Matrica (polje) s kodovima znamenki + jedna funkcija showDigit() koja u petlji upali odgovarajuće segmente. Puno kraće i čišće…

Program: 7-segmentni displej – prikaz znamenki pomoću matrice (polja)

Pretpostavka: **common cathode** (HIGH pali segment, LOW gasi).
Izvodi segmenata su: A–G = D2–D8, DP = D9.

```
/*
 raspored segmenata
   A
  ---
F |   | B
 | G |
  ---
E |   | C
 |   |
  ---
   D
*/

int segmentPins[8] = {2, 3, 4, 5, 6, 7, 8, 9};
// redom: A, B, C, D, E, F, G, DP

// Tablica (matrica) kodova za znamenke i znakove
// Bitovi su u obliku gfedcba, a DP je bit7 (najviši bit)
// (LSB = segment A)
byte digits[12] = {
 0b00111111, // 0  -> A B C D E F
 0b00000110, // 1  -> B C
 0b01011011, // 2  -> A B D E G
 0b01001111, // 3  -> A B C D G
 0b01100110, // 4  -> B C F G
 0b01101101, // 5  -> A C D F G
 0b01111101, // 6  -> A C D E F G
 0b00000111, // 7  -> A B C
 0b01111111, // 8  -> A B C D E F G
 0b01101111, // 9  -> A B C D F G
 0b00000000, // 10 -> prazno
 0b10000000  // 11 -> samo DP (točka)
};

void setup() {
 for (int i = 0; i < 8; i++) {
   pinMode(segmentPins[i], OUTPUT);
   digitalWrite(segmentPins[i], LOW);
 }
}

void loop() {
 // Prikaz 1–9
```

```
  for (int n = 1; n <= 9; n++) {
    showDigit(n);
    delay(1000);
  }

  // Prikaz 0
  showDigit(0);
  delay(1000);

  // Prikaz točke
  showDigit(11);
  delay(1000);

  // Prazno na kraju
  showDigit(10);
  delay(500);
}

// Prikaz znamenke ili znaka iz tablice "digits[]"
void showDigit(int n) {
  for (int i = 0; i < 8; i++) {
    bool on = digits[n] & (1 << i);      // provjera bita za segment
    digitalWrite(segmentPins[i], on ? HIGH : LOW);
  }
}
```

---

1) Zašto koristimo matricu (polje) digits[]?

U prethodnim rješenjima imali smo posebnu funkciju za svaku znamenku (jedan(), dva(), …). To je jasno, ali kod postaje dug.

U ovoj verziji koristimo **matricu (polje)** u kojoj su spremljeni "uzorci" segmenata za svaku znamenku.
Tada je za prikaz dovoljno pozvati samo jednu funkciju:

- showDigit(5) → prikaže 5
- showDigit(0) → prikaže 0
- showDigit(11) → prikaže točku

---

2) Polje segmentPins[]

int segmentPins[8] = {2, 3, 4, 5, 6, 7, 8, 9};

Ovdje određujemo na koje izvode su spojeni segmenti:

indeks u polju	segment	izvod
0	A	D2
1	B	D3
2	C	D4

indeks u polju	segment	izvod
3	D	D5
4	E	D6
5	F	D7
6	G	D8
7	DP	D9

---

3) Polje digits[] – "kodovi" znamenki

```
byte digits[12] = {
  0b00111111, // 0
  ...
  0b01101111, // 9
  0b00000000, // prazno
  0b10000000  // točka
};
```

Svaki element polja je **8-bitni broj** (byte).
Svaki bit tog broja predstavlja jedan segment:

- bit 0 → segment A
- bit 1 → segment B
- bit 2 → segment C
- bit 3 → segment D
- bit 4 → segment E
- bit 5 → segment F
- bit 6 → segment G
- bit 7 → DP (decimalna točka)

Ako je bit = **1**, segment se pali.
Ako je bit = **0**, segment se gasi.

Primjer za znamenku 1:

0b00000110

To znači da su upaljeni samo segmenti **B i C** (bit1 i bit2).

---

4) setup() – priprema izvoda

```
for (int i = 0; i < 8; i++) {
  pinMode(segmentPins[i], OUTPUT);
  digitalWrite(segmentPins[i], LOW);
}
```

Petlja prolazi kroz svih 8 segmenata i postavlja ih kao izlaze.
Zatim ih ugasi (LOW).

---

5) loop() – prikaz znamenki pomoću petlje

```
for (int n = 1; n <= 9; n++) {
  showDigit(n);
  delay(1000);
}
```

Ova petlja automatski prikazuje znamenke 1–9, svaka po 1 sekundu.

Zatim se prikazuje 0 i točka:

- showDigit(0) → znamenka 0
- showDigit(11) → decimalna točka
- showDigit(10) → prazno

---

6) Funkcija showDigit(n)

```
bool on = digits[n] & (1 << i);
digitalWrite(segmentPins[i], on ? HIGH : LOW);
```

Ovdje se događa “trik”:

- uzmemo kod iz polja digits[n]
- provjerimo je li bit za segment i upaljen
- upalimo ili ugasimo odgovarajući izvod

Na taj način istom funkcijom možemo prikazati bilo koju znamenku ili znak.
