Funkcije 3: implementacija, vanjske jedinice

Zadano je *5* zadataka koje je poželjno rješiti primjenom *funkcija*, koje su detaljnije obrađene na prošloj radionici. Svaki zadatak sastoji se od nekoliko, nekad manje očitih, problema koje pokušajte rješiti traženjem dokumentacije na internetu.Rješenja zadataka pohranjujte u poseban direktorij koji ćete napraviti u svrhu ove vježbe, a svaki zadatak neka bude u imenovan u obliku *zadatak_xy.ino*. Slobodno koristite funkciju sa zadnje radionice, koja služi za učitavanje podataka sa serije koje je unio korisnik. Izvorni kod funkcije dostupan je na linku: pastebin.com/rph1FEk4.

Zadatak 1: Implementirati funkciju zadanog prototipa:

```
void toggleLED(int pins[], bool values[], int nLEDs);
```

Koja pali/gasi LED diode na pinovima zadanima nizom *pins*, a vrijednost ovisi o nizu *values*. Argument *nLEDs* označava broj LED-ica s kojima radimo. Zašto nam je potreban ovaj broj? Izlaze za LED-ice potrebno je i inicijalizirati. Implementiraj funkciju protitipa sličnom *toggleLED* koja će inicijalizirati pinove zadane nizom kao argumentom.
Po implementaciji funkcije, pomoću nje implementiraj funkciju koja se ponaša kao semafor, a koristi funkciju *toggleLED*. Mogući prototip zadan je u nastavku:

```
void semaphore();
```

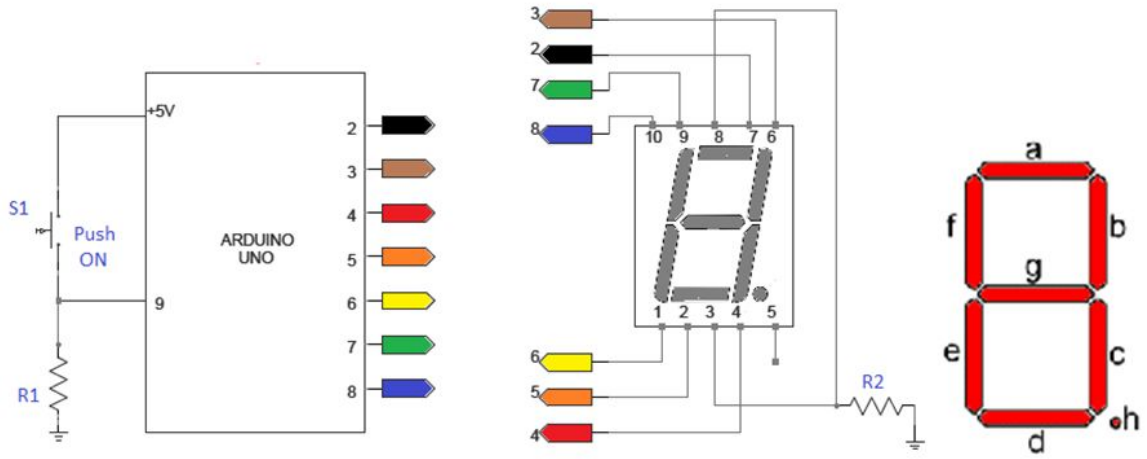
Zadatak 2: Prouči kako funkcionira 7-segmentni display. Pokušaj upaliti njegove segmente samo spajanjem na izlaze GND i 5V. Implementiraj funkciju zadanog prototipa

```
void setNumber7Segment(int number);
```

koja prikazuje broj zadan argumentom
Također, u nastavku je zadan i niz binarnih brojeva koji označavaju koji segmenti se pale za koju brojku. Npr. ukoliko želimo prikazati broj 3, u nizu *segmenti[3]* nalazi se binaran zapis `B01001111` koji označava koji segmenti moraju biti upaljeni, počevši od najnižeg bita (skroz desnog): segment A mora biti upaljen, segment B mora biti upaljen, segment C mora biti upaljen, segment D mora biti upaljen te segment G mora biti upaljen.

```
byte segmenti[] = { B00111111,  B00000110,  B01011011,  B01001111,
B01100110,B01101101,  B01111101,  B00000111,  B01111111,  B01101111 };
```

Display pospajaj kao na skici (zadad nije potrebno implementirati i funkciju gumba):



Primjeti da za implementaciju ovog zadatka trebaš koristiti *bitwise* operacije iz prošle vježbe. Također, moguće je koristiti funkciju *toggleLED* prethodnog zadatka. Po završetku implementacije, u glavnom programu implementiraj brojač sekundi koji se reserita na nulu nakon 9. Sekunde. Za ovo koristi operaciju *modulo* (%).

Zadatak 3: Prouči servo library na internetu. Spoji Servo motor na arduino i probaj ga pokrenuti. Implementiraj funkciju rampe tako da funkcinira kao rampa za vlak koji prolazi okomito na cestu na kojoj je semafor, te obje navedene funkcije koristi za finalan program.

Zadatak 4: Prouči kako čitati podatke s potenciometra koristeći analogne pinove na Arduinu te funkciju *analogRead()*. Prikaži podatke primljene sa potenciometra pomoću serije. Implementiraj program koji će, sukladno ulazu s potenciometra, pomaknuti Servo motor za proizvoljan kut. Ovo što implementiramo naziva se servo-tester i često je korišten u modelarstvu i robotici. Prouči značenje pullup i pulldown resistora.

Zadatak 5 (za brze): Pomoću dva gumba i LED matrice implementiraj jednostavnu “zmijicu”. Zmijica se smije “zabijati” u sebe, a prolazak kroz zidove ne treba biti definiran. Akciju pritiskom na dvaju gumba potrebno je implementirati prekidnim rutinama, prouči ih (engl. *interrupt*). Ukoliko si upoznat s objektom paradigmom, implementiraj zmijicu kao objekt. Interna struktura koju možeš koristiti za pohranu stanja zmijice su dva niza fiksne duljine, svaki niz koristeći kao jednu dimenziju zmijice, a svaki element kao jedan “segment” zmijice.