ROBÒTICA · 3r ESO

UNITATS 1-2: COMPUTACIÓ FÍSICA · INTRODUCCIÓ A ARDUINO · L'IDE

Nom: _____ Curs/Grup: _____ Data: _____

Setmanes 1-2. Al final sabràs:

- Explicar què és la computació física (sensors → microcontrolador → actuadors).
- Identificar la placa Arduino UNO i els seus components.
- Instal·lar i fer servir l'IDE d'Arduino (verificar, carregar, monitor sèrie).
- Pujar el teu primer programa (Blink) a la placa.

1. Computació física amb Arduino

La computació física consisteix a programar objectes físics perquè interactuïn amb el món real. El flux de treball de tot sistema de control és:

ENTRADA
Sensor (llum, pulsació, distància...)

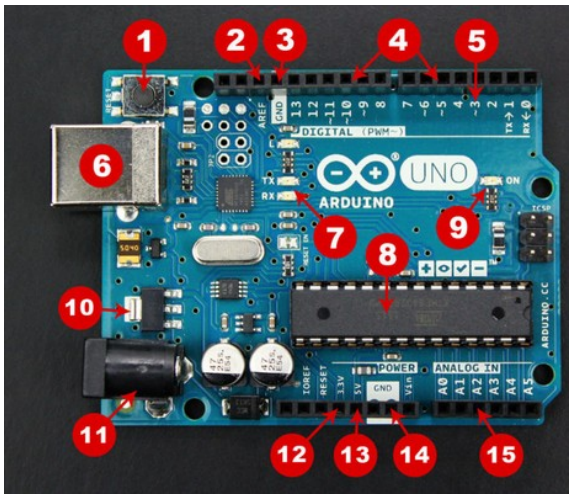
> *OBJECTIUS DEL CURS: fonaments de pensament computacional · programació textual (C++ amb l'IDE d'Arduino) · entrades i sortides digitals · entrades analògiques i sortides PWM · servomotors · comunicació sèrie.*

2. Què és Arduino?

> *DEFINICIÓ: La placa Arduino és un dispositiu electrònic open-source basat en un microcontrolador programable que permet llegir entrades (llum ambiental, pulsació d'un botó, presència d'un objecte...), processar aquesta informació i produir una sortida (moviment d'un motor, encesa d'un LED, so...).*

Arduino és present en milions de projectes: des de robots educatius i projectes escolars fins a microsatèl·lits, impressores 3D i sistemes de control industrials. El

seu èxit ve de popularitzar tecnologies que abans només entenien els enginyers, gràcies a la comunitat open-source.



1- Botó de Reset – Això reiniciarà qualsevol codi que estigui carregat a la placa Arduino 2
2- Pin AREF – Significa "Analog Reference" (Referència Analògica) i s'utilitza per establir un voltatge de referència extern
3- Pin de terra (Ground) – Hi ha uns quants pins de terra a l'Arduino i tots funcionen igual
4- Entrada/Sortida digital – Els pins 0-13 es poden utilitzar per a entrada o sortida digital
5- PWM – Els pins marcats amb el símbol (~) poden simular una sortida analògica
6- Connexió USB – S'utilitza per alimentar el teu Arduino i carregar programes (sketches)
7- TX/RX – LEDs indicadors de transmissió i recepció de dades
8- Microcontrolador ATmega – Aquest és el cervell i és on s'emmagatzemen els programes
9- LED indicador d'engegada (Power) – Aquest LED s'il·lumina sempre que la placa està endollada a una font d'alimentació
10- Regulador de voltatge – Això controla la quantitat de voltatge que entra a la placa Arduino
11- Connector d'alimentació DC (Power Barrel Jack) – S'utilitza per alimentar el teu Arduino amb una font d'alimentació externa
12- Pin de 3.3V – Aquest pin subministra 3.3 volts de potència als teus projectes
13- Pin de 5V – Aquest pin subministra 5 volts de potència als teus projectes
14- Pins de terra (Ground) – Hi ha uns quants pins de terra a l'Arduino i tots funcionen igual
15- Pins analògics – Aquests pins poden llegir el senyal d'un sensor analògic i convertir-lo a digital

La placa Arduino UNO amb els seus components.

La placa pot programar-se en diversos llenguatges

- TEXTUALS: C amb l'IDE d'Arduino (el que usarem en aquest curs).
- GRÀFICS: per blocs, com versions de Scratch.

Alimentació

- USB: proporciona 5 V (la via habitual quan programa).
- JACK d'alimentació: pila de 9 V o font recomanada entre 7 i 12 V.

Pins d'alimentació (per alimentar el circuit del breadboard)

Pin
3.3 V
5 V

GND
Vin

Pins d'entrada i sortida

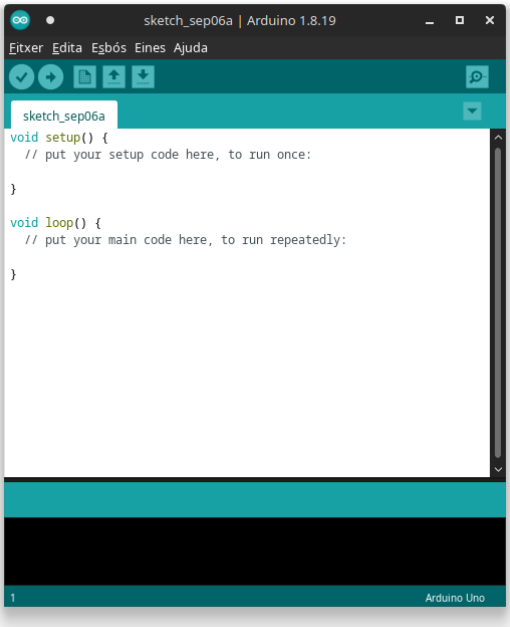
- DIGITAL: sortida 0 V (LOW) o 5 V (HIGH); entrada: 0-2 V es llegeix LOW i 3-5 V HIGH.
- SORTIDA ANALÒGICA: de 0 a 5 V en 256 valors intermedis (precisió de 8 bits). En realitat és un PWM (senyal modulada per amplada de pols).
- ENTRADA ANALÒGICA: de 0 a 5 V en 1024 valors (precisió de 10 bits).
- La intensitat màxima per pin és de 40 mA.

El circuit es munta sobre una placa de prototips (BREADBOARD o PROTOBOARD) i les connexions es fan amb cables JUMPER (no es trenquen als sòcols).

3. Programant Arduino: estructura bàsica

Tot programa Arduino té 3 parts:

Part
1. Declarar variables: Serveix per crear i donar un nom a espais de memòria on guardarem dades (com nombres o estats) que el programa farà servir i podrà modificar.
2. void setup(): És una funció que s'executa **només una vegada** quan s'encén o es reinicia l'Arduino. S'utilitza per configurar aspectes inicials, com dir si un pin funcionarà com a entrada o com a sortida.
3. void loop(): És una funció que s'executa **contínuament en bucle** (un cop rere l'altre, de manera infinita) un cop acaba el setup(). És on va el codi principal que controla el comportament de l'Arduino.



Nomenclatura de variables

Els noms de variables NO poden començar per número ni portar símbols o espais. Noms vàlids: ledPin, estatAnterior, comptaPulsacions, mevaVariable, lecturaSensor...

A la programació d'Arduino (que està basada en C++), existeixen diferents **tipus de variables** segons la dimensió i el tipus de dada que necessitis guardar. Utilitzar el tipus adequat t'ajuda a estalviar memòria a la placa.

Aquí tens els principals tipus de variables que s'utilitzen habitualment:

- **int (Enter):** Guarda nombres enters, tant positius com negatius, des de −32.768 fins a 32.767. Ocupa 2 bytes.
    - *Exemple:* int comptador = 150;
- **float (Decimal):** S'utilitza per guardar nombres amb decimals. Ocupa 4 bytes i pot contenir valors molt grans o petits.
    - *Exemple:* float temperatura = 23.5;
- **bool o boolean (Booleà):** Només pot tenir dos valors possibles: true (cert/1) o false (fals/0). Ideal per a estats com encès/apagat o obert/tancat.
    - *Exemple:* bool estatPolsador = true;
- **char (Caràcter):** Emmagatzema un sol caràcter (una lletra o símbol) utilitzant cometes senzilles. Ocupa 1 byte.
    - *Exemple:* char lletra = 'A';
- **byte:** Guarda nombres enters sense signe, des de 0 fins a 255. Ocupa 1 byte, ideal per estalviar memòria.
    - *Exemple:* byte lluminositat = 200;
- **long (Enter llarg):** Per a nombres enters molt més grans, des de −2.147.483.648 fins a 2.147.483.647. Ocupa 4 bytes (molt útil, per exemple, per mesurar temps en microsegons amb millis()).
    - *Exemple:* long temps = 123456789;
- **String (Cadena de text):** S'utilitza per guardar frases o paraules senceres (text). Va entre cometes dobles.
    - *Exemple:* String missatge = "Hola, mon!";

Activitat 1

Identifica quins noms són vàlids i quins no: 2led, mi_led, led pin, ledPin, contador#1. Corregeix els invàlids.

4. Instal·lació de l'IDE d'Arduino

Passos per instal·lar-lo al teu ordinador:

- 1. Visita https://www.arduino.cc/en/software

- 2. Selecciona l'última versió per al teu sistema operatiu i descarga el paquet.
- 3. Obre'l i segueix l'assistent d'instal·lació (accepta permisos i termes).
- 4. Confirma instal·lar TOT, inclosos els drivers de les placas Arduino.
- 5. Accepta la carpeta proposada per defecte i prem Instalar.
- 6. Quan acabi, prem Cerrar i obre l'IDE des de la icona.

5. Usant l'IDE: els 6 icones bàsics

Icona
VERIFICAR
CARREGAR
NOU
OBRIR
GUARDAR
MONITOR SÈRIE

Localitza'ls al teu programa.

Menús importants

- Archivo → Ejemplos: col·lecció de sketches inclosos que ensenyen tots els comandos (organitzats per temes). MOLT ÚTIL.
- Herramientas → Tarjeta: selecciona la placa que uses (Arduino UNO: ATmega328P, 16 MHz, 6 entrades analògiques, 14 E/S digitals, 6 PWM).
- Herramientas → Puerto: selecciona el port on està connectada la placa (/dev/ttyACMx a Linux, COM4+ a Windows).
- Sketch → Incluir biblioteca: afegeix llibreries (#include) al començament del codi.
- Herramientas → Formato automático: dona format llegible al programa (sangries).

Activitat 1

Obre l'IDE i identifica els 6 icones. Obre Archivo → Ejemplos → Basics i mira quants exemples hi ha. Apunta'n 3 que et cridin l'atenció:

1. _____ 2. _____ 3. _____

5. PROPOSTA 1 · El teu primer programa: Blink

Segueix aquest passos que ara anirem desglossant pas a pas per comprovar que tot funciona:

- 1. Obre l'IDE des de l'escriptori.
- 2. Connecta la placa per USB (si és la primera vegada, pot demanar permís per instal·lar drivers).
- 3. Herramientas → Tarjetas: selecciona Arduino UNO (o Arduino BT si uses la BQ ZUM).
- 4. Herramientas → Puerto: selecciona el port de la placa (TRUC: desconecta-la i mira quin port desapareix).
- 5. Archivo → Ejemplos → Basics → Blink.
- 6. Prem CARREGAR. Durant la pujada els LEDs Tx i Rx parpellegen. Al final: 'Subido' + memòria usada.

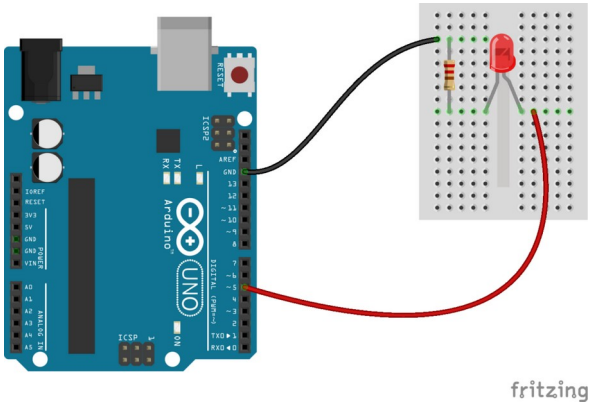
La sortida amb digitalWrite

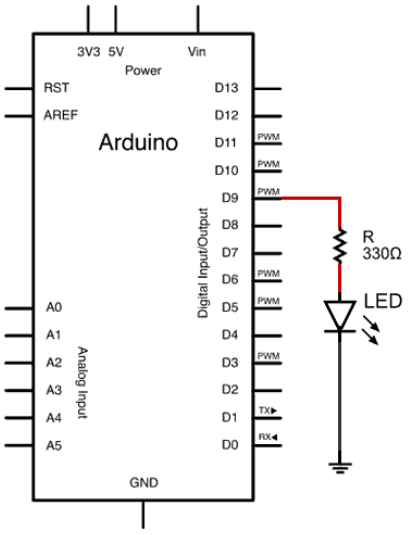
Una sortida digital només val 0 V (LOW) o 5 V (HIGH). La funció és:

> *digitalWrite(pin, HIGH); // encén · digitalWrite(pin, LOW); // apaga*

El cablejat a la placa de prototips i l'esquema

⚠ Sempre que connectem un LED cal posar una RESISTÈNCIA en sèrie (entre 100 Ω i 1 kΩ) per evitar que es cremi.





Muntatge del parpadeig: LED, resistència i GND al breadboard. Cablejat: LED al pin 13 (fila vermella), resistència 220 Ω, tornada a GND.

Pràctica 1: Parpelleig d'un LED

- MUNTATGE: LED al pin 13 + resistència 220 Ω a GND.
- PROGRAMA: setup → pinMode(13, OUTPUT); loop → digitalWrite HIGH, delay(1000), LOW, delay(1000).

S’expressa així al programa IDE:

El codi de Blink

```
void setup() {
pinMode(13, OUTPUT);
}
void loop() {
digitalWrite(13, HIGH);
delay(1000);
digitalWrite(13, LOW);
delay(1000);
}
```

Aquest programa encén el LED integrat al pin 13 i el fa parpellejar 1 segon indefinidament. Observa l'estructura de qualsevol sketch:

- void setup(): s'executa UNA vegada en engegar (configuració).
- void loop(): es repeteix indefinidament (el programa).
- El programa es guarda en memòria FLASH: no s'esborra en desconnectar. Cada vegada que alimentis la placa, s'iniciarà automàticament.
- ⚠ La placa només pot emmagatzemar UN programa: si en puges un de nou, esborres l'anterior.

  **Observa que no es declaren variables perquè aquí no n’hi ha. Pregunta’t què és una variable... altre cop per reafirmar el concepte.*

Activitat 2 — Modifica el Blink

Puja el Blink original i comprova. Després modifica el codi:

- a) Canvia els delays a 200 ms → què passa?
- b) Fes que el LED estigui 3 segons encès i 0,5 apagat.
- c) EXTRA: fes un patró tipus 'tick-tick' (2 curtes + 1 llarga).

Prova
a
b
c

Pràctica 2: Llum que avança a través de 5 LEDs

5 LEDs alineats que s'encenen en seqüència. Fa servir un bucle *for* per recórrer els pins.

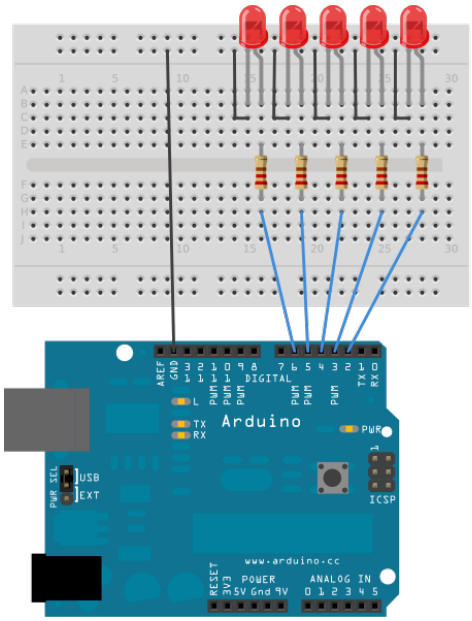
Codi esquelet que posaríem al *void loop()*:

```
for (int i = 4; i < 9; i++) {
digitalWrite(i, HIGH);
delay(100);
digitalWrite(i, LOW);
}
```

Escriu el codi que hem de posar a *void setup()* per configurar els pins de sortida dels 5 leds.

Registre

Pràctica
Parpadeig
5 LEDs en seqüència



Com que només s'encén un LED cada vegada, es podria fer el muntatge amb una sola resistència?

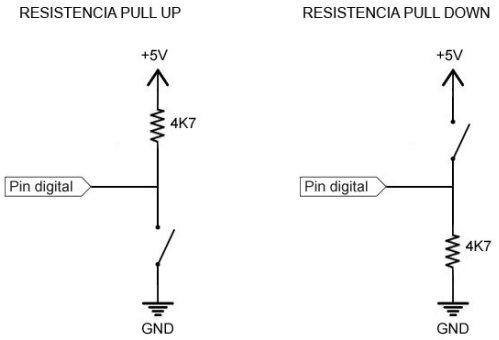
Muntatge dels 5 leds.

6. Entrades digitals: digitalRead i el pulsador

Per llegir un botó cal configurar el pin com a INPUT i fer servir una RESISTÈNCIA

DE DRENATGE (pull-up o pull-down) perquè la lectura no floti.

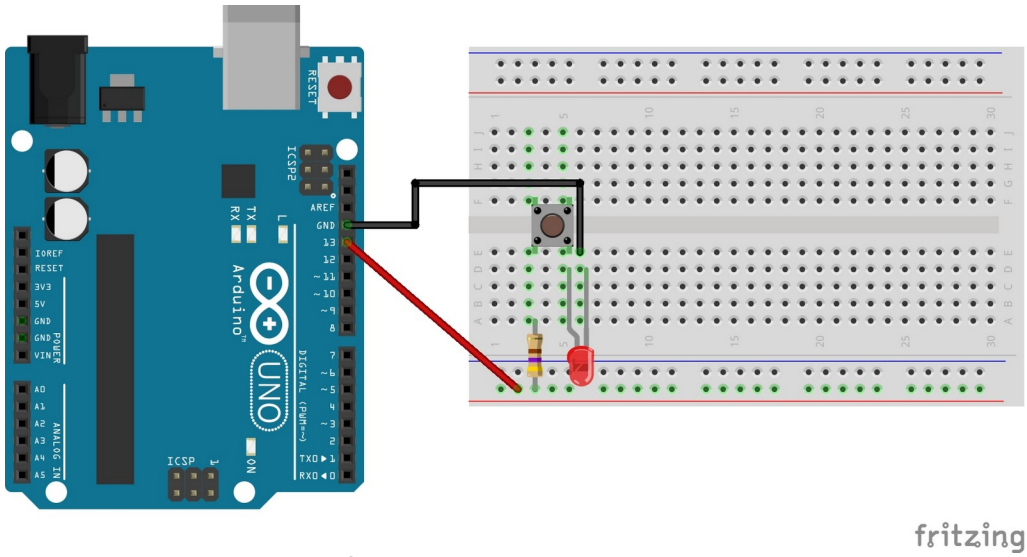
> *digitalRead(pin) retorna HIGH (5 V) o LOW (0 V) segons l'estat del botó.*



Pull-up – En prémer el posador el pin de la placa rep 0v.

Pull-down - En prémer el posador el pin de la placa rep 5v.

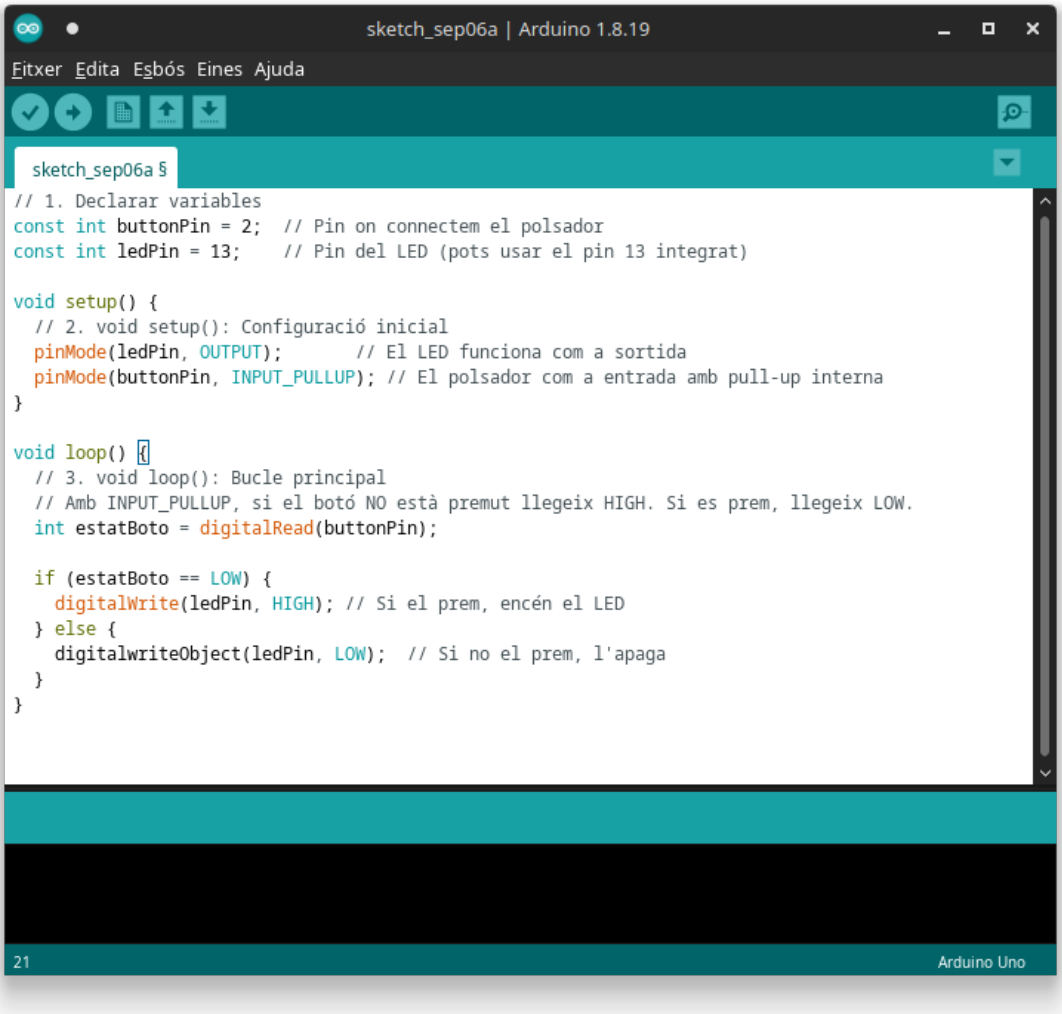
Muntatge del polsador amb la resistència de drenatge.



Muntatge del polsador amb la resistència de drenatge a la placa.

Pràctica 3: Canviar l'estat d'un LED amb un polsador

- VARIANT A: mentre estigui premut, el LED encès (lectura directa).



- VARIANT B: cada pulsació CANVIA l'estat del LED (compta pulsacions i alterna).

Per aconseguir que **cada vegada que premis el polsador** el LED canviï d'estat (s'encengui si estava apagat, o s'apagui si estava encès), necessitem detectar el moment exacte en què el botó passa de no estar premut a estar premut (l'anomenat *front de pujada* o canvi de flanc).

Si no féssim això, el programa canviaria l'estat desenes de vegades per segon mentre mantinguéssis el dit a sobre del botó.

Aquí tens el codi:

Quina diferència hi ha al codi entre la variant A i la B?

Registre

Pràctica

Pulsador variant A
Pulsador variant B

7. Atzar: random

La funció random() s'utilitza per generar nombres aleatoris (a l'atzar), la qual cosa és molt útil per a projectes com jocs, efectes de llums imprevisibles o simulacions.

Sintaxi principal:

- random(max): Genera un nombre enter entre 0 i max - 1.
- random(min, max): Genera un nombre enter entre el min i max - 1.

Exemple pràctic:

> *// Dins del loop() o d'una funció:*
>
> *int tempsEspera = random(100, 1000); // Genera un nombre aleatori entre 100 i 999 mil·lisegons*
>
> *delay(tempsEspera);randomSeed(analogRead(A0)); // inicialitza el generador aleatori*
>
> *random(min, max); // retorna un número aleatori entre min i max-1*

Pràctica 4: LED imitant una espelmaa

Un LED groc/taronja amb brillantor i temps ALEATORIS imita el parpelleig d'una flama. Fa servir PWM (analogWrite) + random.

Aquest és un codi que pots fer servir:

El LED espelma amb valors aleatoris.

Repàs i autoavaluació

Rúbrica: 1 = No assolit · 2 = Satisfactori · 3 = Notable · 4 = Excel·lent

Objectiu

Identifico els elements de la placa UNO i les seves límits (volts, mA)
Expliqu l'estructura variables/setup/loop d'un programa Arduino
Controlo sortides digitals amb digitalWrite i LEDs amb resistència
Llegeixo entrades digitals amb pulsador i resistència de drenaje
He fet les 4 pràctiques del bloc i funcionen
He construït el semàfor de la producció

Glossari

- Computació física: programar objectes físics per interactuar amb el món.
- Sensor: element que llegeix el món (entrada).
- Actuador: element que actua (sortida: LED, motor, so).
- IDE: entorn de desenvolupament integrat (programari per programar).
- Sketch: programa d'Arduino (.ino).
- Verificar / Carregar: comprovar el codi / compilar i pujar-lo.
- Monitor sèrie: finestra de comunicació PC ↔ placa.
- Flash: memòria no volàtil: el programa no s'esborra en apagar.
- Breadboard: placa de prototips per muntar sense soldar.
- Jumper: cable de connexió per al breadboard.
- HIGH / LOW: 5 V / 0 V en una sortida digital.
- PWM: senyal modulada per amplada de pols (simula analògica).
- digitalWrite / digitalRead: escriure / llegir un pin digital.
- Pull-up / pull-down: resistència de drenatge per a entrades digitals.
- random: funció d'atzar.

ROBÒTICA · 3r ESO

UNITAT 3: SENYALS DIGITALS — SORTIDES

Nom: _____ Curs/Grup: _____ Data: _____

Setmanes 3-5. Al final sabràs:

- Explicar què és un senyal digital (bits: 0 i 1) davant l'analògic.
- Controlar els 14 pins digitals amb pinMode i digitalWrite.
- Programar un semàfor amb delays (retards).
- Crear subrutines pròpies (funcions) per simplificar el codi.
- Fer servir un brunzidor com actuador de so.

1. Senyals digitals i analògiques

Els dispositius electrònics digitals processen dades codificades amb només dos valors: 1 i 0 (binary digit = bit). Això és perquè el component que els processa, el transistor, treballa entre dos estats: CORTE (0) i SATURACIÓN (1).

El món que ens envolta és ANALÒGIC: la informació pot prendre infinitat de valors (la temperatura varia contínuament durant el dia). Es pot convertir una senyal analògica en digital mitjançant un MOSTREIG.

2. Sortides digitals a la placa

La placa UNO disposa de 14 sortides digitals accessibles per PINS. Una sortida digital pot estar activada (5 V, HIGH) o desactivada (0 V, LOW):

- Si connectes un LED al pin i tanques el circuit a GND: quan la sortida està HIGH circula corrent i el LED s'encén.
- ⚠ Tots els actuadors connectats a sortides digitals reben 5 V i MÀXIM 40 mA: això limita què pots connectar directament (LEDs, brunzidors, relés, petits servos).

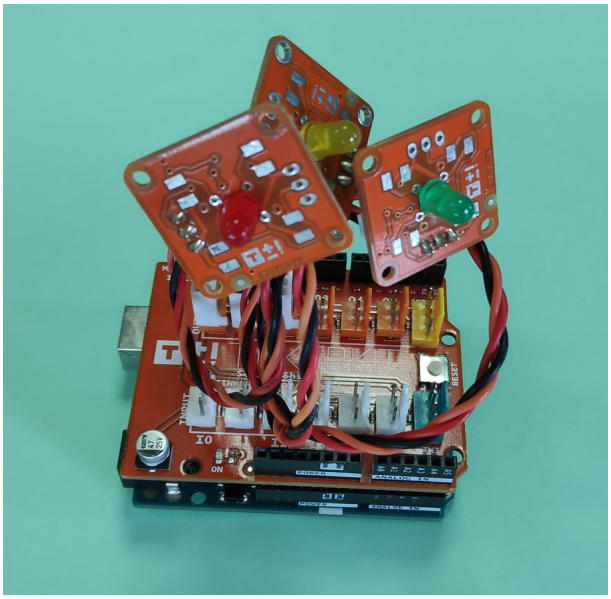
3. Instruccions (comandes): pinMode i digitalWrite

> *pinMode(pin, uso) — defineix si el pin és OUTPUT (sortida) o INPUT (entrada). Es defineix UNA vegada, dins void setup().*

> *digitalWrite(pin, estat) — posa el pin a HIGH (5 V) o LOW (0 V).*

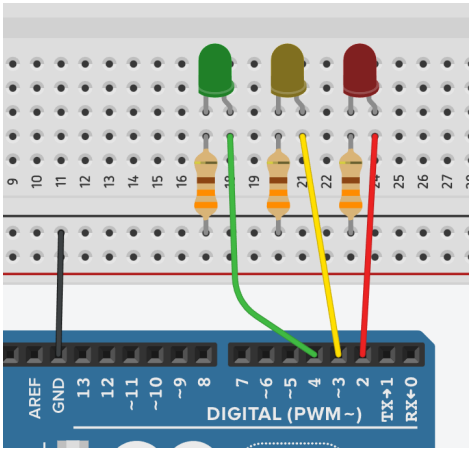
EXEMPLE: digitalWrite(13, HIGH) posa 5 V al pin 13.

4. PROPOSTA 2 · El semàfor



Muntatge del semàfor: 3 LEDs (vermell, groc, verd) connectats a la placa.

Objectiu: un semàfor per vehicles que passi de verd a groc, de groc a vermell i de vermell a verd. 10 segons en vermell i verd, 4 segons en groc.



Cablejat del semàfor: pin 2→LED vermell, pin 3→groc, pin 4→verd, resistències a GND.

- 1. Connecta un LED vermell al pin digital 11, un groc al 10 i un verd al 9 (amb el kit o amb protoboard i resistències).
- 2. pinMode dels 3 pins com OUTPUT dins setup().
- 3. Per cada estat: quins LEDs han d'estar encesos? (vermell = només pin 11 HIGH).
- 4. Completa el programa amb delays i puja'l a la placa.

El codi complet:

```
/* PROGRAMA SEMÀFORO
Programa un semàfor amb tres leds i la placa Arduino */
```

```
void setup()
{
pinMode(11, OUTPUT);
pinMode(10, OUTPUT);
pinMode(9, OUTPUT);
}
void loop()
{
digitalWrite(11, HIGH); // Vermell durant 10s
digitalWrite(10, LOW);
digitalWrite(9, LOW);
delay(10000); // delay espera en mil·lisegons
digitalWrite(11, LOW); // Verd durant 10s
digitalWrite(10, LOW);
digitalWrite(9, HIGH);
delay(10000);
digitalWrite(11, LOW); // Groc durant 4s
digitalWrite(10, HIGH);
digitalWrite(9, LOW);
delay(4000);
}
```

Aclariments importants:

- Un programa és una seqüència d'ordres separades per punt i coma que s'executen una rere l'altra.
- Tot programa Arduino ha de tenir void setup() (una vegada) i void loop() (indefinidament).
- Tot i que les instruccions s'executen seqüencialment, l'efecte és instantani (microsegons entre instruccions).

Activitat 3 — El teu semàfor

Munta el semàfor i puja el codi. Després modifica'l:

- a) Fes que el verd durés només 5 segons.
- b) Afegeix un 4t LED o canvia els temps de pas (per exemple verd 8s).
- c) EXTRA: semàfor de vianants (un LED verd i un vermell que canviïn quan el vehicle és vermell).

5. PROPOSTA 3 · SOS amb un LED (subrutines)

Objectiu: enviar el senyal de socors SOS en codi Morse amb un LED: S (...) i O (---). Fem SERVIR SUBRUTINES per simplificar:

1. El pin 11 s'encén i apaga 3 vegades cada 0,3 s → això és la funció S():

```
void S()
```

```
{
for (int i = 0; i < 3; i++)
{
digitalWrite(11, HIGH);
delay(300);
digitalWrite(11, LOW);
delay(300);
}
}
```

2. La funció O() és igual però amb encès de 0,9 s:

```
void O()
{
for (int i = 0; i < 3; i++)
{
digitalWrite(11, HIGH);
delay(900);
digitalWrite(11, LOW);
delay(300);
}
}
```

3. El programa principal alterna S, O, S amb 1 s de descans. Les subrutines s'escriuen ABANS de void setup():

```
void setup()
{
pinMode(11, OUTPUT);
}
void loop()
{
S();
O();
S();
delay(1000);
}
```

Per què usar subrutines? El programa principal és més senzill de llegir, pots dividir el problema en parts més fàcils i depurar-les per separat si hi ha errors.

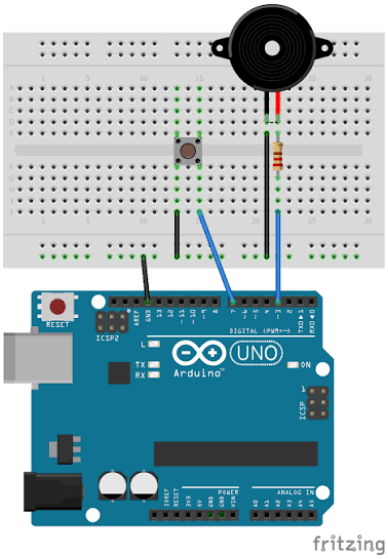
Activitat 4 — El teu SOS

Munta el LED i puja el programa. Després:

- a) Fes que el missatge sigui 'SOS SOS' amb descans de 2 s entre missatges.
- b) Canvia la velocitat del Morse (més ràpid).
- c) EXTRA: afegeix una altra lletra (per exemple E = un sol punt).

6. PROPOSTA 4 · SOS amb un timbre o brunzidor

Fem servir un nou actuador: el TIMBRE o brunzidor. Quan se li aplica voltatge, vibra emetent un to. Segons el voltatge, canvia la freqüència de vibració i per tant el to.



Muntatge amb protoboard: el brunzidor es connecta amb resistència (100 Ω-1 kΩ).

El codi és el mateix que el de la Proposta 2 (SOS): pots usar la mateixa placa. Observa el resultat: ara escoltaràs el SOS en lloc de veure'l.

Activitat 5

Connecta el brunzidor pin 11 i puja el codi SOS. Prova a canviar els delays i escolta la diferència. EXTRA: canvia el codi per fer una senyal diferent (per exemple 'ABC' o una sirena alternativa).

Repàs i autoavaluació

Rúbrica: 1 = No assolit · 2 = Satisfactori · 3 = Notable · 4 = Excel·lent

Objectiu
Diferencio senyal digital (bits) i analògica (contínua)

Uso pinMode i digitalWrite per controlar pins digitals
He programat un semàfor funcional amb 3 LEDs i delays
Sé crear subrutines (funcions) per simplificar el codi
He fet el SOS amb LED i amb brunzidor

Glossari

- Bit: dígits binari: 0 o 1.
- Tall / Saturació: els dos estats del transistor (0 / 1).
- Mostreig: conversió d'analògic a digital.
- pinMode(pin, OUTPUT/INPUT): configura el pin com sortida o entrada.
- digitalWrite(pin, HIGH/LOW): posa 5 V o 0 V al pin.
- delay(ms): pausa el programa en mil·lisegons.
- Subrutina / funció: bloc de codi reutilitzable (void nom()).
- Brunzidor: actuador de so que vibra segons el voltatge.

ROBÒTICA · 3r ESO

UNITAT 4: ENTRADES DIGITALS

Nom: _____ Curs/Grup: _____ Data: _____

Setmanes 6-7. Al final sabràs:

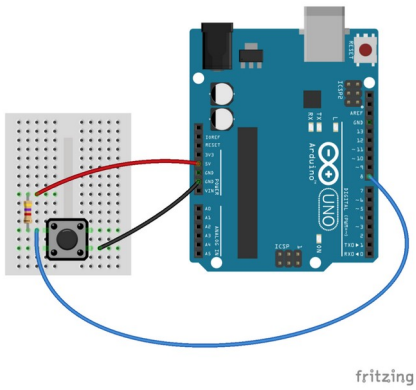
- Connectar sensors (polsador) a les entrades digitals.
- Llegir entrades amb digitalRead.
- Programar el flux complet ENTRADA → PROCÉS → SORTIDA.
- Fer servir variables per guardar estats (toggle).

1. Les entrades digitals

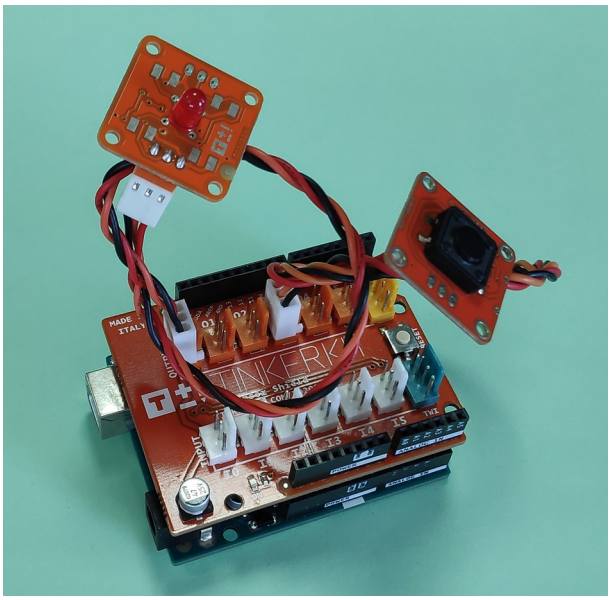
Un dels aspectes més destacables d'Arduino és la facilitat amb què podem connectar sensors externs que detecten canvis a l'entorn. Els sensors es connecten SEMPRE a les entrades.

> *Un sensor converteix una magnitud física (temperatura, llum, pulsació...) en una senyal elèctrica que la placa pot llegir.*

Les entrades digitals només llegeixen DOS estats: HIGH (5 V) o LOW (0 V). El polsador és l'exemple més clar: premut o no.



Cablejat: polsador amb resistència de 10 kΩ a GND (pull-down), LED al pin 11.



Muntatge amb kit: LED + polsador connectats a la placa.

2. PROPOSTA 5 · LED controlat amb polsador

Objectiu: si el pulsador està premut, el LED brilla; en soltar-se, s'apaga. Aquest projecte ens permet apreciar per primera vegada el flux de treball d'un sistema de control:

ENTRADA
Llegir el polsador

- 1. Connecta un LED vermell al pin digital 11 i un polsador al pin digital 6.
- 2. Amb el kit és directe; sense kit, munta amb protoboard i resistències.
- 3. Ambdós pines són d'entrada/sortida: configura el seu ús al setup().

El codi complet (fixa't que usem VARIABLES per als pins per fer-lo més llegible):

```
/* LED CONTROLAT AMB POLSADOR
Si polsem el botó, el led s'il·lumina; si no, es manté apagat */
int pinLed = 11;
int pinBoto = 6;
int valor = 0;

void setup()
{
pinMode(pinLed, OUTPUT);
pinMode(pinBoto, INPUT);
}

void loop()
{
```

```
valor = digitalRead(pinBoto);
digitalWrite(pinLed, valor);
}
```

Com funciona: si el botó està premut, digitalRead retorna HIGH i digitalWrite posa el LED a HIGH també (encès). Si no està premut, LOW i apagat. Per això podem assignar DIRECTAMENT el valor llegit al LED.

Activitat 6

Munta i puja el programa. Prova:

- a) Inverteix la lògica: LED encès sempre, i s'APAGA mentre premis (pista: valor = 1 - lectura).
- b) Afegeix un segon LED que faci el contrari que el primer.

3. PROPOSTA 6 · Toggle: cada pulsació canvia l'estat

Objectiu: cada vegada que premis el botó, el LED canvia d'apagat a encès i viceversa (com un interruptor de paret).

Per fer-ho cal una VARIABLE que recordi l'estat del LED:

```
/* LED TOGGLE AMB POLSADOR
Cada pulsació canvia l'estat del LED */
int pinLed = 11;
int pinBoto = 6;
int estat = 1; // 1 = apagat, -1 = ences
int estatAnterior = 0;

void setup()
{
pinMode(pinLed, OUTPUT);
pinMode(pinBoto, INPUT);
}

void loop()
{
int lectura = digitalRead(pinBoto);
if (lectura == HIGH && estatAnterior == LOW) // acabem de polsar
{
estat = -estat; // canviem d'estat
if (estat == -1)
{
digitalWrite(pinLed, HIGH); // encenem
}
```

```
else
{
digitalWrite(pinLed, LOW); // apaguem
}
}
estatAnterior = lectura;
}
```

Conceptes nous en aquest codi:

- VARIABLE: espai de memòria reservat per guardar un valor i usar-lo o modificar-lo durant l'execució. Als variables els posem noms que indiquin per a què serveixen (estat, estatAnterior...).
- CONDICIONAL if: executa un bloc només si es compleix la condició.
- DETECCIÓ DE FLANC: comparem amb el valor ANTERIOR per detectar la pulsació nova (no que estigui premut continu).

Activitat 7 — El teu interruptor intel·ligent

Munta i puja el programa toggle. Després:

- a) Fes que el LED parpellegi quan està en mode 'encès' (pista: millis() o delay dins el if).
- b) EXTRA: 2 LEDs alternats (cada pulsació en canvia un).

El teu codi modificat (prova'l també!):

Repàs i autoavaluació

Rúbrica: 1 = No assolit · 2 = Satisfactori · 3 = Notable · 4 = Excel·lent

Objectiu
Connecto un pulsador com entrada digital

Llegeixo entrades amb digitalRead i guarde el valor en una variable
Expliqu el flux ENTRADA → PROCÉS → SORTIDA d'un sistema de control
Fa servir condicionals if i detecció de flanc (toggle)
He fet les dues propostes i les modificacions

Glossari

- Sensor: converteix una magnitud física en senyal elèctrica (entrada).
- digitalRead(pin): llegeix HIGH o LOW d'un pin digital.
- Variable: espai de memòria amb nom per guardar un valor.
- Condicional if: executa un bloc només si es compleix la condició.
- Toggle: canviar d'estat en cada pulsació.
- Detecció de flanc: comparar amb l'estat anterior per detectar la pulsació nova.
- Flux de control: entrada → procés → sortida.

ROBÒTICA · 3r ESO

UNITAT 5: ENTRADES ANALÒGIQUES · PWM · SERVOMOTORS

Nom: _____ Curs/Grup: _____ Data: _____

Setmanes 8-9. Al final sabràs:

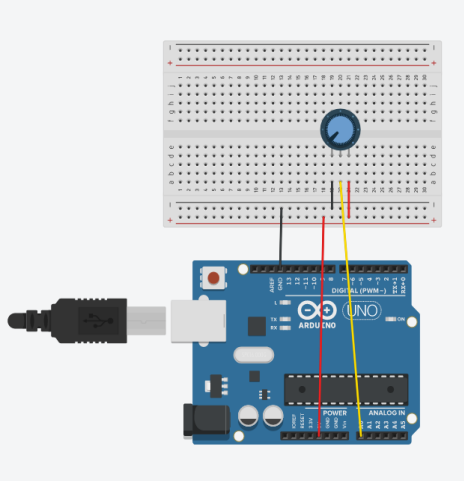
- Llegir sensors analògics amb analogRead (0-1023).
- Controlar sortides PWM amb analogWrite (0-255).
- Reescalar valors amb map().
- Posicionar un servomotor de rotació limitada amb Servo.h.

1. Entrades analògiques: analogRead

Les entrades analògiques (A0-A5 a la UNO) llegeixen tensions de 0 a 5 V i les converteixen en un valor entre 0 i 1023 (precisió de 10 bits).

> *analogRead(A0) — retorna un enter entre 0 (0 V) i 1023 (5 V)*

El potenciòmetre és el sensor analògic més senzill: és una resistència variable. En girar el seu eix obtenim un voltatge variable entre 0 i 5 V segons quanta gira.



Potenciómetre en breadboard: GND, 5V i senyal a A1.

2. Sortides 'analògiques': PWM

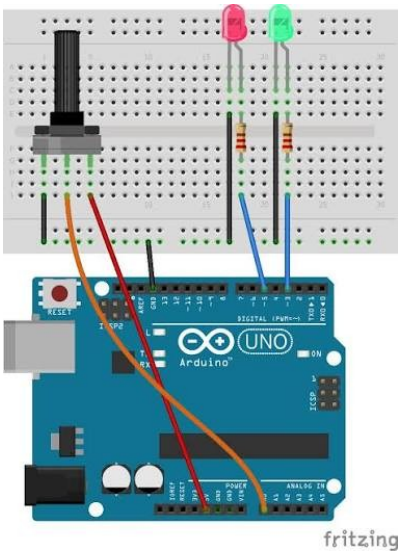
> *PWM = Pulse Width Modulation (modulació per amplada de pols). La sortida PWM no és analògica real: és una senyal digital que s'encén i s'apaga molt ràpid, de manera que la MITJANA sembla un voltatge intermedi.*

La UNO té 6 sortides PWM (els pins marcats amb ~). analogWrite accepta valors de 0 a 255 (precisió de 8 bits).

> *analogWrite(pin, valor) — valor entre 0 (0 V) i 255 (5 V)*

⚠ Compte: les entrades analògiques donen 0-1023 (10 bits) però la sortida PWM només admet 0-255 (8 bits). Per això cal dividir per 4 o usar map().

3. PROPOSTA 8 · Control de la brillantor d'un LED amb potenciòmetre



Cablejat muntatge de 2 LED i un potenciòmetre de control.

- 1. Connecta un LED al pin PWM 11 (amb resistència).
- 2. Connecta el potenciòmetre al pin analògic A0.
- 3. Llegeix A0 amb analogRead i divideix per 4 (1024/256).
- 4. Escriu el resultat al LED amb analogWrite.

El codi:

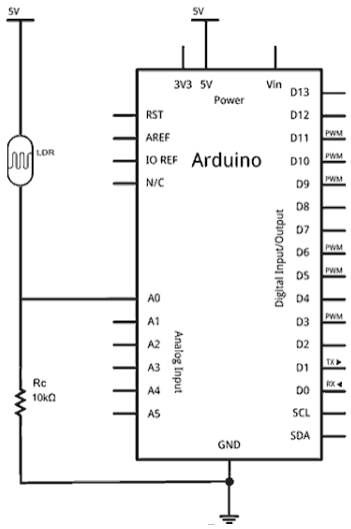
```
/* CONTROL LLUENTOR AMB POTENCIÒMETRE */
int pinLed = 11;
int valorSensor = 0;
```

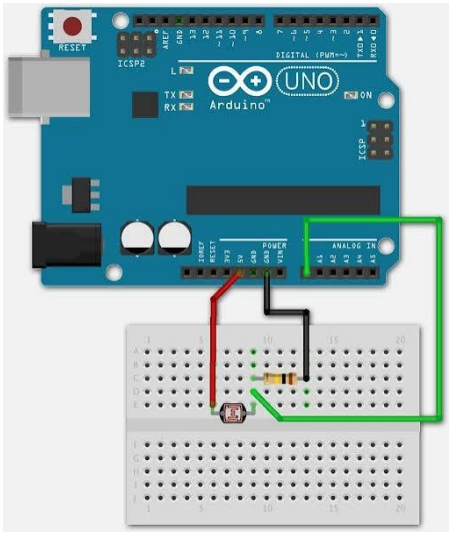
```
void setup()
{
pinMode(pinLed, OUTPUT);
}
void loop()
{
valorSensor = analogRead(A0);
analogWrite(pinLed, valorSensor / 4);
}
```

Puja'l i gira el potenciòmetre: la brillantor del LED canvia segons l'angle que giris.

Ara amb un LDR (Light Depending Resistor)

Una LDR és una resistència depenent de la llum, que fa que en funció de la lluminositat que capta pot posar posar la seva resistència a un valor tan baix que que simula un polsador tancat.





Cablejat de la LDR: divisor de tensió amb 10 kΩ, punt mig a A0.

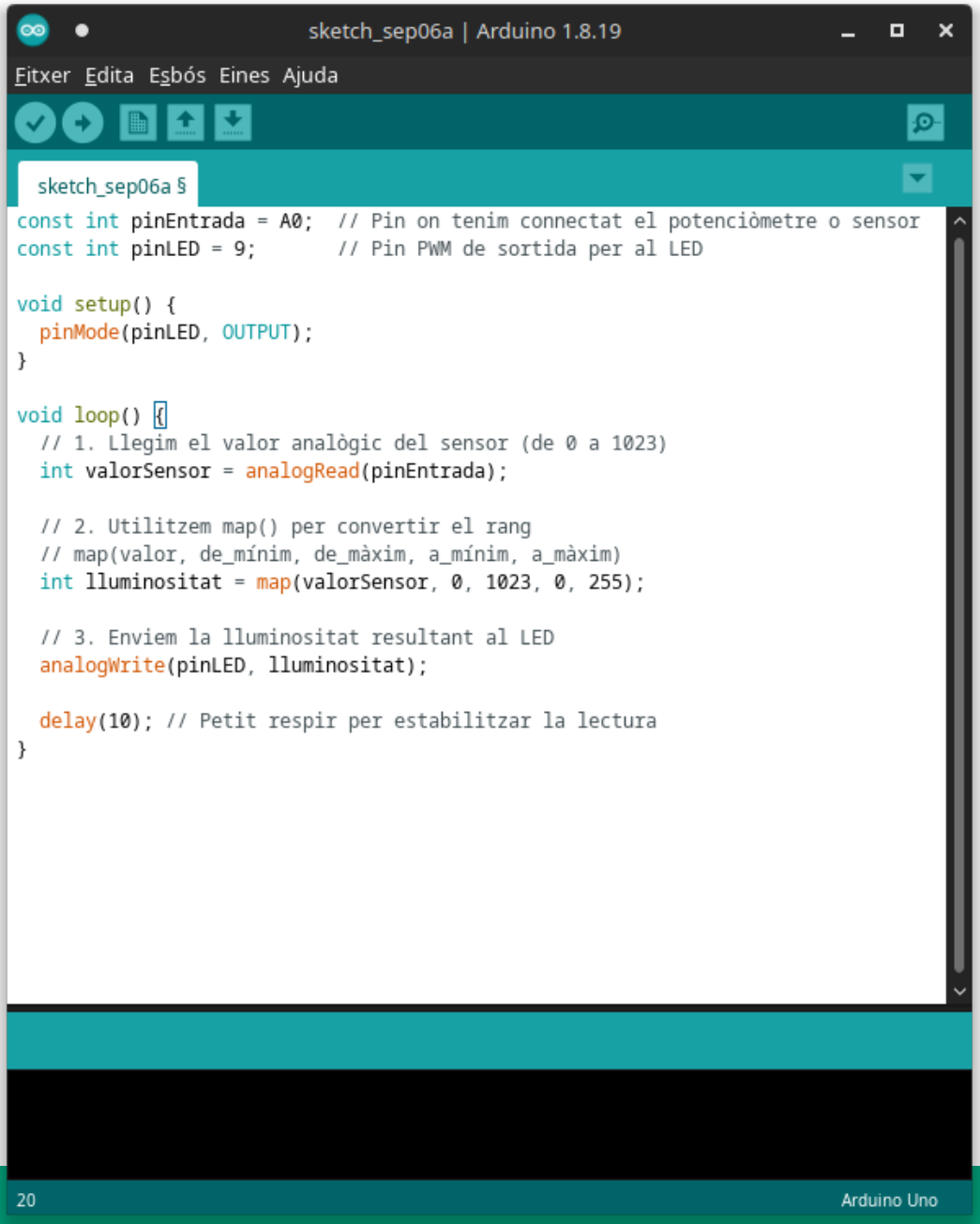
La funció map()

Una alternativa més elegant que dividir és map(), que ajusta uns valors d'entrada a uns de sortida de forma proporcional:

```
map(variable, val_in_min, val_in_max, val_out_min, val_out_max)
```

EXEMPLE: map(valorSensor, 0, 1023, 0, 255) — igual que dividir per 4 però més clar. Al bloc 6 la farem servir també per percentatges.

El codi que podeu provar:



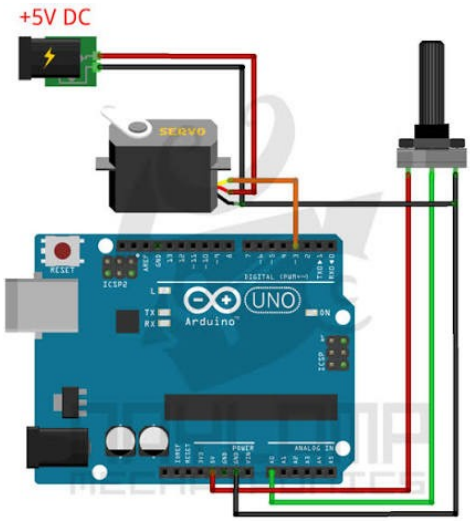
Activitat 8

Munta i puja el programa. Després:

- a) Inverteix la lògica: més gir = MENYS brillo.
- b) Canvia el LED per un RGB (prova un sol canal).
- c) EXTRA: fes que sobre un cert llindar (>800) el LED parpellegi.
- Pots fer que s’encengui un LED i a partir de cert valor se n’encengui un altre?

4. Servomotors

Un dels majors problemes de les plaques Arduino és que no poden proporcionar pels seus pins de sortida la suficient intensitat de corrent per alimentar directament certs elements com els motors. És per això que és recomanable no alimentar el SERVOS directament des d ela placa i impossible si en tenim diversos de connectat. Cap fer-ho amb una font externa.
Els SERVOMOTORS es controlen amb senyals PWM especials i la llibreria Servo.h.



Cablejat del servo: vermell a 5V, negre a GND, senyal (taronga/blanc) al pin 11.

Servo de rotació limitada

Aquest tipus de servo es posiciona en un ANGLE entre 0° i 180°. Té 3 cables: vermell (5 V), negre o marró (GND) i blanc/o taronja (senyal de control al pin PWM).

⚠ CONSIDERACIONS DE SEGURETAT:

- Els servos consumeixen MOLTA intensitat, sobretot si els forces. Si la placa es resseteja sovint, alimenta-la externament (piles o carregador).
- Connecta'ls sempre correctament: si confons alimentació i senyal, pots cremar el servo o la placa.
- Cada servo té un voltatge màxim (5 o 6 V generalment). Més tensió = servo cremat.

PROPOSTA 9 · Moviment del servo en intervals (salts)

Objectiu d'aquest codi: el servo es mou en salts de 10° des de 0° fins 170°, parant 1 segon a cada posició. Al final torna a l'inici i torna a començar.

```
#include <Servo.h>
```

```
Servo meuServo;
int angle = 0;

void setup()
{
meuServo.attach(11);
}

void loop()
{
for (int i = 0; i < 18; i++)
{
meuServo.write(angle);
angle = angle + 10;
delay(1000);
}
}
```

Elements nous:

- #include <Servo.h>: afegeix la llibreria del servo.
- Servo meuServo: crea l'objecte servo.
- meuServo.attach(11): el connecta al pin 11 (dins setup()).
- meuServo.write(angle): posiciona el servo a l'angle indicat (0-180).
- for (int i = 0; i < 18; i++): bucle que es repeteix 18 vegades (0 a 170 en salts de 10).

Activitat 9

Munta el servo (comprova vermell=5V, negre/marró=GND, blanc/taronja=pin 11) i puja el programa.

- a) Canvia els salts a 30° (quants bucles calen ara?).
- b) Fes que vagi més ràpid (delay 200 ms).
- c) EXTRA: fes un 'escombrat' suau: 0→180 ràpid i 180→0 lent (2 bucles for).

6. Sensor d'ultrasons: mesura de distància

El sensor HC-SR04 emet un pols d'ultrasons (pin TRIGGER) i escolta l'eco (pin ECHO). El temps entre pols i eco dona la distància.

> *distancia_cm = temps_echo / 58 (aproximació estàndard)*

Pràctica — Lectura en cm de la distància d'un objecte

Muntatge: HC-SR04 (VCC, GND, Trig, Echo) + lectura per Serial Monitor o a un LED, brunzidor, pantalla LCD...

Aquí tens un exemple de codi per a Arduino utilitzant el sensor d'ultrasons **HC-SR04**.

Aquest programa mesura la distància constantment i, si detecta un objecte a una distància igual o inferior a **10 cm**, encén un LED (o pot activar un brunzidor/zumbador) per avisar-te. Si està a més de 10 cm, s'apaga.

Connexions bàsiques:

- **VCC**: 5V de l'Arduino
- **GND**: GND de l'Arduino
- **Trig**: Pin digital 9
- **Echo**: Pin digital 8
- **LED**: Pin digital 13 (o pots utilitzar el LED integrat a la placa)


sketch_sep06a | Arduino 1.8.19
Fitxer Edita Esbós Eines Ajuda
sketch_sep06a §
// Definim els pins del sensor i del LED
const int trigPin = 9;
const int echoPin = 8;
const int ledPin = 13; // O pots posar un brunzidor (brunzidor)
void setup() {
// Inicialitzem el monitor sèrie per veure les mesures
Serial.begin(9600);
// Configurem els pins
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
pinMode(ledPin, OUTPUT);
}
void loop() {
// 1. Enviem un pols de 10 microsegons pel pin Trig
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// 2. Llegim el temps que triga el pols a tornar pel pin Echo (en microsegons)
long durada = pulseIn(echoPin, HIGH);
// 3. Calculem la distància en centímetres
// La velocitat del so és de 343 m/s (o 0.0343 cm/microsegon).
// Com que el so va i torna, dividim el resultat entre 2.
float distancia = durada * 0.0343 / 2;
// Mostrem la distància al monitor sèrie
Serial.print("Distancia: ");
Serial.print(distancia);
Serial.println(" cm");
// 4. Comprovem si està a 10 cm o menys
if (distancia > 0 && distancia <= 10.0) {
digitalWrite(ledPin, HIGH); // Encén l'avís (LED o brunzidor)
} else {
digitalWrite(ledPin, LOW); // Apaga l'avís
}
// Esperem un xic abans de la següent mesura
delay(100);
}
46
Arduino Uno

Modifica el programa per fer aquestes distàncies i compara amb un regle:

Distància real (regla)
10 cm
20 cm
50 cm

Quina precisió té? A quina distància falla?

PRODUCCIÓ U5 · Sistema de mesura amb sensor

Un sistema que llegeixi un sensor (LDR o ultrasó) i doni informació útil: avis lluminós de proximitat, engegada automàtica de llum, mesurador digital per Serial, alerta sonora si supera un llindar... o fins i tot un brunzidor que canviï de to en funció de la distància mesurada.
Et dono una pista pel que fa la brunzidor. La resta, és cosa teva:

Repàs i

autoavaluació

Rúbrica: 1 = No assolit · 2 = Satisfactori · 3 = Notable · 4 = Excel·lent

Objectiu
Llegeixo sensors analògics amb analogRead (0-1023)
Expliqu què és el PWM i uso analogWrite (0-255)
Reescalo valors amb map() o dividint per 4
Connecto un servo amb les precaucions de seguretat
Posiciono el servo a diferents angles amb Servo.h i bucles for

Glossari

- analogRead(A0): llegeix 0-1023 segons la tensió (0-5 V).
- PWM: modulació per amplada de pols; simula analògic.
- analogWrite(pin, 0-255): escriu valor PWM (pines amb ~).
- map(): reescala proporcionalment un rang a un altre.
- Potenciómetre: resistència variable (divisor de tensió).
- Servo rotació limitada: motor que posiciona un angle 0-180°.
- Servo.h: llibreria dels servomotors.
- attach / write: connectar el servo al pin / posicionar a l'angle.

ROBÒTICA · 3r ESO

UNITAT 6: COMUNICACIÓ SÈRIE + PROJECTE FINAL

Nom: _____ Curs/Grup: _____ Data: _____

Setmanes 10-11. Al final sabràs:

- Fer servir el monitor sèrie per comunicar PC ↔ placa.
- Enviar dades dels sensors al PC i llegir ordres del teclat.
- Comunicar dues plaques Arduino entre elles.

1. Comunicació entre dispositius electrònics

Els sistemes electrònics complexos solen estar formats per diversos elements que s'intercanvien dades: sensors, microcontroladors, actuadors, PC... La comunicació més senzilla entre Arduino i el PC (o entre dues plaques) és la COMUNICACIÓ SÈRIE ASÍNCRONA.

Tasa de transferencia (Baud Rate)

És la velocitat de transmissió en bits per segon (bps/baudis). El valor més comú és 9600. Cal que les dues parts estiguin a la MATEIXA velocitat.

Connexió i hardware

Quan connectes l'Arduino per USB al PC, el cable USB fa la conversió: els pines 0 (RX, rebre) i 1 (TX, transmetre) són els de la comunicació sèrie. ⚠ Si fas servir Serial al programa, NO connectis res als pins 0 i 1.

2. Usant el port sèrie a Arduino

> *Serial.begin(9600); — comença la comunicació a 9600 baudis (dins setup())*
>
> *Serial.println('text'); — envia text al monitor sèrie i salta de línia*

PROPOSTA 10 · Hola món sèrie

```
void setup()
{
Serial.begin(9600);
}
void loop()
{
Serial.println('Hola món des d'Arduino!');
delay(1000);
}
```

Puja'l i obre el Monitor Sèrie (icona de la lupa): veuràs el missatge cada segon.

PROPOSTA 11 · Llegir dades del PC

Ara al revés: el PC envia dades a l'Arduino.

```
void setup()
{
Serial.begin(9600);
}
void loop()
{
if (Serial.available() > 0) // hi ha dades al buffer?
{
int dada = Serial.read(); // llegeix caràcter a caràcter
Serial.write(dada); // els torna al monitor
}
}
```

Envia 'ARDUINO' des del monitor: la funció write() envia els valors rebuts amb el seu caràcter corresponent de la taula ASCII. Si uses char en lloc de int, el mateix amb println().

PROPOSTA 12 · Potenciómetre al monitor sèrie

Mostrem el valor d'un sensor analògic al monitor:

```
// Transmissió serie del valor d'un potenciòmetre
const int POT = A0;
void setup()
{
Serial.begin(9600);
}
void loop()
{
int val = analogRead(POT);
int porcentaje = map(val, 0, 1023, 0, 100);
```

```
Serial.print('Lectura analògica: ');
Serial.print(val);
Serial.print(' Percentatge: ');
Serial.print(Percentatge);
Serial.println('%');
delay(1000);
}
```

Observa la funció map() que ja coneixes: ajusta 0-1023 a 0-100 (percentatge).

PROPOSTA 13 · Encendre un LED des del PC

```
// Control d'un LED des del monitor serie
const int LED = 11;
char dato;
void setup()
{
Serial.begin(9600);
pinMode(LED, OUTPUT);
}
void loop()
{
if (Serial.available() > 0)
{
dato = Serial.read();
if (dada == 'R') // si rep R majúscula
{
digitalWrite(LED, HIGH);
Serial.println('LED VERMELL ON');
}
else if (dada == 'r') // si rep r minúscula
{
digitalWrite(LED, LOW);
Serial.println('LED VERMELL OFF');
}
}
}
```

Envia R → s'encén. Envía r → s'apaga. Aquest programa, modificat fàcilment, permet controlar qualsevol dispositiu des del teclat!

Activitat 10

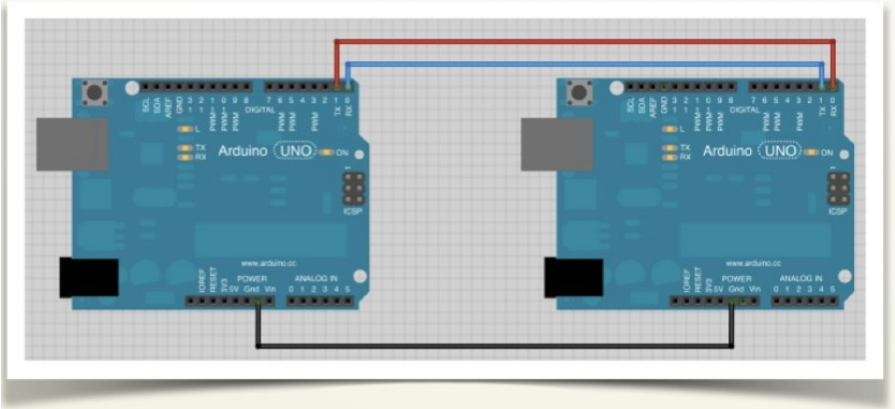
Fes les 4 propostes. Registre:

Proposta

P10 Hola món
P11 Echo
P12 Pot al monitor
P13 LED des del PC

3. Comunicant dues Arduino

Es pot establir comunicació sèrie entre dues plaques: una EMISSORA i una RECEPTORA. Fem servir la llibreria SoftwareSerial (pines digitals qualssevol).



Cablejat creuat: $TX_1 \rightarrow RX_2$, $RX_1 \rightarrow TX_2$, GND comú. Falta connectar 5V comú.

- Uneix les dues terres (GND-GND) i alimenta la receptora des de l'emissora (5V-5V).
- Connecta el pin TX de l'emissora al RX de la receptora, i viceversa (creuat).

Programa de l'EMISSORA (rep ordres del monitor i les reenvia):

```
// Arduino UNO Emissora
#include <SoftwareSerial.h>
SoftwareSerial emisor(8, 9); // Rx, Tx
void setup()
{
Serial.begin(9600);
emisor.begin(9600);
}
void loop()
{
if (Serial.available() > 0)
{
int dada = Serial.read();
if (dada == 'R')
```

```
{
emisor.write('R');
Serial.println('LED VERMELL ON ENVIADA ORDRE');
}
else if (dato == 'r')
{
emisor.write('r');
Serial.println('LED VERMELL OFF ENVIADA ORDRE');
}
}
delay(1000);
}
```

Programa de la RECEPTORA (rep ordres i encén/apaga el LED):

```
// Arduino UNO Receptora
#include <SoftwareSerial.h>
int pinLed = 13;
int lectura = 0;
SoftwareSerial receptor(8, 9);
void setup()
{
pinMode(pinLed, OUTPUT);
receptor.begin(9600);
}
void loop()
{
if (receptor.available() > 0)
{
lectura = receptor.read();
if (lectura == 'R')
{
digitalWrite(pinLed, HIGH);
}
else if (lectura == 'r')
{
digitalWrite(pinLed, LOW);
}
}
delay(1000);
}
```

Puja cada programa a la seva placa (has de canviar de port quan canviïs de placa!). Des del monitor de l'emissora: envia R i el LED de la receptora s'encén. Has construït un sistema de telecomanda amb fil! També es pot fer per WiFi per Bluetooth... si hi arribem, ja ho farem!

Activitat 11

Munta la comunicació entre dues plaques. Prova:

- a) Afegeix més instruccions (G/g per un LED verd a la receptora).
- b) EXTRA: que la receptora respongui confirmació a l'emissora.

4. PROJECTE FINAL ARDUINO (opcional)

Disseny d'un Sistema de Control d'Accés amb Arduino

4.1. Descripció del Projecte

L'objectiu d'aquest projecte és construir un pany electrònic intel·ligent. L'alumnat aprendrà a integrar diferents components d'entrada i sortida controlats per una placa Arduino per crear un sistema de seguretat interactiu.

Quan un usuari introdueixi una contrasenya de 4 xifres mitjançant un teclat matricial, el sistema l'avaluarà:

- Si la contrasenya és **correcta**, la pantalla LCD mostrarà un missatge de benvinguda i s'obrirà el pany (simulat amb un missatge i un senyal sonor).
- Si la contrasenya és **errònia**, mostrarà un avís d'error a la pantalla i emetrà un so d'alerta.
- Durant la introducció de les dades, la pantalla mostrarà asteriscs * * * * per mantenir la privacitat del codi.

4.2. Objectius d'Aprenentatge

- **Electrònica:** Connectar correctament un teclat matricial (4x4), una pantalla LCD amb interfície I2C i un brunzidor a una placa Arduino.
- **Programació:**
  - Utilitzar llibreries externes específiques (Keypad i LiquidCrystal_I2C).
  - Gestionar variables de tipus text (String) i la comparació de cadenes.
  - Implementar estructures condicionals (if / else) i bucles per llegir entrades de l'usuari.
- **Disseny i UX:** Oferir retroalimentació visual i sonora adequada per a l'usuari final.

4.3. Materials Necessaris

Per dur a terme el muntatge pràctic, cada grup de treball disposarà de:

- 1 placa Arduino (o compatible) amb cable USB.
- 1 protoboard (placa de proves) i cables de connexió (mascle-mascle / mascle-femella).

- 1 teclat matricial de 4x4.
- 1 pantalla LCD 16x2 amb mòdul adaptador I2C (4 pins).
- 1 brunzidor (*brunzidor*) piezoelèctric.

Components necessaris:

- 1 placa Arduino (com l'Arduino Uno)
- 1 teclat matricial de 4x4 (*Keypad*)
- 1 pantalla LCD 16x2 amb mòdul I2C (per connectar-la fàcilment amb només 4 cables)
- 1 brunzidor (*brunzidor*) per donar feedback acústic (opcional però molt recomanable)
- Cables de connexió i protoboard

Instal·lació de llibreries necessàries

Abans de carregar el codi, assegura't de tenir instal·lades aquestes dues llibreries a l'Arduino IDE (les pots buscar al *Gestor de llibreries*):

1. **Keypad** (per gestionar el teclat matricial)
2. **LiquidCrystal_I2C** (per controlar la pantalla LCD)

4.4. Esquema de Connexions (Resum tècnic)

- **Pantalla LCD (I2C):**
  - GND $\rightarrow$ GND d'Arduino
  - VCC $\rightarrow$ 5V d'Arduino
  - SDA $\rightarrow$ Pin analògic A4 (en Arduino Uno)
  - SCL $\rightarrow$ Pin analògic A5 (en Arduino Uno)
- **Teclat Matricial (4x4):** Connexió de les 8 línies als pins digitals del 2 al 9 de l'Arduino.
- **Brunzidor:** Connectat al pin digital 10 i a GND.

4.5. El Codi del Projecte (Exemple Base)

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Keypad.h>

// Inicialitzem la pantalla LCD (Adreça I2C 0x27 o 0x3F, 16 columnes i 2 files)
```

```
LiquidCrystal_I2C lcd(0x27, 16, 2);

// Configuració del teclat matricial 4x4
const byte FILES = 4;     // 4 files
const byte COLUMNES = 4;  // 4 columnes

// Definim els caràcters de les tecles del teclat
char tecles[FILES][COLUMNES] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};

// Pins d'Arduino on es connecten les files i columnes del teclat
byte pinsFiles[FILES] = {9, 8, 7, 6};       // Connecta a les files 1 a 4
byte pinsColumnes[COLUMNES] = {5, 4, 3, 2}; // Connecta a les columnes 1 a 4

Keypad teclat = Keypad(makeKeymap(tecles), pinsFiles, pinsColumnes, FILES,
COLUMNES);

// Variables del sistema de contrasenya
String contrasenyaSecret = "1234"; // Pots canviar-la per la que vulguis
String inputUsuari = "";          // Emmagatzema el que l'alumne va teclejant
const int pinBrunzidor = 10;      // Pin opcional per a un brunzidor

void setup() {
  pinMode(pinBrunzidor, OUTPUT);

  // Inicialitzem la pantalla LCD
  lcd.init();
  lcd.backlight(); // Encenem la llum de fons de la LCD
```

```
  missatgeInici();
}

void loop() {
  char tecla, a;
  tecla = teclat.getKey(); // Llegim si s'ha premut alguna tecla

  if (tecla) {
    // Si prem '*', podem esborrar l'intent actual
    if (tecla == '*') {
      inputUsuari = "";
      missatgeInici();
      return;
    }

    // Si prem '#', comprovem directament el codi introduït
    if (tecla == '#') {
      verificarContrasenya();
      return;
    }

    // Si és un nombre (de 0 a 9) i portem menys de 4 xifres
    if (tecla >= '0' && tecla <= '9') {
      if (inputUsuari.length() < 4) {
        inputUsuari += tecla; // Afegim el dígit

        // Dibuixem els asteriscs corresponents al nombre de xifres entrades
        lcd.setCursor(0, 1);
        for (int i = 0; i < inputUsuari.length(); i++) {
          lcd.print("*");
        }
```

```
      // Si ja ha arribat a les 4 xifres, comprovem automàticament
      if (inputUsuari.length() == 4) {
        delay(300); // Petit respir per veure el darrer asterisc
        verificarContrasenya();
      }
    }
  }
}

void missatgeInici() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Introdueix codi:");
  lcd.setCursor(0, 1);
  lcd.print("    "); // Espai buit per començar a mostrar asteriscs
}

void verificarContrasenya() {
  lcd.clear();
  lcd.setCursor(0, 0);

  if (inputUsuari == contrasenyaSecret) {
    // PASSORD CORRECTE
    lcd.print("Porta");
    lcd.setCursor(0, 1);
    lcd.print("desbloquejada");

    // So agut d'èxit amb el brunzidor
    tone(pinBrunzidor, 1000, 200);
    delay(250);
```

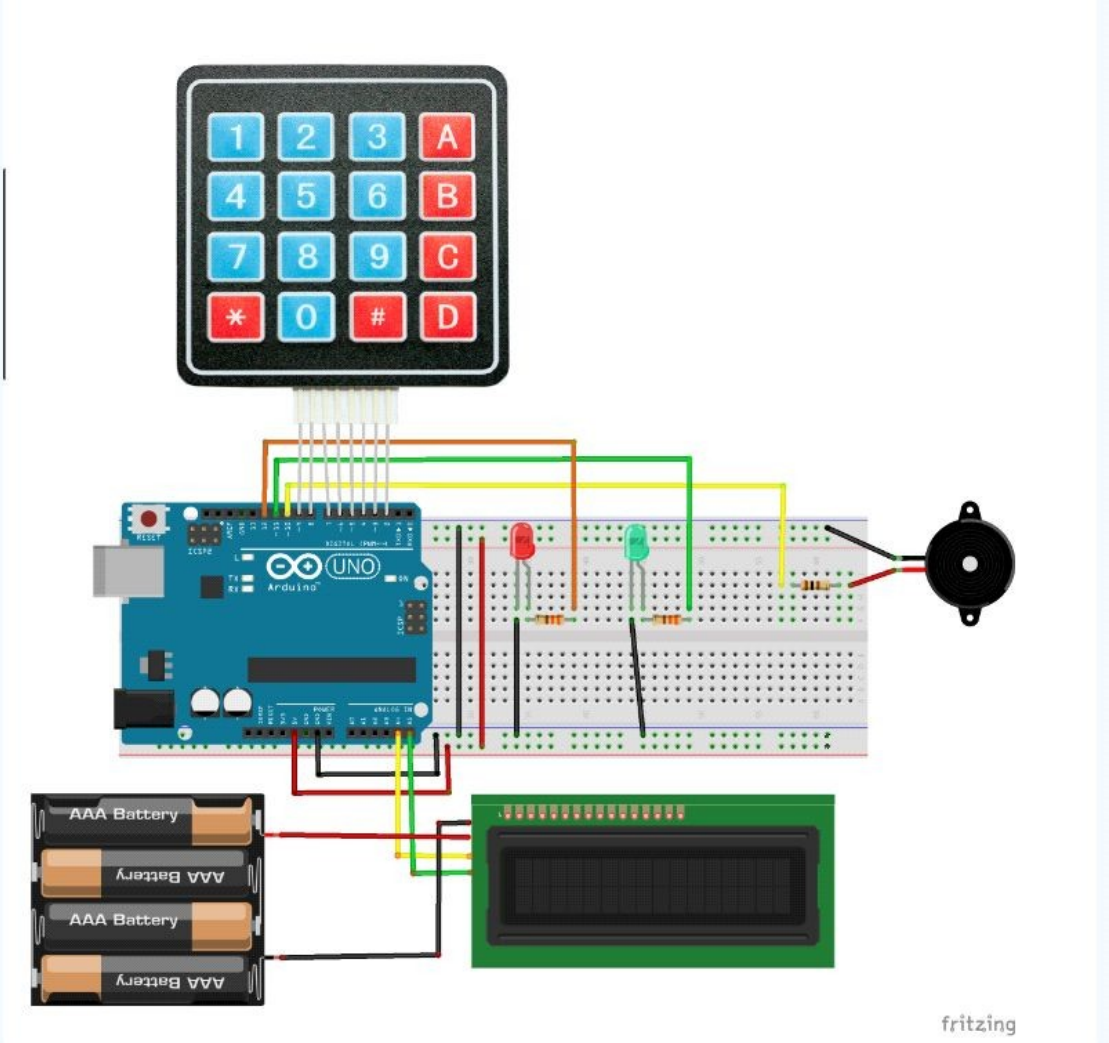
```
    tone(pinBrunzidor, 1500, 300);

    delay(3000); // Es manté el missatge 3 segons
  } else {
    // CONTRASENYA ERRÒNIA
    lcd.print("Codi erroni");

    // So greu d'error amb el brunzidor
    tone(pinBrunzidor, 300, 500);

    delay(2000); // Es mostra l'error 2 segons
  }

  // Reiniciem la variable per a un nou intent
  inputUsuari = "";
  missatgeInici();
}
```



Cablejat keypad matricial, pantall i LCD. Brunzidor i LEDs indicadors, opcionals.

4.6. Fases de Treball i Activitats Avaluables

El projecte es dividirà en les següents fases d'execució:

1. **Fase de Muntatge (Hardware):** Connectar tots els components a la protoboard de manera ordenada i segura sense connectar l'alimentació principal fins a la revisió del docent.
2. **Fase de Programació (Software):** Instal·lar les llibreries necessàries a l'IDE d'Arduino i carregar el codi base.
3. **Fase de Proves i Modificació:**
    - Comprovar que els asteriscs apareixen correctament a mesura que es prem el teclat.
    - Modificar el codi secret de l'Arduino per personalitzar la contrasenya del grup.
    - Afegir millores opcionals (per exemple, un comptador de 3 intents màxims abans de bloquejar el sistema durant 10 segons).

4. **Fase de Memòria de Projecte:** Lliurament d'un informe breu explicant les dificultats trobades i com s'han resolt.

Repàs i autoavaluació

Rúbrica: 1 = No assolit · 2 = Satisfactori · 3 = Notable · 4 = Excel·lent

Objectiu
Fa servir Serial.begin i el monitor sèrie per comunicar amb el PC
Envio dades de sensors al PC (analogRead + map + println)
Rebo ordres del teclat (Serial.read) i contro actuadors
Comunico dues plaques Arduino amb SoftwareSerial
He acabat el projecte final i l'he defensat

Glossari

- Comunicació sèrie asíncrona: enviament bit a bit sense rellotge compartit.
- Baud rate: velocitat en bits/segon (9600 el més comú).
- RX / TX: pin de rebre / transmetre (0 i 1 a la UNO).
- Serial.begin(9600): inicia la comunicació.

- Serial.println / print: envia text amb / sense salt de línia.
- Serial.available() > 0: comprova si hi ha dades al buffer.
- Serial.read / write: llegir / enviar un caràcter (ASCII).
- SoftwareSerial: llibreria per fer sèrie amb pines digitals qualssevol.
- ASCII: taula que assigna un número a cada caràcter.

ROBÒTICA · 3r ESO

FASE 2: MÀQUINES MAKER — DISSENY 2D/3D I FABRICACIÓ

Nom: _____ Curs/Grup: _____ Data: _____

Setmanes 7-9. Al final sabràs:

- Dissenyar peces en 3D i imprimir-les amb la Bambu Lab.
- Dissenyar en 2D i tallar/gravar amb la Beambox (làser).
- Tallar vinil i paper amb la Cricut (plotter).
- Fabricar les peces del teu projecte integral (caixa, carcassa, mecanisme).

1. Les màquines Maker del taller

Màquina
Bambu Lab (impressora 3D)
Beambox (talladora/gravadora làser)
Cricut (plotter de tall)

> *L'eix del curs: programar (Fase 1) + fabricar (Fase 2) = PROJECTE INTEGRAL (Fase 3). Les peces que facis aquí són les del teu projecte final.*

2. Disseny 3D i impressió (Bambu Lab)

Del concepte a la peça

- 1. IDEA: què ha de fer la peça? (suport, caixa, engranatge, palanca...)
- 2. DISSENY: modela-la a TinkerCAD (senzill) o Fusion 360 (avançat).
- 3. EXPORTA: descarrega el fitxer .STL.
- 4. LAMINA: obre'l al Bambu Studio → orienta la peça, afegeix suports si cal, tria infill (10-20% sol ser suficient) → genera el G-code.

- 5. IMPRIMEIX: envia a la Bambu Lab i vigila la primera capa (la més crítica).
- 6. RETOCA: treu suports, llixo si cal, prova si encaixa.

Regles d'or del disseny per imprimir

- Tolerància: deixa 0,2-0,4 mm de joc si la peça ha d'encaixar amb una altra.
- Cara plana al plat: les peces s'imprimeixen millor si tenen una cara plana a baix.
- Voladissos: per sobre de 45° necessiten suports (més material, més temps).
- Paret mínima: 1-2 mm perquè sigui resistent però no lenta d'imprimir.

Repte 7 — 'Dissenya una peça'

Dissenya i imprimeix una peça útil per als teus circuits: suport d'Arduino per a paret, caixeta per a una placa, palanca per a un servo...

Pas
Disseny fet (TinkerCAD/Fusion)
.STL exportat
Laminat (Bambu Studio): infill, suports
Imprès i retocat

Foto de la peça impresa + temps d'impressió + material utilitzat:

3. Disseny 2D i tall làser (Beambox)

La Beambox talla i grava amb làser. El disseny es fa en 2D (Inkscape): línies vermelles = tall, línies negres = gravat (segons configuració del programari).

Seguretat (imprescindible)

- MAI obrir la tapa amb el làser actiu.
- MAI deixar la màquina sola mentre treballa.
- Només materials permesos: fusta fina, acrílic, cartró. MAI PVC (allibera gasos tòxics) ni materials reflectants.

- Ventilació engegada abans de començar.

Repte 8 — 'Talla i grava'

Dissenya en Inkscape i fabrica amb la Beambox: placa identificativa gravada, panell frontal per al teu semàfor, plantilla...

Pas
Disseny 2D (Inkscape)
Configuració làser (potència/velocitat)
Tall/gravat fet amb seguretat

4. Plotter de tall (Cricut)

La Cricut talla vinil, paper i cartolina amb molta precisió (blades giratòries). Ideal per: adhesius, etiquetes, plantilles, decoració de la caixa del projecte.

Repte 8b — 'Cricut'

Talla un adhesiu o plantilla per al teu projecte (logotip, etiqueta de la caixa, marc dels LEDs...).

5. Repte 9 — 'Fabrica la caixa del teu projecte'

Ara combina les eines: dissenya i fabrica la carcassa del teu projecte integral (Fase 3), pensant ON han d'anar els components:

- Forats per als LEDs (mida exacta del diàmetre).
- Finestra per al display o per veure els LEDs.
- Forat per al cable USB / alimentació.
- Suports interns per a la placa (les aletes per a cargols o encaixos).
- Si és làser: caixa amb juntes de encaix (box joints) dissenyades amb generadors d'Inkscape (extensions).

> *CONECTA AMB ARDUINO: mesura les peces reals (LED de 5 mm = forat de 5,2 mm; placa UNO = 68,6 × 53,4 mm) i dissenya amb aquestes mesures. La fabricació s'ajusta a l'electrònica, no al revés.*

Component del projecte
Arduino UNO
LEDs

Polsador/potenciòmetre
Servo (si escau)

Esquema de la caixa (vista frontal + superior):

Repàs i autoavaluació

Rúbrica: 1 = No assolit · 2 = Satisfactori · 3 = Notable · 4 = Excel·lent

Objectiu
Disseny en 3D i imprimeixo peces amb la Bambu Lab (STL → laminació → impressió)
Apliqu les regles d'or del disseny (tolerància, voladissos, parets)
Disseny en 2D i faig talls/gravats amb la Beambox amb seguretat
Faig talls amb la Cricut (vinil/paper)
Fabrico les peces del projecte integrant 3D i tall 2D
Mesuro els components reals i adapto el disseny a l'electrònica

Glossari

- STL: fitxer de disseny 3D per a la impressora.
- G-code: instruccions que executa la impressora (generat pel laminador).
- Infill: farcel·l interior de la peça (10-20% típic).
- Suports: estructura temporal per a voladissos > 45°.
- Tolerància: joc deixat perquè dues peces encaixin (0,2-0,4 mm).
- SVG: fitxer vectorial 2D per a làser i plotter.
- Gravat vs tall: el làser marca la superfície / travessa el material.
- Box joints: juntes d'encaix per a caixes de làser.

ROBÒTICA · 3r ESO

FASE 3: PROJECTE INTEGRAL — PROGRAMACIÓ + FABRICACIÓ

Nom: _____ Grup: _____ Data: _____

> *El projecte final combina les dues meitats del curs: PROGRAMACIÓ (Arduino amb sensors i actuadors) + FABRICACIÓ DIGITAL (peces fetes amb màquines maker). Es treballa en grups de 2-3 i es defensa oralment.*

1. Tria el repte (de més senzill a més ambiciós)

Opció
Nivell A — Semàfor en caixa, per exemple
Nivell B — Mecanisme amb servo, per exemple
Nivell C (avançat) — Canvi de marxes automàtic, per exemple

2. Planificació (abans de fabricar res)

Camp
Tipus de Repte triat (A/B/C)
Què ha de fer (funció)
Components electrònics (sensors + actuadors)
Peces a fabricar (i amb quina màquina)
Repartiment de tasques del grup
Pla de sessions (què fem cada dia)

Aprovació del professor/a abans de començar: _____

3. Desenvolupament (setmanes 10-11)

Registre de procés — apunta-ho TOT (els errors són la part més valuosa):

Sessió
1
2
3
4

Comprovació teoria ↔ pràctica

Al dossier hi ha d'haver la frase clau del curs:

> *«Aquest procés s'ha pogut portar a terme segons [concepte programat: delays, condicionals, map(), Servo...], i s'han obtingut els resultats [X], que coincideixen / no coincideixen amb el que esperàvem perquè [raó].»*

EXEMPLE real: 'La barrera s'aixeca quan l'ultrasons detecta menys de 20 cm, segons el condicional if programat. El servo triga 1,2 s en arribar a 90° (coincideix amb el delay programat). El primer braç impreso no encaixava al servo perquè no vam deixar tolerància de 0,3 mm; l'hem tornat a imprimir amb el forat ampliat.'

4. Requisits mínims del projecte

Requisit
Circuit Arduino funcional amb almenys 1 sensor i 1 actuador
Almenys 1 peça fabricada amb màquines maker (3D, làser o Cricut)
Les peces encaixen amb l'electrònica (forats, toleràncies)

Codi comentat i desat
Dossier de procés (fotos de cada sessió, errors, solucions)
Frase teoria ↔ pràctica a les conclusions
DEFENSA ORAL (3 min): què fa, com ho hem fet, problemes i solucions

5. Defensa oral (3 minuts per grup)

El professor preguntarà sobre el PROCÉS, no només el producte:

- Què fa el vostre projecte i quin repte resol?
- Com funciona la part programada? (conceptes: condicionals, delays, map, servo...)
- Per què heu fabricat les peces així? Quines mesures vau fer?
- Quin error us ha costat més i com l'heu resolt?
- Què millorariu amb més temps?

Espai per a l'esquema del circuit i el codi resum:

Repàs i autoavaluació

Rúbrica: 1 = No assolit · 2 = Satisfactori · 3 = Notable · 4 = Excel·lent

Objectiu
El nostre circuit funciona amb sensor + actuador
Hem fabricat les peces amb màquines maker i encaixen
Hem documentat tot el procés (fotos, errors, solucions)
Hem escrit la frase teoria ↔ pràctica amb el resultat real

Hem defensat el projecte oralment amb domini

Glossari

- Projecte integral: combina programació Arduino + fabricació digital.
- Repte: problema real a resoldre de manera col·laborativa.
- Pensament computacional: descomposar, reconèixer patrons, abstraure, algorismar.
- Iteració: provar → fallar → corregir → tornar a provar.
- Dossier de procés: evidència del camí: fotos, errors, decisions.
- Defensa oral: explicar i justificar les decisions tècniques.