Πληροφορική

Assignment 03

Πρόβλημα 1: https://editor.p5js.org/kouloumi.zoe02/sketches/u2utwCvAe

Αρχικα δημιουργησα ένα κενο πολυδιαστατο array το οποοιο υα περιεχει τα στοιχεια κάθε κελιου και υποκελιου. Στη συναρτηση setup δημιουργησα ένα καμβα 700*600 πιξελς και δηλωσα ότι τα σχηματα που θα δημιουργησω ακολουθως δεν θα εχουν περιγραμμα. Στη συνεχεια ορισα τις μεταβλητες rows και cols που είναι ισες με 10. Δηλαδη ετσι δημιουργησα ένα πινακα 10*10. Μετα όρισα τις μεταβλητες: cellWidth = width / cols κσι cellHeight = height / rows. Με αυτό τον τροπο υπολογιζω τις διαστασεις κάθε βασικου κελιου που θα είναι 70*60 πιξελς.

Συνεχιζοντας ηθελα να φτιαξω το βασικο πλεγμα – grid με τα υποκελια. Ετσι με την μεταβλητη i και με την εντολη: for (let i = 0; i < rows; i++) { grid[i] = []; Δημιουυργησα την γραμμη I στον πινακα. Με τις εντολές for (let j = 0; j < cols; j++) { let subRows = int(random(2, 5)); let subCols = int(random(2, 5)); για κάθε κελι στη στηλη j, επιλέγω τυχαιο αριθμο υποκελιων (subRows) και υποστηλων (subCols), μεταξη 2 και 4.

Ακολούθως δημιουργησα τα υποκελια. Με την μεταβλητη m=0 και την εντολη for (let m = 0; m < subRows; m++) { grid[i][j][m] = []; Που ετσι για κάθε γραμμη υποκελιων δημιουργω το array m. Με την εντολή: for (let n = 0; n < subCols; n++) { let x = j * cellWidth + n * (cellWidth / subCols); let y = i * cellHeight + m * (cellHeight / subRows); Για κάθε στηλη υποκελιων υπολογιζω τη θεση x,y του υποκελιου πάνω στον καμβα. Με την εντολη: let w = cellWidth / subCols; let h = cellHeight / subRows; υπολογιζω το πλατος και το υψος του κάθε υποκελιου και με τις εντολες: let c = color(random(0, 250), random(0, 100), 250); let v = color(255, random(0,255),random(0,250)); Δημιουργω δυο τυχαια χρωματα. Το c είναι τυχαιο χρώμα τις αποχρωσης μπλε-μωβ και το v είναι τυχαιο χρωμμα τις αποχρωσης του κοκκινου.

Για τον σχεδιασμο του καμβα αρχικα ‘’αποθηκευσα‘’ τις πληροφοριες του κελιου στο array. Μετά σχεδιασα ορθογωνια με το χρωμα c και κυκλους στο ιδιο σημειο με το χρωμα v.

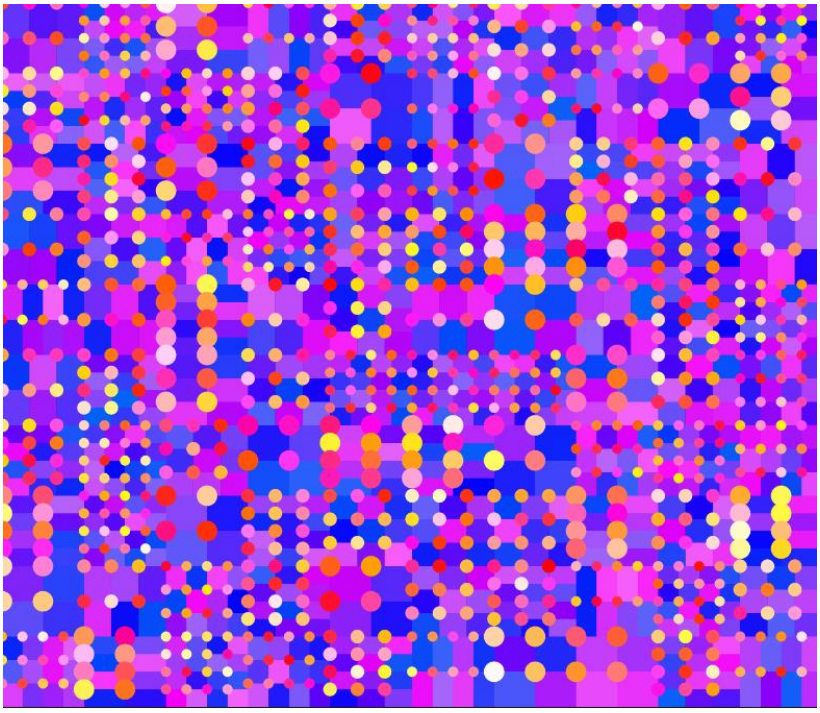
Όσων αφορά την συναρητηση mousePressed, με τη βοηθεια του chatgpt ελεγξα κάθε υποκελι και πειρα τις πληροφοριες του κάθε υποκελιου. Με την εντολη if (mouseX > x && mouseX < x + w && mouseY > y && mouseY < y + h) { ελεγχει αν το ποντικι ειαι μεσα στο υποκελι. Με την εντολή: let newColor = color(random(0, 250), 200, random(0, 250)); //

Ζωή Κουλουμή

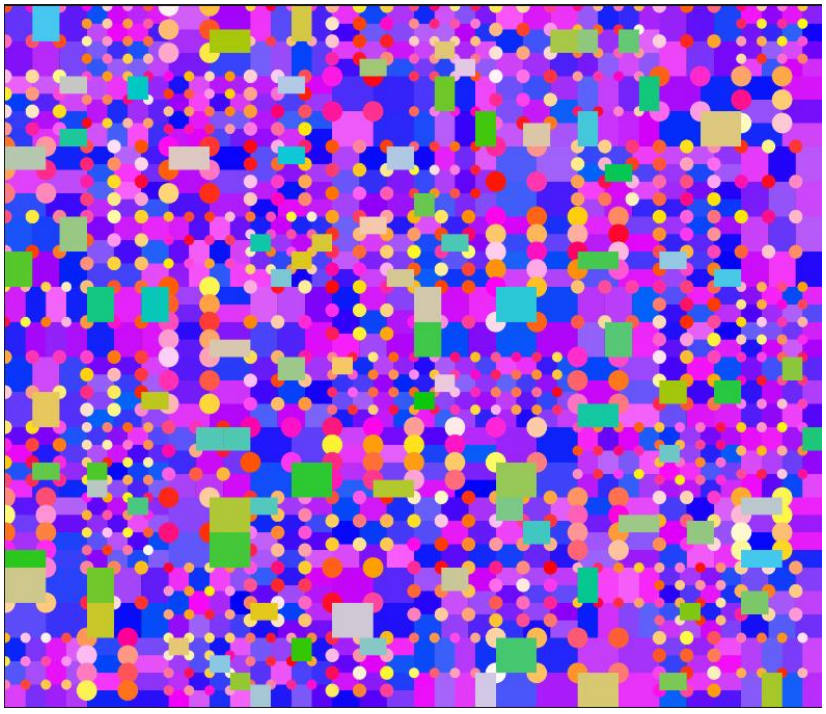
grid[i][j][m][n][4] = newColor; fill(newColor) rect(x, y, w, h); δημιουργω ένα νέο τυχαιο χρωμα και και ξανα χρωματιζει το υποκελι.

```
let grid = [];
// array που εχει ολες τι πληροφοριες για τον καναβο.

function setup() {
  createCanvas(700, 600);
  noStroke();
  // δημιουργια καμβα και οι γραμμες δεν θα εχουν περιγραμμα

  let rows = 10; // γραμμη του βασικου καναβου
  let cols = 10; // στηλη βασικου καναβου
  let cellWidth = width / cols; // πλατος κελιου
  let cellHeight = height / rows; // υψος κελιου

  for (let i = 0; i < rows; i++) {
    grid[i] = [];
    for (let j = 0; j < cols; j++) {
      let subRows = int(random(2, 5));
      let subCols = int(random(2, 5));
      grid[i][j] = [];

      for (let m = 0; m < subRows; m++) {
        grid[i][j][m] = [];
        for (let n = 0; n < subCols; n++) {

          let x = j * cellWidth + n * (cellWidth / subCols); // οριζοντια θεση υποκελιου
          let y = i * cellHeight + m * (cellHeight / subRows); // καθετη θεση υποκελιου
          let w = cellWidth / subCols; // πλατος υποκελιου
          let h = cellHeight / subRows; // υψος υποκελιου
          let c = color(random(0, 250), random(0, 100), 250); // τυχαιο μπλε-μοβ χρωμα
          let v = color(255, random(0,255),random(0,250));

          grid[i][j][m][n] = [x, y, w, h, c];

          // Σχεδιάζω το υποκελί
          fill(c);
          rect(x, y, w, h);
          fill(v);
          circle(x,y,w/2)
        }
      }
    }
  }
}

function mousePressed() {

  for (let i = 0; i < grid.length; i++) {
    for (let j = 0; j < grid[i].length; j++) {
      for (let m = 0; m < grid[i][j].length; m++) {
        for (let n = 0; n < grid[i][j][m].length; n++) {
          let cell = grid[i][j][m][n];
          let [x, y, w, h, c] = cell;

          if (mouseX > x && mouseX < x + w && mouseY > y && mouseY < y + h) {
            let newColor = color(random(0, 250), 200, random(0, 250)); // νέο κόκκινο χρώμα
            fill(newColor);
            rect(x, y, w, h);

          }
        }
      }
    }
  }
}
```

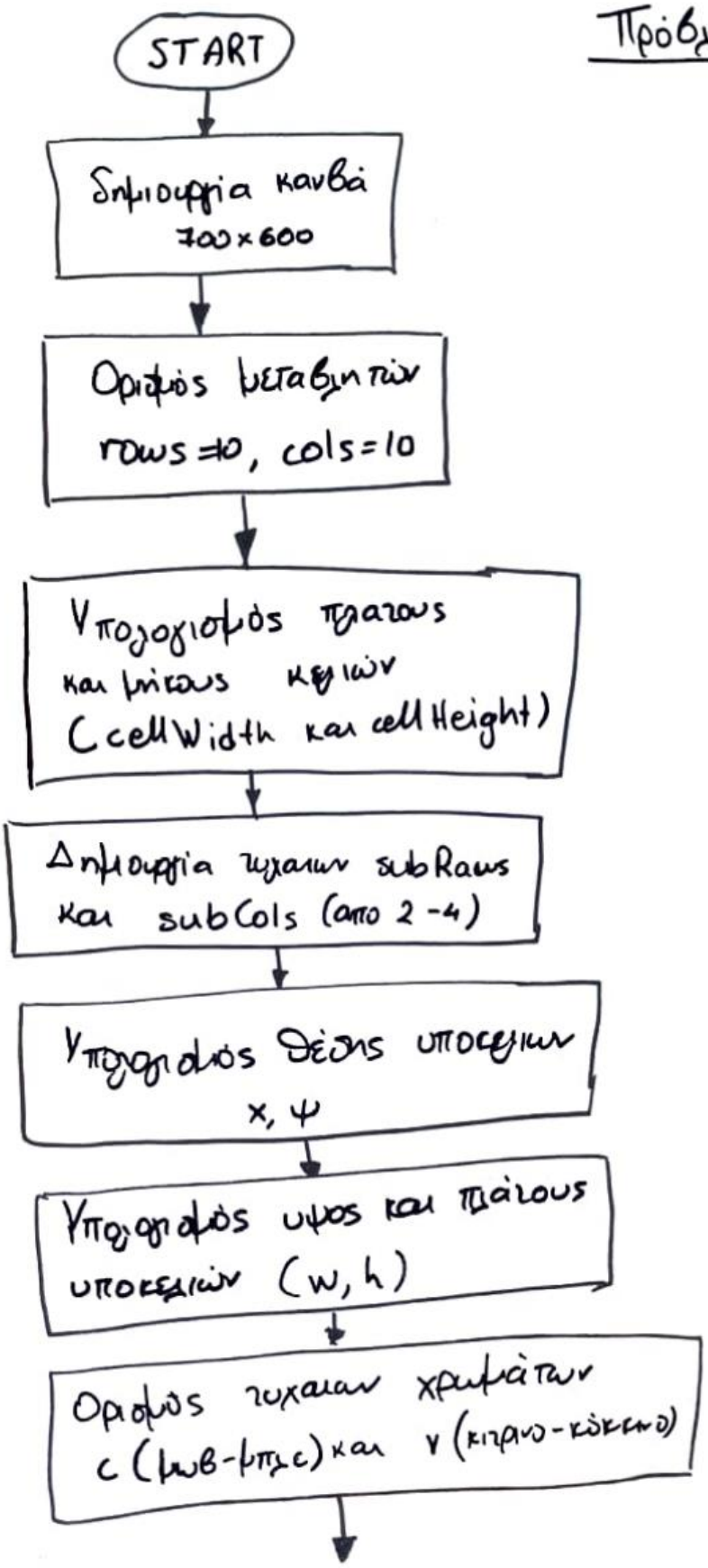
Ζωή Κουλουμή
Assignment 3 Πληροφορική

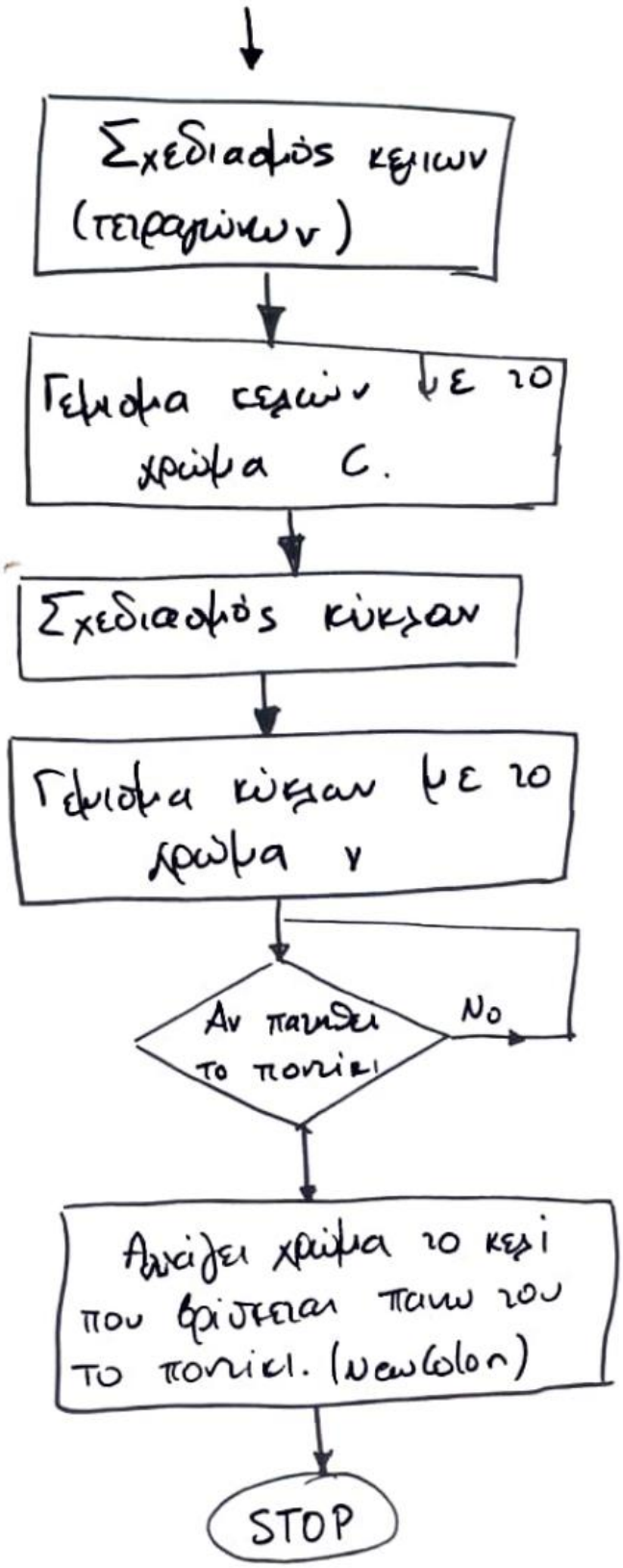


Mouse
Pressed



Πρόβλημα 1


START
Δημιουργία καμβά 700x600
Ορισμός μεταβλητών rows=10, cols=10
Υπολογισμός πλάτους και μήκους κελιών (cellWidth και cellHeight)
Δημιουργία τυχαίων subRows και subCols (από 2-4)
Υπολογισμός θέσης υποκελιών x, y
Υπολογισμός ύψος και πλάτους υποκελιών (w, h)
Ορισμός τυχαίων χρωμάτων c (μωβ-μπλε) και γ (κίτρινο-κόκκινο)


Σχεδιασμός κελιών (τετραγώνων)
Γέμισμα κελιών με το χρώμα C.
Σχεδιασμός κύκλων
Γέμισμα κύκλων με το χρώμα Y
Αν πατήσει το ποντίκι
No
Αλλάζει χρώμα το κελί που βρίσκεται πάνω του το ποντίκι. (newColor)
STOP

Ζωή Κουλουμή
Assignment 3 Πληροφορική

Προβλημα 2: https://editor.p5js.org/kouloumi.zoe02/sketches/bdhQlXQNB

Αυτός κώδικας δημιουργεί ένα ιεραρχικό, συμμετρικό μοτίβο από τετράγωνα, όπου κάθε επίπεδο περιέχει μικρότερα τετράγωνα γύρω από το κέντρο του προηγούμενου.

Αρχικα δημιουρχησα το setup function που δημιουργει καμβα 600*600 πιξελς, ορισα μαυρο background και δηλωσα ότι τα τετραγωνα θα σχεδιαζονται με σημειο αναφορας το κεντρο τους. Δηλωσα τρεις μεταβλητες αρχικα, το x,y που είναι το κεντρο του καβμα και το originalsize που είναι το αρχικο μεγεθος του μεγαλου τετραγωνου.

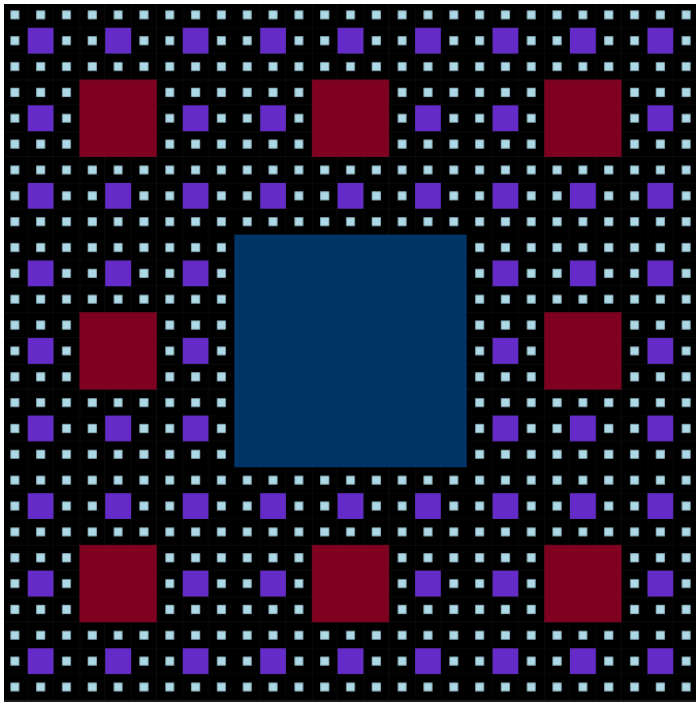
Δημιουργια πρωτου επιπεδου: με τις εντολες: fill(0); noStroke(); rect(X, Y, originalSize, originalSize); Σχεδιασα ένα μεγαλο μαυρο τετραγωνο στο κεντρο του καμβα. Με τις εντολες fill(0, 51, 102); rect(X, Y, originalSize / 3, originalSize / 3); σχεδιασα ένα μπλε σκουρο τετραγωνο στο κεντρ του μαυρου τετραγωνου.

Δημιουργεια δευτερου επιπεδου: εχω ορισει ακομη 2 μεταβλητες που οριζουν την αποσταση του κεντρου του κάθε νεου τετραγωνου από το κεντρο του καβμα και το μεγεθος των νεων τετραγωνων. Με τις εντολες: for (let difX = -1; difX <= 1; difX++) { for (let difY = -1; difY <= 1; difY++) { if (difX == 0 && difY == 0) continue; φτιαχνω 8 θεσεις γυρο από το κεντρο. Στη συνεχεια οριζω τις μεταβλητες let newX = X + difX * centerDist1; let newY = Y + difY * centerDist1; Που ετσι υπολογιζω τις θεσεις των νεων τετραγωνων γυρω από το κεντρο. Μετα, δημιουργω ένα μικροτερο τετραγωνο γυρο παοτο κεντρο αυτό χρωματος μπρορντο.

Δημιουργεια 3^{ου} επιπεδου: εχω βρει το την απαιτουμενη αποσταση και μεγεθος για τα νεα τετραγωνα που θα είναι γυρω από κάθε μπορντο τεταγωνο. Στη συνεχεια αντίστοιχα δημιουργησα 8 σημεια γυρο από το κεντρικο μπορντο τετραγωνο, χρωματος μωβ.

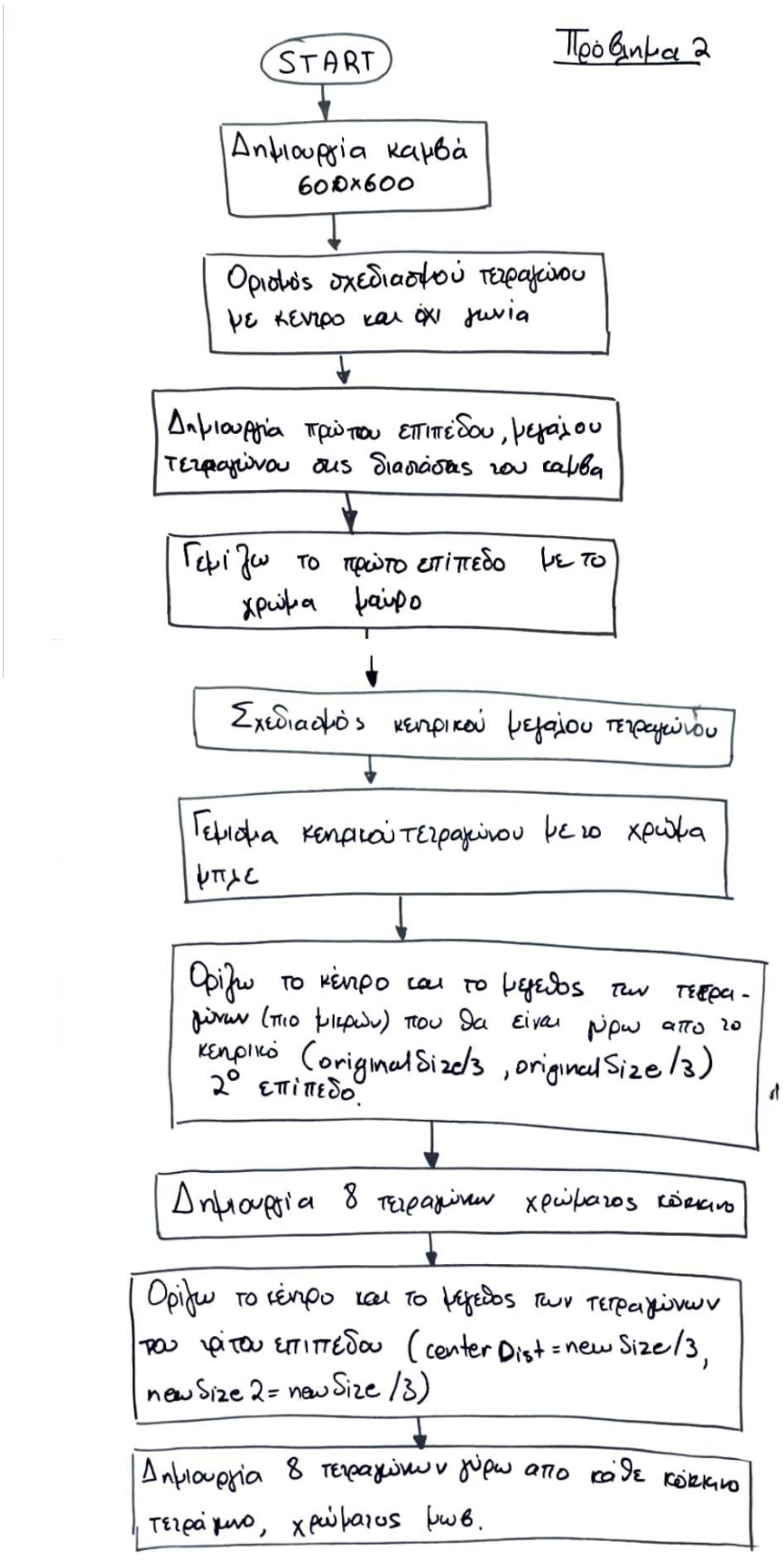
Δημιουργια 4^{ου} επιπεδου: βρισκω το νέο κεντρο και το νέο μεγεθος του ακομα πιο μικρου τετραγωνου. Ξανα υπολογίζω τα 8 σημεια που θα είναι γυρο από τα μωβ τετραγωνα. Δημιουργω τα 8 τετραγωνα γυρο από τα μωβ τετραγωνα χρωματος γαλαζιο.

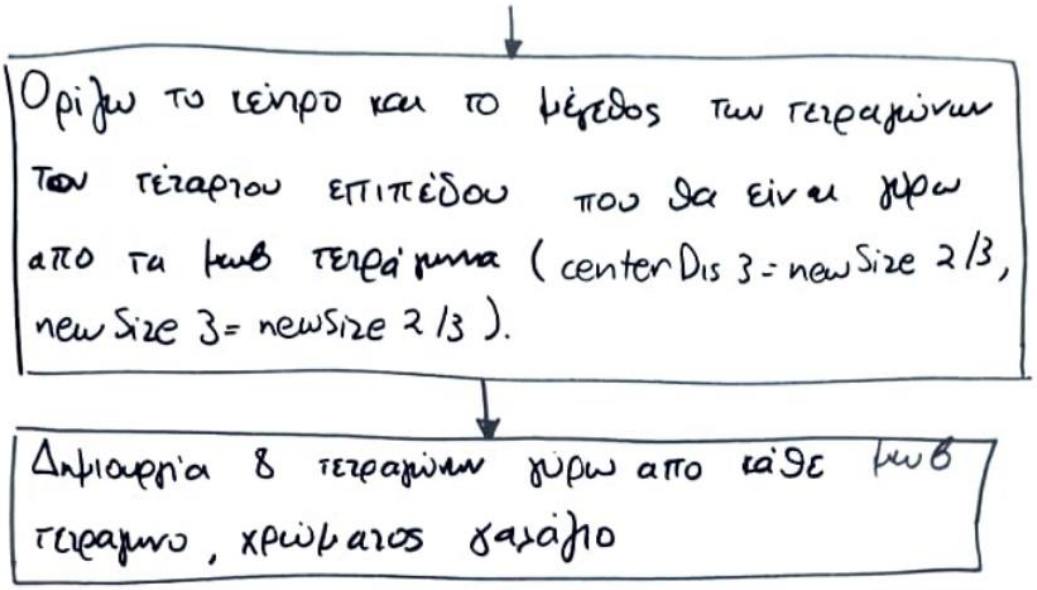
Ζωή Κουλουμή
Assignment 3 Πληροφορική



```
function setup() {
  createCanvas(600, 600);
  background(0);
  rectMode(CENTER);//κεντρο τετραγωνου

  let X = width / 2;
  let Y = height / 2;
  let originalSize = width;

  // Πρώτο επίπεδο (μεγάλο τετράγωνο)
  fill(0);
  noStroke();
  rect(X, Y, originalSize, originalSize);

  // Κεντρικό κενό
fill(0, 51, 102);
  rect(X, Y, originalSize / 3, originalSize / 3);

  // Δεύτερο επίπεδο (8 γύρω τετράγωνα)
  let centerDist1 = originalSize / 3;
  let newSize = originalSize / 3;

  for (let difX = -1; difX <= 1; difX++) {
    for (let difY = -1; difY <= 1; difY++) {
      if (difX == 0 && difY == 0) {
        continue;
      }
      let newX = X + difX * centerDist1;
      let newY = Y + difY * centerDist1;

```

```
31
32        fill(0);
33        rect(newX, newY, newSize, newSize);
34
35        fill(128, 0, 32);
36        rect(newX, newY, newSize / 3, newSize / 3);
37
38        // Τρίτο επίπεδο (μικρότερα γύρω από κάθε μικρό τετράγωνο)
39        let centerDist2 = newSize / 3;
40        let newSize2 = newSize / 3;
41
42        for (let difX2 = -1; difX2 <= 1; difX2++) {
43          for (let difY2 = -1; difY2 <= 1; difY2++) {
44            if (difX2 == 0 && difY2 == 0) {
45              continue;
46            }
47            let newX2 = newX + difX2 * centerDist2;
48            let newY2 = newY + difY2 * centerDist2;
49
50            fill(150);
51            rect(newX2, newY2, newSize2, newSize2);
52
53            fill(100, 43, 200);
54            rect(newX2, newY2, newSize2 / 3, newSize2 / 3);
55
56            // Τέταρτο επίπεδο
57            let centerDist3 = newSize2 / 3;
58            let newSize3 = newSize2 / 3;
59
59
60             for (let difX3 = -1; difX3 <= 1; difX3++) {
61               for (let difY3 = -1; difY3 <= 1; difY3++) {
62                 if (difX3 == 0 && difY3 == 0) {
63                   continue;
64                 }
65                 let newX3 = newX2 + difX3 * centerDist3;
66                 let newY3 = newY2 + difY3 * centerDist3;
67
68                 fill(0);
69                 rect(newX3, newY3, newSize3, newSize3);
70
71                 fill(173, 216, 230);
72                 rect(newX3, newY3, newSize3 / 3, newSize3 / 3);
73
74
75               }
76             }
77           }
78         }
79       }
80     }
81   }
```


Πρόβλημα 2
START
Δημιουργία καμβά 600x600
Ορισμός σχεδιασμού τετραγώνου με κέντρο και όχι γωνία
Δημιουργία πρώτου επιπέδου, μεγάλου τετραγώνου σαν διαστάσεις του καμβά
Γεμίζω το πρώτο επίπεδο με το χρώμα μαύρο
Σχεδιασμός κεντρικού μεγάλου τετραγώνου
Γεμίσμα κεντρικού τετραγώνου με το χρώμα μπλε
Ορίζω το κέντρο και το μέγεθος των τετρα-γώνων (πιο μικρών) που θα είναι γύρω από το κεντρικό (originalSize/3, originalSize/3) 2ο επίπεδο.
Δημιουργία 8 τετραγώνων χρώματος κόκκινο
Ορίζω το κέντρο και το μέγεθος των τετραγώνων του τρίτου επιπέδου (centerDist = newSize/3, newSize2 = newSize/3)
Δημιουργία 8 τετραγώνων γύρω από κάθε κόκκινο τετράγωνο, χρώματος λευκό.


Ορίζω το κέντρο και το μέγεθος των τετραγώνων του τέταρτου επιπέδου που θα είναι μέσα απο τα δυο τετράγωνα (centerDis 3 = newSize 2 /3, newSize 3 = newSize 2 /3).
Δημιουργία 8 τετραγώνων γύρω απο κάθε δυο τετραγωνο, χρώματος γαλάζιο

Μέρος 3: https://editor.p5js.org/kouloumi.zoe02/sketches/c-_v4oI2z

Αρχικά δηλωσα τις μεταβλητες που χρειαζόμουν:

- let snakeHead; -> Η θέση του κεφαλιού του φιδιού
- let stepSize = 9; -> Πόσα pixels κινείται το φίδι κάθε φορά
- let fwd_x = 0; -> Ταχύτητα x
- let fwd_y = 0; -> Ταχύτητα y
- let snakeBody = []; -> Πίνακας με όλα τα τμήματα του σώματος του φιδιού
- let snakeLength = 1; -> Πόσα "κομμάτια" έχει το φίδι (ξεκινάει με 1)
- let food = []; -> Πίνακας με όλα τα αντικείμενα φαγητού
- let totalFood = 450; -> Αριθμός τροφών που θα δημιουργηθούν αρχικά

Στη συνέχεια στη συναρτηση setup δημιουργω καμβα ισο με το παραθυρο του browser και οριχω την αρχικη θεση του κεφαλιου του φιδιου στο κεντρο της οθονης. Με την εντολή:

```
 for (let i = 0; i < totalFood; i++) {
  let foodX = random(width);
  let foodY = random(height);
  food.push(createVector(foodX, foodY));
 }
}
```

Δημιουργησα 450 αντικείμενα ΄φαγητου΄ σε τυχαιες θέσεις και τα δηλωσα ςω vector στον πινακα food.

Επισης στην συναρτρηση draw, χρωματίζω το background με μαυρο χρώμα και ζωγραφίζω την κάθε τροφή ως ένα γαλάζιο κυκλάκι στις αντίστοιχες θέσεις.

```
 for (let i = food.length - 1; i >= 0; i--) {
  if (dist(snakeHead.x, snakeHead.y, food[i].x, food[i].y) < stepSize) {
   snakeLength++;
   food.splice(i, 1);
  }
 }
```

Με την παραπάνω εντολή ελέγχω αν η απόσταση από το κεφάλη του φιδιου μέχρι κάποια τροφή είναι μικρότερη από το stepSize. Αν ναι, τότε αυξάνεται το μήκος του φιδιού και αφαιρείται η συγκεκριμένη τροφή από τον ππινακα food.

```
 if (snakeHead.x < 0) snakeHead.x = width;

 if (snakeHead.x > width) snakeHead.x = 0;

 if (snakeHead.y < 0) snakeHead.y = height;

 if (snakeHead.y > height) snakeHead.y = 0;
```

με την πιο πάνω εντολή αν το φίδι βγει εκτός οθόνης μεταφέρετε από την αντίθετη πλευρά και συνεχίζει την κιμησή του (βοηθεια από chat gpt για την συγκεκριμενη εντολή).

```
 fill(150);

textSize(20);

text("Score: " + (snakeLength - 1), 10, 20);
```

Με την πιο πάνω εντολή εμφανίζει στην οθόνη το σκορ, δηλαδη πόσα κομματια τροφής έφαγε το φίδι.

Μετά δηλωνω την αρχική θέση του κεφαλιου του φιδιου στον πίνακα. Αν το μήκος του πίνακα είναι μεγαλύτερο από το snakeLength, τότε αφαιρείαι το τελευταίο τμήμα. Έτσι το σώμα του φιδιού κινείται ‘τραβώντας’ το πίσω μέρος καθώς προστίθεται το νέο μπροστινό. Ακόμη, σχεδιασα το σώμα του φιδιού με ρόζ τετραγωνακια και ασπρο περίγραμμα, για κάθε σημείο που έχει αποθηκευτεί στον πίνακα snakeBody.

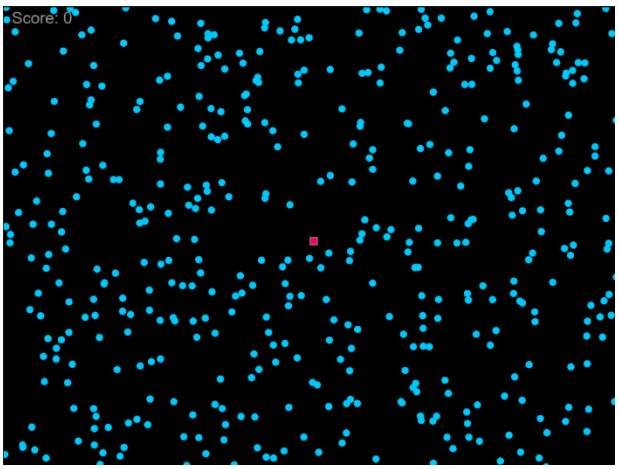
Συάστηση keyPressed:

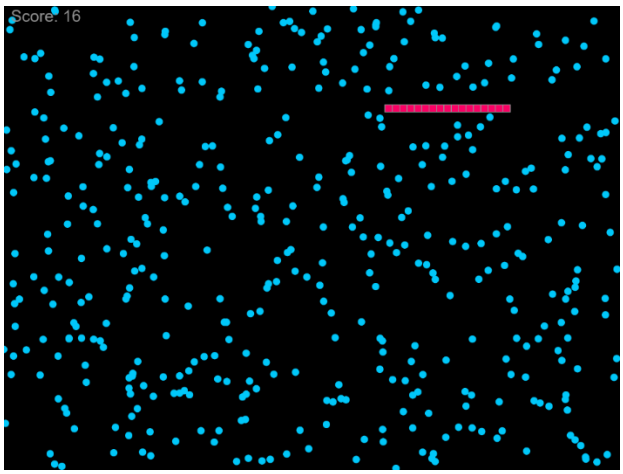
```
function keyPressed() {
 if (keyCode === UP_ARROW) {
   fwd_x = 0;
   fwd_y = -stepSize;
 } else if (keyCode === DOWN_ARROW) {
   fwd_x = 0;
   fwd_y = stepSize;
 } else if (keyCode === LEFT_ARROW) {
   fwd_x = -stepSize;
```

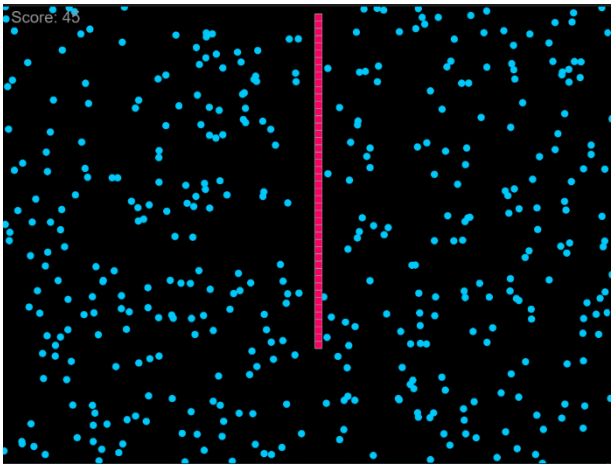
```
    fwd_y = 0;
  } else if (keyCode === RIGHT_ARROW) {
    fwd_x = stepSize;
    fwd_y = 0;
  }   }
```

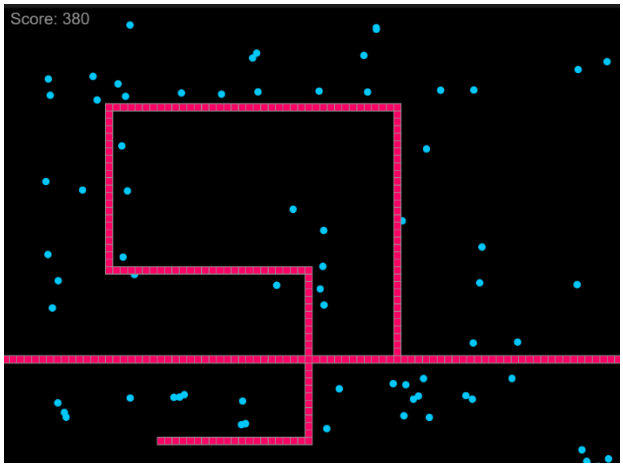
Αυτές οι γραμμές κώδικα δηλώνουν ότι με το πάτημα των κουμπιών (βελάκια) αλλάζει κατεύθηνση το φίδι.

Εικόνες:









```
1  //metavlites
2  let snakeHead;
3  let stepSize = 9;
4  let fwd_x = 0;
5  let fwd_y = 0;
6  let snakeBody = [];
7  let snakeLength = 1;
8  let food = [];
9  let totalFood = 450;
10
11 function setup() {
12   createCanvas(windowWidth, windowHeight);
13   snakeHead = createVector(width / 2, height / 2);
14   snakeBody.push(createVector(snakeHead.x, snakeHead.y));
15
16   // Δημιουργία τροφής
17   for (let i = 0; i < totalFood; i++) {
18     let foodX = random(width);
19     let foodY = random(height);
20     food.push(createVector(foodX, foodY));
21   }
22 }
23
24 // ----- DRAW FUNCTION -----
25 function draw() {
26   background(0);
27
```

```
28   // Σχεδίαση τροφής
29   fill(0, 200,250);
30   noStroke();
31   for (let i = 0; i < food.length; i++) {
32     ellipse(food[i].x, food[i].y, stepSize, stepSize);
33   }
34
35   // Έλεγχος αν τρώμε τροφή
36   for (let i = food.length - 1; i >= 0; i--) {
37     if (dist(snakeHead.x, snakeHead.y, food[i].x, food[i].y) < stepSize) {
38       snakeLength++;
39       food.splice(i, 1); // αφαιρούμε τη φαγητό που φάγαμε
40     }
41   }
42
43 //   // Ενημέρωση θέσης φιδιού
44   snakeHead.x += fwd_x;
45   snakeHead.y += fwd_y;
46
47   fill(150);
48 textSize(20);
49 text("Score: " + (snakeLength - 1), 10, 20);
50
```

```
51
52    if (snakeHead.x < 0) snakeHead.x = width;
53    if (snakeHead.x > width) snakeHead.x = 0;
54    if (snakeHead.y < 0) snakeHead.y = height;
55    if (snakeHead.y > height) snakeHead.y = 0;
56
57
58    snakeBody.unshift(createVector(snakeHead.x, snakeHead.y));
59    if (snakeBody.length > snakeLength) {
60      snakeBody.pop();
61    }
62
63    // Σχεδίαση σώματος φιδιού
64    fill(250,0,100);
65    stroke(150);
66    for (let i = 0; i < snakeBody.length; i++) {
67      rect(snakeBody[i].x, snakeBody[i].y, stepSize, stepSize);
68    }
69  }
```

```
70
71  function keyPressed() {
72    if (keyCode === UP_ARROW) {
73      fwd_x = 0;
74      fwd_y = -stepSize;
75    } else if (keyCode === DOWN_ARROW) {
76      fwd_x = 0;
77      fwd_y = stepSize;
78    } else if (keyCode === LEFT_ARROW) {
79      fwd_x = -stepSize;
80      fwd_y = 0;
81    } else if (keyCode === RIGHT_ARROW) {
82      fwd_x = stepSize;
83      fwd_y = 0;
84    }
85  }
86
```