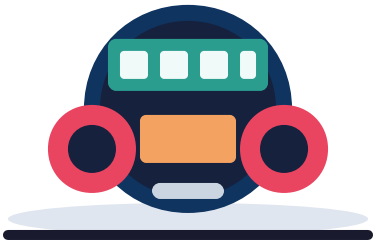


# VONALKÖVETŐ ROBOT

Robotika tábor – tervezési és oktatási segédlet

5 nap · 4 nap építés és programozás + 1 versenynap
5 csapat · 3 fő/csapat · 11–18 éves korosztály

ESP32 · TB6612FNG · TSM-5 · N20 enkóderes motorok

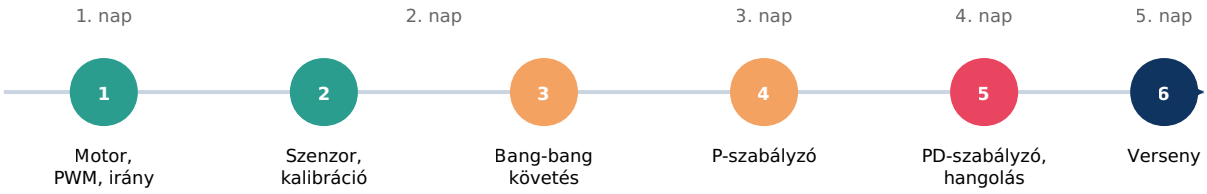
# 1. A tábor áttekintése

## 1.1 Mit építünk?

Öt nap alatt minden csapat egy **vonalkövető robotot** épít és programoz: két hajtott, enkóderes kerék, elöl szabadonfutó támpont, öt infravörös szenzor, ESP32 vezérlő. A robot fekete szigetelőszalag-vonalat követ világos alapon. Az utolsó napon időmérő verseny dönt: egy teljes kör a pályán, érintés nélkül, a legjobb idő nyer.

A tábor valódi tétje azonban nem a robot, hanem a **szabályozástechnika**: a csapatok a hét során a legegyszerűbb kétállapotú vezérléstől eljutnak a PD-szabályzóig, és megtapasztalják, mit jelent egy rendszert hangolni. Ez az a tudás, ami az egyetemi szabályozástechnikáig, sőt az ipari robotikáig elkísér.

## 1.2 A tanulási ív




*1. ábra – A hét tanulási íve: minden lépcső a előzőre épül*

## 1.3 Csapatok és szerepek

15 gyerek, **5 csapat, csapatonként 3 fő**. A korosztály vegyes (11–18), ezért a csapatbeosztásnál érdemes minden csapatba egy idősebb és egy fiatalabb gyereket is tenni – a magyarázás mindkettőjüknek tanulás. A három szerep naponta **forog**:

| Szerep | Feladata |
|---|---|
| **Programozó** | A billentyűzetnél ül, ő gépel. A többiek mondhatják, mit, de csak ő ír. |
| **Szerelő** | A robotot kezeli: kábelek, elemcsere, pályára helyezés, mechanikai állítás. |
| **Mérnök** | Jegyzetel: milyen Kp/Kd értéket próbáltak, mi történt. A hangolási napló nélkül a csapat körben jár! |

> **Mentori tipp**
>
> A hangolási napló nem adminisztráció, hanem a tábor egyik legfontosabb tanulsága: mérnöki munkát csak dokumentálva lehet végezni. Egy egyszerű táblázat elég: idő, Kp, Kd, BASE_PWM, mit csinált a robot. A versenynap reggelén az a csapat lesz nyugodt, amelyiknek van naplója.

## 1.4 Amire szükség lesz (csapatonként)

- ☐ 1× nyomtatott váz (PETG) motorbölcsőkkel
- ☐ 2× N20E-06-500 enkóderes motor
- ☐ 2× 42 mm kerék (3PI miniQ)
- ☐ 1× TB6612FNG-M motorvezérlő modul
- ☐ 1× TSM-5 ötcsatornás vonalszenzor
- ☐ 1× ESP32 DevKit + 30 tűs kifejtő panel
- ☐ 2× 4×AA elemtartó + 8 db AA elem
- ☐ 1× 1000 µF elkó + 1× 100 nF kerámia
- ☐ dupont kábelek (~20 db)
- ☐ 1× laptop Arduino IDE-vel

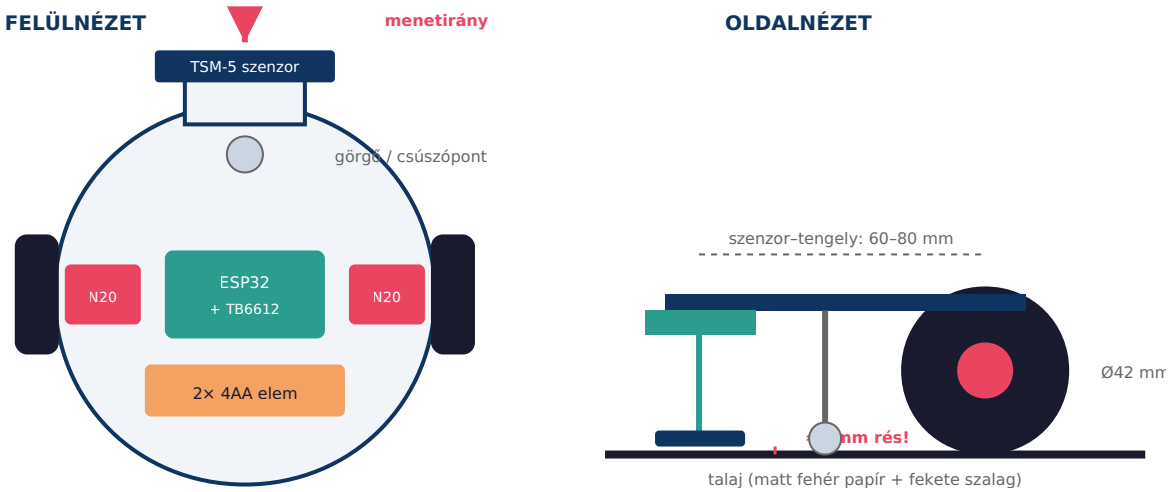
- ☐ micro-USB kábel (adatkábel, nem töltő!)
- ☐ kis csavarhúzó, oldalcsípő fogó

**Közös eszközök (a teremben)**

Multiméter (min. 2 db) · tartalék elemek (min. 16 db) · tartalék dupont kábelek · kétoldalas ragasztó, kábelkötegelő · a tartalék robot alkatrészei összeszerelve · pálya: matt fehér papír + 25 mm fekete szigetelőszalag · stopper vagy telefonos időmérés a versenyhez.

# 2. Hardver és bekötés

## 2.1 A robot felépítése




*2. ábra – Elrendezés: hajtott kerekek hátul, görgő és szenzorpanel elöl, a szenzor 2 mm-re a talajtól*

| Méret | Érték | Miért fontos |
|---|---|---|
| Váz átmérő | 150 mm | kerekekkel együtt ~165 mm nyomtáv |
| Kerékátmérő | 42 mm | tengelymagasság = 21 mm |
| Szenzor–talaj rés | **2 mm** | a TCRT5000 közelre lát a legjobban; állítható rögzítés kell! |
| Szenzor–hajtott tengely | 60–80 mm | messzebb = előbb látja a kanyart, de jobban túllendül |
| Robot tömege | ~300–350 g | elemekkel együtt; a súlypont a hajtott tengely közelében legyen |

## 2.2 Lábkiosztás (pinout)

Minden csapat **ugyanezt** a kiosztást használja – így a kódok csereszabatosak, és a mentor egy pillanat alatt átlát bármelyik robotot.

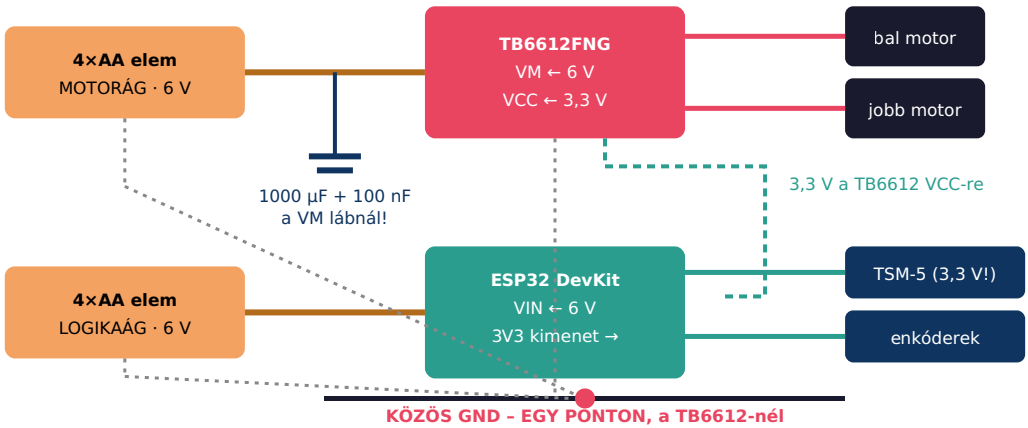
| ESP32 láb | Mire megy | Megjegyzés |
|---|---|---|
| GPIO 36, 39, 34, 35, 32 | TSM-5 OUT1…OUT5 | mind ADC1-es analóg bemenet; a 34–39 csak bemenet lehet, pont jó ide |
| GPIO 16 | TB6612 PWMA | bal motor sebesség |
| GPIO 17 / 5 | TB6612 AIN1 / AIN2 | bal motor irány |
| GPIO 4 | TB6612 PWMB | jobb motor sebesség |
| GPIO 18 / 19 | TB6612 BIN1 / BIN2 | jobb motor irány |
| GPIO 23 | TB6612 STBY | HIGH = motorok engedélyezve |
| GPIO 25 / 26 | bal enkóder A / B | megszakításos számlálás |
| GPIO 27 / 14 | jobb enkóder A / B | |
| 3V3 | | |

| | | |
|---|---|---|
| | TB6612 VCC, TSM-5 5V láb, enkóderek tápja | a TSM-5-öt 3,3 V-ról hajtjuk – így a kimenete sem megy 3,3 V fölé! |
| VIN | logikai elemcsomag + (6 V) | |
| GND | közös föld | **egy ponton** találkozik a motoros földdel |

**A leggyakoribb bekötési hibák**

- **TSM-5 5 V-ra kötve:** a kimenet 5 V-ig mehet, az ESP32 ADC-je viszont csak 3,3 V-ot bír. Mindig 3,3 V-ról!
- **STBY láb bekötetlen:** a TB6612 néma marad. Ha „semmi nem mozog, pedig minden jó" – ez az első gyanúsított.
- **VM és VCC felcserélve a TB6612-n:** VM = 6 V motorfeszültség, VCC = 3,3 V logika. Fordítva a modul halott.
- **Enkóder tápja a motorágról:** az enkóder panel 3,3 V-os logika, a motorfeszültségtől tönkremehet.
- **USB-kábel = töltőkábel:** ha az IDE nem látja a portot, először a kábelt cseréld adatkábelre.

## 2.3 Tápellátás – két külön kör




*3. ábra – Kettéválasztott tápellátás: a motorzaj nem éri el a vezérlőt*

A két elemcsomag **szétválasztása** a robot megbízhatóságának kulcsa. A motorok indításkor nagy áramlökést vesznek fel, amitől a tápfeszültség beesik – ha a vezérlő ugyanarról a csomagról menne, újraindulna, és a hibát a kódban keresnénk, hiába. A közös földpont nélkül viszont a jelek nem értelmezhetők, ezért a két kör GND-je **pontosan egy helyen**, a TB6612 mellett találkozik.

**Miért van kondenzátor a motorágon?**

Az 1000 µF-os elektrolit a motorindítás milliszekundumos áramcsúcsát fedezi, a 100 nF-os kerámia pedig a H-híd gyors kapcsolási tüskéit nyeli el – a kettő **együtt** működik, mert más-más frekvencián hatásos. A kerámiát a lehető legrövidebb lábbal, közvetlenül a TB6612 VM–GND lábai közé!

## 2.4 Összeszerelési sorrend (1. nap)

1. **Motorok a bölcsőkbe** – a kábel hátrafelé nézzen. Ha lötyög, egy csík kétoldalas ragasztó.
2. **Kerekek a tengelyre** – a D-tengely lapos oldala illeszkedjen, ne erőltesd ferdén.
3. **Görgő magasságállítás** – a robot vízszintes legyen: a váz elöl és hátul azonos magasságban.

4. **TSM-5 a nyúlványra** – hosszlyukas rögzítés, **2 mm-re a talajtól**. Ellenőrzés: csúsztass alá két egymásra tett bankkártyát – épp férjen el.
5. **Elektronika felcsavarozása** – ESP32 kifejtő és TB6612 a rácsos furatokra.
6. **Elemtartók** – hátul, a hajtott tengely fölé/mögé (a súlypont miatt).
7. **Kábelezés** – a 2.2-es táblázat szerint, színkódolva: piros = táp, fekete = GND, egyéb = jel.
8. **Ellenőrzés bekapcsolás ELŐTT** – lásd az alábbi checklistet!

**Bekapcsolás előtti checklist (minden csapat, mentorral együtt!)**

- ☐ VM-re megy a motorelem +, VCC-re a 3,3 V?
- ☐ TSM-5 tápja 3,3 V (NEM VIN/5V)?
- ☐ GND-k összekötve egy ponton?
- ☐ Nincs rövidzár: multiméterrel a + és GND között ne legyen 0 Ω!
- ☐ Kerekek szabadon forognak, semmi nem ér a talajhoz a görgőn és kerekeken kívül?

# 3. Szoftver – az Arduino IDE és a kódváz

## 3.1 Az IDE beállítása (a tábor előtt minden gépre!)

1. Arduino IDE letöltése: `arduino.cc/en/software`
2. ESP32 támogatás: File → Preferences → Additional Board Manager URLs mezőbe: `https://espressif.github.io/arduino-esp32/package_esp32_index.json`
3. Tools → Board → Boards Manager → keresés: „esp32" → Install (Espressif Systems)
4. Beállítás: Board = **ESP32 Dev Module**, Upload Speed = 921600
5. Teszt: a robot USB-re dugva Tools → Port alatt megjelenik-e a port? Ha nem: adatkábel? illesztőprogram (CP2102)?

**Ezt a mentor csinálja meg ELŐRE**

Az IDE + ESP32 csomag telepítése gépenként 10–20 perc és internetfüggő. Ha ez a tábor első órájára marad, az első fél nap elmegy vele. Minden laptopra előre telepítve, egy próbafeltöltéssel ellenőrizve!

## 3.2 A kódváz filozófiája

Minden csapat **ugyanazt a kész kódot** kapja és másolja be az IDE-be. A kód teljes és működik – de a szabályzó rész két változatban van benne: a bang-bang kikommentezve, a P/PD élesben. A hét során a csapatok nem új kódot írnak, hanem **ezt a kódot értik meg, kapcsolgatják és hangolják**. A hangolható értékek egy helyen, a kód tetején vannak:

| Változó | Kezdőérték | Mit csinál |
|---|---|---|
| `MAX_PWM` | 120 | Sebességplafon. A 3. napig NEM áruljuk el, hogy feljebb vehető! |
| `BASE_PWM` | 100 | Alapsebesség egyenes szakaszon |
| `Kp` | 0.08 | Arányos tag: mekkora a korrekció a hibával arányosan |
| `Kd` | 0.0 | Differenciáló tag: a hiba változási sebességére reagál (3. napon!) |

## 3.3 A négy kód – lépcsőzetes bemásolás

A csapatok **nem egyszerre kapják meg a teljes programot**, hanem négy lépcsőben, a napi tananyaghoz igazítva. Mindegyik kód önmagában teljes és működik: File → New Sketch, a régi tartalom törlése, az új kód bemásolása egy az egyben, feltöltés. Így minden lépcsőnél csak **egy** új dolgot kell megérteni – és ha valami elromlik, tudjuk, melyik rétegben keressük.

| Kód | Mikor | Mi az új benne |
|---|---|---|
| **1. Motorteszt** | 1. nap délelőtt | PWM, irány, a motorL/motorR függvények |
| **2. Enkóder** | 1. nap délután | megszakításos számlálás, mérés |
| **3. IR szenzor** | 2. nap délelőtt | kalibráció, súlyozott átlag, hibajel – motor nélkül! |
| **4. Teljes robot** | 2. nap délutántól | minden együtt + bang-bang és PD szabályzó |

**Bemásolási szabályok (mondjuk el az első alkalommal!)**

- Mindig a **teljes** kódot másold, az első sortól az utolsóig – fél kód nem fordul le.
- Feltöltés előtt mentés (Ctrl+S), névvel: `1_motorteszt`, `2_enkoder`…
- Ha a fordító hibát ír, a hibaüzenet **első** sorát olvasd – az mondja meg, hányadik sorban a gond (jellemzően: lemaradt az eleje vagy a vége a másolásnak).
- Feltöltés közben a robot kerekei megmozdulhatnak – **bakra téve** (kerekek a levegőben) töltsünk!

### 1. kód – Motorteszt (1_motorteszt.ino)

A robot bakon áll (kerekek a levegőben)! A program körbejárja az alapmozgásokat: előre, stop, hátra, fordulás. Figyeld meg: a `motorL(100)` szám a sebesség, az előjel az irány. Feladat feltöltés után: írd át a loop()-ot úgy, hogy a robot négyzet alakban járjon körbe a padlón (előre 2 mp → 90° fordulás → ismétlés).

```
/* ========================================
 *  1. KÓD - MOTORTESZT (1. nap délelőtt)
 *  Csak a motorok: előre, hátra, fordulás.
 * ======================================== */

// TB6612 motorvezérlő lábak
const int PWMA = 16;   // bal motor sebesség
const int AIN1 = 17;   // bal motor irány
const int AIN2 = 5;
const int PWMB = 4;    // jobb motor sebesség
const int BIN1 = 18;   // jobb motor irány
const int BIN2 = 19;
const int STBY = 23;   // engedélyezés (HIGH = megy)

int MAX_PWM = 120;     // sebességplafon

void motorL(int speed) {
  speed = constrain(speed, -MAX_PWM, MAX_PWM);
  digitalWrite(AIN1, speed >= 0);
  digitalWrite(AIN2, speed < 0);
  analogWrite(PWMA, abs(speed));
}

void motorR(int speed) {
  speed = constrain(speed, -MAX_PWM, MAX_PWM);
  digitalWrite(BIN1, speed >= 0);
  digitalWrite(BIN2, speed < 0);
  analogWrite(PWMB, abs(speed));
}

void motorStop() { motorL(0); motorR(0); }

void setup() {
  Serial.begin(115200);
  pinMode(AIN1, OUTPUT); pinMode(AIN2, OUTPUT);
  pinMode(BIN1, OUTPUT); pinMode(BIN2, OUTPUT);
  pinMode(STBY, OUTPUT);
  digitalWrite(STBY, HIGH);
  motorStop();
  delay(3000);           // 3 mp a letevéshez!
}

void loop() {
  Serial.println("ELORE");
  motorL(100); motorR(100);
  delay(2000);

  Serial.println("STOP");
  motorStop();
  delay(1000);

  Serial.println("HATRA");
  motorL(-100); motorR(-100);
  delay(2000);

  Serial.println("FORDULAS HELYBEN");
  motorL(100); motorR(-100);
  delay(1000);

  motorStop();
  delay(2000);
}
```

## 2. kód – Enkóder (2_enkoder.ino)

Ugyanazok a motorlábak + az enkóderek. A számlálók megszakításból pörögnek, a loop() csak kiírja őket. Első feladat: **kézzel** forgasd a kereket egy teljes fordulatot, és olvasd le, mennyit lépett a számláló – ez a robotod impulzus/fordulat értéke, írd a naplóba! Második feladat: engedd el a motorokat (kommentek törlése), és nézd meg, a két oldal pontosan ugyanannyit lép-e. (Nem fog – és pont ez a tanulság: ezért nem sikerült délelőtt a pontos négyzet.)

```
/* ========================================
 *  2. KÓD - ENKÓDER (1. nap délután)
 *  A kerékfordulat számlálása megszakítással.
 *  Először KÉZZEL forgasd a kereket és figyeld
 *  a Serial Monitort (115200 baud)!
 * ======================================== */

// Enkóder lábak
const int ENC_L_A = 25;  const int ENC_L_B = 26;
const int ENC_R_A = 27;  const int ENC_R_B = 14;

// Motor lábak (a méréshez hajtani is fogunk)
const int PWMA = 16, AIN1 = 17, AIN2 = 5;
const int PWMB = 4,  BIN1 = 18, BIN2 = 19, STBY = 23;
int MAX_PWM = 120;

volatile long encL = 0, encR = 0;

void IRAM_ATTR encL_isr() {
  if (digitalRead(ENC_L_A) == digitalRead(ENC_L_B)) encL++; else encL--;
}
void IRAM_ATTR encR_isr() {
  if (digitalRead(ENC_R_A) == digitalRead(ENC_R_B)) encR++; else encR--;
}
void motorL(int speed) {
  speed = constrain(speed, -MAX_PWM, MAX_PWM);
  digitalWrite(AIN1, speed >= 0); digitalWrite(AIN2, speed < 0);
  analogWrite(PWMA, abs(speed));
}
void motorR(int speed) {
  speed = constrain(speed, -MAX_PWM, MAX_PWM);
  digitalWrite(BIN1, speed >= 0); digitalWrite(BIN2, speed < 0);
  analogWrite(PWMB, abs(speed));
}
void setup() {
  Serial.begin(115200);
  pinMode(AIN1, OUTPUT); pinMode(AIN2, OUTPUT);
  pinMode(BIN1, OUTPUT); pinMode(BIN2, OUTPUT);
  pinMode(STBY, OUTPUT); digitalWrite(STBY, HIGH);
  motorL(0); motorR(0);

  pinMode(ENC_L_A, INPUT_PULLUP); pinMode(ENC_L_B, INPUT_PULLUP);
  pinMode(ENC_R_A, INPUT_PULLUP); pinMode(ENC_R_B, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(ENC_L_A), encL_isr, CHANGE);
  attachInterrupt(digitalPinToInterrupt(ENC_R_A), encR_isr, CHANGE);
}
void loop() {
  // FELADAT 1: motorok állnak - forgasd kézzel a kereket!
  //   Hány impulzus egy teljes fordulat? Írd fel!
  // FELADAT 2: töröld a // jeleket, és nézd meg,
  //   a két motor PONT ugyanannyit lép-e 2 mp alatt.
  // motorL(100); motorR(100);

  Serial.printf("bal: %ld   jobb: %ld\n", encL, encR);
  delay(200);
}
```

## 3. kód – IR szenzor (3_szenzor.ino)

Motor nincs! A robot kézben van, a program csak lát: kalibráció indításkor (5 mp húzogatás a vonal fölött), utána az öt normalizált érték + a kiszámolt pozíció és hibajel. Nézd Serial Monitoron, majd válts **Serial Plotterre**: told a robotot lassan keresztbe a vonalon, és figyeld, hogyan fut végig a pozíció 0-tól 4000-ig. Ha ezt látod, megértetted a súlyozott átlagot.

```
/* ========================================
 *  3. KÓD - IR SZENZOR (2. nap délelőtt)
 *  Nyers értékek, kalibráció, vonalpozíció.
 *  A motorok itt még NEM mozognak!
 *  Nézd Serial Monitoron ÉS Serial Plotteren is.
 * ======================================== */

const int SENSOR_PIN[5] = {36, 39, 34, 35, 32};

int sensorMin[5], sensorMax[5];
int lastError = 0;
// Normalizált érték: 0 = fehér, 1000 = fekete
int readNorm(int i) {
  int v = analogRead(SENSOR_PIN[i]);
  v = map(v, sensorMin[i], sensorMax[i], 0, 1000);
  return constrain(v, 0, 1000);
}
// Vonalpozíció: 0 ... 2000 (közép) ... 4000
int readPosition() {
  long sum = 0, weighted = 0;
  bool onLine = false;
  for (int i = 0; i < 5; i++) {
    int v = readNorm(i);
    if (v > 200) onLine = true;
    sum += v;
    weighted += (long)v * i * 1000;
  }
  if (!onLine) return (lastError < 0) ? 0 : 4000;
  return weighted / sum;
}
void calibrate() {
  Serial.println("KALIBRACIO - huzogasd a robotot a vonal folott!");
  for (int i = 0; i < 5; i++) { sensorMin[i] = 4095; sensorMax[i] = 0; }
  unsigned long start = millis();
  while (millis() - start < 5000) {
    for (int i = 0; i < 5; i++) {
      int v = analogRead(SENSOR_PIN[i]);
      if (v < sensorMin[i]) sensorMin[i] = v;
      if (v > sensorMax[i]) sensorMax[i] = v;
    }
    delay(5);
  }
  Serial.println("KESZ! Ertekek:");
  for (int i = 0; i < 5; i++) {
    Serial.printf("  S%d: min=%d max=%d", i + 1, sensorMin[i], sensorMax[i]);
    if (sensorMax[i] - sensorMin[i] < 500) Serial.print("  <-- KEVES KONTRASZT!");
    Serial.println();
  }
}
void setup() {
  Serial.begin(115200);
  delay(1000);
  calibrate();
}
void loop() {
  // Nyers és normalizált értékek + pozíció
  int position = readPosition();
  int error = position - 2000;
  lastError = error;

  for (int i = 0; i < 5; i++) {
    Serial.printf("S%d:%d ", i + 1, readNorm(i));
  }
  Serial.printf("  pos:%d  err:%d\n", position, error);
  delay(200);
}
```

## 4. kód – A teljes robot (4_teljes.ino)

Minden eddigi réteg együtt, plusz a szabályzó. A loop()-ban két változat van: a **bang-bang** kikommentezve és a **P/PD** élesben. A 2. nap délutánján cseréld meg (bang-bang élesítése, P kikommentezése), a 3. napon vissza. A hangolható értékek a kód tetején, egy helyen.

```
/*
 * ============================================================
 *  VONALKÖVETŐ ROBOT - TÁBORI KÓDVÁZ
 *  Enterpartner Robotika Tábor
 *
 *  Hardver:
 *   - ESP32 DevKit (30 tűs) + kifejtő panel
 *   - TB6612FNG dual motorvezérlő
 *   - TSM-5 ötcsatornás vonalszenzor (analóg!)
 *   - 2x N20E-06-500 enkóderes motor
 *   - 2x 4xAA elemtartó (motor + logika, közös GND!)
 *
 *  Arduino IDE beállítás:
 *   Alaplap: "ESP32 Dev Module"
 *   Feltöltési sebesség: 921600
 * ============================================================
 */

// ============ 1. LÁBKIOSZTÁS - NE ÍRD ÁT! ============

// TSM-5 szenzor (balról jobbra, menetirányban)
const int SENSOR_PIN[5] = {36, 39, 34, 35, 32};

// TB6612 motorvezérlő
const int PWMA = 16;   // bal motor sebesség
const int AIN1 = 17;   // bal motor irány 1
const int AIN2 = 5;    // bal motor irány 2
const int PWMB = 4;    // jobb motor sebesség
const int BIN1 = 18;   // jobb motor irány 1
const int BIN2 = 19;   // jobb motor irány 2
const int STBY = 23;   // engedélyező láb (HIGH = megy)

// Enkóderek
const int ENC_L_A = 25;
const int ENC_L_B = 26;
const int ENC_R_A = 27;
const int ENC_R_B = 14;

// ============ 2. HANGOLHATÓ ÉRTÉKEK - EZEKET ÁLLÍTSD! ============

int MAX_PWM   = 120;    // sebességplafon (0..255) - kezdetben hagyd 120-on!
int BASE_PWM  = 100;    // alapsebesség egyenesben

float Kp = 0.08;        // P tag erősítés
float Kd = 0.0;         // D tag erősítés (3. napon jön!)

// ============ 3. BELSŐ VÁLTOZÓK ============

int sensorMin[5], sensorMax[5];       // kalibrációs értékek
volatile long encL = 0, encR = 0;     // enkóder számlálók
int lastError = 0;

// ============ 4. ENKÓDER MEGSZAKÍTÁSOK ============

void IRAM_ATTR encL_isr() {
  if (digitalRead(ENC_L_A) == digitalRead(ENC_L_B)) encL++; else encL--;
}
void IRAM_ATTR encR_isr() {
  if (digitalRead(ENC_R_A) == digitalRead(ENC_R_B)) encR++; else encR--;
}

// ============ 5. MOTORVEZÉRLÉS ============
// sebesség: -255..+255 (negatív = hátra)

void motorL(int speed) {
  speed = constrain(speed, -MAX_PWM, MAX_PWM);
  digitalWrite(AIN1, speed >= 0);
  digitalWrite(AIN2, speed < 0);
  analogWrite(PWMA, abs(speed));
}

void motorR(int speed) {
  speed = constrain(speed, -MAX_PWM, MAX_PWM);
  digitalWrite(BIN1, speed >= 0);
  digitalWrite(BIN2, speed < 0);
  analogWrite(PWMB, abs(speed));
}
```

```
void motorStop() {
  motorL(0);
  motorR(0);
}

// ============ 6. SZENZOROLVASÁS ============

// Nyers érték egy csatornáról (0..4095)
int readRaw(int i) {
  return analogRead(SENSOR_PIN[i]);
}

// Normalizált érték (0 = fehér, 1000 = fekete)
int readNorm(int i) {
  int v = readRaw(i);
  v = map(v, sensorMin[i], sensorMax[i], 0, 1000);
  return constrain(v, 0, 1000);
}

// Vonalpozíció súlyozott átlaggal
// Eredmény: 0 (vonal balra kint) ... 2000 (középen) ... 4000 (jobbra kint)
// Ha nem lát vonalat: az utolsó ismert irány szerint 0 vagy 4000
int readPosition() {
  long sum = 0, weighted = 0;
  bool onLine = false;

  for (int i = 0; i < 5; i++) {
    int v = readNorm(i);
    if (v > 200) onLine = true;       // legalább egy szenzor lát vonalat
    sum += v;
    weighted += (long)v * i * 1000;   // súly: 0, 1000, 2000, 3000, 4000
  }

  if (!onLine) {
    // elvesztettük a vonalat: arra fordulunk, amerre utoljára láttuk
    return (lastError < 0) ? 0 : 4000;
  }
  return weighted / sum;
}

// ============ 7. KALIBRÁCIÓ ============
// Bekapcsolás után 5 másodpercig húzogasd a robotot
// a vonal fölött ide-oda, hogy minden szenzor lásson
// fehéret ÉS feketét is!

void calibrate() {
  Serial.println("KALIBRACIO - huzogasd a robotot a vonal folott!");
  for (int i = 0; i < 5; i++) {
    sensorMin[i] = 4095;
    sensorMax[i] = 0;
  }
  unsigned long start = millis();
  while (millis() - start < 5000) {
    for (int i = 0; i < 5; i++) {
      int v = readRaw(i);
      if (v < sensorMin[i]) sensorMin[i] = v;
      if (v > sensorMax[i]) sensorMax[i] = v;
    }
    delay(5);
  }
  Serial.println("KESZ! Ertekek:");
  for (int i = 0; i < 5; i++) {
    Serial.printf("  S%d: min=%d max=%d\n", i + 1, sensorMin[i], sensorMax[i]);
  }
  // Ellenőrzés: ha egy csatornán kicsi a különbség, az gyanús
  for (int i = 0; i < 5; i++) {
    if (sensorMax[i] - sensorMin[i] < 500) {
      Serial.printf("  FIGYELEM: S%d keves kontrasztot lat!\n", i + 1);
    }
  }
}

// ============ 8. SETUP ============

void setup() {
  Serial.begin(115200);

  pinMode(AIN1, OUTPUT); pinMode(AIN2, OUTPUT);
  pinMode(BIN1, OUTPUT); pinMode(BIN2, OUTPUT);
```

```
  pinMode(STBY, OUTPUT);
  digitalWrite(STBY, HIGH);

  pinMode(ENC_L_A, INPUT_PULLUP); pinMode(ENC_L_B, INPUT_PULLUP);
  pinMode(ENC_R_A, INPUT_PULLUP); pinMode(ENC_R_B, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(ENC_L_A), encL_isr, CHANGE);
  attachInterrupt(digitalPinToInterrupt(ENC_R_A), encR_isr, CHANGE);

  motorStop();
  delay(1000);
  calibrate();

  Serial.println("INDULAS 3 masodperc mulva...");
  delay(3000);
}

// ============ 9. FŐHUROK - ITT DOLGOZOL! ============

void loop() {
  int position = readPosition();       // 0..4000, közép = 2000
  int error = position - 2000;         // negatív = vonal balra, pozitív = jobbra

  // ---- 2. NAP: BANG-BANG ----
  // Töröld a // jeleket, a P-szabályzót pedig kommentezd ki!
  //
  // if (error < -300)      { motorL(60);  motorR(MAX_PWM); }   // balra fordul
  // else if (error > 300)  { motorL(MAX_PWM); motorR(60);  }   // jobbra fordul
  // else                   { motorL(BASE_PWM); motorR(BASE_PWM); } // egyenesen

  // ---- 3. NAP: P (majd PD) SZABÁLYZÓ ----
  float correction = Kp * error + Kd * (error - lastError);
  lastError = error;

  motorL(BASE_PWM + correction);
  motorR(BASE_PWM - correction);

  // ---- Kiírás hangoláshoz (Serial Plotter is érti!) ----
  static unsigned long lastPrint = 0;
  if (millis() - lastPrint > 100) {
    lastPrint = millis();
    Serial.printf("pos:%d err:%d encL:%ld encR:%ld\n",
                  position, error, encL, encR);
  }

  delay(2);   // ~400 Hz szabályzási frekvencia
}
```

## 3.4 A Serial Plotter – a hangolás műszere

A kód 100 ms-onként kiírja a pozíciót és a hibát. Az Arduino IDE **Serial Plotter** ablaka (Tools menü, 115200 baud) ezt élő grafikonon mutatja. Hangolásnál ez a legfontosabb eszköz: a hibagörbén azonnal látszik a lengés, a túllendülés, a D-tag csillapító hatása. Tanítsd meg a csapatoknak az első naptól – aki a grafikont nézi, az érti, mit csinál; aki csak a robotot, az találgat.

# 4. Napi terv

Minden nap azonos ritmusra épül: **reggeli demó** (a mentor 10 percben megmutatja a napi célt egy kész roboton) → **páros munka** → **déli checkpoint** (minden csapat megmutatja, hol tart) → **délutáni munka** → **napzáró kör** (mi ment, mi nem, mit tanultunk). A checkpointok kőbe vésettek: aki lemarad, mentori segítséget kap, nem több időt – a cél, hogy az 5. napra **minden** robot körbeérjen.

## 1. nap – „Megmozdul!" (összeszerelés és motorvezérlés)

| Idő | Program |
|---|---|
| **9:00** | Köszöntő, csapatalakítás, szerepek. A tábor íve és a versenyszabály kihirdetése – a cél az első perctől világos. |
| **9:30** | **Demó:** a mentor kész robotja végigmegy a pályán. „Péntekre a tiétek is ezt fogja tudni – sőt gyorsabban." |
| **9:45** | Összeszerelés a 2.4-es sorrend szerint. Mechanika, kábelezés, checklist. |
| **12:00** | Ebéd |
| **13:00** | **Checkpoint 1:** bekapcsolási checklist mentorral, első bekapcsolás. |
| **13:15** | **1. kód (motorteszt)** bemásolása, feltöltés bakon. Előre, hátra, forgás. Feladat: négyzet alakban körbejárás a padlón (előre 2 mp → fordulás 90° → ismétlés). |
| **15:00** | **2. kód (enkóder)**: a Serial Monitoron figyelik a számlálókat kézzel forgatott keréknél. Kérdés: „hány impulzus egy teljes kerékfordulat?" – mérjék meg és naplózzák! |
| **16:00** | **Checkpoint 2:** minden robot megy előre-hátra-fordul. Napzáró kör. |

**Miért a négyzetjárás?**

Mert garantáltan nem sikerül pontosan – a robot ferdén megy, a fordulat nem 90°. Ez a kudarc a tananyag: a gyerekek maguktól jönnek rá, hogy nyílt hurokban (visszacsatolás nélkül) nem lehet pontosan vezérelni. Holnaptól ezért kell a szenzor.

## 2. nap – „Látja a vonalat!" (szenzor, kalibráció, bang-bang)

| Idő | Program |
|---|---|
| **9:00** | **Demó:** mit lát az IR-szenzor? Multiméterrel megmutatjuk az OUT feszültségét fehéren és feketén. Miért kell kalibrálni? |
| **9:20** | **3. kód (IR szenzor)**: nyers és normalizált értékek figyelése mind az öt csatornán, fehér és fekete fölött. Feladat: táblázat a saját robot értékeivel. |
| **10:30** | Kalibráció megértése és gyakorlása. A „húzogatós" 5 másodperc rutinná válik. |
| **11:15** | A súlyozott átlag – táblás magyarázat a 4. ábra szerint, majd a `readPosition()` figyelése Serial Plotteren, kézzel tologatott robotnál. |
| **12:00** | Ebéd |
| **13:00** | **4. kód (teljes robot)** bemásolása, bang-bang élesítése (kommentek cseréje). Első vonalkövetés! Cikkcakkos, de megy. |
| **14:30** | **Checkpoint 3:** minden robot végigmegy a gyakorlópálya oválján bang-banggel. |
| **15:00** | |

| | |
|---|---|
| | Beszélgetés: MIÉRT cikkcakkozik? (Mert csak 3 állapotot ismer.) Hogyan lehetne simább? – a csapatok ötletelnek, a mentor a P-szabályzó felé terelgeti a gondolatot, de nem mondja ki. |
| **16:00** | Szabad kísérletezés + napzáró. |

## 3. nap – „Simán megy!" (P- és PD-szabályzó)

| Idő | Program |
|---|---|
| **9:00** | **Demó + tábla:** a P-szabályzó gondolata. Korrekció = Kp × hiba. Kis hiba → kis kormányzás, nagy hiba → nagy. Ez az, amit tegnap ki akartatok találni! |
| **9:30** | Bang-bang kikommentezése, P élesítése. Kp hangolása: 0.02-től duplázgatva. Mit látunk kis Kp-nál (lomha, kisodródik) és nagy Kp-nál (leng, cikkcakk visszajön)? |
| **11:00** | **Hangolási napló kötelező:** minden próbához Kp + megfigyelés. A mérnök szerep ma főszereplő. |
| **12:00** | Ebéd |
| **13:00** | **A D-tag:** tábla + Serial Plotter. A hiba VÁLTOZÁSA előre jelzi a lengést – a D-tag fékez, mielőtt túllendülnénk. Kd hangolása Kp mellé. |
| **14:30** | **A nagy leleplezés:** a MAX_PWM eddig 120-on volt! Aki stabilan megy, feljebb veheti – de a hangolás sebességfüggő: ami 120-on jó volt, 180-on leng. Újrahangolás. |
| **16:00** | **Checkpoint 4:** minden robot PD-vel, simán követi a vonalat. Napzáró. |

**Hangolási recept (a csapatoknak kiadható)**

1. Kd = 0. Kp-t emeld addig, amíg a robot épp elkezd lengeni a vonal körül.
2. Vedd vissza Kp-t a lengő érték ~60%-ára.
3. Emeld Kd-t 0.1-es lépésekben, amíg a kanyar utáni túllendülés eltűnik.
4. Ha kész: emeld a BASE_PWM-et 20-szal, és kezdd elölről. A gyorsabb robot más hangolást kér!

## 4. nap – „Gyorsabban!" (finomhangolás, kvalifikáció, haladó feladatok)

| Idő | Program |
|---|---|
| **9:00** | A versenypálya felfedése! Mostantól ezen gyakorolnak – de a robotokat a futamok közt vissza kell vinni az asztalhoz (a versenyszimuláció része). |
| **9:15** | Szabad hangolás a versenypályán. Cél: először STABILAN körbeérni, csak utána gyorsítani. |
| **11:00** | Haladó feladatok azoknak, akik stabilan mennek (választható): **(a)** egyenes-felismerés: ha a hiba sokáig kicsi, emeld a sebességet – kanyarban vissza; **(b)** köridő mérése enkóderrel: a robot maga írja ki, mennyi utat tett meg; **(c)** WiFi-s indítógomb telefonról. |
| **12:00** | Ebéd |
| **13:00** | Hangolás folytatása. A mentor ma már NEM segít kódban – csak kérdez. („Mit mond a napló? Mit látsz a Plotteren?") |
| **15:00** | **KVALIFIKÁCIÓ:** minden robotnak egyszer körbe kell érnie a versenypályán, idő nem számít. Aki nem megy körbe, holnap reggel mentori segítséggel javít – de holnap délután már csak versenyezni lehet. |
| **16:00** | Napzáró, versenyszabály ismertetése újra, kérdések. |

## 5. nap – VERSENYNAP

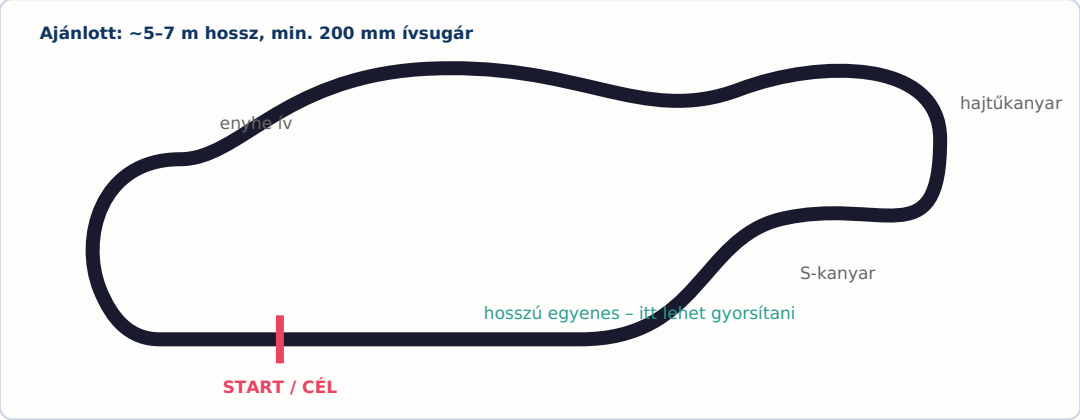
| Idő | Program |
|---|---|
| **9:00** | Reggeli javítási idő azoknak, akik nem kvalifikáltak; a többiek szabadedzést tartanak. |
| **10:30** | Pálya lezárva. Robotok „parc fermé"-be: az asztalra, hozzányúlás nélkül. Sorsolás a futamsorrendre. |
| **11:00** | **1. futam** – minden csapat egyszer fut, időmérés. |
| **12:00** | Ebéd + a futamok közt 15 perc hangolási lehetőség (elemcsere is!). |
| **13:00** | **2. futam** |
| **14:00** | **3. futam** – fordított sorrendben (az addigi leggyorsabb megy utolsónak: dráma!) |
| **15:00** | Eredményhirdetés. Díj a leggyorsabbnak – és külön elismerés a legjobb hangolási naplónak, a legszebb kódnak, a legkitartóbb hibakeresésnek. |
| **15:30** | Zárókör: ki mit visz haza? Merre tovább? (szakkör, OTIO, a cég robotlaborja…) |

# 5. Versenyszabályzat

## 5.1 A szabályok

1. **A feladat:** egy teljes kör a versenypályán, a start/cél vonaltól a start/cél vonalig.
2. **Időmérés:** a robot orra (szenzorpanel eleje) metszi a startvonalat → indul az óra; ugyanez a célban → áll az óra. Két mérő méri telefonnal, a kettő átlaga számít.
3. **Érintés tilos:** a start után a robothoz hozzáérni nem szabad. Bármilyen érintés = a futam érvénytelen.
4. **Indítás:** a robotot a csapat a startvonalra helyezi, a mentor visszaszámol (3-2-1), és a csapat egy gombnyomással / USB-kihúzással / WiFi-vel indít. Az indítás pillanatától érvényes az érintési tilalom.
5. **Vonalvesztés:** ha a robot elhagyja a vonalat, de magától visszatalál, a futam érvényes. Ha 10 másodpercig nem talál vissza, vagy elhagyja a pályaasztalt, a futam érvénytelen.
6. **3 futam,** a legjobb idő számít. Érvénytelen futam = nincs idő arra a futamra.
7. **A futamok közt** szabad hangolni, elemet cserélni, kódot módosítani. A futam alatt nem.
8. **Holtverseny** esetén egy azonnali negyedik futam dönt.

## 5.2 A versenypálya




*5. ábra – Példa versenypálya: legyen egyenes (sebesség), enyhe ív, S-kanyar és egy hajtű (hangolási próbatétel)*

- **Alap:** matt fehér papír (tapéta- vagy csomagolópapír tekercsből), a fényes fa asztal tükröz és becsapja a szenzort!
- **Vonal:** 25 mm széles matt fekete szigetelőszalag. Az íveket sok kis egyenes szakaszból rakd ki, vagy melegítsd a szalagot hajszárítóval és húzd ívesre.
- **Minimális ívsugár 200 mm** – ennél élesebb kanyart a 165 mm nyomtávú robot csak nagyon lassan vesz be.
- A **gyakorlópálya** (2–4. nap) egyszerű ovál legyen; a versenypálya a 4. napon kerül elő. Így a versenynapig marad izgalom, de van idő rá hangolni.
- A pálya **ne érjen falig** – hagyj 30 cm kifutót, ahová a megvadult robot mehet.

# 6. Hibakeresési útmutató

A táblázat a mentor és a csapatok közös nyelve. A szabály: **mérj, mielőtt kódot írsz át!** A hibák 80%-a kábel, táp vagy mechanika – nem szoftver.

| Tünet | Valószínű ok | Mit tegyél |
|---|---|---|
| Az IDE nem látja a portot | Töltőkábel adatkábel helyett; hiányzó CP2102 driver | Kábelcsere; driver telepítés; másik USB-port |
| Semmi nem mozog, pedig a kód fut | STBY láb nincs HIGH-on / bekötetlen; motorelem lemerült/nincs bekapcsolva | Multiméterrel: van 6 V a VM-en? 3,3 V az STBY-n? |
| Egyik motor jó, másik nem | Kábel kicsúszott; PWMB/BIN lábak felcserélve | Cseréld meg a két motort a driveren: ha a hiba átugrik, kábel; ha marad, motor |
| A robot hátrafelé megy „előre" helyett | Motorpolaritás fordítva | NE a kódot írd át – cseréld meg a motor két vezetékét a driveren |
| Az ESP32 újraindul, amikor a motorok indulnak | Közös táp, beeső feszültség; hiányzó kondenzátorok | Két külön elemcsomag? 1000 µF a VM-en? Friss elemek? |
| A robot délután „rosszabb", mint délelőtt, ugyanazzal a kóddal | Merülő elemek – az alkáli feszültsége folyamatosan csökken | Elemcsere; ez tananyag is: a tápfeszültség rendszerparaméter! |
| Egy szenzorcsatorna mindig ugyanazt mutatja | Kábel; rossz lábra kötve; a csatorna a kalibrációnál nem látott kontrasztot | Nyers érték figyelése Serial Monitoron ujjal takargatva; újrakalibrálás |
| Kalibráció után „FIGYELEM: keves kontraszt" | A húzogatásnál az a szenzor nem járt a vonal fölött; túl magasan van a panel | Újrakalibrálás lassabb, szélesebb húzogatással; 2 mm-es rés ellenőrzése |
| A robot fényes padlórészen / napfoltban megőrül | Tükröződés, környezeti IR fény | Matt pályaalap; pálya árnyékba; kalibrálás a tényleges fényviszonynál |
| Egyenesen jó, kanyarban kisodródik | Kp kicsi; vagy túl gyors az adott hangoláshoz | Kp emelése VAGY BASE_PWM csökkentése – egyszerre csak egyet! |
| A vonal körül cikkcakkban leng | Kp túl nagy; Kd hiányzik | Hangolási recept 2–3. lépés |
| Hajtűkanyarban elveszti a vonalat és körbe pörög | A vonalvesztő ág rossz irányba fordít; túl gyors | A readPosition() vonalvesztő logikája jó irányba tér-e vissza? Sebesség vissza a hajtűre |
| Az enkóder számláló nem változik / ugrál | Enkóder táp hiányzik; A/B csatorna felcserélve; zaj | 3,3 V az enkóderen? Kézzel forgatva monoton nő/csökken? |
| Feltöltésnél „Failed to connect" | ESP32 boot-gomb probléma | Feltöltés indításakor tartsd nyomva a BOOT gombot, amíg a „Connecting…" fut |

**Amikor minden furcsa egyszerre**

Ha egy robot több, egymással összefüggéstelen hibát produkál (szenzor is ugrál, motor is akadozik), az szinte mindig **táp vagy föld**: merülő elem, laza GND-kábel, rossz csatlakozás. Első lépés ilyenkor: friss elem + minden kábel újradugása + GND-pont ellenőrzése. Csak ezután kód.

# 7. Mentori háttéranyag

## 7.1 A tábor előtti teendők (időrendben)

- ☐ **T−10 nap:** tesztdarab nyomtatása (motorbölcső + szenzortartó), illesztés ellenőrzése az N20-szal
- ☐ **T−9 nap:** TSM-5 bemérése 3,3 V-ról: fehér/fekete kontraszt multiméterrel → ha rendben, vázsorozat nyomtatása indul (7 db ≈ 3 nap)
- ☐ **T−7 nap:** egy teljes referencia-robot összeszerelése, kódváz feltöltése, mind a 6 tanulási lépcső végigjátszása – ITT derülnek ki a meglepetések, nem a táborban
- ☐ **T−5 nap:** gyakorlópálya megépítése, kalibráció és hangolás a valós pályán; a kódváz véglegesítése a mért értékekkel
- ☐ **T−4 nap:** minden laptopra Arduino IDE + ESP32 csomag + próbafeltöltés
- ☐ **T−3 nap:** a maradék 6 robot mechanikai előszerelése (motorbölcső, kerék, görgő – a kábelezés a gyerekeké marad!)
- ☐ **T−2 nap:** versenypálya megépítése és letakarása/elzárása; tartalék alkatrészek dobozolása csapatonként
- ☐ **T−1 nap:** terem berendezése: 5 csapatasztal + pályaasztal + mentorasztal; nyomtatott segédletek

## 7.2 Hol segíts – és hol ne

A mentor legnehezebb feladata a **nem-segítés**. Irányelvek:

- **Hardverhibánál segíts gyorsan.** Egy kicsúszott kábel megtalálása nem tananyag, csak frusztráció. A hibakeresési táblázat első három oszlopát a mentor bátran mutassa meg.
- **Hangolásnál csak kérdezz.** „Mit mond a Plotter? Nőtt vagy csökkent a lengés? Mit írtál a naplóba?" A jó Kp-t megtalálni az ő élményük.
- **Kódot soha ne gépelj a gyerek gépén.** Ha muszáj mutatni, papírra vagy táblára.
- **A 4. naptól a mentor „csak" versenybíró** – ez előre kihirdetve a csapatoknak is keretet ad.

## 7.3 Korosztályi különbségek kezelése

A 11 éves és a 18 éves ugyanazt a kódot kapja – a differenciálás nem a feladatban, hanem a **mélységben** van:

| Lépcső | Alapszint (mindenkinek) | Mélyítés (aki kéri) |
|---|---|---|
| Motor | PWM = „gyorsabban kapcsolgatjuk = gyorsabban megy" | kitöltési tényező, H-híd működése, miért 20 kHz fölött néma |
| Szenzor | fekete elnyeli az IR-t, fehér visszaveri | fototranzisztor, CTR, miért kell kalibrálni (darabszórás) |
| Pozíció | súlyozott átlag = „hol van a vonal súlypontja" | centroid, mi történik a képlettel vonalvesztéskor (0/0!) |
| P-szabályzó | nagyobb hiba → nagyobb kormányzás | zárt hurok, erősítés, stabilitáshatár |
| D-tag | „fékez, mielőtt túlszaladnánk" | derivált ≈ különbség, miért zajérzékeny, miért nincs I-tag (integrátor-felcsavarodás) |
| Enkóder | számolja a fordulatot | kvadratúra, irányfelismerés fáziskésésből, megszakítások |

**Miért nincs I-tag a kódban?**

Szándékosan. Vonalkövetésnél az állandósult hiba nem probléma (a pálya folyton változik), az integrátor viszont felcsavarodik a hajtűkanyarban és nehezen érthető hibákat okoz. A PD itt a szakmailag helyes választás is – és ha egy idősebb gyerek rákérdez a „PID" I-jére, ez maga a beszélgetés, amiért érdemes volt eljönnie.

## 7.4 Ha valami nagyon félremegy – B tervek

| Vészhelyzet | B terv |
|---|---|
| Egy robot végleg meghal (pl. ESP32 tönkremegy) | A tartalék robotból/alkatrészből azonnali csere – ezért van összeszerelve a tartalék! |
| Egy csapat a 3. nap végén sem tud bang-banggel körbeérni | A mentor a referencia-robot hangolási értékeit adja át kiindulásnak, és negyedórát céltudatosan együtt debuggol |
| Sok a fényprobléma a teremben | Pálya áthelyezése ablaktól el; függöny; végső esetben: kalibráció óránként |
| Kész a PD mindenkinek már a 3. nap délben | Haladó feladatok előrehozása (4. napi lista) + „fordított feladat": rontsd el szándékosan a Kp-t és írd le, mit látsz |
| Wifi/internet nincs a teremben | Semmi gond: a tábor teljes egésze offline működik, az IDE és a csomagok előre telepítve |

## 7.5 Költség- és eszközösszesítő

| Tétel | Robotonként | 6 robotra (5 + tartalék) |
|---|---|---|
| N20E-06-500 motor | 2 db | 12 db + 1 tartalék |
| 42 mm kerék | 2 db | 12 db + tartalék |
| TB6612FNG-M | 1 db | 6 db + tartalék |
| TSM-5 | 1 db | 6 db + tartalék |
| ESP32 DevKit + kifejtő | 1+1 db | 6+6 db + tartalék |
| 4×AA elemtartó | 2 db | 12 db |
| AA elem | 8 db + napi csere a motorágon | ~60–80 db a hétre |
| 1000 µF + 100 nF | 1+1 db | a rendelt 50/100 db-ból |
| Nyomtatott váz (PETG) | 1 db | 7 db (1 tartalék) |

A teljes elektronikai beszerzés a HESTORE-rendelés szerint bruttó ~138 000 Ft; a pálya (papír + szalag), az elemek és a nyomtatási anyag ezen felül ~15–20 000 Ft.

## 7.6 Merre tovább? (a zárónapra)

A tábor végén érdemes megmutatni a folytatás útjait – a legjobb pillanat, amikor a versenyláz még forró:

- **Ugyanez a robot, új feladat:** önegyensúlyozó átépítés (a motor ugyanaz az 500 RPM-es!), útvonalmérés enkóderrel, két robot kergetőzése.
- **Versenyek:** iskolai robotverseny, OTIO, TDK – a cég robotlaborjában van hely és mentor.
- **A kód hazavihető:** a teljes kódváz és ez a segédlet a csapatoké. Egy ESP32 + pár alkatrész otthon is elérhető zsebpénzből.