Merjenje časa in globalne spremenljivke

Naučili se bomo, kako se izogniti funkciji `delay` (in zakaj bi to sploh hoteli). Videli bomo, kako nam pridejo prav globalne spremenljivke: medtem ko v “običajnem programiranju” veljajo za slabo prakso, jih pri programiranju Arduina uporabljamo zato, da med različnimi izvedbami `loop`-a ne “izgubi spomina” in ve, v kakšnem stanju je in kdaj mora kaj storiti. Boste videli.

Dve tipki in dve diodi

Naloga 1. Napiši program, ki takrat, ko je pritisnjena tipka, prižge diodo in jo ugasne čez tri sekunde.

Naloga 2. Zdaj pa dodaj v ta program še drugo tipko, ki prižiga drugo diodo.

Ti je uspelo? Morda. Prepričajmo se, da res. Pritisni tipko. Medtem ko dioda sveti, pritisni še drugo tipko. Se je druga dioda prižgala (in zakaj ne :)?

Arduino čepi v ukazu `delay`. Čaka. Tri sekunde ne počne ničesar. Ne prižiga, ne ugaša - in tudi tipk ne gleda. Program miruje. Ukaz `delay` je bolj zoprna zadeva, kot smo si mislili.

Namesto ukaza `delay`

Da si poenostavimo razmišljanje, se bomo spet ukvarjali z eno samo tipko. Ko jo pritisnemo, želimo prižgati diodo in jo čez 3 sekunde ugasniti. Pri tem - v izogib sitnostim - nočemo uporabiti ukaza `delay`.

Bojni načrt je takšen:

```
void setup() {
    nastavi pin mode, kot je potrebno
}

void loop() {
    če je tipka pritisnjena {
        prižgi diodo
        zapomni si, kdaj jo moraš ugasniti (trenutni čas + 3 sekunde)
    }

    če je že čas za ugašanje diode {
        ugasni diodo
    }
}
```

Ob tem programu se moramo zavedati, da Arduino stalno ponavlja, kar piše v `loop`. Tako bo stalno pregledoval oba pogoja.

Kaj od gornjega že znamo napisati? Skoraj vse.

```
void setup() {
    pinMode(3, OUTPUT);
    pinMode(10, INPUT_PULLUP);
}

void loop() {
    if (digitalRead(10) == LOW) {
        digitalWrite(3, HIGH);
        zapomni si, kdaj jo moraš ugasniti (trenutni čas + 3 sekunde)
    }

    če je že čas za ugašanje diode {
        digitalWrite(3, LOW);
    }
}
```

Kako pa Arduino pogleda na uro? Uporabiti moramo `millis()`: ta pove, koliko milisekund je minilo, odkar smo prižgali Arduina. To sicer ni prav zares ura (petnajst čez tri), a za naše potrebe bo zadoščala.

Morda tako?

```
void setup() {
    pinMode(3, OUTPUT);
    pinMode(10, INPUT_PULLUP);

}
```

```
void loop() {
    int ugasni_ob;

    if (digitalRead(10) == LOW) {
        digitalWrite(3, HIGH);
        ugasni_ob = millis() + 3000;
    }

    if (millis() >= ugasni_ob) {
        digitalWrite(3, LOW);
    }
}
```

Tole ne deluje. Spremenljivka `ugasni_ob` se na novo ustvari vsakič, ko Arduino izvede, kar piše znotraj `loop`. To ne bo šlo: vsa ideja našega programa je v tem, da Arduino vedno znova izvaja `loop` in vsakič primerja `millis()` z `ugasni_ob`. Če se `ugasni_ob` ustvari vsakič znova (in ima kar neko vrednost), to ne bo dobro.

```
int ugasni_ob;

void setup() {
    pinMode(3, OUTPUT);
    pinMode(10, INPUT_PULLUP);
    ugasni_ob = 0;
}

void loop() {
    if (digitalRead(10) == LOW) {
        digitalWrite(3, HIGH);
        ugasni_ob = millis() + 3000;
    }

    if (millis() >= ugasni_ob) {
        digitalWrite(3, LOW);
    }
}
```

Veliko boljše. Zdaj smo `ugasni_ob` ustvarili izven `loop` in kar zapišemo vanjo, se ohranja med različnimi "izvajanji" tega, kar je v `loop`. Začetno vrednost `ugasni_ob` smo nastavili na 0. Zaradi lepšega.

Takšnim spremenljivkam pravimo *globalne spremenljivke*, saj so vidne v vseh blokih - `setup`, `loop` in takšnih, ki jih bomo morda še napisali.

Dokler ne pritisnemo tipke, bo `ugasni_ob` enak 0 in Arduino bo vedno znova ugašal diodo. Nič hudega. Ko pritisnemo tipko, jo prižge in ponovno jo bo začel ugašati, ko bo čas za to.

Zakaj pišemo `millis() >= ugasni_ob` in ne `millis() == ugasni_ob`? Saj menda

hočemo ugasniti diodo *točno takrat* in ne malo kasneje? To je potrebno, ker se lahko zgodi, da bo Arduino zamudil točen čas.

Poženi program in ga preskusi.

Najbrž boš opazil, da nekaj časa deluje, potem pa nenadoma neha. Težava je tule: spremenljivke vrste `int` lahko shranjujejo števila od -32768 do 32767. Če je Arduino prižgan že 50 sekund, `millis()` vrne 50000. Ko poskušamo to shraniti v `prizgal_ob`, se shrani … kaj drugega. Ni važno kaj, a 50000 prav gotovo ne, saj tako veliki števil ne more shranjevati.

Kako naredimo spremenljivke za večje številke? Namesto `int` napišemo `long`.

```
long ugasni_ob;

void setup() {
    pinMode(3, OUTPUT);
    pinMode(10, INPUT_PULLUP);
    ugasni_ob = 0;
}

void loop() {
    if (digitalRead(10) == LOW) {
        digitalWrite(3, HIGH);
        ugasni_ob = millis() + 3000;
    }

    if (millis() >= ugasni_ob) {
        digitalWrite(3, LOW);
    }
}
```

V `ugasni_ob` bodo šle številke od -2 milijardi do 2 milijardi. To bo dovolj za 2 milijona sekund ali 555 ur ali 23 dni. Dovolj za naše potrebe.

Naloga 3 Spremeni program tako, da bo delal z dvema diodama.

Arduino si zapomni stanje

Znamo napisati program, ki bo štel pritiske na tipko? Ob prvem pritisku na tipko prižge prvo diodo. Ob drugem prižge še eno več, ob tretjem gorijo že tri in tako naprej…

Potrebovali bomo nekaj takega:

```
void setup() {
    int i = 3;
    while(i < 10) {
        pinMode(i, OUTPUT);
        i += 1;
    }
    pinMode(10, INPUT_PULLUP);
}

void loop() {
    if (digitalRead(10) == LOW) {
        prižgi naslednjo diodo;
    }
}
```

Pravzprav je program že skoraj napisan, le “prižgi naslednjo diodo” še manjka. Da bi lahko “prižgali naslednjo” si moramo zapomniti, do kje smo prižgali doslej. Tako kot smo v prejšnjem programu uvedli stalno, globalno spremenljivko, ki je povedala, kdaj je potrebno ugasniti diodo, bomo zdaj uvedli spremenljivko, ki bo beležila, recimo, koliko je prižganih diod.

```
int prizganih;

void setup() {
    int i = 3;
    while(i < 10) {
        pinMode(i, OUTPUT);
        i += 1;
    }
    pinMode(10, INPUT_PULLUP);
    prizganih = 0;
}

void loop() {
    if (digitalRead(10) == LOW) {
        prizganih += 1;
        digitalWrite(2 + prizganih, HIGH);  // če je prva dioda na pinu
3
    }
}
```

Če preskusite program, boste opazili, da ne deluje čisto dobro: že ob prvem pritisku se najbrž prižgejo vse diode. Arduino je, kot smo opazili že, ko smo prvič delali z njim, hiter. Če tipko pritisnemo za še tako kratek čas, bo Arduino v tem času večkrat izvedel `loop` in tako prižgal več diod.

Lahko bi si pomagali takole

```
    if (digitalRead(10) == LOW) {
        prizganih += 1;
        digitalWrite(2 + prizganih, HIGH);
        while (digitalRead(10) == LOW) {
          // dokler je tipka pritisnjena, počakaj - ne delaj nič
        }
    }
```

Vendar to ne deluje iz čisto mehanskih razlogov: ko pritisnemo tipko, je stik v prvih trenutkih slab, zato signal na pinu 10 zaniha med LOW in HIGH, kot da bi tipko zelo na hitro pritisnili večkrat. Zato bo najboljše, da dodamo kratko pavzo, recimo 100 ali 200 milisekund.

```
int prizganih;

void setup() {
    int i = 3;
    while(i < 10) {
        pinMode(i, OUTPUT);
        i += 1;
    }
    pinMode(10, INPUT_PULLUP);
    prizganih = 0;
}

void loop() {
    if (digitalRead(10) == LOW) {
        prizganih += 1;
        digitalWrite(2 + prizganih, HIGH);  // če je prva dioda na pinu
3
        delay(200);
    }
}
```

Naloga 4. Dopolni program: če so prižgane vse diode in pritisnemo tipko, naj se diode ugasnejo. Ob naslednjem pritisku naj potem spet gori ena dioda.

Naloga 5. Gori le ena dioda naenkrat. Vsakič, ko pritisnemo tipko, se prižge naslednja dioda, trenutno prižgana pa ugasne. Če gori zadnja dioda in pritisnemo tipko, naj se spet prižge prva.

Naloga 6. Dodaj še eno tipko. Če pritisnemo prvo tipko, se prižge dioda, ki je desno od trenutno prižgane, če drugo, pa dioda, ki je levo.

Naloga 7. Da ne pozabimo na tisto, kar smo počeli v začetku: napiši program, ki po vrsti prižiga diode. Najprej prvo, sekundo kasneje ugasne prvo in prižge drugo, sekundo kasneje ugasne drugo in prižge tretjo … in tako naprej. Za zadnjo diodo spet sledi prva. To smo počeli že davno, zdaj pa to naredi brez `delay`.

Arduino si zapomni stanje in smer

V prejšnjem poglavju smo spogramirali Arduino tako, da je “štel” pritiske na tipko. V prvi nalogi ste ga dopolnili tako, da je takrat, ko so vse diode prižgane, ob pritisku na tipko ugasnil diode in začel od začetka. Zdaj pa ga spremenimo program tako, da se bodo diode ob pritiskanju na tipko najprej ugašale, ko svetijo vse, pa naj se po pritisku na tipko ugašajo.

Zdaj si mora Arduino zapomniti dve stvari: do katere diode smo prišli in v katero smer gremo - ali jih prižigamo ali ugašamo.

Pa si zapomnimo smer kar kot `+1` (prižigam) ali `-1` (ugašam). V začetku je prižganih 0 diod, smer pa je `+1`. Enkrat za spremembo nastavimo obe vrednosti kar tam, kjer ustvarimo spremenljivki - da bomo videli, da gre tudi to. :)

```
int prizganih = 0;
int smer = 1;

void setup() {
    int i = 3;
    while(i < 10) {
        pinMode(i, OUTPUT);
        i += 1;
    }
    pinMode(10, INPUT_PULLUP);
}

void loop() {
    if (digitalRead(10) == LOW) {
        if (smer == 1) {
            prizganih += 1;
            digitalWrite(2 + prizganih, HIGH);
            if (prizganih == 7) {
                smer = -1;
            }
        }
        else {
            digitalWrite(2 + prizganih, LOW);
            prizganih -= 1;
            if (prizganih == 0) {
                smer = 1;
            }
        }
        delay(200);
    }
}
```

Program morda niti ni tako zapleten, kot je siten - zaradi obračanja. Začetnik se lahko ob tem

pošteno zaplete. Bodite pozorni na vrstni red vrstic.

- Ko prižigamo, najprej povečamo število prižganih diod (recimo z 2 na 3) in potem prižgemo to (recimo tretjo) diodo. Če ugotovimo, da so prižgane že vse, zabeležimo, da jih bo potrebno zdaj ugašati.
- Ko ugašamo, najprej ugasnemo zadnjo prižgano diodo. Šele nato zmanjšamo `prizganih`. Če ugotovimo, da smo ugasnili vse, si zabeležimo, da jih bomo zdaj prižigali.

Naloga 8. Diode se prižigajo od leve proti desni, z eno sekundo pavze, ne da bi morali kaj pritiskati. Če pritisnemo tipko, se smer prižiganja obrne, tako da gredo od desne proti levi. Če ponovno pritisnemo, se spet obrnejo.

Naloga 9. Podobno kot prej, a z dvema tipkama. Če pritisnemo eno tipko, se prižigajo proti desni, če drugo, proti levi.

Namigi

Naloga 1. Če ti kaj pomaga, je tu program v slovenščini:

```
če je pritisnjena tipka
    prižgi diodo
    počakaj tri sekunde
    ugasni diodo
```

Le to prevedi v jezik, ki ga govori (no, bere) Arduino.

Naloga 2. Spet le podvoji program, napisan za eno diodo.

Naloga 3. Potreboval boš dve spremenljivki: ena pove, kdaj se ugasne prva dioda, druga pa, kdaj se ugasne druga. V bistvu le podvojiš program za eno diodo.

Naloga 4. Ko bi hotel prižgati zadnjo diodo, ugasni vse in postavi vrednosti spremenljivk na začetne.

Naloga 5. Kot prej, le da ugašaš za sabo. Nasvet: ugasni, povečaj, prižgi. V tem vrstnem redu. Če povečaš čez mejo, greš na 0.

Naloga 6. Kot prej, le da povečaš ali zmanjšaš, in paziš na obe meji.

Naloga 7. Spet boš potreboval `ugasni_ob`, le da ga zdaj lahko imenuješ `naprej_ob`. V `if` bo namesto preverjanja tipke preverjanje časa. In ko prižgeš naslednjo diodo, nastaviš nov `naprej_ob`.

Naloga 8. in 9. Tole le združuje več gornjih nalog.

Rešitve

Naloga 1

```
void setup() {
    pinMode(3, OUTPUT);
    pinMode(10, INPUT_PULLUP);
}

void loop() {
    if (digitalRead(10) == LOW) {
        digitalWrite(3, HIGH);
        delay(3000);
        digitalWrite(3, LOW);
    }
}
```

Naloga 2

```
void setup() {
    pinMode(3, OUTPUT);
    pinMode(4, OUTPUT);
    pinMode(10, INPUT_PULLUP);
    pinMode(11, INPUT_PULLUP);
}

void loop() {
    if (digitalRead(10) == LOW) {
        digitalWrite(3, HIGH);
        delay(3000);
        digitalWrite(3, LOW);
    }
    if (digitalRead(11) == LOW) {
        digitalWrite(4, HIGH);
        delay(3000);
        digitalWrite(4, LOW);
    }
}
```

Naloga 3

```
void setup() {
    pinMode(3, OUTPUT);
    pinMode(4, OUTPUT);
    pinMode(10, INPUT_PULLUP);
    pinMode(11, INPUT_PULLUP);
}

long ugasni_ob_3 = 0;
long ugasni_ob_4 = 0;

void loop() {
    if (digitalRead(10) == LOW) {
        digitalWrite(3, HIGH);
        ugasni_ob_3 = millis() + 3000;
    }
    if (millis() >= ugasni_ob_3) {
        digitalWrite(3, LOW);
    }

    if (digitalRead(11) == LOW) {
        digitalWrite(4, HIGH);
        ugasni_ob_4 = millis() + 3000;
    }
    if (millis() >= ugasni_ob_4) {
        digitalWrite(4, LOW);
    }
}
```

Naloga 4

```
int prizganih;

void setup() {
    int i = 3;
    while(i < 10) {
        pinMode(i, OUTPUT);
        i += 1;
    }
    pinMode(10, INPUT_PULLUP);
    prizganih = 0;
}

void loop() {
    if (digitalRead(10) == LOW) {
        prizganih += 1;
        if (prizganih == 8) {
            // če prižigamo "osmo" diodo (ki je ni), ugasnemo vseh sedem
            int i = 1;
            while (i < 8) {
                digitalWrite(2 + i, LOW);
                i += 1;
            }
            prizganih = 1; // in potem naj bo spet prižgana ena
        }
        digitalWrite(2 + prizganih, HIGH);
        delay(200);
    }
}
```

Naloga 5

```
int prizganih;

void setup() {
    int i = 3;
    while(i < 10) {
        pinMode(i, OUTPUT);
        i += 1;
    }
    pinMode(10, INPUT_PULLUP);
    prizganih = 0;
}

void loop() {
    if (digitalRead(10) == LOW) {
        digitalWrite(2 + prizganih, LOW);
        prizganih += 1;
        if (prizganih == 8) {
            prizganih = 1;
        }
        digitalWrite(2 + prizganih, HIGH);
        delay(200);
    }
}
```

Naloga 6

```
int prizganih;

void setup() {
    int i = 3;
    while(i < 10) {
        pinMode(i, OUTPUT);
        i += 1;
    }
    pinMode(10, INPUT_PULLUP);
    pinMode(11, INPUT_PULLUP);
    prizganih = 0;
}

void loop() {
    if (digitalRead(10) == LOW) {
        digitalWrite(2 + prizganih, LOW);
        prizganih += 1;
        if (prizganih == 8) {
            prizganih = 1;
        }
        digitalWrite(2 + prizganih, HIGH);
        delay(200);
    }
    if (digitalRead(11) == LOW) {
        digitalWrite(2 + prizganih, LOW);
        prizganih -= 1;
        if (prizganih == 0) {
            prizganih = 7;
        }
        digitalWrite(2 + prizganih, HIGH);
        delay(200);
    }
}
```

Naloga 7

```
int prizganih;
long naprej_ob;

void setup() {
    int i = 3;
    while(i < 10) {
        pinMode(i, OUTPUT);
        i += 1;
    }
    prizganih = 0;
    naprej_ob = millis(); // naprej gremo takoj
}

void loop() {
    if (millis() >= naprej_ob) {
        digitalWrite(2 + prizganih, LOW);
        prizganih += 1;
        if (prizganih == 8) {
            prizganih = 1;
        }
        digitalWrite(2 + prizganih, HIGH);
        naprej_ob += 1000; // naprej gremo eno sekundo kasneje kot prej
    }
}
```

Naloga 8

```
int prizganih;
int smer = 1;
long naprej_ob;

void setup() {
    int i = 3;
    while(i < 10) {
        pinMode(i, OUTPUT);
        i += 1;
    }
    pinMode(10, INPUT_PULLUP);
    prizganih = 0;
    naprej_ob = millis(); // naprej gremo takoj
}

void loop() {
    if (digitalRead(10) == LOW) {
        smer = -smer;
        // tole ni najbolj posrečeno, ker lahko za 0.2 s zaustavi luči
        delay(200);
    }

    if (millis() >= naprej_ob) {
        digitalWrite(2 + prizganih, LOW);
        prizganih += smer;
        if (prizganih == 8) {
            prizganih = 1;
        }
        if (prizganih == 0) {
            prizganih = 7;
        }
        digitalWrite(2 + prizganih, HIGH);
        naprej_ob += 1000; // naprej gremo eno sekundo kasneje kot prej
    }
}
```

Naloga 9

```
int prizganih;
int smer = 1;
long naprej_ob;

void setup() {
    int i = 3;
    while(i < 10) {
        pinMode(i, OUTPUT);
        i += 1;
    }
    pinMode(10, INPUT_PULLUP);
    pinMode(11, INPUT_PULLUP);
    prizganih = 0;
    naprej_ob = millis(); // naprej gremo takoj
}

void loop() {
    if (digitalRead(10) == LOW) {
        smer = -1;
        delay(200);
    }
    if (digitalRead(11) == LOW) {
        smer = 1;
        delay(200);
    }

    if (millis() >= naprej_ob) {
        digitalWrite(2 + prizganih, LOW);
        prizganih += smer;
        if (prizganih == 8) {
            prizganih = 1;
        }
        if (prizganih == 0) {
            prizganih = 7;
        }
        digitalWrite(2 + prizganih, HIGH);
        naprej_ob += 1000; // naprej gremo eno sekundo kasneje kot prej
    }
}
```