 2 * d - 0.5 * a * (1 / cos(45));
    dy = 3 * f + 1.5 * (0.5 * a * (1 / cos(45)));
    doff = dx / 2;

    //approximate the nRow and nCol values
    nRow = ceil(height / dy);
    nCol = 1 + ceil(width / dx);
}

function draw() {
    push();
        for (let r = 0; r < nRow; r++) {
            for (let c = 0; c < nCol; c++) {
                push();
                    if (r % 2 == 1) {
                        //row 1,3,5,7
                        translate(-doff, 0);
                    }
                    translate(dx * c, dy * r);
                    motif.display();
                pop();
            }
        }
    pop();
}
```

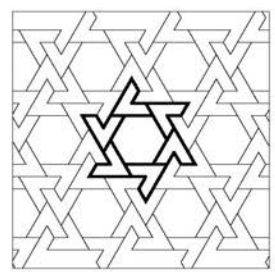
Geometrik Deseni Kodlamak
Örnek 7

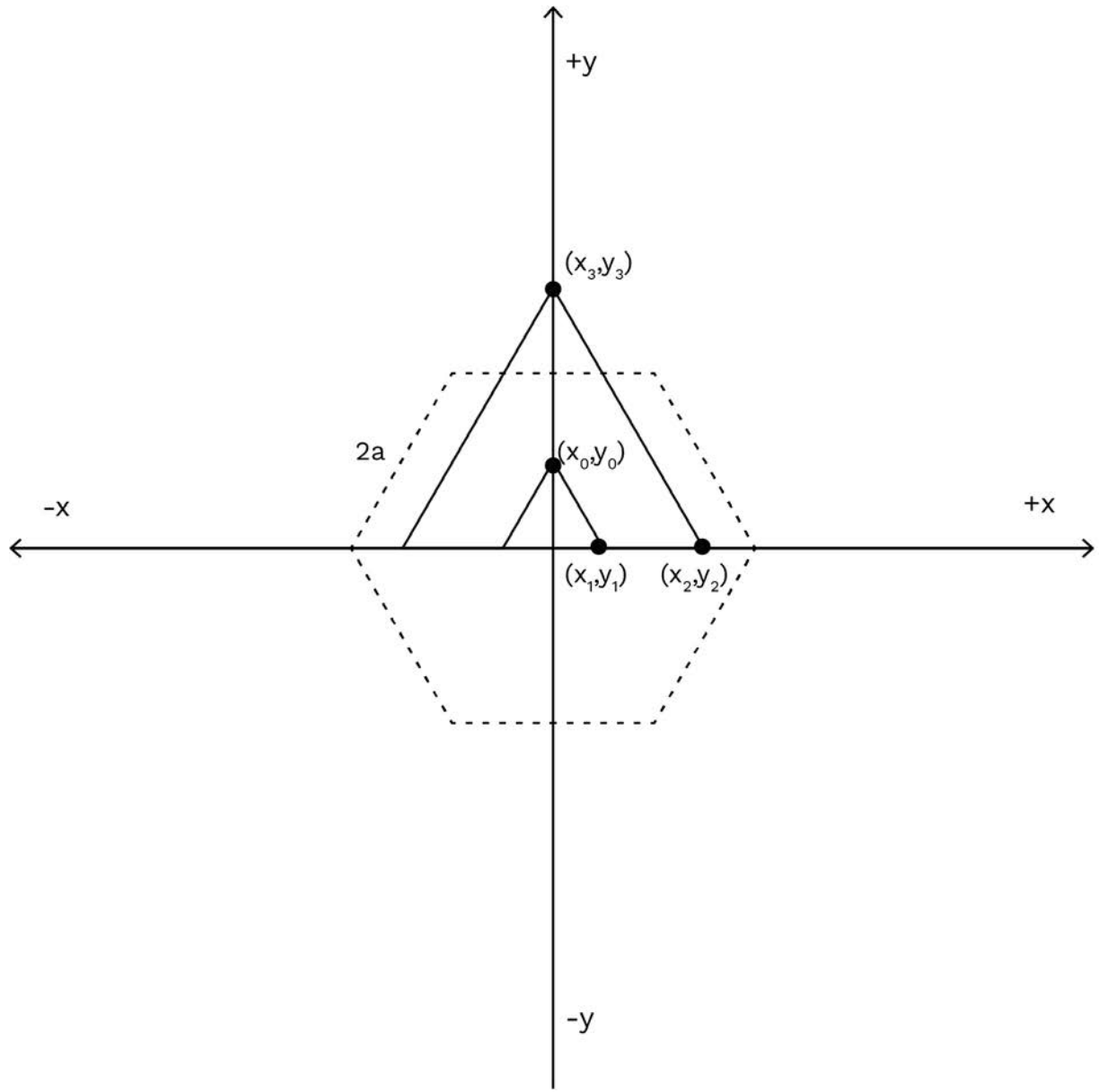
Aşağıdaki deseni inceleyin ve bu deseni oluşturan temel görsel bileşeni bulmaya çalışın



Motif

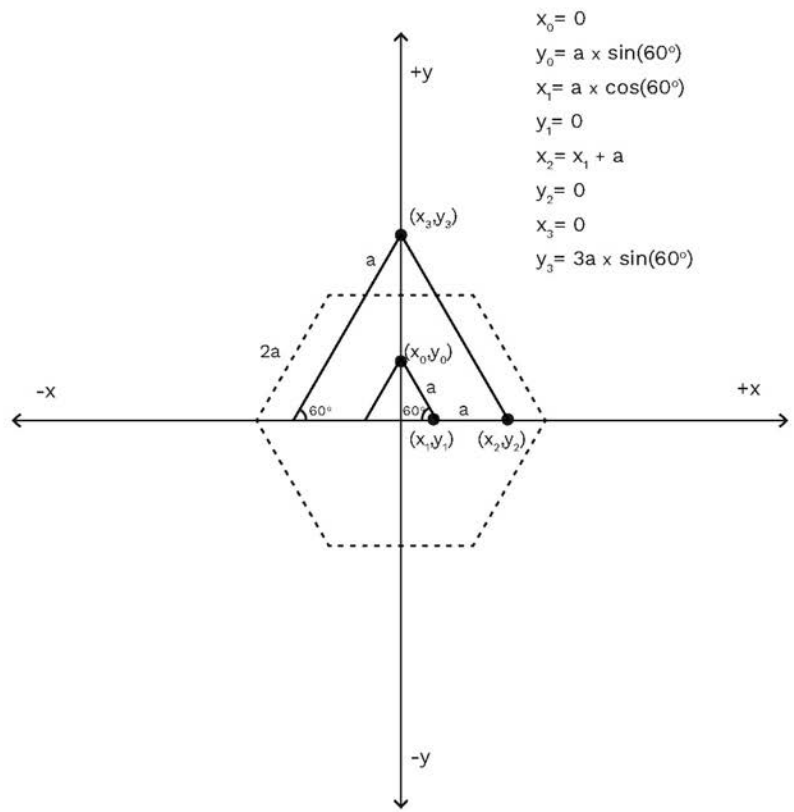





+y
(x_3,y_3)
2a
(x_0,y_0)
-x
+x
(x_1,y_1)
(x_2,y_2)
-y

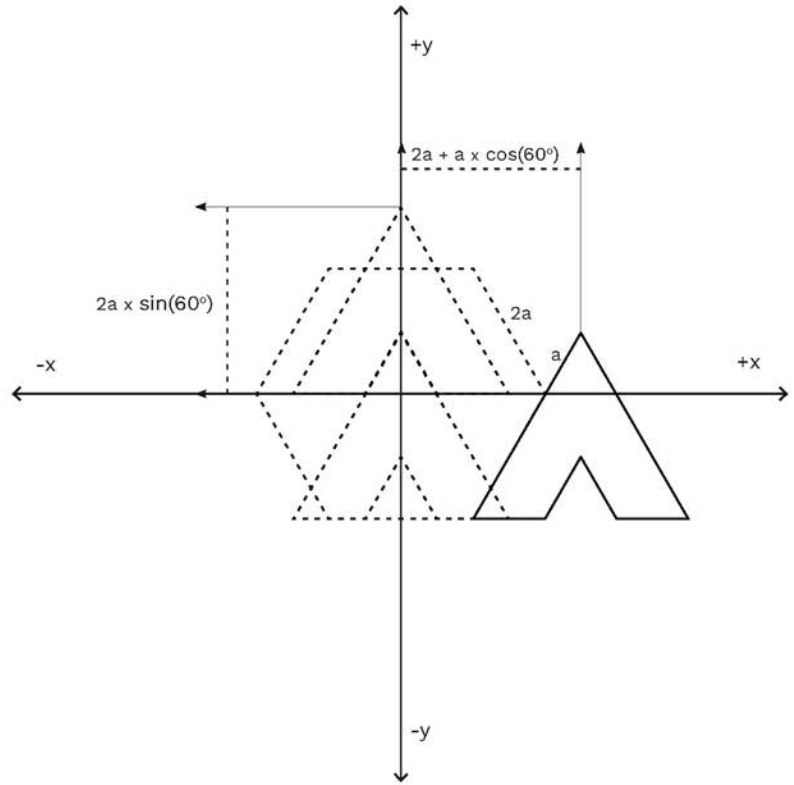
Temel Görsel Bileşini İnceleyelim

Aşama 1: Yapıcı elemanın köşe noktalarını bulalım

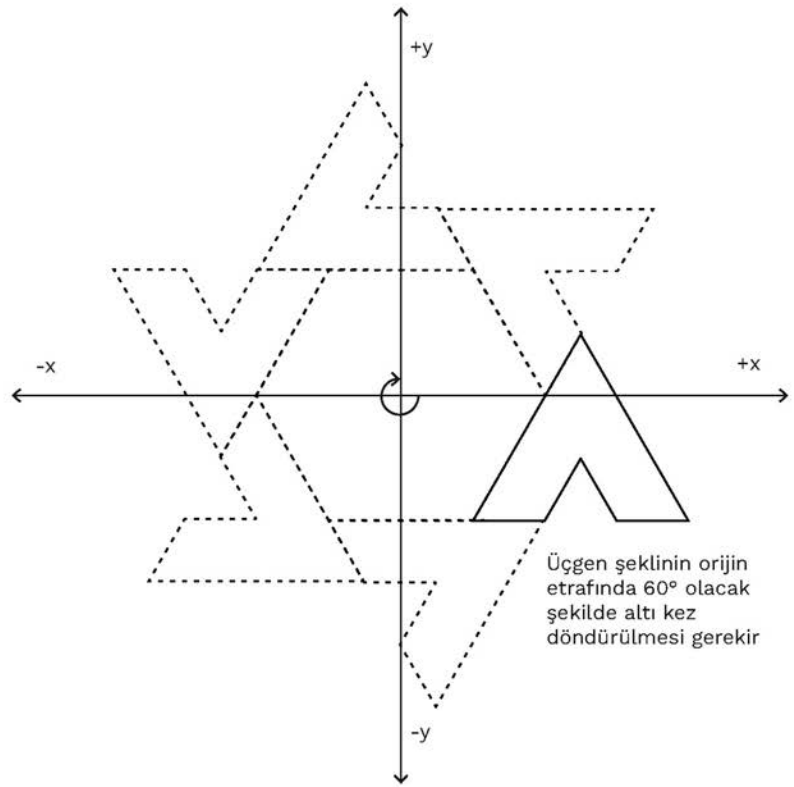


Temel Görsel Bileşini İnceleyelim

Aşama 2: Üçgen şeklin aşağıdaki konuma çevrilmesi gerekiyor



Temel Görsel Bileşini İnceleyelim



Motifi Oluşturmak

```
//scale factor
let a = 40;

function setup() {
  createCanvas(400, 400);
  angleMode(DEGREES);
  noFill();
  noLoop();
}

function draw() {
  let x0,y0,x1,y1,x2,y2,x3,y3;
  x0 = 0;
  y0 = a * sin(60);
  x1 = a * cos(60);
  y1 = 0;
  x2 = x1 + a;
  y2 = 0;
  x3 = 0;
  y3 = 3*a * sin(60);

  push();
    translate(width*0.5,height*0.5);
    for(let i=0;i<6;i++){
      push();
        rotate(i*60);
        push();
          translate(2*a+a*cos(60),2*a*sin(60));
          beginShape();
          vertex(x0,-y0);
          vertex(x1,-y1);
          vertex(x2,-y2);
          vertex(x3,-y3);
          endShape();
          //mirror on y-axis
          push();
            scale(-1,1);
            beginShape();
            vertex(x0,-y0);
            vertex(x1,-y1);
            vertex(x2,-y2);
            vertex(x3,-y3);
            endShape();
          pop();
        pop();
      pop();
    }
  pop();
}
```

Bezeme Kodu

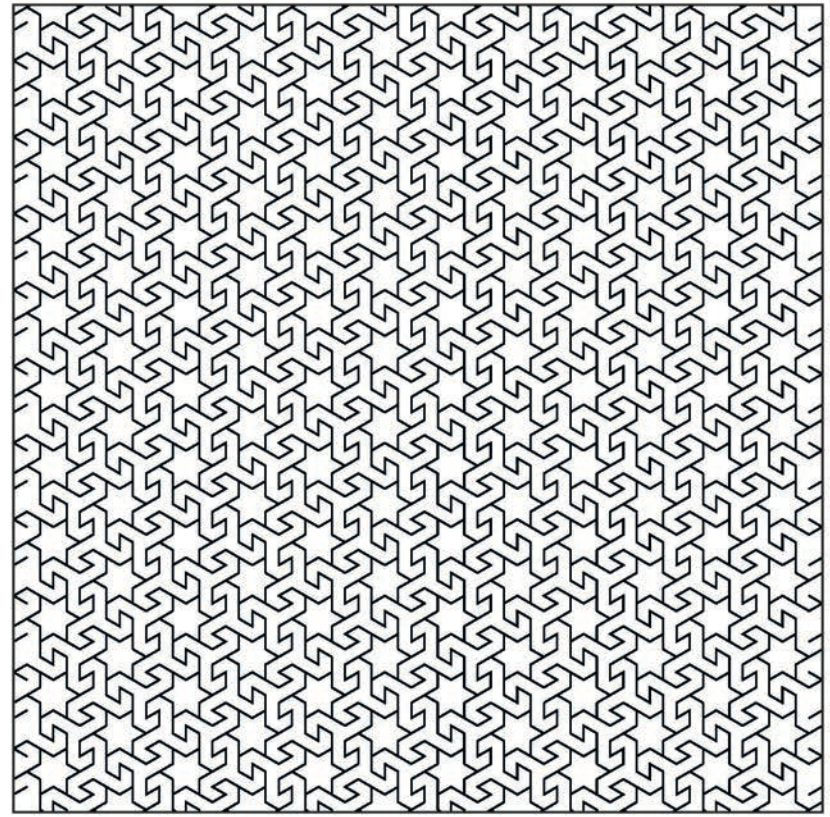
```
// Motif class
class Motif {
  constructor(a) {
    this.a = a;
  }

  display() {
    let x0, y0, x1, y1, x2, y2, x3, y3;
    x0 = 0;
    y0 = this.a * sin(60);
    x1 = this.a * cos(60);
    y1 = 0;
    x2 = x1 + this.a;
    y2 = 0;
    x3 = 0;
    y3 = 3 * this.a * sin(60);

    for (let i = 0; i < 6; i++) {
      push();
        rotate(i * 60);
        push();
          translate(2 * this.a + this.a * cos(60), 2 * this.a * sin(60));
          beginShape();
          vertex(x0, -y0);
          vertex(x1, -y1);
          vertex(x2, -y2);
          vertex(x3, -y3);
          endShape();
          //mirror on y-axis
          push();
            scale(-1, 1);
            beginShape();
            vertex(x0, -y0);
            vertex(x1, -y1);
            vertex(x2, -y2);
            vertex(x3, -y3);
            endShape();
          pop();
        pop();
      pop();
    }
  }
}
```



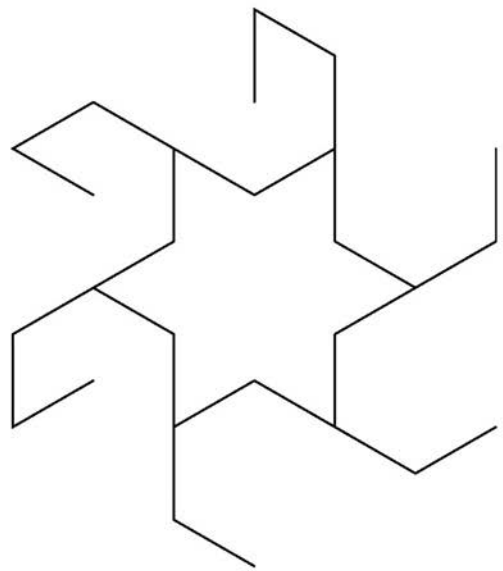
```
//scale factor
let a = 16;
let motif = new Motif(a);
let nRow;
let nCol;
let dx, dy, doff;

function setup() {
  createCanvas(800, 800);
  angleMode(DEGREES);
  noFill();
  noLoop();

  dx = 6 * a;
  dy = 6 * a * sin(60);
  doff = dx * 0.5;

  //approximate the nRow and nCol values
  nCol = 1 + ceil(width / dx);
  nRow = 1 + ceil(height / dy);
}

function draw() {
  for (let c = 0; c < nCol; c++) {
    for (let r = 0; r < nRow; r++) {
      push();
        if (r % 2 == 0) {
          //columns 0,2,4,6
          translate(doff, 0);
        }
        translate(dx * c, dy * r);
        motif.display();
      pop();
    }
  }
}
```

Geometrik Deseni Kodlamak

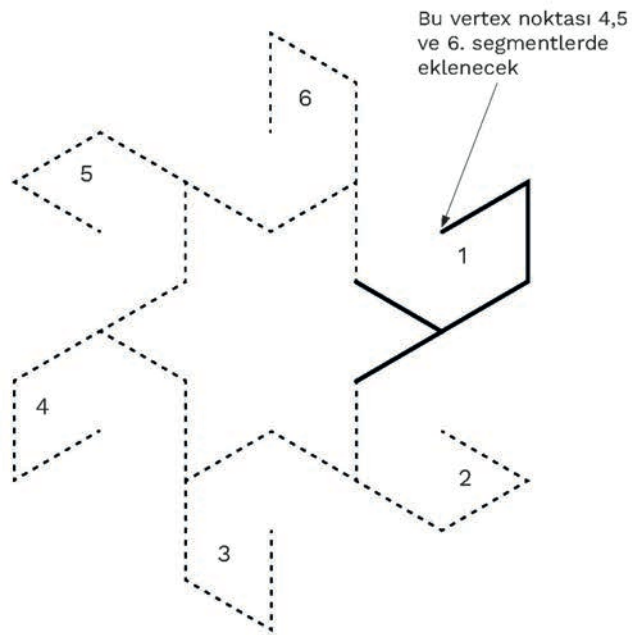
Aşağıdaki deseni inceleyin ve bu deseni oluşturan temel görsel bileşeni bulmaya çalışın.



Motif

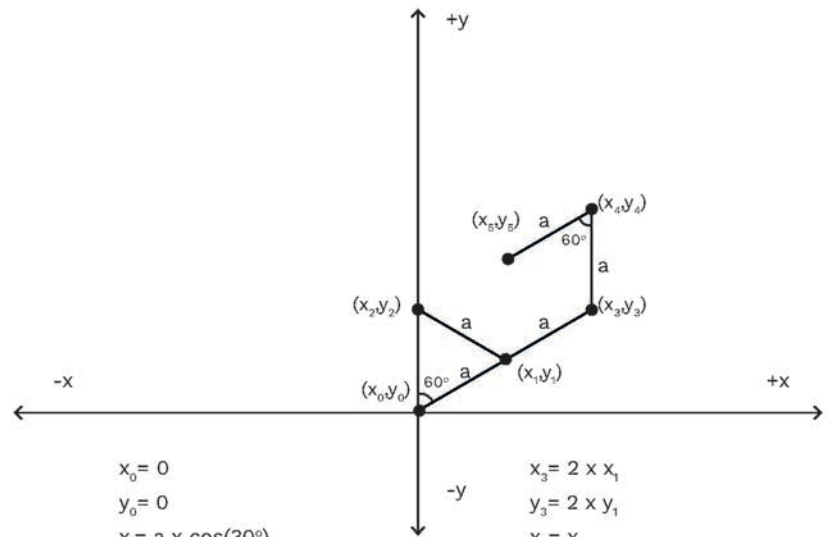


Temel Görsel Bileşini İnceleyelim



Açıları ve Vertex noktalarını tespit etmek

Aşama 1 : Vertex noktalarını bulalım



$x_0 = 0$

$y_0 = 0$

$x_1 = a \times \cos(30^\circ)$

$y_1 = a \times \sin(30^\circ)$

$x_2 = 0$

$y_2 = 2 \times y_1$

$x_3 = 2 \times x_1$

$y_3 = 2 \times y_1$

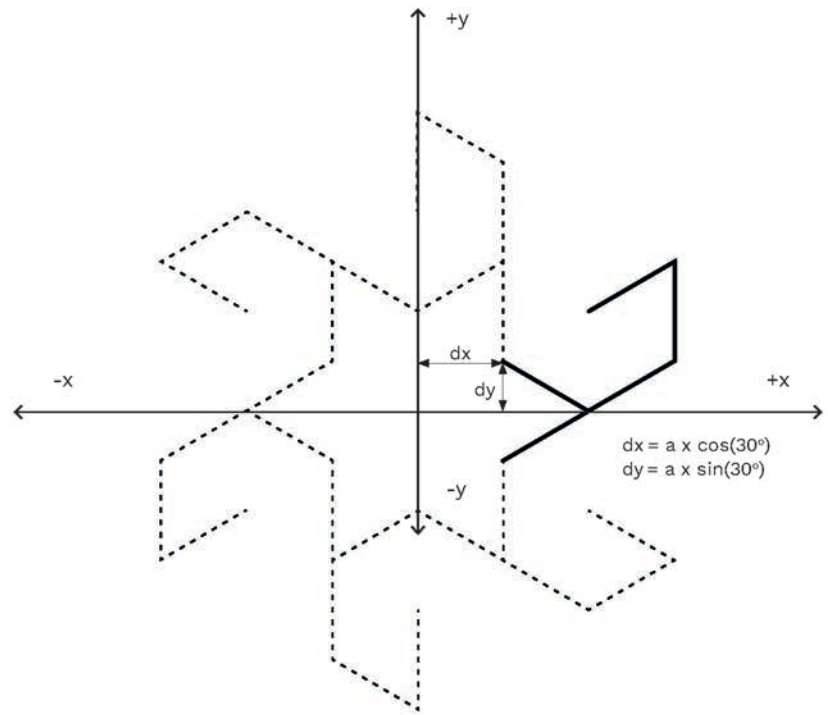
$x_4 = x_3$

$y_4 = y_3 + a$

$x_5 = x_1$

$y_5 = y_1 + a$

Açıları ve Vertex noktalarını tespit etmek



Motifi Oluşturmak

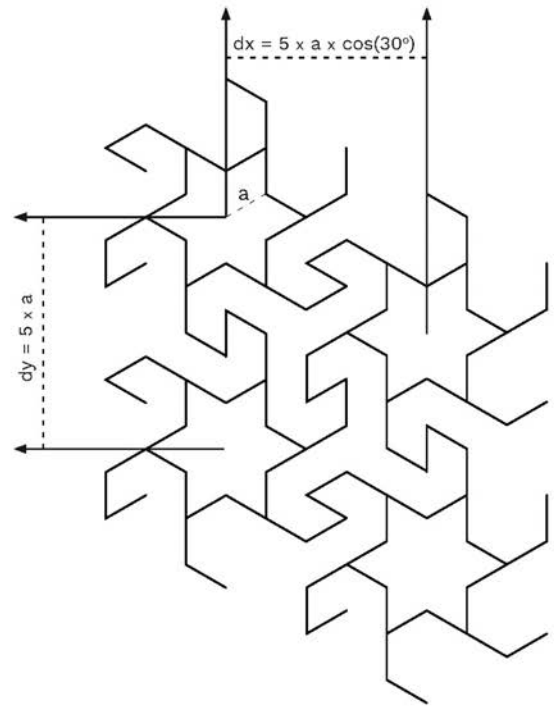
```
/*
Code written by Selcuk ARTUT 2022
Geometric Patterns with Creative Coding
All rights reserved
*/
let a;
function setup() {
  createCanvas(400, 400);
  angleMode(DEGREES);
  noFill();
  a = 40;
  noLoop();
}

function draw() {
  background(255);
  noFill();
  let x0,y0,x1,y1,x2,y2,x3,y3,x4,y4,x5,y5;
  let dx = a * cos(30);
  let dy = a * sin(30);
  push();
  translate(width*0.5,height*0.5);
  for(let i = 0; i<6; i++){
    push();
    rotate(60*i);
    translate(dx,dy);

    beginShape();
    x0 = 0;
    y0 = 0;
    x1 = a * cos(30);
    y1 = a * sin(30);
    x2 = 0;
    y2 = 2 * y1;
    vertex(x0,-y0);
    vertex(x1,-y1);
    vertex(x2,-y2);
    endShape();

    beginShape();
    x3 = 2 * x1;
    y3 = 2 * y1;
    x4 = x3;
    y4 = y3 + a;
    x5 = x1;
    y5 = y1 + a;
    vertex(x1,-y1);
    vertex(x3,-y3);
```

Motifi Oluşturmak

```
    vertex(x4,-y4);
    //special condition to include the 4th,5th and 6th
    if(i>=3){
      vertex(x5,-y5);
    }
    endShape();
    pop();
  }
  pop();
}
```

Bezeme Yapısını İnceleyelim

Aşama 2 : Yukarı ve aşağıya kaymaları belirleyecek xoffset ve yoffset değerlerini hesaplayalım.



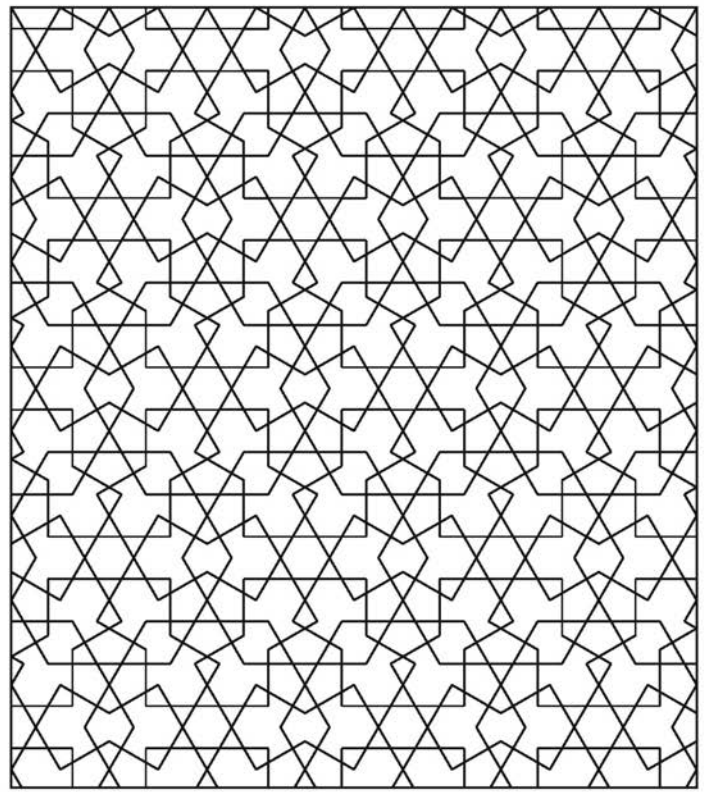
Bezeme Kodu

```
/*
Code written by Selcuk ARTUT 2022
Geometric Patterns with Creative Coding
All rights reserved
*/
class Motif {
  constructor(a) {
    this.a = a;
  }

  display() {
    let x0,y0,x1,y1,x2,y2,x3,y3,x4,y4,x5,y5;
    let dx = this.a * cos(30);
    let dy = this.a * sin(30);
    for(let i = 0; i<6; i++){
      push();
      rotate(60*i);
      translate(dx,dy);

      beginShape();
      x0 = 0;
      y0 = 0;
      x1 = this.a * cos(30);
      y1 = this.a * sin(30);
      x2 = 0;
      y2 = 2 * y1;
      vertex(x0,-y0);
      vertex(x1,-y1);
      vertex(x2,-y2);
      endShape();

      beginShape();
      x3 = 2 * x1;
      y3 = 2 * y1;
      x4 = x3;
      y4 = y3 + this.a;
      x5 = x1;
      y5 = y1 + this.a;
      vertex(x1,-y1);
      vertex(x3,-y3);
      vertex(x4,-y4);
      if(i>=3){
        vertex(x5,-y5);
      }
      endShape();
      pop();
    }
  }
}
```

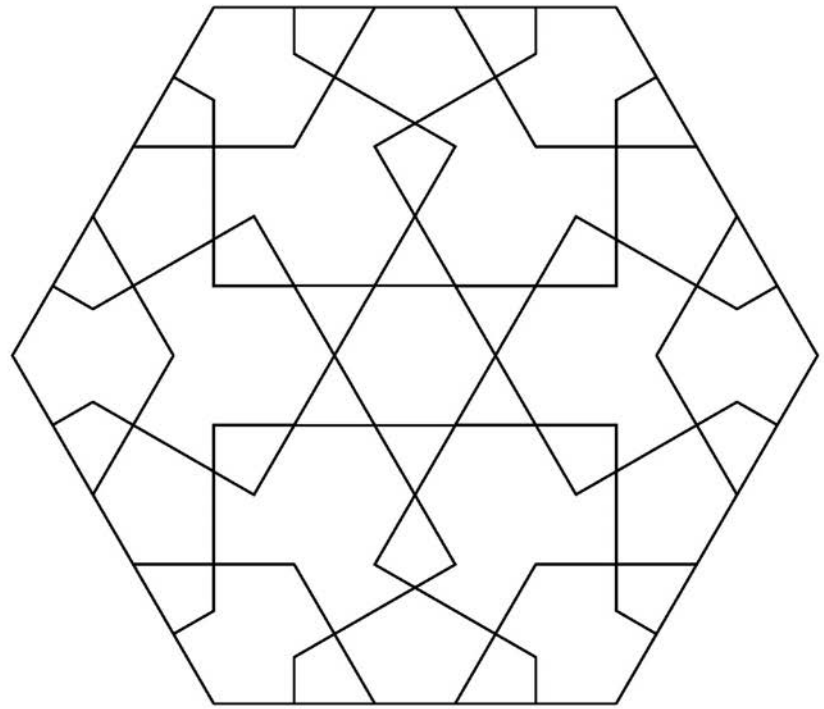
```
let a = 20;
let xOff,yOff;
let nRow;
let nCol;
let motif = new Motif(a);

function setup() {
  createCanvas(800, 800);
  angleMode(DEGREES);
  noFill();
  noLoop();

  xOff = 5 * a * cos(30);
  yOff = 5 * a;
  yOffSet = yOff * 0.5;

  nRow = floor(height / xOff);
  nCol = floor(width / yOff);
}

function draw() {
  push();
  for (let c = 0; c < nCol; c++) {
    for (let r = 0; r < nRow; r++) {
      push();
      translate(xOff * c,  yOff * r);
      if(c%2==1){
        translate(0,  yOffSet);
      }
      motif.display();
      pop();
    }
  }
  pop();
}
```

Geometrik Deseni Kodlamak

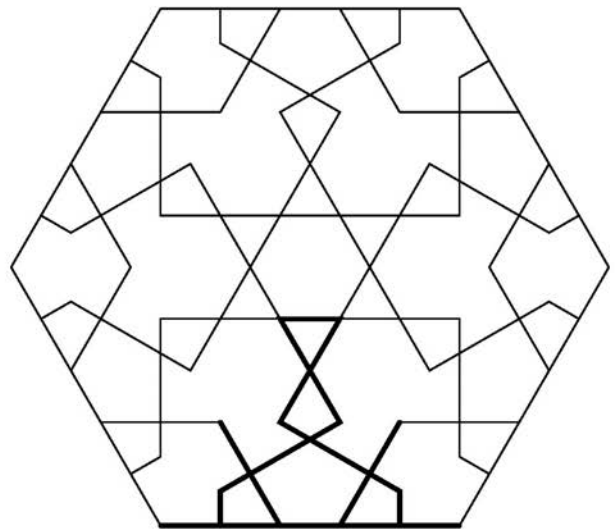
Aşağıdaki deseni inceleyin ve bu deseni oluşturan temel görsel bileşeni bulmaya çalışın.



Motif

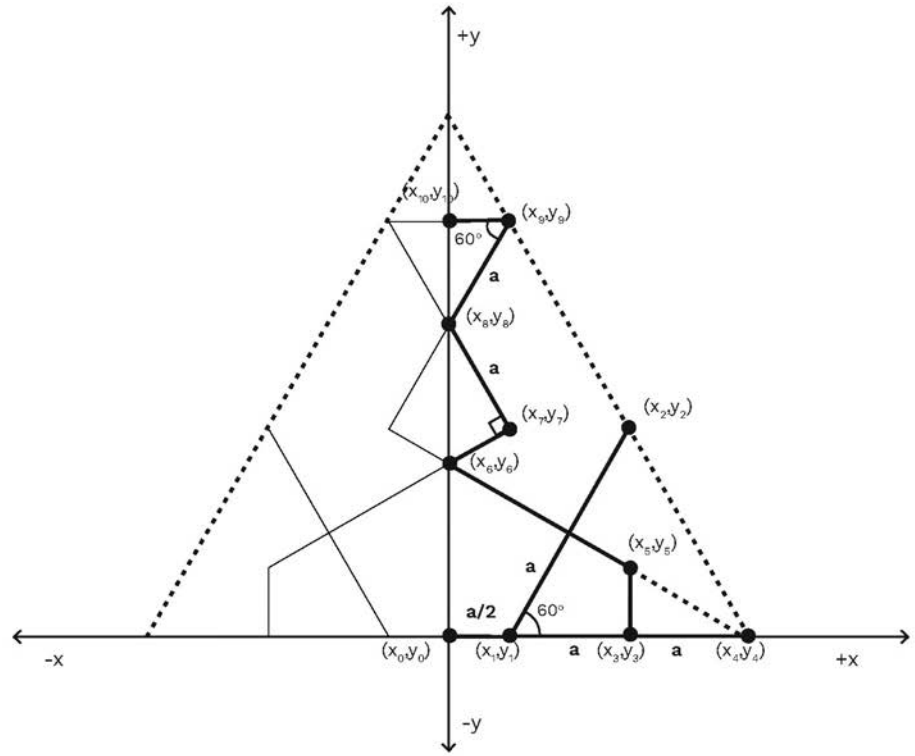


Temel Görsel Bileşini İnceleyelim



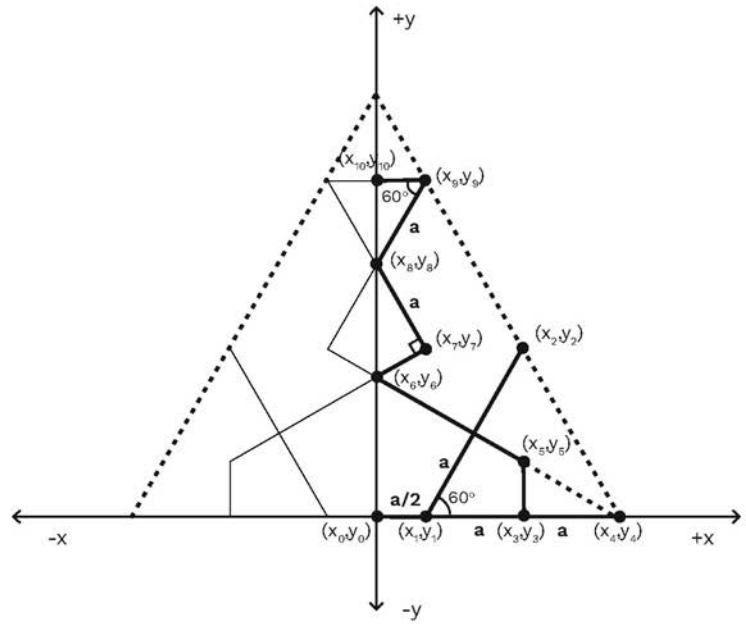
Açıları ve Vertex noktalarını tespit etmek

Aşama 1 : Vertex noktalarını bulalım



Açıları ve Vertex noktalarını tespit etmek

Aşama 1 : Vertex noktalarını bulalım



$x_0 = 0$

$y_0 = 0$

$x_1 = 0.5 \times a$

$y_1 = 0$

$x_2 = 2 \times a \times \cos(60^o) + 0.5 \times a$

$y_2 = 2 \times a \times \sin(60^o)$

$x_3 = 1.5 \times a$

$y_3 = 0$

$x_4 = 2.5 \times a$

$y_4 = 0$

$x_5 = 1.5 \times a$

$y_5 = a/\tan(60^o)$

$x_6 = 0$

$y_6 = 2.5 \times a/\tan(60^o)$

$x_7 = a \times \sin(30^o)$

$y_7 = 2 \times a \times \sin(60^o)$

$x_8 = 0$

$y_8 = 3 \times a \times \sin(60^o)$

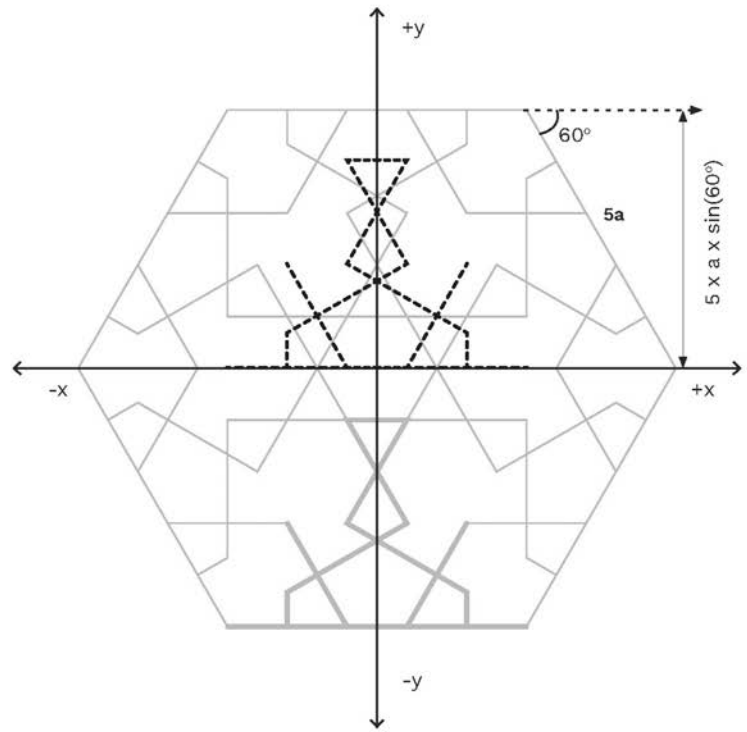
$x_9 = a \times \sin(30^o)$

$y_9 = 4 \times a \times \sin(60^o)$

$x_{10} = 0$

$y_{10} = 4 \times a \times \sin(60^o)$

Aşama 2 : Motif aşağıya kaydırılmalı ve merkeze göre 60 derece rotasyon uygulanarak 6 defa oluşturulmalı



Motifi Oluşturmak

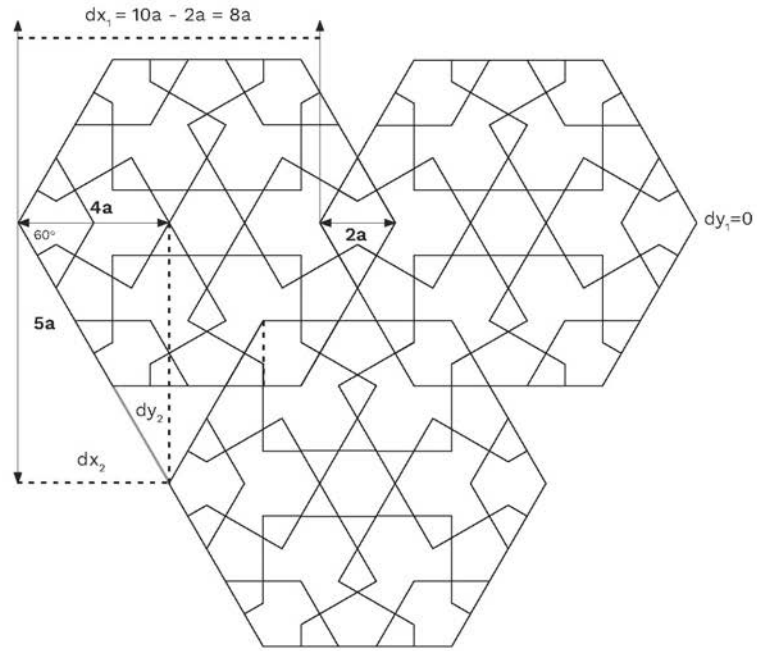
```
let a = 40;

function setup() {
        createCanvas(600, 600);
        noFill();
        angleMode(DEGREES);
}
function draw() {
        let x0,y0,x1,y1,x2,y2,x3,y3,x4,y4,x5,y5,x6,y6,x7,y7,x8,y8,x9,y9,x10,y10;
        background(255);
        push();
        translate(width*0.5, height*0.5);
        x0 = 0;
        y0 = 0;
        x1 = 0.5 * a;
        y1 = 0;
        x2 = 2 * a * cos(60) + 0.5 * a;
        y2 = -(2 * a * sin(60));
        x3 = 1.5 * a;
        y3 = 0;
        x4 = 2.5 * a;
        y4 = 0;
        x5 = 1.5 * a;
        y5 = -a / tan(60);
        x6 = 0;
        y6 = (-2.5 * a) / tan(60);
        x7 = a * sin(30);
        y7 = -(2 * a * sin(60));
        x8 = 0;
        y8 = -(3 * a * sin(60));
        x9 = a * sin(30);
        y9 = -(4 * a * sin(60));
        x10 = 0;
        y10 = -(4 * a * sin(60));

        for (let i = 0; i < 6; i++) {
        push();
        rotate(i * 60);
        translate(0, 5 * a * sin(60));
        beginShape();
        vertex(x0, y0);
        vertex(x1, y1);
        vertex(x2, y2);
        endShape();

        beginShape();
        vertex(x1, y1);
        vertex(x4, y4);
        endShape();
```



```
        beginShape();
        vertex(x3, y3);
        vertex(x5, y5);
        vertex(x6, y6);
        vertex(x7, y7);
        vertex(x8, y8);
        vertex(x9, y9);
        vertex(x10, y10);
        endShape();

         //mirror on y axis
        beginShape();
        vertex(-x0, y0);
        vertex(-x1, y1);
        vertex(-x2, y2);
        endShape();

        beginShape();
        vertex(-x1, y1);
        vertex(-x4, y4);
        endShape();

        beginShape();
        vertex(-x3, y3);
        vertex(-x5, y5);
        vertex(-x6, y6);
        vertex(-x7, y7);
        vertex(-x8, y8);
        vertex(-x9, y9);
        vertex(-x10, y10);
        endShape();

        pop();
    }

  pop();
  noLoop();
}
```

Bezeme Yapısını İnceleyelim

Step 3 : Yukarı ve aşağıya kaymaları belirleyecek xoffset ve yoffset değerlerini hesaplayalım.



$dx_1 = 8a$

$dy_1 = 0$

$dx_2 = 4a$

$dy_2 = \tan(60°) \times 4a$

Bezeme Kodu

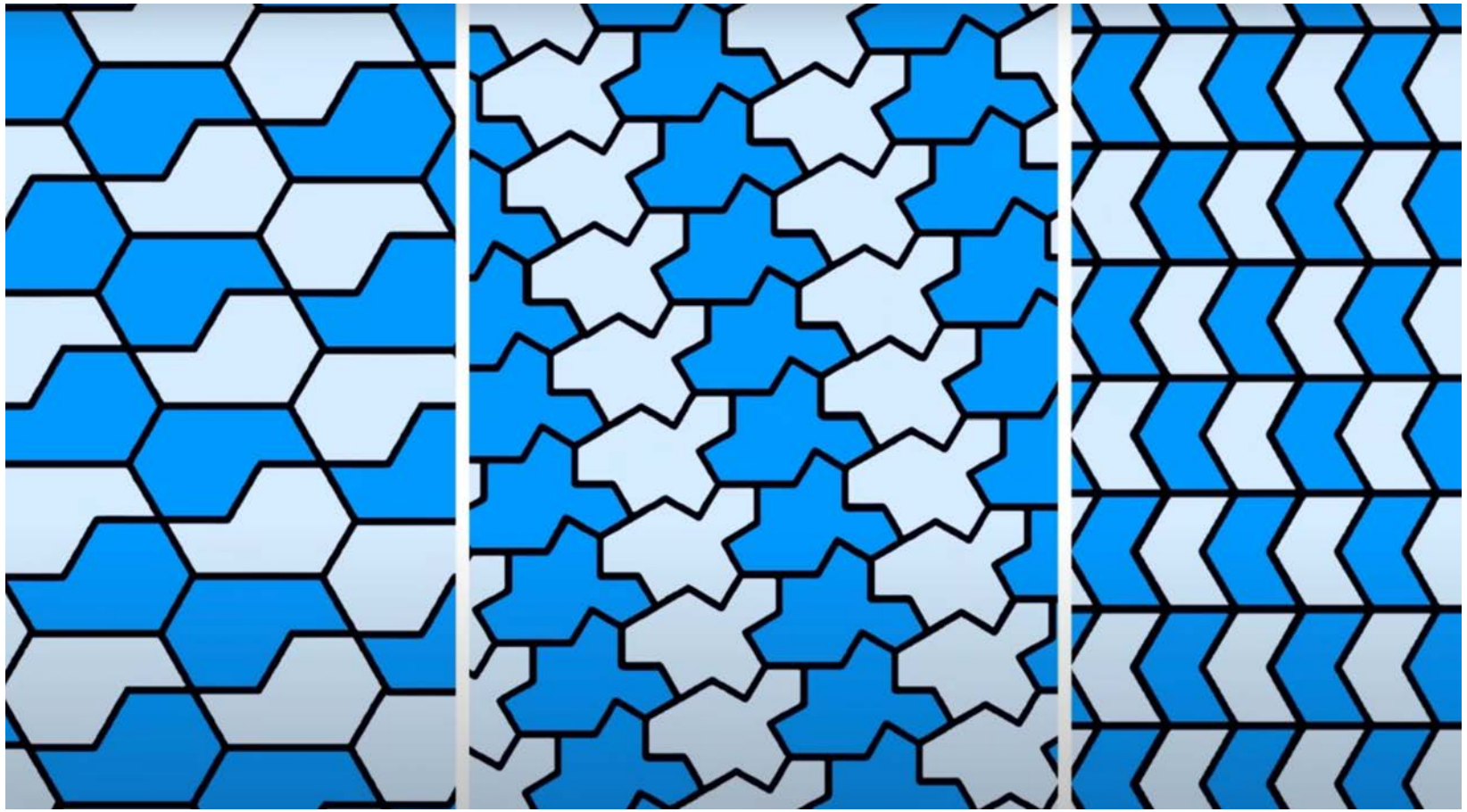
```
/*
Code written by Selcuk ARTUT 2022
Geometric Patterns with Creative Coding
All rights reserved
*/
// Tile class
class Tile {
  constructor(r) {
    this.a = r;
  }

  display() {

    let x0,y0,x1,y1,x2,y2,x3,y3,x4,y4,x5,y5,x6,y6,x7,y7,x8,y8,x9,y9,x10,y10;
    x0 = 0;
    y0 = 0;
    x1 = 0.5 * this.a;
    y1 = 0;
    x2 = 2 * this.a * cos(60) + 0.5 * this.a;
    y2 = -(2 * this.a * sin(60));
    x3 = 1.5 * this.a;
    y3 = 0;
    x4 = 2.5 * this.a;
    y4 = 0;
    x5 = 1.5 * this.a;
    y5 = -this.a / tan(60);
    x6 = 0;
    y6 = (-2.5 * this.a) / tan(60);
    x7 = this.a * sin(30);
    y7 = -(2 * this.a * sin(60));
    x8 = 0;
    y8 = -(3 * this.a * sin(60));
    x9 = this.a * sin(30);
    y9 = -(4 * this.a * sin(60));
    x10 = 0;
    y10 = -(4 * this.a * sin(60));
    for (let i = 0; i < 6; i++) {
      push();
      rotate(i * 60);
      translate(0, 5 * this.a * sin(60));

      beginShape();
      vertex(x0, y0);
      vertex(x1, y1);
      vertex(x2, y2);
      endShape();

      beginShape();
      vertex(x1, y1);
```



```
      vertex(x4, y4);
      endShape();

      beginShape();
      vertex(x3, y3);
      vertex(x5, y5);
      vertex(x6, y6);
      vertex(x7, y7);
      vertex(x8, y8);
      vertex(x9, y9);
      vertex(x10, y10);
      endShape();

      //mirror on y axis
      beginShape();
      vertex(-x0, y0);
      vertex(-x1, y1);
      vertex(-x2, y2);
      endShape();

      beginShape();
      vertex(-x1, y1);
      vertex(-x4, y4);
      endShape();

      beginShape();
      vertex(-x3, y3);
      vertex(-x5, y5);
      vertex(-x6, y6);
      vertex(-x7, y7);
      vertex(-x8, y8);
      vertex(-x9, y9);
      vertex(-x10, y10);
      endShape();

      pop();
    }
  }
}

let tiles = []; // Declare array
let nRow;
let nCol;
let dx1, dy1, dx2, dy2;

let r = 16;
```

```
function setup() {
  createCanvas(1080, 1080);
  angleMode(DEGREES);
  noFill();
  strokeWeight(1);

  nRow = floor(height / (2*r));
  nCol = floor(width / (2*r));

  for (let i = 0; i < nRow * nCol; i++) {
    tiles.push(new Tile(r));
  }

  dx1 = 8.0*r;
  dy1 = 0;
  dx2 = 4.0*r;
  dy2 = 4.0*r*tan(60);

}

function draw() {
  background(255);
  stroke(0);
  for (let r = 0; r < nRow; r++) {
    for (let c = 0; c < nCol; c++) {
      push();
      translate(c * dx1, dy1 + dy2*r);
      if (r % 2 == 1) {
        //rows 1,3,5,7
        translate(dx2,0);
      }
      tiles[r + c * nRow].display();
      pop();
    }
  }
}
```

How a Hobbyist Solved a 50-Year-Old Math Problem (Einstein Tile)



https://www.youtube.com/watch?v=A1BhOVW8qZU

Teşekkürler.
İletişimde kalalım!

Facebook/selcuk.artut
Instagram: selcukartut
Web: www.selcukartut.com
Email: selcukartut@gmail.com