int SPKpin = 2;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "SPKpin" და ინახება მასში რიცხვი 2.

int D2 = 4;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "D2" და ინახება მასში რიცხვი 4.

int Rpin = 3;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "Rpin" და ინახება მასში რიცხვი 3.

int Gpin = 5;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "Gpin" და ინახება მასში რიცხვი 5.

int Bpin = 6;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "Bpin" და ინახება მასში რიცხვი 6.

int D1 = 7;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "D1" და ინახება მასში რიცხვი 7.

int IC8 = 8;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "IC8" და ინახება მასში რიცხვი 8.

int IC4 = 9;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "IC4" და ინახება მასში რიცხვი 9.

int IC2 = A5;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "IC2" და ინახება მასში რიცხვი 10.

int IC1 = A4;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "IC1" და ინახება მასში რიცხვი 11.

int C_LED = 13;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "C_LED" და ინახება მასში რიცხვი 13.

int Vcc = 12;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "Vcc" და ინახება მასში რიცხვი 12.

int Rvalue;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "Rvalue".

int Gvalue;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "Gvalue".

int Bvalue;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "Bvalue".

int RGBtime = 10;// იქმნება მთელი რიცხვების გლობალური ცვლადი, სახელად "RGBtime" და ინახება მასში რიცხვი 10.

unsigned long t;// იქმნება მთელი რიცხვების გლობალური ცვლადი (4 ბაიტი, არაუარყოფითი), სახელად "t".

boolean a;

double C_TIME;

boolean b;

unsigned long t2;

unsigned long t3;

unsigned long t4;

unsigned long t5;

unsigned long t6;

boolean SPKstate;

boolean SPK_ENABLE;

```
boolean unit1;
boolean unit3;
double count1;
double count2;
unsigned long t7;
int c;
int d;
boolean COUNTER;
int RGBcount;
double SPK_T;
int second = 0;
int minute = 15;
int hour = 23;
int day;
int week_day;
int month;
int year;
unsigned long t8;
int counter;
boolean e;
int D3 = A0;
int D4 = A1;
int D5 = A2;
int D6 = A3;
int second_digit1;
int second_digit2;
int minute_digit1;
int minute_digit2;
int hour_digit1;
int hour_digit2;
boolean displays_enable = 1;
int GND_7seg = 11;
int value_7seg;
unsigned long t9;
```

```
unsigned long t10;
boolean clock_enable = 1;
int DP = 10;
boolean morse_enable;
unsigned long T;
boolean enable_2019;
boolean enable_2019_2;
unsigned long t11;
boolean enable_2019_3;
unsigned long t12;
boolean f;
boolean g;
int value_DP;
boolean DP_enable = 1;

void setup()  {
  pinMode(SPKpin, OUTPUT);// ფეხი, სახელად "SPKpin" (2), ხდება გამოსავალი.
  pinMode(D2, OUTPUT);// ფეხი, სახელად "D2" (4), ხდება გამოსავალი.
  pinMode(Rpin, OUTPUT);// ფეხი, სახელად "Rpin" (3), ხდება გამოსავალი.
  pinMode(Gpin, OUTPUT);// ფეხი, სახელად "Gpin" (5), ხდება გამოსავალი.
  pinMode(Bpin, OUTPUT);// ფეხი, სახელად "Bpin" (6), ხდება გამოსავალი.
  pinMode(D1, OUTPUT);// ფეხი, სახელად "D1" (7), ხდება გამოსავალი.
  pinMode(IC8, OUTPUT);// ფეხი, სახელად "IC8" (8), ხდება გამოსავალი.
  pinMode(IC4, OUTPUT);// ფეხი, სახელად "IC4" (9), ხდება გამოსავალი.
  pinMode(IC2, OUTPUT);// ფეხი, სახელად "IC2" (A5), ხდება გამოსავალი.
  pinMode(IC1, OUTPUT);// ფეხი, სახელად "IC1" (11), ხდება გამოსავალი.
  pinMode(C_LED, OUTPUT);// ფეხი, სახელად "C_LED" (13), ხდება გამოსავალი.
  pinMode(Vcc, OUTPUT);// ფეხი, სახელად "Vcc" (12), ხდება გამოსავალი.
  pinMode(D3, OUTPUT);
  pinMode(D4, OUTPUT);
  pinMode(D5, OUTPUT);
  pinMode(D6, OUTPUT);
  pinMode(DP, OUTPUT);
  pinMode(GND_7seg, OUTPUT);
```

```
  analogWrite(GND_7seg, 255);
  analogWrite(DP, 255);
  pinMode(1, INPUT_PULLUP);
  t = millis();// პროგრამის მსვლელობიდან ამ ბრძანებამდე გასული დრო ინახება "t" ცვლადად.
}

void loop() {
  while(digitalRead(1));
  if (millis() == 0)  {// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება თუ "millis()" = 0.
    t = 0;// ცვლადი "t" ხდება 0-ის ტოლი.
    t2 = 0;// ცვლადი "t2" ხდება 0-ის ტოლი.
    t3 = 0;// ცვლადი "t3" ხდება 0-ის ტოლი.
    t4 = 0;// ცვლადი "t4" ხდება 0-ის ტოლი.
    t5 = 0;// ცვლადი "t5" ხდება 0-ის ტოლი.
    t6 = 0;// ცვლადი "t6" ხდება 0-ის ტოლი.
    t7 = 0;// ცვლადი "t7" ხდება 0-ის ტოლი.
    t8 = 0;// ცვლადი "t8" ხდება 0-ის ტოლი.
    t9 = 0;// ცვლადი "t9" ხდება 0-ის ტოლი.
    t10 = 0;// ცვლადი "t10" ხდება 0-ის ტოლი.
    T = 0;
  }
  if (millis() - T >= 1000)  {
    second = second + 1; // ცვლად "second"-ს ემატება რიცხვი 1.
    T = millis();
  }
  if (second == 60)  {
    second = 0;// ცვლად "second"-ის მნიშვნელობა ხდება "0".
    minute = minute + 1; // ცვლად "minute"-ს ემატება რიცხვი 1.
  }// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება იმ შემთხვევაში, თუ ცვლადი "second"-ის მნიშვნელობა იქნება რიცხვი 60.
  if (minute == 60)  {
    minute = 0;// ცვლად "minute"-ის მნიშვნელობა ხდება "0".
    hour = hour + 1; // ცვლად "hour"-ს ემატება რიცხვი 1.
```

}// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება იმ შემთხვევაში, თუ ცვლადი "minute"-ის მნიშვნელობა იქნება რიცხვი 60.

```
 if (hour == 24)  {
  hour = 0;// ცვლად "hour"-ის მნიშვნელობა ხდება "0".
  minute = 0;// ცვლად "minute"-ის მნიშვნელობა ხდება "0".
  second = 0;// ცვლად "second"-ის მნიშვნელობა ხდება "0".
 }// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება იმ შემთხვევაში, თუ ცვლადი "hour"-ის მნიშვნელობა იქნება რიცხვი 24.
 if (second == 0 & minute == 0 & hour == 0)  {
  enable_2019 = 1;
  SPK_ENABLE = 0;
 }
 if (second == 5 & minute == 0 & hour == 0)  {
  displays_enable = 0;
  digitalWrite(DP, 0);
 }
 if (second == 8 & minute == 0 & hour == 0)  {
  clock_enable = 0;
  enable_2019_2 = 1;
  displays_enable = 1;
 }
 if (second == 20 & minute == 0 & hour == 0)  {
  displays_enable = 0;
 }
 if (second == 23 & minute == 0 & hour == 0)  {
  enable_2019_2 = 0;
  clock_enable = 1;
  digitalWrite(DP, 1);
  displays_enable = 1;
 }
 if (second == 30 & minute == 0 & hour == 0)  {
  enable_2019 = 0;
  enable_2019_3 = 1;
  SPK_ENABLE = 1;
```

```
}
if (second == 0 & minute == 1 & hour == 0)  {
  enable_2019_3 = 0;
  SPK_ENABLE = 0;
  COUNTER = 1;
  enable_2019 = 1;
  SPK_ENABLE = 0;
}
if (displays_enable == 0)  {
  if (millis() - t8 >= 10)  {
    if (value_7seg > 0)  {
      analogWrite(GND_7seg, 255 - value_7seg);
      value_7seg--;
    }
    t8 = millis();
  }
}
if (displays_enable == 1)  {
  if (millis() - t10 >= 10)  {
    if (value_7seg < 255)  {
      analogWrite(GND_7seg, 255 - value_7seg);
      value_7seg++;
    }
    t10 = millis();
  }
}
if (enable_2019_2)  {
  morse_enable = 1;
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D1, 1);
  digitalWrite(D1, 0);
```

```
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D2, 1);
  digitalWrite(D2, 0);
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D3, 1);
  digitalWrite(D3, 0);
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 1);
  digitalWrite(D4, 1);
  digitalWrite(D4, 0);
}
second_digit1 = (second / 1U) % 10;
second_digit2 = (second / 10U) % 10;
minute_digit1 = (minute / 1U) % 10;
minute_digit2 = (minute / 10U) % 10;
hour_digit1 = (hour / 1U) % 10;
hour_digit2 = (hour / 10U) % 10;
if (clock_enable == 1) {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 1);
  digitalWrite(DP, 1);
  digitalWrite(D2, 1);
  digitalWrite(D2, 0);
  digitalWrite(D4, 1);
```

```
digitalWrite(D4, 0);
digitalWrite(DP, 0);
if (second_digit1 == 0)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D6, 1);
  digitalWrite(D6, 0);
}
else if (second_digit1 == 1)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D6, 1);
  digitalWrite(D6, 0);
}
else if (second_digit1 == 2)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D6, 1);
  digitalWrite(D6, 0);
}
else if (second_digit1 == 3)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D6, 1);
  digitalWrite(D6, 0);
}
```

```
else if (second_digit1 == 4)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D6, 1);
  digitalWrite(D6, 0);
}
else if (second_digit1 == 5)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D6, 1);
  digitalWrite(D6, 0);
}
else if (second_digit1 == 6)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D6, 1);
  digitalWrite(D6, 0);
}
else if (second_digit1 == 7)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D6, 1);
  digitalWrite(D6, 0);
}
else if (second_digit1 == 8)  {
  digitalWrite(IC1, 0);
```

```
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 1);
  digitalWrite(D6, 1);
  digitalWrite(D6, 0);
 }
 else if (second_digit1 == 9)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 1);
  digitalWrite(D6, 1);
  digitalWrite(D6, 0);
 }
 if (second_digit2 == 0)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D5, 1);
  digitalWrite(D5, 0);
 }
 else if (second_digit2 == 1)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D5, 1);
  digitalWrite(D5, 0);
 }
 else if (second_digit2 == 2)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 0);
```

```
  digitalWrite(IC8, 0);
  digitalWrite(D5, 1);
  digitalWrite(D5, 0);
}
else if (second_digit2 == 3)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D5, 1);
  digitalWrite(D5, 0);
}
else if (second_digit2 == 4)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D5, 1);
  digitalWrite(D5, 0);
}
else if (second_digit2 == 5)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D5, 1);
  digitalWrite(D5, 0);
}
else if (second_digit2 == 6)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D5, 1);
```

```
  digitalWrite(D5, 0);
}
else if (second_digit2 == 7)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D5, 1);
  digitalWrite(D5, 0);
}
else if (second_digit2 == 8)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 1);
  digitalWrite(D5, 1);
  digitalWrite(D5, 0);
}
else if (second_digit2 == 9)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 1);
  digitalWrite(D5, 1);
  digitalWrite(D5, 0);
}
if (minute_digit1 == 0)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D4, 1);
  digitalWrite(D4, 0);
}
```

```
else if (minute_digit1 == 1)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D4, 1);
  digitalWrite(D4, 0);
}
else if (minute_digit1 == 2)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D4, 1);
  digitalWrite(D4, 0);
}
else if (minute_digit1 == 3)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D4, 1);
  digitalWrite(D4, 0);
}
else if (minute_digit1 == 4)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D4, 1);
  digitalWrite(D4, 0);
}
else if (minute_digit1 == 5)  {
  digitalWrite(IC1, 1);
```

```
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D4, 1);
  digitalWrite(D4, 0);
 }
 else if (minute_digit1 == 6)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D4, 1);
  digitalWrite(D4, 0);
 }
 else if (minute_digit1 == 7)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D4, 1);
  digitalWrite(D4, 0);
 }
 else if (minute_digit1 == 8)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 1);
  digitalWrite(D4, 1);
  digitalWrite(D4, 0);
 }
 else if (minute_digit1 == 9)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
```

```
  digitalWrite(IC8, 1);
  digitalWrite(D4, 1);
  digitalWrite(D4, 0);
 }
 if (minute_digit2 == 0)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D3, 1);
  digitalWrite(D3, 0);
 }
 else if (minute_digit2 == 1)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D3, 1);
  digitalWrite(D3, 0);
 }
 else if (minute_digit2 == 2)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D3, 1);
  digitalWrite(D3, 0);
 }
 else if (minute_digit2 == 3)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D3, 1);
```

```
  digitalWrite(D3, 0);
}
else if (minute_digit2 == 4)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D3, 1);
  digitalWrite(D3, 0);
}
else if (minute_digit2 == 5)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D3, 1);
  digitalWrite(D3, 0);
}
else if (minute_digit2 == 6)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D3, 1);
  digitalWrite(D3, 0);
}
else if (minute_digit2 == 7)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D3, 1);
  digitalWrite(D3, 0);
}
```

```
else if (minute_digit2 == 8)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 1);
  digitalWrite(D3, 1);
  digitalWrite(D3, 0);
}
else if (minute_digit2 == 9)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 1);
  digitalWrite(D3, 1);
  digitalWrite(D3, 0);
}
if (hour_digit1 == 0)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D2, 1);
  digitalWrite(D2, 0);
}
else if (hour_digit1 == 1)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D2, 1);
  digitalWrite(D2, 0);
}
else if (hour_digit1 == 2)  {
  digitalWrite(IC1, 0);
```

```
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D2, 1);
  digitalWrite(D2, 0);
 }
 else if (hour_digit1 == 3)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D2, 1);
  digitalWrite(D2, 0);
 }
 else if (hour_digit1 == 4)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D2, 1);
  digitalWrite(D2, 0);
 }
 else if (hour_digit1 == 5)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D2, 1);
  digitalWrite(D2, 0);
 }
 else if (hour_digit1 == 6)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 1);
```

```
  digitalWrite(IC8, 0);
  digitalWrite(D2, 1);
  digitalWrite(D2, 0);
}
else if (hour_digit1 == 7)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 1);
  digitalWrite(IC4, 1);
  digitalWrite(IC8, 0);
  digitalWrite(D2, 1);
  digitalWrite(D2, 0);
}
else if (hour_digit1 == 8)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 1);
  digitalWrite(D2, 1);
  digitalWrite(D2, 0);
}
else if (hour_digit1 == 9)  {
  digitalWrite(IC1, 1);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 1);
  digitalWrite(D2, 1);
  digitalWrite(D2, 0);
}
if (hour_digit2 == 0)  {
  digitalWrite(IC1, 0);
  digitalWrite(IC2, 0);
  digitalWrite(IC4, 0);
  digitalWrite(IC8, 0);
  digitalWrite(D1, 1);
```

```
    digitalWrite(D1, 0);
   }
   else if (hour_digit2 == 1)  {
    digitalWrite(IC1, 1);
    digitalWrite(IC2, 0);
    digitalWrite(IC4, 0);
    digitalWrite(IC8, 0);
    digitalWrite(D1, 1);
    digitalWrite(D1, 0);
   }
   else if (hour_digit2 == 2)  {
    digitalWrite(IC1, 0);
    digitalWrite(IC2, 1);
    digitalWrite(IC4, 0);
    digitalWrite(IC8, 0);
    digitalWrite(D1, 1);
    digitalWrite(D1, 0);
   }
  }
  if (SPK_ENABLE == 0)
   digitalWrite(SPKpin, 0);
  if (RGBcount == 9)
   RGBcount = 0;
  if (enable_2019)  {
   if (millis() - t >= RGBtime)  {// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება თუ პროგრამის მსვლელობიდან ამ ხაზამდე გასულ დროს გამოკლებული ცვლადი "t" მეტია ცვლად "RGBtime"-ზე ან მისი ტოლია.
    if (Rvalue < 255 & Gvalue == 0 & Bvalue == 0)  {// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება თუ "Rvalue" ნაკლებია 255-ზე და თუ "Gvalue" 0-ის ტოლია და თუ "Bvalue" 0-ის ტოლია.
     analogWrite(Rpin, Rvalue);// ფეხზე, სახელად "Rpin" (3), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Rvalue" ცვლადის მნიშველობის.
     analogWrite(Gpin, Gvalue);// ფეხზე, სახელად "Gpin" (5), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Gvalue" ცვლადის მნიშველობის.
     analogWrite(Bpin, Bvalue);// ფეხზე, სახელად "Bpin" (6), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Bvalue" ცვლადის მნიშველობის.
     Rvalue++;// ცვლად "Rvalue"-ს სიდიდე იზრდება 1 ერთეულით.
```

}

else if (Rvalue > 0 & Gvalue < 255 & Bvalue == 0) {// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება თუ "Rvalue" მეტია 0-ზე და თუ "Gvalue" ნაკლებია 255-ზე და თუ "Bvalue" 0-ის ტოლია.

analogWrite(Rpin, Rvalue);// ფეხზე, სახელად "Rpin" (3), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Rvalue" ცვლადის მნიშველობის.

analogWrite(Gpin, Gvalue);// ფეხზე, სახელად "Gpin" (5), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Gvalue" ცვლადის მნიშველობის.

analogWrite(Bpin, Bvalue);// ფეხზე, სახელად "Bpin" (6), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Bvalue" ცვლადის მნიშველობის.

Rvalue--;// ცვლად "Rvalue"-ს სიდიდე მცირდება 1 ერთეულით.

Gvalue++;// ცვლად "Gvalue"-ს სიდიდე იზრდება 1 ერთეულით.

}

else if (Rvalue == 0 & Gvalue > 0 & Bvalue < 255) {// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება თუ "Gvalue" მეტია 0-ზე და თუ "Bvalue" ნაკლებია 255-ზე და თუ "Rvalue" 0-ის ტოლია.

analogWrite(Rpin, Rvalue);// ფეხზე, სახელად "Rpin" (3), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Rvalue" ცვლადის მნიშველობის.

analogWrite(Gpin, Gvalue);// ფეხზე, სახელად "Gpin" (5), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Gvalue" ცვლადის მნიშველობის.

analogWrite(Bpin, Bvalue);// ფეხზე, სახელად "Bpin" (6), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Bvalue" ცვლადის მნიშველობის.

Gvalue--;// ცვლად "Gvalue"-ს სიდიდე მცირდება 1 ერთეულით.

Bvalue++;// ცვლად "Bvalue"-ს სიდიდე იზრდება 1 ერთეულით.

}

else if (Rvalue < 255 & Gvalue == 0 & Bvalue > 0 & a == 0) {// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება თუ "Bvalue" მეტია 0-ზე და თუ "Rvalue" ნაკლებია 255-ზე და თუ "Gvalue" 0-ის ტოლია და თუ "a" 0-ის ტოლია.

analogWrite(Rpin, Rvalue);// ფეხზე, სახელად "Rpin" (3), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Rvalue" ცვლადის მნიშველობის.

analogWrite(Gpin, Gvalue);// ფეხზე, სახელად "Gpin" (5), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Gvalue" ცვლადის მნიშველობის.

analogWrite(Bpin, Bvalue);// ფეხზე, სახელად "Bpin" (6), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Bvalue" ცვლადის მნიშველობის.

Bvalue--;// ცვლად "Bvalue"-ს სიდიდე მცირდება 1 ერთეულით.

Rvalue++;// ცვლად "Rvalue"-ს სიდიდე იზრდება 1 ერთეულით.

if (Rvalue == 255) // ამ ხაზის შემდეგ დაწერილი ბრძანება შესრულდება თუ "Rvalue" 255-ის ტოლია.

a = 1;// ცვლად "a"-ს მნიშვნელობა ხდება ციფრი 1.

}

else if (Rvalue > 0 & Gvalue < 255 & Bvalue == 0) {// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება თუ "Rvalue" მეტია 0-ზე და თუ "Gvalue" ნაკლებია 255-ზე და თუ "Bvalue" 0-ის ტოლია.

analogWrite(Rpin, Rvalue);// ფეხზე, სახელად "Rpin" (3), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Rvalue" ცვლადის მნიშველობის.

analogWrite(Gpin, Gvalue);// ფეხზე, სახელად "Gpin" (5), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Gvalue" ცვლადის მნიშველობის.

analogWrite(Bpin, Bvalue);// ფეხზე, სახელად "Bpin" (6), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Bvalue" ცვლადის მნიშველობის.

Rvalue--;// ცვლად "Rvalue"-ს სიდიდე მცირდება 1 ერთეულით.

Gvalue++;// ცვლად "Gvalue"-ს სიდიდე იზრდება 1 ერთეულით.

}

else if (Rvalue == 0 & Gvalue > 0 & Bvalue < 255) {// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება თუ "Gvalue" მეტია 0-ზე და თუ "Bvalue" ნაკლებია 255-ზე და თუ "Rvalue" 0-ის ტოლია.

analogWrite(Rpin, Rvalue);// ფეხზე, სახელად "Rpin" (3), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Rvalue" ცვლადის მნიშველობის.

analogWrite(Gpin, Gvalue);// ფეხზე, სახელად "Gpin" (5), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Gvalue" ცვლადის მნიშველობის.

analogWrite(Bpin, Bvalue);// ფეხზე, სახელად "Bpin" (6), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Bvalue" ცვლადის მნიშველობის.

Gvalue--;// ცვლად "Gvalue"-ს სიდიდე მცირდება 1 ერთეულით.

Bvalue++;// ცვლად "Bvalue"-ს სიდიდე იზრდება 1 ერთეულით.

}

else if (Rvalue < 255 & Gvalue < 255 & Bvalue == 255 & a == 1) {// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება თუ "Rvalue" ნაკლებია 255-ზე და თუ "Gvalue" ნაკლებია 255-ზე და თუ "Bvalue" 0-ის ტოლია და თუ "a" 1-ის ტოლია.

analogWrite(Rpin, Rvalue);// ფეხზე, სახელად "Rpin" (3), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Rvalue" ცვლადის მნიშველობის.

analogWrite(Gpin, Gvalue);// ფეხზე, სახელად "Gpin" (5), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Gvalue" ცვლადის მნიშველობის.

analogWrite(Bpin, Bvalue);// ფეხზე, სახელად "Bpin" (6), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Bvalue" ცვლადის მნიშველობის.

Rvalue++;// ცვლად "Rvalue"-ს სიდიდე იზრდება 1 ერთეულით.

Gvalue++;// ცვლად "Gvalue"-ს სიდიდე იზრდება 1 ერთეულით.

if (Rvalue == 255) // ამ ხაზის შემდეგ დაწერილი ბრძანება შესრულდება თუ "Rvalue" 255-ის ტოლია.

a = 0;// ცვლად "a"-ს მნიშვნელობა ხდება ციფრი 0.

}

else if (Rvalue == 255 & Gvalue > 0 & Bvalue > 0) {// ამ ფიგურულ ფრჩხილებში ჩაწერილი ბრძანებები შესრულდება თუ "Rvalue" 255-ის ტოლია და თუ "Gvalue" მეტია 0-ზე და თუ "Bvalue" მეტია 0-ზე.

analogWrite(Rpin, Rvalue);// ფეხზე, სახელად "Rpin" (3), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Rvalue" ცვლადის მნიშველობის.

analogWrite(Gpin, Gvalue);// ფეხზე, სახელად "Gpin" (5), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Gvalue" ცვლადის მნიშველობის.

analogWrite(Bpin, Bvalue);// ფეხზე, სახელად "Bpin" (6), გამოვა PWM სიგნალი (ანალოგურის მსგავსი), რომლის ძაბვაც შესაბამისია "Bvalue" ცვლადის მნიშველობის.

Gvalue--;// ცვლად "Gvalue"-ს სიდიდე მცირდება 1 ერთეულით.

Bvalue--;// ცვლად "Bvalue"-ს სიდიდე მცირდება 1 ერთეულით.

}

t = millis();// პროგრამის მსვლელობიდან ამ ბრძანებამდე გასული დრო ინახება "t" ცვლადად.

}

}

if (enable_2019_3) {

COUNTER = 1;

if (RGBcount == 0) {

analogWrite(Rpin, 255);

analogWrite(Gpin, 0);

analogWrite(Bpin, 0);

SPK_T = ((1 / ((((480 + 400) / 2) / 770) * 20000)) * 1000) / 2;

}

else if (RGBcount == 1) {

analogWrite(Rpin, 255);

analogWrite(Gpin, 127);

analogWrite(Bpin, 0);

SPK_T = ((1 / ((((505 + 480) / 2) / 770) * 20000)) * 1000) / 2;

}

else if (RGBcount == 2) {

analogWrite(Rpin, 255);

analogWrite(Gpin, 255);

analogWrite(Bpin, 0);

SPK_T = ((1 / ((((521 + 512) / 2) / 770) * 20000)) * 1000) / 2;

}

else if (RGBcount == 3) {

```
    analogWrite(Rpin, 0);
    analogWrite(Gpin, 255);
    analogWrite(Bpin, 0);
    SPK_T = ((1 / ((((575 + 525) / 2) / 770) * 20000)) * 1000) / 2;
  }
  else if (RGBcount == 4) {
    analogWrite(Rpin, 0);
    analogWrite(Gpin, 0);
    analogWrite(Bpin, 255);
    SPK_T = ((1 / ((((670 + 610) / 2) / 770) * 20000)) * 1000) / 2;
  }
  else if (RGBcount == 5) {
    analogWrite(Rpin, 0);
    analogWrite(Gpin, 127);
    analogWrite(Bpin, 255);
    SPK_T = ((1 / ((((620) / 770) / 2) * 20000)) * 1000) / 2;
  }
  else if (RGBcount == 6) {
    analogWrite(Rpin, 255);
    analogWrite(Gpin, 0);
    analogWrite(Bpin, 255);
    SPK_T = ((1 / ((((790 + 666) / 2) / 770) * 20000)) * 1000) / 2;
  }
  else if (RGBcount == 7) {
    analogWrite(Rpin, 255);
    analogWrite(Gpin, 255);
    analogWrite(Bpin, 255);
    SPK_T = ((1 / (((((480 + 400) / 2) + ((575 + 525) / 2) + ((670 + 610) / 2)) / 3 / 770) * 20000)) * 1000) / 2;
  }
  else if (RGBcount == 8) {
    RGBcount = 0;
  }
}
if (morse_enable)  {
```

```
if (c == 0)  {
  SPK_ENABLE = 1;
  SPK_T = 0.5;
  d = 0;
  c = 1;
}
if (millis() - t7 >= 150 & c == 1) {
  SPK_ENABLE = 0;
  d = 1;
  t7 = millis();
}
if (d == 1)
  c = 2;
if (millis() - t7 >= 150 & c == 2) {
  SPK_ENABLE = 1;
  d = 2;
  t7 = millis();
}
if (d == 2)
  c = 3;
if (millis() - t7 >= 150 & c == 3) {
  SPK_ENABLE = 0;
  d = 3;
  t7 = millis();
}
if (d == 3)
  c = 4;
if (millis() - t7 >= 150 & c == 4) {
  SPK_ENABLE = 1;
  d = 4;
  t7 = millis();
}
if (d == 4)
  c = 5;
```

```
if (millis() - t7 >= 450 & c == 5)  {
  SPK_ENABLE = 0;
  d = 5;
  t7 = millis();
}
if (d == 5)
  c = 6;
if (millis() - t7 >= 150 & c == 6)  {
  SPK_ENABLE = 1;
  d = 6;
  t7 = millis();
}
if (d == 6)
  c = 7;
if (millis() - t7 >= 450 & c == 7)  {
  SPK_ENABLE = 0;
  d = 7;
  t7 = millis();
}
if (d == 7)
  c = 8;
if (millis() - t7 >= 150 & c == 8)  {
  SPK_ENABLE = 1;
  d = 8;
  t7 = millis();
}
if (d == 8)
  c = 9;
if (millis() - t7 >= 450 & c == 9)  {
  SPK_ENABLE = 0;
  d = 9;
  t7 = millis();
}
if (d == 9)
```

```
  c = 10;
if (millis() - t7 >= 450 & c == 10)  {
  SPK_ENABLE = 1;
  d = 10;
  t7 = millis();
}
if (d == 10)
  c = 11;
if (millis() - t7 >= 450 & c == 11)  {
  SPK_ENABLE = 0;
  d = 11;
  t7 = millis();
}
if (d == 11)
  c = 12;
if (millis() - t7 >= 150 & c == 12)  {
  SPK_ENABLE = 1;
  d = 12;
  t7 = millis();
}
if (d == 12)
  c = 13;//
if (millis() - t7 >= 450 & c == 13)  {
  SPK_ENABLE = 0;
  d = 13;
  t7 = millis();
}
if (d == 13)
  c = 14;
if (millis() - t7 >= 150 & c == 14)  {
  SPK_ENABLE = 1;
  d = 14;
  t7 = millis();
}
```

```
if (d == 14)
  c = 15;//
if (millis() - t7 >= 450 & c == 15)  {
  SPK_ENABLE = 0;
  d = 16;
  t7 = millis();
}
if (d == 16)
  c = 17;
if (millis() - t7 >= 150 & c == 17)  {
  SPK_ENABLE = 1;
  d = 17;
  t7 = millis();
}
if (d == 17)
  c = 18;//
if (millis() - t7 >= 450 & c == 18)  {
  SPK_ENABLE = 0;
  d = 18;
  t7 = millis();
}
if (d == 18)
  c = 19;
if (millis() - t7 >= 150 & c == 19)  {
  SPK_ENABLE = 1;
  d = 19;
  t7 = millis();
}
if (d == 19)
  c = 20;//
if (millis() - t7 >= 450 & c == 20)  {
  SPK_ENABLE = 0;
  d = 20;
  t7 = millis();
```

```
}
if (d == 20)
  c = 21;//
if (millis() - t7 >= 450 & c == 21)  {
  SPK_ENABLE = 1;
  d = 21;
  t7 = millis();
}
if (d == 21)
  c = 22;
if (millis() - t7 >= 150 & c == 22)  {
  SPK_ENABLE = 0;
  d = 22;
  t7 = millis();
}
if (d == 22)
  c = 23;//
if (millis() - t7 >= 150 & c == 23)  {
  SPK_ENABLE = 1;
  d = 23;
  t7 = millis();
}
if (d == 23)
  c = 24;
if (millis() - t7 >= 450 & c == 24)  {
  SPK_ENABLE = 0;
  d = 24;
  t7 = millis();
}
if (d == 24)
  c = 25;
if (millis() - t7 >= 150 & c == 25)  {
  SPK_ENABLE = 1;
  d = 25;
```

```
  t7 = millis();
}
if (d == 25)
  c = 26;
if (millis() - t7 >= 450 & c == 26)  {
  SPK_ENABLE = 0;
  d = 26;
  t7 = millis();
}
if (d == 26)
  c = 27;//
if (millis() - t7 >= 150 & c == 27)  {
  SPK_ENABLE = 1;
  d = 27;
  t7 = millis();
}
if (d == 27)
  c = 28;
if (millis() - t7 >= 450 & c == 28)  {
  SPK_ENABLE = 0;
  d = 28;
  t7 = millis();
}
if (d == 28)
  c = 29;//
if (millis() - t7 >= 150 & c == 29)  {
  SPK_ENABLE = 1;
  d = 29;
  t7 = millis();
}
if (d == 29)
  c = 30;
if (millis() - t7 >= 450 & c == 30)  {
  SPK_ENABLE = 0;
```

```
  d = 30;
  t7 = millis();
}
if (d == 30)
  c = 31;
if (millis() - t7 >= 450 & c == 31)  {
  SPK_ENABLE = 1;
  d = 31;
  t7 = millis();
}
if (d == 31)
  c = 32;
if (millis() - t7 >= 450 & c == 32)  {
  SPK_ENABLE = 0;
  d = 32;
  t7 = millis();
}
if (d == 32)
  c = 33;//1
if (millis() - t7 >= 150 & c == 33)  {
  SPK_ENABLE = 1;
  d = 33;
  t7 = millis();
}
if (d == 33)
  c = 34;
if (millis() - t7 >= 450 & c == 34)  {
  SPK_ENABLE = 0;
  d = 34;
  t7 = millis();
}
if (d == 34)
  c = 35;//2
if (millis() - t7 >= 150 & c == 35)  {
```

```
  SPK_ENABLE = 1;
  d = 35;
  t7 = millis();
}
if (d == 35)
  c = 36;
if (millis() - t7 >= 450 & c == 36)  {
  SPK_ENABLE = 0;
  d = 36;
  t7 = millis();
}
if (d == 36)
  c = 37;//3
if (millis() - t7 >= 150 & c == 37)  {
  SPK_ENABLE = 1;
  d = 37;
  t7 = millis();
}
if (d == 37)
  c = 38;
if (millis() - t7 >= 450 & c == 38)  {
  SPK_ENABLE = 0;
  d = 38;
  t7 = millis();
}
if (d == 38)
  c = 39;//4
if (millis() - t7 >= 150 & c == 39)  {
  SPK_ENABLE = 1;
  d = 39;
  t7 = millis();
}
if (d == 39)
  c = 40;
```

```
    if (millis() - t7 >= 150 & c == 40)  {
      SPK_ENABLE = 0;
      d = 40;
      t7 = millis();
    }
    if (d == 40)
      c = 41;
  }
  if (SPK_ENABLE == 1)  {
    if (millis() - t6 >= SPK_T)  {
      SPKstate = !SPKstate;
      digitalWrite(SPKpin, SPKstate);
      t6 = millis();
    }
  }
  if (COUNTER == 1) {
    digitalWrite(Vcc, 1);// ფეხზე, სახელად "Vcc" (13), გამოდის ლოგიკური 1.
    if (b == 0) {
      C_TIME = 500;
    }
    else if (millis() - t2 >= 4000 & C_TIME >= 0.9765625) {
      C_TIME /= 2;
      t2 = millis();// პროგრამის მსვლელობიდან ამ ბრძანებამდე გასული დრო ინახება "t" ცვლადად.
    }
    if (millis() - t5 >= C_TIME)  {
      digitalWrite(C_LED, 1);
      digitalWrite(C_LED, 0);
      RGBcount++;
      b = 1;
      t5 = millis();// პროგრამის მსვლელობიდან ამ ბრძანებამდე გასული დრო ინახება "t" ცვლადად.
    }
  }
}
```