

Lütfen ad-soyad, öğrenci numarası ve imzanızı yazınız.

Ad Soyad: _____ Öğrenci No: _____
İmza: _____

1. (10 points) Aşağıdaki kelimelerin Türkçe karşılıklarını yazınız.

- (a) background
- (b) line
- (c) fill
- (d) variable
- (e) circle

2. (15 points) Her bir ifadenin hangi renkte boyama yapacağını yazınız.

- (a) background(0);
- (b) background(255);
- (c) fill(50);
- (d) fill(255,0,0);
- (e) fill(0,0,255);

3. (15 points) Aşağıdaki p5.js programının ne yaptığını ve ekranda nasıl bir çıktı ürettiğini açık ve anlaşılır biçimde açıklayınız. Programın üreteceği görselin hangi renk veya renklerde görüneceğini açıklayınız.

```
function setup() {  
  createCanvas(256, 100);  
  for(let x = 0; x < 256; x++){  
    stroke(x,0,0);  
    line(x,0,x,100);  
  }  
}  
  
function draw() {  
}
```

4. (15 points) Aşağıdaki p5.js programının çizeceği şekli gösteriniz. Çizdiğiniz şekilde hangi bölgenin ne renkte gözükeceğini de mutlaka gösteriniz.

```
function setup() {  
  createCanvas(600, 600);  
  for (let i = 0; i < 5; i++){  
    for (let j = 0; j < 5; j++){  
      if ((i + j) % 4 == 0){  
        fill(255,0,0);  
      }  
      else if ((i + j) % 4 == 1){  
        fill(0,255,0);  
      }  
      else if ((i + j) % 4 == 2){  
        fill(0,0,255);  
      }  
      else{  
        fill(255);  
      }  
      rect(i * 120, j * 120, 120, 120);  
    }  
  }  
}  
  
function draw() {  
}
```

5. (15 points) Aşağıda size verilen p5.js kodunda herhangi bir satırı silmeden sadece kodda eklemeler yaparak kodun şu şekilde çalışmasını sağlayınız:

- Çizdirilen çemberin merkezinin x koordinatı ($200 \leq x \leq 300$) ve y koordinatı ($200 \leq y \leq 300$) ise çemberin içini maviye boyayınız.
- Çizdirilen çemberin merkezinin x koordinatı ($150 \leq x \leq 220$) ve y koordinatı ($50 \leq y \leq 100$) ise çemberin içini yeşile boyayınız.
- Tüm diğer çemberlerin içini kırmızıya boyayınız.
- Kodun çizdirdiği çemberler çakışabilir ve bu nedenle sonradan çizdirilen bir çember önceki çemberi kısmen kapayabilir. Bu gibi detay durumlar üzerinde düşünmeyiniz.

```
function setup() {  
  createCanvas(600, 400);  
  for(let i = 0; i < 10; i++){  
    let x = random(50, 500);  
    let y = random(50, 350);  
    ellipse(x,y,100,100);  
  }  
}  
  
function draw() {  
}
```

6. (20 points) Canvas'ta her yeni frame'de aşağıda belirtilen kurallara uygun rastgele yeni bir karenin çizileceği kodu yazınız.

- Her yeni frame'de program canvasa rastgele yeni bir kare ekleyecek.
- Kare alanı 100–400 piksel² aralığında olmalıdır.
- Renk kuralı: $Alan \leq 200$ ise kare kırmızı; $Alan > 200$ ise kare maviye boyanmalıdır.
- Çizdirilen karelerin tamamı canvas içinde görünmelidir. Kareler kısmen de olsa canvas dışına taşmamalıdır.
- Kareler çakışabilir ve bu nedenle sonradan çizdirilen bir kare önceki kareyi kısmen kapayabilir. Bu gibi durumlar üzerinde düşünmeyiniz.
- Örneğin 10 frame sonra canvasta alanı 100 piksel² ile 400 piksel² arasında değişen 10 kare görünecek. Bu karelerden alanı 200 piksel²'den büyük olan kareler mavi, diğerleri kırmızı renkte boyalı olacak.

7. (20 points) Aşağıdaki p5.js kodu ile **aynı çıktıyı** üretecek bir program yazınız.

- Bu soru için nesne (object) kullanmayınız; bunun yerine birden fazla **dizi** kullanınız.
- Aşağıdaki bilgiler için ayrı diziler tanımlayınız: *x koordinatları*, *y koordinatları*, *kare kenar uzunlukları*, *renkler*.
- Tek bir **döngü** ile bu dizilerden yararlanarak kareleri çizdiriniz.

```
function setup() {
  createCanvas(600,600);
  fill('blue');
  rect(50,50,20,20);
  fill('red');
  rect(100,80,40,40);
  fill('yellow');
  rect(200,250,60,60);
  fill('orange');
  rect(400,100, 50,50);
  fill('purple');
  rect(500,500,10,10);
}
```

```
function draw() {
}
```

Yönerge:

- *x*, *y*, boyut ve renk bilgilerini ayrı dizilerde saklayınız.
- Tek bir for döngüsü ile `fill(renk[i])` ve `rect(x[i], y[i], boyut[i], boyut[i])` mantığıyla çiziniz.
- Çizimler `setup()` içinde yapılabilir, `draw()` boş kalabilir.

8. (20 points) Yerçekimi simülasyonu — Dünya ve Mars sabit yerçekimi ivmesi g altında hız ve konum güncellemeleri ayrı zamanlarda yaklaşık olarak $v \leftarrow v + g$ ve $y \leftarrow y + v$ biçiminde yapılır. Dünya için $g_{\text{Dünya}} \approx 9.81 \text{ m/s}^2$, Mars için $g_{\text{Mars}} \approx 3.71 \text{ m/s}^2$ olup oran yaklaşık 0.38'dir. Piksel-temelli bir simülasyonda bu değerlerin ölçeklenmiş sürümleri (örn. $g_{\text{Dünya}} = 0.5$, $g_{\text{Mars}} = 0.19 \text{ px/frame}^2$) kullanılabilir.

Örnek (inceleyiniz): Aşağıdaki kod tek bir topu yerçekimi ile düşürür, yere değdiğinde durdurur.

```
let y = 20;
let v = 0;
const r = 20;
const g = 0.5; // Dünya için örnek ölçek

function setup() {
  createCanvas(600, 400);
}

function draw() {
  background(240);
  // hız ve konum güncelle
  v = v + g;
  y = y + v;
  // zemine çarpınca durdur
  if (y + r >= height) {
    y = height - r;
    v = 0;
  }
  // top ve zemin
  noStroke();
  fill(70, 120, 240);
  circle(width * 0.25, y, 2 * r);
  stroke(0);
  line(0, height - 0.5, width, height - 0.5);
}
```

(a) Yeni bir kod yazarak tuvali ikiye bölünüz: solda Dünya, sağda Mars simülasyonu beraber çalışsın.

- Her iki tarafta da top aynı yükseklikten bırakılmalı; her framede hız ve konum Dünya ve Marsta düşen top için ayrı ayrı güncellenmelidir.
- Kodda Mars ve Dünya için ayrı hız, konum ve ivme değişkenleri kullanılmalıdır.
- Solda $g_{\text{Dünya}}$ (örn. 0.5 px/frame^2), sağda $g_{\text{Mars}} \approx 0.38 \cdot g_{\text{Dünya}}$ (örn. 0.19) kullanınız.
- Top zemine değdiğinde durmalıdır (konumu zeminde sabitleyip hızı sıfırlayınız).

(b) Terminal hız (limit hız) kavramını uygulayınız: Dünya tarafında hız üst sınırı $v_{\max} = 20$ px/frame (veya belirlediğiniz başka bir değer) olacak şekilde, hız bu değeri aşmayacak biçimde simülasyonu güncelleyiniz. Mars'ın atmosferi ihmal edilebilir ve Mars tarafında limit hız uygulanmayabilir.