```
//Setter opp tilkobling til CoT

  #include <CircusESP32Lib.h> // Biblioteket til Circus of Things.
  char ssid[] = "ssid"; // Skriv inn navnet til ruteren din her.
  char password[] = "passord"; // Skriv inn passordet til ruteren din her.
  char server[] = "www.circusofthings.com"; // Her ligger serveren.
  char token[] = "eyJhbGciOiJIUzI1NiJ9.eyJqdGkiOiI0OTQ1In0.D38ksENOh7rz5DODvLrx9aPemF39WIVLNd6V
niKaOyo"; // Plasser din brukeridentitet her (token).

  char keyRom1[] = "15089"; // Nøkkel-informasjon om personen i rom 1 er der eller ikke (1/0)
  char keyRom2[] = "3417";
  char keyRom3[] = "9261";

  char keyBesokendeRom1[] = "25932"; //Nøkkel-informasjon om hvor mange besøkende det er på rom 1 (0-2)
  char keyBesokendeRom2[] = "14537";
  char keyBesokendeRom3[] = "25233";

  char keyAlleFolka[] = "5842"; //alle som er i kollektivet

  char callAllKey[] = "8894"; //hvis signalet blir 1 får alle beskjed i esp-kontrollen

  CircusESP32Lib circusESP32(server,ssid,password);// Her leses nettadressen til CoT, ssid, og
  //... passord inn. Ikke gjør noen endringer her.

//----------------------------

//verdier for systemet

  int sensorValue;
  bool reValue;
  bool prevreValue = 1;
  bool prevbuttonValue = 1;
  int potValue;
  int tilsammen;
  int besoker = 0;
  int alleFolka = 0;

//----------------------------

//kodelås

  bool buttonValue;
  int attempt[4]; //array. Kan holde 4 verdier
  int buttonMultiplier = 1000;
  int buttonCount = 0;

//----------------------------

//verdier for timers til de besøkende

  unsigned long timerMax = 72000; //72000 * 25 = 1800000 ms (30 min)

  int rom1Timer1;
  int rom1Timer2;
```

```
  int rom2Timer1;
  int rom2Timer2;
  int rom3Timer1;
  int rom3Timer2;

  int thirtyMin1 = 0;
  int thirtyMin12 = 0;
  int thirtyMin2 = 0;
  int thirtyMin22 = 0;
  int thirtyMin3 = 0;
  int thirtyMin32 = 0;

  int klarTimer1 = 0;
  int klarTimer12 = 0;
  int klarTimer2 = 0;
  int klarTimer22 = 0;
  int klarTimer3 = 0;
  int klarTimer32 = 0;

  int ingenPlass = 0;

//----------------------------

//Brukes for å finne hvilket besøk som har vært lengst på besøk

  int klarTimerVerdi1 = 0;
  int klarTimerVerdi12 = 0;
  int klarTimerVerdi2 = 0;
  int klarTimerVerdi22 = 0;
  int klarTimerVerdi3 = 0;
  int klarTimerVerdi32 = 0;

  int storstRom1;
  int storstRom2;
  int storstRom3;
  int storstSemi;
  int storstFinale;
  float storstFinaleIMin;

//----------------------------

//Variabler for CoT

  int rom1Antall = 0;
  int rom2Antall = 0;
  int rom3Antall = 0;

  int rom1Besokende = 0;
  int rom2Besokende = 0;
  int rom3Besokende = 0;

//----------------------------

//pins
```

```
  int potPin = 34;
  int redPin1 = 4;
  int redPin2 = 5;
  int redPin3 = 18;
  int redPin4 = 25;

  int greenPin = 19;
  int buttonPin = 14;
  int rePin = 33;

//----------------------------

//Doorbell

  unsigned long keyPrevMillis = 0;
  unsigned long Intervall = 25;
  int langKeyPressCountMax = 120;    //  (120 * 25 = 3000 ms)
  int langKeyPressCount = 0;


  int doorbellPin = 26;
  bool doorbellValue = 1;
  bool prevdoorbellValue = 1;

//----------------------------

//Green LED

  int greenLedCount = 240; //Er -=1 i loopen. 240-120, 120*25= 3000ms => Lyser i 3 sekunder.
  int greenLedMax = 120; // lyser i 3 sekunder (120 * 25 = 3000 ms)

//----------------------------

void setup() { //kjører en gang når man starter programmet
  pinMode(potPin, INPUT);
  pinMode(buttonPin, INPUT);
  pinMode(rePin, INPUT);

  pinMode(redPin1, OUTPUT);
  pinMode(redPin2, OUTPUT);
  pinMode(redPin3, OUTPUT);
  pinMode(redPin4, OUTPUT);
  pinMode(greenPin, OUTPUT);

  pinMode(doorbellPin, INPUT);


  analogReadResolution(10); // potmeteret vil gi ut verdiene 0-1023
  Serial.begin(9600); //åpner serial porten og setter data raten til 9600 bps

  circusESP32.beginESP(); // Initialiserer oppkobling mot CoT
}
```

```
void loop() { //kjører hele tiden
  if (millis() - keyPrevMillis >= Intervall) { //Hele koden er i denne if-setningen. Det betyr koden kjører hvert 25ms. Det gjør at man kan lage timings med å øke verdier hver gang koden kjører. 25ms er så liten tid, det ikke blir problematisk.
    keyPrevMillis = millis(); //resetter den forrige millisen slik den kjører hvert 25ms.

    timingCounters(); //Counters som går opp eller ned 1. Blir brukt for timings.


    //Variabler som kan forandre seg. Derfor de er i void loop.
    sensorValue = analogRead(potPin);
    buttonValue = digitalRead(buttonPin);
    reValue = digitalRead(rePin);
    potValue = map(sensorValue, 0, 1023, 1, 4); // splits the potvalues to only 4 possiblities (0-1023 --> 0-4).
    doorbellValue = digitalRead(doorbellPin);

    hvorMangeLedLyser(); //Sier hvor mange LEDs som skal lyse iforhold til verdien til potmeteret.


    //kodelås

    registrerKoden(); //Registrerer hvert siffer i pin-koden du trykker inn.
    setterKodenSammen(); //Setter en variabel, tilsammen, til en 4-sifret kode i rett rekkefølge. Eks. tilsammen = 1234.
    sjekkPassordPerson1(); //sjekker om passordet stemmer med person 1.
    sjekkPassordPerson2();
    sjekkPassordPerson3();
    sjekkPassordFeil(); //sjekker om passordet ikke stemmer med noen personer.

    //ringeklokke

    sjekkOmTrykketRingeklokke(); //Hvis du trykker får alle beskjed. Hvis du holder inne i 3 sek kan du velge hvem som får besøket.

    //besokende

    registrerBesokende1(); //sjekker om knappen har blitt holdt inne i 3 sekunder og potverdien er 1. Da blir besøkende 1 registrert.
    registrerBesokende2();
    registrerBesokende3();
    registrerOmNullesUt();

    sjekkResetKnapp(); //sjekker om reset knappen er trykt.


    prevbuttonValue = buttonValue;
    prevreValue = reValue;
    prevdoorbellValue = doorbellValue;

    besokendeTimer1Ferdig(); //sjekker om timeren har gått over 30 min. Hvis den har får thirtymin1 verdien 1
    besokendeTimer12Ferdig();
    besokendeTimer2Ferdig();
    besokendeTimer22Ferdig();
    besokendeTimer3Ferdig();
```

```
    besokendeTimer32Ferdig();

    greenLedTimer(); //sjekker om green LED har blitt under en verdi, slik at den går tilbake til LOW
  }
}

//for hele systemet

void hvorMangeLedLyser() { //hvor mange LED som lyser i forhold til potmeteret.
   if (potValue == 1) {
     digitalWrite(redPin1, HIGH);
     digitalWrite(redPin2, LOW);
     digitalWrite(redPin3, LOW);
     digitalWrite(redPin4, LOW);
  }

   if (potValue == 2) {
     digitalWrite(redPin1, HIGH);
     digitalWrite(redPin2, HIGH);
     digitalWrite(redPin3, LOW);
     digitalWrite(redPin4, LOW);
  }

   if (potValue == 3) {
     digitalWrite(redPin1, HIGH);
     digitalWrite(redPin2, HIGH);
     digitalWrite(redPin3, HIGH);
     digitalWrite(redPin4, LOW);
  }

   if (potValue == 4) {
     digitalWrite(redPin1, HIGH);
     digitalWrite(redPin2, HIGH);
     digitalWrite(redPin3, HIGH);
     digitalWrite(redPin4, HIGH);
  }
}

//----------------------------

//resetknapp

void sjekkResetKnapp() { //sjekker om reset knappen har blitt trykt
  if (reValue == HIGH && prevreValue == LOW) { //reset button
    reset(); //function
  }
}

void reset() { //resetter pinkoden og ringeknappen
  tilsammen = 0;
  buttonMultiplier = 1000;
  buttonCount = 0;
  potValue = 6;
  besoker = 0;
```

```
  prevreValue = 1;
  reValue = 1;
  doorbellValue = 1;
  prevdoorbellValue = 1;
  langKeyPressCount = 0;
  keyPrevMillis = 0;
  Serial.println("\n----------------------");
  return;
}

//---slutt på resetknapp---

//kodelås

void registrerKoden() { //kobler antall LEDs som lyser til tall
     if (buttonValue == HIGH && prevbuttonValue == LOW && potValue == 1) { //trigger når du slepper knappen
     Serial.println("1");
     attempt[buttonCount] = 1*buttonMultiplier; //Bruker arrayen og buttoncount til å se hvilken rekkefølge knappen
e har blitt trykket inn.
     buttonMultiplier/=10; //Den neste verdien vil være delt på 10, altså vil være neste siffer i pinkoden.
     buttonCount++; // For å gå til neste arrayverdi.
     delay(175); // liten delay slik den ikke tror du trykker flere ganger
     }

   if (buttonValue == HIGH && prevbuttonValue == LOW && potValue == 2) {
     Serial.println("2");
     attempt[buttonCount] = 2*buttonMultiplier;
     buttonMultiplier/=10;
     buttonCount++;
     delay(175);
     }


   if (buttonValue == HIGH && prevbuttonValue == LOW && potValue == 3) {
     Serial.println("3");
     attempt[buttonCount] = 3*buttonMultiplier;
     buttonMultiplier/=10;
     buttonCount++;
     delay(175);
     }


   if (buttonValue == HIGH && prevbuttonValue == LOW && potValue == 4) {
     Serial.println("4");
     attempt[buttonCount] = 4*buttonMultiplier;
     buttonMultiplier/=10;
     buttonCount++;
     delay(175);
     }
}

void setterKodenSammen() { //setter isammen koden i rett rekkefølge
  if (buttonValue == HIGH && prevbuttonValue == LOW && buttonCount == 4) {  //buttonCount == 4, ferdig å tr
ykke en pinkode.
```

```
    potValue=5; //Setter potverdien til 5 for å gå utav loopen og inn til om koden stemmer.
    tilsammen = (attempt[0]+attempt[1]+attempt[2]+attempt[3]); ////f.eks: 2000 (trykt med 2 LEDs første trykk) + 400 (trykt med 4 LEDs andre trykk) + 20(...) + 1(...) = 2421 => KODE: 2421

  }
}

void sjekkPassordPerson1() { //sjekker om pinkoden trykt stemmer overens med person 1.
 if (tilsammen == 3333 && potValue == 5) { // koden til person 1 er 3333.
    rom1Antall = circusESP32.read(keyRom1, token); //leser av CoT og ser om personen allerede har registrert seg.

    Serial.println("");

    if (rom1Antall == 0) { //hvis personen ikke har registrert seg
     alleFolka = circusESP32.read(keyAlleFolka, token); //leser av CoT som brukes for å sjekke om det er fullt
     sjekkOmFullt(); //sjekker om noen må bli kastet ut.
     alleFolka++;
     circusESP32.write(keyAlleFolka, alleFolka, token); //skriver en mer til CoT
     circusESP32.write(keyRom1, 1, token); //registrerer person 1 til CoT
     Serial.println("\n----------------------");
     Serial.print("\nPerson 1 er registret.");
     Serial.println("\n----------------------");
     digitalWrite(greenPin, HIGH);
     greenLedCount = 240;
     rom1Antall = 1; // brukes for å sjekke om personen er hjemme når folk prøver å besøke
     reset();
     return;
    }

    if (rom1Antall == 1) { //hvis du tar koden din når du allerede er registrert så avregistrerer du deg.
     circusESP32.write(keyRom1, 0, token);

     rom1Besokende = circusESP32.read(keyBesokendeRom1, token);

     if (rom1Besokende == 0) { //for å se hvor mange som må trekkes fra alleFolka
      alleFolka = circusESP32.read(keyAlleFolka, token);
      alleFolka -= 1;
     }

     if (rom1Besokende == 1) {//for å se hvor mange som må trekkes fra alleFolka
      alleFolka = circusESP32.read(keyAlleFolka, token);
      alleFolka -= 2; //hvis person 1 går og har 1 besøk, så går det to personer
      Serial.println("\n----------------------");
      Serial.print("Rom 1 besok 1 er avregistrert.");
      Serial.println("\n----------------------");
     }

     if (rom1Besokende == 2) {//for å se hvor mange som må trekkes fra alleFolka
      alleFolka = circusESP32.read(keyAlleFolka, token);
      alleFolka -= 3;
      Serial.println("\n----------------------");
      Serial.print("Begge besokende til rom 1 er avregistrert.");
      Serial.println("\n----------------------");
     }
```

```
      circusESP32.write(keyBesokendeRom1, 0, token); //alle besøkene blir kastet ut hvis person 1 sjekker ut
      circusESP32.write(keyAlleFolka, alleFolka, token);

      Serial.println("\n----------------------");
      Serial.print("Person 1 er avregistrert.");
      Serial.println("\n----------------------");
      rom1Antall = 0; // brukes for å sjekke om personen er hjemme når folk prøver å besøke
      klarTimer1 = 0; //timeren vil ikke lengre være gyldig. Besøket finnes ikke lengre
      klarTimer12 = 0; //timeren vil ikke lengre være gyldig. Besøket finnes ikke lengre
      reset();
      return;
    }
   }
}

void sjekkPassordPerson2() { //se forklaring for sjekkPassordPerson1()
  if (tilsammen == 1421 && potValue == 5) { // koden for person 2
    rom2Antall = circusESP32.read(keyRom2, token);
    Serial.println("");

     if (rom2Antall == 0) {
      alleFolka = circusESP32.read(keyAlleFolka, token);
      sjekkOmFullt();
      alleFolka++;
      circusESP32.write(keyAlleFolka, alleFolka, token);
      circusESP32.write(keyRom2, 1, token);
      Serial.println("\n----------------------");
      Serial.print("\nPerson 2 er registret.");
      Serial.println("\n----------------------");
      digitalWrite(greenPin, HIGH);
      greenLedCount = 240;
      rom2Antall = 1;
      reset();
      return;
     }

     if (rom2Antall == 1) { //avregistrering
      circusESP32.write(keyRom2, 0, token);
      rom2Besokende = circusESP32.read(keyBesokendeRom2, token);

      if (rom2Besokende == 0) {
       alleFolka = circusESP32.read(keyAlleFolka, token);
       alleFolka -= 1;
      }

      if (rom2Besokende == 1) {
       alleFolka = circusESP32.read(keyAlleFolka, token);
       alleFolka -= 2; //hvis person 2 går og har 1 besøk, så går det to personer
       Serial.println("\n----------------------");
       Serial.print("Rom 2 besok 1 er avregistrert.");
       Serial.println("\n----------------------");
      }
```

```
      if (rom2Besokende == 2) {
        alleFolka = circusESP32.read(keyAlleFolka, token);
        alleFolka -= 3;
        Serial.println("\n----------");
        Serial.print("Begge besokende til rom 2 er avregistrert.");
        Serial.println("\n----------");
      }


      circusESP32.write(keyBesokendeRom2, 0, token); //alle besøkene blir kastet ut hvis person 2 sjekker ut
      circusESP32.write(keyAlleFolka, alleFolka, token);

      Serial.println("\n----------");
      Serial.print("Person 2 er avregistrert.");
      Serial.println("\n----------");
      rom2Antall = 0; // brukes for å sjekke om personen er hjemme når folk prøver å besøke
      klarTimer2 = 0;
      klarTimer22 = 0;
      reset();
      return;
    }
  }
}

void sjekkPassordPerson3() { //se forklaring for sjekkPassordPerson1()
  if (tilsammen == 1234 && potValue == 5) { // koden for person 3
    rom3Antall = circusESP32.read(keyRom3, token);
    Serial.println("");

     if (rom3Antall == 0) {
      alleFolka = circusESP32.read(keyAlleFolka, token);
      sjekkOmFullt();
      alleFolka++;
      circusESP32.write(keyAlleFolka, alleFolka, token);
      circusESP32.write(keyRom3, 1, token);
      Serial.println("\n----------");
      Serial.print("\nPerson 3 er registret.");
      Serial.println("\n----------");
      digitalWrite(greenPin, HIGH);
      greenLedCount = 240;
      rom3Antall = 1; // brukes for å sjekke om personen er hjemme når folk prøver å besøke
      reset();
      return;
    }
      if (rom3Antall == 1) {
      circusESP32.write(keyRom3, 0, token);
      rom3Besokende = circusESP32.read(keyBesokendeRom3, token);


      if (rom3Besokende == 0) {
        alleFolka = circusESP32.read(keyAlleFolka, token);
        alleFolka -= 1;
      }
```

```
      if (rom3Besokende == 1) {
        alleFolka = circusESP32.read(keyAlleFolka, token);
        alleFolka -= 2; //hvis person 3 går og har 1 besøk, så går det to personer
        Serial.println("\n----------------------");
        Serial.print("Rom 3 besok 1 er avregistrert.");
        Serial.println("\n----------------------");
      }

      if (rom3Besokende == 2) {
        alleFolka = circusESP32.read(keyAlleFolka, token);
        alleFolka -= 3;
        Serial.println("\n----------------------");
        Serial.print("Begge besokende til rom 3 er avregistrert.");
        Serial.println("\n----------------------");
      }

      circusESP32.write(keyBesokendeRom3, 0, token); //alle besøkene blir kastet ut hvis person 3 sjekker ut
      circusESP32.write(keyAlleFolka, alleFolka, token);

      Serial.println("\n----------------------");
      Serial.print("Person 3 er avregistrert.");
      Serial.println("\n----------------------");
      rom3Antall = 0; // brukes for å sjekke om personen er hjemme når folk prøver å besøke
      klarTimer3 = 0;
      klarTimer32 = 0;

      reset();
      return;
      }
  }
}

void sjekkPassordFeil() { //sjekker om passordet ikke stemmer med noen av kodene
  if ((tilsammen != 1421) && (potValue == 5) && (tilsammen != 3333) && (tilsammen != 1234)) { //hvis feil kode
   Serial.println(String(tilsammen) + " var ikke rett");
   reset();
  }
}

//---slutt på kodelås---

//ringeklokke

void sjekkOmTrykketRingeklokke() { //sjekker om man har trykt eller sluppet ringeknappen
  if ((prevdoorbellValue == HIGH) && (doorbellValue == LOW) && (besoker == 0)) {
     keyTrykk();
    }

   else if ((prevdoorbellValue == LOW) && (doorbellValue == HIGH) && (besoker == 0)) {
     keySlipp();
    }

   else if ((doorbellValue == LOW) && (besoker == 0)) { //jo lengre man holder nede knappen, jo større vil langKe
yPressCount bli. Den stiger hvert 25 ms.
```

```
    langKeyPressCount++;
   }
}

void keyTrykk() { //kjøres hvis man trykker ned ringeknappen
  Serial.println("key press"); //slik man vet systemet registrerte trykket
  langKeyPressCount = 0; //resetter counteren slik man kan se om
}

void keySlipp() { //kjøres når man slipper ringeknappen
  Serial.println("key release");

  if (langKeyPressCount >= langKeyPressCountMax) { //max er 120. Hvis langkeypresscount har gått igjennom loo
pen 120 ganger (120*25=3000ms), er det et langt trykk.
      langKeyTrykk();                    // inspirasjon hentet fra https://learn.adafruit.com/fiber-optic-whip/the-code l
astet ned 14.05.21
  else {
      kortKeyTrykk();
  }
}

void kortKeyTrykk() { //kjøres hvis man har holdt inne ringeknappen mindre enn 3 sekunder.
  Serial.println("\n----------------------");
  Serial.print("Du er ukjent. Alle faar beskjed. Vent paa hjelp.");
  Serial.println("\n----------------------");
  circusESP32.write(callAllKey, 1, token); //alle får beskjed
  delay(400);
  Serial.println("\n----------------------");
  Serial.print("Hjelpen stiller inn hvem du besoker. 4 LEDs betyr du nulles ut av systemet");
  Serial.println("\n----------------------");
  besoker = 1; //går over til å velge besøker
  doorbellValue = 1; //knappen kunne være buggy hvis ikke
  prevdoorbellValue = 1;
}

void langKeyTrykk() { //kjøres hvis man har holdt inne ringeknappen lengre enn 3 sekunder.
  alleFolka = circusESP32.read(keyAlleFolka, token); //reader for å se om det er fullt. Blir brukt senere.
  Serial.println("\n----------------------");
  Serial.print("Du er besoker. Still inn hvem du besoker og trykk paa ringeknappen for aa bli registrert.");
  Serial.println("\n----------------------");
  besoker = 1;
}

//for rom 1 besokende 1

void besokendeTimer1() { //skjer når ny besøkende skal bli registrert
  Serial.println("\n----------------------");
  Serial.print("besokende 1, rom 1 timer startet");
  Serial.println("\n----------------------");
  rom1Timer1 = 0; //nullstiller timeren
  klarTimer1 = 1; //gjør timeren gyldig
}

void besokendeTimer1Ferdig() { //kjører hele tiden i loopen for å sjekke om en timer er gått ut
```

```
  if ((rom1Timer1 >= timerMax) && (klarTimer1 == 1)) {
      thirtyMin1 = 1; //verdien tilsier at timeren er over grensen, og de kan bli kastet ut hvis noen nye prøver å besøk
e.
  }
  else {
    thirtyMin1 = 0;
  }
}

//-------------
//for rom 1 besokende 2

void besokendeTimer12() { //se forklaring besokendeTimer1
  Serial.println("\n----------------------");
  Serial.print("besokende 2, rom 1 timer startet");
  Serial.println("\n----------------------");
  rom1Timer2 = 0;
  klarTimer12 = 1;
}

void besokendeTimer12Ferdig() { //se forklaring besokendeTimer1Ferdig
  if ((rom1Timer2 >= timerMax) && (klarTimer12 == 1)) {
    thirtyMin12 = 1;
  }

  else {
    thirtyMin12 = 0;
  }
}

//-------------
//for rom 2 besokende 1

void besokendeTimer2() { //se forklaring besokendeTimer1
  Serial.println("\n----------------------");
  Serial.print("besokende 1, rom 2 timer startet");
  Serial.println("\n----------------------");
  rom2Timer1 = 0;
  klarTimer2 = 1;
}

void besokendeTimer2Ferdig() { //se forklaring besokendeTimer1Ferdig
  if ((rom2Timer1 >= timerMax) && (klarTimer2 == 1)) {
    thirtyMin2 = 1;
  }

  else {
    thirtyMin2 = 0;
  }
}

//-------------
//for rom 2 besokende 2
```

```
void besokendeTimer22() {
  Serial.println("\n----------------------");
  Serial.print("besokende 2, rom 2 timer startet");
  Serial.println("\n----------------------");
  rom2Timer2 = 0;
  klarTimer22 = 1;
}

void besokendeTimer22Ferdig() {
  if ((rom2Timer2 >= timerMax) && (klarTimer22 == 1)) {
    thirtyMin22 = 1;
  }

  else {
    thirtyMin22 = 0;
  }

}

//-------------
//for rom 3 besokende 1

void besokendeTimer3() {
  Serial.println("\n----------------------");
  Serial.print("besokende 1, rom 3 timer startet");
  Serial.println("\n----------------------");
  rom3Timer1 = 0;
  klarTimer3 = 1;
}

void besokendeTimer3Ferdig() {
  if ((rom3Timer1 >= timerMax) && (klarTimer3 == 1)) {
    thirtyMin3 = 1;
  }
  else {
    thirtyMin3 = 0;
    }
}

//-------------
//for rom 3 besokende 2

void besokendeTimer32() {
  Serial.println("\n----------------------");
  Serial.print("besokende 2, rom 3 timer startet");
  Serial.println("\n----------------------");
  rom3Timer2 = 0;
  klarTimer32 = 1;
}

void besokendeTimer32Ferdig() {
  if ((rom3Timer2 >= timerMax) && (klarTimer32 == 1)) {
    thirtyMin32 = 1;
  }
```

```
  else {
    thirtyMin32 = 0;
   }
}
//-------------

void sjekkOmFullt30Min() { //brukes når besøkende prøver å komme seg inn.
  if (alleFolka > 4) { //har satt maksimum 5 personer.
    Serial.println("\n----------------------");
    Serial.print("det er fullt. Sjekker om noen har vaert paa besok lengre enn 30 min...");
    Serial.println("\n----------------------");
    finnStorstTid30Min(); //ingenPlass blir 1 hvis det ikke er noen ledige plasser
    return;
  }
  else {
    ingenPlass = 0;
  }

}

void sjekkOmFullt() { //brukes når person prøver å komme seg inn. De er prioritert over besøkende

  if (alleFolka > 4) {
    Serial.println("\n----------------------");
    Serial.print("det er fullt. Forste besokende blir kastet ut");
    Serial.println("\n----------------------");
    finnStorstTid(); //finner største tiden og kaster besøket ut uansett
  }
  return;
}

void greenLedTimer() { //sjekker om green LED har blitt under en verdi, slik at den går tilbake til LOW
  if (greenLedCount <= greenLedMax) {
    digitalWrite(greenPin, LOW);
  }
}

void finnStorstTid() { //finner hvilket besøk som har vært på besøk lengst.

  //sjekker hvilke timers som er gyldige, som vil være med på beregningen. De som ikke er klare vil bare være 0.
  if (klarTimer1 == 1) {
    klarTimerVerdi1 = rom1Timer1;
  }
  if (klarTimer12 == 1) {
    klarTimerVerdi12 = rom1Timer2;
  }
  if (klarTimer2 == 1) {
    klarTimerVerdi2 = rom2Timer1;
  }
  if (klarTimer22 == 1) {
    klarTimerVerdi22 = rom2Timer2;
  }
  if (klarTimer3 == 1) {
    klarTimerVerdi3 = rom3Timer1;
```

```
  }
  if (klarTimer32 == 1) {
    klarTimerVerdi32 = rom3Timer2;
  }
  //-------------
  //En slags cup, der vinneren blir kastet ut
  storstRom1 = max(klarTimerVerdi1, klarTimerVerdi12);
  storstRom2 = max(klarTimerVerdi2, klarTimerVerdi22);
  storstRom3 = max(klarTimerVerdi3, klarTimerVerdi32);

  storstSemi = max(storstRom1, storstRom2);
  storstFinale = max(storstSemi, storstRom3);
  //-------------
  storstFinaleIMin = ((storstFinale*0.001*25)/60); //(storstFinale*0.001*25) = i sekund, delt på 60 => i minutt (kode
n kjører hvert 25 millisek må gange med 25)
  Serial.println("\n-----------------------");
  Serial.print("Den personen som har vaert her lengst, har vaert her i " + String(storstFinaleIMin) +" min(s)." );
  Serial.println("\n-----------------------");

  //Her sjekkes hver besokende for om den er lik den som vant finalen. Hvis den er det skal det besøket kastes ut.
  //----- rom 1 besøker 1

  if ((storstFinale == klarTimerVerdi1)) { //hvis rom 1 besøker 1 har vært der lengst
     rom1Besokende = circusESP32.read(keyBesokendeRom1, token); //sjekker med CoT hvor mange besøkende det
var før.
     alleFolka = circusESP32.read(keyAlleFolka, token); //sjekker med CoT hvor mange det var til sammen før.
     rom1Besokende -= 1; //fjerner en pga. besøker 1 ble kastet ut
     alleFolka -= 1; //fjerner en pga. besøker 1 ble kastet ut
     circusESP32.write(keyAlleFolka, alleFolka, token); //skriver til CoT nye verdier
     circusESP32.write(keyBesokendeRom1, rom1Besokende, token); //skriver til CoT nye verdier

    if (klarTimer1 == 1 && klarTimer12 == 1) {
      //hvis vi har to besøkende og første blir kastet ut, blir den andre besokende til den eneste, som betyr den blir til f
ørste besøker
      thirtyMin1 = 0; // blir til 1 med en gang hvis den andre besøkende som blir første, er over 30 min. Hvis den aller
ede er 1, må den bli 0.
      rom1Timer1 = rom1Timer2;
      klarTimer1 = 1;
      klarTimer12 = 0;
      thirtyMin12 = 0;
      rom1Timer2 = 0;
      Serial.println("\n-----------------------");
      Serial.print("rom 1 besokende 1 har blitt kastet ut. Besokende 2 har blitt om til besokende 1.");
      Serial.println("\n-----------------------");
    }
    else {
      klarTimer1 = 0;
      thirtyMin1 = 0;
      rom1Timer1 = 0;
      Serial.println("\n-----------------------");
      Serial.print("rom 1 besokende 1 har blitt kastet ut.");
      Serial.println("\n-----------------------");
    }
```

```
  }
  //-------------
  //----- rom 1 besøker 2
  if ((storstFinale == klarTimerVerdi12)) {
    rom1Besokende = circusESP32.read(keyBesokendeRom1, token);
    alleFolka = circusESP32.read(keyAlleFolka, token);
    rom1Besokende -= 1;
    alleFolka -= 1;
    circusESP32.write(keyAlleFolka, alleFolka, token);
    circusESP32.write(keyBesokendeRom1, rom1Besokende, token);
    klarTimer12 = 0;
    thirtyMin12 = 0;
    thirtyMin12 = 0;
    rom1Timer2 = 0;

    Serial.println("\n----------------------");
    Serial.print("rom 1 besokende 2 har blitt kastet ut.");
    Serial.println("\n----------------------");
  }

  //-------------
  //----- rom 2 besøker 1

  if ((storstFinale == klarTimerVerdi2)) {
    rom2Besokende = circusESP32.read(keyBesokendeRom2, token);
    alleFolka = circusESP32.read(keyAlleFolka, token);
    rom2Besokende -= 1;
    alleFolka -= 1;
    circusESP32.write(keyAlleFolka, alleFolka, token);
    circusESP32.write(keyBesokendeRom2, rom2Besokende, token);
    if (klarTimer2 == 1 && klarTimer22 == 1) { //hvis vi har to besøkende og første blir kastet ut, blir den andre beso
kende til den eneste, som betyr den blir til første besøker
    thirtyMin2 = 0;
    rom2Timer1 = rom2Timer2;
    klarTimer22 = 0;
    thirtyMin22 = 0;
    klarTimer2 = 1;
    rom2Timer2 = 0;
    Serial.println("\n----------------------");
    Serial.print("rom 2 besokende 1 har blitt kastet ut. Besokende 2 har blitt om til besokende 1.");
    Serial.println("\n----------------------");
    }
    else {
      klarTimer2 = 0;
      thirtyMin2 = 0;
      rom2Timer1 = 0;
      Serial.println("\n----------------------");
      Serial.print("rom 2 besokende 1 har blitt kastet ut.");
      Serial.println("\n----------------------");
    }

  }
  //-------------
  //----- rom 2 besøker 2
```

```
  if ((storstFinale == klarTimerVerdi22)) {
    rom2Besokende = circusESP32.read(keyBesokendeRom2, token);
    alleFolka = circusESP32.read(keyAlleFolka, token);
    rom2Besokende -= 1;
    alleFolka -= 1;
    circusESP32.write(keyAlleFolka, alleFolka, token);
    circusESP32.write(keyBesokendeRom2, rom2Besokende, token);
    thirtyMin22 = 0;
    klarTimer22 = 0;
    rom2Timer2 = 0;
    Serial.println("\n----------------------");
    Serial.print("rom 2 besokende 2 har blitt kastet ut.");
    Serial.println("\n----------------------");
  }

  //-------------
  //----- rom 3 besøker 1

  if ((storstFinale == klarTimerVerdi3)) {
    rom3Besokende = circusESP32.read(keyBesokendeRom3, token);
    alleFolka = circusESP32.read(keyAlleFolka, token);
    rom3Besokende -= 1;
    alleFolka -= 1;
    circusESP32.write(keyAlleFolka, alleFolka, token);
    circusESP32.write(keyBesokendeRom3, rom3Besokende, token);
    if (klarTimer3 == 1 && klarTimer32 == 1) { //hvis vi har to besøkende og første blir kastet ut, blir den andre beso
kende til den eneste, som betyr den blir til første besøker
    thirtyMin3 = 0;
    rom3Timer1 = rom3Timer2;
    klarTimer32 = 0;
    thirtyMin32 = 0;
    klarTimer3 = 1;
    rom3Timer2 = 0;
    Serial.println("\n----------------------");
    Serial.print("rom 3 besokende 1 har blitt kastet ut. Besokende 2 har blitt om til besokende 1.");
    Serial.println("\n----------------------");
    }
    else {
      rom3Timer1 = 0;
      klarTimer3 = 0;
      thirtyMin3 = 0;
    }
    Serial.println("\n----------------------");
    Serial.print("rom 3 besokende 1 har blitt kastet ut.");
    Serial.println("\n----------------------");
  }
  //-------------
  //----- rom 3 besøker 2
  if ((storstFinale == klarTimerVerdi32)) {
    rom3Besokende = circusESP32.read(keyBesokendeRom3, token);
    alleFolka = circusESP32.read(keyAlleFolka, token);
    rom3Besokende -= 1;
    alleFolka -= 1;
    circusESP32.write(keyAlleFolka, alleFolka, token);
```

```
    circusESP32.write(keyBesokendeRom3, rom3Besokende, token);
    klarTimer32 = 0;
    thirtyMin32 = 0;
    rom3Timer2 = 0;
    Serial.println("\n----------");
    Serial.print("rom 3 besokende 2 har blitt kastet ut.");
    Serial.println("\n----------");
  }
  delay(30);
  resetKlarTimerVerdi(); //tilbakestiller alle verdiene til neste gang den blir kjørt.
}

void finnStorstTid30Min() { //veldig lik finnStorstTid(), men har en ekstra variabel thirtyMin.
  //når thirtyMin er 1, betyr det timeren har vært lengre enn 30 min, og kan bli kastet ut.
  if ((thirtyMin1 == 0) && (thirtyMin12 == 0) && (thirtyMin2 == 0) && (thirtyMin22 == 0) && (thirtyMin3 == 0)
 && (thirtyMin32 == 0)) {
    besoker = 0;
    ingenPlass = 1;
    return;
  }

  if (klarTimer1 == 1 && thirtyMin1 == 1) {
    klarTimerVerdi1 = rom1Timer1;
  }
  if (klarTimer12 == 1 && thirtyMin12 == 1) {
    klarTimerVerdi12 = rom1Timer2;
  }
  if (klarTimer2 == 1 && thirtyMin2 == 1) {
    klarTimerVerdi2 = rom2Timer1;
  }
  if (klarTimer22 == 1 && thirtyMin22 == 1) {
    klarTimerVerdi22 = rom2Timer2;
  }
  if (klarTimer3 == 1 && thirtyMin3 == 1) {
    klarTimerVerdi3 = rom3Timer1;
  }
  if (klarTimer32 == 1 && thirtyMin32 == 1) {
    klarTimerVerdi32 = rom3Timer2;
  }




  storstRom1 = max(klarTimerVerdi1, klarTimerVerdi12);
  storstRom2 = max(klarTimerVerdi2, klarTimerVerdi22);
  storstRom3 = max(klarTimerVerdi3, klarTimerVerdi32);

  storstSemi = max(storstRom1, storstRom2);
  storstFinale = max(storstSemi, storstRom3);

  storstFinaleIMin = ((storstFinale*0.001*25)/60); //(storstFinale*0.001*25) = i sekund, delt på 60 => i minutt (kode
n kjører hvert 25 millisek må gange med 25)

  Serial.println("\n----------");
```

```
  Serial.print("Den personen som har vaert her lengst, har vaert her i " + String(storstFinaleIMin) +" min(s)." );
  Serial.println("\n----------------------");

  //----- rom 1 besøker 1
  //-------------

  if ((storstFinale == klarTimerVerdi1)) { //hvis rom 1 besøker 1 har vært der lengst
    rom1Besokende = circusESP32.read(keyBesokendeRom1, token);
    alleFolka = circusESP32.read(keyAlleFolka, token);
    rom1Besokende -= 1;
    alleFolka -= 1;
    circusESP32.write(keyAlleFolka, alleFolka, token);
    circusESP32.write(keyBesokendeRom1, rom1Besokende, token);

    if (klarTimer1 == 1 && klarTimer12 == 1) { //hvis vi har to besøkende og første blir kastet ut, blir den andre beso
kende til den eneste, som betyr den blir til første besøker
    thirtyMin1 = 0; //blir fort til 1 hvis den som blir til første besøkende har vært i lang tid. men kan være bare første
var 30min.
    rom1Timer1 = rom1Timer2;
    klarTimer1 = 1;
    klarTimer12 = 0;
    thirtyMin12 = 0;
    rom1Timer2 = 0;
    Serial.println("\n----------------------");
    Serial.print("rom 1 besokende 1 har blitt kastet ut. Besokende 2 har blitt om til besokende 1.");
    Serial.println("\n----------------------");
    }
    else {
      klarTimer1 = 0;
      thirtyMin1 = 0;
      rom1Timer1 = 0;
      Serial.println("\n----------------------");
      Serial.print("rom 1 besokende 1 har blitt kastet ut.");
      Serial.println("\n----------------------");
    }

  }
  //----- rom 1 besøker 2
  //-------------
  if ((storstFinale == klarTimerVerdi12)) {
    rom1Besokende = circusESP32.read(keyBesokendeRom1, token);
    alleFolka = circusESP32.read(keyAlleFolka, token);
    rom1Besokende -= 1;
    alleFolka -= 1;
    circusESP32.write(keyAlleFolka, alleFolka, token);
    circusESP32.write(keyBesokendeRom1, rom1Besokende, token);
    klarTimer12 = 0;
    thirtyMin12 = 0;
    rom1Timer2 = 0;
    Serial.println("\n----------------------");
    Serial.print("rom 1 besokende 2 har blitt kastet ut.");
    Serial.println("\n----------------------");
  }
```

```
  //-------------
  //----- rom 2 besøker 1

  if ((storstFinale == klarTimerVerdi2)) {
    rom2Besokende = circusESP32.read(keyBesokendeRom2, token);
    alleFolka = circusESP32.read(keyAlleFolka, token);
    rom2Besokende -= 1;
    alleFolka -= 1;
    circusESP32.write(keyAlleFolka, alleFolka, token);
    circusESP32.write(keyBesokendeRom2, rom2Besokende, token);
    if (klarTimer2 == 1 && klarTimer22 == 1) { //hvis vi har to besøkende og første blir kastet ut, blir den andre beso
kende til den eneste, som betyr den blir til første besøker
      thirtyMin2 = 0;
      rom2Timer1 = rom2Timer2;
      klarTimer2 = 1;
      klarTimer22 = 0;
      thirtyMin22 = 0;
      rom2Timer2 = 0;
      Serial.println("\n-----------------------");
      Serial.print("rom 2 besokende 1 har blitt kastet ut. Besokende 2 har blitt om til besokende 1.");
      Serial.println("\n-----------------------");
    }
    else {
      klarTimer2 = 0;
      thirtyMin2 = 0;
      rom2Timer1 = 0;
      Serial.println("\n-----------------------");
      Serial.print("rom 2 besokende 1 har blitt kastet ut.");
      Serial.println("\n-----------------------");
    }

  }
  //-------------
  //----- rom 2 besøker 2
  if ((storstFinale == klarTimerVerdi22)) {
    rom2Besokende = circusESP32.read(keyBesokendeRom2, token);
    alleFolka = circusESP32.read(keyAlleFolka, token);
    rom2Besokende -= 1;
    alleFolka -= 1;
    circusESP32.write(keyAlleFolka, alleFolka, token);
    circusESP32.write(keyBesokendeRom2, rom2Besokende, token);
    thirtyMin22 = 0;
    klarTimer22 = 0;
    rom2Timer2 = 0;
    Serial.println("\n-----------------------");
    Serial.print("rom 2 besokende 2 har blitt kastet ut.");
    Serial.println("\n-----------------------");
  }

  //-------------
  //----- rom 3 besøker 1

  if ((storstFinale == klarTimerVerdi3)) {
    rom3Besokende = circusESP32.read(keyBesokendeRom3, token);
```

```
    alleFolka = circusESP32.read(keyAlleFolka, token);
    rom3Besokende -= 1;
    alleFolka -= 1;
    circusESP32.write(keyAlleFolka, alleFolka, token);
    circusESP32.write(keyBesokendeRom3, rom3Besokende, token);
    if (klarTimer3 == 1 && klarTimer32 == 1) { //hvis vi har to besøkende og første blir kastet ut, blir den andre beso
kende til den eneste, som betyr den blir til første besøker
      thirtyMin3 = 0;
      rom3Timer1 = rom3Timer2;
      klarTimer3 = 1;
      klarTimer32 = 0;
      thirtyMin32 = 0;
      rom3Timer2 = 0;
      Serial.println("\n----------------------");
      Serial.print("rom 3 besokende 1 har blitt kastet ut. Besokende 2 har blitt om til besokende 1.");
      Serial.println("\n----------------------");
    }
    else {
      klarTimer3 = 0;
      thirtyMin3 = 0;
      rom3Timer1 = 0;
      Serial.println("\n----------------------");
      Serial.print("rom 3 besokende 1 har blitt kastet ut.");
      Serial.println("\n----------------------");
    }

  }

  //-------------
  //----- rom 3 besøker 2

  if ((storstFinale == klarTimerVerdi32)) {
    rom3Besokende = circusESP32.read(keyBesokendeRom3, token);
    alleFolka = circusESP32.read(keyAlleFolka, token);
    rom3Besokende -= 1;
    alleFolka -= 1;
    circusESP32.write(keyAlleFolka, alleFolka, token);
    circusESP32.write(keyBesokendeRom3, rom3Besokende, token);
    thirtyMin32 = 0;
    klarTimer32 = 0;
    rom3Timer2 = 0;
    Serial.println("\n----------------------");
    Serial.print("rom 3 besokende 2 har blitt kastet ut.");
    Serial.println("\n----------------------");
  }
  //-------------
  delay(30);
  ingenPlass = 0;
  resetKlarTimerVerdi();
}

void timingCounters() { //disse er alltid i 25ms loopen og vil forandre seg hele tiden
    rom1Timer1++;
    rom1Timer2++;
```

```
    rom2Timer1++;
    rom2Timer2++;
    rom3Timer1++;
    rom3Timer2++;
    greenLedCount -= 1;
}

void registrerBesokende1() { //registrerer besøkende 1 etter å ha valgt med potmeter og trykt på ringeknappen
  //kan ha circus read romAntall her slik at man kan stille inn på circus og koden
  //vil forstå byttingen, men blir så mye som stopper opp koden.
  if ((besoker == 1) && (prevdoorbellValue == HIGH) && (doorbellValue == LOW) && (potValue == 1) && (ro
m1Antall == 1)) { //besoker == 1, altså etter man har hold nede knappen i 3 sek
     rom1Besokende = circusESP32.read(keyBesokendeRom1, token);

     if (rom1Besokende == 2) { //har satt maksimum besøkende som 2
       Serial.println("\n----------------------");
       Serial.print("Person 1 har maksimum besokende. Kom tilbake senere.");
       Serial.println("\n----------------------");
       reset();
     }

     if (rom1Besokende == 1) {
       sjekkOmFullt30Min(); //sjekker om fullt. Hvis det er det, men ingen har vært over grensen (30 min) så vil inge
nPlass bli 1 og man må prøve igjen senere.
       if (ingenPlass == 0) {
       alleFolka = circusESP32.read(keyAlleFolka, token);
       alleFolka++;
       circusESP32.write(keyAlleFolka, alleFolka, token);
       klarTimer12 = 1; //timeren blir gyldig
       thirtyMin12 = 0; //timeren starter
       circusESP32.write(keyBesokendeRom1, 2, token); //vil bli to besøkere
       besokendeTimer12(); //skriver ut at timeren har startet.
       Serial.println("\n----------------------");
       Serial.print("\nBesokende 2 er registret paa rom 1.");
       Serial.println("\n----------------------");
       digitalWrite(greenPin, HIGH);
       greenLedCount = 240; //vil gå av igjen etter 3 sekunder ifra greenLedTimer();
       reset();
       }

       if (ingenPlass == 1) {
         Serial.println("\n----------------------");
         Serial.print("\nDet er ingen som har vaert paa besok lengre enn 30 min. Prov igjen senere.");
         Serial.println("\n----------------------");
         reset();
       }

     }

     if (rom1Besokende == 0) {
       sjekkOmFullt30Min(); //sjekker om fullt. Hvis det er det, men ingen har vært over grensen (30 min) så vil inge
nPlass bli 1 og man må prøve igjen senere.
       if (ingenPlass == 0) {
       alleFolka = circusESP32.read(keyAlleFolka, token);
```

```
      alleFolka++;
      circusESP32.write(keyAlleFolka, alleFolka, token);
      klarTimer1 = 1;
      thirtyMin1 = 0;
      circusESP32.write(keyBesokendeRom1, 1, token);
      besokendeTimer1();
      Serial.println("\n----------------------");
      Serial.print("\nBesokende 1 er registret paa rom 1.");
      Serial.println("\n----------------------");
      digitalWrite(greenPin, HIGH);
      greenLedCount = 240;
      reset();
      }

      if (ingenPlass == 1) {
        Serial.println("\n----------------------");
        Serial.print("\nDet er ingen som har vaert paa besok lengre enn 30 min. Prov igjen senere.");
        Serial.println("\n----------------------");
        reset();
      }

    }
   }
   else if((besoker == 1) && (prevdoorbellValue == HIGH) && (doorbellValue == LOW) && (potValue == 1) &&
 (rom1Antall == 0)) {
    Serial.println("\n----------------------");
    Serial.println("Person 1 er ikke hjemme. Prov igjen senere.");
    Serial.println("\n----------------------");
    delay(300);
    reset();
   }
}

void registrerBesokende2() { //se registrerBesokende1 for forklaringer
  if ((besoker == 1) && (prevdoorbellValue == HIGH) && (doorbellValue == LOW) && (potValue == 2) && (ro
m2Antall == 1)) { //besoker == 1, altså etter en har hold nede knappen i 3 sek
    rom2Besokende = circusESP32.read(keyBesokendeRom2, token);
    Serial.println("");
    if (rom2Besokende == 2) {
      Serial.println("Person 2 har maksimum besokende. Kom tilbake senere.");
      reset();
    }

    if (rom2Besokende == 1) {
      sjekkOmFullt30Min();
      if (ingenPlass == 0) {
      alleFolka = circusESP32.read(keyAlleFolka, token);
      alleFolka++;
      circusESP32.write(keyAlleFolka, alleFolka, token);
      klarTimer22 = 1;
      thirtyMin22 = 0;
      circusESP32.write(keyBesokendeRom2, 2, token);
      besokendeTimer22();
      Serial.println("\n----------------------");
```

```
      Serial.print("\nBesokende 2 er registret paa rom 2.");
      Serial.println("\n-----------------------");
      digitalWrite(greenPin, HIGH);
      greenLedCount = 240;
      reset();
      }

      if (ingenPlass == 1) {
        Serial.println("\n-----------------------");
        Serial.print("\nDet er ingen som har vaert paa besok lengre enn 30 min. Prov igjen senere.");
        Serial.println("\n-----------------------");
        reset();
      }
    }

    if (rom2Besokende == 0) {
      sjekkOmFullt30Min();
      if (ingenPlass == 0) {
      alleFolka = circusESP32.read(keyAlleFolka, token);
      alleFolka++;
      circusESP32.write(keyAlleFolka, alleFolka, token);
      klarTimer2 = 1;
      thirtyMin2 = 0;
      circusESP32.write(keyBesokendeRom2, 1, token);
      besokendeTimer2();
      Serial.println("\n-----------------------");
      Serial.print("\nBesokende 1 er registret paa rom 2.");
      Serial.println("\n-----------------------");
      digitalWrite(greenPin, HIGH);
      greenLedCount = 240;
      reset();
      }

      if (ingenPlass == 1) {
        Serial.println("\n-----------------------");
        Serial.print("\nDet er ingen som har vaert paa besok lengre enn 30 min. Prov igjen senere.");
        Serial.println("\n-----------------------");
        reset();
      }
    }
  }
  else if((besoker == 1) && (prevdoorbellValue == HIGH) && (doorbellValue == LOW) && (potValue == 2) &&
 (rom2Antall == 0)) {
    Serial.println("\n-----------------------");
    Serial.println("Person 2 er ikke hjemme. Prov igjen senere.");
    Serial.println("\n-----------------------");
    delay(300);
    reset();
  }
}

void registrerBesokende3() { //se registrerBesokende1 for forklaringer
  if ((besoker == 1) && (prevdoorbellValue == HIGH) && (doorbellValue == LOW) && (potValue == 3) && (ro
m3Antall == 1)) { //besoker == 1, altså etter en har hold nede knappen i 3 sek
```

```
rom3Besokende = circusESP32.read(keyBesokendeRom3, token);
Serial.println("");
if (rom3Besokende == 2) {
  Serial.println("Person 3 har maksimum besokende. Kom tilbake senere.");
  reset();
}

if (rom3Besokende == 1) {
  sjekkOmFullt30Min();
  if (ingenPlass == 0) {
  alleFolka = circusESP32.read(keyAlleFolka, token);
  alleFolka++;
  circusESP32.write(keyAlleFolka, alleFolka, token);
  klarTimer32 = 1;
  thirtyMin32 = 0;
  circusESP32.write(keyBesokendeRom3, 2, token);
  besokendeTimer32();
  Serial.println("\n-----------------------");
  Serial.print("\nBesokende 2 er registret paa rom 3.");
  Serial.println("\n-----------------------");
  digitalWrite(greenPin, HIGH);
  greenLedCount = 240;
  reset();
  }

  if (ingenPlass == 1) {
    Serial.println("\n-----------------------");
    Serial.print("\nDet er ingen som har vaert paa besok lengre enn 30 min. Prov igjen senere.");
    Serial.println("\n-----------------------");
    reset();
  }

}

if (rom3Besokende == 0) {
  sjekkOmFullt30Min();

  if (ingenPlass == 0) {
    alleFolka = circusESP32.read(keyAlleFolka, token);
    alleFolka++;
    circusESP32.write(keyAlleFolka, alleFolka, token);
    klarTimer3 = 1;
    thirtyMin3 = 0;
    circusESP32.write(keyBesokendeRom3, 1, token);
    besokendeTimer3();
    Serial.println("\n-----------------------");
    Serial.print("\nBesokende 1 er registret paa rom 3.");
    Serial.println("\n-----------------------");
    digitalWrite(greenPin, HIGH);
    greenLedCount = 240;
    reset();
  }

  if (ingenPlass == 1) {
```

```
        Serial.println("\n-----------------------");
        Serial.print("\nDet er ingen som har vaert paa besok lengre enn 30 min. Prov igjen senere.");
        Serial.println("\n-----------------------");
        reset();
       }

     }
   }
   else if((besoker == 1) && (prevdoorbellValue == HIGH) && (doorbellValue == LOW) && (potValue == 3) &&
 (rom3Antall == 0)) {
     Serial.println("\n-----------------------");
     Serial.println("Person 3 er ikke hjemme. Prov igjen senere.");
     Serial.println("\n-----------------------");
     delay(300);
     reset();
   }
}

void registrerOmNullesUt() { //hvis det blir valgt 4 LEDs så vil ingen bli satt inni systemet.
  if ((besoker == 1) && (prevdoorbellValue == HIGH) && (doorbellValue == LOW) && (potValue == 4)) {
    Serial.println("Nulles ut av systemet");
    delay(500);
    reset();
    }
}

void resetKlarTimerVerdi() { //resetter verdier brukt i finnStorstTid
  int klarTimerVerdi1 = 0;
  int klarTimerVerdi12 = 0;
  int klarTimerVerdi2 = 0;
  int klarTimerVerdi22 = 0;
  int klarTimerVerdi3 = 0;
  int klarTimerVerdi32 = 0;
}

//---slutt på ringeklokke---
```