```
1   // Johan Svendsen og Kenneth Johansen
2   // Active PDE program version.
3   // 8-bit Binary to Unsigned integer number
4
5
6   Bit[] bits = new Bit[8];    // array with room for 8 on/off Bit instances
7   int decimal = 0;
8   int binary = 0;
9   PFont font;
10
11  void setup() {
12    size(600, 300);
13    noStroke();
14    font = createFont("Arial", 48, true);    // Windows 10 have a wrong font path
15    for (int i = 0; i < bits.length; i++) {
16      bits[i] = new Bit(i);
17    }  // for
18  }  // setup
19
20  void draw() {
21    background(0);
22    for (int i = 0; i < bits.length; i++) {
23      bits[i].display();
24      fill(255);
25      int bitValue = 1 << (bits.length - i - 1);    // very fast calculation of 2^i
26      text(bitValue, width/9 * bits[i].position - 10, 50);
27    }  // for
28    fill(255);
29    textFont(font, 48);
30    textAlign(RIGHT);
31    text(nf(binary, 8), width/9*8, 180);
32    text(decimal, width/9*8, 230);
33    textAlign(LEFT);
34    text("Binært:", width/9, 180);
35    text("Decimalt:", width/9, 230);
36    textFont(font, 18);
```

```
37      fill(0, 255, 255);  // Cyan text
38      text("Klik en bit for at tænde (og addere værdien 2^n) eller sluk for en bit.", width/22, 25);
39    }  // draw
40
41    void keyReleased() {
42      decimal = 0;
43      binary = 0;
44      for (int i = 0; i < bits.length; i++) {
45        bits[i].updateKey();
46        decimal += bits[i].value;
47        binary += bits[i].digit;
48      }  // for
49    }  // keyReleased
50
51    void mouseReleased() {
52      decimal = 0;
53      binary = 0;
54      for (int i = 0; i < bits.length; i++) {
55        bits[i].updateMouse();
56        decimal += bits[i].value;
57        binary += bits[i].digit;
58      }  // for
59    }  // mouseReleased
60
61    class Bit {             // Bit object class
62      int position;
63      color colour = (55);  // Grey
64      int value = 0;
65      int digit = 0;
66
67      Bit(int pos) {
68        position = pos + 1;
69      }
70
71      void display() {
72        rectMode(CENTER); //siger at rektangelet skal starte fra midten i stedet for øverste venstre hjørne
```

```
    fill(colour);
    rect(width / 9 * position, 80, 50, 50); //rect i stedet for ellipse for at lave en firkant
  }

  void updateKey() {
    if (key == position + 48) {
      switch(colour) {
        case (55):
          colour = (255);
          value = int (pow(2, 8 - position));
          digit = int (pow(10, 8 - position));
          break;
        case (255):
          colour = (55);
          value = 0;
          digit = 0;
          break;
      }  // switch
    }  // if
  }  // updateKey

  void updateMouse() {
    if (onSquare(width/9 * position, 80, 50)) { //giver værdierne for firkanternes centrum og dens
sidelængde
      switch(colour) {
        case (55):
          colour = (255);
          value = int (pow(2, 8 - position));    // slow calculations
          digit = int (pow(10, 8 - position));
          break;
        case (255):
          colour = (55);
          value = 0;
          digit = 0;
          break;
      }  // switch
```

```
109         }  // onCircle
110       }  // updateMouse
111     }  // class
112
113     boolean onSquare(int x, int y, int side) {
114       int halvSide = side / 2; //finder halvdelen af sidelængden
115       int rectX1 = x - halvSide;  //trækker en halvside fra centrum
116       int rectX2 = rectX1 + side;  // addere den ene side med sidelængden for at finde den anden side
117       int rectY1 = y - halvSide;
118       int rectY2 = rectY1 + side; //finder 2 koordinatsæt for firkanten, 1 er for øverste højre hjørne 2 er for
119     nederste venstre hjørne
120
121       if ( mouseX >= rectX1 && mouseX <= rectX2 && mouseY >= rectY1 && mouseY <= rectY2 ) {
122     //tester om musen er mellem en af firkanternes øverstehøjre hjørne og nederste venstre hjørne
123         return true;
124       } else {
125         return false;
126       }  // if
127     }  // end onSquare
128
129     // end
```