PROGRAMIRANJE

je sestavljanje sosledja ukazov / operacij, ki jih izbrani računalnik (mikrokontroler) lahko prevede v avtomatsko delovanje. Človek programira z "višjimi jeziki", ki so prilagojeni njegovemu načinu razmišljanja. Ti zapisi se za uporabo v računalniku prevedejo v "nižji" jezik – strojno kodo.

Programsko okolje Arduino (**Arduino IDE**) uporablja jezik, ki je zelo podoben c++ (izg.: "ce plus plus"). Je eden najbolj razširjenih višjih programskih jezikov, tako da je dober razlog za učenje programiranja. "Prekriva" (abstrahira) precej "nižji" jezik (set instrukcij) imenovan Atmel AVR – namenjen programiranju Atmel mikrokontrolerjev, ki so tudi osnova skoraj vseh ploščic Arduino. Za te mikrokontrolerje se kot končni prevajalnik (ang.: "compiler") uporablja **avrdude**, ki je del Arduino IDE.

https://www.arduino.cc/

Arduino IDE je sestavljen iz menuja, kjer so predvidljivo razporejeni različni segmenti. Pod menujem Tools imamo opcijo Boards, kjer izberemo mikrokontroler, ki ga uporabljamo (→ Arduino/ Genuino UNO). Po priključitvi arduino ploščice s pomočjo USB povezave izberemo ustrezna serijska vrata (vhod in izhod) → Port.

V vrstici nižje so ikone za najbolj pogosto uporabljane funkcije programa:
- Verify: prevedi in preveri kodo
- Upload: prevedi in preveri kodo ter pošlji v mikrokontroler
- New: začni novi prazni program
- Open: odpri obstoječi program
- Save: shrani trenutni program

Na skrajni desni je še ikona
- Serial Monitor: odpri okno za prikaz serijske povezave

Zgradba programa v Arduino IDE se vzpostavi sama, ko kliknemo FILE→NEW.

Spodaj je programček za Arduino IDE v katerega smo s pomočjo komentarjev dopisali nekaj najbolj značilnih elementov programa in nekaj osnovnih definicij: konstant, spremenljivk in funkcij. Definicije so vse kar prejme enoznačno ime. Ime je lahko je dokaj poljubno, ne sme pa se podvajati ali biti enako rezerviranim imenom.

```
/********/

/*
spodnja in zgornja dvojica znakov pomenita,
da je vmes večvrstični komentar.
Komentarji so pomembni, saj pojasnjujejo, kaj kosi programa počnejo
*/

// ta dvojica poševnic označuje enovrstični komentar

// izven funkcij je mesto definicij morebitnih konstant in spremenljivk
```

```
// definicija je potrebna, da rezerviramo ustrezem prostor v pomnilniku
// zato je nujno tudi definirati tip podatka

// te konstante in spremenljivke so “globalne” – dostopne povod
// v funkcijah jih spreminjamo, spremenjene vrednosti pa ohranjajo

/*
// konstante v programu ne moremo spreminjati!
*/
// uporabljamo jih največ za definicijo nožic (pinov)
const int STOJ = 5; // nožica 5 bo določala, ali štetje ustavimo ali nadaljujemo

// logična (boolean) konstanta
const boolean KONSTANTA = false; // True ali False

// konstanta celoštevilčnega tipa – med -32768 in  32767 (obsega 2 besedi / byta – 2 x 8 bitov = 16 bitov)
const int CELOSTEVILO= 60; // cela števila – negativna in pozitivna ter 0

// logična spremeljivka
boolean STEJ = true; // True ali False

// Arduino UNO ima na nožici 13 LED svetilko
// uporabimo jo kot indikator štetja
int LED = 13;

/*
// spremeljivke lahko v programu spreminjamo
*/

// celoštevilčna spremenljivka
int STEVEC = 0;

// beseda void (“prazno”) pomeni, da ta funkcija ni namenjena vračanju spremeljivk
// to je obvezna vgrajena funkcija nastavitve pogojev na začetku izvajanja programa

void setup() {
  // put your setup code here, to run once:
  // funkcija setup() je obvezna in označuje del programa, ki se izvrši najprej in le enkrat
  // lahko je tudi prazna

  // definirajmo nožico STOJ kot vhod z notranjim pull-up uporom vključenim
  pinMode(STOJ, INPUT_PULLUP);
  // zunanje stikalo s preklopom na maso bo visok nivo spremenilo v nizek
  // kar bo znak za zaustavitev

  // definirajmo nožico LED kot izhod
  pinMode(LED, OUTPUT);
```

```
  // odprimo serijsko povezavo za pošiljanje številk
  //  Serial je interni skupek metod / funkcij
  // ki se tičejo serijske povezave
  Serial.begin(9600);       // use the serial port to print the number

}

// beseda void (“prazno”) pomeni, da ta funkcija ni namenjena vračanju spremeljivk
// to je obvezna vgrajena fukcija neprestanega ponavljanja

void loop() {
  // put your main code here, to run repeatedly:
  // funkcija loop() se začne izvrševati takoj po setup() in se neprestano ponavlja

 // preverimo, ali smo ukazali zaustavitev števca
 // stanje na vhodi pina STOJ bo nizko (LOW ali 0), če stikalo preklopimo
 //  PREKLOP je lokalna spremenljivka
 int PREKLOP  = digitalRead(STOJ);

 if (PREKLOP) {
     // nismo preklopili na maso (0), saj je PREKLOP visok (1)
     // štej naprej
     STEJ = true;
 } else {
   // preklopili smo na maso (0), zato je PREKLOP nizek (0)
    // prekini štetje
    STEJ = false;
 }

 if (!STEJ) {
    // pogojni stavek: če ni STEJ = true (torej je STEJ = false → preskoči vse, kar je v nadaljevanju
    return;
 }
 // sicer pa nadaljujemo

  // prištej 1 k prejšnji vrednosti
 STEVEC++;              //Adds 1 to the countUp int on every loop

 // pošljimo številko prek serijske povezave
 Serial.println(STEVEC );  // prints out the current state of countUp

 // interna arduino funkcija, ki ukaže zaustavitev za 0,9 sekunde (= 900 milisekund)
 delay(900);

 // vključimo in izključimo LED
 digitalWrite(LED, 1);

 delay(100); // 1/10 sekunde = 100 milisekund
 // skupaj z zgornjo časovno zaustavitvijo je čas 1 sekunda
```

```
 digitalWrite(LED, 0);

 // preverimo, če je STEVEC naštel 60 sekund → >= večje ali enako
 // če bi hoteli preveriti zgolj enako, bi napisali STEVEC == CELOSTEVILO
 if (STEVEC >= CELOSTEVILO) {
   // postavimo STEVEC spet na 0
   STEVEC = 0;

   // če bi hoteli skociti na naključno število, bi lahko uporabili spodnjo funkcijo
   STEVEC = poljubno();

 } // konec if

} // konec loop()

// beseda int (kot integer "celo število") pomeni, da ta funkcija vrne celoštevilno vrednost
int poljubno() {
  // uporabimo vgrajeno funkcijo random() , ki vrne vrednost tipa "long" (16-bitno številko)
  // random(max) – v tej obliki dobimo naključno število med 0 in max-1
  // random(min, max) – v tej obliki dobimo naključno število med min in max-1

  // mi uporabimo podatek CELOSTEVILO
  // prav tako nas namesto "long" zanima omejeno število "int"
  // zato namesto "long" vsilimo "int" podatkovni tip – spredaj v oklepaju
  int rand = (int) random(CELOSTEVILO);

  // pošljimo številko prek serijske povezave
  Serial.print("STEVEC naključno spremenjen na ");  // prints out the number
  Serial.println(rand);  // prints out the number

  // vrni izračunano poljubno vrednost
  return rand;

}
/********/
```

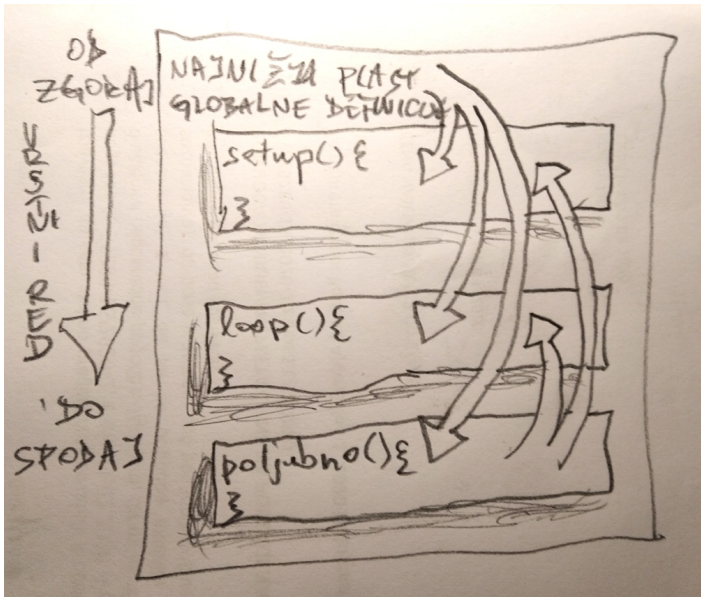
Na primeru smo videli, da je zelo pomembno pisati kodo strukturirano – uvrščati predvidljive leve odmike, saj so oznake za sestavljene operacije in funkcije vsebovane v zavitih oklepajih {}.

Prav tako so zelo pomembni komentarji, saj nam povejo razloge za posamezne ukaze.

Geometrična struktura programa je v splošnem:
1. po plasteh vertikalno glede na površino zapisa:
- najnižje so definicije globalnih spremenljivk, vključitev knjižnic
- malo višje so funkcije v zavitih oklepajih {} → setup(){}, loop(){} in naše definicije funkcij (poljubno(){})
- najvišje ležijo so v teh funkcijah definirane lokalne spremenljivke in pogojni stavki → if, for while, itd.

Višje ali istoležeče vertikalne plasti lahko uporabijo in spreminjajo vse naše definicije v nižje ležečih plasteh. Izjema so vgrajene funkcije in ključne besede, ki jih (pogosto) ne smemo zlorabiti. Te se vedno obarvajo drugače.

2. po smeri zapisa od zgoraj navzdol:
- globalne ali lokalne spremenljivke / definicije morajo biti definirane pred uporabo v funkcijah, ki jih uporabljajo
- funkcije v zavitih oklepajih lahko uporabljajo druge funkcije (razen setup(){} in loop(){}) - ne glede na lego spodaj ali zgoraj. Običjno jih pišemo čisto spodaj (za loop(){}).



Arduino IDE ima naslednje osnovne vgrajene funkcije, ki so hkrati tudi rezervirana imena.

Digital I/O:
digitalRead()
digitalWrite()
pinMode()

Analog I/O:
analogRead()
analogReference()
analogWrite()

Advanced I/O:
tone()
noTone()
pulseIn()
pulseInLong()
shiftIn()
shiftOut()

Time:
delay()
delayMicroseconds()
micros()
millis()

Math:
abs()
constrain()
map()
max()
min()
pow()
sq()
sqrt()

Trigonometry:
cos()
sin()
tan()

Characters:
isAlpha()
isAlphaNumeric()
isAscii()
isControl()
isDigit()
isGraph()
isHexadecimalDigit()
isLowerCase()
isPrintable()
isPunct()
isSpace()
isUpperCase()
isWhitespace()

Random Numbers:
random()
randomSeed()

Bits and Bytes:
bit()
bitClear()
bitRead()
bitSet()
bitWrite()
highByte()
lowByte()

External Interrupts:
attachInterrupt()
detachInterrupt()

Interrupts:
interrupts()
noInterrupts()

Communication:
Serial
Stream

USB:
Keyboard
Mouse