1ère BAC Pro CIEL		
	Lire et comprendre un programme Arduino	 Année 2026/2027

Nom		
Prénom		
Date		
Matériel Outillage 	⇨ N/A	**Durée** : 3H 
Travaux à réaliser 	⇨ Analyse de code Arduino	

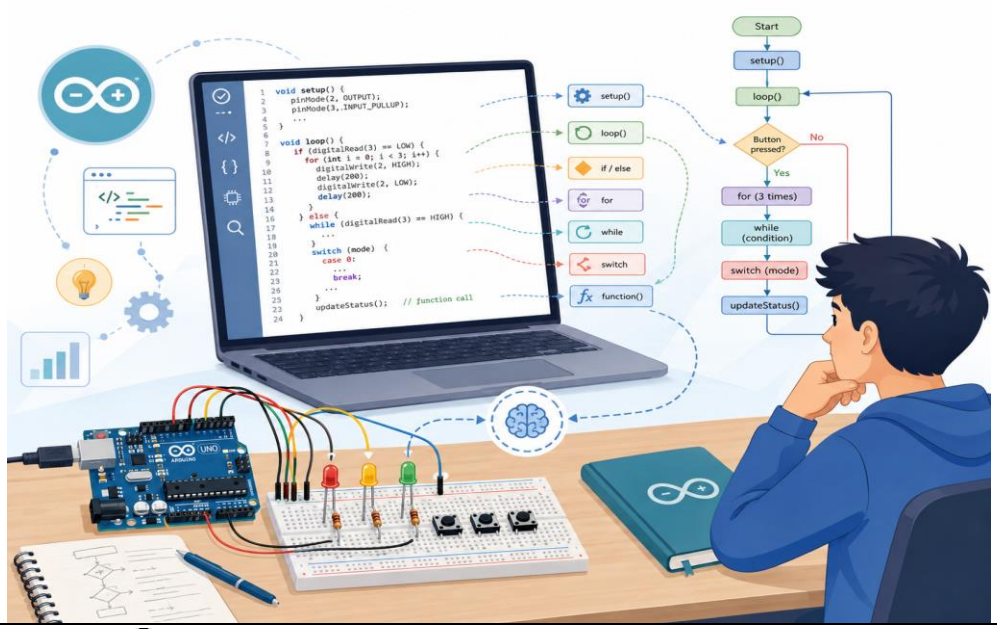
Pôle d'activité : Réalisation et maintenance de produits électroniques

Activités :

⇨ **E4** : Intégration matérielle et logicielle

Compétences :

⇨ **C04** : Analyser une structure matérielle et logicielle



Lorsque le logo  apparaît, il est indispensable d'appeler l'enseignant pour vérification.

A. Compétences

C01	COMMUNIQUER EN SITUATION PROFESSIONNELLE (ANGLAIS/FRANÇAIS)	
	La présentation (typographie, orthographe, illustration, lisibilité) est soignée et soutient le discours avec des enchaînements cohérents	
	La présentation orale (support et expression) est de qualité et claire	
	L'argumentation développée lors de la présentation et de l'échange est de qualité	
	L'argumentation tient compte des éventuelles situations de handicap des personnes avec lesquelles il interagit	
C03	PARTICIPER A UN PROJET	
	Les rôles et tâches de chacun sont identifiés ; le cas échéant, les besoins spécifiques des personnes en situation de handicap sont pris en compte	
	Le planning prévisionnel est compris	
	Le suivi du projet est respecté	
	L'espace collaboratif est correctement utilisé	
C04	ANALYSER UNE STRUCTURE MATÉRIELLE ET LOGICIELLE	
	Le besoin est identifié ainsi que les ressources matérielles, logicielles et humaines	
	Les logiciels d'analyse et de tests sont utilisés selon les procédures de traitement d'incidents	
	Les informations nécessaires sont extraites des documents réglementaires et/ou constructeurs	X
	Les indicateurs de fonctionnement sont interprétés	X
	Les fiches de test ou d'intervention sont renseignées	
C06	VALIDER LA CONFORMITÉ D'UNE INSTALLATION	
	Les exigences du cahier des charges sont respectées	
	Les tests sont effectués	
	Les résultats attendus sont vérifiés	
	La procédure de test est respectée	
C07	RÉALISER DES MAQUETTES ET PROTOTYPES	
	Le placement et routage sont conformes au cahier des charges	
	La génération des fichiers de fabrication du PCB est conforme aux attentes	
	Le PCB est réalisé, contrôlé et conforme aux IPC (tolérances mécaniques, finition de surface, propreté, ESD etc.)	
	Les composants sont conformes à la nomenclature (marquage, étiquetage)	
	La nomenclature des composants est respectée	
	Le brasage de la carte est conforme à la nomenclature et aux IPC	
	Les contraintes liées aux impacts environnementaux sont intégrées	
	Le contrôle visuel de la carte assemblée est conforme au dossier de fabrication	
	Les risques d'une situation de travail sont repérés et les mesures appropriées pour sa santé, sa sécurité et celle des autres sont adoptées	
C08	CODER	
	Les environnements de développement et de test sont mis en oeuvre en tenant compte des contraintes de fonctionnalités et de sécurité	
	Le module logiciel est débogué et syntaxiquement correct	
	Les composants logiciels individuels sont développés et testés conformément aux spécifications du cahier des charges et des bonnes pratiques	
	La solution (logicielle et matérielle) est intégrée et testée conformément aux spécifications du cahier des charges et des bonnes pratiques	
	Le code est commenté et le logiciel est documenté	
C09	INSTALLER LES ÉLÉMENTS D'UN SYSTÈME ÉLECTRONIQUE OU INFORMATIQUE	
	L'ensemble des éléments pour l'installation du système est complet et vérifié par rapport au cahier des charges	
	Les éléments du système sont installés et raccordés selon une procédure	
	La configuration est réalisée	
	La mise en service est réalisée	
	L'état de l'installation est renseigné de manière écrite ou orale	
	Les risques d'une situation de travail sont repérés et les mesures appropriées pour sa santé, sa sécurité et celle des autres sont adoptées	
C10	EXPLOITER UN RÉSEAU INFORMATIQUE	
	Les alertes et problèmes rencontrés sont renseignés	
	Les différents éléments d'un réseau ou d'un système à partir d'un schéma fourni sont identifiés	
	La mise à jour des équipements (iOS, OS, logiciel, firmware) est effectuée	
	Les optimisations nécessaires sont effectuées	
C11	MAINTENIR UN SYSTÈME ÉLECTRONIQUE OU RÉSEAU INFORMATIQUE	
	L'intervention est préparée	
	Le dysfonctionnement est constaté	
	La maintenance ou la réparation est réalisée	
	La fiche d'intervention est correctement renseignée	
	Les risques d'une situation de travail sont repérés et les mesures appropriées pour sa santé, sa sécurité et celle des autres sont adoptées	

Nature de complexité de l'activité

Découverte	X
Intermédiaire	
Bac Pro	

B. Mise en contexte

Une entreprise doit intégrer une carte Arduino Uno dans un petit équipement électronique destiné à piloter plusieurs voyants et organes de commande.

Le programme embarqué a déjà été développé par un autre technicien. Avant de procéder au raccordement, au paramétrage et à la mise en service de l'équipement, le technicien chargé de l'intégration doit vérifier qu'il comprend correctement le fonctionnement du logiciel fourni.

Il doit notamment identifier les variables utilisées, repérer les différentes fonctions du programme, comprendre les conditions et les boucles, puis déterminer les actions réalisées sur les entrées et les sorties de la carte Arduino.

Cette analyse préalable doit permettre d'éviter une erreur d'intégration et de vérifier que le comportement prévu du système est cohérent avec son fonctionnement attendu.

C. Problématique

Comment analyser un programme Arduino existant afin de comprendre son fonctionnement et de préparer correctement son intégration dans un système électronique ?

D. Partie 1 – Structure d'un programme Arduino

Soit le programme Arduino suivant :

```
const byte LED_VERTE = 8;
int duree = 500;
bool etatLed = false;

void setup()
{
    pinMode(LED_VERTE, OUTPUT);
}

void loop()
{
    digitalWrite(LED_VERTE, HIGH);
    delay(duree);

    digitalWrite(LED_VERTE, LOW);
    delay(duree);
}
```

Repérage de la structure

Quelle est le nom de la constante utilisée pour désigner la broche de la LED ?

Quelle est la valeur de cette constante ?

Combien de fois la fonction setup() est-elle exécutée après la mise sous tension ?

Que devient la fonction loop() après l'exécution de la dernière instruction ?

Quelle variable est déclarée mais n'est pas utilisée dans ce programme ?

Tableau de données

Nom	Type	Valeur initiale	Constante ou variable ?	Rôle dans le programme
LED_VERTE				
duree				
etatLed				

Prévision du fonctionnement

Décrire le comportement de la LED après la mise sous tensions.

Quelle est la durée d'un cycle complet allumé + éteint ?

E. Partie 2 – Conditions if, else if et else

Condition if...else avec un bouton

```
const byte BP_MARCHE = 2;
const byte LED_VERTE = 8;

void setup()
{
    pinMode(BP_MARCHE, INPUT_PULLUP);
    pinMode(LED_VERTE, OUTPUT);
}

void loop()
{
    if (digitalRead(BP_MARCHE) == LOW)
    {
        digitalWrite(LED_VERTE, HIGH);
    }
    else
    {
        digitalWrite(LED_VERTE, LOW);
    }
}
```

Etat physique du bouton	Valeur lue	Condition vraie ?	Etat de la LED
Bouton relâché			
Bouton appuyé			

Pourquoi la condition recherche-t-elle la valeur LOW lorsque le bouton est appuyé ?

Choix entre plusieurs situations

```
int temperature = 27;

if (temperature < 15)
{
    etat = 0;
}
else if (temperature < 25)
{
    etat = 1;
}
else
{
    etat = 2;
}
```

Température	Premier test	Deuxième test	Valeur finale de etat
10°c			
15°c			
20°c			
25°c			
27°c			
35°c			

Pour une température de 10°c, le test température < 25 **est**-il exécuté ? **Justifier**.

Quelle différence faites-vous entre l'opérateur = et l'opérateur == ?

F. Partie 3 – Boucles while et for

Boucle while : répéter tant que la condition est vraie

```
while (digitalRead(BP_MARCHE) == HIGH)
{
    digitalWrite(LED_ROUGE, HIGH);
    digitalWrite(LED_VERTE, LOW);
}

digitalWrite(LED_ROUGE, LOW);
digitalWrite(LED_VERTE, HIGH);
```

Quelle condition permet de rester dans la boucle while ?

Quel est l'état des deux LED tant que le bouton n'est pas appuyé ?

A quel moment les deux dernières instructions sont-elles exécutées ?

La boucle while **est**-elle exécutée une seule fois ou plusieurs fois ?

Boucle for : répéter un nombre connu de fois

```
for (int compteur = 0; compteur < 4; compteur++)
{
    digitalWrite(LED_JAUNE, HIGH);
    delay(250);
    digitalWrite(LED_JAUNE, LOW);
    delay(250);
}
```

Passage	Valeur du compteur	Test compteur < 4	Bloc exécuté ?
1			
2			
3			
4			
Test suivant			

Combien de clignotement sont réalisés ?

Quelle est la durée totale de la séquence ?

Compléter :

while est utile lorsque le nombre de répétitions est _____

for est utile lorsque le nombre de répétition est _____

G. Partie 4 – structure switch … case

Le programme doit allumer une LED différente selon la valeur de la variable mode

```
int mode = 2;

switch (mode)
{
    case 0:
        digitalWrite(LED_ROUGE, HIGH);
        break;

    case 1:
        digitalWrite(LED_VERTE, HIGH);
        break;

    case 2:
        digitalWrite(LED_JAUNE, HIGH);
        break;

    default:
        eteindreToutesLesLED();
        break;
}
```

<u>Analyse du programme</u>

Quelle variable est testée par la structure switch ?

Quel bloc est exécutée pour la valeur actuelle de mode ?

Quelle LED est allumée ?

Quel est le rôle de l'instruction break ?

Dans quel situation le bloc default est-il exécuté ?

Etude plusieurs valeurs

Valeur de mode	Bloc exécutée	Action réalisée
0		
1		
2		
4		
-1		

Comparaison

Dans quel cas switch...case peut-il être plus lisible qu'une longue suite de if...else if ?

Que pourrait-il se passer si l'instruction break de case 1 était supprimé ?

H. Partie 5 – Comprendre les fonctions

Sans argument et sans valeur de retour

```
void eteindreToutesLesLED()
{
    digitalWrite(LED_ROUGE, LOW);
    digitalWrite(LED_VERTE, LOW);
    digitalWrite(LED_JAUNE, LOW);
}

eteindreToutesLesLED();
```

Quel est le nom de la fonction ?

Reçoit-elle un argument ? **Renvoie**-t-elle une valeur ?

Avec argument et sans valeur de retour

```
void commanderLED(byte broche, bool etat)
{
    digitalWrite(broche, etat);
}

commanderLED(LED_VERTE, HIGH);
```

Quels sont les deux arguments déclarés ?

Quelles valeurs sont transmises lors de l'appel de la fonction ?

Sans argument et avec valeur de retour

```
int lirePotentiometre()
{
    int mesure = analogRead(A0);
    return mesure;
}

int valeur = lirePotentiometre();
```

Quel est le type de la valeur renvoyée ?

Quelle variable récupère la valeur renvoyée ?

Avec argument et avec valeur de retour

```
bool boutonAppuye(byte broche)
{
    if (digitalRead(broche) == LOW)
    {
        return true;
    }
    else
    {
        return false;
    }
}

bool marche = boutonAppuye(BP_MARCHE);
```

Quel argument est transmis à la fonction ?

Quelles sont les deux valeurs que cette fonction peut renvoyer ?

I. Partie 5 – Synthèse sur les fonctions

Fonction	Argument(s) ?	Type de retour	Rôle
void eteindreToutesLesLED()			
void commanderLED(byte broche, bool etat)			
float lirePotentiometre()			
bool boutonAppuye(byte broche)			

Associer chaque terme à sa définition

Terme	Numéro de la bonne définition
Nom de la fonction	
Argument	
Appel de la fonction	
Type de retour	
return	

Définitions :

1 - Instruction qui termine la fonction en renvoyant une valeur.

2 - Information fournie à une fonction lors de son utilisation.

3 - Instruction qui déclenche l'exécution d'une fonction.

4 - Identifiant utilisé pour désigner la fonction.

5 - Nature de la valeur produite par la fonction.

Suivre les appels

```
1  commanderLED(LED_ROUGE, HIGH);
2  int niveau = lirePotentiometre();
3  bool arret = boutonAppuye(BP_ARRET);
4  eteindreToutesLesLED();
```

Ligne	Fonction appelée	Information transmise	Valeur récupérée ?
1			
2			
3			
4			

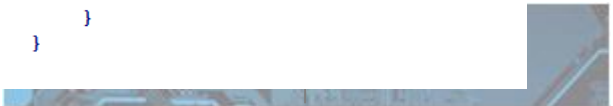
Pourquoi est-il intéressant de regrouper plusieurs instructions dans une fonction ?

Une fonction de type void **peut**-elle être utilisée à droite du signe = ? **Justifier**.

J. Partie 6 – Analyse d'un programme complet

Le programme suivant pilote 3 modes de fonctionnement. Vous devez comprendre son fonctionnement.

```
const byte BP_MODE = 2;
const byte LED_ROUGE = 8;
const byte LED_VERTE = 9;
const byte LED_JAUNE = 10;

byte mode = 0;

void eteindreToutesLesLED()
{
    digitalWrite(LED_ROUGE, LOW);
    digitalWrite(LED_VERTE, LOW);
    digitalWrite(LED_JAUNE, LOW);
}

bool boutonAppuye(byte broche)
{
    return digitalRead(broche) == LOW;
}

void afficherMode(byte numeroMode)
{
    eteindreToutesLesLED();

    switch (numeroMode)
    {
        case 0:
            digitalWrite(LED_ROUGE, HIGH);
            break;
        case 1:
            digitalWrite(LED_VERTE, HIGH);
            break;
        case 2:
            digitalWrite(LED_JAUNE, HIGH);
            break;
    }
}

void setup()
{
    pinMode(BP_MODE, INPUT_PULLUP);
    pinMode(LED_ROUGE, OUTPUT);
    pinMode(LED_VERTE, OUTPUT);
    pinMode(LED_JAUNE, OUTPUT);

    for (int i = 0; i < 2; i++)
    {
        digitalWrite(LED_ROUGE, HIGH);
        digitalWrite(LED_VERTE, HIGH);
        digitalWrite(LED_JAUNE, HIGH);
        delay(300);
        eteindreToutesLesLED();
        delay(300);
    }

    afficherMode(mode);
}

void loop()
{
    if (boutonAppuye(BP_MODE))
    {
        mode = mode + 1;

        if (mode > 2)
        {
            mode = 0;
        }

        afficherMode(mode);

        while (boutonAppuye(BP_MODE))
        {
            // Attente du relâchement
        }
    }
}
```



Question d'analyse

Identifier l'entrée numérique du système.

Identifier les 3 sorties numériques.

Quelle est la valeur initiale de la variable mode ?

Quelles fonctions ne possèdent aucun argument ?

Quelle fonction possède un argument et renvoie une valeur ?

Combien de fois les 3 LEDs clignotent-elles pendant setup() ?

Quelle fonction est appelée à la fin de setup() ?

Quelle LED reste allumée après l'initialisation ?

Quelle condition déclenche le changement de mode dans loop() ?

Quelle est la nouvelle valeur de mode après le premier appui ?

Pourquoi le programme teste-t-il if (mode>2) ?

Quel est le rôle de la boucle while placée après afficherMode(mode) ?

Evaluation du mode

Etape	Valeur de mode	LED allumée	Justification courte
Après setup()			
Après le 1er appui			
Après le 2ème appui			
Après le 3ème appui			
Après le 4ème appui			
Après le 5ème appui			

Ordre d'exécution

Numéroter de 1 à 7 les actions suivantes dans l'ordre où elles se produisent après un appui sur le bouton Mode :

Ordre	Action
	Le programme entre dans la fonction afficherMode(mode).
	Le programme détecte que le bouton est appuyé.
	La valeur de mode est augmentée.
	Les trois LED sont d'abord éteintes.
	La structure switch sélectionne une LED.
	Le programme attend le relâchement du bouton.
	Le programme revient au début de loop().

Bilan

Résumer en quatre ou cinq phrases le fonctionnement global du programme.

Citer deux avantages de l'utilisation de fonctions dans ce programme.

K. Livrable

Enregistrer et **renommer** le fichier « *NOM*-LireComprendreProgrammeArduino.pdf » avec *NOM* votre nom de famille.

Téléverser le fichier pdf.